SMC 2009
http://smc2009.smcnetwork.org

6th Sound and Music Computing Conference

SMC 2009
23-25 July 2009
Casa da Música
Porto, Portugal

Proceedings of the 6th Sound and Music Computing Conference

Fabien Gouyon, Álvaro Barbosa, Xavier Serra (eds)

organizers:

INESCPORTO LABORATÓRIO ASSOCIADO
ESMAE POLITÉCNICO DO PORTO
CITAR
casa da música
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO
DECC DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Fabien Gouyon, Álvaro Barbosa, Xavier Serra (Editors)

SMC 2009

Proceedings of the 6th Sound and Music Computing Conference

23-25 July 2009 Casa da Música, Porto - Portugal

Organisers:

INESCPORTO® LABORATÓRIO ASSOCIADO

CITAR

ESMAE POLITÉCNICO DO PORTO

casa da música

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

DECC DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Sponsors:

CATÓLICA UNIVERSIDADE CATÓLICA PORTUGUESA | PORTO Escola das Artes

INSTITUTO POLITÉCNICO DO PORTO

UT Austin | Portugal INTERNATIONAL COLLABORATORY FOR EMERGING TECHNOLOGIES, CoLab

cost

Sonic Interaction Design SID

FCT Fundação para a Ciência e a Tecnologia MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

U.PORTO

Passos Manuel

Metro do Porto

bmat music innovation

ISBN: 978-989-95577-6-5

Dear fellow researchers,

We are very proud to present you the book of proceedings of the SMC 2009 scientific program. We were able to put together this program thanks to the numerous contributions of the scientific community in response to our call for participation. The biggest thanks therefore goes to you, reading this book of proceedings.

When we started to organise this year's conference we were not sure of the appeal that this conference would have for the research community. The Sound and Music Computing conference is a very young event that was born with a somewhat local perspective and that competes with a number of consolidated conferences. But the large number of submissions in all the different categories and the involvement of the research community in the peer review process of the submissions allowed us to organise a conference that we are proud of. We sincerely hope that these proceedings will be of interest for all of you.

In organising the scientific program we had two objectives in mind: To push for a high quality technical program and promote the involvement of young researchers; The best recipe for quality requires first to get many submissions and then to organise a good peer review process of the submissions.

This year we received 160 paper and poster submissions, the largest number to date. The reviews have been coordinated by the Scientific program Chairs and the General Chair, and performed by a Scientific Committee of 70+ key SMC researchers, specialists in all the topics of the conference, from most of the main research centers in the field.

From the very outset, we decided not to have parallel sessions, at the risk of having to accept fewer contributions. After the review process we accepted 26 papers and 37 posters, which represents a 39% acceptance rate. We asked reviewers to provide as much feedback as possible to the authors. We acknowledge that this is traditionally quite hard in our community and that there is certainly room for improvement in the review process. In our particular case we are aware that not all the authors received all the feedback that we had wished for. They should however rest assured that the utmost care has been given to all submissions and that decisions of rejection were based on a number of different factors, including time restrictions and program coherence.

Another aspiration was to promote the involvement of young researchers. In order to do that we have joined forces with the SMC Summer School which, in the past, has been organised independently of the conference. The School takes place just before the conference and from the 54 applications 21 students were selected. During the Summer School, these young researchers will get training in a specific area of the Sound and Music Computing field and also feedback on their particular research projects. Students will attend lectures by three recognised researchers from our field and 7 tutors will work closely with the students on several practical projects.

An additional track of events aiming at the practical involvement of researchers, especially the young ones, lies in the three tutorials held on the day prior to the beginning of the conference. We received 7 proposals for tutorials, of which 3 have been selected, offering a varied spectrum of topics to the audience.

Apart from paper and poster presentations that focus on specific research topics, we wanted to have presentations that would give the participants a broader view on our field and that would inspire new research directions. This was our main aim in selecting the three keynote addressed by the following highly-respected personalities: José Carlos Principe, Atau Tanaka and Bruce Pennycook Broad coverage and insightfulness were also what drove us to organise the four "inspirational sessions". In these sessions, a special focus will be put on interaction between participants. They have been designed as a venue for preliminary, frontier-research ideas (musical, technical, scientific, theoretical, practical, etc.), where no implementation, proofs, results nor evaluations will be required. Just great, inspirational ideas. We very much hope they will give the opportunity to present and discuss relevant topics of our field in a different context and in a very different way.

It has been a hard work to put together the scientific program of SMC 2009. But we had fun doing it, we learned a lot from it and we wish, and are very much confident, that conference attendants will come out from the conference having learned something new and with new ideas for their research activities.

Fabien Gouyon
Álvaro Barbosa
Xavier Serra

SMC 2009 Committee

General Chair
Fabien Gouyon (INESC Porto, Porto, Portugal)

Scientific Programme Chairs
Álvaro Barbosa (CITAR, Universidade Católica Portuguesa, Porto, Portugal)
Xavier Serra (MTG, Universitat Pompeu Fabra, Barcelona, Spain)

Music Programme Chairs
Carlos Guedes (ESMAE, Porto, Portugal)
Pedro Rebelo (SARC, Queen's University, Belfast, UK)

Conference Organization Committee
Fabien Gouyon (INESC Porto, Porto, Portugal)
Álvaro Barbosa (CITAR, Universidade Católica Portuguesa, Porto, Portugal)
Carlos Guedes (ESMAE, Porto, Portugal)
Paulo Rodrigues (Casa da Música, Porto, Portugal)
Artur Pimenta Alves (Department of Electrical and Computer Engineering of the Faculty of Engineering of the University of Porto, Portugal)

Music Programme Curators
Evan Parker
Nicolas Collins
Pauline Oliveros
Robert Rowe

Scientific Committee
Christina Anagnostopoulou (University of Athens, Greece)
Daniel Arfib (Laboratoire d'Informatique de Grenoble, France)
Gérard Assayag (IRCAM, Paris, France)
Jean-Julien Aucouturier (Ikegami Laboratory, Tokyo, Japan)
Stephan Baumann (Competence Center Computational Culture, Kaiserslautern, Germany)
Juan Bello (New York University, USA)
Nicola Bernardini (Conservatorio C.Pollini, Padova, Italy)
Emmanuel Bigand (Laboratoire d'Etude de l'Apprentissage et du Développement, Dijon, France)
Roberto Bresin (Royal Institute of Technology, Stockholm, Sweden)
Emilios Cambouropoulos (Aristotle University of Thessaloniki, Greece)
Antonio Camurri (InfoMus Lab, Genova, Italy)
Oscar Celma (Barcelona Music and Audio Technologies, Spain)
Joel Chadabe (Electronic Music Foundation, New York, USA)
Chris Chafe (CCRMA, Stanford, USA)
Roger Dannenberg (Carnegie Mellon University, Pittsburgh, USA)
Laurent Daudet (LAM, Paris, France)
Myriam Desainte-Catherine (LaBRI, University of Bordeaux, France)
Simon Dixon (Queen Mary, University of London, UK)
Gerhard Eckel (IEM, Graz, Austria)
Cumhur Erkut (Helsinki University of Technology, Finland)
Mikael Fernstrom (University of Limerick, Ireland)
Aníbal Ferreira (FEUP, Porto, Portugal)
Paulo Ferreira-Lopes (CITAR, Porto, Portugal)
Arthur Flexer (Austrian Research Institute for Artificial Intelligence, Wien, Austria)
Dominique Fober (GRAME, Lyon, France)
Karmen Franinovic (Zurich University of the Arts, Switzerland)
Gunter Geiger (MTG, Barcelona, Spain)
Anastasia Georgaki (University of Athens, Greece)
Rolf Inge Godoy (University of Oslo, Norway)
Werner Goebl (Johannes Kepler University, Linz, Austria)
Emilia Gómez (MTG, Barcelona, Spain)
Masataka Goto (National Institute of Advanced Industrial Science and Technology, Tsukuba, Japan)
Holger Grossmann (Fraunhofer Institute for Digital Media Technology, Ilmenau, Germany)
Carlos Guedes (ESMAE, Porto, Portugal)
Thomas Henriques (Faculdade de Ciências Sociais e Humanas, Universidade Nova de Lisboa, Portugal)
Thomas Hermann (Bielefeld University, Germany)
Perfecto Herrera (Escola Superior de Música de Catalunya/MTG, Barcelona, Spain)
Martin Kaltenbrunner (MTG, Barcelona, Spain)
Anssi Klapuri (Tampere University of Technology, Tampere, Finland)
Alessandro Koerich (Pontifícia Universidade Católica do Paraná, Curitiba, Brazil)
Olivier Lartillot (University of Jyväskylä, Finland)
Fernando Lopez-Lezcano (CCRMA, Stanford University, USA)

SMC is a privileged forum in Europe for the promotion of international exchanges around Sound and Music Computing.

The SMC initiative is jointly supervised by the following European associations:
AFIM (Association Française d'Informatique Musicale)
AIMI (Associazione Italiana di Informatica Musicale)
DEGEM (Deutsche Gesellschaft für Elektroakustische Musik)
HACI (Hellenic Association of Music Informatics)
IAMT (Iberian Association for Music Technology)

Table of Contents

Full Papers – Oral presentations

Poster Presentations

THE FLOPS GLASS: A DEVICE TO STUDY EMOTIONAL REACTIONS ARISING FROM SONIC INTERACTIONS

G. Lemaitre
IRCAM,
Paris, France
lemaitre@ircam.fr

O. Houix
IRCAM,
Paris, France

K. Franinović
Zürcher Hoschule
der Künste, Zürich,
Switzerland

Y. Visell
McGill
University
Montréal, Canada

P. Susini
IRCAM,
Paris, France

ABSTRACT

This article reports on an experimental study of emotional reactions felt by users manipulating an interactive object augmented with sounds: the *Flops* glass. The Flops interface consists of a glass embedded with sensors, which produces impact sounds after it is tilted, implementing the metaphor of the falling of objects out of the glass. The sonic and behavioural design of the glass was conceived specifically for the purpose of studying emotional reactions in sonic interactions. This study is the first of a series. It aims at testing the assumption that emotional reactions are influenced by three parameters of the sounds: spectral centroid, tonality and naturalness. The experimental results reported here confirm the significant influence of perceptual centroid and naturalness, but fail to show an effect of the tonality.

1 INTRODUCTION

New technologies make it possible for designers to consider sonic augmentations of a wide array of everyday objects that incorporate electronic sensing and computational capabilities. Sound-mediated interactions raise several interesting issues related to the functionality and the aesthetic of a design [8]. Another interesting question is that of the emotional reactions induced by such sonically augmented interactions. This question is of further importance when considering that a user's preferences and other evaluations of a product are influenced by her emotional reactions when using this product [9].

To adress this issue, an interactive object was designed, called the *Flops*. It is a glass embedded with a tilt sensor allowing it to control the generation of impact sounds when tilted. It implements the metaphor of a glass full of virtual items that may be poured out of it. The sounds can be easily modified, in order to assess the influence of the sound parameters on the emotional reactions of the users.

In the experiment reported upon here, 25 participants were required to watch a set of videos displaying a user pouring virtual items out of the Flops glass. They had to report their emotional reactions by providing judgements on three scales: valence, arousal and dominance. The images in the videos were all the same, and only the sounds changed. The sounds were created on the basis of conclusions of other experimental studies that have suggested that the affective reactions to aircraft noises were influenced by several aspects of the sounds: sharpness, tonality and naturalness [20]. The goal of our study is to explore whether these conclusions are valid for the sounds used in the Flops glass, which consisted of very short impact sounds sequenced in various temporal patterns. This study is intended to be the first of a series. It investigates how the sounds only might influence the emotional reactions of the users.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

2 EMOTIONAL REACTIONS TO SOUNDS

Emotions Emotions have been studied by philosophers and scientists for centuries. Yet the question of *what* are emotions is still a matter of debate. Most modern emotion theorists have adopted a componential approach to emotions, suggesting that an *emotion episode* consists of coordinated changes in several components: physiological arousal, motor expression, subjective feelings, behavior preparation, cognitive processes [13, 14, 15, 5]. *Feelings* (or *core affects* [13]) are considered as the conscious reflection of changes occurring in these components. There exists several approaches for the assessment of emotions, the most widespread being the physiological measures (heart rate, skin conductance, facial EMG, startle reflex, etc.) [2], the *basic emotions* approach [4] and the *dimensional* approach, on which we choose to focus in this study. From more than 50 years indeed, studies have suggested that the emotional reactions observed in, or reported by subjects can be accounted by a two- or three-dimensional framework [16]. For instance, Osgood suggested the following dimensions as primary referents of facial expressions of emotions: pleasantness, control, and activation [10]. This approach has been formalized in the circumplex model of affects proposed by Russell [12]. Three dimensions are generally considered: the *va-*

lence dimension, describing unpleasant to pleasant feelings, the *arousal* dimension, describing the degree of arousal (from calm to excited) felt by the subject, and the *dominance* dimension, describing how dominated or dominant feel the subjects.

Emotional reactions in sound design Assessing emotional reactions of a user using a product is of major importance for designers [18]. It has particularly been reported that attractive products are perceived easier to use [9]. Emotional reactions to the sounds of everyday products have been primary studied in terms of unpleasantness or annoyance (see for instance [6]), or preference (see for instance [19]). Specifically, Västfjäll et al. [20] have found significant correlations between valence and arousal ratings and several psychoacoustical descriptors of aircraft sounds: they found valence to be correlated with loudness and naturalness (naturalness was rated by listeners), and activation with sharpness and tonal content.

3 INTERACTION AND SOUND DESIGNS OF THE FLOPS GLASS

The Flops glass is an interface similar to a glass containing virtual objects. When tilted, virtual objects drop out of the Flops glass, producing impact sounds when hitting the surface above the glass. The sounds were created in order to test the results reported above: they were made so as to vary along their spectral centroids (similar to the sharpness descriptor used in [20]), and tonality indexes. Natural and synthetic sounds were used.

3.1 Physical design

The physical interface of the Flops glass is shown in Figure 1. Its shell is modeled in 3D software and is extruded in ABS plastic. The interface contains an accelerometer (Analog Devices ADXL 320 3-axis MEMS accelerometer), sensing the gesture performed with the glass. The sensor is wired to an Arduino BT board, sending the tilting data through Bluetooth connection to a remote computer. The sensor data processing and the playback of the sounds are real time processed in Cycling'74 Max/MSP 5.0.6.

3.2 Interaction design

The interaction model, transforming tilt angle to a flow of objects falling, is based on the model of items sliding without friction on a tilted rod, and falling when reaching the extremity of the rod (see Figure 2). A reservoir of virtual items is situated aft inside the Flops glass (at a distant d of the mouth of the Flops glass), where virtual items are stored (regularly separated by a distance d_0). Assuming no friction, when the Flops glass is tilted with a constant angle α,

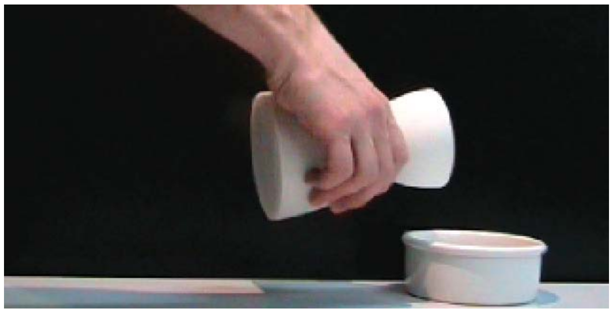


Figure 1. A video showing a user using the Flops glass.

the position of each item is:

$$x_n(t) = -nd_0 + k\sin(\alpha)t^2 \tag{1}$$

where k is a constant. Assuming that $d \gg d_0$, the time between two successive dropped items is $\Delta t = K/\sqrt{\sin\alpha}$, where K is a constant. The rate of impacts is therefore constant for α constant.

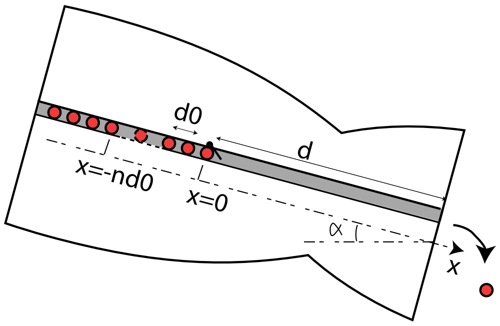


Figure 2. Model used for the interaction.

3.3 Sound design

The model described above is used to drive the generation of impact sounds: when the Flops glass is tilted, a series of impact cues is generated, with a rate computed as described above. Each cue triggers the playback of a sample of an impact sounds.

Thirty-two samples were created. Sixteen sounds ("natural sounds") were created by recording different impact sounds, from collision of everyday objects to musical percussions. Sixteen ("synthetic sounds") were samples of sounds synthesized by various kinds of algorithms (mostly additive and subtractive synthesis). The creation and the selection of these sounds were made such as homogeneously sampling across two psychoacoustical descriptors: spectral centroid and tonality index (computed according the IrcamDescriptor toolbox [11]). The spectral centroid of the sounds vary from 410 Hz to 1890 Hz, and the tonality of the sounds vary from 0.07 to 0.96 (the index of tonality can theoretically vary from 0 to 1, with 0 corresponding to a white noise, and 1 to a pure tone). All the sounds were created to have a rather short attack time (from 44 ms to 90 ms). All samples have the same duration and last approximatively 350 ms.

4 EXPERIMENTAL STUDY

The experimental study reported here aims at testing the assumption that the parameters used to create the Flops glass sounds (spectral centroid, tonality, naturalness) influence the emotional reactions of participants watching a set of videos displaying a user manipulating the Flops.

4.1 Method

Participants Twenty-five participants (14 women and 11 men) volunteered as listeners and were paid for their participation. They were aged from 19 to 45 years old (median: 28 years old). They were selected on the basis of the Spielberger trait anxiety inventory [3]. They had to have a score lower than 39 (indicated low trait anxiety).

Stimuli Thirty-two videos were generated, corresponding to the 32 sounds described above. They showed a user manipulating the Flops glass (see Figure 1). All the videos were the same, except for the sounds. Each video was 8 s long. All the soundtracks had been equalized for loudness in a preliminary study (for we are not interested in this parameter). The levels of the sounds varied from 52 dB(A) to 79.9 dB(A) (median 71.5 dB(A)). The video showed a user tilting three times the Flops glass: first, he drops slowly three items out of the glass, then tilts the Flops glass more quickly to increase the rate of times dropping out of the Flops glass. A total of 28 items are dropped.

Apparatus The stimuli were amplified over a pair Yamaha MSP5 loudspeakers. Participants were seated in a double-walled IAC sound-isolation booth. The experiment was run using the PsiExp v3.4 experimentation environment including stimulus control, data recording, and graphical user interface [17]. The sounds were played with Cycling'74 Max/MSP version 5.0.6, with Jitter displaying the videos. The scales were presented on an Elo Touch Screen. This allowed the participants to interact with the interface by only touching the screen.

Procedure The participants were first presented with a text explaining the procedure, and explaining the meaning of the 3 scales of valence, arousal and dominance. Then they were

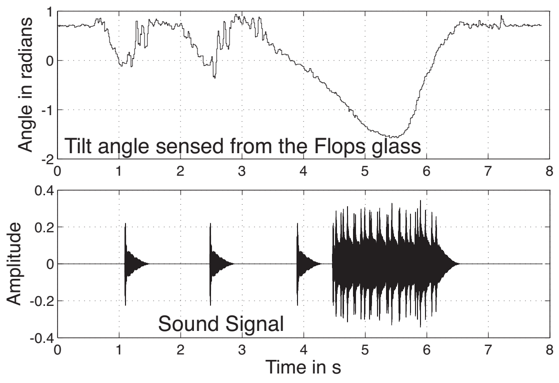


Figure 3. Example of the soundtrack of one of the videos used in the experiment, together with the tilt angle of the Flops glass.

presented with a selection of pictures from the IAPS set of images [7], and required to report their emotional reactions. Then, they were presented with all the videos played one after the other. Finally, they watched again each video, and had to report for each video how their emotional reactions.

The participants had to indicate their emotional reactions by selecting an item on each of the three 9-point scales, using the Self-Assessment Manikins (SAM) [1]. The SAM is non-verbal pictorial assessment technique that directly measures the pleasure, arousal, and dominance associated with a person's affective reaction. The SAM scales used in this study are reported on Figure 4.

4.2 Analysis

For the 32 sounds, the standard deviation of the judgements made by the participants varies from 1.27 to 2.2 for the valence scale, from 1.38 to 2.13 for the arousal scale, and from 1.36 to 1.91 for the dominance scale, which is consistant with the data gathered for the IAPS image sets [7]. This indicates that the emotional reactions caused by the videos tend to be rather consistent across the participants. It also indicates that the videos have elicited emotional reactions in the subjects. In the following, the judgements will therefore be averaged across participants.

The distributions of the judgements averaged over the 25 participants are represented on Figure 5 for the three scales, and for the 2 groups of sounds. The judgements on the valence scale vary from 2.08 to 5.8 (the range of the judgements is therefore 3.1 on a scale of 9) with an average of 4.39, indicating that the participants have mainly used the center of the scale for all the videos. The judgements are rather concentrated, and skipped toward the "unpleasant part" of the scale. When considering separately the natural and synthetic sounds, it appears that the natural sounds

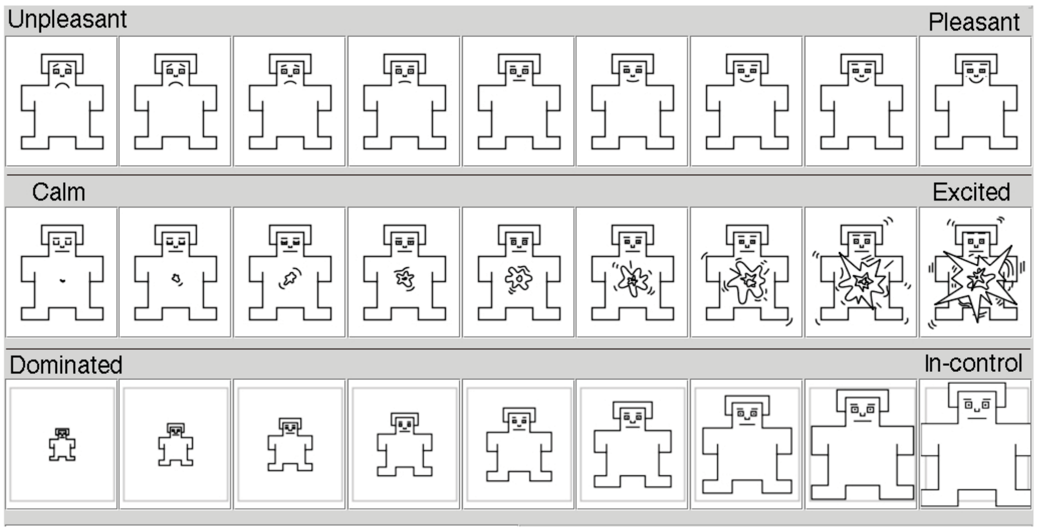


Figure 4. Interface used to report the emotional reactions. The SAM is non-verbal pictorial assessment technique that directly measures the pleasure, arousal, and dominance associated with a person's affective reaction.

have caused a rather neutral judgement on the valence scale among the participants (average = 4.9), whereas the synthetic sounds have caused a slightly more unpleasant feeling in the participants (average = 3.9). A Student t-test indeed reveals that the averages of these two distributions of judgements are significantly different ($t(30)$=3.81, $p < 0.01$).

The judgements of arousal are also concentrated on the middle of the scale. Across all the 32 videos, the judgements of arousal vary from 4 to 7.04 (the range of the judgements is therefore 3.0 on a scale of 9), with an average of 5.06: the participants have not used the whole range of the scale. The averages of the distributions of judgements for the two sets of 16 sounds are not statistically significant: $t(30)$=-0.93, p=0.82. Overall, the sounds have caused a rather medium arousal in the participants.

The judgements of dominance vary from 3.96 to 6.6 (the range of the judgements is therefore 2.2 on a scale of 9) for all the 32 sounds, with an average value of 5.60. For this scale also, the participants have used a narrow range of the scale, slightly skipped toward the "in control" part of the scale. The averages of the distributions of judgements for the two sets of 16 sounds (natural vs. synthetic) are significantly different ($t(30)$=3.80, p <0.01). The average dominance judgement is 5.91 for the natural sounds, and 5.23 for the synthetic sounds. The participants have therefore felt slightly more in control when watching the videos with the natural sounds than the videos with the synthetic sounds, yet this difference is small.

The judgements on three scales are significantly correlated: Valence vs. Arousal, $r(30)$=-0.78, p <0.01 Valence vs. Dominance $r(30)$=0.92, , p <0.01, Arousal vs. Dominance, $r(30)$=-0.76, p <0.01. This indicates that the participants have not used the three scales independently. There are systematic patterns in the judgements: the videos causing emotional reactions judged as pleasant tended to cause at the same time calm and dominant feelings, whereas the

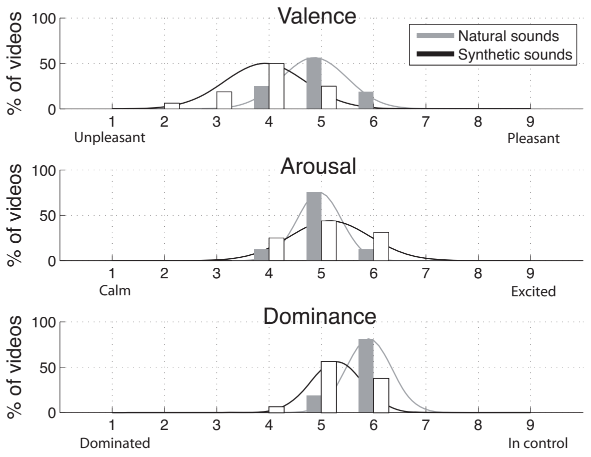


Figure 5. Bar plots of the distributions of the judgements on the three scales (valence, arousal, dominance), averaged across the participants, for the two groups of 16 sounds (natural vs. synthetic). A Gaussian curve with the same average and standard deviation as the distributions of judgements is also represented on top of each bar plot.

videos causing emotional reactions judged as unpleasant tended to cause systematically feelings judged as excited and dominated. This was confirmed by the informal post experimental interviews with the participants: many participants had indeed spontaneously indicated that "shrill" sounds tended to irritate them (i.e. unpleasant, excited and dominated judgements), whereas "soft and low" sounds tended to be felt as more relaxing (pleasant, calm, and in-control).

These observations are further confirmed when studying the correlations between the judgements on the scales and several acoustic descriptors. We tested not only the spectral centroid measure and tonality index (which were used to create and select the sounds), but also all the descriptors contained in the IrcamDescriptor toolbox [11]. Overall, the judgements on the three scales are correlated with the many variants of the spectral centroid, the best correlations being obtained by the "perceptual" spectral centroid. This descriptor is computed as the centroid computed using the specific loudness of the Bark scale [11]. The correlations with the three scales is statistically significant: Valence, $r(30)$=-0.63, p <0.01, Arousal, $r(30)$=0.64, p <0.01, Dominance, $r(30)$=-0.48, p <0.01. Interestingly, these correlations are different for the two groups of sounds: whereas the correlations are very good for the natural sounds (Valence: $r(30)$=-0.83; Arousal: $r(30)$=0.81; Dominance: $r(30)$=-0.72; each: p <0.01), the judgements are much more spread for the synthetic sounds (Valence: $r(30)$=-0.64; Arousal: $r(30)$=0.57; p <0.01; Dominance: $r(30)$=-0.43; p <0.05).

4.3 Discussion

The emotional reactions to the 32 videos extend over a rather small portion of the valence-arousal-dominance space, and are centered around the neutral positions of each scale. This is not really surprising, because these sounds only vary along basic acoustical properties. It could therefore not be expected that a set of videos displaying a user dropping virtual items out of a glass would cause emotional reactions comparable with those caused by sounds or images with a strong semantic content (e.g. violent images, etc.).

More problematic however are the correlations of the three scales. It is obvious from the experimental results that we have not succeeded in creating sounds that cause emotions varying independently along the three dimensions of emotions used here. On the contrary, systematic patterns of judgements appear in the judgements: videos causing pleasant emotions caused at the same time dominant reactions, and conversely. This is further confirmed when considering that no scale is correlated with any metric of ity. Indeed, following the results of Västfjäll et al. [20], the sounds were created along three aspects: naturalness, spectral centroid, and tonality. This last parameter was assumed to be correlated to the arousal judgements, which is not the case here. A possible explanation is that tonality is probably a relevant parameter for long and continuous sounds such as aircraft noise, but not for short impact sounds. Note that other systematic patterns of variations in the arousal valence space (i.e. "boomerang-shaped") have also been reported for other acoustic stimuli [2].

The variations of spectral centroids of the sounds are fairly correlated with the judgements on the three scales. The two groups of sounds (natural vs. synthetic) have produced slightly different emotional reactions: natural sounds are judged as causing more pleasant and more dominant feelings than synthetic sounds. These two aspects of the sounds are therefore here the predictors of the emotional reactions.

5 CONCLUSION

This article is the first in a series aiming at studying users' emotional reactions when manipulating sound-mediated interactive objects. A experimental object (the Flops glass) was designed. It is a plastic glass capable of capturing the extent to which it is tilted. The object implements the metaphor of virtual items stored in it that are dropped when it is tilted. Each item virtually dropped out of the Flops glass produces sound when impacting the surface below.

The study reported here aimed at assessing the influence of the sounds solely on users' emotional reactions. Thirty-two sounds were created, with the purpose of testing the validity of the conclusions found in [20] for aircraft sounds, when extended to the case of the impact sounds used in the Flops glass. The sounds were therefore created so as to vary along two acoustical parameters: spectral centroid, and tonality index. Two kinds of sounds were used: records of "natural" impacts, and samples of sounds synthesized by various additive-substractive algorithms. The three parameters investigated had previously been found to influence the emotional reactions of participants listening to the aircraft sounds.

In the experimental study conducted here, 25 participants watched videos of a user manipulating the Flops glass. These videos were all the same, except for the sounds. They had to report their emotional reactions on three scales: valence, arousal and dominance. They used the Self-Assessment Manikin proposed in [1]. The results show that the two types of sounds influenced the emotional reactions: natural sounds were found to be slightly more pleasant, and they caused participants to feel more in control than synthetic sounds. These conclusions are consistent with those found in [20]. However, no scale appeared to be correlated with any descriptor related to the tonality of the sounds. Furthermore, the judgements on the three scales were correlated, indicating that the sounds caused emotional reactions that varied along a single axis: from pleasant, calm, and in-control feelings, to unpleasant, exciting and dominated feelings. This suggests a single positive-negative dimension. However, it can be noted that whereas natural sounds caused more pleasant feelings than synthetic sounds, they did not cause calmer feelings, indicating that the participants were able to distinguish the three scales.

These conclusions offer interesting results for followup studies on emotional reactions to sound-mediated interactive objects. They allow, for instance, to select sounds causing negative, neutral or positive reactions. The next study that is planned will address the influence of the usability of the interface on users' emotional reactions. A interesting related issue is the interaction between emotional reactions caused by the sounds, and those caused by the usability.

Acknowledgments

This work was funded by the EU project *CLOSED* FP6-NEST-PATH no. 29085. The glass was firstly designed as a part of a Flops installation where three glasses were used. The development of this project was also supported by the Interstices Research Group. The glass was produced by Karmen Franinovic and Yon Visell, with the support of Martin Peach, Fabienne Meyer and Thomas Tobler.

6 REFERENCES

[1] Bradley M. M. and Lang P. J., "Measuring emotion: the Self-Assessment Manikin and the semantic differential", *Journal of behavior Therapy and Experimental Psychiatry*, volume 25(1), pp. 49-59, 1994.

[2] Bradley M. M. and Lang P. J., "Affective reactions to acoustic stimuli", *Psychophysiology*, vol. 37(1), pp. 204-215, 2000

[3] Cattell R. B. and Scheier I.H. "The meaning and measurement of neuroticism and anxiety", *Ronald*, NewYork, 1961.

[4] Ekman P., "Are there basic emotions?", *Psychological review*, vol. 99(3), p. 550-553, 1992, Special issue 2001-2002

[5] Juslin P. N. and Västfjäll D., "Emotional responses to music: the need to consider underlying mechanisms", *Behavioral and Brain Sciences*, vol. 31(1), pp. 559-621, 2008

[6] Kumar S., Forster H. M., Bailey P. and Griffiths T. D., "Mapping unpleasantness of sounds to their auditory representation, *Journal of the Acoustical Society of America*, vol. 124(6), pp. 3810-3817, 2008

[7] Lang, P. J., Bradley, M. M., and Cuthbert, B. N. "International affective picture system (IAPS): Affective ratings of pictures and instruction manual", Technical Report A-7, *University of Florida, Gainesville, FL*, 2008.

[8] Lemaitre G., Houix O., Visell Y., Franinovic K., Misdariis N. and Susini P., "Toward the Design and Evaluation of Continuous Sound in Tangible Interfaces: The Spinotron", *International Journal of Human-Computer Studies, special issue on Sonic Interaction Design*, 2009

[9] Norman D. A., "Emotional Design - Why we love (or hate) everyday things", *Basic Books*, New York, 2004

[10] Osgood C. E. "Dimensionality of the semantic space for communication with spatial expressions" *Scandinavian Journal of Psychology* vol. 7, pp. 1-30, 1966

[11] Peeters G., "A large set of audio features for sound description (similarity and classification) in the CUIDADO project", Cuidado Projet report, *Institut de Recherche et de Coordination Acoustique Musique (IRCAM)*, 2004

[12] Russell J. A., "A circumplex model of affect", *Journal of personality and social psychology*, vol. 39(6), pp. 1161-1178, 1980

[13] Russell J. A., "Core affect and the psychological construction of emotion", *Psychological review*, vol. 110(1), pp. 145-172, 2003

[14] Scherer K. R. "Which emotions can be induced by music? What are the underlying: mechanisms? And how can we measure Them?", *Journal of new music research*, vol. 33(3), 2004

[15] Scherer K. R., "What are emotions? And how can they be measured?", *Social Science Information*, vol. 44(4), pp. 695-729, 2005

[16] Schlosberg H., "Three dimensions of emotion, *The psychological review*, vol. 61(2), p. 81-88, 1954

[17] Smith B. K., "PsiExp: an environment for psychoacoustic experimentation using the IRCAM musical workstation", *Society for Music Perception and Cognition Conference '95*, 1995, University of Berkeley, CA

[18] Västfjäll D. and Kleiner M., "Emotion in Product Sound Design, Proceedings of "Les journées du design sonore", Paris, France, March 2002

[19] Västfjäll D., Gulbol M. A., Kleiner M. and Gärling T., "Affective evaluations of and reactions to exterior and interior vehicle auditory quality", *Journal of Sound and Vibration*,vol. 255(3), pp. 501-518, 2002

[20] Västfjäll D., Kleiner M. and Gärling T., "Affective reactions to Interior Aicraft Sounds, *Acta Acustica united with Acustica* vol. 89, pages 693-701, 2003

EMPIRICALLY BASED AUDITORY DISPLAY DESIGN

Eoin Brazil, Mikael Fernström
Interaction Design Centre
University of Limerick
Limerick, Ireland
eoin.brazil@ul.ie,mikael.fernstrom@ul.ie

ABSTRACT

This paper focuses on everyday sounds and in particular on sound description, sound understanding, sound synthesis/modelling and on sonic interaction design. The argument made in this paper is that the quantitative-analytical reductionist approach reduces a phenomenon into isolated individual parts which do not reflect the richness of the whole, as also noted by Widmer et al. [1]. As with music, so is it for everyday sounds that multidimensional approaches and techniques from various domains are required to address the complex interplay of the various facets in these types of sounds. An empirically inspired framework for sonic interaction design is proposed that incorporates methods and tools from perceptual studies, from auditory display theories, and from machine learning theories. The motivation for creating this framework is to provide designers with accessible methods and tools, to help them bridge the semantic gap between low-level perceptual studies and high-level semantically meaningful concepts. The framework is designed to be open and extendable to other types of sound such as music.

1 Introduction

There is a growing acknowledgement [1] that reductionistic approaches cannot reflect the rich variety within sound. However, it has yet to be systematically addressed in the SMC [1] community or within any of the related fields such as auditory display or sonic interaction design. This article synthesises and organises the existing research within these fields. It presents a discussion on the qualitative and quantitative research that led to the development of a foundation for a framework, its structure and components, and examples of its application towards a practical empirically based design framework. This framework uses multiple approaches to capture different aspects of the sounds under exploration as a means of providing a better reflection of their richness.

There is no single methodological framework that can deal adequately with the complex socio-cultural context of auditory display design in a coherent and non-reductionist manner. This is a similar problem faced by most design oriented research, a suggestion by Melles [2] has been to take a pragmatic stance towards methodology, where methods are selected and combined according to their usefulness for achieving specific goals. This approach of design research has found support in many methodological dialogues such as those discussing multimethod research [3]. The framework presented is structured to support the selection of sounds while allowing the exploration of specific aspects of the sounds. Our approach suggests it is possible to gather the necessary information using complementary techniques [4].

A general observation from many auditory display designers is that auditory icons are not easy to design [5, 6]. This research has synthesised and organised the existing work to provide an empirically based auditory design process. The studies and methods explored provide indicative trends, which can assist designers in making the best selection and use of everyday sounds in their interface. In selecting these methods, preference was given to lighter weight approaches suitable for use outside strict laboratory conditions. This allows designers access these methods and the framework at an acceptable cost and without access to dedicated facilities such as listening booths or anechoic chambers. A number of additional criteria such as ease of use, prior similar use in the field or related fields, ability to concisely present the results, and time required to use the method were also considered.

The underlying rationale was to provide a similar type of approach to that of discount HCI as proposed by Nielsen [7]. Designers need empirically based or inspired methods to guide their overall design process, which do not suffer from the specificity of psychoacoustic studies or that require a relatively long time to conduct. A typical design problem is wider than those addressed by psychoacoustic studies and the approach of this framework joins these disciplines in a manner that is accessible at a reasonable cost to designers. The benefit from this type of approach should be a reduction in the ad-hoc selection of auditory icons and similar sounds [6].

[1] http://www.smcnetwork.org/

SMC 2009, July 23-25, Porto, Portugal

2 Existing Design Methodologies In Sonic Interaction Design

There are few design methodologies which are specifically situated within the field of Sonic Interaction Design (SID). Our framework was inspired by the work in the EU FET Closed project [8], by work on interactive public installations [9], and by work on narrative inspired interactive artefacts [10]. The first methodology was focused at the creation of *functional artefacts* and had close roots to industrial design and interaction design. The second methodology was targeted at *interactive public spaces* and came from an interaction design background that was complemented by empirical explorations. The third methodology came from an interaction design background with strong influences from film and game design to focus on creating narrative driven interactive artefacts. It aimed to create *narrative sound artefacts*. The three methodologies had different goals and understanding their origins can help in clarifying their distinct methodologies.

2.1 Designing Functional Artefacts

The EU FET Closed project [8] explored many aspects of sonic interaction design including the creation of functional artefacts as shown in Figure 1. It looked especially at kitchen sounds and how to integrate basic design practises together with interaction design to create functional artefacts. An example of this type of artefact is the *Spinotron* [11], which explored the link between sound objects and pumping actions. It used rolling and wheel/ratchet parameterised sound synthesis models [2] linked to real-time sensor data. The synthesis model design was based on the concept of a ratcheted wheel, where the motion or pumping of the device controlled the rotation of the wheel. The methodology promotes a complementary use of basic design methods to formalise and structure ethnographic approaches. The evaluation aspects in this methodology incorporate the ideas of material analysis and of interaction gestalts as shown in Figure 1. The goal of this methodology is to integrate these aspects to help products fit within their contexts of use by providing broader views of evaluation. This wider view includes holistic measures of experience and takes the functional performance of end users of the device or interface into account.

2.2 Designing Interactive Public Spaces

The Shared Worlds project [9] explored designing for interactive public spaces, in particular public transport spaces and market spaces. Shannon airport in County Clare, Ireland was the site of one of these interventions. An interactive portal was designed to allow travellers in the departures lounge the ability to send electronic postcards home using either stock photos or their own digital images. The sonic aspect of this installation was used to help travellers browse the collection

[2] SDT impact and rolling models - http://closed.ircam.fr/uploads/media/SDT-0.4.3b.zip

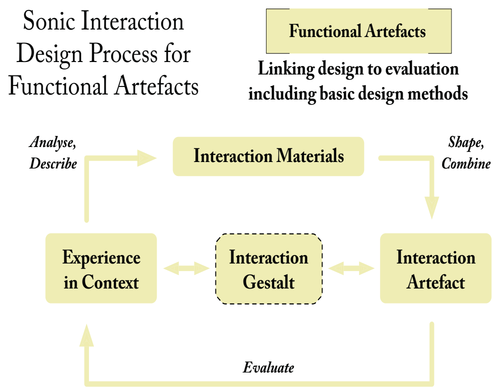


Figure 1: *The design process developed by the CLOSED project [8] for creating functional artefacts.*

of images. The scenario and further details are discussed by Fernström et al. [12]. Brainstorming and mood boards helped generate the initial ideas. These were then sketched and video prototyped or role-played to help evaluate the concepts. The most promising concepts were evaluated and tested using rapid audio prototyping tools such as PureData. This approach and the rapid audio prototyping tools allowed for the creation of four different iterations and evaluations within the space of a month. The process is illustrated in Figure 2.

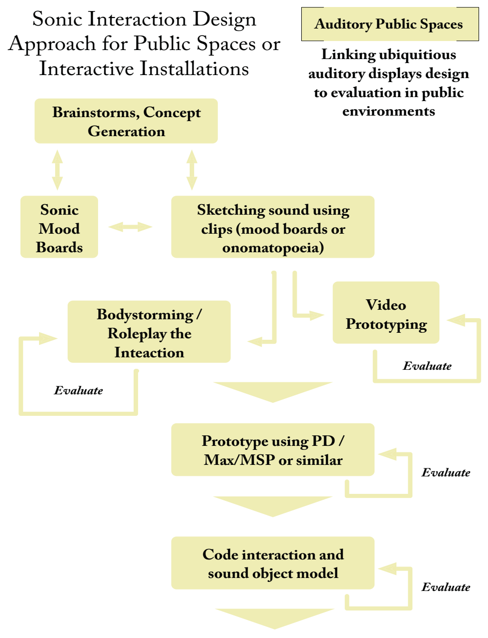

Sonic Interaction Design Approach for Public Spaces or Interactive Installations
Auditory Public Spaces
Linking ubiquitious auditory displays design to evaluation in public environments
Brainstorms, Concept Generation
Sonic Mood Boards
Sketching sound using clips (mood boards or onomatopoeia)
Bodystorming / Roleplay the Inteaction
Video Prototyping
Evaluate
Evaluate
Prototype using PD / Max/MSP or similar
Evaluate
Code interaction and sound object model
Evaluate

Figure 2: *The design process developed for interactive public spaces [12].*

2.3 Designing Narrative Sound Artefacts

The concept of narrative is important in both film and game design and inspired this methodology proposed by Hug [10]. It applies a design oriented research process to explore narrative approaches in the creation of interactive sound artefacts as shown in Figure 3. The view in this methodology is that artefacts are socio-cultural components within everyday life and are dynamic rather than static things. This approach creates exemplar prototypes for possible future scenarios and evaluates them using wizard of oz prototyping in a workshop setting. The studies in this approach generate a set of metatopics that can help create new scenarios and ideas.

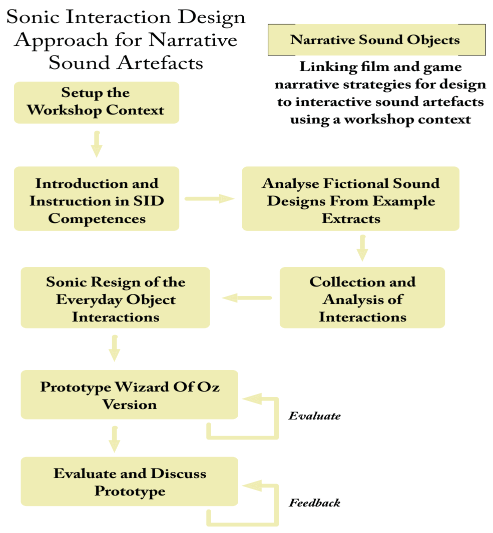

Sonic Interaction Design Approach for Narrative Sound Artefacts
Narrative Sound Objects
Linking film and game narrative strategies for design to interactive sound artefacts using a workshop context
Setup the Workshop Context
Introduction and Instruction in SID Competences
Analyse Fictional Sound Designs From Example Extracts
Collection and Analysis of Interactions
Sonic Resign of the Everyday Object Interactions
Prototype Wizard Of Oz Version
Evaluate
Evaluate and Discuss Prototype
Feedback

Figure 3: *The design process for designing narrative sound objects [10].*

2.4 Shortcomings of the Existing Methodologies

The three methodologies show the focused nature of the existing methodologies in sonic interaction design. They are often heavily design biased and aim at creating or prototyping artefacts for evaluation. This approach typically fits within a sound creation view and often does not focus on the analysis or empirical investigations of the created artefacts. our framework attempts to bridge the gap between sound creation and analysis while ensuring the empirically inspired methods remain accessible and useful for interaction designers.

3 An empirically inspired framework for sonic interaction design

The framework we propose is aimed at providing designers with accessible tools and methods in a manner that allows for the easy bridging of the semantic gap between low-level perceptual studies and high-level semantically meaningful concepts. Our framework takes the view that the sonic interaction design process is split into two stages, sound creation and sound analysis. The sound creation stage is where a real sound is adapted or designed to meet the needs of the designer. It includes where the designer creates a new sound that is specifically tailored to the auditory design or context. The second stage, is the sound analysis stage where the sound or group of sounds are examined to ensure their suitability for use or to gain further insights into them from the perspective of potential listeners. In the cases of the *functional artefacts* [8] and of the *interactive public spaces* [12], the methodologies are both somewhat contained within the first stage of sound creation. The *narrative sound artefacts* [10] methodology is situated within the sound analysis or second stage of the framework. The approaches from these methodologies are tailored for specific goals, while the framework presented here aims to be more generalised. This means that there is a certain overlap from these methodologies that is implicit in the framework. A further caveat is the focus of the methods is at an individual level rather than at a group level, however the framework could easily be combined with group oriented techniques such as rich user cases [13] or the descriptive analysis process [14] to address this issue. The focus on the individual level is because time is a practical consideration for many designers and individual techniques are much less time consuming than most group oriented approaches [14].

3.1 Framework of Sonic Interaction Design

The implicit view we used as the method for evaluation of auditory icons selection in the early conceptual stages of design is shown in Figure 4. This approach consists of a number of successive steps, beginning with a definition of the context and purpose of the auditory display and ending with an actual evaluation of the auditory icons. The framework is open and adaptable to include new types of sounds or methods. The foundations of this framework are presented in this paper, as it is hoped that future research will improve its potential and practicality for interaction designers. The existing methods used in the framework include repertory grids [15], similarity ratings/scaling [16], sonic maps & '*earwitness accounts*' [17], '*earbenders*' [18], the context to basic design approach [19], in addition to aspects from the three earlier methodologies. A deeper introduction into these techniques is given in our earlier research [20].

3.1.1 Sound Creation:

The first stage is the definition, selection, creation, and adhoc evaluation of the sounds. This workflow creates and rapidly assesses the sounds within the design group or by the designer on their own. This approach depends on the skill of the designer as incorrect combinations or choices of sounds may occur, in addition to inappropriate mappings for the domain. The second empirically inspired stage can

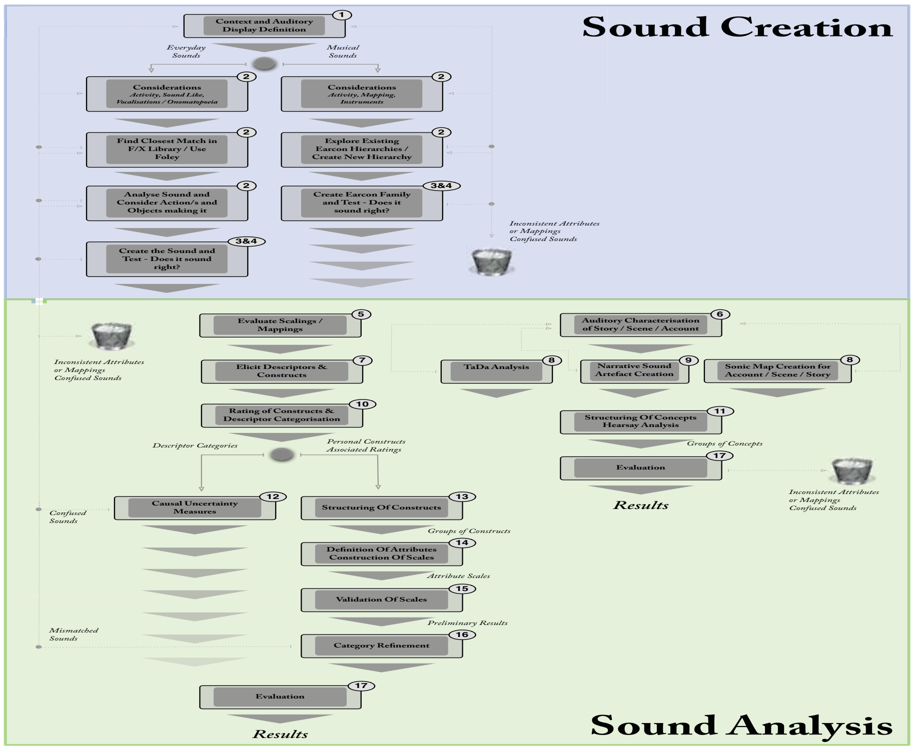

Sound Creation
Context and Auditory Display Definition
1
Everyday Sounds
Musical Sounds
Considerations Activity, Sound Like, Vocalisations / Onomatopoeia
2
Considerations Activity, Mapping, Instruments
2
Find Closest Match in F/X Library / Use Foley
2
Explore Existing Earcon Hierarchies / Create New Hierarchy
2
Analyse Sound and Consider Action/s and Objects making it
2
Create Earcon Family and Test - Does it sound right?
3&4
Inconsistent Attributes or Mappings Confused Sounds
Create the Sound and Test - Does it sound right?
3&4
Evaluate Scalings / Mappings
5
Auditory Characterisation of Story / Scene / Account
6
Inconsistent Attributes or Mappings Confused Sounds
Elicit Descriptors & Constructs
7
TaDa Analysis
8
Narrative Sound Artefact Creation
9
Sonic Map Creation for Account / Scene / Story
8
Rating of Constructs & Descriptor Categorisation
10
Structuring Of Concepts Hearsay Analysis
11
Descriptor Categories
Personal Constructs Associated Ratings
Groups of Concepts
Evaluation
17
Results
Inconsistent Attributes or Mappings Confused Sounds
Confused Sounds
Causal Uncertainty Measures
12
Structuring Of Constructs
13
Groups of Constructs
Definition Of Attributes Construction Of Scales
14
Attribute Scales
Validation Of Scales
15
Preliminary Results
Mismatched Sounds
Category Refinement
16
Evaluation
17
Results
Sound Analysis

Figure 4: *Two stages in our sonic interaction design process.*

help designers build on this stage to ensure the best sound selections are made for the particular context. This stage is shown as the top part (blue highlighting) of Figure 4.

- *1 - Context and Auditory Display Definition*: The purpose of the auditory display is defined, the context is determined, the initial conceptual design including possible sounds and mappings are created.
- *2 - Selection of Sounds*: A pool of sounds that can fit the selected mappings are gathered and organised for evaluation. These sounds can be real, synthetic or a mix of both.
- *3 - Create the Sounds*: If necessary edit the existing sounds or create new sounds. These sounds can be real, synthetic or a mix of both.
- *4 - Listen to the Sounds*: If they do not sound right for the mapping or events, try again with other sounds.

3.1.2 Sound Analysis:

The second stage is the use of empirically inspired methods to improve the selection and understanding of the sounds. The methods present a number of perspectives, depending on whether it is attributes / mappings, confusion metrics, or listeners' narratives being explored. The methods available in the framework are designed to be open for extension to include other adaptions or new methods. This allows for many different perspectives on the sounds and helps inform the designer about the range of possibilities that exist within the given design space. The sound analysis stage is shown as the bottom part (green highlighting) of Figure 4.

- *5 - Evaluate Scaling / Mappings of the Sounds*: The participants listen and compare the sounds and the mappings or attributes being used.
- *6 - Auditory Characterisation of Story/Scene/Account*: This is where a narrative for the sounds and environment are created.
- *7 - Elicit Descriptors & Constructs*: The participants created descriptors for the sounds presented.
- *8 - TaDA & Sonic Mapping*: Analyse the narrative and break it down into the different types and aspects of sounds occurring.
- *9 - Narrative Sound Artefact Creation*: The workshop narrative approach as discussed in section 2.3.
- *10 - Rating of Constructs & Descriptor Categorisation*: Each participant rated the stimuli using these constructs created in the previous stage.

- *11 - Hearsay Analysis / Structuring*: Take the auditory patterns and key sounds to create a short summary of salient points that could be reused in other auditory display contexts.
- *12 - Causal Uncertainty Measures*: The categorisation details can be used to calculate the causal uncertainty of sounds.
- *13 - Structuring of Constructs*: Cluster analysis, multi-dimensional scaling and principal component analysis of the ratings data can clarify attributes and reduce dimensionality of the data as well as removing redundancy.
- *14 - Definition of Attributes, Construction of Scales*: The construct groups are analysed for their content. The appropriate descriptions for the participant identified attributes are then formulated. The rating scales are defined from these attributes.
- *15 - Validation of Scales*: The scales created can be explored in terms of existing categorisations and taxonomies to test the appropriateness of the scales.
- *16 - Category Refinement*: The details from the earlier causal uncertainty measures and from the scales can help suggest the removal of particular sounds as unsuitable for use in the particular sonic context.
- *17 - Evaluation*: The details and results are further analysed to produce the final evaluation results and summary of the evaluation.

3.2 Simplification of the framework

This evaluation method consists of a number of steps, it is envisaged that in future when auditory icons and their subjective qualities are better know that some stages may be simplified or found to be redundant. The use of several methods helps to triangulation the results and shows where additional steps may be added to incorporate new techniques within the framework.

3.3 How to use this framework

There is no how-to or best practise for using this framework or the suggested techniques either individually or collectively. The most appropriate way to adapt these subjective methods is to adopt one or two complementary techniques and use them in a small exploratory design study to see the value they bring to address a particular design issue. The main goal of this paper is to provide a short review for practitioners of the framework and allow them to make the appropriate choice of technique for their design goal.

While some of these methods may not be as ‘rich’ as others, they can still provide additional insights on different facets of the sound or sounds. A number of the methods overlap in terms of what is needed from participants and as a result a single experimental session can easily generate data which can be analysed by several of the methods. The listening test approach [21] asks participants to write verbal descriptions of what they have just heard. These descriptions are similar to the personal constructs collected with the Repertory Grid method [15], the key sounds found using the Sonic Map & Earwitness approach [17], and when described in more detail are similar to the short stories in the Earbenders method [18]. Previous studies [4] have shown how the Repertory Grid method [15] and Ballas’s causal uncertainty method [22] can be used on the same set of collected responses to analysis different yet complementary aspects. The similarity scaling technique [16] uses direct scaling of sound stimuli and as such it requires a separate experimental session. This can be an advantage as participants focus on a single scaling task rather than being asked to scale and provide written descriptions. The method could easily be combined with a context-based rating [23] task, a sorting task or with a discrimination task [22].

4 Discussion

A motivation in creating this framework work was the lack of support for designers wishing to use empirically inspired methods to answer their design questions. The issue is that a typical design problem is more wide ranging than those typically addressed by psychoacoustic studies. This framework presents an approach that is accessible at a reasonable cost to designers and without the need for dedicated facilities such as listening booths or anechoic chambers.

5 Conclusions

This paper introduced a framework for empirically based design within the domains of auditory display and of sonic interaction design. The two key conceptual stages were introduced and related back to the existing methodologies covered in Section 2. This approach builds upon existing techniques and highlights certain areas of overlap where the methods within the framework can be used to complement each other. This framework is the foundation for an accessible empirical approach that can be easily used by novice designers.

The results of this framework will provide greater details to designers on the salient cognitive attributes of sound and help to uncover pragmatic mental models. The aim of this work is to help guide newcomers to sonic interaction design and help them in determining what methods are most appropriate to answer their particular questions or design needs. This review has provided an overview of techniques which, when applied can help deepen knowledge and contribute to answering the question raised by Hug [24] about how to design sounds for ubiquitous technology.

6 Acknowledgements

The research is supported by Grant No. ILRP/202x from Enterprise Ireland under the Industry Lead Research Programme. The authors wish to thank John Bowers, Stephen Barrass, Guillaume Lemaitre, and Nicolas Misdariis for their useful conversations and advice.

7 References

[1] G. Widmer, D. Rocchesso, V. Välimäki, C. Erkut, F. Gouyon, D. Pressnitzer, H. Penttinen, P. Polotti, and G. Volpe, "Sound and music computing: Research trends and some key issues," *Journal of New Music Research*, vol. 36, no. 3, pp. 169–184, 2007.

[2] G. Melles, "An enlarged pragmatist inquiry paradigm for methodological pluralism in academic design research," *Artifact*, vol. 2, no. 1, pp. 3–11, 2008.

[3] J. M. Morse, *Handbook of Mixed Methods in Social and Behavioral Research*, chapter Principles of mixed methods and multimethod research design, Sage, Thousand Oaks, California, 2003.

[4] E. Brazil and J. M. Fernström, "Investigating ambient auditory information systems," in *ICAD 2007*, Montreal, Canada, June 26-29 2007, pp. 326–333.

[5] E.D. Mynatt, "Designing with auditory icons," in *Second International Conference on Auditory Display (ICAD '94)*, G. Kramer and S. Smith, Eds., Santa Fe, New Mexico, 1994, pp. 109–119, Santa Fe Institute.

[6] C. Frauenberger, T. Stockman, and M. L. Bourguet, "A survey on common practice in designing audio in the user interface," in *Proceedings of BCS HCI2007*. British HCI Group, 2007.

[7] J. Nielsen, "Usability engineering at a discount," in *International Conference on Human-Computer Interaction*, Boston, MA, USA, 1989, pp. 394–401, Elsevier Science.

[8] Y. Visell, K. Franinovic, and J. Scott, "Closing the loop of sound evaluation and design (closed) deliverable 3.2 experimental sonic objects: Concepts, development, and prototypes," FP6-NEST-PATH project no: 29085 Project Deliverable 3.2, HGKZ (Zurich), 2008.

[9] L. Ciolfi, M. Fernström, L. J. Bannon, P. Despande, P. Gallagher, C. McGettrick, N. Quinn, and S. Shirley, "The shannon portal installation: Interaction design for public places," *IEEE Computer*, vol. 40, no. 7, pp. 64–71, July 2007.

[10] D. Hug, "Using a systematic design process to investigate narrative sound design strategies for interactive commodities," in *ICAD 2009*, Copenhagen, Denmark, 2009, pp. 19–26.

[11] G. Lemaitre, O. Houix, Y. Visell, K. Franinovic, N. Misdariis, and P. Susini, "Toward the design and evaluation of continuous sound in tangible interfaces: The spinotron," *International Journal of Human-Computer Studies*, vol. in print, 2009.

[12] M. Fernström and E. Brazil, "The shannon portal: Designing an auditory display for casual users in a public environment," in *ICAD 2009*, Copenhagen, Denmark, 2009, pp. 27–30.

[13] A. Pirhonen, K. Tuuri, M-S. Mustonen, and E. Murphy, "Beyond clicks and beeps: In pursuit of an effective sound design methodology," in *Proc. of HAID 2007*, Jyvaskyla, Finland, 2007, LNCS 4813, pp. 133–144, Springer-Verlag Berlin.

[14] N. Zacharov and G. Lorho, "Sensory analysis of sound (in telecommunications)," in *Proc. of European Sensory Network Conference 05*, Madrid, Spain, 2005.

[15] F. Fransella, R. Bell, and D. Bannister, *A manual for repertory grid technique*, John Wiley and Sons, 2004.

[16] T. Bonebright, "Perceptual structure of everyday sounds: A multidimensional scaling approach," in *ICAD 2001*, Helsinki, Finland, 2001, pp. 73–78.

[17] G. W. Coleman, *The Sonic Mapping Tool*, Ph.D. thesis, University of Dundee, August 2008.

[18] S. Barrass, *Auditory Information Design*, Ph.D. thesis, Australian National University, 1997.

[19] K. Franinovic and Y. Visell, "Strategies for sonic interaction design: From context to basic design," in *ICAD 2008*, Paris, France, June 24-27 2008.

[20] E. Brazil and M. Fernström, "Subjective experience methods for early conceptual design of auditory displays," in *ICAD 2009*, Copenhagen, Denmark, 2009, pp. 11–18.

[21] N. J. Vanderveer, *Ecological Acoustics- Human Perception of Environmental Sounds*, Ph.D. thesis, University Of Cornell, 1979.

[22] J. A. Ballas, "Common factors in the identification of an assortment of brief everyday sounds," *J. of Experimental Psychology*, vol. 19, no. 2, pp. 250–267, 1993.

[23] T. L. Bonebright, N. E. Miner, T. E. Goldsmith, and T. P. Caudell, "Data collection and analysis techniques for evaluating the perceptual qualities of auditory stimuli," *ACM Transactions on Applied Perceptions*, vol. 2, no. 4, pp. 505–516, 2005.

[24] D. Hug, "Towards a heremeneutics and typology of sound for interactive commodities," in *CHI'08 Workshop on Sonic Interaction Design: Sound, Information, and Experience*, 2008, pp. 11–16.

A FRAMEWORK FOR SOUNDSCAPE ANALYSIS AND RE-SYNTHESIS

Andrea Valle, Mattia Schirosa, Vincenzo Lombardo
CIRMA-Università di Torino
via Sant'Ottavio 20, 10124, Torino, Italy
andrea.valle@unito.it, mattiaschirosa@yahoo.it, vincenzo@di.unito.it

ABSTRACT

This paper presents a methodology for the synthesis and interactive exploration of real soundscapes. We propose a soundscape analysis method that relies upon a sound object behavior typology and a notion of "sound zone" that collocates objects typologies in spatial locations. Then, a graph-based model for organising sound objects in space and time is described. Finally, the resulting methodology is discussed in relation to a case study.

1 INTRODUCTION

The term "soundscape" was firstly introduced (or at least, theoretically discussed) by R. Murray Schafer in his well-known book *The tuning of the world* [8]. Murray Schafer and his associates of the World Forum For Acoustic Ecology studied for the first time the relation between sounds, environments and cultures. Then, the diffusion of the term has continuously increased, and currently the notion of soundscape plays a pivotal role at the crossing of many sound-related fields. It is worth noting that, despite the profusion of usages, there are neither models nor applications aiming at a simulation of a soundscape starting from the analysis of an existing one. To this goal, here we propose an analytical methodology for the description of soundscapes and a system for its re-synthesis, driven by the analytical results. The system allows for a real-time interaction.

2 A METHODOLOGY FOR ANALYZING EXISTING SOUNDSCAPES

The simulation of an existing soundscape requires to analyze the soundscape itself in order to provide data to be used in the re-synthesis process. The analysis aims at gathering data from the real environment. Our methodology is based on a multi-step process. We start by focusing on an "absent-minded" exploration of the soundscape: the analyst must be perceptually open and adhere to a passive listening strategy. In this way it becomes possible to identify the most relevant sound objects of the overall soundscape, i.e. the ones that are evident even to the least aware listeners. Traditionally, the soundscape studies have insisted on a tripartite typology of sounds in relation to their socio-cultural function: keynote sounds, signal sounds, soundmarks [8]. The identification of sound objects allows for a subsequent classification based on phenomenological and semiotic elements (more later). An active listening strategy is then performed, with the aim of locating the sound objects in the space. It is thus possible to create a sound map. Then, we focus on the analysis of the temporal organization of the soundscape, in order to retrieve specific sequences of sound objects. A database is produced containing the recordings of raw audio material related to the identified sound objects. On one side, large portions of soundscape are recorded with an omnidirectional microphone: so, a large quantity of raw material is available for editing and processing. On the other side, high directivity microphones are used to capture a wide variety of emissions while minimizing undesired background.
Two issues emerge from the analysis.
The first is related to sound classification. The one used by studies in acoustic ecology typically refers to socio-cultural and aesthetic aspects of sound, and it needs to be oriented toward the simulation of soundscape [11], [12]. Here we propose a supplementary classification that focuses on the perceptual and indexical properties of the soundscape and integrates elements from the theory of "sound object" [10] and from the research in "audiovision" [5]. Sound objects can be classified according to the following types:

- events: an event is a single sound object of well-defined boundaries appearing as an isolated figure. In this sense, it is similar to a signal as defined in soundscape studies.
- sound subjects: a sound subject represents the behavior of a complex source in terms of sequencing relations between events. In other words, a sound subject is a description of a sound source in terms of a set of events and of a set of sequencing rules.
- atmospheres: in relation to sound, Böhme has proposed an aesthetics of atmospheres [3]. Every soundscape has indeed one or more specific "scenic atmo-

SMC 2009, July 23-25, Porto, Portugal

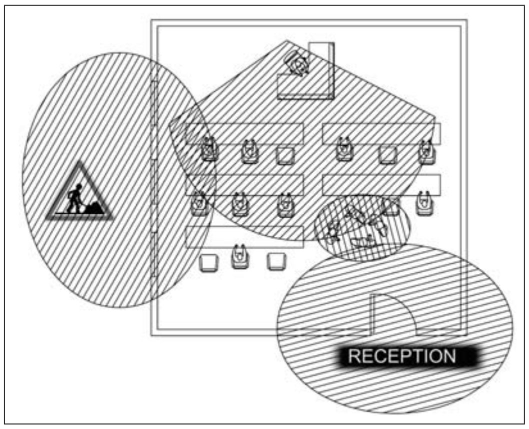


Figure 1. The classroom example: four sound zones.

spheres", which includes explicitly an emotional dimension. An atmosphere is an overall layer of sound, which cannot be analytically decomposed in single sound objects, as no particular sound object emerges. Atmosphere characterizes quiet states without relevant sound events.

A second issue concerns the distribution of sounds in space. Many scholars have noted that a soundscape can be decomposed as a group of several acoustic scenographies, which are then recomposed through the listener's exploring experience [1], [2], [14], [7]. As soundscapes are not uniform, the listener's experience is enhanced when s/he encounters sound aural transitions during her/his exploration of the environment [3]. When a listener is spatially exploring the soundscape, he can notice several perceptual differences in the sounds. In particular, the sound environment can be decomposed into internally homogeneous sub-parts. These sub-parts are here referred to as "sound zones". By the study of sound zones, sound aural transitions between them can be individuated, analysed and and re-synthesized in the simulation. Sound zones can differ in dimension and in number of elements, but they are characterized by typical sources, i.e. sound emissions are often present in a region and absent (or only rarely heard) in the others. The soundscape will then results from the summation of all the sound zones, that the listener will be able to explore. As an example, we can consider the following situation: in a university classroom with acoustic insulation walls, closed doors and windows, a professor is speaking in front of a very silent audience. The professor voice is loud and clear in all the classroom, without any relevant irregularity. By contrast, we can imagine the opposite situation: doors and windows are open, the thin walls are incapable of blocking any environmental sound, outside there are roadworks, a reception party is running in the hall just behind the door, a few students joke and laugh while the professor keeps explaining loudly. This second soundscape (represented in Figure 1) is completely different from the first one. Considering that the classroom is wide enough, it would be very simple to move around the space and run across several recognizable micro-soundscapes. Someone near the door can notice that reception party sounds are louder than any other sound source coming from the classroom. As s/he moves to the desk, s/he can hear the professor's voice, and so on. In the first case it is possible to identify a soundscape consisting of one only zone; in the second case four sound zones are clearly defined. Thus, even if their boundaries can be fuzzy, each zone can be considered as completely independent from another. This means that it is possible to describe the behavior of each zone. A soundscape results from the interweaving of each zone behavior.

3 A GRAPH-BASED MODEL FOR SOUNDSCAPE RE-SYNTHESIS

In this section we propose a model able to re-synthesize a soundscape starting from sound object introduced before. We discuss a system to organise sequences of sound objects in space and time, then we take into account the relation between the system and sound objects/sound zones.
The re-synthesis process meets two requirements. First, it must be generative, i.e. capable to create an infinite set of sequences of sound objects from a finite set of sampled sound objects. Second, the algorithm must be able to merge the information coming from the sequencing process with the user's navigation data. In this way, the simulated soundscape can be explored interactively. The generative model, named GeoGraphy, is based on graphs (for a more detailed description see [13]). Graphs have proven to be powerful structure to describe musical structures ([9]). Still, a common feature of all these graph representations devised for music is that they generally do not model temporal information: on the contrary, the model relies on time-stamped sequences of sound objects. The sequencing model is a direct graph (see Figure 2), where each vertex represents a sound object (sampled from the analysis phase) and each edge represents a possible sequencing relation on pairs of sound objects. This graph is actually a multigraph, as it is possible to have more than one edge between two vertices; it can also include loops. Each vertex is labeled with its relative sound object duration and each edge with the temporal distance between the onsets of the two sound objects connected by the edge itself. The graph defines all the possible sequencing relations between adjacent vertices. A sequence of sound objects (a *track*) is achieved through the insertion of dynamic elements, called "graph actants". A graph actant is initially associated with a vertex (that becomes the origin of a path); then the actant navigates the graph by following the directed edges according to some probability distribution. Each vertex emits a sound object at the passage of a

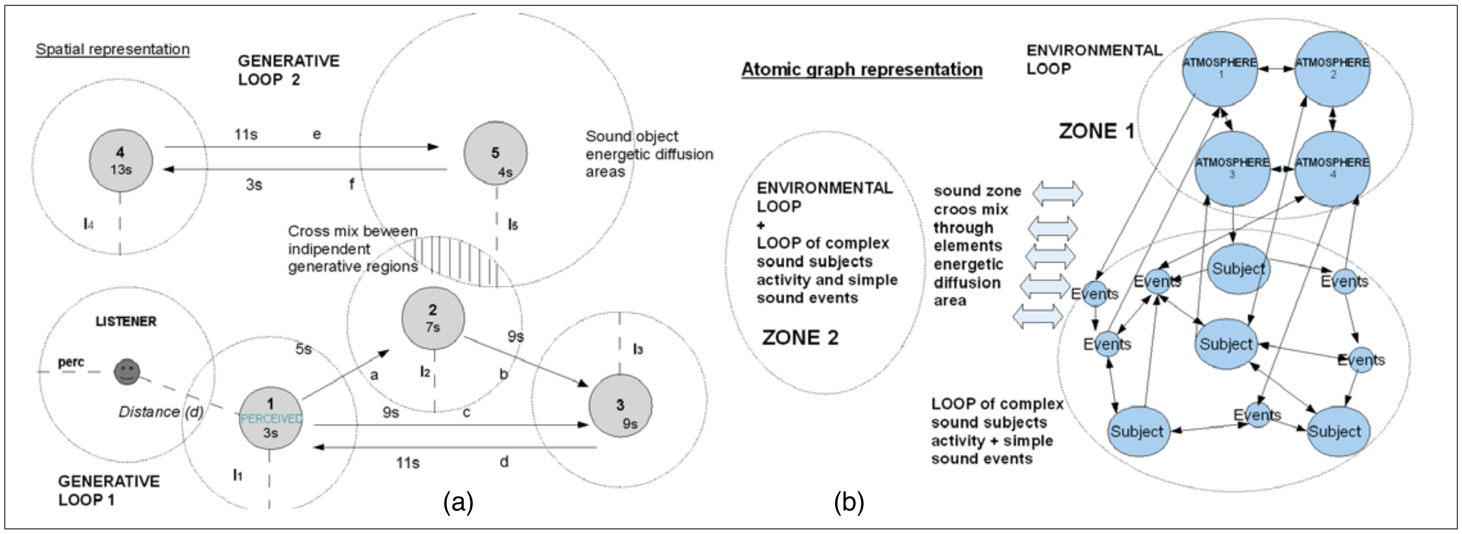

Spatial representation
GENERATIVE LOOP 2
Sound object energetic diffusion areas
Cross mix beween indipendent generative regions
LISTENER
perc
Distance (d)
PERCEIVED
GENERATIVE LOOP 1
(a)
Atomic graph representation
ENVIRONMENTAL LOOP
ZONE 1
ENVIRONMENTAL LOOP + LOOP of complex sound subjects activity and simple sound events
ZONE 2
sound zone croos mix through elements energetic diffusion area
LOOP of complex sound subjects activity + simple sound events
ATMOSPHERE 1
ATMOSPHERE 2
ATMOSPHERE 3
ATMOSPHERE 4
Events
Subject
(b)

Figure 2. (a) Graphs and Listener. (b) A graph representing the relations among atmospheres, events and sound subjects in a zone.

graph actant. Multiple independent graph actants can navigate a graph structure at the same time, thus producing more than one track. In case a graph contains loops, tracks can also be infinite. As modeled by the graph, the sound object duration and the delay of attack time are independent: as a consequence, it is possible that sound objects are superposed. This happens when the duration of the starting vertex label is longer than the duration of the chosen edge (in Figure 2, the edge e between vertex 4 and 5). Thus, there will be as many superposed tracks as graph actants. In order to allow the inclusion of the exploration process graphs are placed in a two-dimensional space: in this way, the original location of a sound object can be represented. Each vertex is given a radiation area: the radius indicates the maximum distance at which the associated sound object can be heard. Inside the map of graphs, a "Listener" is defined. The Listener is identified by a position, an orientation and an audibility area (see Figure 2, a). The position is expressed as a point in the map; the orientation as the value in radiant depending on the user's interaction control; the audibility area defines the perceptual boundaries of the Listener. The Listener can be thought as a function that filters and parameterizes the sequences of sound objects generated by the graph actants. Every time a vertex is activated by a graph actant, the algorithm calculates the position of the Listener. If the intersection between the Listener's audibility area and the vertex's energetic area is not void, then the Listener's orientation and distance from the vertex are calculated, and all the data (active vertex, position, distance and orientation of the Listener) are passed to a DSP module retrieving from the database the recorded samples and processing them according to some spatialization model (e.g. reverberation, low-pass filtering, amplitude scaling, etc.).

To sum up, in our system a soundscape emerges as the relation between the set of tracks generated by the graph actant navigating the graphs and the filtering function defined by the Listener.

Through the GeoGraphy model and the sound zone modular description it is possible to generate a target complex soundscape. In Figure 2 (a), each zone is described by a dedicated graph. As discussed, zones can overlap their audibility with other zones, depending on the radiation of the sound sources that compose it. For that reason GeoGraphy provides the radiation area concept: it indicates the maximum distance at which the associated sound object can be heard. It is set according to dynamics annotations taken during the analysis phase. It is thus possible to regulate the radius of each element to interbreed the parent zone with the others. This modular description process allows to easily represent "sound pollution zones". In Figure 2 (a) a zone is represented by the generative loop 2 between vertices 4 and 5, respectively having radii I_4 and I_5. Similarly, "the generative loop 1" individuates one more zone, and the intersection between the elements 2 and 5 is the portion of space where the sounds of the two zones can be hearable. A soundscape is the summation of all the graphs apt to describe the sound zones that a listener will be able to explore. A zone in itself is the summation of all the sub-graphs representing atmospheres, events, and sound subjects. The soundscape is made by a continuous fusion process between sound figures and backgrounds. In this sense, each zone has a core granting continuity to the structure, and to the resulting auditory stream. The core is formed by the atmospheres, that consist several ambient sound materials (i.e. long field recordings), aiming at representing different natural “almost-quiet state” nuances. In Figure 2 (b) the atmospheres 1 to 4 allow the formation of a background against which semiotically and indexically relevant signals can emerge. Atmospheres can be connected to other atmospheres and to events and sound subjects. Their durations is typically set longer than the edges connecting them to events and sound subjects: in this way, atmospheres are still present when the events and sound subjects are activated by the actant. Atmospheres can then generate a background layer, while sound events and sound subjects reach the close-up perceptual level and then quickly disappear. Events can be thought as isolated signals. On the contrary, sound subjects feature a double nature. In Figure 2 (a) they are represented, for sake of simplicity, as single vertices, standing for a complex but unitary acoustic behavior. But a sound subject properly is a subgraph, organized recursively in a core surrounded by events, like a sub-zone. An example of sound subject is discussed in the next section.

4 A CASE STUDY: THE MARKET OF THE PORTA PALAZZO IN TURIN

The market is a typical case of a socio-culturally relevant soundscape. In particular, the market of the Porta Palazzo in Turin has a long tradition as it has been established more than 150 years ago. It is the greatest outdoor market in Europe, and it represents the commercial expression of the cultural heritage of the city of Turin. During the century, it has tenaciously retained its identity, characterized by the obstinate will of the workers of sharing its government’s responsibility. It is probably the part of Turin where the largest number of social different realities and cultures coexist, both of Italian and of foreign origins [6]. As a consequence, its soundscape manifests an impressive acoustic richness. First, it includes languages and dialects from all the regions of Italy, South America, Eastern Europe, North Africa. More, there are many qualitatively different sound sources: every day the market serves 20,000 persons (80,000 on Saturday), and 5,000 persons are working in it every day. It can be said that the Porta Palazzo soundscape belongs to the Italian cultural heritage.

The analysis of the case-study initially focused on the socio-cultural dimension of the market and has started from the urban redevelopment survey by the Porta Palazzo public advisory committee [6]. More, short informal interviews with local workers, customers and workers’ representatives have been realized. The interviews were done during the first “absentminded” explorations of the place, that made it possible to annotate the most pervasive sound objects. As an example, the sound of plastic shopping-bags is a unique keynote sound represented as a mass of sound events. The shouts of the merchants is another multi-particle keynote in which the listener of the marketplace soundscape is immersed: the most intense, vibrant, repetitive, significant advertising messages have been recorded to be simulated. In some sense, their sum is the pervasive call of the market: the Porta Palazzo voice. Finally, one can notice that there is a specific keynote sound in certain border regions that invades all the space: the noise of motor vehicles and carriages. As an example, in the customers opinion, the arrival of the streetcar number 4 is the unique sound source that can be heard throughout the whole soundscape, acquiring specific nuances in each zone (i.e. due to reverberation and to low frequency distance attenuation). Hearing this sound object makes one think immediately of the Porta Palazzo market. This is a very interesting strong semantic association picked up during interviews.

Then, sound signals have been taken into account, in particular those related to specific stands, giving origin to complex sound subjects: five typical stand sounds have been analyzed. It must be noted that a reduced set of samples has proven to be rich enough to describe different stands, as they could be differentiated by their grouping in different graph structures. A particular sequence of events has led to create the specific “shopping” sound subject: plastic rustle, paper rustle, clinking coins, cash opening, clinking coins, cash closing. The *stands of the anchovy sellers* have proven to be very different from all other stands: they include sounds of metal cans, of anchovies being beaten over wood plates, of olives thrown in oil, of noisy old scales.

Subsequently, we define the sound zones: the analysis of the soundscape led to five independent zones formed by distinctive elements. In this way, it has been possible to record specific atmospheres. In Figure 3 all the zones are assigned an identifying index.

The zones 1 and 2 are characterized by the sounds of motor vehicles. Zone 1 is mainly characterized by a sound atmosphere made up of little delivery trucks, hand-carts and gathering of packing boxes from stands. It is the only street accessible by any vehicles as bus, trams, cars and motorbike. Instead, in the zone 2 there are two important sound features: the load area of big delivery trucks and the street dedicated to public transport, with rail system allowing streetcar passage. Both the zones present sounds related to bread, mint and spice hawkers. Zone 3 is a diffused area showing a mixup of sounds related to market and to street/parking areas. This feature has required to aptly adjust the radius of sound sources to describe its fuzzy sonic boundaries. In addition, some emissions related to the daily process of assembling/disassembling stands are present. Zone 4 is formed by different and rare stands; it presents a less prominent sound density because the passage area is bigger, so the sound of walking costumers, hand-cart distribution, empty box collecting process, are louder than other sound objects. More, many atypical stands are positioned in this zone, making its atmosphere unique. The motor sound is almost imperceptible, with the exception of some very loud source (as

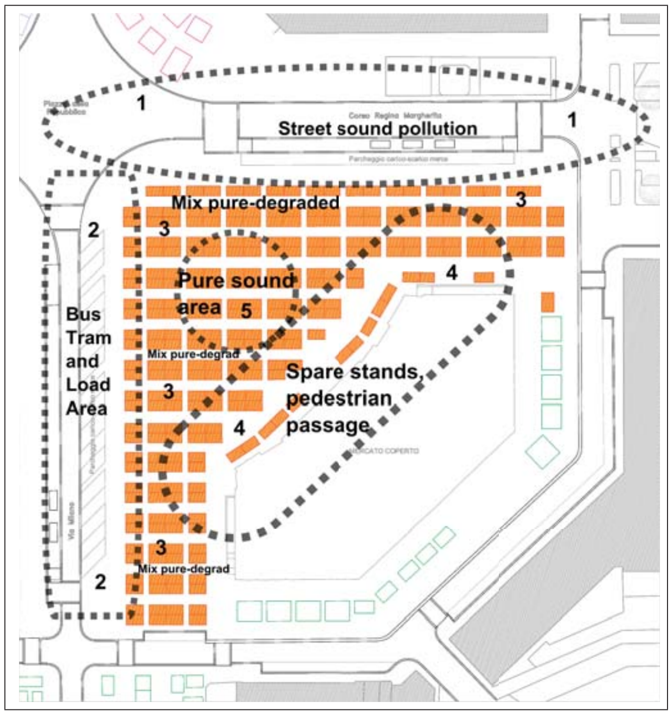


Figure 3. Plant of the Porta Palazzo market: organization in sound zones.

streetcar 4). Zone 5 presents only vegetable and fruit stands: transit ways are thin and rare, and only walking people can pass through. The shouts of the merchants reach the highest intensity and mask all the pollution sound coming from the other zones, while the many sound signals (activities and voices) make the soundscape particularly frenetic, a disarranged composition of sound objects making it a "pure" example of market soundscape.

The graph structure of a sub-part of zone 2 is showed in Figure 4 (a). The graph is a sub-part of zone 2. Edges define the sequencing rules between a possible atmosphere (labelled "1:XAtmo...", a four minute recording) of that specific sub-part, and nine indexical sound events. The atmosphere describes the almost quiet state of that area, generated by the continuous walking of costumers and the activity of some mint hawkers. The sound events describe activities by different vehicles. The number of repetitions of a sound object (i.e bus, tram, delivery truck, motorbike) is proportional to its statistical relevance: there are four tram objects, then two for bus and trucks, and only one for motorbike. No car was noticed here. The graph is cyclic, thus generating potentially infinite tracks. In this case, each possible path is designed to have the same duration of the atmosphere. So the time duration of edge connection $Edur_{x_y}$ between vertices are set according to the following rule:

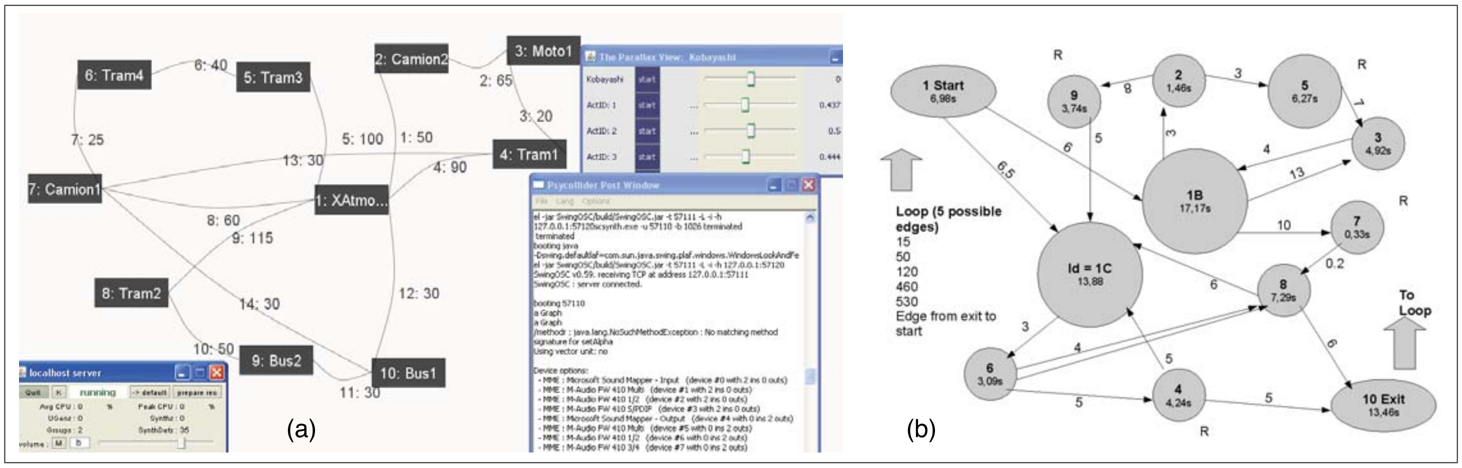
$$Edur_{Atm_2} + Edur_{2_3} + ... + Edur_{x_{Atm}} = Vdur_Atm$$

In this way a long, looping background is continuously varied by the superposition of different other sound objects. By only using nine objects it has been possible to represent a complex soundscape.

Figure 4 (b) depicts the graph of a sound subject. The graph represents the behavior of a delivery truck. The delivery trucks arrive at that zone, unload the products, and leave back. By connecting three "core" objects $1b$, $1c$, 5 and nine sound events, it allows the simulation of several instances of the truck. Here the sound subject reveals its sub-zone nature. The topological structure of the graph includes a start event (1) and an end event (10). The core objects are almost quiet recordings. As an example, $1b$ refers to a stationary truck with running motor while $1c$ refers to a truck making some accelerations. They are placed topologically in the center of the graph structure, providing a continuous background against which other smaller sound objects appear and disappear. As in the previous example, all the possible paths reactivate a core before its duration has finished. But there is an exception: after the end event the auditory stream stops, as the graph is acyclic. The graph can be made cyclic by the addition of edges connecting the end vertex 10 to the start vertex 1. These looping edges can have durations spanning over a large interval, from 15 to 530 seconds. After a path simulating the truck delivery has reached the end event, it is thus possible that the start event is emitted straight after that: the sound result will then be perceived as an activity of the same acousmatic subject. By contrast, when the path restarts a long time after the end vertex has finished, the result can be perceived as the arriving in the soundscape of a new truck.

5 CONCLUSIONS AND FUTURE WORK

The notion of soundscape is increasingly relevant not only in contemporary culture and acoustic ecology but also in many other fields, such as, e.g. audiovisual productions, in which soundscape is now a fundamental part of the whole soundtrack. The proposed system is able to generate soundscapes from original sound materials but without relying on loops. In this way, the typical "sound mood" of the original space is preserved: at the same time, the resulting soundscape is no more fixed, but undergoes a continuous variation thanks to the graph dynamics, due to probabilistic connections. The system was been implemented in the SuperCollider audio real-time synthesis programming environment, which allows the efficient real-time synthesis of the soundscape and its interactive exploration by the user [15]. A major issue in our system concerns the generation of multigraphs, actually to be carried out manually and potentially quite time-consuming. We are planning to extend the system so to include the automatic generation of graphs starting from information stored in the database or from sound-related semantic repertoires (see [4]). The



```
el -jar SwingOSC/build/SwingOSC.jar -t 57111 -L -i -h
127.0.0.1:57120scsynth.exe -u 57110 -b 1026 terminated
 terminated
booting java
-Dswing.defaultlaf=com.sun.java.swing.plaf.windows.WindowsLookAndFe
el -jar SwingOSC/build/SwingOSC.jar -t 57111 -L -i -h 127.0.0.1:57120
SwingOSC v0.59. receiving TCP at address 127.0.0.1:57111
SwingOSC : server connected.

booting 57110
a Graph
a Graph
/methodr : java.lang.NoSuchMethodException : No matching method
signature for setAlpha
Using vector unit: no

Device options:
  - MME : Microsoft Sound Mapper - Input   (device #0 with 2 ins 0 outs)
  - MME : M-Audio FW 410 Multi   (device #1 with 2 ins 0 outs)
  - MME : M-Audio FW 410 1/2   (device #2 with 2 ins 0 outs)
  - MME : M-Audio FW 410 S/PDIF   (device #3 with 2 ins 0 outs)
  - MME : Microsoft Sound Mapper - Output   (device #4 with 0 ins 2 outs)
  - MME : M-Audio FW 410 Multi   (device #5 with 0 ins 2 outs)
  - MME : M-Audio FW 410 1/2   (device #6 with 0 ins 2 outs)
  - MME : M-Audio FW 410 3/4   (device #7 with 0 ins 2 outs)
```

```
Avg CPU : 0 %     Peak CPU : 0 %
UGens : 0         Synths : 0
Groups : 2        SynthDefs : 35
volume : M b
```

Figure 4. (a) Screenshot from the current SuperCollider implementation: the graph describes the sequencing rules of a part of zone 2. (b) Graph sequencing rules of the delivery truck sound subject.

database itself can eventually include not only sound samples created from direct recording but also from available sound libraries. Sound automatic recognition could led the automatic definition of some parameter (i.e. sample durations labelling vertices). An interesting perspective is to investigate user-generated, online databases such as Freesound [1] : in this case the graph generation process can be governed by social tagging.

6 REFERENCES

[1] Pascal Amphoux. *L'identité sonore des villes européennes, Guide méthodologique à l'usage des gestionnaires de la ville, des techniciens du son et des chercheurs en sciences sociales*. publication IREC, EPF–Cresson, Lausanne–Grenoble, 1993.

[2] Jean-François Augoyard and Henry Torgue. *Repertorio degli effetti sonori*. Lim, Lucca, 2003.

[3] Gernot Böhme. *Ecologia della musica: Saggi sul paesaggio sonoro*, chapter Atmosfere acustiche. Un contributo all'estetica ecologica. Donzelli, 2004.

[4] P. Cano, L. Fabig, F. Gouyon, M. Koppenberger, A. Loscos, and A. Barbosa. Semi-automatic ambiance generation. In *Proceedings of the International Conference of Digital Audio Effeccts (DAFx'04)*, pages 1–4, 2004.

[5] Michel Chion. *L'audiovision. Son et image au cinéma*. Nathan, Paris, 1990.

[6] Studio di Ingegneria ed Urbanistica Vittorio Cappato. 50 centesimi al kilo: La riqualificazione del mercato di Porta Palazzo dal progetto al cantiere. Technical report, Comune di Torino, Torino, 2006.

[7] Albert Mayr, editor. *Musica e suoni dell'ambiente*. CLUEB, Bologna, 2001.

[8] R. Murray Schafer. *The Tuning of the World*. McClelland & Steward and Knopf, Toronto and New York, 1977.

[9] Curtis Roads. *The computer music tutorial*. The MIT Press, Cambridge, Mass., 1996.

[10] Pierre Schaeffer. *Traité des objets musicaux*. Seuil, Paris, 1966.

[11] Barry Truax. *Acoustic Communication*. Greenwood, Westport, CT, 1984.

[12] Barry Truax. Model and strategies for acustic design. In H. Karlsson, editor, *Hör upp! Stockholm, Hey Listen! - Papers presented at the conference on acoustic ecology*, Stockholm, 1998.

[13] Andrea Valle, Vincenzo Lombardo, and Mattia Schirosa. A graph-based system for the dynamic generation of soundscapes. In *Proceedings of the 15th International Conference on Auditory Display*, pages 217–224, Copenhagen, 2009.

[14] VV.AA. Résumé de l'étude de conception et d'amenagement du paysage sonore du secteur de la Sucrerie - St. Cosme. Technical report, Acirene–atelier de traitement culturel et estetique de l'environnement sonore, 2007.

[15] Scott Wilson, David Cottle, and Nick Collins, editors. *The SuperCollider Book*. The MIT Press, Cambridge, Mass., 2009.

[1] http://www.freesound.org/

SOUND AND THE CITY: MULTI-LAYER REPRESENTATION AND NAVIGATION OF AUDIO SCENARIOS

Luca A. Ludovico, Davide A. Mauro
Laboratorio di Informatica Musicale (LIM)
Dipartimento di Informatica e Comunicazione (DICO)
Università degli Studi di Milano
Via Comelico 39/41 - 20135 Milano (Italy)
{ludovico, mauro}@dico.unimi.it

ABSTRACT

IEEE 1599-2008 is an XML-based standard originally intended for the multi-layer representation of music information. Nevertheless, it is versatile enough to describe also information different from traditional scores written according to the Common Western Notation (CWN) rules. This paper will discuss the application of IEEE 1599-2008 to the audio description of paths and scenarios from the urban life or other landscapes. The standard we adopt allows the multi-layer integration of textual, symbolical, structural, graphical, audio and video contents within a unique synchronized environment. Besides, for each kind of media, a number of digital objects is supported. As a consequence, thanks to the features of the format the produced description will be more than a mere audio track, a slideshow made of sonified static images or a movie. Finally, an *ad hoc* evolution of a standard viewer for IEEE 1599 documents will be presented, in order to enjoy the results of our efforts.

1 INTRODUCTION

IEEE 1599-2008 is originally a format to describe single music pieces. For example, an IEEE 1599 document can be related to a pop song, to an operatic aria, or to a movement of a symphony.

Based on XML (eXtensible Markup Language), it follows the guidelines of IEEE P1599, "Recommended Practice Dealing With Applications and Representations of Symbolic Music Information Using the XML Language". This IEEE standard has been sponsored by the Computer Society Standards Activity Board and it was launched by the Technical Committee on Computer Generated Music (IEEE CS TC on CGM) [1].

The innovative contribution of the format is providing a comprehensive description of music and music-related materials within a unique framework. In fact, the symbolic score - intended here as a sequence of music symbols - is only one of the many descriptions that can be provided for a piece. For instance, all the graphical and audio instances (scores and performances) available for a given piece are further descriptions; but also text elements (e.g. catalogue metadata, lyrics, etc.), still images (e.g. photos, playbills, etc.), and moving images (e.g. video clips, movies with a soundtrack, etc.) can be related to the piece itself. Please refer to [2] for a complete treatment of the subject. As explained in Section 4, such a rich description allows the design and implementation of advanced browsers.

In this work we are interested in a particular application of IEEE 1599 that goes beyond the original goals of the standard. In fact, instead of applying it to a traditional CWN score, we are going to describe in IEEE 1599 the soundscape of a urban environment using a city map as a score and the different hours of a day to generate different performances.

In the following we will introduce the key features of the standard comparing their traditional meaning in the music field to our new perspective. After, we will describe a generalized viewer for IEEE 1599 format usable both for traditional music pieces and for our goal, namely the audio scenario reproduction. Finally, a case study will be discussed, by using the audio material recorded during 2008 Sound and Music Computing conference held in Genoa (Italy).

Before starting the discussion, a point should be clarified. In our work, a format to encode music information is adapted in order to provide a comprehensive description of a sound environment. This is made possible by the flexibility of the XML encoding we adopt, but it could seem a forcing. We have chosen a format oriented to music since a score is made of symbols corresponding to music events; in our case, the concepts of score and event must be generalized, but the navigation of the audio scenario is similarly driven by events belonging to a predetermined "score". Among many XML-based formats available for music description, IEEE 1599 has proved to be effective, and the reasons are explained in

SMC 2009, July 23-25, Porto, Portugal

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ieee1599 SYSTEM
 "http://standards.ieee.org/downloads/1599/1599
                              -2008/ieee1599.dtd ">
<ieee1599>
  <general>...</general>
  <logic>...</logic>
  <structural>...</structural>
  <notational>...</notational>
  <performance>...</performance>
  <audio>...</audio>
</ieee1599>
```

Figure 1. The XML stub corresponding to the IEEE 1599 multi-layer structure.

Section 2. As shown by the case study, the final result is a multimedia description of city sounds that goes beyond a collection of unrelated materials: the audio environment is depicted at different moments of a typical day; these audio objects are related and synchronized through the occurrence of shared events, which constitute the common score. Furthermore, audio contents are integrated with other multimedia objects both to provide a comprehensive description and to allow innovative ways of browsing.

2 DEFINITION OF MUSIC EVENT

The mentioned comprehensiveness in music description is realized in IEEE 1599 through a multi-layer environment. The XML format provides a set of rules to create strongly structured documents. IEEE 1599 implements this characteristic by arranging music and music-related contents within six layers [3]:

- *General* - music-related metadata, i.e. catalogue information about the piece;
- *Logic* - the logical description of score in terms of symbols;
- *Structural* - identification of music objects and their mutual relationships;
- *Notational* - graphical representations of the score;
- *Performance* - computer-based descriptions and executions of music according to performance languages;
- *Audio* - digital or digitized recordings of the piece.

In IEEE 1599 code, this 6-layers layout corresponds to the one shown in Figure 1, where the root element `ieee1599` presents 6 sub-elements.

The previous list is clearly related to music contents, but in our work layers can be used in a wider context. Before discussing this matter in depth, we have to introduce a key concept of the format, namely the spine.

Since contents are displaced over various levels, what is the device that keeps heterogeneous descriptions together and allows to jump from one description to another? The *Logic* layer contains an *ad hoc* data structure that answers the question. When a user encodes a piece in IEEE 1599 format, he/she must specify a list of music events to be organized in a linear structure called "spine". Please refer to Figure 2 for a simplified example of spine. Inside this structure, music events are uniquely identified by the `id` attribute, and located in space and time dimensions through `hpos` and `timing` attributes respectively.

Each event is "spaced" from the previous one in a relative way. In other words, a 0 value means simultaneity in time and vertical overlapping in space, whereas a double value means a double duration of the previous music event with respect to a virtual unit. The measurement units are intentionally unspecified, as the logical values expressed in spine for time and space can correspond to many different absolute values in the digital objects available for the piece.

In the example shown in Figure 2, and regarding it as a music composition, event `e3` forms a chord together with `e2`, belonging either to the same or to another part/voice, as the attributes values of the former are 0s. Similarly, we can affirm that event `e3` should last twice the duration of `e0` (and `e1`), as `e4` occurs after 2 time units whereas `e1` (and `e2`) occurs after only 1 time unit. For further details please refer to the official IEEE draft of the format [4].

In conclusion, the role of the structure known as spine is central for an IEEE 1599 encoding: it provides a complete and sorted list of events which will be described in their heterogeneous meanings and forms inside other layers. Please note that only a correct identification inside spine structure allows an event to be described elsewhere in the document, and this is realized through references from other layers to its unique `id` (see Section 3). Inside the spine structure only the entities of some interest for the encoding have to be identified and sorted, ranging from a very high to a low degree of abstraction.

In the context of music encoding in IEEE 1599, how can be a *music event* defined from a semantic point of view? One of the most relevant aspects of the format, which confers both descriptive power and flexibility, consists in the loose but versatile definition of event. In the music field, which is the typical context where the format is used, a *music event* is a clearly recognizable music entity, characterized by well-defined features, which presents aspects of interest for the author of the encoding. This definition is intentionally vague in order to embrace a wide range of situations. A common case is represented by a score where each note and rest are considered music events. The corresponding spine will list such events by as many XML sub-elements (also referred as *spine events*).

However, the interpretation of the concept of music event can be relaxed. A music event could be the occurrence of a

```
<ieee1599>
  ...
  <logic>
    <spine>
      <event id="e0" timing="0" hpos="0" />
      <event id="e1" timing="1" hpos="1" />
      <event id="e2" timing="1" hpos="1" />
      <event id="e3" timing="0" hpos="0" />
      <event id="e4" timing="2" hpos="2" />
      <event id="e5" timing="2" hpos="2" />
      ...
    </spine>
    ...
  </logic>
  ...
</ieee1599>
```

Figure 2. An example of simplified spine.

new chord or tonal area, in order to describe only the harmonic path of a piece instead of its complete score, made of notes and rests.

Brought to the extreme, the meaning of music event can be extended to comprehend audio events, such as the ringing of church bells or the environmental sounds of a square. Starting from this point of view, our works aims at discovering and exploiting the potentialities of IEEE 1599 format. This a challenging matter as both the format and a number of software tools (e.g. viewers and editors) are already available, but the attempt to apply them to this context is completely original.

3 EVENTS IN A MULTI-LAYER ENVIRONMENT

After giving a correct interpretation to the concept of spine-event, and after the creation of the spine structure, events are ready to be described in the multi-layer environment provided by IEEE 1599. As stated in Section 2, the format includes six layers, which implies 6 families of descriptors for contents.

This section will show that the concept of heterogeneous description is implemented in IEEE 1599 by heterogeneous descriptions of each event contained in spine. While heterogeneity is supported by the whole, inside each layer we find homogeneous contents, namely contents of the same type. The *Audio* layer, for example, can link *n* different performances of the same piece, as well as recordings taken at different times in the same place. In order to obtain a valid IEEE 1599 document, not all the layers must be filled; however their presence provides richness to the description.

In musical terms, the layer-based mechanism allows heterogeneous descriptions of the same piece. For a composition, not only its logical score, but also the corresponding music sheets, performances, etc. can be described. In this context instead, heterogeneity is employed in order to provide a wide range of audio descriptions of the same environment. This concept will become clear in the following.

Now let us focus on the presence and meaning of events inside each layer. The *General* layer contains mainly catalogue metadata that are not referable to single music events (e.g. title, authors, genre, and so on). From the perspective of this paper, the *General* layer could have a poor meaning. Nevertheless, this layer presents a sub-element called `related_files`, a container for 1..*n* specification(s) of external digital objects such as photos, somehow related to the piece but not directly related to the occurrence of music events. For `related_files` sub-element, two attributes are available: `start_event_ref` and `end_event_ref`, containing the identifiers of events listed in spine. These attributes allow to synchronize respectively the appearance and disappearance of static graphical objects with the occurrence of spine events, and they are useful for multimedia presentations. In our case study, we will employ this feature to implement a slideshow of the route, made of images and short text descriptions.

The *Logic* layer, which is the core of the format, faces music description from a symbolic point of view: it contains both the spine, i.e. the main time-space construct aimed at the localization and synchronization of events, and the symbolic score in terms of pitches, durations, etc. The latter aspect is not present in our work; on the contrary the former "logic" description takes a key role for all the other layers, which refer to spine identifiers in order to link heterogeneous descriptions to the same events. Please note that only spine is strictly required by IEEE 1599 format.

Originally, the *Structural* layer has been designed to contain the description of music objects and their causal relationships, from both the compositional and musicological point of view. This layer is aimed at the identification of music objects as aggregations of music events and it defines how music objects can be described as a transformation of previously described objects. Here music events are referred in order to create horizontal (e.g. melodic themes), vertical (e.g. chords), or other aggregations of symbols (e.g. generic segments). In our work, this layer can be used to highlight relationships among events along the route. For example, if two squares are encountered along the way, the *Structural* layer can link the corresponding events. As usual, event localization in time and space is realized through spine references.

For the remaining layers, the meaning of events is more straightforward. The *Notational* layer describes and links the graphical implementations of the logic score, where music events - identified by their spine id - are located on digital objects by absolute space units (e.g. points, pixels, millimeters, etc.). In the case of environmental sounds, the places where they are recorded can be identified over a map. These maps can be the counterpart of the graphical scores as regards our work.

The *Performance* layer is devoted to computer-based performances of a piece, typically in sub-symbolic formats such as Csound, MIDI, and SASL/SAOL. This layer is not used for our goals.

In the *Audio* layer events are described and linked to audio digital objects. Multiple audio tracks and video clips, in a number of different formats, are supported. The device used to map audio events is based on absolute timing values expressed in milliseconds, frames, and so on. Our case study, as discussed in Section 5, includes only three audio tracks, but video clips could be included as well.

Finally, let us concentrate on the cardinalities supported for events layer-by-layer. In the *Logic/Spine* sub-layer, the cardinality is 1 - namely the presence is strictly required - as all the events must be listed in the spine structure. In the *Audio* layer, on the contrary, the cardinality is [0..*n*] as the layer itself can be empty (0 occurrences), it can encode one or more partial tracks where the event is not present (0 occurrences), it can link a complete track without repetitions (1 occurrence), a complete track with repetitions (*n* occurrences), and finally a number of different tracks with or without repetitions (n occurrences). Similarly, the *Notational* layer supports [0..*n*] occurences. In our case, the relevant events listed in spine will be mapped only once for each digital object.

4 BROWSING OF IEEE 1599 DOCUMENTS

In the current section we treat the problem of browsing when many multimedia objects are available, as in the IEEE 1599 environment. Our purpose is presenting a comparison between the standard use with strictly music-related contents and our new application of the format. The interface illustrated in this section represents the evolution of earlier software demos and working applications based on the IEEE 1599 format. It implements the functions and follows the guidelines detailed in [5]. However, till now such an interface has been used only for traditional CWN scores, and in this sense our approach is completely new.

Thanks to the standard, contents can be presented textually, aurally and visually in near real-time to maximize multimedia and multimodal enjoyment of music. In the upper part of Figure 3 an interface for pop songs encoded in IEEE 1599 is proposed. Heterogeneity in music contents is reflected by the layout of controls and views. Players, panels, floating windows or other devices are used to present multimedia contents in a unique framework. Different multimedia types are kept separated by using different controls, whereas objects of the same type are grouped within the same control. For instance, the part of the interface dedicated to audio/video contents contains the playlist of such media objects (dynamically loaded and syncrhonized from the IEEE 1599 file) and the common controls of a media player. Similarly, the panel dedicated to score images contains the list of scores, a control to select the pages of each

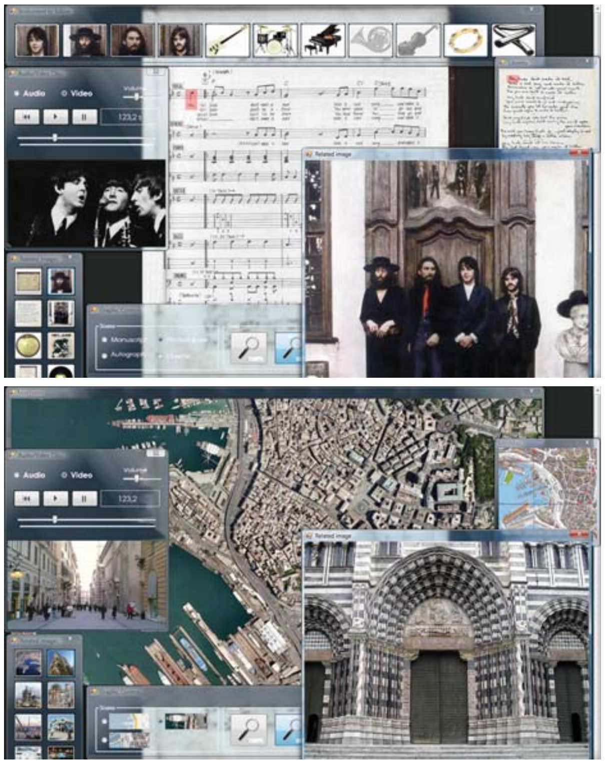


Figure 3. The interface for multi-layer browsing applied to a pop song and to city sounds.

score (once again dynamically loaded from the IEEE 1599 file) and a number of image-oriented navigation tools.

For the goals of this paper the interface has been adapted to the presentation of our material, as illustrated by the lower part of Figure 3. Multimedia and navigation controls can remain unalterated, as the key differences between a music application and this case are not due to a change in media types, rather to a change in the paradigm used to interpret their functions. For instance, now the custom media player loads an audio track of the route and the slider allows to go backward and forward in the audio/video material, whereas the main window (previously used as the “score” panel) contains one of the provided maps of the path itself. The selection tools, that originally have been designed to switch the current score page and music performance, still work in real-time to switch the current map and audio.

Moreover, the interface has been designed to allow the simultaneous enjoyment of all the views involved in the representation of the same piece. Please note that also non-temporized descriptions (e.g. the related files) are accessible. Related files often do not require synchronization, as they are ancillary representations in general not strictly referrable to music events. Usually, in music field this sub-element is employed to link on-stage photos, sketches, fashion plates, and so on. Our software application, on the contrary, takes full advantage of related files temporization by showing photos taken during the recording session. The result is a sort of slideshow that enriches the overall description of the route.

5 AN EXAMPLE: ENVIRONMENTAL SOUNDS ALONG A PEDESTRIAN ROUTE

In order to demonstrate the effectiveness of our approach, we have encoded in IEEE 1599 format the results of an experience made during the Sound and Music Computing conference held in Genoa last year (SMC 2008). In that occasion, we recorded the environmental sounds along a short pedestrian path, going from the cathedral (Piazza San Lorenzo, ① in Figure 4) to the harbour (Ponte degli Spinola, ⑥ in Figure 4). This route is about 0.5 km long and it takes about 7 minutes on foot. Sounds were completely acquired three times, namely during three different sessions, trying to respect the same temporization for each capture. We were interested in unveiling the similarities and differences that characterize city life during the phases of a day. To this end, we chose 1am, 9am and 6pm. As a result, we realized that:

- some audio events were quite similar and characteristic for a given place (e.g. the bells of San Lorenzo church), even disregarding time;
- other audio scenarios clearly identified a place or context, but during the day they suffered the consequences of variable human activities (e.g. at the harbour);
- finally, some environmental sounds occurred only in a track (e.g. the transit of an ambulance or the noise of children playing soccer), which adds descriptive richness but decreases the characterizing effect of the audio event over the environment.

Listening to the three mentioned audio tracks was an involving activity indeed, but these digital objects appeared to the listener as something unrelated. In other words, the rationale behind the experiment was clear, but many aspects of interest could have been unveiled only through a synchronization among tracks, the integration with other materials (texts, static images, videos, etc.) and the implementation of *ad hoc* navigation tools to jump from a media to another and to enjoy such a comprehensive description in a unique framework. From this perspective, many similarities emerged with the multi-layer fruition of music provided by IEEE 1599. In that very moment, the idea presented in this paper was born.

Thanks to the features explored in the previous sections, translating our pedestrian route into a city-map based score is easy. Music events identified in spine now become places of interest along the chosen path. The original composition is made of a sequence of music events, as well as a route is made of a sequence of places to visit. Music symbols have a space location over the score, say (x,y) coordinates in pixels, but this information can change from one printed version to another; similarly, the exact localization of places over a map depends on the graphical representation provided by the map itself. In Figure 4 graphical representations have been scaled to make comparisons easier, nonetheless this operation is not required for an IEEE 1599 encoding. However some slight differences are evident between the two upper maps and the lower one. Finally, the original sequence of music events can be translated into different temporized sequences during performances, and this originates a number of audio tracks; similarly, the transit

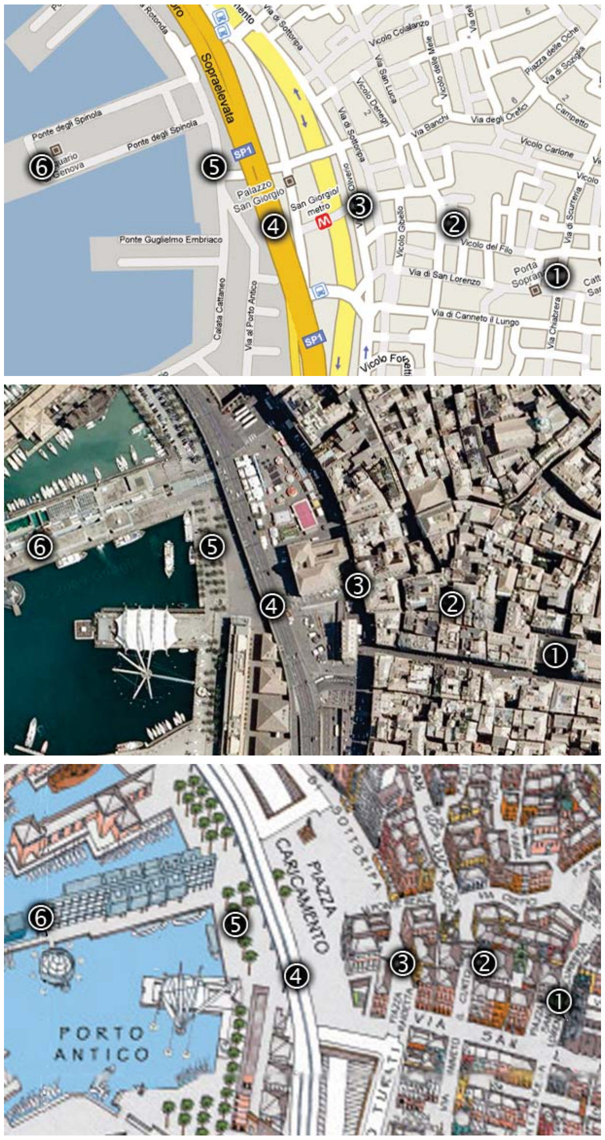


Figure 4. Three representations of the route.

across given map points in general occurs at different moments, and this aspect was captured by our audio recordings.

The points of interest along the route have been marked by numbers. In our path, ① denotes the start point, namely San Lorenzo church, ② identifies the intersection among narrow alleys of the centre, ③ is the crossing with a congestioned avenue, ④ denotes an underpass across the urban elevated motorway, ⑤ identifies the tracking pier for ferries, and ⑥ represents the destination, located at the old dock near the aquarium.

The resulting IEEE 1599 document contains 6 events, listed in the spine structure like in Figure 2. The `hpos` and `timing` attributes have a similar meaning, since the virtual localization in time and space does not refer to a score but to the mentioned pedestrian path. For instance, the relative spacing between each couple of places could be expressed in meters as well as in number of steps. Such events have been mapped within 3 graphical objects and 3 audio objects. For each point of interest, also static images with a text description have been inserted in the *General* layer, in order to generate a slideshow too.

Through this case study, we have proposed only a basic demonstration of the potentialities provided by the format in union with a browsing tool. In broader terms, this experience could be generalized to take into account a number of different scenarios and purposes. For instance, all the main touristic routes of a typical city visit could be represented and proposed in a Web interface to visitors. Another approach consists in encoding sounds not along a continuous path, but statically in a number of places of interest (from a historical, scientific, or other perspective), once again at different moments of either the day or the year. Furthermore, applications to artistic expression and multimedia art installation could emerge.

6 RELATED WORKS

Our work moves from previous experiences such as those cited in [6] and [7]. A number of projects have been carried out about sonification, environmental sounds recording and interaction with city soundscapes. In this sense, we have been explicitly inspired by the "Sons de Barcelona" project by the Grup de Recerca en Tecnologia Musical (MTG) of Universitat Pompeu Fabra - Barcelona. Another source of inspiration is the Freesound project, namely a collaborative database of Creative Commons licensed sounds uploadable and downloadable from the Web.

However, our approach is original as we propose an integrated interface to navigate continuously a map of environmental sounds. Besides, we have explored the use of a new XML-based standard format in order to provide an overall description of city soundscapes. Thanks to the features previously mentioned, IEEE 1599 in our opinion can be efficiently adopted as the format underlying other similar projects.

7 CONCLUSIONS

IEEE 1599 is an XML-based standard originally designed for music pieces. As demonstrated by this paper, the flexibility of the format allows to describe also not-strictly musical contents. We have applied such an encoding to the environmental sounds of a pedestrian route, and developed an application for the visualization and the interaction with multimedia contents. Such an experimental work can be extended in order to provide a virtual visit of an environment driven by a navigable soundtrack.

8 REFERENCES

[1] Baggi, D., "Technical Committee on Computer-Generated Music", *Computer*, vol. 28, no. 11, 1995, pp. 91-92.

[2] Haus, G. and Longari, M., "A Multi-Layered, Time-Based Music Description Approach Based on XML", *Computer Music Journal*, vol. 29, no. 1, 2005, pp. 70-85.

[3] Ludovico, L.A., "Key Concepts of the IEEE 1599 Standard", *Proceedings of the IEEE CS Conference The Use of Symbols To Represent Music And Multimedia Objects*, IEEE CS, Lugano, Switzerland, 2008.

[4] "IEEE Recommended Practice for Defining a Commonly Acceptable Musical Application Using XML", IEEE, 1599-2008, 2008.

[5] Baratè, A. and Ludovico, L.A., "Advanced interfaces for music enjoyment", *Proceedings of the working conference on Advanced visual interfaces*, ACM New York, NY, USA, 2008, pp. 421-424.

[6] Gaye, L. and Mazé, R. and Holmquist, L.E., "Sonic City: the urban environment as a musical interface", *Proceedings of the 2003 conference on New interfaces for musical expression*, National University of Singapore Singapore, Singapore, 2003, pp. 109-115.

[7] Kabisch, E. and Kuester, F. and Penny, S., "Sonic panoramas: experiments with interactive landscape image sonification", *Proceedings of the 2005 international conference on Augmented tele-existence*, ACM New York, NY, USA, 2005, pp. 156-163.

THE HYPER-KALIMBA: DEVELOPING AN AUGMENTED INSTRUMENT FROM A PERFORMER'S PERSPECTIVE

Fernando Rocha
Escola de Música,
Universidade Federal de Minas Gerais (UFMG)
fernandorocha@ufmg.br

Joseph Malloch
IDMIL and CIRMMT,
McGill University, Montreal, Canada
joseph.malloch@mcgill.ca

ABSTRACT

The paper describes the development of the hyper-kalimba, an augmented instrument created by the authors. This development was divided into several phases and was based on constant consideration of technology, performance and compositional issues. The basic goal was to extend the sound possibilities of the kalimba, without interfering with any of the original features of the instrument or with the performer's pre-existing skills. In this way performers were able to use all the traditional techniques previously developed, while learning and exploring all the new possibilities added to the instrument.

1 INTRODUCTION

The Hyper-kalimba is a Digital Musical Instrument created by the authors with the support of the Input Devices and Music Interaction Laboratory [1] at McGill University, directed by Prof. Marcelo Wanderley. It consists of a kalimba (a traditional African thumb piano) augmented by the use of sensors, which control various parameters of sound processing performed by custom software developed in Max/MSP. All the sounds produced are the result of real-time processing of the kalimba sound. The hyper-kalimba has been used in concerts since October 2007, both in improvisational contexts and in written pieces. There were several stages in the development of the instrument. Throughout this development, all the traditional kalimba techniques and sound possibilities were preserved. In each of the stages, some performance capabilities were added to the instrument. The mapping was then fixed for a certain amount of time, allowing the performer to learn the new techniques, before more possibilities were added.

[1] www.idmil.org

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

2 AUGMENTED INSTRUMENTS

Miranda and Wanderley indentify four main types of digital musical instruments, according to their resemblance to acoustic instruments: *augmented musical instruments*, *instrument-like gestural controllers*, *instrument-inspired gestural controllers*, and *alternate gestural controllers*[8]. Augmented musical instruments, also called hyper-instruments, are created by adding sensors and new performance possibilities to traditional, pre-existing musical instruments. Examples of augmented instruments include the hypercello created by Tod Machover [7] and, in the percussion area, a zarb with sensors developed at IRCAM by percussionist Roland Auzet [1].

Sensors added to an instrument can capture both gestures that are made to produce the normal sounds of the instrument (effective or instrumental gestures), as well as accompanying gestures that performers often make while playing[2]. New gestures can also be added to the performance technique. If new gestures are required of the performer, it is very important that they do not interfere too much with their existing playing technique. For example, most standing instrumentalists, like violinists, are not used to using foot pedals when playing, so this new gesture - pressing the pedal - can interfere with the performance[6]. The choice of sensors and the way they can be used and placed on the instrument require careful study. In developing the hyper-kalimba efforts were made to make use of existing instrumental and accompanying gestures from traditional kalimba technique, and only a few simple new gestures were used.

3 THE KALIMBA: CHARACTERISTICS AND LIMITATIONS

The kalimba is a modern development of the Mbira (a traditional African thumb piano). The instrument used in this project was the Hugh Tracey Alto Kalimba with pick-up. This alto model, created by the English ethnomusicologist Hugh Tracey, has 15 notes, corresponding to two octaves of a western diatonic G major scale (figure 1). It also has a built-in piezo contact microphone inside the wood body of the instrument.

The instrument is played with right and left thumbs pressing down pieces of metal of different lengths, which are called tines[3]. Each tine is tuned to one of the 15 notes. The right thumb usually plays the right side - tuned to thirds from G to G - and the left thumb plays the left side - tuned to thirds from A to F♯ (Figure 1). Two consecutive notes can be played with the same thumb, producing the interval of a third. Combining both thumbs can produce different intervals, but they are always restricted by which notes each thumb can reach. The range (2 octaves) and the tonality (G Major) are other limitations of the instrument. One effect that is possible to control in some acoustic kalimbas is *tremolo*. Some instruments have sound holes on the front and/or back part (figure 2). Covering and uncovering these holes (the front one with the thumb, the back ones with the third fingers) produces a tremolo effect. This is not acoustically possible in the kalimba used for this project, since it has a solid wooden body and no holes. The effect, however, can be imitated electronically, as will be described later.

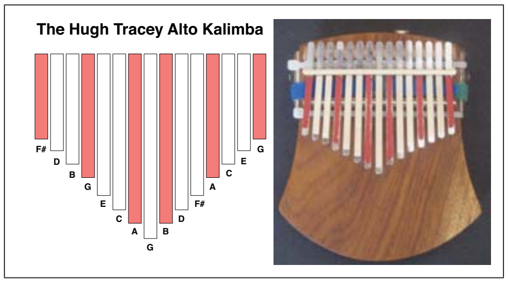


Figure 1. The Hugh Tracey Alto Kalimba.

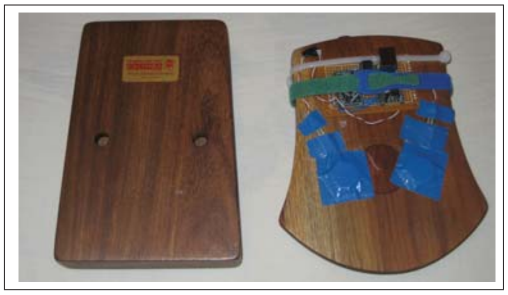


Figure 2. Back view of a kalimba (with two holes); Back view of the hyper-kalimba (with two pressure sensors).

4 EXTENDING THE INSTRUMENT

4.1 The first version (October 2007)

The first sensors added to the instrument were two pressure sensors. They were added to the back of the instrument, inspired by the holes that are present in some kalimbas. A kalimba performer is accustomed to using the third fingers of each hand to cover and uncover these holes. The use of this instrumental gesture was thus very natural.

The pressure sensors detect the amount of pressure applied to them. The first two sliders in the Max patch shown in figure 3 register this: the figure indicates that the right sensor is not being pressed (value=0) and the left sensor is being moderately pressed (value=300, on a scale from 0 to 1000). These two sensors were used to control pitch modulation and ring modulation effects, which can also be used to imitate a tremolo effect (as described later in section 4.2.1).

The piece *A la luna*, by Fernando Rocha and Ricardo Cortés, was written and performed for the instrument at this stage of development. *A la luna* is a structured improvisation for hyper-kalimba and pre-recorded and processed voice sounds. The structure of the piece is based on the poem "Noturno Esquematico" by Frederico Garcia Lorca. The words of the poem (recorded by Raquel Gorgojo) were manipulated electronically by Ricardo Cortés to create a rich sound texture, which is combined with the sounds produced in real time by the hyper-kalimba. The result is a dense atmosphere of sounds, in which the words of the poem are masked, appearing clearly only in the last section, when they are triggered by the notes F♯, E, D and C, produced by the kalimba. The pitch detection was made with the use of the Max/MSP port of Miller Puckette's `fiddle~` object.

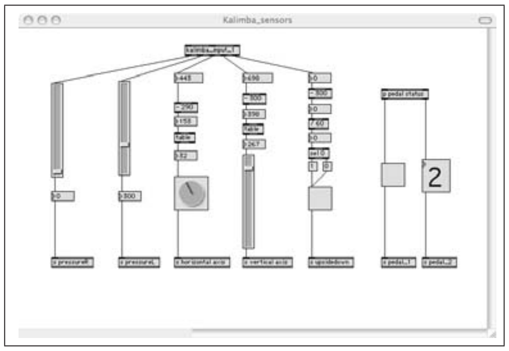


Figure 3. Hyper-kalimba Max/MSP patch showing input data from sensors.

4.2 The second version (February 2008)

The next sensor added to the instrument was a three-axis accelerometer, which enabled the measurement of the tilt of the instrument, both in the vertical and horizontal axis, as illustrated in Figure 3. The dial in the center shows the instrument's left-right tilt position (in this case the instrument is slightly tilted to the left). The next slider shows the instrument's front-back rotation, from a downward position of the front of the instrument to an upward one (in this example, the front of the instrument is in an upward position). Finally, the patch is able to recognize when the instrument is upside-down (since the box is not marked, in this case the instrument is not upside-down). Tilting the instrument is an accompanying gesture that is very common among kalimba players, and it does not affect the playing technique. Moreover, in many traditional African kalimbas, the front-back rotation is, indeed, an instrumental gesture. Some instruments present a metal ring placed around each tine (Figure 4). These ringers vibrate when the tine is vibrating, creating a buzz effect. Rotating the instrument controls the amount of the effect. By pointing the instrument down, the rings move close to the end of the tines (where they are fixed to the instrument). The vibration is then minimum, so it is the buzz effect. When the instrument is level and the ringers are in the middle part of the tines, the vibration is maximized and so is the effect[4].

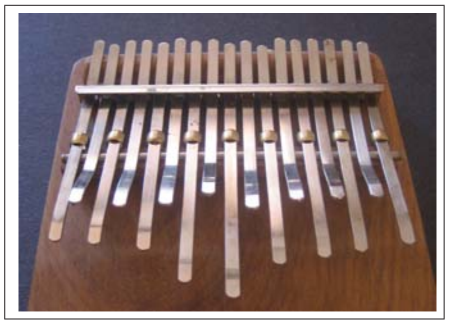


Figure 4. The Hugh Tracey Karimba with metal rings for buzz effect.

4.2.1 Mapping of the instrument at this stage

Pressure Sensor 1 (pressureR): controls a pitch transposition effect. The harder it is pressed, the greater the effect.
Pressure Sensor 2 (pressureL): controls a ring modulation effect. The pressure applied to the sensor determines the frequency used to modulate the sound of the kalimba. When the pressure is low (less than 300), this frequency is less than 8Hz, which creates an effect similar to a tremolo. Pressing harder causes the frequency to become higher and the effect is a change in the timbre of the instrument.
Position (horizontal axis): controls a multi-tap delay effect. When tilted to the left the delay is panned to the left; when tilted to the right, it is panned to the right.
Position (vertical axis): Pointing the front of the instrument down adds reverb; pointing it up adds very short delays, imitating the buzz effect found in traditional kalimbas. Maintaining an extreme upward position can generate a feedback effect.

The vertical axis also influences the pitch transposition process. Pointing the front of instrument down causes the transposition to go down. The further down the instrument is pointed, the larger the downward range of the transposition will be. Thus, the lowest pitch can be obtained when the instrument is pointing down and strong pressure is applied to the right pressure sensor. Conversely, the highest notes on the instrument can be obtained by pointing it up and strongly pressing the right sensor. When the instrument is level, however, the maximum range of the transposition is one half tone up. In fact, at this position any pressure (on pressureR) larger than 600 produces a half tone transposition. This is a gesture that can be repeated with accuracy by the performer as well as captured by the sensors. By using it, the performer is able to play chromatic passages.

Upside down: When the instrument is upside down, the loop being played (if there is one) is randomly altered in speed and volume. After 45 seconds the loop goes to normal speed and fixed volume. If the instrument remains upside down, the loop starts to fade out. In addition to these sensors, two Midi pedals were used in this mapping. The information coming from these pedals is shown on the right side of figure 3. Pedal 1 is an ON/OFF switch; and pedal 2 triggers one of 6 positions.

Pedal 1: When the pedal is pressed (ON), it freezes the value of all the sensors, so the parameters for sound processing remain fixed until the pedal is pressed again (OFF position). The only exception is the right pressure sensor, which is also fixed when the pedal is ON, but as soon as it is pressed for more than 100 milliseconds, it starts to work again, allowing for pitch transposition.

Pedal 2: controls the recording and playback of the audio loops used in the performance. When the pedal is pressed (status 1), the patch records the live sound until it is released (status 2). When the pedal is pressed (status 3) and then released (status 4) again, the loop is played in reverse at a speed that makes the length of the loop the same as the time interval between status 3 and 4. Status 5 (pressing the pedal) starts a fade out, and status 6 (releasing) ends the loop. The next time the pedal is pressed this cycle starts again.

The work *Improvisation for Hyper-kalimba* is a structured improvisation that explores all these possibilities. It also highlights some traditional characteristics of the kalimba and its repertoire: its melodic aspect and the use of ostinato, here transformed by electronic treatments. The first phrase of the piece, e.g., is played at a lower transposition (one octave down), and recorded. This recording is later looped as the performer plays variations of the same phrase, untransposed (i.e., one octave higher). The piece ends after the performer places the instrument upside-down in a stand. This gesture, clearly a conclusive one, triggers an erratic effect of altering the loop in volume and speed, followed by a 30 second fade-out.

4.3 The current version (from July 2008)

In the current version, the pedals were replaced by two digital buttons, one placed on the left and one on the right side of the instrument. The buttons have the same functions as the pedals. With the use of buttons all the performer's control over the sound processing is made directly on the instrument.

Some new features have been added to the instrument, using combinations of gestures. Pressing the left button (which corresponds to pedal 1) and the left pressure sensor together adds an extra sound to the one played by the performer. If the instrument is tilted down the added sound is an octave lower; if the instrument is level it is a fourth lower; finally, if the instrument is tilted up, the added sound is a fourth above. To stop this effect the performer has to repeat the same original gesture. All and any of the three sounds can be combined, creating the possibility of playing chords. The current mapping also allows for starting the loops of phrases at the same speed at which they were recorded; after recording the phrase, the performer should just press and release the right button very quickly (less than 400ms between pressing and release). These new possibilities have been used in several concerts, both in Brazil and Canada.

5 TECHNICAL DESCRIPTION

Since the kalimba model chosen for this project already contains a piezo-electric contact microphone, it was only necessary to add sensors - for measuring finger pressure and tilt - and two buttons. An Arduino Mini microcontroller board [2] was used for sensor data acquisition and data were communicated to the computer over USB. Two Interlink force-sensing resistors were used on the underside of the kalimba to sense finger pressure; the sensors were covered with a layer of closed-cell foam to give the performer some kinesthetic feedback as pressure is applied. An STMicro LIS3L02AS4 three-axis accelerometer was used to measure movement and tilt.

The audio signal and sensor data from the hyper-kalimba is sent to a laptop computer, where a custom-made Max/MSP patch is used to map the control parameters to sound processing. The mapping used in the instrument aims to preserve the melodic characteristic of the instrument, but also to create new sound possibilities.

6 DISCUSSION: TECHNOLOGY, COMPOSITION, AND PERFORMANCE

The development of the hyper-kalimba was based on careful consideration and balancing of technology, performance and compositional concerns. An example of these interrelations is the evolution of the control over the transposition effect. In the first mapping of the instrument, the transposition was controlled only by the right pressure sensor and was always upward. Strong pressure could create up to a four-octave transposition, producing a very different timbre, which was explored in *A la luna*. However, small transpositions, like a half tone, were difficult to achieve accurately. With the introduction of the 3-axis accelerometer, the angle of the instrument could then be mapped to control the range of the modulation. This new mapping made it considerably easier to play half tone transpositions and also allowed for a downward transposition. These new features were used in *Improvisation for Hyper-kalimba*.

In fact, as the instrument developed, new gestures were explored and could potentially be mapped. Mapping choices facilitated certain sounds or effects, and this had consequences in the musical material. Conversely, when specific sounds were sought in the improvisations or compositions, effective ways and gestures to create and control them had to be found. The use of instrumental and accompanying gestures that are characteristic to the instrument was prioritized, since new gestures might interfere with pre-existing technique. The active participation of an accomplished kalimba player was essential to understanding and utilizing the gestures to be used. The new gestures that have been added are very simple: pressing and releasing the two buttons. Another important aspect was considered in the mapping choices: gestures that could be controlled and captured accurately were mapped to musical results that needed to be more accurately controlled (e.g., control of a half tone pitch modulation). On the other hand, gestures that could not be controlled or captured very accurately were mapped to musical results that allowed some indeterminacy (e.g., amount and speed of delay effect).

The realization of any piece of music, independent of style, period or culture, depends not only on the possibility of producing sounds but also on the level of control over these sounds [5]. The potential for incorporating both traditional techniques and new features with a considerable level of control makes the hyper-kalimba very engaging for performers, composers and audience. The instrument has already been used in several concerts, both in improvisational contexts and in different written pieces.

7 CONCLUSION

The sensors and the patch provide the kalimba with new performance and sound possibilities, such as pitch bend, tremolo, extended range, and delay. All the sounds produced are the result of manipulation in real time of the kalimba sound. This helps to preserve the melodic characteristic of the instrument and its own characteristic voice. In addition, the sensors added to the instrument do not interfere with the traditional technique. Rather, they create new gestures that complement this technique. All the new possibilities were

[2] www.arduino.cc

implemented in different stages. While exploring a new vocabulary of gestures, the performer was able to explore new sound possibilities, and composers were able to know the instrument better. The instrument was played and tested in several concerts.

Currently, new features are being added to the instrument. The main idea is to keep introducing new possibilities without affecting the technique already developed. Different mappings can be created and are welcome, but having one fixed mapping allows the performer to develop a better control of the instrument, being able to explore all its possibilities and nuances, like a traditional acoustic instrument. In fact, when one learns to play an instrument, he or she always starts with the basic techniques before moving to extended techniques. In this sense, the process of developing the hyper-kalimba, divided in stages, is closely related to learning an acoustic instrument, and it takes advantage of all the performer's existing skills.

8 REFERENCES

[1] R. Auzet, "Gesture-following devices for percussionists," in *Trends in Gestural Control of Music*, M. M. Wanderley and M. Battier, Eds. IRCAM, 2000.

[2] C. Cadoz and M. M. Wanderley, "Gesture-music," in *Trends in Gestural Control of Music*, M. Wanderley and M. Battier, Eds. Paris, France: IRCAM, 2000, pp. 71–94.

[3] M. Holdaway, *Kalimba Fundamentals*. Tucson: Kalimba Magic, 2005.

[4] ——, *Playing the Hugh Tracey Karimba*. Tucson: Kalimba Magic, 2006.

[5] F. Iazzetta, "Interação, Interfaces e Instrumentos em Música Eletroacústica," in *Proceedings of the II 'IHC - Interação Humano-Computador' Conference*. Campinas: Unicamp, 1998.

[6] M. Kimura, "Creative process and performance practice of interactive computer music: a performer's tale," *Organised Sound*, vol. 8, no. 3, pp. 289–296, December 2003.

[7] T. Machover, "Hyperinstruments - A Progress Report 1987 - 1991," Massachusetts Institut of Technology, Tech. Rep., 1992.

[8] E. R. Miranda and M. M. Wanderley, *New Digital Instruments: Control and Interaction Beyond the Keyboard*. Middleton, Wisconsin: A-R Publications, 2006.

REAL-TIME DTW-BASED GESTURE RECOGNITION EXTERNAL OBJECT FOR MAX/MSP AND PUREDATA

Frédéric Bettens
Faculty of Engineering (FPMs) - TCTS Lab
7000 Mons, BELGIUM
frederic.bettens@fpms.ac.be

Todor Todoroff
ARTeM
1030 Bruxelles, BELGIUM
Faculty of Engineering (FPMs) - TCTS Lab
7000 Mons, BELGIUM
todor.todoroff@skynet.be

ABSTRACT

This paper focuses on a real-time Max/MSP implementation of a gesture recognition tool based on Dynamic Time Warping (DTW). We present an original ”multi-grid” DTW algorithm, that does not require prior segmentation. The num.dtw object will be downloadable on the `numediart` website both for Max/MSP and for Pure Data. Though this research was conducted in the framework described below, with wearable sensors, we believe it could be useful in many other contexts. We are for instance starting a new project where we will evaluate our DTW object on video tracking data as well as on a combination of video tracking and wearable sensors data.

1 INTRODUCTION

The ”Dancing Viola” project, described in more details in [7], was led at the Faculté Polytechnique de Mons within the `numediart`program and is linked to viola player Dominica Eyckmans. It covers some of the aspects of the long-term project ”Extension du corps sonore” launched by Musiques Nouvelles, a contemporary music ensemble in Mons, that aims at giving intrumental music performers an extended control over the sound of their instrument. The intention is to extend the understanding of the sound body from the instrument only to the combination of the instrument and the whole body of the performer. Whereas usual augmented instruments designs track the gestures used to play the instrument to expand its possibilities, this specific project focuses on using non-musical gestures to transform the sound of the instrument. Our approach is dictated by the nature of Dominica’s project: she is actually dancing while playing the viola and we track her dancing movements rather than her hands movements. But the recognition algorithm we present here is not limited in any way by this specific context, as we have successfully demonstrated using a database of hands gestures measured with sensors placed on the hands. Gesture recognition is a welcome addition to an interactive performance and can be used to trigger events, to adapt the response of the virtual instruments according to the detected gestures, or to move through the various steps of a performance. As other modules like hit detection, mapping, interpolation (also developed within the ”Dancing Viola” project and described in [8]) or sound synthesis and transformations, must be running simultaneously on the same computer, it is essential to minimize the computational load. This Max/MSP object is being integrated in the ARTeM software framework for the concerts with Dominica Eyckmans, as well as for other artistic works.

SMC 2009, July 23-25, Porto, Portugal

While using similar hardware (cf. 2.1), the atomic gesture recognition algorithm developed by Benbasat and Paradiso [1] is not suitable in our project: as dance movements are usually chained without pauses and cannot be decomposed in a concatenation of elementary movements along one accelerometer axis only, we have to consider an algorithm that can deal with unconstrained fluid motions, without the knowledge of the start and end of a gesture.

As for Automatic Speech Recognition (ASR) applications, the most popular algorithms used for gesture recognition are Dynamic Time Warping (DTW) [5, 4] and Hidden Markov Models (HMMs) [2]. In our framework, the aim is to develop a user-dependent recognition system with a small gesture vocabulary and a database of limited size. As some gestures should be added, removed, enabled, or disabled easily and quickly, without any training procedure, we chose for the DTW algorithm, which we adapted to make it usable in real-time without the need for segmentation.

This report is divided in following sections: after a brief description of the system, we present the gesture recognition module, by describing our ”multi-grid” DTW algorithm and its real-time Max/MSP implementation, as well as some preliminary results, and we conclude with future investigations.

2 SYSTEM

2.1 Sensors

The sensor system allows for the data of two sensors (each a combination of a 3-axes accelerometer and a 2-axes gyroscope), placed on both ankles of the performer, to be transmitted every 8ms wirelessly over Wi-Fi. More details on the sensors can be found in [7]. The placement on the ankles presents a minimal hinderance even for movements on the ground. Depending on the results of further experimentation with the new software tools we will consider the need, the type, and the placing of additional sensors on Dominica's body.

2.2 Software framework

The ARTeM software, developed inside the Max/MSP environment to map sensor data to parameters of various sound transformation algorithms, is organized around a modular concept: the audio paths of the various virtual instruments are connected through a matrix, with external inputs and outputs of virtual instruments injected from the top and redirected with selectable level, to the inputs of the virtual instruments and the external sound outputs.

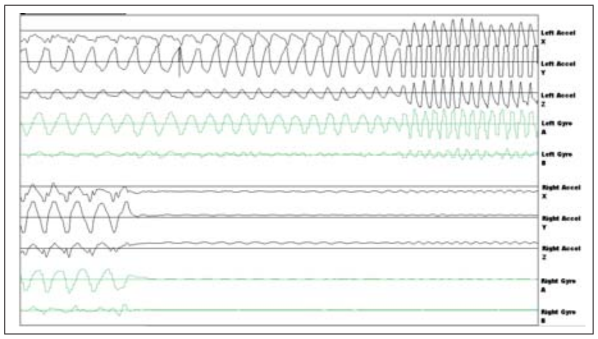


Figure 1. A data recording: 3-axes accelerometer (black) and 2-axes gyroscopic data (green) for left and right ankles.

The sensors data (Figure 1) are received as UDP packets through the normal Wi-Fi interface. An external decodes the custom protocol, scales the raw data and defines a name space depending on configuration messages sent to its input and outputs data as messages. All samples are then made available through a send/receive scheme throughout all the patches.

3 GESTURE RECOGNITION

3.1 DTW algorithm

The classical DTW algorithm uses Dynamic Programming (DP) principles to determine the best nonlinear mapping (Figure 2) between the temporal indices of the test sequence $(i = 1..I)$ and those of the reference sequence $(j = 1..J)$, assuming that both these sequences have been segmented. We denote by $d(i, j)$ the (non-negative) "local distance" (or dissimilarity measure) between the test frame T_i and the reference frame R_j (where a frame is composed of the data of all sensors and axes at a given time), and by $D(i, j)$ the "accumulated distance" along the sub-path between the origin and the current node (i, j). The algorithm aims at minimizing these accumulated distance values and/or at extracting the associated best path (i.e., the sequence of nodes) in the DTW grid (Figure 2). A classical way of computing the accumulated distance value $D(i, j)$ along a sequence of nodes (i_k, j_k) $(k = 1..K)$ consists in weighting the local distance elements $d(i_k, j_k)$ with transition costs that depend on the predecessor (i_{k-1}, j_{k-1}), and summing up the weighted values:

$$D(i, j) = \sum_{k=1}^{K} W(i_k, j_k; i_{k-1}, j_{k-1})\, d(i_k, j_k) \qquad (1)$$

These transition costs raise the issue of normalization when computing paths of different lengths (e.g. when a test gesture is compared with several reference gestures of unequal duration). Dividing the optimal distance by the "path length" (i.e. the sum of all weights along the path) leads to the mathematical expression of an average "cost per node" and, using the following symmetric transition cost type [6]:

$$W_k = (i_k - i_{k-1}) + (j_k - j_{k-1}) \qquad (2)$$

the normalization factor $(I + J)$ is path-independent. The question of the weight of the local distance corresponding to the first node is solved by computing the transition cost between a "fictitious" original node $(0, 0)$ and the first node.

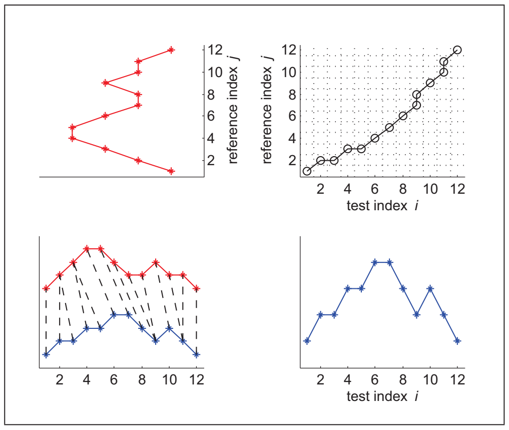


Figure 2. Mapping between two time series and DTW grid.

3.1.1 DTW search constraints

- Monotonicity and strict endpoint constraint. In its strictest form, any candidate path must not only be monotonic, meaning that $i_{k-1} \leq i_k$ and $j_{k-1} \leq j_k$, but also begin at $(1, 1)$ and end at (I, J) exactly.

- Global path constraints. Itakura [3] suggests the specification of the maximum allowable compression and expansion factors ($\lambda_{max} \geq 1$ and $\lambda_{min} \leq 1$, with e.g. $\lambda_{min} = 1/\lambda_{max}$), whereby all paths must entirely lie within a parallelogram (Figure 3a). Another global constraint, proposed by Sakoe and Chiba [6], requires that the paths lie within a simple strip around a purely linear path: $|j_k - i_k| \leq R$, where R is the ”window width” (Figure 3c).

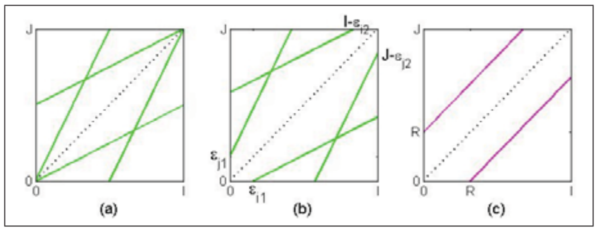


Figure 3. Global constraints: (a) Itakura, (b) Itakura (relaxed), (c) Sakoe and Chiba

- Local path constraints. The expansion or compression ratio between test and reference can also be limited locally, in the neighbourhood of each node. These local constraints are usually defined by listing the legal transitions. Equations 3 and 4 show the local path constraint implemented, where each node (i_k, j_k) can be reached from three different sets of predecessors (Figure 4):

$$D(i_k, j_k) \; = \; min(\; D_1, \; D_2, \; D_3) \qquad (3)$$

with:

$$\begin{aligned} D_1 &= D(i_k - 1, j_k - 2) + 2d(i_k, j_k - 1) + d(i_k, j_k) \\ D_2 &= D(i_k - 2, j_k - 1) + 2d(i_k - 1, j_k) + d(i_k, j_k) \\ D_3 &= D(i_k - 1, j_k - 1) + 2d(i_k, j_k) \end{aligned} \qquad (4)$$

- Relaxed endpoint constraint. To address the issue of locating accurately and in real-time the endpoints of a test sequence, the constraints are relaxed by permitting the path to start from one of the following nodes: $(1, 1)$ to $(1+\epsilon_{i_1}, 1)$, or $(1, 1)$ to $(1, 1+\epsilon_{j_1})$, and to end at one of the following nodes: $(I - \epsilon_{i_2}, J)$ to (I, J), or $(I, J - \epsilon_{j_2})$ to (I, J) (Figure 3b). Consequently, the different paths associated to each of the candidate terminal nodes are compared on the basis of their normalized accumulated distances, where the global normalization factor $(i_K + j_K)$ is determined by the final coordinates only.

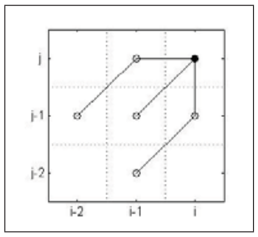


Figure 4. Local constraints

When only the starting point is approximately known, lower and upper bounds of the other endpoint may be found: e.g. $I_{min} = J/2$ and $I_{max} = 2J$, when the expansion/compression ratio lies in the range between 1/2 and 2 (if we neglect ϵ_{i_1} and ϵ_{j_1} values). In this context, since the ending point is *a priori* almost unknown, we decide to remove the margin parameters ϵ_{i_2} and ϵ_{j_2}, as well as to remove the global constraints that were linked to that ending point (i.e. two straight lines in Figure 3b). Finally, the gesture is restricted to end somewhere between the bounds I_{min} and I_{max} along the i axis, and strictly at J along the j axis (Figure 5). In other words, the warping consists in aligning the whole reference sequence with a test sequence (or subsequence) that may be up to twice as long or twice as short.

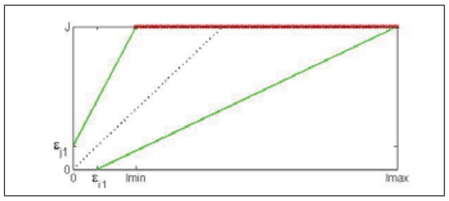


Figure 5. Final global constraints (with $R = \infty$) and set of admissible ending points

Figure 6 shows an example of local (left) and normalized accumulated (right) distance matrices for similar (top) and different (bottom) gestures, with following parameter values: $\epsilon_{i_1} = 8$, $\epsilon_{j_1} = 0$, $\lambda_{min} = 0.5$, $\lambda_{max} = 2$, and $R = \infty$. Low distance values (depicted by blue pixels) are obtained when comparing similar gestures. Conversely, high dissimilarities are observed when the tested gesture is very different from the reference one, resulting in a worse matching score.

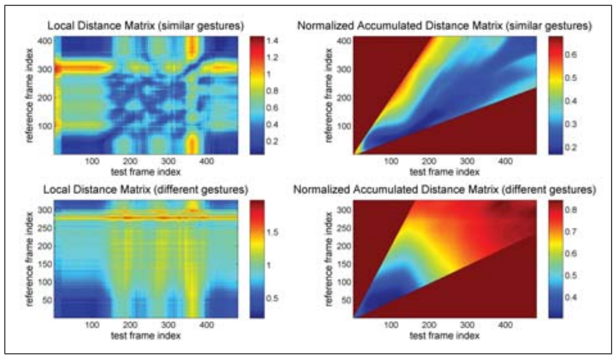


Figure 6. Local (left) and Normalized Accumulated (right) Distance Matrices for similar (top) and different (bottom) gestures

3.1.2 "Multi-grid" DTW algorithm and real-time Max/MSP implementation

Only few implementations of DTW do not require prior segmentation. Oka [5] presents a continuous DP algorithm, which is an efficient real-time method as the different paths originating from all possible starting points are simultaneously competing in the same DTW grid (one per reference gesture). However, he does not explain how to include global constraints. On the other hand, Ko [4] describes a method including these constraints, but at the cost of a higher computational load, as whole new paths are calculated from each new starting point (i.e. at every time instant) in the accumulated distance matrix (for each reference gesture).

Our "multi-grid" DTW algorithm provides a compromise solution. The method uses simultaneously a set of shifted DTW grids, each one hypothesizing another starting point (or set of consecutive starting point candidates when $\epsilon_{i1} \neq 0$) for the test sequence. The time shift between two successive DTW grids will generally be equal to $hop\_size = 1 + \epsilon_{i1}$. The number of simultaneously active grids can be limited to the following quantity: $S_{max} = ceil(I_{max}/hop\_size)$. As J may vary from one gesture to the other, I_{max} and S_{max} are also depending on the specific reference gesture. At every time instant, one best score (possibly "infinite" at the beginning) is computed in each shifted grid, and the minimum value of all these normalized accumulated distances is assigned to the given reference gesture. Despite the computation of several shifted grids, a low complexity can be achieved via an iterative implementation (like in [4]), where only one partial column $D(i, j)$ is evaluated in each grid at a given time i (for each reference gesture), instead of all (partial) preceding columns from the starting point.

Figure 7 illustrates normalized accumulated distance matrices for successive shifted DTW grids when test and reference gestures are similar. A good matching score is obtained for the low shift values, while it becomes worse when the delay increases.

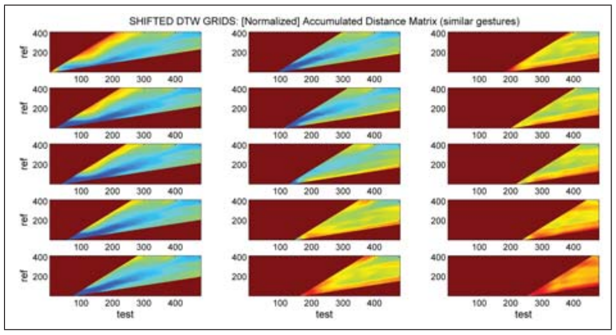


Figure 7. Normalized Accumulated Distance Matrices for successive shifted DTW grids (similar gestures)

Figure 8 also illustrates normalized accumulated distance matrices for successive shifted DTW grids, but when test and reference gestures are different. Again, the matching scores obtained in this latter figure are worse than the scores obtained in the former one.

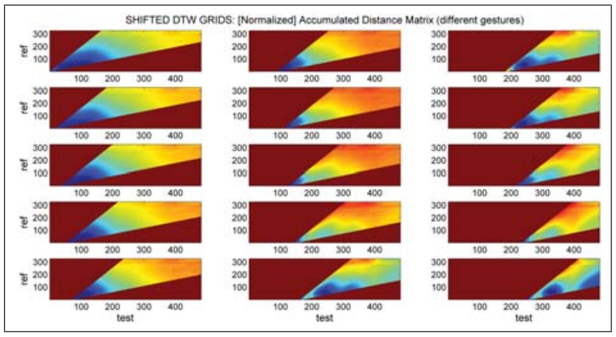


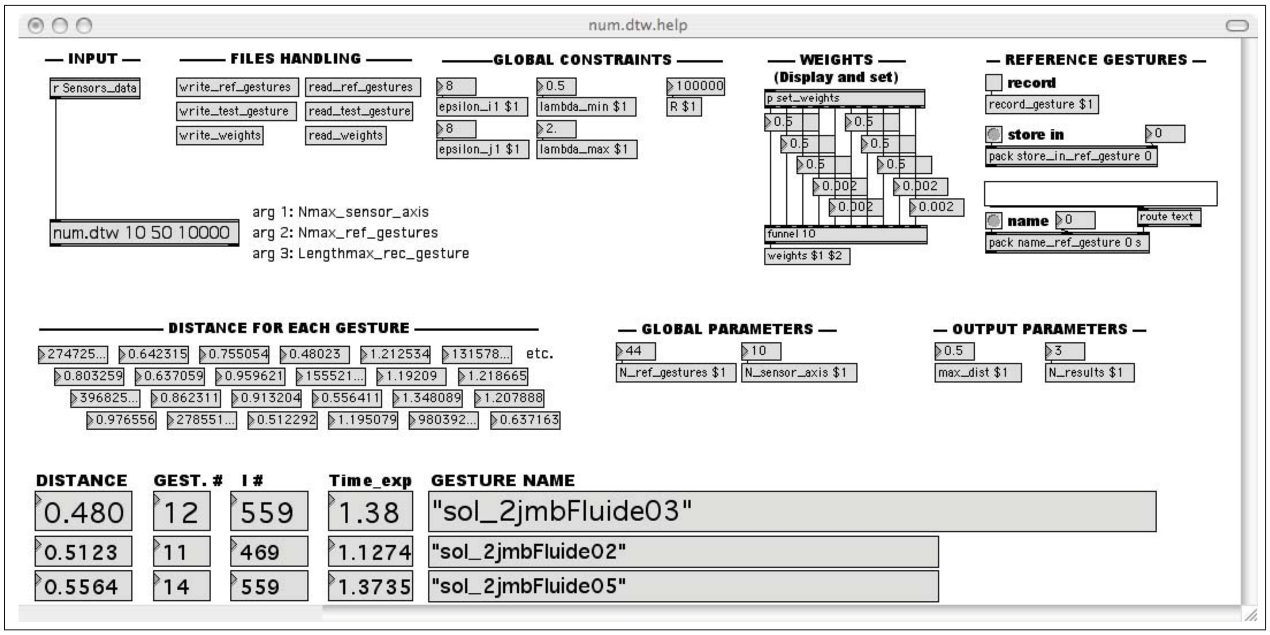
Figure 8. Normalized Accumulated Distance Matrices for successive shifted DTW grids (different gestures)

Finally, the overall gesture recognition module has been implemented as a Max/MSP external (see Figure 9), which includes the "multi-grid" DTW algorithm, as well as the pre- and post-processing stages described hereafter. It also evaluates and displays the time compression/expansion ratio, providing feedback to the artist (e.g. during rehearsals).

3.2 Pre-processing and distance metrics

The pre-processing of the sensors data and the calculation of the local distances are not part of the DTW algorithm itself, but their computation is a preliminary stage, briefly explained in this subsection.

The current version of our system implements a downsampling stage (with a factor 4), preceded by a lowpass filtering step, and uses the L_1-distance, whose computation is



arg 1: Nmax_sensor_axis
arg 2: Nmax_ref_gestures
arg 3: Lengthmax_rec_gesture

8, 0.5, 100000, 8, 2.

0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.002, 0.002, 0.002, 0.002

0, 0

274725... 0.642315 0.755054 0.48023 1.212534 131578... etc.
0.803259 0.637059 0.959621 155521... 1.19209 1.218665
396825... 0.862311 0.913204 0.556411 1.348089 1.207888
0.976556 278551... 0.512292 1.195079 980392... 0.637163

44, 10, 0.5, 3

DISTANCE	GEST. #	I #	Time_exp	GESTURE NAME
0.480	12	559	1.38	"sol_2jmbFluide03"
0.5123	11	469	1.1274	"sol_2jmbFluide02"
0.5564	14	559	1.3735	"sol_2jmbFluide05"

Figure 9. num.dtw Max/MSP external

very efficient as its expression is made of a (weighted) sum of the absolute value of differences. During this calculation, the sensors data are weighted, as some of them are varying within completely different ranges of values and expressed in different units (e.g. accelerometer data $\pm 2g$ and angular velocity $\pm 500°/s$). The easiest way consists in normalizing the samples axis per axis (e.g. dividing them by 2 and 500, respectively).

3.3 Post-processing

In the current Max/MSP implementation, the post-processing consists in selecting, at each moment, the gesture with the lowest normalized accumulated distance and validating its recognition if this value is below a user-defined global threshold.

3.4 Preliminary results

We first tested our ”multi-grid” DTW algorithm offline, on a small database composed of recordings of 44 isolated dance gestures (with a sampling period of $8ms$). Each individual unsegmented test gesture was compared with each segmented reference gesture.

As a result of all these pair-wise comparisons, we obtained a ”pseudo confusion matrix” (Figure 10), the small amount of recorded data preventing us from deriving actual statistics. However, one can see that the main diagonal is in blue colour, because each gesture is very similar to itself, and the blocks of blue pixels are explained by the presence

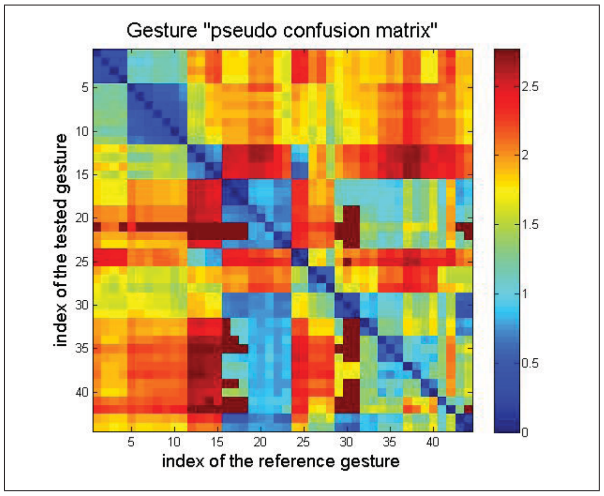


Figure 10. Gesture ”pseudo confusion matrix”

of several occurences of the same gesture in our database. This representation allowed us to examine the ambiguity between some different pre-defined gestures and to get information about an appropriate fixed global threshold or a series of gesture-based threshold values.

Our DTW algorithm was also used in a second application. The sensors were attached to the wrists of the second author and a dozen of left and/or right arm movements were successfully recognized in real-time. The post-processing was slightly modified into an N-best strategy ($N = 3$), that is, displaying continuously the three best matched gestures. However, the correct gesture was always classified in first position, except when the execution was too fast (e.g. more than two times faster, while a factor 2 was the maximum fixed by local and global constraints).

4 CONCLUSION AND FUTURE WORK

A real-time DTW-based gesture recognition tool has been developed, with a great flexibility provided by its set of parameters (minimum and maximum expansion and compression ratios, ”window width”, sensor axes weights, user-defined global threshold, etc.) and it has been successfully tested on two different small databases. We are finalizing the port of the external to Pd.

Algorithmic improvements include the addition of other local constraints types (only equation 4 is implemented now) and the ability to activate and/or deactivate specific reference gestures on the fly.

Some investigations are worth trying as far as the preprocessing is concerned: e.g. removing the gravity component to derive tilt-invariant features, testing different levels of downsampling, applying nonlinear quantification, etc. Some work could also be accomplished to improve post-processing: the single global distance threshold might be replaced by gesture-dependent threshold values and the measured time expansion/compression ratio could be taken into account.

5 ACKNOWLEDGMENTS

Research supported by `numediart`, a long-term research program centered on Digital Media Arts, funded by Région Wallonne, Belgium (grant N°716631).

6 REFERENCES

[1] A. Y. Benbasat and J. A. Paradiso, “An inertial measurement framework for gesture recognition and applications,” in *Gesture Workshop*, 2001, pp. 9–20.

[2] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy, “Wireless sensor interface and gesture-follower for music pedagogy,” in *Proc. NIME ’07*. New York, NY, USA: ACM, 2007, pp. 124–129.

[3] F. Itakura, “Minimum prediction residual principle applied to speech recognition,” vol. 23, 1975, pp. 67–72.

[4] M. H. Ko, G. West, S. Venkatesh, and M. Kumar, “Using dynamic time warping for online temporal fusion in multisensor systems,” vol. 9, no. 3. Elsevier Science Publishers B. V., 2008, pp. 370–388.

[5] R. Oka, “Spotting method for classification of real world data,” vol. 41, no. 8, 1998, pp. 559–565.

[6] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” in *IEEE Trans. ASSP*, vol. 26, 1978, pp. 43–49.

[7] T. Todoroff, F. Bettens, W.-Y. Chu, and L. Reboursière, “Extension du corps sonore - dancing viola,” in *Proc. NIME ’09*, Pittsburgh, Pennsylvania, USA, 2009, pp. xx–xx.

[8] T. Todoroff and L. Reboursière, “1-d, 2-d and 3-d interpolation tools for max/msp/jitter,” in *Proc. ICMC ’09*, Montreal, Quebec, Canada, 2009, pp. xx–xx.

MOTION-ENABLED LIVE ELECTRONICS

Gerhard Eckel, David Pirrò, Gerriet K. Sharma
University of Music and Performing Arts Graz
Institute of Electronic Music and Acoustics
{eckel, pirro, sharma}@iem.at

ABSTRACT

Motion-Enabled Live Electronics (MELE) is a special approach towards live electronic music aiming at increasing the degree of the performers' embodiment in shaping the sound processing. This approach is characterized by the combination of a high-resolution and fully-3D motion tracking system with a tracking data processing system tailored towards articulating the relationship between bodily movement and sound processing. The artistic motivations driving the MELE approach are described, an overview of related work is given and the technical setup used in a workshop exploring the approach is introduced. Brief descriptions of the pieces realized in the workshop and performed in the final concert inform the presentation of the conclusions drawn from the workshop.

1 INTRODUCTION

This paper describes a workshop exploring a particular approach towards live electronic music aiming at increasing the degree of the performers' embodiment in directing or shaping the sound processing. The workshop took place in the context of the *impuls 2009*[1] international ensemble and composers academy for contemporary music held biannually at the University of Music and Performing Arts Graz (KUG). Six composers and six performers from Europe, North America and Japan participated in the one-week workshop entitled *Motion-Enabled Live Electronics* held in the CUBE performance space [1] at the Institute of Electronic Music and Acoustics (IEM). The six pieces prepared for – and further developed during – the workshop were presented in the CUBE in a concert entitled *Enacted Electronics*. This concert took place on February 22nd 2009 and was transmitted via multi-channel Internet streaming to two remote locations in the context of the *CO-ME-DI-A*[2] project, one of which in Graz (MKL at Kunsthaus) and the other one in Paris (IRCAM). In this paper we explain what motivates our approach, we describe the system developed to realize it and we report about the experiences made in the workshop and the concert.

[1] c.f. http://www.impuls.cc/, accessed 2009/04/12
[2] c.f. http://www.comedia.eu.org, accessed 2009/04/08

SMC 2009, July 23-25, Porto, Portugal

2 MOTIVATIONS

Motion-Enabled Live Electronics (MELE) uses state-of-the-art motion tracking[3] of the performers' instruments or body parts (e.g. head, arm, or wrist) to inform the sound processing and projection. There are several motivations for the MELE approach.

In a typical live electronics concert, where the sound of the instruments is picked up with microphones to be processed and then projected via loudspeakers, the performers on stage may either resort to simple interfaces such as pedals and switches to control the processing or the live electronics are controlled by an additional operator off-stage. MELE was developed with the aim of providing performers with autonomous and intuitive control of the live electronics. MELE insures intuitiveness by unobtrusive bodily control (no need for physical interfaces other than tracking markers) and autonomy through independence from additional operators. These will still be needed to ensure optimal sound pickup and projection, but they will be less concerned with actually performing the live electronics. Performing should be in the hands of the performers as much as possible, for allowing them to fully identify with their very role, especially with respect to the live electronics.

In playing a musical instrument, the performer's body typically extends into this instrument – the instrument becoming part of the performer's body schema (in the sense defined in [2]). In order for the performers' bodies to extend into the sound processing and projection as much as they usually do into their instruments, a more bodily access to live electronics is in need. This is difficult to achieve with standard controllers because of the low dimensionality and low spatial or temporal resolution of their control spaces. In using refined motion-based interfaces for live electronics, sound processing and projection may rather be *enacted* than *operated* by the musicians, which is one of the goals of our approach.

[3] c.f. http://www.vicon.com, accessed 2009/04/08

From a composer's point of view, being able to make use of the location and orientation of the performers' instruments or body parts to inform the sound processing opens up completely new perspectives of conceiving the stage space as an interface. For instance, this space may be structured as a parameter space in which the performers navigate by moving about the stage. Or the relative distances and orientations of the musicians may inform the live electronics such as to include socio-spatial and psycho-spatial aspects of performance. One may think of the performers to *inhabit* a composed virtual stage space. Of course, all these aspects assume employing mobile instruments that may be carried around by the performers while playing.

Through detailed motion tracking of the musicians or their instruments, accompanist or ancillary gestures [4] may be harnessed ("instrumentalized") to shape the transformation and spatialisation of the instrumental sound. Besides their effect as "expressive movements", these gestures also color the sound of the instrument as a consequence of the movement, which results into a varying acoustic excitation of the performance space [4]. Therefore, when used for controlling the spatialisation of the processed instrumental sound, the audible effect of these gestures will be recognized as highly familiar by the audience, heightening their empathy with the performer and enhancing their immersion in the performance.

3 RELATED WORK

As described in the section 4, the MELE approach and setup are rather particular. Therefore there is not so much other work directly related to MELE. Of course, various kinds of motion tracking have been used to control sound processing and synthesis (e.g. with the EyesWeb system [5] or with VNS [5] used to create what Winkler calls "motion-sensing music" [6]), but very rarely high-definition systems such as the one employed in MELE have been used in stage performances. This is due to the low availability of such technology and the complexity of using it in live performance. It was one of the objectives of the MELE workshop, to show that such systems can be used successfully in a concert situation. Nevertheless, MELE shares many aspects with work in the field of gestural control of sound synthesis, processing and spatialisation, which has received wide attention in the field of sound and music computing during the last decade (e.g.[7] and [8]). MELE is closely related to tracking-based approaches such as TrakHue [9], where the live electronics are controlled via body motion. Recently a special focus on questions concerning gesture controlled spatialisation can be noticed (e.g. the work of Marshall et al. [10] and

[4] "those gestures that are part of a performance, but that are not produced for the purpose of sound generation" [3])

[5] c.f. http://homepage.mac.com/davidrokeby/vns.html, accessed 2009/04/10

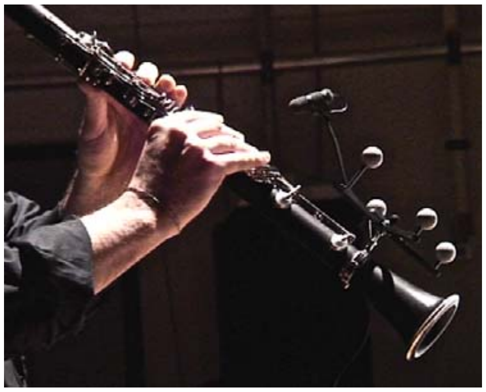


Figure 1. A 4-marker tracking target mounted on a clarinet using a Marschgabel fitting (and microphone attached)

Schacher [11]). Although MELE has also been used to control sound spatialisation, its approach is more holistic in the sense that sound processing and spatialisation are considered in common – as being *inseparable aspects of live electronics*. One of the objectives of MELE is to offers a framework to treat both aspects in concert. In most of the work related to MELE, the sound is produced or transformed by the motion of the audience (such as in installation situations, e.g. using a system like VNS) or by dance performers (e.g. [6]). The project that had the biggest influence on MELE is the Embodied Generative Music (EGM) project [6], which also supplied much of the technological infrastructure described next.

4 THE MELE SETUP

This section describes the features of the setup that has been proposed by the workshop organizers to the participants. These features have been communicated to the composers before they developed their pieces for the MELE setup. The preparation of these pieces has been followed closely by the workshop organizers prior to the workshop, so composers arrived at the workshop with almost finished scores or at least clearly defined concepts.

The setup was determined by the studio and performance space in which the workshop and the concert took place. This space is equipped with a 24-channel hemispherical loudspeaker array optimized for Ambisonics spatialisation and a video motion tracking system [7]. The setup constrained the stage space to a circular region in the center with a diameter of about 6 meters. The audience was seated in a circle around the stage. The stage space was fully covered by the tracking system, allowing for relatively large movements of up to three musicians to be tracked. As the loudspeakers

[6] c.f. http://embodiedgenerativemusic.org, accessed 2009/04/10

[7] composed of 15 M2 cameras and a V624 data station by Vicon

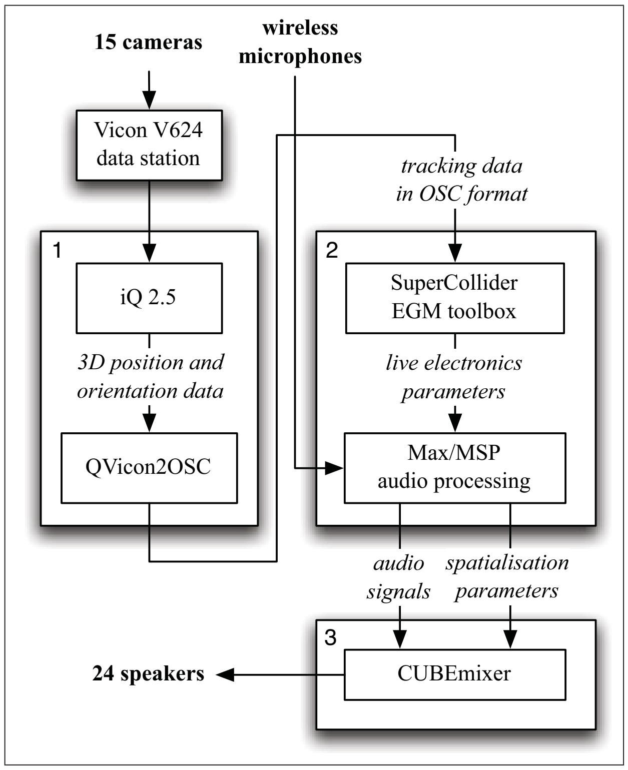

15 cameras
wireless microphones
Vicon V624 data station
tracking data in OSC format
1
iQ 2.5
3D position and orientation data
QVicon2OSC
2
SuperCollider EGM toolbox
live electronics parameters
Max/MSP audio processing
audio signals
spatialisation parameters
3
24 speakers
CUBEmixer

Figure 2. The MELE setup, signal and data flow

were located behind the audience, a very intimate situation arose for performance and sound spatialisation – the musicians and the audience actually sharing the same acoustic and visually unoriented space.

The musicians were equipped with wireless microphones and tracking markers, either mounted on their instruments (figure 1) or worn on their arms (figure 4) or heads (figure 6). The mounting solutions were developed with the musicians prior to the workshop and were also determined by the way the musicians' motions were used in the pieces. An important requirement for the solutions adopted was that they should not interfere in any way with the musicians' normal playing and moving about the stage.

The computer infrastructure of the MELE setup consists of 3 machines, one dedicated to the tracking, a second one to tracking data and audio signal processing and a third one to the spatialisation (figure 2). The tracking system was controlled with the iQ2.5 software by Vicon. The tracking data was translated to OSC with the utility QVicon2OSC[8] and the Ambisonics spatialisation was realized with the Pd application CUBEmixer [12]. The tracking data was processed using a specialized toolkit implemented in SuperCollider (the EGM toolkit) and developed in the context of the EGM project. Sound processing was realized with Max/MSP or SuperCollider – depending in the composers preferences. All communication between the mentioned programs was realized via OSC.

[8] c.f. http://sonenvir.at/downloads/qvicon2osc, accessed 2009/04/10

Figure 3. Video still of Annegret Mayer-Lindenberg (viola), Jason Alder (clarinet), and Dana Jessen (bassoon) playing Jesse Broekman's piece *Langs Rafels*

The particularity of the MELE setup can be seen in the combination of a high-resolution fully-3D tracking system with a tracking data processing system tailored towards applications articulating the relationship between bodily movement and sound processing – the EGM toolkit. The tracking system ensures a very high spatial (below 1 mm) and temporal (120 Hz) resolution, a low overall latency (about 20 ms from movement to sound), and a large tracking volume (more than 60 m^3). The EGM toolkit provides modules for data conditioning (e.g. geometrical transformations, scaling, filtering, clipping), feature extraction (e.g. speed, acceleration, periodicity analysis, relative distances and orientations), and physical modeling for the specification of the dynamics of virtual instruments (e.g. mass spring systems, potential energy surfaces). These physical models are typically used for generating synthesis and processing control parameters rather than sound.

5 WORKSHOP

Every composer used the MELE tool in his or her own way and developed an individual approach and concept of composing within the setup. In every piece a different idea for linking bodily movement and sound was investigated. In some cases a collective instrument was built that established a performative relationship between musicians and composer while in other pieces a "scene" was set up where spatial relations between the musicians were established. The performers shaped and spatialised their own or each others' processed sounds. Thus the idea of influencing sound through individual physicality was in some compositions expanded to a collective behavior. In some pieces these choices focused on the physical and musical interplay between the musicians. In other compositions an environment was set up in which the performer interacted with sound sources or sound objects.

Different choices were taken and mixed in the use of improvisation as a compositional tool. Some compositions fixed the performers movements in the score while others left the kind and range of activity open for improvisation with space or sounds. Also different approaches using and integrating the proposed tool itself in the composition could be noticed. While some composers created a scored piece that was then projected into space and complemented with processed sounds, others started from the interaction and sound processing possibilities offered by the setup to develop the whole piece. Yet another option that was explored was to present an electroacoustic composition that was then interpreted with the help of the MELE tool by a musician controlling the spatialisation or the dynamics of the electronic part.

In his trio *Langs Rafels* for clarinet, alto, and bassoon, Jesse Broekman explores hidden layers of instrumental timbre revealed by his sound treatment. The musicians navigate each others' timbre spaces by moving about the stage, their spatial orientation shaping the sound spatialisation. In *Langs Rafels*, the clarinet, the bassoon and the right arm of the alto player are tracked (figure 3).

In Carlo Ciceri's duo *Violata* for alto and flute, the spatialisation of the processed sound is related to the positions of the musicians on stage as they revolve around the centrally placed music stands. The movements of the musicians' right arms induce subtle and organic micro-variations in the spatialisation keeping the projected sound alive (figure 4).

For his piece *Tball* for trumpet, David Pirrò created a virtual object with which the performer plays by participating in a real-time physical simulation. In listening to the sound resulting from the interaction and watching the behavior of the instrumentalist, the object appears in our imagination. In *Tball*, the trumpet's bell is tracked (figure 5).

For his violin and bass clarinet improvisation duo *A Short Walk Through Time*, Stephan Prins built a granulation-based virtual instrument which is played collectively by the performers' head positions and orientations. The composer is performing as well by controlling certain aspects of the instrument with a fader box. In *A Short Walk Through Time*, the two musicians wear tracked caps (figure 6).

In Gerriet K. Sharma's piece *cornerghostaxis #1* the bassoonist is accompanied by a fixed four-channel electroacoustic composition. The spatial behavior of the performer very subtly controls the spatialisation of the piece, thus allowing for an intimate relationship between the unprocessed instrument and the electronic sounds. In *cornerghostaxis #1*, a tracking target is attached to the bassoon (figure 7).

Tuning into paranoia by Shiori Usui is a piece for a trum-

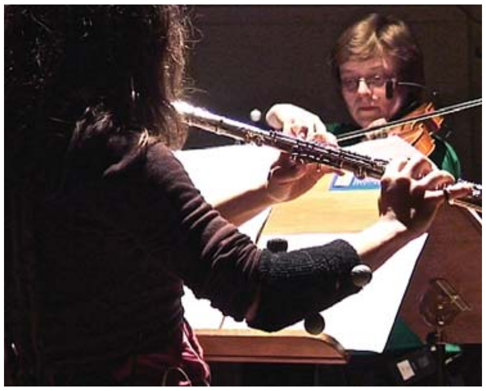


Figure 4. Video still of Annegret Mayer-Lindenberg (viola) and Marie-Noëlle Choquette (flute) playing Carlo Ciceri's piece *Violata*

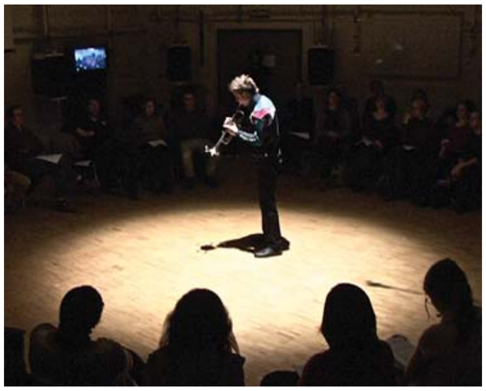


Figure 5. Video still of Paul Hübner playing David Pirrò's piece *TBall*

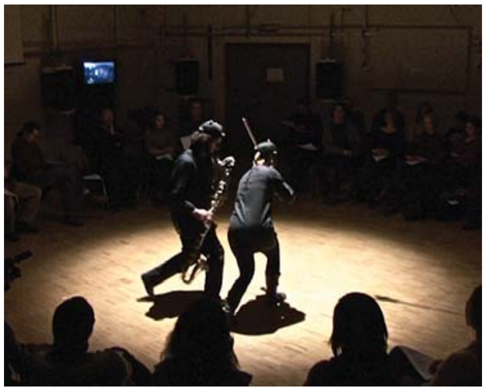


Figure 6. Video still of Marieke Berendsen (violin) and Jason Alder (bass clarinet) playing Stefan Prins' piece *A Short Walk Through Time*

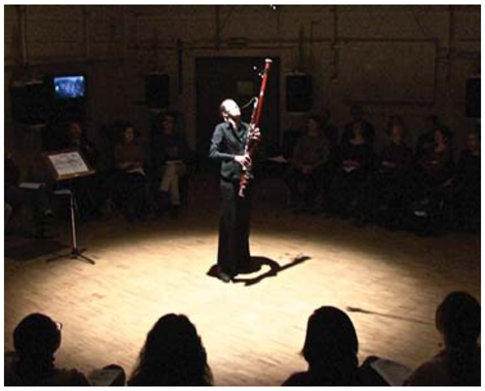


Figure 7. Video still of Dana Jessen (bassoon) playing Gerriet K. Sharma's piece *cornerghostaxis #1*

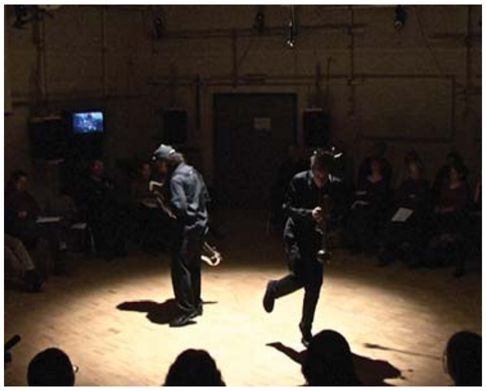


Figure 8. Video still of Jason Alder (bass clarinet) and Paul Hübner (trumpet) playing Shiori Usui's piece *Tuning into paranoia*

pet and a bass clarinet player engaging in a dramatic situation on stage. The expression of their musically enacted state of mind is enhanced by the live electronics processing and its control through their socio-spatial relationship. In *Tuning into paranoia* the second bell of the trumpet and the head of the bass clarinet player are tracked (figure 8).

6 CONCLUSIONS

The role of the performers was central to the whole workshop. They confirmed that they could gain a new and different access to the issues concerning performance with live electronics. As the control of the electronics was "attached" to their bodies and their movement, the effect on the resulting sounds was more direct, without mediation through other external devices that they would have to learn to use or play. Because of the unobtrusiveness of the tracking they could move relatively freely in the space and in some cases forget the markers they were wearing or that were attached to their instruments. These preconditions assured that they could get more conscious about the changes they could provoke in the sound and in the spatialisation and get a more precise control of these. In a conclusive meeting after the final concert, having then a clearer overview of the possibilities of MELE, some of them felt that in some pieces not all the potential implicit in this approach has been explored by the composers and that they would like to explore live electronics much further in such a setup. In other cases the musicians – especially those involved in the pieces that used physical models to drive sound synthesis and projection – underlined that they felt having achieved a clearer understanding of the dynamics of the electronics and how they could influence it. In fact, during the rehearsals of these pieces the musicians surprisingly asked for a more complex thus a more "realistic" interaction with the programmed physical model, that was in the beginning kept simple, in order to achieve a finer control on the sound.

The fact that a large volume was reliably tracked gave the performers the possibility to move relatively unconstrained in the space. For the audience this created the impression that the musicians were playing in a "scene", an environment in which different things happen, controlling diverse aspects of the live electronics. This situation is opposed to similar contexts in which gestures or smaller movements are used to drive the live electronics. If a limited range of action is used by the performers, the impression is created that they are playing an additional instrument, highlighting their interaction with this "device".

As a consequence, the musicians felt themselves and their actions on stage very much in the focus of the audience's attention, which resulted in a different awareness of their performance. After the final concert, besides underlining that they surely will integrate these experiences in future performances, the performers formulated the need for a choreographic support, especially concerning the "mise en scene" aspects. But also the composers had to deal with issues concerning more explicitly the performance situation, which demanded to be composed or choreographed besides the notes that have to be played by the musicians.

A general issue that emerged during the MELE workshop concerned how composers and performers work together on a piece. In the end it became clear to all of the participants, that in the particular situation of this workshop, where the performers were deeply linked also with the electronics part of the pieces and thus to compositional choices and ideas, a collaborative way of working was needed involving equally both musicians and composers. This resulted in most of the cases in a process in which the performers took actively part in the composition by taking decisions and developing ideas. The composers had to relate to aspects of the performance from a compositional point of view, guiding and supporting the musicians and taking into account their needs and constraints. This way of working together that established itself almost naturally was felt as very inspiring and rewarding by all the participants.

The MELE workshop was an intensive period of experimentation where many new but also already developed ideas were tested and put into work. Another important result is that both composers and performers could gain different insights in their work and especially in the relation between each other. These aspects are not specific to the particular context in which they were worked out – the CUBE or MELE – but refer to general issues concerning performance aspects thus applying to very different contexts.

As we have described in section 5 of this paper, the pieces realized during the workshop were very different from each other, adopting different strategies for the use of the tracking data and ways to link bodily movement to sound production and spatialisation. The spectrum of the solutions ranged from "classical" mappings to new approaches that used physical models as an intermediary level in the interaction of the musicians with the dynamics of the sound production and spatialisation. Particularly these last approaches were very inspiring especially for the performers as they could relate easily to such systems, gaining a very precise and intimate control of the electronics in those pieces.

Given the success of the workshop and the great interest of composers and instrumentalists in our approach, the next MELE workshop will be offered in the context of *impuls 2011* at KUG in Graz. Interested instrumentalists and composers should contact the first author of this paper in time, as only a few participants can be accepted.

7 ACKNOWLEDGEMENTS

The authors thank the following persons who have helped to make the MELE workshop a very successful one: Rémi Desmonet (audio and video streaming, IRCAM), Wolfgang Jäger (sound projection, MKL), Dieter Kovacic (camera and video mixing), Thomas Musil (live electronics and sound projection), Olga Neuwirth (composition coaching), Markus Noisternig (sound projection, audio and video streaming, IRCAM), Ute Pinter (organization *impuls 2009*), Winfried Ritsch (sound projection, audio and video streaming, MKL) Billy Roisz (camera, video mixing and editing), Stephan Warum (sound engineering), and IOhannes Zmölnig (audio and video streaming).

8 REFERENCES

[1] J. M. Zmölnig, W. Ritsch, and A. Sontacchi. The IEM-Cube. In E. Brazil and B. Shinn-Cunningham, editors, *Proceedings of the 9th International Conference on Auditory Display (ICAD2003)*, pages 127–130, Boston, USA, 2003. Boston University Publications Production Department.

[2] S. Gallagher. Phenomenological and experimental research on embodied experience. [online] http://pegasus.cc.ucf.edu/~gallaghr/paris2000.html, accessed 2009/04/09.

[3] M. M. Wanderley, B. W. Vines, N. Middleton, C. McKay, and W. Hatch. The musical significance of clarinetists' ancillary gestures: An exploration of the field. *Journal of New Music Research*, 34(1):97–113, 2005.

[4] M. M. Wanderley, P. Depalle, and O. Warusfel. Improving instrumental sound synthesis by modeling the effects of performer gesture. In *Proceedings of the 1999 International Computer Music Conference*, 1999.

[5] A. Camurri, S. Hashimoto, M. Ricchetti, A. Ricci, K. Suzuki, R. Trocca, and G. Volpe. Eyesweb: Toward gesture and affect recognition in interactive dance and music systems. *Comput. Music J.*, 24(1):57–69, 2000.

[6] T. Winkler. Motion-sensing music: Artistic and technical challenges in two works for dance. In *International Computer Music Conference*, San Francisco, 1998. International Computer Music Association.

[7] M. M. Wanderley and P. Depalle. Gestural control of sound synthesis. *Proceedings of the IEEE*, 92(4):632–644, 2004.

[8] M. T. Marshall, J. Malloch, and M. M. Wanderley. Gesture control of sound spatialization for live musical performance. In M. S. Dias, S. Gibet, M. M. Wanderley, and R. Bastos, editors, *Gesture-Based Human-Computer Interaction and Simulation, Revised Selected Papers*, pages 227–238, Berlin, Heidelberg, 2009. Springer-Verlag.

[9] N. Peters, M. Evans, and E. Britton. Trakhue - intuitive gestural control of live electronics. In *Proceedings of the 2007 International Computer Music Conference*, 2007.

[10] M. T. Marshall, N. Peters, A. R. Jensenius, J. Boissinot, M. M. Wanderley, and J. Braasch. On the development of a system for gesture control of spatialization. In *Proceedings of the 2006 International Computer Music Conference*, pages 260–266, 2006.

[11] J. C. Schacher. Gesture control of sounds in 3d space. In *NIME '07: Proceedings of the 7th international conference on New interfaces for musical expression*, pages 358–362, New York, NY, USA, 2007. ACM.

[12] T. Musil, W. Ritsch, and J. Zmölnig. The CUBEmixer a performance, mixing and mastering tool. In *Proceedings of the 2008 Linux Audio Conference*, 2008.

MUSICAL VOICE INTEGRATION/SEGREGATION: *VISA* REVISITED

Dimitris Rafailidis
Department of Informatics
Aristotle Univ. of Thessaloniki
draf@csd.auth.gr

Emilios Cambouropoulos
Department of Music Studies
Aristotle Univ. of Thessaloniki
emilios@mus.auth.gr

Yannis Manolopoulos
Department of Informatics
Aristotle Univ. of Thessaloniki
manolopo@csd.auth.gr

ABSTRACT

The Voice Integration/Segregation Algorithm (VISA) proposed by Karydis et al. [7] splits musical scores (symbolic musical data) into different voices, based on a perceptual view of musical voice that corresponds to the notion of auditory stream. A single 'voice' may consist of more than one synchronous notes that are perceived as belonging to the same auditory stream. The algorithm was initially tested against a handful of musical works that were carefully selected so as to contain a steady number of streams (contrapuntal voices or melody with accompaniment). The initial algorithm was successful on this small dataset, but was proven to run into serious problems in cases were the number of streams/voices changed during the course of a musical work. A new version of the algorithm has been developed that attempts to solve this problem; the new version, additionally, includes an improved mechanism for context-dependent breaking of chords and for keeping streams homogeneous. The new algorithm performs equally well on the old dataset, but gives much better results on the new larger and more diverse dataset.

1. INTRODUCTION

It appears that the term 'voice' has different meanings for different research fields (traditional musicology, music cognition and computational musicology) - a detailed discussion is presented in [2]. A perceptual view of voice adopted in previous voice separation modelling attempts [7, 13], allows for multi-tone simultaneities in a single 'voice' – this is the most significant difference of such model(s) with other existing voice separation models [4, 9, 10, 11, 12, 14].

Standard understanding of the term voice refers to a *mono*phonic sequence of successive non-overlapping musical tones; a single voice is thought not to contain multi-tone sonorities. However, if 'voice' is seen in the light of auditory streaming, then, it is clear that the standard meaning is not sufficient. It is possible that a single monophonic sequence may be perceived as more than one voice/stream (e.g., pseudopolyphony or implied polyphony) or that a passage containing concurrent notes may be perceived as a single perceptual entity.

In Figure 1, all existing algorithms that are based on purely monophonic definitions of voice (except Kilian and Hoos's [8] algorithm that allows fewer voices if forced by the user), would detect five voices that clearly are not independent voices. The VISA algorithm [7] and the new version presented in this paper detect two voices/streams that correspond to melody and accompaniment.

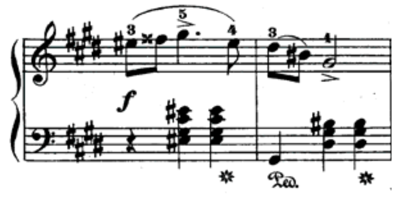


Figure 1 How many voices in this excerpt from Chopin's Mazurka Op.6, No.2?

It is suggested that a general musical voice/stream segregation algorithm should be able to cope with any kind of music, not just musical textures that are constructed by the use of a steady number of monophonic voices (e.g. fugues, chorales, string quartets, etc.). Such an algorithm, among other things, is very useful for developing MIR systems that enable pattern recognition and extraction within musically pertinent 'voices' – for instance, there is no reason to 'look' for melodic patterns in homophonic accompanimental parts of songs.

In this paper, initially, a number of problems related to voice/stream separation not addressed by the model proposed in [7] are presented. A brief description of the first prototype version of the Voice Integration/ Segregation Algorithm (VISA) follows, and, then, a number of improvements to the algorithm are given. After an evaluation of the new prototype on a more extended

SMC 2009, July 23-25, Porto, Portugal

and diverse groundtruth dataset, the paper is concluded by some future suggestions for further improvements.

2. VOICE SEPARATION MODELS

Voice separation models based on a monophonic definition of voice (REFERENCES) attempt to determine a minimal number of lines/voices such that each line consists of successions of tones that are maximally proximal in the temporal and pitch dimensions. Such models perform well on music that is composed of a steady number of voices/lines, but fail to give musicologically or perceptually relevant results in most other cases. The horizontal integration of notes relies primarily on two fundamental auditory streaming principles: *Temporal Continuity* and *Pitch Proximity* [6].

Adopting a perceptual view of voice, which is very close to the notion of auditory stream, two recent studies [7, 12] allow multi-tone simultaneities in a single 'voice'. In addition to the two previously mentioned perceptual principles these models enable vertical integration based on the Synchrony Note Principle [2], whereby 'notes with synchronous onsets and same inter-onset intervals IOIs (durations) tend to be merged into a single sonority.'

VISA [7] starts by identifying synchronous notes that tend to be merged into single sonorities and, then, uses the horizontal streaming principles to break them down into separate streams (most algorithms ignore the vertical component). This is an optimisation process wherein various perceptual factors compete for the production of a 'simple' interpretation of the music in terms of a minimal number of streams. If the reader is not acquainted with VISA, we suggest that section 3.1 be read before the next section (2.1).

The algorithm presented herein has been developed as a means to explore more systematically the ideas and principles of musical auditory streaming in symbolic musical data; it is an exploratory prototype that requires further development. The proposed prototype is not directly comparable to other voice separation algorithms as its underlying definition of 'voice' is different and has a different aim. In this paper we will compare our new version of VISA with the earlier version [7].

2.1. Problems of VISA and improvements

VISA was initially tested on ten musical examples [7] that were carefully selected so as to contain a steady number of streams (i.e. musical works comprising of contrapuntal melodic lines, or of melody and homorhythmic accompaniment). The algorithm performed well on this limited dataset. However, we discovered that the algorithm ran into serious problems when tested on music that contained non-homorhythmic homophonic accompanimental textures or diverse musical textures (homophonic and polyphonic together).

The main problem of this early version of the algorithm is that, when a new voice/stream appears, it is available for continuation throughout the rest of the piece. This is no serious problem in contrapuntal polyphonic works where the number of voices remains steady throughout a musical work. However, in homophony we usually have a single stream, i.e. one harmonic homorhythmic stream, or two streams, i.e. one melodic voice plus rhythmically independent accompaniment. Occasionally, additional rhythmically independent lines may appear locally but these usually disappear after their emergence rather than remain active throughout the rest of the piece.

When the early version of VISA breaks a homophonic piece into three (or more) streams locally, it tries to find the best continuation for these three streams throughout the rest of the piece; occasionally the third stream may erroneously be selected to continue stream 1 or 2, or all three streams may continue in parallel. For instance, in Figure 2 (measure 11) we have three streams - the algorithm considers the upper voice as stream 3 since it has already allocated streams 1 and 2 to the first notes in the measure – the mistake is then propagated to the rest of the score as the next top notes are closer to stream 3 (actually, stream 2 is abandoned and stream 3 and 1 remain active in reverse order, i.e. stream 3 above stream 1). In Figure 3 the algorithm erroneously locates three streams in measure 29 (as the bass note overlaps with the following notes of the chord they cannot all be placed in one stream – see discussion in Section 4), and from there on it continues 'giving' notes to all three streams rather than returning to two streams (melody and accompaniment). Such a relatively simple mistake may decrease accuracy dramatically as a local increase in streams may be erroneously be propagated throughout the rest of the score.

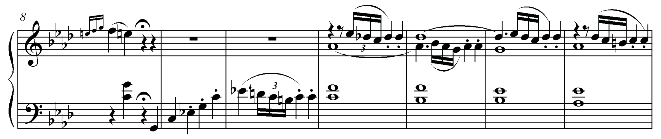


Figure 2 Excerpt from Beethoven's Sonata Op.2, No.1, Allegro Con Brio.

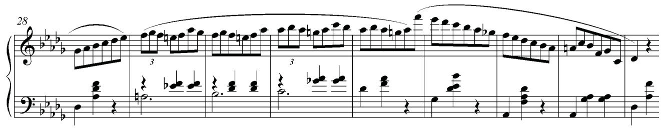


Figure 3 Excerpt from Chopin's Waltz Op. 64, No.1

A simple solution has been introduced to address this problem. The solution is based on the observation that in homophony, music is perceived as a single stream that may be 'fattened' or 'thinned' by adding or subtracting extra streams, whereas in polyphonic music, streams have an independent life and are equally important. Following this, when the texture is locally homophonic (i.e. many notes start and end together), the algorithm is forced to switch back to streams 1 and 2 after having identified three (or more) streams. This simple modification increased accuracy significantly as seen in Table 1.

A second important improvement involves the breaking of chords consisting of equal duration notes. In the early version of VISA this was partially incorporated in the program in association to the Top Voice Rule, i.e., the top voice should be minimally fragmented. This is handled by adding a penalty to the cost of a voice continuation that does not fulfil this rule. To find the continuation with the minimal cost, a chord may be split so that one note can be assigned to the top voice. In our new proposal, a chord consisting of equal duration notes is split into sub-chords based on the context of existing or forthcoming independent streams. That is, if in the vicinity of the current chord there are more voices, the chord may be split so as to match the adjacent voice structure. The Top Voice Rule, thus, becomes a special case of this general vertical cluster splitting process. For instance, in Figure 4 we perceive a melodic line that lies within a static harmonic stream; the chords marked by an asterisk consist of equal duration notes so initially they are merged into a single vertical cluster by VISA – the proposed function that breaks chords (vertical clusters) 'pulls out' the second note of these chords and assigns it to the independent melodic voice.

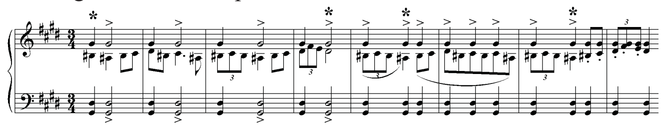


Figure 4 Opening of Chopin's Mazurka Op. 6, No.2.

Further smaller modifications that improve the algorithm's performance include the following: Firstly, the pitch distance between notes/chords takes into account not only the pitch of the current notes and the last notes of preceding voices, but also the second-to-last notes of preceding voices. In case the last pitches of two voices coincide, the algorithm could not decide which current pitch should be assigned to which of the two unison pitches; taking into account the second-to last pitches resolves such ambiguous cases. Secondly, the distance metric takes into account not only pitch and temporal distance, but additionally a new parameter that favours homogeneity of streams in terms on number of co-sounding tones. In other words, linking a chord cluster with many tones to a single note is discouraged and contributes to a larger distance, whereas linking similar density clusters adds smaller cost. We discovered that this homogeneity factor solved problems in a number of cases; however, there are cases where this factor is counterproductive.

3. THE REVISED *VISA* ALGORITHM

The previous algorithm posed by Karydis et al. [7] and also our current revised implementation consist of two steps: fist, vertical integration which merges notes with same onsets and durations if the musical context is homophonic, and second, links notes/chords horizontally into voices/streams.

3.1. Brief description of VISA

The original Voice Integration/Segregation Algorithm [7] accepts as input a musical piece in symbolic form and outputs the number of detected musical voices/streams. At present, the algorithm is applied to quantized musical data; expressively performed musical data require quantization before being fed into the algorithm. The appropriate number of streams is determined automatically by the algorithm and can be lower than the maximum number of notes of the largest chord.

VISA moves in a step-wise fashion through the input sequence of musical events (individual notes or concurrent note sonorities). Let the entire musical piece be represented as a list L of notes that are sorted according to their onset times. A sweep line, starting from the beginning of L, proceeds through the onset times in L. The set of notes that have onsets equal to a position of the sweep line is denoted as sweep line set (SLS).

For a set of concurrent notes at a given point (SLS), we have to determine when to merge them according to the *Synchronous Note Principle*. Because it is possible that synchronous notes may belong to different voices, we need a way to decide if such merging should be applied. For each SLS, the algorithm examines a certain musical context (window) around them. If inside the window, most co-sounding notes have different onsets or offsets, then it is most likely that we have polyphonic texture (independent monophonic voices), so occasional synchronous notes should not be merged - each note is considered to be a singleton cluster. If most notes are concurrent (same onsets and IOIs) implying a homophonic texture, then they should be merged - concurrent notes form a cluster. This way, each SLS is split into a number of note clusters. At the present stage, the window size *w* and homophony/polyphony threshold *T* have been determined manually (same for all the data) by finding values that give optimal results for the selected test data set.

For each SLS in the piece, we have a set of previously detected voices (V) and the current set of note clusters (C). Between every detected voice of V and each note cluster of C, we draw an edge to which we assign a cost. The cost function calculates the cost of assigning each cluster to each voice according to the *Temporal Continuity Principle* and the *Pitch Proximity Principle*. Notes that overlap receive a cost value equal to infinity.

A dynamic programming technique finds the best matching (lowest cost) in the bipartite graph between previous voices and current clusters. If voices are fewer than clusters, then one or more voices may be (temporarily) terminated. If clusters are fewer than voices, then new voices may appear. The matching process, additionally, takes into account two constraints. The first one is that voice crossing should be avoided. Therefore a sub-optimal solution in terms of cost may be required that avoids voice crossing. The second one is that the top voice should be minimally fragmented (Top Voice Rule by [11]). This is handled by adding a penalty to the cost of a matching that does not fulfill this rule - to find the matching with the minimal cost - a cluster may be split into sub-clusters, so that one can be assigned to the top voice.

3.2. Revised Version of VISA

3.2.1. *Numbering of Voices/Streams*

In music that is primarily homophonic, the tendency is to have one or two stable streams (pure homorhythmic texture, or melody and harmonic accompaniment), whereas further independent voices/streams appear only locally (see discussion in Section 2.1). To avoid keeping 'alive' extra voices/streams (e.g. third or fourth stream), a simple solution has been introduced: when the texture is locally homophonic (i.e. many notes start and end together), the algorithm is forced to switch back to streams 1 and 2 after having identified three (or more) streams. That is, when in the MatchingVoicesToClusters procedure we have more voices than clusters and also the context is homophonic, the current clusters are assigned to the basic streams 1 and 2.

In the middle of the excerpt in Figure 5 we have three Voices, V1: {N5, N10}, V3: {N6, N9, N11, N12} and V2: {N7, N8}. In the next SLS, note N13 is closer to V2 but is assigned to V1 because the algorithm prefers to abandon V3 moving back to the main two voices.

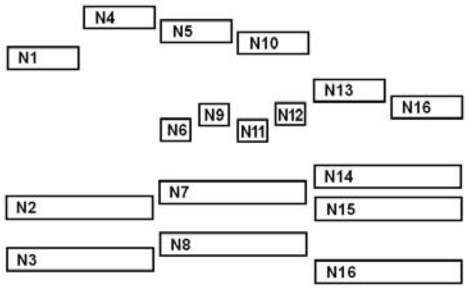


Figure 5 The third voice {N6, N9, N11, N12} is abandoned and, N13 continues the first voice – see text.

3.2.2. *Vertical Integration and BreakCluster Method*

If a number of notes are integrated vertically (they have same durations) and if the local context is homophonic, then the *BreakCluster* procedure is activated. This procedure looks ahead in the next three SLSs (more generally it can be designed to look in the local neighborhood before and/or after the current SLS); if it finds (using ClusterVertically) that there exist more clusters in one of the following SLSs than in the current SLS, it moves backwards from the SLS (with more clusters) breaking one by one its preceding clusters till it breaks the current SLS cluster. Preceding clusters are broken according to how close notes in the to-be-broken clusters are to the notes of the SLS with more clusters.

In the example of Figure 6, notes in the current SLS1 are clustered vertically into a single cluster as they have same onsets and durations, and also the context is homophonic. In this case, *BreakCluster* is activated and checks whether in any of the next three SLSs there are more clusters than in the current SLS. SLS2 and SLS3 contain a single cluster, but ClusterVertically splits SLS4 into 3 clusters: {N13}, {N14}, and {N15, N16}. Now, moving backwards it breaks the cluster in SLS3 into three clusters based on pitch proximity: {N9}, {N10} and {N11, N12}, then breaks SLS2 into {N5}, {N6} and {N7, N8} and, finally, the current cluster SLS1 into {N1}, {N2} and {N3, N4}. In a different scenario, if an SLS before the third SLS contained more than one clusters, then the BreakCluster procedure would have moved from that SLS backwards to the current SLS.

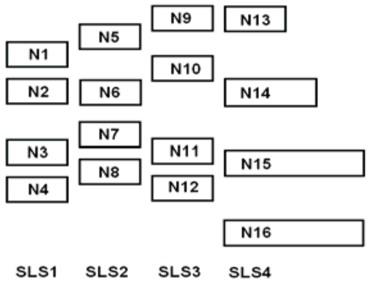


Figure 6 Breaking vertical clusters based on context.

3.2.3. *Matching notes to voices and cost calculation*

As mentioned in Section 3.1, after determining the clusters for each SLS, a bipartite graph is created for matching notes to voices. Each cell (i,j) of the graph designates the cost (distance) between the last cluster assigned to voice i and the current cluster j. In the previous implementation only pitch and time difference is taken into account for the calculation of the cost. In the current implementation we add a factor that relates to the difference of the number of notes in the two clusters. This difference, that is a kind of homogeneity factor (see Section 2.1), is calculated as $d_{h=}|n_i - n_j|/ n_i + n_j$ (where n_i is the number of notes in cluster i) and contributes by 25% to the total cost (along with 50% pitch difference contribution plus 25% inter-onset difference). In the example of Figure 7, note N6 is closer to cluster {N2, N3, N4} than cluster {N7, N8, N9} is in terms of average pitch, but the {N7, N8, N9} is assigned to cluster {N2, N3, N4} because the total cost is lower when the homogeneity factor is taken into account.

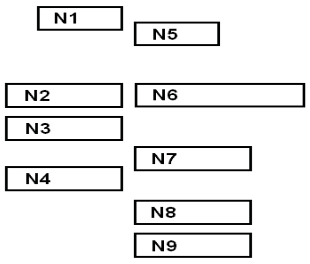


Figure 7 Homogeneity factor (lower chords are assigned to the same voice).

In the previous implementation for the cost calculation, only the last note/cluster in each voice was taken into account. There are cases, however, where more notes/clusters from the past are necessary to resolve ambiguity. In the current implementation, the pitch for each voice is calculated as the weighted average of the pitch of the last two notes/clusters of each voice (80% of the last cluster and 20% of the second-to-last cluster). In the example of figure 8, notes N1 and N3 have the same pitch, so there is ambiguity in assigning the next notes N4 and N5 to the previous voices. If next-to-last notes are taken into account, then the second voice containing note N2 and N3 will have a lower average pitch than the first voice (containing N1) and will be matched correctly to N5.

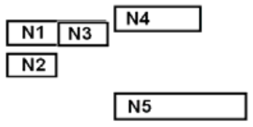


Figure 8 Resolving ambiguity in pitch distance.

4. RESULTS AND FUTURE WORK

The proposed algorithm has been tested on a set of musical extracts for piano.[1] The dataset has been annotated by a music theory research student that was instructed to indicate voices/streams on the scores after listening to the excerpts – a number of musical examples were discussed with him before doing this task – the student did not have knowledge of the computational implementation. The dataset that acted as groundtruth contains the ten pieces used in the initial testing of VISA [7] plus 22 excerpts primarily from piano sonatas by Beethoven (only the openings of the different sections have been annotated, as it is a very tedious task to manual annotate the full scores). The sonatas have been selected as they comprise of diverse musical textures, i.e. homophonic and contrapuntal textures. In future, larger number of music experts may provide groundtruth and/or empirical studies may generate more reliable datasets against which to test algorithms.

The accuracy of the proposed algorithm is measured as the weighted sum for each voice of the proportion of notes correctly assigned to a voice *i* over the total number of notes of voice *i* – each such proportion is multiplied by P_i, where P_i is the percentage of notes belonging to voice *i* against the overall number of notes. Assuming *N* is the number of voices, the accuracy is measured according to equation (1).

$$\text{Accuracy} = \sum_{i=1}^{N} P_i \frac{\#\,\text{notes, correctly assigned to voice i}}{\#\,\text{notes of voice i}} \quad (1)$$

In essence, accuracy counts the total number of notes that have been correctly assigned to the appropriate voice (according to the groundtruth), divided by the total number of notes. This accuracy measure is rather strict in the sense that notes that may have been placed together correctly in the same voice but may have been tagged incorrectly (e.g. placed together in voice *x* instead of voice *y*) are all counted as wrong. This is the main reason why in some cases accuracy is still low.

As can be seen in Table 1, the modifications incorporated in the current version of VISA improve significantly the performance of the algorithm. The average performance of the old version of VISA for the 22 new excerpts is 0.68 (first 22 excerpts in Table 1), whereas the average performance for the new version of VISA is 0.84, which means a 23% increase (16 percent units). The new algorithm does not improve performance on the limited old dataset (last 10 excerpts in Table 1) that was carefully selected to contain excerpts with steady number of 'clean' voices/streams. However, these tests show that overall we have a more flexible algorithm that performs well on diverse musical textures.

Voice/stream segregation is a difficult problem influenced by many different competing factors. The development of computational models such as the VISA algorithm is seen as a means to explore the mechanisms of voice separation to gain a better understanding of the problem with a view to developing more reliable computer models.

The current model can be improved in two ways: firstly, by redesigning the whole algorithm so as to take into account local context in a more integrated manner. Rather than matching clusters of one SLS to the last notes/clusters of previous voices (adding ad hoc cases in which the context is taken into account), it may be more powerful to look continuously for optimal solutions within a larger context.

[1] These pieces were downloaded in Melisma format from the Kern collection (http://kern.humdrum.org)

Secondly, further segregation factors must be taken into account such as tonal fusion, parallelism, pattern similarity, and, even, new overall integration/segregation strategies. For instance, the current method does not allow merging non-isochronous overlapping notes – it is clear, however, that there are cases where this should be allowed (e.g. the notes in the accompaniment of mm. 29-31 in Figure 3 clearly belong to the same voice/stream due to harmonic reasons).

	Old VISA	New VISA
Beethoven, Sonata 2-1 Allegro	0,66	0,86
Beethoven, Sonata 2-1 Adagio	0,82	0,86
Beethoven, Sonata 2-1 Minuet	0,61	0,73
Beethoven, Sonata 2-1 Prestissimo	0,93	0,93
Beethoven, Sonata 2-2 AllegroVivace	0,62	0,80
Beethoven, Sonata 2-2 LargoApp	0,69	0,91
Beethoven, Sonata 2-2 Scherzo	0,49	0,75
Beethoven, Sonata 2-2 Rondo	0,60	0,82
Beethoven, Sonata 2-3 AllegroConBrio	0,40	0,87
Beethoven, Sonata 2-3 Adagio	0,62	0,77
Beethoven, Sonata 2-3 Scherzo	0,74	0,73
Beethoven, Sonata 2-3 AllegroAssai	0,96	0,94
Beethoven, Sonata 10-2 Allegro	0,87	0,89
Beethoven, Sonata 10-2 Allegretto	0,43	0,73
Beethoven, Sonata 10-2 Presto	0,90	0,92
Beethoven, Sonata 13 Grave	0,72	0,98
Beethoven, Sonata 13 AdagioCantabile	0,23	0,56
Beethoven, Sonata 13 Rondo	0,94	0,85
Brahms, Waltz Op39 No8	0,80	0,89
Chopin, Mazurka Op6 No2	0,84	0,93
Chopin, Mazurka Op7 No1	0,70	0,92
Chopin, Waltz Op64 No1	0,43	0,91
Bach, Fugue BWV846	0,92	0,92
Bach, Fugue BWV859	0,96	0,93
Bach, Fugue BWV856	0,87	0,94
Bach, Fugue BWV852	0,97	0,91
Bach, Fugue BWV772	0,99	0,99
Bach, Fugue BWV784	0,96	0,96
Chopin, Mazurka Op7 No5	1,00	0,97
Chopin, Mazurka Op67 No4	0,88	0,88
Chopin, Waltz Op69 No2	0,90	0,96
Joplin, Harmony Club Waltz	0,98	0,92

Table 1 Accuracy for voice separation by the previous and the current implementation of VISA (the last ten pieces were used in the evaluation of the old VISA [7]).

5. CONCLUSIONS

The proposed voice separation algorithm incorporates the two principles of temporal and pitch proximity, and additionally, the Synchronous Note Principle. Allowing both horizontal and vertical integration enables the algorithm to perform well not only in polyphonic music that has a fixed number of 'monophonic' lines, but in the general case where both polyphonic and homophonic elements are mixed together. We have shown in the above preliminary experiment that the proposed algorithm can achieve good performance in diverse musical textures in terms of identifying perceptually relevant voices/streams.

6. REFERENCES

[1] Bregman, A (1990) *Auditory Scene Analysis: The Perceptual Organisation of Sound.* The MIT Press, Cambridge, MA.

[2] Cambouropoulos, E. (2008) Voice and Stream: Perceptual and Computational Modeling of Voice Separation. *Music Perception* 26(1):75-94.

[3] Cambouropoulos, E. (2000) From MIDI to Traditional Musical Notation. In *Proceedings AAAI Workshop on Artificial Intelligence and Music*, Austin, TX.

[4] Chew, E. and Wu, X. (2004) Separating Voices in Polyphonic Music: A Contig Mapping Approach. In *Proceedings 2nd International Symposium on Computer Music Modeling and Retrieval: (CMMR'2004)*, pp. 1-20.

[5] Deutsch, D. (1999) Grouping Mechanisms in Music. In D. Deutsch (ed.), *The Psychology of Music* (revised version). Academic Press, San Diego, CA.

[6] Huron, D. (2001) Tone and Voice: A Derivation of the Rules of Voice-Leading from Perceptual Principles. *Music Perception*, 19(1):1-64.

[7] Karydis, I., Nanopoulos, A., Papadopoulos, A.N. & Cambouropoulos, E., (2007) VISA: the Voice Integration/ Segregation Algorithm. In *Proceedings 8th International Conference on Music Information Retrieval (ISMIR'07),* Vienna, Austria, pp. 445-448.

[8] Kilian j. and Hoos H. (2002) Voice Separation: A Local Optimisation Approach. In *Proceedings 3rd International Conference on Music Information Retrieval* (ISMIR' 2002), Paris, France, pp.39-46.

[9] Kirlin, P.B. and Utgoff, P.E. (2005) VoiSe: Learning to Segregate Voices in Explicit and Implicit Polyphony. In *Proceedings 6th International Conference on Music Information Retrieval (ISMIR'2005),* London, UK, pp. 552-557.

[10] Madsen, S.T. and Widmer, G. (2006) Separating Voices in MIDI. In *Proceedings 9th International Conference in Music Perception and Cognition (ICMPC'2006)*, Bologna, Italy.

[11] Temperley, D. (2001) *The Cognition of Basic Musical Structures*. The MIT Press, Cambridge, MA.

[12] Rafailidis, D., Nanopoulos, A., Cambouropoulos, E. & Manolopoulos, Y. (2008), Detection of Stream Segments in Symbolic Musical Data. In *Proceedings 9th International Conference on Music Information Retrieval (ISMIR'08),* Philadelphia, PA, pp.83-88.

[13] Szeto, W.M. and Wong, M.H. (2003) A Stream Segregation Algorithm for Polyphonic Music Databases. In *Proceedings 7th International Database Engineering and Applications Symposium (IDEAS'03),* Hong Kong, pp.130-138.

A COMPUTATIONAL MODEL THAT GENERALISES SCHOENBERG'S GUIDELINES FOR FAVOURABLE CHORD PROGRESSIONS

Torsten Anders and Eduardo R. Miranda
Interdisciplinary Centre for Computer Music Research (ICCMR)
University of Plymouth
{torsten.anders,eduardo.miranda}@plymouth.ac.uk

ABSTRACT

This paper presents a formal model of Schoenberg's guidelines for convincing chord root progressions. This model has been implemented as part of a system that models a considerable part of Schoenberg's Theory of Harmony. This system implements Schoenberg's theory in a modular way: besides generating four-voice homophonic chord progressions, it can also be used for creating other textures that depend on harmony (e.g., polyphony).

The proposed model generalises Schoenberg's guidelines in order to make them applicable for more use cases. Instead of modelling his rules directly (as constraints on scale degree intervals between chord roots), we actually model his explanation of these rules (as constraints between chord pitch class sets and roots, e.g., whether the root pitch class of some chord is an element in the pitch class set of another chord). As a result, this model can not only be used for progressions of diatonic triads, but in addition also for chords with a large number of tones, and in particular also for microtonal music beyond 12-tone equal temperament and beyond 5-limit harmony.

1 INTRODUCTION

Computational models of music theory are interesting for at least two reasons. Firstly, declarative models improve our understanding of the theory. Secondly, computational models can also be used as tools in the composition process.

Tonal harmony has often been modelled declaratively. Surveys on this subject are provided in [7] and [2]. Particular important is the system CHORAL [3, 4], which creates four-part harmonisations in the style of Johann Sebastian Bach for given choral melodies. It implements about 350 rules, and received much attention for the musical quality of its output. The music representation MusES [6] has been used for harmonic analysis, melody harmonisation, and modelling jazz improvisation. A number of other systems also do automatic melody harmonisation. For example, [13] proposes a lucid system with a small set of 20 rules, which creates four-part harmonisations of a choral melody. [10] describes another system that automatically harmonises a given melody. Coppelia [14] creates homophonic chord progressions, which additionally feature a rhythmical structure. [9] presents a further system that generates choral harmonisations in the style of Johann Sebastian Bach.

The authors of [7] claim that the "technical problem of four-voice harmonization may now be considered as solved". However, existing systems only solve a special subtask of harmony: instead of creating a harmonic progression from scratch, these systems harmonise a given melody, most often creating a new chord for each melody note (choral harmonisation). Also, most existing systems create solutions that are very modest musically. For example, only the systems of Ebcioglu and Phon-Amnuaisuk address modulation at all. Even Ebcioglu's highly complex system CHORAL formalises possible chord progressions simply by quasi a transition table that only allows for common progressions. For example, in major the degree II is mostly followed by V, and by I or VI only under specific conditions, [1] while the diminished triad vii^o is always followed by the tonic I [3, p. 240 f]. Yet, these chords can progress to any degree in principle.

We argue that modelling harmony is not a solved problem yet. Harmony is a highly complex phenomenon as demonstrated by the library of harmony textbooks available. Still missing is a system that models

SMC 2009, July 23-25, Porto, Portugal

[1] Ebcioglu's rule set states that II can only be followed by I or VI if some non-bass voice moves by a third skip from the VI (the fifths of the II chord) to the I.

harmony on the level of abstraction presented by acclaimed theory texts like [11]. For example, using a transition table for chord progressions is a useful shortcut, but theorists like Schoenberg teach us better alternatives.

This paper describes a system that models a considerable part of Schoenberg's Theory of Harmony [11]. This system generates self-contained harmonic progressions – instead of harmonising existing melodies – and can so create the harmonic backbone of new compositions. The harmonic model is modular for applications beyond four-part harmonisations. It can serve as a foundation for modelling musical styles that depend on harmony (e.g., Baroque counterpoint), and can also be an interesting composition tool. Schoenberg has been selected as theoretical foundation, because this textbook is unique in its focus on writing convincing chord progressions, instead of focusing on analysis, melody accompaniment or figured bass (as many other harmony textbooks do).

For space limitation, this paper details only one aspect of Schoenberg's theory, but this aspect is of particular importance. In the chapter "Some Directions on Writing Favourable Progressions" ("Einige Anweisungen zur Erzielung günstiger Folgen" [11, p. 134 ff]), Schoenberg presents guidelines on root progressions that result in particularly convincing chord sequences. [2] This paper formalises these guidelines. To our knowledge, these guidelines have never been modelled before. The model has been implemented in Strasheela [1].

In addition, this model generalises these guidelines for chords with a large number of notes (as long as we know their root), and in particular also for microtonal music beyond 12-tone equal temperament and beyond 5-limit harmony [8]. Schoenberg discusses his guidelines only in the context of triads in a diatonic scale as he formulates his rules on the scale degree of chords. Nevertheless, his detailed explanation of these rules are more general. Instead of formalising Schoenberg's rules directly, this paper actually models his explanation of these rules. Doing so makes his concepts applicable for more music, but also puts some corner cases of classical harmony in a new light.

Plan of Paper

The rest of this paper is organised as follows. Schoenberg's guidelines on root progressions are recapitulated in Section 2. Section 3 presents a model that formalises and generalises these guidelines. Musical results are presented in Section 4. The paper ends in a summary (Section 5).

[2] A summary of these guidelines can also be found at the beginning of his book "Structural Functions of Harmony" [12].

2 THE MUSIC THEORY

Schoenberg distinguishes three root progression cases: ascending, descending and super-strong progressions. In an *ascending* progression, the chord root progresses a fourth up / a fifths down (e.g., $V - I$) or a third down (e.g., $I - VI$). Schoenberg calls such progressions also *strong* and advocates their unreserved use.

A *descending* progression – quasi a reversed ascending progression – proceeds a fifths up ($I - V$), or a third up (e.g., $I - III$). Schoenberg avoids the term *weak*, but nevertheless discourages their unconfined use. Instead, Schoenberg recommends that in a sequence of three chords C_1, C_2, C_3 the sequence C_1, C_2 can only be descending if C_1, C_3 is ascending (e.g., $III - V - I$). In that case, the purpose of the middle chord C_2 is similar to the purpose of a passing note in a melody.

Finally, a *super-strong* progression connects two chords whose root are a second apart (e.g., V, VI or V, IV). Such progressions are typically used in a deceptive cadence. Because their quality can be considered too strong, Schoenberg advises to use them sparely.

Schoenberg argues at length possible reasons for the different qualities of these progressions. These will be briefly reported below when they are formalised.

3 THE FORMAL MODEL

This section presents the formal model of Schoenberg's guidelines for favourable chord progressions. The model implements Schoenberg's explanation instead of his actual rules.

Our full system defines a rich and highly extendable music representation designed for modelling a wide range of music theories. This representation provides a rich collection of score objects including elements such as notes, or rests, analytical concepts such as intervals, scales, chords, or meter, grouping concepts such as containers that arrange their content sequentially or parallel in time, as well as concepts for organising musical form such as motifs. For brevity, this section introduces only a small fraction of this representation that is sufficient for modelling Schoenberg's guidelines on root progressions.

A chord C is a score object that represents the analytical notion of a chord or harmony. This analytical object is silent when the score is played, but influences the pitches of note objects. For the present model, a chord object encapsulates only two attributes: the pitch classes of the chord *pcs* and its root *root*. Both these attributes are variables in the logic or constraint programming sense. *root* is a finite domain integer, and *pcs* is a finite set of integers. In Schoenberg's theory, the root of a chord is always a member of its pitch class set: $root(C) \in pcs(C)$. For example, the root of the diminished triad $\{B, D, F\}$ is B.

We will now model Schoenberg's notion of ascending, descending and super-strong progressions as Boolean functions on pairs of consecutive chord objects C_1 and C_2. Schoenberg explains that in an ascending progression the root of a former chord is "over-ruled" by a new root in the following chord. Formally, the root of C_1 is also a member of the pitch class set of C_2, but the root of C_2 was not contained in C_1 (Figure 1).

$$\begin{aligned} isAscending_1(C_1, C_2) := \quad & root(C_2) \notin pcs(C_1) \\ & \wedge\, root(C_1) \in pcs(C_2) \end{aligned}$$

Figure 1. Ascending progression: the root of the first chord is also contained in the second chord, but the root of the second chord is new

In a descending progression, the root of the second chord is a "parvenu" according to Schoenberg, the ruler (root) of the first chord quasi backs down to one of his former "subjects". Formally, a non-root pitch class of the first root becomes root in the second chord (Figure 2).

$$\begin{aligned} isDescending(C_1, C_2) := \quad & root(C_2) \in pcs(C_1) \\ & \wedge\, root(C_1) \neq root(C_2) \end{aligned}$$

Figure 2. Descending progression: a non-root pitch class of the first root becomes root in the second chord

In ascending and descending progressions, chords share common pitch classes (what Schoenberg calls a "harmonic band"). In a super-strong progression, all pitch classes of the second chord are new and there are no common pitch classes (Figure 3).

$$isSuperstrong(C_1, C_2) := pcs(C_1) \cap pcs(C_2) = \varnothing$$

Figure 3. Super-strong progression: two consecutive chords do not share any pitch classes

Schoenberg only discusses these three cases, because he discusses only diatonic triads. However, there exist two further cases in principle. Firstly, two different chords can share the same root as in $C - Cmin$ (Figure 4).

Secondly, outside the set of diatonic triads there exist progressions that are connected by a harmonic band,

$$isConstant(C_1, C_2) := root(C_1) = root(C_2)$$

Figure 4. Constant progression: two (possibly different) chords share the same root

but that are neither ascending nor descending progressions according to the definitions above. For example, the triadic progression $C - E\flat$ shares common pitch classes (the tone G), but it belongs to none of the categories above. In our subjective assessment these progressions also feel strong, like the ascending progressions. Instead of introducing a fifths category, we therefore propose a generalised version of *isAscending* as an alternative that includes also those progressions where the second chord has a new root, but the root of the first chord is not contained in the second (Figure 5).

$$\begin{aligned} isAscending_2(C_1, C_2) := \quad & root(C_2) \notin pcs(C_1) \\ & \wedge\, pcs(C_1) \cap pcs(C_2) \neq \varnothing \end{aligned}$$

Figure 5. Ascending progression (generalised version): the root of the second chord is new, but both chords share common pitch classes

Following some speculation in Schoenberg's treatise, we also implemented a progression strength measurement that combines all the cases above in a single numeric measurement, and that for a more fine-grained discrimination additionally takes the cardinality of the harmonic band into account, weighted against the total number of chord pitch classes.

Finally, Figure 6 implements Schoenberg's recommendation that a descending progression is resolved as quasi a "passing chord". [3]

$$\begin{aligned} & resolveDescending(C_1, C_2, C_3) := \\ & \quad isDescending(C_1, C_2) \Rightarrow isAscending(C_1, C_3) \end{aligned}$$

Figure 6. Resolve descending progressions quasi as "passing chords"

These functions have been implemented as constraints in our system: they can be combined with other constraints and a solver can find one or more solutions. For efficiency, our constraint programming system uses

[3] Schoenberg recommends this strict version of the rule, a relaxed version also permits interchange progressions (e.g., $I - V - I$)

constraint propagation and dynamic variable orderings customised for this problem [1] (e.g., the solver progresses from "left to right" in score time but for simultaneous score objects always first determines rhythmic parameters, then scale or chord parameters and finally the actual note pitch classes and octaves).

4 RESULTS

This section provides musical results that have been generated by a system that implements the presented model. Figure 7 shows a chord progression that was generated with the proposed model. Whereas we only formalised Schoenberg's root progression guidelines in this paper, generating this example obviously required modelling further aspects of Schoenberg's theory such as part leading rules (e.g., avoid parallels, and keep the harmonic band in the same voice and octave), or the treatment of chord inversions.

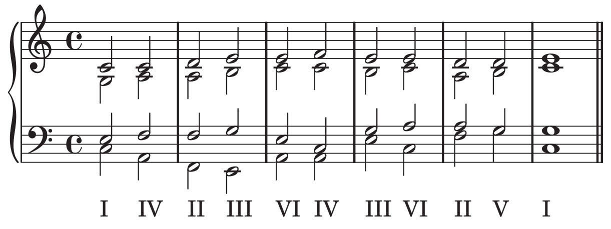


Figure 7. Chord progression generated by the presented model

Nevertheless, the sequence of the analytical chord objects is primarily controlled by the constraints presented above. Only few further constraints are applied on the analytical chord objects: all chords are diatonic chords in C major, the progression starts with the tonic I, and it ends in a cadence. Note that in this particular case it so happened that no descending progressions occurred at all. The number of superstrong progressions was explicitly restricted to 20 percent at maximum. The examples section of the Strasheela website (`http://strasheela.sourceforge.net`) contains a page with further results generated by the presented model, which also demonstrate other aspects of Schoenberg's theory. These examples are provided with full source code.

The presented model is highly flexible. It is applicable beyond the common four-voice setting, beyond the conventional triads, and is even suitable for microtonal music. The first author used this model for composing in 31-tone equal temperament, a temperament very close to quarter-comma meantone [5]. Figure 8 shows the beginning of a movement of "Harmony Studies", a 7-limit harmony cadence, which consists solely of ascending chord progressions. Remember that enharmonic spelling indicate different pitches in 31-tone equal temperament. While the interval $C - E\flat$ is the minor third (6/5), the interval $C - D\sharp$ is the subminor third (6/7). In order to assist deciphering the notation, also a harmonic analysis is provided: C harm 7 indicates the harmonic seventh chord over C ($4 : 5 : 6 : 7$, notated in meantone as $C, E, G, A\sharp$), while subharm 6 is a subharmonic sixth chord ($\frac{1}{4} : \frac{1}{5} : \frac{1}{6} : \frac{1}{7}$).

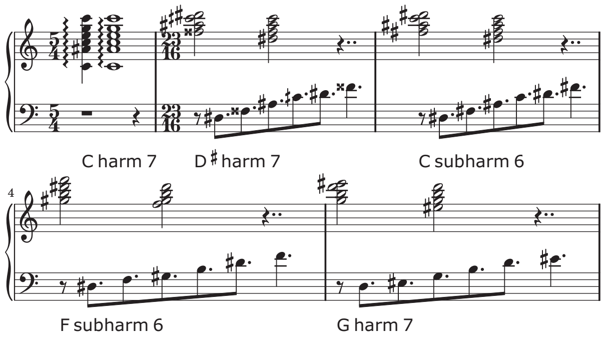


Figure 8. Beginning of a movement of "Harmony Studies", notated in 31-tone equal temperament (meantone)

5 SUMMARY

This paper detailed a formal model of Schoenberg's root progression guidelines, an important aspect of his Theory of Harmony. Instead of modelling his rules directly (as constraints on scale degree intervals between chord roots), we modelled his explanation of these rules (as constraints between chord pitch class sets and roots).

For chord progressions of diatonic triads in major – the context in which Schoenberg discusses his guidelines – Schoenberg's rules and the proposed model are equivalent. Our constraints can thus be used for implementing exercises proposed by Schoenberg's book as shown above.

However, the behaviour of our model and Schoenberg's rules differ for more complex cases. According to Schoenberg, a progression is superstrong if the root interval proceeds a step up or down. For example, the progression $V^7 - IV$ is superstrong according to Schoenberg. In the presented model, however, this progression is descending! The root of IV is contained in V^7 (e.g. in $G^7 - F$, the root pitch class F is already the sevenths of the preceding chord). Indeed, this progression is rare in music. By contrast, the progression $I - III\flat$ (e.g., $C - E\flat$) is a descending progression according to Schoenberg's rules. In our proposed model (the variant $isAscending_2$), this is an ascending progression (the root of $E\flat$ is not contained in C), and in our intuitive rating this progression does indeed feel strong.

6 REFERENCES

[1] Torsten Anders. *Composing Music by Composing Rules: Design and Usage of a Generic Music Constraint System*. PhD thesis, School of Music & Sonic Arts, Queen's University Belfast, 2007.

[2] Tosten Anders and Eduardo R. Miranda. A Survey of Constraint Programming Systems for Modelling Music Theories and Composition. *ACM Computing Surveys*, submitted.

[3] Kemal Ebcioglu. Report on the CHORAL Project: An Expert System for Harmonizing Four-Part Chorales. Technical Report Report 12628, IBM, Thomas J. Watson Research Center, 1987.

[4] Kemal Ebcioglu. An Expert System for Harmonizing Chorales in the Style of J.S. Bach. In Mira Balaban, Kemal Ebcioglu, and Otto Laske, editors, *Understanding Music with AI: Perspectives on Music Cognition*, chapter 12. MIT Press, 1992.

[5] Adriaan Daniël Fokker. Equal Temperament and the Thirty-one-keyed Organ. *The Scientific Monthly*, 81(4):161–166, 1955.

[6] François Pachet. The MusES system: an environment for experimenting with knowledge representation techniques in tonal harmony. In *First Brazilian Symposium on Computer Music, SBC&M '94*, Caxambu, Minas Gerais, Brazil, 1994.

[7] François Pachet and Pierre Roy. Musical Harmonization with Constraints: A Survey. *Constraints Journal*, 6(1):7–19, 2001.

[8] Harry Partch. *Genesis of a Music: An Account of a Creative Work, Its Roots and Its Fulfillments*. DaCapo Press, 2nd edition, 1974.

[9] Somnuk Phon-Amnuaisuk. *An Explicitly Structured Control Model for Exploring Search Space: Chorale Harmonisation in the Style of J.S. Bach*. PhD thesis, Centre for Intelligent Systems and their Application, Division of Informatics, University of Edinburgh, 2001.

[10] Rafael Ramirez and Julio Peralta. A constraint-based melody harmonizer. In *ECAI 98 Workshop on Constraints for Artistic Applications*, Brighton, 1998.

[11] Arnold Schoenberg. *Harmonielehre*. Universal Edition, Wien, 7th edition, 1986, orig. 1911. Rev. ed. 1922. Trans. by Roy Carter as *Theory of Harmony*. Berkeley and Los Angeles: University of California Press. 1978.

[12] Arnold Schoenberg. *Structural Functions of Harmony*. Faber and Faber, second revised edition, 1999, orig. 1969. Ed. Leonard Stein.

[13] C. P. Tsang and M. Aitken. Harmonizing music as a discipline of constraint logic programming. In *Proceedings of then International Computer Music Conference*, Montréal, 1991.

[14] Detlev Zimmermann. Modelling Musical Structures. *Constraints*, 6(1), 2001.

DISSONANCES: BRIEF DESCRIPTION AND ITS COMPUTATIONAL REPRESENTATION IN THE RTCC CALCULUS

Salim Perchy
AVISPA Research Group
Universidad Javeriana - Cali
ysperchy@cic.puj.edu.co

Gerardo M. Sarria M.
AVISPA Research Group
Universidad Javeriana - Cali
gsarria@cic.puj.edu.co

ABSTRACT

Dissonances in music have had a long evolution history dating back to days of strictly prohibition to times of enricheness of musical motives and forms. Nowadays, dissonances account for most of the musical expressiveness and contain a full application theory supporting their use making them a frequently adopted resource of composition. This work partially describes their theoretical background as well as their evolution in music and finally proposing a new model for their computational use.

1 THE CONSONANCE AND THE DISSONANCE

1.1 Cognitive Definition

The term *Consonance* in music is considered, from a psychological stance, a sound (i.e. interval in chord or arpeggio) that emits a sensation of ease or relaxation of the ear, something enjoyable. In contrast, *dissonance* is the opposite sensation, something confusing or aggressive to the ear [1].

Although the difference between this two terms may be well-defined with respect to tonal music, their meanings are profoundly attached to and varies depending on the culture, the music genre, and even the spoken language. For this, their cognitive definitions are relative and other factor come to fulfill their descriptions [2].

1.2 Physical Definition

Physically speaking, a *dissonance* may be conceived as the union of the acoustic waves that try to *destroy* themselves, *consonance* being the opposite physical phenomena[3].

[1] This interval was considered dissonant in early counterpoint

SMC 2009, July 23-25, Porto, Portugal

Interval	Ratio
1st	1:1
8ve	2:1
4th [1]	4:3
5th	3:2

Table 1. Perfect consonant intervals and their ratios

Mathematically speaking, the concept of consonance and dissonance is attached vastly to the **ratios** between the sound waves that conform the sound being executed. This ratio, for a consonance consideration, should be with low numbers as observed in table 1

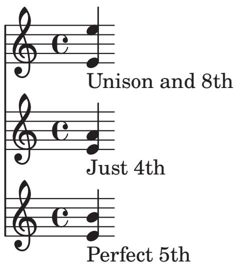


Figure 1. Perfect Consonant Intervals

In tonal music, the intervals considered consonant may be divided in *perfect* (figure 1) and *imperfect* (figure 2), *perfect* ones being simpler ratios than their counterpart. Table 2 shows some imperfect consonant intervals and their ratios. On the other hand, the dissonances are the intervals excluding the ones mentioned before (i.e. Minor 2nd and 7th).

Interval	Ratio
Major 3rd	5:4
Minor 3rd	6:5
Major 6th	5:3
Minor 6th	8:5

Table 2. Imperfect consonant intervals and their ratios

The reader may note that the concrete definition and dif-

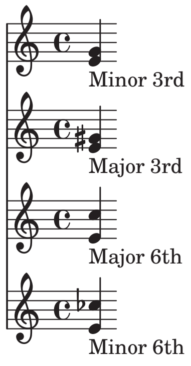


Figure 2. Imperfect Consonant Intervals

ferentiation of consonances and dissonances given above are purely tonal and based on western (common practice) music. Throughout time, the concept of consonance has evolved constantly giving way to before-considered dissonances being reclassified as consonances because of their frequent use, as noted in [4].

The choice of this particular vision, as the reader will see in the model proposed, of dissonance is purely convenient because of its strong theory background and documented use.

1.3 Brief historical evolution

The dissonance, as the majority of the compositional tools, has seen an evolution in which it has enriched itself from both theory and practice. Each important age in music history has contributed with its convention of use and theoretical and stylistic meaning, some more than other but always keeping a concrete advance.

Initially, the dissonance was non-existent in the sense of the avoidness of its use because the young age of harmony and its theory concepts targeted to enrich the consonance compendium.

In ancient Greece, the dissonance was not explored as a melodic resource although its meaning was already established, but yet, the Greeks considered the tritone (3 density chord) a disturbing sound, this being the base of the harmony used in the classical period. [5]

Farther ahead, the baroque age would consider intervals like the 3^{rd} and the 6^{th} consonant and extend their use to not just independent entities but also as artistic expressive means. At the dawn of the classical period the first direct dissonance is explored as a form with a well defined objective, this was the V^7 or *seventh dominant* chord.

At the arriving of the romantic and nationalism ages the use of dissonances was widely frequent and have defined purpose and process, primary helping the composer explore new concepts in the harmony theory.

With the establishment of the contemporary music, the dissonances have now the role of directing the harmony in actual composition, musicologists and composers turn to them in a daily basis.[5]

2 THE FUNCTION OF THE DISSONANCE

For an accurate computational definition of the dissonance we have to define its function, use and purpose in music. Also we have to bear in mind that the function is **relative** to each age or musical genre, but is keeps a stable base theory and form of use.

2.1 Harmonic Function

In defining a concrete function in the process of dissonance, there also has to be a definition of an inherent characteristic of musical harmony. The music literature, as normal literature, possesses moments of tension and relaxation, this simple characteristic allows to provide a purpose and at the same time a tool for the description of a given melody.

In the case of the dissonances, their harmonic function is no other than to create tension or confusion to the listener, this objective is more remarked seeing its aggressive character depending on the composer's thoughts, so it is this that sets its goal to the conversion of a melody to a point of obscurity and discomfort to the ear[3].

From the aforementioned, modern music has expanded its use not just to a tension-relaxation characteristic but also to an independent and complete auditive element that might represent an idea on its own[6].

2.2 Usage

The cycle of *tension-relaxation* can be seen as a sequential process that happens over time, each of the parts of this process may be seen as a well-defined component or agent (process with goal). The cycle is generally seen as this flow [7]:

$$\boxed{\text{Preparation}} \rightarrow \boxed{\text{Dissonance}} \rightarrow \boxed{\text{Resolution}}$$

Each stage contains unique and shared characteristics:

Preparation : Its function, as its name indicates, prepares the listener to the confusion of tension that the dissonance may generate in the melody. Generally, this preparation carries a harmonic line corresponding to its tonality.

Dissonance : In this stage, the dissonance or dissonances are produced, often in weak rhythmic beats and in strong ones depending on its relevance and sonority.

Resolution : Here, the dissonance need to move to a state of resolution or relaxation, it is here that the dissonance is carried to a more pleasing form, often taken to the main tonality on long beats.

The most evident and early example of this above process can be observed in the progression I – IV – V^7 – I (figure 3), in this the preparation consists of the first 2 chords and the dissonance is caused by the *dominant seventh*, lastly resolving in the tonic chord C.

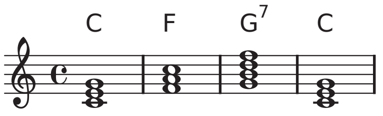


Figure 3. Dissonance using V^7

The usage of dissonances expands when the repertory of dissonant chords grow over time, this was also a crucial aspect of the evolution of this technique, chord like the *augmented sixth* or *Neapolitan sixth* make a wider space of sonority. An example can be seen a figure 4.

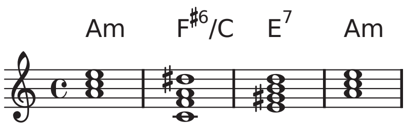


Figure 4. Dissonance enriched with the *German augmented sixth*

The next evolution in the use of dissonances is considering the resolution step as omissible or indefinitely postponed, often leaving many accumulated to an eventual resolution, *Frédérick Chopin* and *Richard Wagner* were amongst the main developers of evolution[5]. The illustration of these can be seen in figure 5.

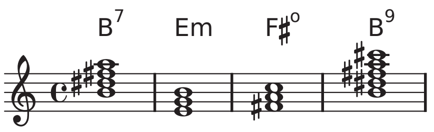


Figure 5. Dissonance without resolution

Modern music refines the concept of dissonance and transforms it when declaring it a unique entity, self-described and self-functional, making dissonances a complete form without the need of preparation or resolutions. A very recognized composer using this concept was *Claude Debussy*.

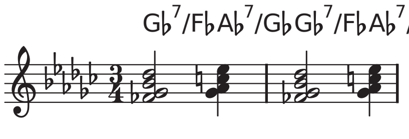


Figure 6. *Claude Debussy - "La fille aux cheveux de lin"*, measures 8 and 9. Dissonance as individual entity

3 THE REAL-TIME CONCURRENT CONSTRAINT CALCULUS

Concurrent constraint programming (ccp [8]) is a model for specifying concurrent systems in terms of constraints. A *constraint* is a formula representing partial information about the shared variables of the system. Examples of constraints are: $pitch_1 = 60$ or $pitch_2 > pitch_1 + 2$; If variables $pitch_1$ and $pitch_2$ are in the domain of MIDI values these constraints specify that $pitch_1$ must be C and $pitch_2$ must be at least a tone higher than $pitch_1$. The information about the shared variables resides in a *store*, which is, in fact, the conjunction of all the constraints applied to the variables. This store can be accessed by *agents* (processes who interact with the store) with two basic operations: **ask** and **tell**.

The *Real-Time Concurrent Constraint Calculus* (rtcc [9, 10]) is an extension of ccp developed to specify reactive systems with real-time behaviour. In reactive systems time is conceptually divided into *discrete intervals* (or *time units*). In a time interval, a process receives a stimulus from the environment, it computes (reacts) and responds to the environment. The computational processes of rtcc are summarized in table 3.

$$\begin{array}{lll} P, Q, \ldots & ::= & \textbf{tell}(c) \mid \sum_{i \in I} \textbf{when } c_i \textbf{ do } P_i \\ & \mid & P \parallel Q \mid \textbf{local } x \textbf{ in } P \\ & \mid & \textbf{unless } c \textbf{ next } P \\ & \mid & \textbf{catch } c \textbf{ in } P \textbf{ finally } Q \\ & \mid & \textbf{next } P \mid \textbf{!}P \mid \star P \end{array}$$

Table 3. rtcc Processes

Intuitively, the process **tell**(c) adds constraint c to the store within the current time unit. The ask process **when** c **do** P is generalized with a non-deterministic choice of the form $\sum_{i \in I}$ **when** c_i **do** P_i (I is a finite set of indices). This process, in the current time unit, must non-deterministically choose one of the P_j ($j \in I$) whose corresponding guard constraint c_j is entailed by the store, and execute it. The non-chosen processes are precluded. Two processes P and Q acting concurrently are denoted by the process $P \parallel Q$. In one time unit P and Q operate in parallel, communicating through the store by telling and asking information. The process **local** x **in** P declares a variable x private to P (hidden to other processes). This process behaves like P, except that all information about x produced by P can only be seen by P and the information about x produced by other processes is hidden to P. The weak time-out process, **unless** c **next** P, represents the activation of P the next time unit if c cannot be inferred from the store in the current time interval (i.e. $d \not\models c$). Otherwise, P will be discarded. The strong time-out process, **catch** c **in** P **finally** Q, represents the interruption of P in the current time interval when the store can entail c; otherwise, the execution of P continues. When process P is interrupted, process Q is executed. If P finishes, Q is discarded. The execution of a process P can be delayed one time unit with **next** P (P will be activated in the next time interval). The operator "**!**" is used to define infinite behaviour. The process $\mathbf{!}P$ represents $P \parallel \textbf{next}\ P \parallel \textbf{next}(\textbf{next}\ P) \parallel \ldots$, (i.e. $\mathbf{!}P$ executes P in the current time unit and it is replicated in the next time interval). An arbitrary (but finite) delay is represented with the operator "$\star$". The process $\star P$ represents an unbounded but finite $P + \textbf{next}\ P + \textbf{next}(\textbf{next}\ P) + \ldots$, (i.e. it allows to model asynchronous behaviour across the time intervals).

We write $\prod_{i \in I} P_i$, where $I = \{i_1, \ldots, i_n\}$ to denote the parallel composition of all the P_i, that is, $P_{i_1} \parallel \ldots \parallel P_{i_n}$. A bounded replication and asynchrony can be specified using summation and product. $!_I P$ and $\star_I P$ are defined as abbreviations for $\prod_{i \in I} \textbf{next}^i P$ and $\sum_{i \in I} \textbf{next}^i P$, respectively. For example, process $!_{[m,n]} P$ means that P is always active between the next m and $m + n$ time units.

The following simple example illustrates a computational model in `rtcc`:

In the case of changing pace of a song's natural timing such a *ritardando*, this behaviour can be modeled as:

$$\begin{aligned}&\mathbf{!}(\textbf{when}\ \mathit{ritardando} = \texttt{true}\ \textbf{do next tell}(\mathit{bpm} = 60))\\ &\quad \parallel \textbf{catch}\ \mathit{ritardando} = \texttt{true}\ \textbf{in}\ \mathbf{!}(\textbf{tell}(\mathit{bpm} = 150))\end{aligned}$$

Intuitively, this process states that the speed of a quarter note (or crotchet) will be 150 (with process $\mathbf{!}(\textbf{tell}(\mathit{bpm} = 150))$) until a *ritardando* signal is given (a presence of constraint $\mathit{ritardando} = \texttt{true}$ in the store). In the case of the signal is given, the process $\mathbf{!}(\textbf{tell}(\mathit{bpm} = 150))$ is interrupted and the speed will change to 60.

4 COMPUTATIONAL REPRESENTATION OF DISSONANCES

Since the dissonance phenomena in music can be seen as an ordered sequence of processes (as shown above), we can express it using concurrent agents that synchronize each other through signals (constraints) that are global to the whole system. Each agent may represent each phase in the dissonance process and also delay its execution until the previous (dependant) phase has been carried out and signals the system to continue the sequence onto the next phase. The model we propose is the following:

$$\begin{aligned}\mathit{Conductor}_{[n,m]} \stackrel{\text{def}}{=}\ & \mathit{Musician} \parallel \mathit{Cycle}_{[n,m]}\\ & \parallel \mathbf{!}(\textbf{tell}(\mathit{go} = 1))\\ & \parallel \star(\mathbf{!}\textbf{tell}(\mathit{stop} = 1))\\ & \parallel \mathbf{!}(\textbf{unless}\ \mathit{stop} = 1\\ & \qquad \textbf{next when}\ \mathit{end} = 1\\ & \qquad\quad \textbf{do}(\mathit{Musician} \parallel \mathit{Cycle}_{[n,m]}))\end{aligned}$$

$$\begin{aligned}\mathit{Cycle}_{[n,m]} \stackrel{\text{def}}{=}\ & \star\ (\textbf{tell}(\mathit{prep} = 1)\\ & \quad \parallel \star_{[1,n]}(\textbf{tell}(\mathit{diss} = 1)\\ & \qquad \parallel \star_{[1,m]}(\textbf{tell}(\mathit{res} = 1))))\end{aligned}$$

The main entry point of the model is the agent *Conductor*, this agent will activate the *Musician* and a process *Cycle* to motivate a dissonance. It also gives a signal to the musician for starting the melody ($\mathbf{!}(\textbf{tell}\ (\mathit{go} = 1))$ and eventually (some time in the future) it will give another signal to make the musician stop producing music ($\star(\mathbf{!}\textbf{tell}\ (\mathit{stop} = 1))$). Additionally, if the stop signal has not already given and the musician ends a dissonance, *Conductor* will activate the *Musician* and the process *Cycle* again (this could be seen as a loop for the musician to continue playing the melody and eventually to perform a dissonance until the *stop* signal).

The *Cycle* process posts the signals for each stage of the dissonance. Parameters n and m bound the time to change from one stage to the next.

The agent *Musician* is defined as follows:

$$\begin{aligned}\mathit{Musician} \stackrel{\text{def}}{=}\ & \textbf{when}\ \mathit{go} = 1\ \textbf{do}\\ & \quad \textbf{catch}\ \mathit{prep} = 1\ \textbf{in}\ \mathit{Melody}\\ & \qquad \textbf{finally}\ \mathit{Stage1}\end{aligned}$$

$$\begin{aligned}\mathit{Stage1} \stackrel{\text{def}}{=}\ & \textbf{catch}\ \mathit{diss} = 1\ \textbf{in}\ \mathit{Preparation}\\ & \quad \textbf{finally}\ \mathit{Stage2}\end{aligned}$$

$$Stage2 \overset{\text{def}}{=} \textbf{catch}\ res = 1\ \textbf{in}\ Dissonance \\ \textbf{finally}\ Stage3$$

$$Stage3 \overset{\text{def}}{=} Resolution \parallel \textbf{tell}(end = 1)$$

The $Musician$ will start executing the process $Melody$ (supposed to play the main melody of the whole song, may be through MIDI) waiting to catch the signal $prep = 0$ during it. When it catches the signal, it interrupts (stops) the $Melody$ and launches the $Stage1$ of the dissonance.

The same philosophy applies to the agents $Stage1$ and $Stage2$ each of them waiting for the signal that tells to carry on the next stage in the dissonance sequence, also assuming that process $Preparation$ plays the preparation and process $Dissonance$ executes the dissonance.

To conclude the sequence, the agent $Stage3$ waits for no signal, instead it launches the process $Resolution$ (also assumed to play a resolution congruent with the dissonance) and post a signal $end = 1$ telling the conductor that the current dissonance is over.

The process $Melody$ is the main harmonic structure the musician has planned for the song and is in charge of *evolving* the melody so to speak. Processes $Preparation$, $Dissonance$ and $Resolution$ will select non-determinalistically a chord to play from a set of chords specifically built to fulfill the process's task. For example, the set of chords from the process $Preparation$ is able to transcend to a dissonance and at this point the process $Dissonance$ will take the lead and the set of chords from where it will choose to play now will be dissonant ones.

These *chords set* may be constructed using a *relative distance* to the current tonality the melody is carrying. Using ranges over these distances a set chords can be discriminated to imply they belong to certain set. The *relative distance* of a certain chord is estimated using its notes' harmonic ratios against the root chord of the tonality and taking the same principle discussed above of deciding the degree of consonance and dissonance. Note that the dissonance concept vary in genre or music so the ranges used to make the chord sets are left for the musician using the model to decide.

5 CONCLUDING REMARKS

In this work, we described the concept of dissonance from various perspectives and also provided its mathematical relation with the consonances, we presented their musical evolution and its main function in the context of composing.

For the appropriate modeling of such problem, it was required that the usage of the dissonances be expressed in a sequential form because music is, as many more view it, a phenomenon occurring over time (melody) and concurrently (instruments or voices). Because of this, we chose the rtcc calculus, its concrete and direct way of treating time and how it manages asynchronous behaviour made possible the appropriate modeling of the dissonances as a non-deterministic process over time.

We also proposed a concurrent model that may be expanded or reduced easily to fit the management of the dissonance according to the need of the musician. The reader may see that any of the steps to make the sequence can be easily left out without affecting the integrity of the whole system. For example the musician may avoid the *resolution* step and leave all the dissonances unresolved or postpone it indefinitely using the operator $\star$.

We plan to pursue this work in a more practical direction. We have begun the implementation of an interpreter of rtcc. We are convinced that a software helps to better visualize the behaviour of systems, to make possible listening the audio results of the models in real-time, and to prove properties in those models. In the AVISPA research group [2] some interpreters and simulators have been developed for some other extensions of ccp such as ntcc and utcc (see for example [11, 12, 13]). This knowledge has been useful for the development of our interpreter. Initial work on the software has given us encouraging results.

6 REFERENCES

[1] D. Deustch, *The Psychology of Music, Second Edition (Cognition and Perception)*. 2nd Edition, Academic Press, 1998.

[2] R. Jourdain, *Music, The Brain, And Ecstasy: How Music Captures Our Imagination*. Harper Perenial, 1st Edition, 1998.

[3] D.C. Melnick, *Fullness of Dissonance: Modern Fiction and the Aesthetics of Music*. Fairleigh Dickinson Univ. Press, 1994.

[4] T. Harrison, *1910: The Emancipation of Dissonance*. University of California Press, Ilustrated Edition, 1996.

[5] A. Copland, *Cómo escuchar la Música*. Fondo de Cultura Económica, Spanish Edition - Bogotá, 1997.

[2] http://avispa.puj.edu.co/

[6] I. Xenakis, *Formalized Music: Thought and Mathematics in Music*. Pendragon Press, Revised Edition, 1992.

[7] A. Schoenberg, *Fundamentals of Musical Composition*. Faber and Faber Limited, 1970.

[8] V.A. Saraswat. *Concurrent Constraint Programming*. ACM Doctoral Dissertation Award. The MIT Press. Cambridge, MA, USA. 1993.

[9] G. Sarria and C. Rueda Real-Time Concurrent Constraint Programming. In *CLEI'2008*, Santa Fe, Argentina, 2008.

[10] G. Sarria. *Formal Models of Timed Musical Processes*. PhD Thesis, Universidad del Valle, Cali-Colombia, 2008.

[11] C. Rueda and G. Assayag and S. Dubnov A Concurrent Constraints Factor Oracle Model for Music Improvisation. In *CLEI'2006*, Santiago de Chile, 2006.

[12] M. Toro-Bermudez, C. Rueda, C. Agon, G. Assayag NTCCRT: A Concurrent Constraint Framework for Real-Time Interaction. In *ICMC'2009*, Montreal, Canada, 2009, To appear.

[13] C. Olarte and C. Rueda. A Declarative Language for Dynamic Multimedia Interaction Systems. *Communications in Computer and Information Science (CCIS)*, vol.38, Springer, 2009. To appear.

ALBUM AND ARTIST EFFECTS FOR AUDIO SIMILARITY AT THE SCALE OF THE WEB

Arthur Flexer[1]
[1]Austrian Research Institute
for Artificial Intelligence
arthur.flexer@ofai.at

Dominik Schnitzer[1,2]
[2]Department of Computational Perception
Johannes Kepler University Linz
dominik.schnitzer@jku.at

ABSTRACT

In audio based music recommendation, a well known effect is the dominance of songs from the same artist as the query song in recommendation lists. We verify that this effect also exists in a very large data set at the scale of the world wide web (> 250000). Since our data set contains multiple albums from individual artists, we can also show that the album effect is relatively bigger than the artist effect.

1 INTRODUCTION

In Music Information Retrieval, one of the central goals is to automatically recommend music to users based on a query song or query artist. This can be done using expert knowledge (e.g. `pandora.com`), social meta-data (e.g. `last.fm`), collaborative filtering (e.g. `amazon.com/mp3`) or by extracting information directly from the audio. In audio based music recommendation, a well known effect is the dominance of songs from the same artist as the query song in recommendation lists.

This effect has been studied mainly in the context of genre classification experiments. If songs from the same artist are allowed in both training and test sets, this can lead to over-optimistic results since usually all songs from an artist have the same genre label. It can be argued that in such a scenario one is doing artist classification rather than genre classification. One could even speculate that the specific sound of an album (mastering and production effects) is being classified. In [9] the use of a so-called "artist filter" ensuring that all songs from an artist are in either the training or the test set is proposed. The authors found that the use of such an artist filter can lower the classification results quite considerably (with one of their music collection even from 71% down to 27%). These over-optimistic accuracy results due to not using an artist filter have been confirmed in other studies [7] [1]. Other results suggest that the use of an artist filter not only lowers genre classification accuracy but may also erode the differences in accuracies between different techniques [2].

SMC 2009, July 23-25, Porto, Portugal

All these results were achieved on rather small data bases (from 700 to 15000). Often whole albums from an artist were part of the data base, maybe even more than one. These specifics of the data bases are often unclear and not properly documented. In extending these results, we analyse a very large data set at the scale of the world wide web (> 250000) with multiple albums from individual artists. We try to answer the following questions:

- Is there an album and artist effect even in very large data bases?
- Is the album effect larger than the artist effect?
- What is the influence of the size of a data base on music recommendation and classification?

2 DATA

For our experiments we used a data set $D(ALL)$ of $S = 254398$ song excerpts (30 seconds) from a popular web-shop selling music. The freely available preview song excerpts were obtained with an automated web-crawl. All meta information (artist name, album title, song title, genres) is parsed automatically from the hmtl-code. The excerpts are from $U = 18386$ albums from $A = 1700$ artists. From the 280 existing different hierarchical genres, only the $G = 22$ general ones on top of the hierarchy are being kept for further analysis (e.g. "Pop/General" is kept but not "Pop/Vocal Pop"). The names of the genres plus percentages of songs belonging to each of the genres are given in Tab. 1. Please note that every song is allowed to belong to more than one genre, hence the percentages in Tab. 1 add up to more than 100%. The genre information is identical for all songs on an album. The numbers of genre labels per albums are given in Tab. 2. Our data base was set up so that every artist contributes between 6 to 29 albums (see Tab. 3).

To study the influence of the size of the database on results, we created random non-overlapping splits of the entire data set: $D(1/2)$ - two data sets with mean number of song excerpts $= 127199$, $D(1/20)$ - twenty data sets with mean number of songs excerpts $= 12719.9$, $D(1/100)$ - one hundred data sets with mean number of songs excerpts $= 2543.98$. An artist with all their albums is always a member of a single data set.

Pop	Classical	Broadway
49.79	12.89	7.45
Soundtracks	Christian/Gospel	New Age
1.00	10.20	2.48
Miscellaneous	Opera/Vocal	Alternative Rock
6.11	3.24	27.13
Rock	Rap/Hip-Hop	R&B
51.78	0.98	4.26
Hard Rock/Metal	Classic Rock	Country
15.85	15.95	4.07
Jazz	Children's Music	International
6.98	7.78	9.69
Latin Music	Folk	Dance & DJ
0.54	11.18	5.24
Blues		
11.24		

Table 1. Percentages of songs belonging to the 22 genres with multiple membership allowed.

# labels	1	2	3	4	5 to 8
percentage	22.74	20.68	29.64	20.62	6.32

Table 2. Percentages of albums having 1,2,3,4 or 5 to 8 genre labels.

# albums	6	7	8	9	10
percentage	0.06	22.88	18.59	11.59	8.35
# albums	11	12	13	14	15
percentage	6.88	6.29	5.59	3.59	3.24
# albums	16	17	18	19	20 to 29
percentage	1.65	1.76	2.35	1.18	6.00

Table 3. Percentages of artists having 6, 7, ..., 20 to 29 albums.

3 METHODS

We compare two approaches based on different parametrisations of the data. Whereas Mel Frequency Cepstrum Coefficients (MFCCs) are a quite direct representation of the spectral information of a signal and therefore of the specific "sound" or "timbre" of a song, Fluctuation Patterns (FPs) are a more abstract kind of feature describing the amplitude modulation of the loudness per frequency band. It is our hypothesis, that MFCCs are more prone to pick up production and mastering effects of a single album as well as the specific "sound" of an artist (voice, instrumentation, etc).

3.1 Mel Frequency Cepstrum Coefficients and Single Gaussians (G1)

We use the following approach to music similarity based on spectral similarity. For a given music collection of songs, it consists of the following steps:

1. for each song, compute MFCCs for short overlapping frames
2. train a single Gaussian (G1) to model each of the songs
3. compute a similarity matrix between all songs using the symmetrised Kullback-Leibler divergence between respective G1 models

The 30 seconds song excerpts in mp3-format are recomputed to 22050Hz mono audio signals. We divide the raw audio data into non-overlapping frames of short duration and use Mel Frequency Cepstrum Coefficients (MFCC) to represent the spectrum of each frame. MFCCs are a perceptually meaningful and spectrally smoothed representation of audio signals. MFCCs are now a standard technique for computation of spectral similarity in music analysis (see e.g. [4]). The frame size for computation of MFCCs for our experiments was $46.4ms$ (1024 samples). We used the first 25 MFCCs for all our experiments.

A single Gaussian (G1) with full covariance represents the MFCCs of each song [5]. For two single Gaussians, $p(x) = \mathcal{N}(x; \mu_p, \Sigma_p)$ and $q(x) = \mathcal{N}(x; \mu_q, \Sigma_q)$, the closed form of the Kullback-Leibler divergence is defined as [10]:

$$KL_N(p\|q) = \frac{1}{2}\Bigg(\log\left(\frac{\det(\Sigma_p)}{\det(\Sigma_q)}\right) + Tr\left(\Sigma_p^{-1}\Sigma_q\right) + (\mu_p - \mu_q)'\,\Sigma_p^{-1}(\mu_q - \mu_p) - d\Bigg) \tag{1}$$

where $Tr(M)$ denotes the trace of the matrix M, $Tr(M) = \Sigma_{i=1..n} m_{i,i}$. The divergence is symmetrised by computing:

$$KL_{sym} = \frac{KL_N(p\|q) + KL_N(q\|p)}{2} \tag{2}$$

3.2 Fluctuation Patterns and Euclidean Distance (FP)

Fluctuation Patterns (FP) [6] [8] describe the amplitude modulation of the loudness per frequency band and are based on ideas developed in [3]. For a given music collection of songs, computation of music similarity based on FPs consists of the following steps:

1. for each song, compute a Fluctuation Pattern (FP)
2. compute a similarity matrix between all songs using the Euclidean distance of the FP patterns

Closely following the implementation outlined in [7], an FP is computed by: (i) cutting an MFCC spectrogram into three second segments, (ii) using an FFT to compute amplitude modulation frequencies of loudness (range $0 - 10Hz$) for each segment and frequency band, (iii) weighting the modulation frequencies based on a model of perceived fluctuation strength, (iv) applying filters to emphasise certain patterns and smooth the result. The resulting FP is a 12 (frequency bands according to 12 critical bands of the Bark scale [11]) times 30 (modulation frequencies, ranging from 0 to $10Hz$) matrix for each song. The distance between two FPs i and j is computed as the Euclidean distance:

$$D(FP^i, FP^j) = \sum_{k=1}^{12} \sum_{l=1}^{30} (FP^i_{k,l} - FP^j_{k,l})^2 \qquad (3)$$

4 RESULTS

4.1 Album/Artist Precision

For the full data base $D(ALL)$

For every song in the data base $D(ALL)$, we computed the first nearest neighbour for both methods G1 and FP. For method G1, the first nearest neighbour is the song with minimum Kullback Leibler divergence (Equ. 2) to the query song. For method FP, the first nearest neighbour is the song with minimum Euclidean distance of the FP pattern (Equ. 3) to the query song. We then computed the percentage of instances, where the first nearest neighbour is from the same album (1st AL) or from other albums by the same artist (1st AR) as the query song (see Tab. 4).

For method G1, 27.87% are from the same album and 35.76% from other albums by the same artist. On average, there are 13.46 songs on an album and 131.2 songs from one artist. Considered that there are always more than 250000 songs from other artists, it is quite astonishing that only in 36.37% a song from a different artist turns up as a first nearest neighbour. For method FP, percentages are quite lower with only 2.24% from the same album and 26.85% from other albums by the same artist.

Next we computed the album and artist precision at n. Album precision at n (AL prec) is the percentage of songs from the album in a list of the n nearest neighbours, with n being equal to the number of other songs in the same album as the query song. Artist precision at n (AR prec) is the percentage of songs from the artist in a list of the n nearest neighbours, with n being equal to the number of other songs from the same artist as the query song. For $D(ALL)$ and method G1, album precision is at 13.86% and artist precision at 8.14% (see Tab. 4). Precision values for method FP are very small.

Method	1st AL	1st AR	AL prec	AR prec
G1	27.87	35.76	13.86	8.14
FP	2.24	26.85	0.90	1.63

Table 4. Percentage of first nearest neighbour from same album (1st AL), from other albums from same artist (1st AR), album and artist precision (AL prec, AR prec) for G1 and FP.

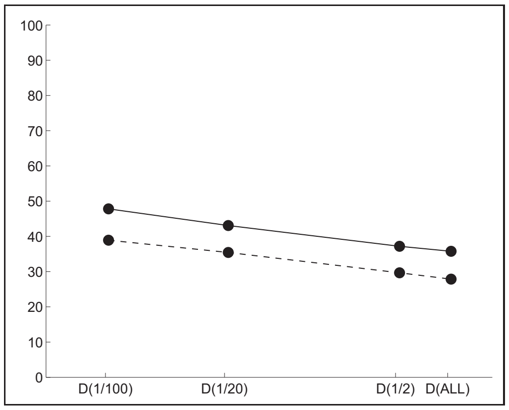


Figure 1. Percentage (y-axis) of **first nearest neighbour** from same album (dashed line), from other albums from same artist (solid) for **G1** and different size of data set (x-axis, log-scale).

To sum up, there is both an album and an artist effect in nearest neighbour based music recommendation for method G1. For this timbre based method, the album effect is even relatively bigger than the artist effect. For method FP, there is only a smaller artist effect but no album effect.

Influence of the size of the data base

We repeated the experiments for all the subsets of the data base as described in Sec. 2. The results depicted in Figs. 1, 2, 3 and 4 show mean values over 100 ($D(1/100)$), 20 ($D(1/20)$), 2 ($D(1/2)$) data sets or the respective single result for the full data set $D(ALL)$. The percentage of the first nearest neighbour from the same album decreases from 38.91% for $D(1/100)$ to 27.82% for $D(ALL)$ for method G1 (Fig. 1). There is a parallel decrease for the first nearest neighbour from other albums from the same artist for

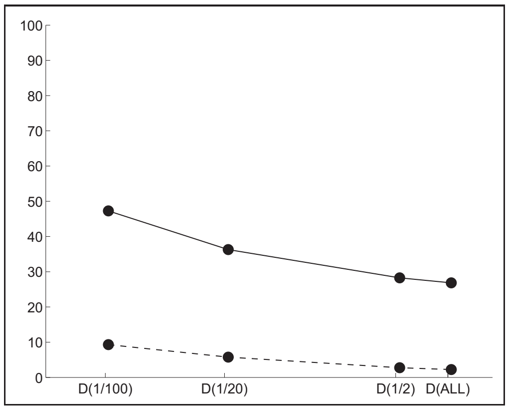


Figure 2. Percentage (y-axis) of **first nearest neighbour** from same album (dashed line), from other albums from same artist (solid) for **FP** and different size of data set (x-axis, log-scale).

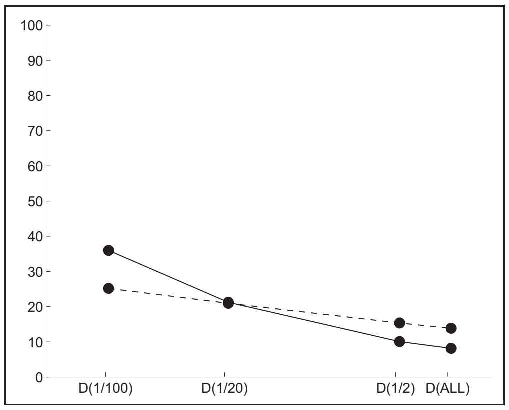


Figure 3. **Precision** (y-axis) of album (dashed line) and artist (solid) for **G1** and different size of data set (x-axis, log-scale).

method G1 (Fig. 1). A similar decrease at lower levels can be seen for method FP (Fig. 2). As the data sets get larger, the probability that songs from other artists are more similar to the query song than songs from the same album or artist, clearly seems to increase.

Album and artist precision also decrease with increasing size of data set. For method G1, artist precision drops from 35.99% for $D(1/100)$ to 8.14% for $D(ALL)$ even falling below album precision (Fig 3). For method FP, artist precision drops from 19.19% for $D(1/100)$ to 1.63% for $D(ALL)$ which is at the same low level as album precision (Fig. 4).

To sum up, both first nearest neighbour rates and precision values are over estimated when smaller data sets are used.

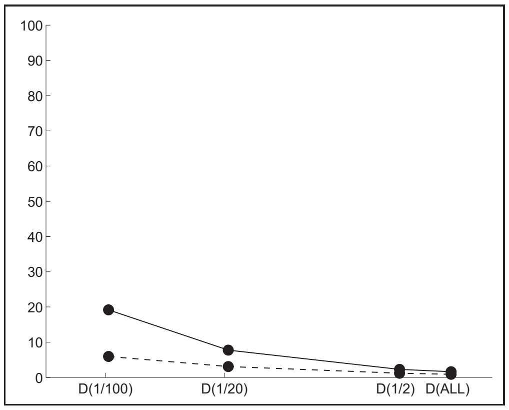


Figure 4. **Precision** (y-axis) of album (dashed line) and artist (solid) for **FP** and different size of data set (x-axis, log-scale).

4.2 Genre Classification

For the full data base $D(ALL)$

We also did experiments on the influence of album and artist filters on genre classification performance. We used nearest neighbour classification as a classifier. For every song in the data base $D(ALL)$, we computed the first nearest neighbour for both methods G1 and FP. For method G1, the first nearest neighbour is the song with minimum Kullback Leibler divergence (Equ. 2) to the query song. For method FP, the first nearest neighbour is the song with minimum Euclidean distance of the FP pattern (Equ. 3) to the query song. When using an album filter (ALF), all other songs from the same album as the query song were excluded from becoming the first nearest neighbour. When using an artist filter (ARF), all other songs from the same artist as the query song were excluded from becoming the first nearest neighbour. When using no filter (NOF), any song was allowed to become the first nearest neighbour. To estimate genre classification accuracy, the genre label of a query song s_{query} and its first nearest neighbour s_{nn} were compared. The accuracy is defined as:

$$acc(s_{query}, s_{nn}) = \frac{|(g_{query} \cap g_{nn})|}{max(|g_{query}|, |g_{nn}|)} \tag{4}$$

with g_{query} (g_{nn}) being a set of all genre labels for the query song (nearest neighbour song) and $|.|$ counting the number of members in a set. Therefore accuracy is defined as the number of shared genre labels divided by the maximum set size of g_{query} and g_{nn}. The latter is done to penalise nearest neighbour songs with high numbers of genre labels. The range of values for accuracy is between 0 and 1. The baseline accuracy achieved by always guessing the three most probable genres (“Rock”, “Pop”, “Alternative Rock”, see Tab. 1) is 37.03%. We decided to use a number of three genres for this baseline accuracy because the majority of songs is labelled with three genres (see Tab. 2). Average accuracy results for methods G1 and FP are given in Tab. 5. Without using any filter (NOF), G1 clearly outperforms FP (70.69% vs. 46.97%). Using an album filter (ALF) strongly degrades the performance of G1 down to 58.49%, but hardly impairs method FP. Using an artist filter (ARF) further degrades the performance of G1 but also of FP. The difference between G1 and FP is now much closer (39.56% vs. 32.38%). However, method G1 barely outperforms the baseline accuracy of 37.03% and method FP clearly falls below it.

Method	NOF	ALF	ARF
G1	70.69	58.49	39.56
FP	46.97	45.69	32.38

Table 5. Average accuracies for G1 and FP without (NOF) and with album filter (ALF) and artist filter (ARF).

To sum up, not using any filter yields very over-optimistic accuracy results. As a matter fact, results after artist filtering are very close or even below baseline accuracy. There is both an album and an artist filter effect for G1. There is only an album filter effect for FP. Using filters diminishes the differences in accuracies between methods G1 and FP, since filters have a bigger impact on G1 than FP.

Influence of the size of the data base

We repeated the experiments for all the subsets of the data base as described in Sec. 2. The results depicted in Figs. 5 and 6 show mean accuracy values over 100 ($D(1/100)$), 20 ($D(1/20)$), 2 ($D(1/2)$) data sets or the respective single result for the full data set $D(ALL)$. For both methods G1 and FP, the accuracy without using a filter (dotted lines in Figs. 5 and 6) decreases with increasing size of data set. For G1, from 81.66% for $D(1/100)$ to 70.69% for $D(ALL)$. For FP, from 59.24% for $D(1/100)$ to 46.79% for $D(ALL)$. There is an almost parallel decrease in accuracy when using album filters (dashed lines in Figs. 5 and 6). For both methods G1 and FP, the accuracy when using an artist filter (solid lines in Figs. 5 and 6) increases with increasing size of data set. For G1, from 31.28% for $D(1/100)$ to 39.56% for $D(ALL)$. For FP, from 27.19% for $D(1/100)$ to 32.38% for $D(ALL)$.

How can this contrary behaviour of decreasing accuracy for no filter and album filter versus increasing accuracy for artist filters be explained? Larger data sets allow for a larger choice of songs to become the first nearest neighbour. This larger choice of songs can come with the wrong or correct genre labels. If we use artist filters, this larger choice seems to make it more probable that a song with the correct genre label is first nearest neighbour. Otherwise we would not

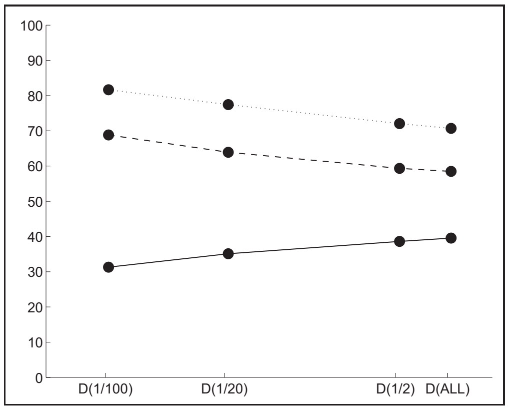


Figure 5. **Accuracy** (y-axis) for no filter (dotted line), album filter (dashed line), artist filter (solid) for **G1** and different size of data set (x-axis, log-scale).

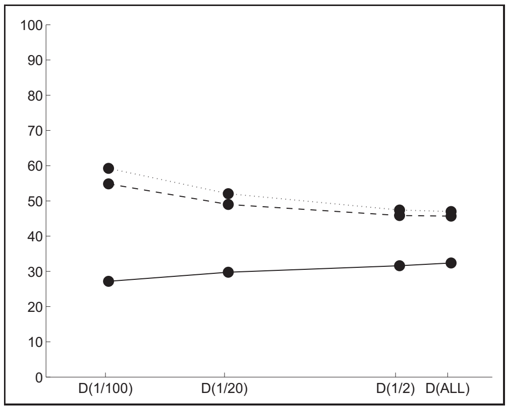


Figure 6. **Accuracy** (y-axis) for no filter (dotted line), album filter (dashed line), artist filter (solid) for **FP** and different size of data set (x-axis, log-scale).

see the increase in accuracy. If we use no filter or only an album filter, the larger choice seems to interfere with the songs from the same artist still in the data base. Songs from the larger choice sometimes end up being first nearest neighbour instead of a song from the same artist as the query song. Since most songs from an artist share the same labels, the larger choice in this case diminishes the accuracy.

To sum up, there clearly is an influence of the data base size on accuracy performance. Small data sets are too pessimistic when artist filters are used. But they are over optimistic if no or only album filters are used.

5 CONCLUSION

There clearly is both an album and an artist effect in music recommendation even in very large data bases. For the timbre based method G1, about one third of the first recommendations are from the same album and about another third from other albums from the same artist as the query song. Considering that every artist has multiple albums in the data base and that an album contains only about 13 songs on average, the album effect is relatively bigger than the artist effect. This suggests that the direct representation of the spectral information is sensitive to production and mastering effects of individual albums. For method FP, there is only a smaller artist effect but no album effect. This suggests that the more abstract signal representation of the fluctuation patterns is not sensitive to production and mastering effects of individual albums. But it is still able to model the common musical sound of an artist across different albums. Please note that we have no way to know whether an artist is working together with the same recording studio or sound engineer for more than one album. Our experiments also show that album and artist effects in music recommendations are over estimated when smaller data sets are being used.

Since most research on artist filters so far concentrated on genre classification, we did large scale experiments on classification accuracy also. We corroborated earlier results that not using any filter yields very over-optimistic accuracy results. Using artist filters even reduces results close to or even below baseline accuracy. As reported before, using artist filters also diminishes the differences in accuracies between methods that are effected distinctly by filtering. Additionally, there clearly is an influence of the data base size on accuracy performance.

As with all large scale performance studies, there remains the question as to how representative and universally valid our results are. We are convinced that our data base is representative of music that is generally listened to and available in the Western hemisphere since it is a large and random subset of about 5 million songs from a popular web-shop. As to the methods employed, we chose one method that closely models the audio signal and one that extracts information on a somewhat higher level. It is our guess that other method's performance will be close to either of our methods depending on their level of closeness to the analysed audio signal. The choice of our methods was also influenced by considerations of computability. After all, 250000 song excerpts are a lot of data to analyse and both our methods can be implemented very efficiently. Using nearest neighbour methods for music recommendation seemed to be the obvious choice.

With audio based music recommendation maturing to the scale of the web, our work provides important insight into the behavior of music similarity for very large data bases. Even with hundreds of thousands of songs, album and artist filtering remain an issue.

ACKNOWLEDGEMENTS: Parts of this work have been funded by the "Austrian National Science Foundation (FWF)" (Projects P21247 and L511-N15) and by the "Austrian Research Promotion Agency (FFG)" (Bridge-project 815474).

6 REFERENCES

[1] Flexer A.: Statistical Evaluation of Music Information Retrieval Experiments, Journal of New Music Research, Vol. 35, No. 2, pp.113-120, 2006.

[2] Flexer A.: A closer look on artist filters for musical genre classification, in Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07), Vienna, Austria, 2007.

[3] Fruehwirt M., Rauber A.: Self-Organizing Maps for Content-Based Music Clustering, Proceedings of the Twelth Italian Workshop on Neural Nets, IIAS, 2001.

[4] Logan B.: Mel Frequency Cepstral Coefficients for Music Modeling, Proceedings of the International Symposium on Music Information Retrieval (ISMIR'00), 2000.

[5] Mandel M.I., Ellis D.P.W.: Song-Level Features and Support Vector Machines for Music Classification, *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05)*, London, UK, September 11-15, 2005.

[6] Pampalk E.: Islands of Music: Analysis, Organization, and Visualization of Music Archives, MSc Thesis, Technical University of Vienna, 2001.

[7] Pampalk E.: Computational Models of Music Similarity and their Application to Music Information Retrieval, Vienna University of Technology, Austria, Doctoral Thesis, 2006.

[8] Pampalk E., Rauber A., Merkl D.: Content-based organization and visualization of music archives, Proceedings of the 10th ACM International Conference on Multimedia, Juan les Pins, France, pp. 570-579, 2002.

[9] Pampalk E., Flexer A., and Widmer G.: Improvements of Audio-Based Music Similarity and Genre Classification , Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05), London, UK, September 11-15, pp.628-633, 2005.

[10] Penny W.D.: Kullback-Liebler Divergences of Normal, Gamma, Dirichlet and Wishart Densities, *Wellcome Department of Cognitive Neurology*, 2001.

[11] Zwicker E., Fastl H.: Psychoaccoustics, Facts and Models, Springer Series of Information Sciences, Volume 22, 2nd edition, 1999.

EXTENDING THE FOLKSONOMIES OF FREESOUND.ORG USING CONTENT-BASED AUDIO ANALYSIS

Elena Martínez, Òscar Celma, Mohamed Sordo, Bram de Jong, Xavier Serra
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
elena.martinez@openbravo.com, ocelma@bmat.com, mohamed.sordo@upf.edu, bdejong@iua.upf.edu, xavier.serra@upf.edu

ABSTRACT

This paper presents an in–depth study of the social tagging mechanisms used in *Freesound.org*, an online community where users share and browse audio files by means of tags and content–based audio similarity search. We performed two analyses of the sound collection. The first one is related with how the users tag the sounds, and we could detect some well–known problems that occur in collaborative tagging systems (i.e. polysemy, synonymy, and the scarcity of the existing annotations). Moreover, we show that more than 10% of the collection were scarcely annotated with only one or two tags per sound, thus frustrating the retrieval task. In this sense, the second analysis focuses on enhancing the semantic annotations of these sounds, by means of content–based audio similarity (autotagging). In order to "autotag" the sounds, we use a k–NN classifier that selects the available tags from the most similar sounds. Human assessment is performed in order to evaluate the perceived quality of the candidate tags. The results show that, in 77% of the sounds used, the annotations have been correctly extended with the proposed tags derived from audio similarity.

1 INTRODUCTION

Since 2004, collaborative tagging seems a natural way for annotating objects, in contrast to using predefined taxonomies and controlled vocabularies. Internet sites with a strong social component (e.g. *last.fm*, *flickr*, and *del.icio.us*), allow users to tag web objects according to their own criteria. The tagging process can improve then, content organization, navigation, search and retrieval tasks [9].

Nowadays, in the multimedia domain, *prosumers* hold an important role. The term comes from producing and consuming at the same time: they create and annotate a vast amount of data. In fact, audiovisual assets can be manually and automatically described. On the one hand, users can organize their music collection using personal tags like: *late night*, *while driving*, *love*. On the other hand, content–based (CB) audio annotation can propose, with some confidence degree, audio related tags such as: *pop*, *acoustic guitar*, or *female voice*. It is clear that both approaches create a rich tag cloud representing the actual content. Still, automatic annotation based solely on CB cannot bridge the Semantic Gap. Hybrid approaches, exploiting both the wisdom of crowds and automatic content description, are needed in order to close the gap. In this sense, *Freesound.org*, a collaborative sound database, contains both elements: it allows users to annotate sounds, and they can also browse similar sounds to a given one, according to audio similarity. However, there are some sounds that are scarcely annotated, thus frustrating their retrieval using keyword–based search.

The main goal of this paper is to enhance semantic annotations in the *Freesound.org* sound collection, by means of content–based audio similarity. We propose an approach to "autotag" sounds based on the tags available in their most similar sounds.

SMC 2009, July 23-25, Porto, Portugal

2 COLLABORATIVE TAGGING

One of the most interesting aspects of collaborative tagging is that the whole community benefits from sharing information [17]. However, "collective tagging has also the potential to aggravate the problems associated with the fuzziness of linguistic and cognitive boundaries" [7]. Users' contributions produce a huge classification system that consists in an idiosyncratically personal categorization. Some of the main problems concerning collaborative tagging are: polysemy, synonymy and data scarcity. Furthermore, spelling errors, plurals and parts of speech also clearly affect a tagging system.

Sometimes, polysemous tags can return undesireable results. For example, in a music collection if one is searching using the tag *love*, the results can contain both love songs, and songs that users like it very much (i.e. a user that loves a *death metal Swedish* song, not related with the love theme).

Tag synonymy is also an interesting problem. Even though it enriches the vocabulary, it presents also inconsistencies among the terms used in the annotation process. For example, *bass drum* sounds can be annotated with the *kick drum* tag; but these sounds will not be returned when searching for *bass drum*. To avoid this problem, sometimes users tend to add redundant tags to facilitate the retrieval (e.g. using *synth*, *synthesis*, and *synthetic* for a given sound excerpt). Yet, there are some approaches to measure semantic relatedness between tags [3]. These metrics could be used to decrease the size of the vocabulary, and also for (automatic) query expansion to increase the recall in the sound retrieval task.

Finally, the scarcity and inequality nature of a collaborative annotation process—where usually a few sounds are well annotated, and the rest contain very few tags—limits the coverage retrieval of a collection.

3 RELATED WORK

In [16], the authors propose a query–by–semantic audio information retrieval system. The proposed system can learn the relationships between acoustic information and words (tags) from a manually annotated audio collection. The learning task is based on a supervised multiclass labeling model, with a multinomial distributions of words over a predefined vocabulary.

Torres et. al propose a method to construct a musically meaningful vocabulary [15]. By means of acoustic correlation using canonical component analysis (sparse CCA), they can remove from the vocabulary those noisy words (not related with the actual audio content) that have been inconsistently used by human annotators.

The *bag–of–frames* (BOF) approach has been extensively used to describe timbrical properties of an audio signal. This approach is used to extract mid–level descriptions from music signals, such as their genre or instrument, but it is also used to perform timbre similarity between songs. In [1], the authors find out that this approach tends to generate false positives songs which are irrelevantly close to many other songs. These songs are called hubs, and the authors propose measures to quantify the "hubness" of a given song. This property affects any system that uses timbrical features to compute content–based audio similarity.

Cano has studied the strengths and limitations of audio fingerprinting, and suggests that it can be extended to allow content–based similarity search, such as finding similar sounds using query–by–example [2]. Similarly to our approach, [14] proposes a non–parametric strategy for automatically tagging songs, using content–based audio similarity to propagate tags from annotated songs to similar, non–annotated, songs.

In [5], the authors present a method to recommend tags to unlabeled songs. Automatic tags are computed by means of a set of boosted classifiers (Adaboost), in order to provide tags to tracks poorly (or not) annotated. This method allows music recommenders to include in a playlist unheard music that otherwise would be missed, enhancing the novelty component of the recommendations.

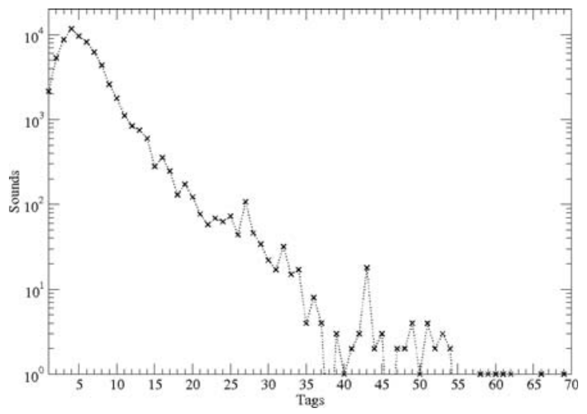


Figure 1. A linear–log plot depicting the number of tags per sound. Most of the sounds are annotated using 3–5 tags, and only a few sounds are annotated with more than 40 tags.

4 THE FREESOUND.ORG COLLECTION

Freesound.org is a collaborative sound database where people from different disciplines share recorded sounds and samples under the Creative Commons license, since 2005. The initial goal was to giving support to sound researchers, who often have trouble finding large sound databases to test their algorithms. After four years since its inception, *Freesound.org* serves more than 23,000 unique visits per day. Also, there is an engaged community—with almost a million registered users—accessing more than 66,000 uploaded sounds.

Yet, only few dozens of users uploaded hundreds of sounds, whilst the rest uploaded just a few. In fact, 80% of the users uploaded less than 20 sounds, and only 8 users uploaded more than one thousand sounds each. It is worth noting that these few users can highly influence the overall sound annotation process.

4.1 Tag behaviour

In this section we provide some insights about the tag behaviour and user activity in the *Freesound.org* community. We are interested in analyzing how users tag sounds assets, as well as the concepts used when tagging. The data, collected during March 2009, consists of around 66,000 sounds annotated with 18,500 different tags

Figure 1 shows the number of tags used to annotate the audio samples. The x-axis represent the number of tags used per sound. We can see that most of the sounds are annotated using 3–5 tags. Also, around 7,500 sounds are insufficiently annotated using only 1 or 2 tags. These sounds represent more than 10% of the whole collection. It would be desirable, then, to—automatically—recommend relevant tags to these scarcely annotated sounds, enhancing their descriptions. This is the main goal of the experiments presented in section 5.

Interestingly enough, in [2], the author analyzed a sound effects database, which was annotated by only one expert. A similar histogram distribution to the one presented in Figure 1 was obtained. Specifically, most of the sounds were annotated by the expert using 4 or 5 tags, as it is our case. This could be due to human memory constraints when assigning words to sounds or to any object, in order to describe them [11]. Based on Figure 1, we classify the sounds in three different categories, according to the number of tags used. Table 1 shows the data for each class.

Table 1. Sound–tag classes and the number of sounds in each category.

	Tags per sound	Sounds
Class I	1–2	7,481
Class II	3–8	42,757
Class III	> 8	7,148

Tag frequency distribution is presented in Figure 2. The x-axis refers to the 18,500 tags used, ranked by descending frequency. On the one hand, 44% of the tags were applied only once. This reflects the subjectivity of the tag process. Thus, retrieving these sounds in the heavy tail area is nearly impossible using only tag–based search (to overcome this problem, *Freesound.org* offers a content–based audio similarity search to retrieve similar sound samples). On the other hand, just 27 tags were used to annotate almost the 70% of the whole collection. The best fit of the tag distribution is obtained with a log–normal function, $\frac{1}{x}e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}$, with parameters mean of log $\mu = 1.15$, and standard deviation of log, $\sigma = 1.46$ [4].

The top–5 most frequent tags are presented in Table 2, and it gives an idea about the nature of the sounds available in the *Freesound.org* collection. Field–recording is the most frequent tag used to describe 6,787 different sounds. All these frequent tags are very informative when describing the sounds, in contrast to the photo domain in *flickr.com*, were popular tags are considered too generic to be "useful" [13].

Table 2. Top–5 most frequent tags from Figure 2.

Rank	Tag	Frequency
1	field–recording	6,787
2	noise	5,650
3	loop	5,487
4	electronic	4,329
5	synth	4,307

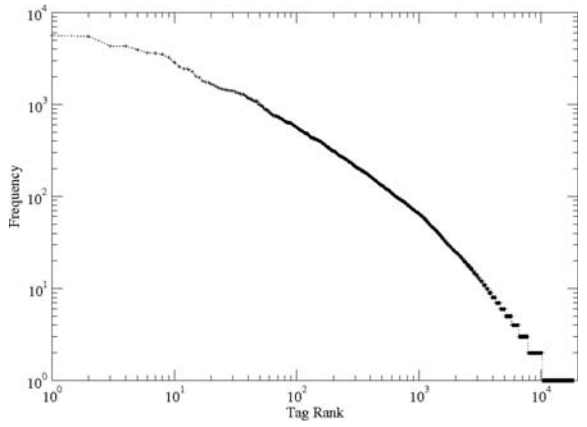


Figure 2. A log–log plot showing the tag distribution in *Freesound.org*. The curve follows a log–normal distribution, with mean of log $\mu = 1.15$, and standard deviation of log, $\sigma = 1.46$.

4.2 Tag categorization

In order to understand the vocabulary that the *Freesound.org* community uses when tagging sounds, we mapped the 18,500 different tags to broad categories (hypernyms) in the Wordnet [1] semantic lexicon. In some cases, a given tag matches multiple entries, so we bound the tag (noun or verb) to the highest ranked category. The selected Wordnet categories are: *(i)* artefact or object, *(ii)* organism, being, *(iii)* action or event, *(iv)* location, and *(v)* attribute or relation. Yet, 20.3% of the tags remain unclassified.

Most of the tags (38%) are related with objects (e.g. *seatbelt*, *printer*, *missile*, *guitar*, *snare*, etc.), or about the qualities and attributes of the objects (30%); such as state attributes (*analog*, *glitch*, *scratch*), or magnitude relation characteristics (*bpm*). Then, some tags (19%) are classified as an action (*hiss*, *laugh*, *glissando*, *scream*, etc.), whilst 11% are related with organisms (*cat*, *brass band*, etc.). Finally, only a few tags (2%) were bound to locations (e.g. *iraq*, *vietnam*, *us*, *san francisco*, *avenue*, *pub*, etc.). Therefore, we conclude that the tags are mostly used to describe the objects that produce the sound, and the characteristics of the sound. In this case, the wisdom of crowds concords with the studies of [12] and [6]. The former study focused on the attributes of the sound itself without referencing the source causing it (e.g *pitchiness*, *brightness*), while the latter introduced a taxonomy of sounds, on the assertion that they are produced by means of interaction of materials.

[1] `http://wordnet.princeton.edu/`

5 EXPERIMENTS

Our goal is to evaluate the quality of the recommended tags, for some specific sounds available in *Freesound.org*. By means of content–based audio similarity, our algorithm selects a set of candidate tags for a given sound (autotagging process). Then, the evaluation process is based on human assessment. Three subjects validated each candidate tag for all the sounds in the test dataset.

5.1 Dataset

The sounds selected for the experiments were a subset of the Class I (see Table 1). We selected those sounds whose tags' frequency was very low (i.e. rare tags, in the ranking of $\sim 10^4$ in Figure 2). In fact, all the sounds which were annotated with one tag whose frequency was equal to 1 were selected. Also, for the sounds annotated with 2 tags, we selected those which had at least one tag with frequency 1. The test dataset for the experiments consists of 260 sounds. The goal here is to automatically extend the annotation of these sounds, unsufficiently annotated with one or two very rare tags.

5.2 Nearest–neighbor classifier

We used a nearest neighbor classifier (k–NN, $k = 10$) to select the tags from the most similar sounds of a given sound. The choice of a memory–based nearest neighbor classifier avoids the design and training of every possible tag. Another advantage of using an NN classifier is that it does not need to be redesigned nor trained whenever a new class of sounds is added to the system. The NN classifier needs a database of labeled instances and a similarity distance to compare them. An unknown sample will borrow the metadata associated with the most similar registered sample.

Based on the results from [2], the similarity measure used is a normalized Manhattan distance of audio features belonging to three different groups: a first group gathering spectral and temporal descriptors included in the MPEG-7 standard [10]; a second one built on Bark Bands perceptual division of the acoustic spectrum, using the mean and variance of relative energies for each band; and, finally a third one, composed of Mel-Frequency Cepstral Coefficients (20) and their corresponding variances [8]. The normalized Manhattan distance of the above enumerated features is:

$$d(x, y) = \sum_{k=1}^{N} \frac{|x_k - y_k|}{(max_k - min_k)} \qquad (1)$$

where x and y are the vectors of audio features, N the dimensionality of the feature space, and max_k and min_k the maximum and minimum values of the *k–th* feature.

5.3 Procedure

Our technique for calculating the candidate tags consists on finding the *10–th* most similar sounds from the *Freesound.org* database, for a given seed sound of the test dataset. That is, given a seed sound, we get the tags from the similar sounds. A tag is proposed as a candidate if it appears among the neighbors over a specific threshold. For example, a threshold of 0.3, means that a tag is selected as candidate when it appears at least in 3 sounds of the 10 nearest neighbors. This way we select the set of candidate tags for each sound in the test dataset.

The experiments have been computed using two thresholds: 0.3 and 0.4. When using a threshold of 0.3 the number of candidate tags is higher than for 0.4, but also there are more "noisy" or potentially irrelevant tags, since it is using a less constrained approach. Afterwards, all the candidate tags will be evaluated by human assessment. The differences between both thresholds is presented in section 6.1.

5.4 Evaluation

In order to validate the candidate tags for the test sounds, we use human assessment. The aim is to evaluate the perceived quality of the candidate tags. It is worth noting that neither Precision nor Recall measures are applicable as the test sound contains only two or less tags, and these are very rare in the vocabulary. We performed a listening experiment where the subjects were asked to listen to the sounds, and decide whether they agreed or not with the candidate tags. For each candidate tag, they had to select one of these options: *Agree* (recommend candidate tag), *Disagree* (do not recommend), or *Don't know*. Each sound was rated by three different subjects.

Similar to [16], to evaluate the results we group human responses for each sound s, and score them in order to compact them into a single vector per sound. The length of the vector is the number of candidate tags of s. Each value of the vector, w_{s,t_i}, contains the weight of the subjects' scores for a candidate tag t_i in sound s. If a subject agrees with the candidate tag, the score is $+1$, -1 if disagrees, and 0 if she does not know. The formula for calculating the weight of the candidate tag in s is:

$$\mathbf{w}_{s,t_i} = \frac{\#(PositiveVotes) - \#(NegativeVotes)}{\#Subjects} \qquad (2)$$

A candidate tag is recommended to the original sound if $\mathbf{w}_{s,t_i}$ is greater than zero, otherwise, the tag is rejected (either because it is a bad recommendation, or the subjects cannot judge the quality of the tag). For example, given a candidate tag t_i for s, if the three subjects scored, respectively, $+1$, -1, $+1$ (two of them agree, and one disagree), the final weight is $\mathbf{w}_{s,t_i} = 1/3$. Since this value is greater than zero, t_i is considered a good tag to be recommended.

Furthermore, we use $\mathbf{w}_{s,t_i}$ to compute the confidence agreement among the subjects. First, we consider all the sounds where the system proposed j candidate tags, S_j. We sum, for each sound $s \in S_j$, the weights of all the candidate tags t_i whose values were greater than zero. Then, we divide this value with the total score that the candidate tags would had if all the subjects would agree. The formula for calculating the agreement of S_j sounds, A_j, is:

$$A_j = \frac{\sum_{s \in S_j} [\mathbf{w}_{s,t_i} > 0]}{\#Subjects \cdot \left[\sum_{s \in S_j} length\,(s)\right]} \qquad (3)$$

Similarly, to compute the agreement of the bad candidate tags, we use the weights of candidate tags whose values were lesser than zero ($\mathbf{w}_{s,t_i} < 0$), in the numerator of the equation 3. Finally, to get the total agreement for all the sounds in the test set, A_{total}, we use the weighted mean of all A_j, according to the number of sounds in A_j.

6 RESULTS

6.1 Perceived quality of the recommended tags

Using 10–NN and the content–based audio similarity, and setting a threshold of 0.3, the system proposed a total of 781 candidate tags, distributed among the 260 sounds of the test dataset. Besides that, setting a threshold of 0.4 the system proposes 358 candidate tags, which represents almost the half compared with a threshold of 0.3.

Table 3 shows the human assessment results. As expected, a slightly higher percentage of candidate tags were recommended with a threshold of 0.4 (66.23%). Yet, using a threshold of 0.3, more than half of the candidate tags (56.6%) were finally recommended to the original sounds, with an agreement confidence of 0.74. This human agreement is sufficiently high to rely on the perceived quality of the recommended tags. The rest of the candidate tags (43.4%) were not recommended, either because the tags recommended were not appropiated (31.59%), or the tags were not sufficiently informative (11.41%). Even though with a threshold of 0.3 we get less percentage of recommended tags, the absolute number of candidate tags is more than twice the ones with a threshold of 0.4. Therefore, we can consider a threshold of 0.3 a good choice for this task.

6.2 Recommended tags per class

On the one hand, using a threshold of 0.4 we are able to enhance the annotation of half of the sounds (128 sounds out of 260). On the other hand, with a threshold of 0.3, we have enhanced the annotation of 200 sounds, which represent the 77% of the sounds in the test dataset used. The rest of the sounds (60) from the test set did not get any plausible tags to extend its current annotation.

Table 3. Percentage of recommended tags, with confidence agreement among the subjects. The table shows the results using thresholds 0.3 and 0.4 (in parenthesis, it is shown the total number of candidate tags).

Threshold	Recommend tag	%	A_{total}
0.3 (781)	Yes	56.60%	0.74
	No	31.59%	0.62
	Don't know	11.41%	—
0.4 (358)	Yes	66.23%	0.78
	No	23.11%	0.58
	Don't know	10.66%	—

Table 4. Number of sounds in each category, after automatically extending the annotations of 200 sounds from the test dataset.

	Tags per sound	Sounds
Class I	1–2	20
Class II	3–8	171
Class III	> 8	9

Table 4 shows the results using a threshold of 0.3, and it classifies the 200 autotagged sounds according to the classes defined in Table 1. Originally, all the test sounds belonged to Class I. We can observe now the number of sounds per class, after extending the annotation of these 200 sounds. Note that most of the sounds have 3 or more tags (Class II), and some even have more than 8 tags (Class III). However, there are 20 sounds still belonging to Class I. This happens because before the experiment they only had one tag, and now they have another one, the one recommended.

The results obtained so far look promising; using a simple classifier we were able to automatically extend sound annotations that were difficult to retrieve. Furthermore, due to the classifier method used (k–NN), there is a strong correlation among the more frequently proposed tags, and their frequency of usage (rank position in Figure 2). The ten most proposed tags are also in the top–15 ranking of frequency use. Although our approach is prone to popular tags, once the sounds are autotagged it allows the users to get a higher recall of those scarcely annotated sounds when doing a keyword–based search.

7 CONCLUSIONS

This paper presents an analysis of the *Freesound.org* collaborative database, where the users share and browse sounds by means of tags, and content–based audio similarity search. First we studied how users annotate the sounds in the database, and detected some well–known problems in collaborative tagging, such as polysemy, synonymy, and the scarcity of the existing annotations.

Regarding the experiments, we selected a subset of the sounds that are rarely tagged, and proposed a content–based audio similarity to automatically extend these annotations (autotagging). Since the sounds in the test set contained only one or two rare tags, neither precision nor recall were applicable, so we used human assessment to evaluate the results. The reported results show that 77% of the test collection were enhanced using the recommended tags, with a high agreement among the subjects.

As future work, we are planning to extend the experiments using more sounds. In this case, automatic evaluation is needed. A possible solution is to select sounds belonging to similar sound categories (e.g all the percussive sounds scarcely annotated), and follow the same procedure of finding similar sounds from the *Freesound.org* database. So, the recommended tags should also belong to the same sound category. We are also working on a hybrid approach that combines tag similarity and content–based similarity to improve the recommendations of the similar sounds.

8 ACKNOWLEDGMENTS

The authors would like to thank the Freesound community for making Freesound.org a great research platform on which to study many interesting sound problems. We also want to acknowledge the help of many researchers of our research group in the technical aspects of this paper.

9 REFERENCES

[1] J.-J. Aucouturier and F. Pachet. A scale-free distribution of false positives for a large class of audio similarity measures. *Pattern Recognition*, 41(1):272–284, 2008.

[2] P. Cano. *Content-Based Audio Search from Fingerprinting to Semantic Audio Retrieval*. PhD thesis, Universitat Pompeu Fabra, 2007.

[3] C. Cattuto, D. Benz, A. Hotho, and G. Stumme. Semantic analysis of tag similarity measures in collaborative tagging systems. In *Proceedings of the 3rd Workshop on Ontology Learning and Population*, Patras, Greece, July 2008.

[4] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Reviews*, June 2007.

[5] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *Advances in Neural Information Processing Systems 20*, pages 385–392. MIT Press, Cambridge, MA, 2008.

[6] W. W. Gaver. What in the world do we hear? an ecological approach to auditory event perception. *Ecological Psychology*, 5(1):1–29, 1993.

[7] S. Golder and B. A. Huberman. The structure of collaborative tagging systems, 2005.

[8] P. Herrera, A. Yeterian, and F. Gouyon. Automatic classification of drum sounds: A comparison of feature selection methods and classification techniques. In *Proceedings of the Second International Conference on Music and Artificial Intelligence*, pages 69–80, London, UK, 2002.

[9] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In *Proceedings of the 3rd European Semantic Web Conference*, pages 411–426, Budva, Montenegro, June 2006. Springer.

[10] B. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG 7: Multimedia Content Description Language*. Ed. Wiley, 2002.

[11] G. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information, 1956.

[12] P. Schaeffer. *Trait des objects musicaux*. Editions du Seuil, Paris, 1966.

[13] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceeding of the 17th international conference on World Wide Web*, pages 327–336, New York, NY, USA, 2008. ACM.

[14] M. Sordo, C. Laurier, and O. Celma. Annotating music collections: How content-based similarity helps to propagate labels. In *Proceedings of 8th International Conference on Music Information Retrieval*, Vienna, Austria, 2007.

[15] D. Torres, D. Turnbull, L. Barrington, and G. Lanckriet. Identifying words that are musically meaningful. In *Proceedings of 8th International Conference on Music Information Retrieval*, Vienna, Austria, 2007.

[16] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards musical query-by-semantic-description using the cal500 data set. In *Proceedings of 30th International SIGIR Conference*, pages 439–446, New York, NY, USA, 2007. ACM.

[17] J. Walker. Feral hypertext: when hypertext literature escapes control. In *Proceedings of the 16th conference on Hypertext and hypermedia*, pages 46–53, New York, NY, USA, 2005. ACM.

CURRENT DIRECTIONS WITH MUSICAL PLUS ONE

Christopher Raphael
Indiana Univ., Bloomington
craphael@indiana.edu

ABSTRACT

We discuss the varieties of musical accompaniment systems and place our past efforts in this context. We present several new aspects of our ongoing work in this area. The basic system is presented in terms of the tasks of score following, modeling of musical timing, and the computational issues of the actual implementation. We describe some improvements in the probabilistic modeling of the audio data, as well as some ideas for more sophisticated modeling of musical timing. We present a set of recent pieces for live player and computer controlled pianos, written specifically for our accompaniment system. Our presentation will include a live demonstration of this work.

1 APPROACHES TO MUSICAL ACCOMPANIMENT SYSTEMS

Musical accompaniment systems are computer programs that serve as musical partners for live musicians. The types of possible interaction between live player and computer are widely varied, to some extent defying classification. Some approaches create sound by processing the live player's audio using simple analysis of the audio content itself, perhaps distorting, echoing, harmonizing, or commenting on the soloist's audio in largely predefined ways, [1], [2]. Other orientations are directed toward improvisatory music such as jazz, in which the computer follows and perhaps even composes a rendered musical part [3]. A third approach models the traditional "classical" concerto setting in which the computer's task is to perform a precomposed musical part in a way that follows a live soloist such as [4],[5]. There are a number of examples that blend these scenarios, while other approaches may be entirely outside this realm of possibilities.

Our work has focused on the latter "concerto-type" setting, as in a non-improvisatory composition for soloist and accompaniment — say a violin concerto. While the music has already been composed in this domain, the solo player may take great liberty with the performance, requiring the accompanying ensemble to be both flexible and responsive.

SMC 2009, July 23-25, Porto, Portugal

The motivation for this kind of accompaniment system is evident in the omitted for review (JSoM) at omitted for review where most of our recent experiments have been performed. For example, the JSoM contains about 200 student pianists while the regular orchestras perform two piano concerti every year using student soloists. With this in mind, it is clear that most of these aspiring pianists will never perform as orchestral soloist during their studies here. We believe this is truly unfortunate, as nearly all of these students have the necessary technical skills and musical depth to greatly benefit from the concerto experience. Our work in musical accompaniment systems strives to bring this rewarding experience to the music students, amateurs, and many others who would like to play as orchestral soloist, though, for whatever reason, don't have the opportunity.

Even within the realm of classical music, there are a number of ways to cast the accompaniment problem, requiring substantially different approaches. For instance, when accompanying early-stage musicians, the accompanist's role is not simply to follow the young soloist, but rather to encourage habits of accurate rhythm, steady tempo, while introducing musical ideas. In a sense, this is the hardest of all classical music accompaniment problems, since the accompanist must be expected to know *more* than the soloist, thus dictating when the accompanist must lead and when to follow. A coarse approximation to this accompanist role is to provide a rather rigid accompaniment that is not overly responsive to the soloist's interpretation (or errors); there are several commercial programs that take this approach. The notion of a pedagogical music system — one that follows and leads as appropriate — is largely undeveloped, possibly due to the difficulty of modeling the objectives. However, we see this area as fertile for lasting research contributions and hope that we, and others, will be able to contribute to this cause.

An entirely different scenario deals with music that evolves largely without a sense of rhythmic flow, such as in some compositions of Penderecki, Xenakis, Boulez, Cage, and Stockhausen, to name only a few. Such music is often notated in terms of seconds, rather than beats or measures, to emphasize the irrelevance of traditional pulse to the music's agenda. For works of this type involving soloist and accompaniment, the score indicates points of synchronicity, or time relations, between various anchor points in the solo and accompaniment parts. Due to the lack of predictability of such music, a natural accompaniment approach is simply to wait until various solo events are detected, and then to *respond* to these events. This is the approach taken by the IRCAM score follower, with considerable success in a variety of pieces of this type [6],[7].

The third scenario, which includes our system [5],[8], treats works for soloist and accompaniment having a continuing musical pulse, including the overwhelming majority of "common practice" art music. This music is the primary focus of most of our performance-oriented music students, and is the music where our accompaniment system is most at home. Music containing a regular, though not rigid, pulse requires close synchronization between the solo and accompanying parts, as the overall result suffers greatly as this synchrony degrades. We will argue that this music cannot be performed effectively with the purely "responsive" approach as discussed above.

Our system is known as omitted (MPO) due to its alleged improvement on the play-along accompaniment records from "Music Minus One" that inspired our work. We have been collaborating for several years with faculty and students in the JSoM on this traditional kind of concerto setting, in an ongoing effort to improve the performance of our system. What follows contains a description of some of these improvements not discussed elsewhere, as well as a number of illuminating examples and demonstrations. We will also discuss strengths and weakness of our rhythm model, while sketching possible improvements. We conclude with a presentation of our accompaniment system in new music, focusing on works by omitted for review, specifically written for our system.

2 OVERVIEW OF MUSIC PLUS ONE

2.1 Score Following

Score following is the task of computing an ongoing alignment between a symbolic music score and an audio performance of the score, as the audio data accumulates. Also known as on-line alignment, the problem is more difficult that its off-line cousin, since an on-line algorithm cannot consider future audio data in determining the times of audio events. Thus, one of the principal challenges of on-line alignment is the tradeoff between accuracy —- reporting the correct times of note events — and latency — the lag in time between the reporting time and and estimated note event time. As with all of the accompaniment systems discussed above, score following plays a crucial role in MPO. [9] gives a nice annotated bibliography of the many contributions to score following.

Our approach to score following is based on a hidden Markov model and is described in [10]. Perhaps one of the main virtues of the HMM-based score follower is the grounding it gives to navigating the accuracy-latency tradeoff. One of the worst things a score follower can do is report events before they have occurred. In addition to the sheer impossibility of producing accurate estimates in this case, the musical result often involves the accompanist arriving at a point of coincidence before the soloist does. When the accompanist "steps on" the soloist in this manner, the soloist must struggle to regain control of the performance, perhaps feeling desperate and irrelevant in the process. Since the consequences of false positives are so great, the score follower must be reasonably certain that a note event has already occurred before reporting its location. Through the probabilistic nature of the HMM, one can compute the *probability* that the currently pending note has passed. Once this has occurred, our score follower looks backward in time to find the most likely onset position for the note.

We will omit a detailed discussion of the innards of our score follower here, and content ourselves with a simple, obvious, and crucial observation: Before an audio event can be detected it must have sounded for some brief period of time. Thus any score follower must necessarily deliver its observations with latency. That is, while a note onset time may estimated correctly, the reporting of this time must come after the event has occurred.

This observation has important consequences for the musical accompaniment system: If coordination is to be achieved in a "responsive" way — by waiting until a solo event is detected and then playing the corresponding accompaniment note, the system will always be late. In theory, one may be able to construct a score follower whose latency is musically insignificant. However, this has not been possible in our experience with such latencies usually in the 60-90 ms. range. If all coincident accompaniment notes lag this far behind, the result is musically fatal.

Instead, we accept as a basic tenet that detection latency will be musically significant and base our coordination of parts on *prediction* rather than response. Thus, central to our approach is the recognition that score following alone is not enough to produce good musical accompaniment. In addition we need a means of predicting future musical events and scheduling them accordingly. In contrast, the IRCAM system's approach is responsive, playing events in direct response to observations of solo events. This system has been quite successful in music that does not have a sense of ongoing pulse — the IRCAM system was developed with this kind of music in mind. However, the extension of this work to other musical styles including the overwhelming majority of common practice art music and popular music, seems problematic. In contrast, with minor adaptations our approach is equally at home in pulseless music.

A video demonstrating our score following ability can be seen at `http://www.music.informatics.indiana.edu/papers/smc09`. In this video the rather eccentric performer ornaments wildly, makes extreme tempo changes, plays wrong notes, and even repeats a measure, thus demonstrating the robustness of the system.

2.2 Modeling Musical Timing

As discussed above, our approach to accompaniment relies on the prediction of *future* musical events. We present here the model serving as the backbone for this process. We begin with three important traits we believe such a model must have.

1. Since our accompaniment must be constructed in real time, the computational demand of our model must be feasible in real time.

2. We anticipate training our prediction algorithm using a sequence of rehearsals in which the solo player demonstrates her interpretation, with all its variability. In order to benefit from these rehearsals our model must be automatically trainable. Thus, rehearsal will allow our system to more accurately anticipate the way future musical timing will unfold. This is certainly one of the objectives of human rehearsal, as well.

3. If our rehearsals are to be successful in guiding the system toward the desired musical end, the system must "sightread" (perform without rehearsal) reasonably well. Otherwise, the player will become distracted by the poor ensemble and not be able to play her part consistently with her convictions. Thus our model must be constructed around widely applicable musical assumptions, so it can perform reasonably well "out of the box."

Our model is expressed in terms of two sequences, $\{t_n\}$ and $\{s_n\}$ where t_n is the time, in seconds, at which the nth note begins and s_n is the tempo, in seconds per beat, for the nth note. The model is then

$$\begin{aligned} s_{n+1} &= s_n + \qquad \sigma_n & (1) \\ t_{t+1} &= t_n + l_n s_n + \tau_n & (2) \end{aligned}$$

where l_n is the length of the nth note, in beats. With the $\{\sigma_n\}$ and $\{\tau_n\}$ variables set to 0, this model gives a literal and robotic musical performance. The introduction of these variables allow time-varying tempo through the σ's and elongation or compression of note lengths with the τ's. To complete the model we assume that

$$\begin{pmatrix} \sigma_n \\ \tau_n \end{pmatrix} \sim N(\mu_n, \Sigma_n)$$

where $N(\mu, \Sigma)$ denotes a joint normal distribution with mean μ and covariance Σ. Thus the $\{\mu_n\}$ vectors represent the *tendencies* of the performance — where the player tends to speed up ($\sigma_n < 0$), slow down ($\sigma_n > 0$), and stretch ($\tau_n > 0$), while the $\{\Sigma_n\}$ matrices capture the repeatability of these tendencies.

If the actual note observations generated by our score follower, $\{y_n\}$ are viewed as imperfect estimates of the *true* onset times,

$$y_n = t_n + \epsilon_n \qquad (3)$$

where $\epsilon_n \sim N(0, \rho^2)$, and all of the $\{\sigma_n, \tau_n, \epsilon_n\}$ variables are modeled as *independent*, then the model is seen as a straightforward example of the Kalman filter. In this context, all of our desired traits are satisfied. We predict future evolution by first computing our knowledge of the current state given our observations, $p(s_n, t_n | y_1, \ldots, y_n)$. From this information we can predict future note onset times by applying our basic model to our current belief, thus allowing the system to sightread. Furthermore, using standard ideas from the Bayesian network literature, we can perform maximum likelihood estimation on the $\{\mu_n, \Sigma_n\}$ parameters, thus training our model from actual rehearsal data. Finally, the computational burden of these calculations is modest, at most, easily suiting the approach for real time.

Our system is concerned only with the scheduling of the currently pending accompaniment note. Every time new information becomes available, either in the form of a played accompaniment note or a detected solo note, we have new information about the pending note. Thus we reestimate the current state, predict the accompaniment location, and reschedule the note accordingly. If we consider the common situation involving a run of solo notes culminating in a point of coincidence between solo and accompaniment parts, we see that this time of coincidence will be rescheduled many times before its scheduled time finally occurs and the note is played. In this way, our system makes use of all information currently available, continually modifying its view of musical timing until it must finally act.

We have created a video to demonstrate this process, available at the aforementioned website. The video shows the estimated solo times from our score follower appearing as green marks on a spectrogram. Predictions of our accompaniment system are shown as analogous red marks. One can see the pending accompaniment event "jiggling" as new solo notes are estimated, until finally the time currently predicted time passes.

At this point there seems to be so much "good news" that one is loathe to make criticisms. However, long experience with this model in action has demonstrated a number of deficiencies, mostly perceived as a kind of musical naivete. We will discuss these and pose possible improvements in a later section.

2.3 Computational Approach

Our program consists of about 100,000 lines of C code with the graphical interface written in C++. The score follower is implemented as a *thread* which continually polls to see if a new frame audio data is ready, with about 31 audio frames per second. When a new frame is available, the thread runs an iteration of the HMM forward algorithm. If the forward algorithm detects that the pending solo note has passed, the most likely onset frame is computed through the forward-backward algorithm, using all currently-available audio data. This most likely time is then modeled as a noisy estimate of the *true* solo time, (Eqn. 3) and the pending accompaniment note is rescheduled using the Kalman filter model.

Our system can create the audio output using either MIDI, or resynthesizing the output audio from an accompaniment-only recording. This latter method is our preferred approach for traditional common practice art music, since it preserves much of the tonal quality and some of the performance intent of the original recording. We often use the Music Minus One recordings for this purpose. When using a recording, we resynthesize the audio using phase vocoding, thus allowing time warping in the original recording without any change of pitch.

A separate high-priority thread handles this audio output — while there is no great danger in delaying the processing of audio input, a delay in audio output can result in a "drop-out" with an associated click or gap in the audio output. This thread is time critical since we create the audio at the last possible moment allowing it to be influenced by the most current information from the audio analysis thread. Typically we buffer about .064 seconds of outgoing audio. This thread constructs each frame of audio according to the current vocoding "play rate," computed from the prediction model as the rate needed to arrive and the pending event at the predicted time.

While originally written for the Linux operating system, our preferred home, in recent years we have ported the system to Windows. Ideology aside, the target community of this work is actual practicing "classical" musicians, more familiar with Windows. No special-purpose hardware is needed to run the system.

3 MODELING THE ORCHESTRA'S CONTRIBUTION TO THE AUDIO

One of the often-touted virtues of the HMM is its trainability. That is, an HMM can use representative data to automatically improve its transition and output models, perhaps resulting in better performance. Though we continue to place faith in this trainable aspect of the HMM we have replaced a fully trained output model with a different model that performs significantly better, even without training.

This model computes the likelihood of an audio magnitude spectrum $q = (q_1, \ldots, q_K)$ given an assumption about the note or notes sounding in the solo part. In doing so, we construct a probability template $p = p_1, \ldots, p_K$ for the note or notes that may be sounding at a particular time. For a single note we have modeled p as a mixture of Gaussians centered at the harmonic frequencies of the note with decreasing mixture weights as harmonic number increases:

$$p_k = \sum_{h=1}^{H} w_h N(k; hf_0, (hf_0)^2 \rho^2) \tag{4}$$

where $\sum_h w_h = 1$, f_0 is the fundamental frequency of the note, and $N(k; \mu, \sigma^2)$ is a discrete approximation to the normal density function. With this probability model in place, we view the actual audio magnitude spectrum as a random sample from the probability model. That is, we regard q_k as the number of observations at frequency k — q_k must be discretized for this to make sense. Then we have

$$p(q|p) = c(q) \prod_k p_k^{q_k}$$

where $c(q)$ is the multinomial constant. In the event that we are following a polyphonic instrument, we simply model p in Eqn. 4 with an additional sum over the collection of currently-sounding solo notes. This model has worked well in practice in a wide variety of situations and can be extended in some interesting ways, as follows.

The model above may describe reasonably well the audio signal that comes from the soloist, for purposes of note discrimination. However, our microphone will receive both this solo audio as well as the audio generated by our accompaniment system. When the accompaniment audio contains components that are confused with the solo audio, this can lead to the highly undesirable possibility of the accompaniment system *following itself* — in essence, chasing its own shadow. To a certain degree, the likelihood of this outcome can be diminished by "turning off" the score follower when the soloist should not be playing. We do this. However, there is still significant potential for shadow-chasing since the pitch content of the solo and accompaniment parts is often similar.

Our solution to this difficulty is to directly model the contribution of the accompaniment to the incoming audio signal we process. Since we *know* what the orchestra is playing, we add a component of this contribution to our probability model. More explicitly, if p_s is the solo template described above, and p_o is the known contribution of the orchestra to the currently analyzed audio frame, we create a probability model for the observed magnitude spectrum q by $p = \lambda p_s + (1 - \lambda) p_o$. This is the actual p we use in evaluating the data likelihood.

This addition creates significantly better results in many situations. The surprising difficulty in actually implementing the approach, however, is that there seems to be only weak agreement between the audio that our system plays and the accompaniment audio the comes in from the microphone. We can improve our model of p_o by various averaging tricks, thus modeling the room acoustics to some degree.

Doing so leads to a p_o estimate that seems to largely eliminate the undesirable shadow-chasing.

4 BETTER MODELING OF MUSICAL TIMING

We have already discussed the strengths of the musical timing model of Eqns. 1-2, however, it would be disingenuous to claim there are no weaknesses. Clearly our model must allow for a range of possible musical performances, since we know we will encounter variation in practice. Since we do not know the nature of this variation, we have over-parametrized the model, allowing for way too much flexibility — and perhaps not the right kind. Surely the player will not make a change to the tempo *and* apply tempo-independent note length variation on every note. However, our model allows such a performance (and accommodates it reasonably well). We propose a couple of possible variations on the basic rhythm model here.

Our first observation concerns the $\{\tau_n\}$ variables of the model, which represent changes in note length not naturally expressed through tempo. The prime musical example would be the *agogic* accent, in which one stresses a note by lengthening it, though keeping the same basic tempo in subsequent notes. This is a common expressive device in playing passages of fast running fast notes, to highlight important metric positions, harmonic changes, dissonances, etc. While this example of note lengthening is familiar in a variety of musical styles, we don't believe the same holds for *shortening* of note length. Of course, there are musical examples where the conceptual rhythm may differ from that explicit notation, such as the double-dotting of a French overture, or the swing of jazz. But these are examples where "stolen" time is given back elsewhere, unlike the case of $\tau_n < 0$. We expect that the musical plausibility of our model is improved by removing this possibility.

Our second observation is that tempo changes and note length variation introduced by the player is sparse — most notes are rendered without any such deviation, while it may not be musically meaningful to have both agogic accent and tempo change in the same position. Phrased in terms of our model, most of the $\{\sigma_n, \tau_n\}$ variables are 0 and we should not allow $\sigma_n \neq 0$ *and* $\tau_n \neq 0$ for fixed n.

We propose the following model to capture these notions. We let $x_1, x_2, \ldots$ be a hidden discrete process where n continues to index the notes of the piece. We assume $x_n \in \{1, 2, 3, 4\}$, with the following interpretations:

$$\begin{aligned} x_n = 1 &\iff \sigma_n = \tau_n = 0 \\ x_n = 2 &\iff \tau_n = 0 \\ x_n = 3 &\iff \sigma_n = 0, \tau_n \sim N(\mu_3, \rho_3^2) \\ x_n = 4 &\iff \sigma_n = 0, \tau_n \sim N(\mu_4, \rho_4^2) \end{aligned}$$

That is,

1. When $x_n = 1$ we arrive at note n exactly in tempo.
2. When $x_n = 2$ the tempo may change between notes $n - 1$ and n, but there is no additional note length variation.
3. When $x_n = 3$ we have have an agogic accent and no tempo variation. This is the case of a small agogic accent where the parameters $\mu_3 > 0$ and ρ_3^2 are chosen to ensure that a negative value is highly unlikely.
4. When $x_n = 4$ we have have a similar situation, but now account for the longer agogic accent. Thus $\mu_4 > \mu_3$ with ρ_4^2 also chosen to make negative values of τ_n rare.

Of course these 4 cases are not equally likely, thus we model the probabilities of $p(x_n = i)$ to reflect that $i = 1$ is, *a priori*, the most likely, with reasonable choices for the other 3 cases. It may even be reasonable to model the $x_1, x_2, \ldots,$ process as a Markov chain allowing for some small degree of memory in the choice of expressive actions.

The model is now a Switching Kalman filter [11]. For the Switching Kalman filter, the exact computation of the filtered distribution: $p(x_n, s_n, t_n | y_1, \ldots, y_n)$ is not tractable due to the large number of paths $x_1, \ldots, x_n$ that must be marginalized over, in accounting for all of the ways we can arrive at state (x_n, s_n, t_n). However, there are numerous ways to approximate this calculation, using various approximation schemes. In addition, such models are also amenable to automatic training using ideas analogous to those employed with Kalman Filters and HMMs. Here we train the $p(\sigma_n, \tau_n)$ parameters, as before, and additionally train the $p(x_n)$ probabilities. Thus we learn the *qualitative* behavior of the soloist through the $p(x_n)$ probabilities, which tell us where various kinds of actions are likely to occur, as well as the quantitative description learned through the $p(\sigma_n, \tau_n)$ parameters. Experiments are currently underway with such a model.

5 NEW MUSIC WITH ACCOMPANIMENT SYSTEM

Our work with accompaniment systems has mostly focused on common practice music for soloist and orchestra, however, we believe the accompaniment system is by no means limited to this domain. There has been a long tradition of compositions for live soloist and accompanying electronica, with many possible techniques for coordinating parts. In some of these, the live player is completely responsible for synchronization, by either following a tape or playing along with a click track. In others, a human plays the role of the "conductor," cueing electronic or computer parts at the appropriate times. There have also been some examples in which the computer genuinely *follows* the live player, but with some of the best results in music not relying on regular pulse, such as with IRCAM's score follower mentioned above. We believe that the notion of pulse is in no way limited to common practice music, as exemplified by the vast collection of contemporary music that employs metered rhythm. Thus we believe our accompaniment system may create possibilities for new music, perhaps not playable by any other means, whose composition is of genuine interest to living composers.

Recently we have recorded two such new works for oboe and computer-controlled pianos written specifically for our accompaniment system by Swiss composer name omitted for review: *Mist Covered Mountains* and *Winter*. While the pieces use traditional rhythmic notation and sometimes have a highly rhythmic feel, they require a level of pianistic virtuosity and ease with complex polyrhythms posing nearly superhuman demands on the pianist. This is fitting, since the piano part(s) were not intended to be played by humans.

One of the main challenges for the oboist is in understanding the rhythmic relationship between the parts; the score notates all rhythm precisely, though there is an aleatoric feel to large sections. While a good deal of score study was necessary to accomplish this, quite a bit of rote memorization was also necessary, accomplished through regular listening over a period of several months. Perhaps the author's original understanding of this music was something like the young student's knowledge of the "Pledge of allegiance" — knowing the sequence of syllables, but perhaps not the meaning of the words. However, the music began to make sense after passing through this stage. The accompaniment system was a significant aid in *learning* these pieces, since it came to our rehearsals already understanding the complex rhythmic relations and reinforced these through repetition and automatic adaptation to the soloist's errors.

The music was recorded in a studio, recording the live oboe while listening to a MIDI performance of the pianos through headphones, as controlled by the accompaniment system. The MIDI piano performance was as captured and later used to control a Bösendorfer reproducing piano. The resulting piano audio was then mixed with the original oboe. Recordings of sections of these pieces are available at the web page mentioned earlier. Though the merit of these pieces does not lie in their reliance on new technology, it seems nearly impossible to perform these pieces with anything other than an accompaniment system.

6 REFERENCES

[1] Lippe C., "Real-time Interaction Among Composers, Performers, and Computer Systems", *Information Processing Society of Japan, SIG Notes*, Vol. 2002, Num. 123, pp 1-6, 2002.

[2] Rowe R., *Interactive Music Systems*, MIT Press, Cambridge, MA, 1993.

[3] Dannenberg R.and Mont-Reynaud B., " Following an Improvisation in Real Time" *Proc. of the 1987 International Computer Music Conference*, pp. 241-248, 1987.

[4] R. Dannenberg, H. Mukaino "New Techniques for Enhanced Quality of Computer Accompaniment" *Proc. of the International Computer Music Conference* 243–249, Köln, 1988.

[5] omitted for review, "A Bayesian Network for Real-Time Musical Accompaniment" *Advances in Neural Information Processing Systems, NIPS 14*, MIT Press, 2002.

[6] Cont A., Schwarz D., Schnell N., "Training IRCAM's score follower", *Proceedings of IEEE Int. Conf. on Acoustics Speech and Signal Processing (ICASSP)*, Philadelphia, USA 2005.

[7] Cont A., Schwarz D., Schnell N., "From Boulez to Ballads: training IRCAM's score follower", *Proceedings of the Int. Computer Music Conf. (ICMC)*, Barcelona, Spain, 2005.

[8] omitted for review, "Music Plus One: a System for Expressive and Flexible Musical Accompaniment" *Proc. Int. Comp. Music Conf., 2001* Havana, Cuba, 2001.

[9] Schwarz D., "Score following commented bibliography", *http://www.ircam.fr/equipes/temps-reel/suivi/bibliography.html*, 2003.

[10] omitted for review, " Segmentation of Acoustic Musical Signals Using Hidden Markov Models", *IEEE Trans. on Pattern Analysis and Machine Intelligence* 21(4), 1999.

[11] Bar-Shalom Y., *Estimation and Tracking: Principles, Techniques, and Software*, Artech House, Boston, Massachusetts, 1993.

TOWARDS AUDIO TO SCORE ALIGNMENT IN THE SYMBOLIC DOMAIN

Bernhard Niedermayer
Department for Computational Perception
Johannes Kepler University Linz, Austria
bernhard.niedermayer@jku.at

ABSTRACT

This paper presents a matrix factorization based feature for audio to score alignment. We show that in combination with dynamic time warping it can compete with chroma vectors, which are the probably most frequently used approach within the last years. A great benefit of the factorization-based feature is its sparseness, which can be used in order to transform it into a symbolic representation. We will show that music to score alignments using the symbolic version of the feature is less accurate but on the other hand reduces the memory required for feature representation and during the alignment process to a fraction of the original amount. This is of special value when dealing with very long pieces of music where the limits of default DTW are reached.

1 INTRODUCTION

The problem of audio to midi alignment is well known and has been of broad interest within the last years. The task is to link information from a score representation to an audio recording of a certain performance of a piece of music. Since symbolic transcriptions of a large number of classical as well as modern pieces are available, alignment can replace the much more complex task of blind audio transcription in most scenarios where the performed piece is known in advance.

This is the case in a number of applications like in the field of computational musicology. Performance analysis for example relies on the exact transcription of various performances in order to describe or compare the styles of different artists. Other applications of audio alignment are pedagogical systems and special query engines as well as intelligent audio editors or players.

State-of-the-art approaches like [2], [4], or [12], just to name a few, use a combination of a local distance measure and a specific kind of dynamic time warping (DTW) or Hidden Markov Models (HMM) in order to find the optimal global alignment between the score and a corresponding audio file. Although the idea of calculating such alignments in the symbolic domain is not new [1], modern approaches avoid transcription of the audio data into a symbolic representation. Instead local distances are calculated from acoustic features extracted from the audio signal on the one hand, and either from a rendering or directly from the score on the other hand.

Amongst these acoustic features chroma vectors seem to be most frequently used. Each vector has 12 elements corresponding to the pitch classes (i.e. C, C#, D,. . .). The computation is based on a short time Fourier transform (STFT) where each frequency bin is mapped to a musical note. The notes are then folded into a single octave by calculating the average energy of all STFT bin contributions to the same pitch class. A more detailed description can be found in [7]. In [4] this representation was compared to others like Pitch Histograms or MFCC based features in the context of audio matching and alignment. It was shown that chroma vectors perform significantly better than the other features.

Another approach is to use features composed of estimations of the presence of individual pitches instead of pitch classes. The idea of such a feature based on non-negative matrix factorization was initially proposed in [2]. In this work we describe a feature that also represents f_0-observation probabilities for single pitches but amongst other things differs in the optimization criterion and algorithms used in the matrix factorization step. Whereas in [2] a qualitative evaluation is given, we will present a quantitative evaluation on a large data set, comparing our feature to chroma vectors and show that the two features yield comparable results when used in combination with dynamic time warping.

Although this feature is acoustic in nature it can be easily converted into a symbolic form in order to use the advantages of both representations. One could be a reduction of computational costs since local similarities can be computed on note events instead of the much larger number of audio frames.

In Section 2 we will give an overview of the algorithm used to extract the proposed feature. Section 3 then describes the global alignment using this feature in the acoustic domain. An evaluation and comparison with the performance of chroma vectors is given in Section 4 before we show and discuss an analogous alignment method in the symbolic domain in Section 5.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

2 METHODOLOGY

2.1 Pitch decomposition

A feature based on non-negative matrix factorization (NMF) for audio alignment was originally proposed in [2]. The basic idea is that a non-negative input V of the size $m \times n$ is decomposed into two as well non-negative output matrices W and H of size $m \times r$ and $r \times n$ respectively, such that

$$V \approx W \cdot H \tag{1}$$

The quality of a factorization is measured by a cost function over V and $W \cdot H$. Common choices for these functions are the Euclidean distance or the Kullback-Leibler divergence. By minimizing the cost function W and H are learned as a fixed number r of basis vectors and the aggregation of their activation patterns over time.

Applying this principle to audio processing, one can use a spectrogram, as obtained by the short time Fourier transform, in order to learn a base set W of weighted frequency groups in an unsupervised manner. In the ideal case these would either represent single pitches played on a certain instrument or pitches that are often found together like the notes of a chord.

However in the context of audio alignment, where the piece and its score are known in advance, we assume the instruments used to be known as well. So there is no absolute need for unsupervised learning of base vectors. Instead a dictionary W of tone models, adjusted to those instruments, can be trained in advance. This leaves us with only H being unknown.

As described by [11] and [8] this reduces the NMF problem to the much simpler decomposition task where each column vector of V can be processed independently, such that equation 1 resolves to

$$v \approx \overline{W} \cdot h \tag{2}$$

where $\overline{W}$ is the fixed set of tone models. v and h represent the spectrogram of a single time frame and the pitch activation respectively. This pitch activation h is the feature vector describing one time frame. In order to find an optimal h, again a cost function is needed. Throughout this work the mean square criterion given as

$$f = \frac{1}{2} \parallel \overline{W}h - v \parallel^2 \tag{3}$$

is chosen and optimized using a standard algorithm for solving non-negative least square problems as described in [6]. Reassembling the activation patterns of all time frames results in a multiple-f_0 estimation over the whole piece of music.

On the other hand extracting feature vectors describing the score is trivial since pitch information can be directly taken from the midi representation.

2.2 Dictionary learning

In order to process the pitch decomposition as described above a dictionary of tone models is required. Each of these models represent one pitch by its weighted frequency components. As pointed out in [8] the exactly same method as used for pitch decomposition in the performing step can be used for model learning in the training step.

Given a database of transcribed audio training samples, the activation patterns of single pitches are known due to the transcription. Therefore H in equation 1 can be fixed to these activation patterns while now W is calculated. Using monophonic training samples where only one pitch is present can simplify the learning step even more. On the one hand W is reduced to a vector and h becomes one scalar at each time frame. On the other hand such training samples can be created with minimal effort. Since the activation energy can be described by the amplitude envelope, the only information required for each sample is the pitch as well as the instrument that has been playing.

3 ALIGNMENT

3.1 Local Distances

Given two sequences of feature vectors a global alignment has to be found that matches each element of one sequence to a corresponding element within the other sequence. In order to measure the similarity between two such elements a local distance function is required. A common choice is the Euclidean distance or the Kullback-Leibler divergence. However two properties of the factorization based feature suggest the use of another distance measure.

In the first place the feature produces a different quantity of deletion (false negative) and insertion (false positive) errors. Especially in high pitch ranges the majority of errors is made up by spurious note detections. Therefore the two types of errors should be treated differently.

Secondly the STFT we use here divides the spectrum into linearly distributed frequency bins. On the contrary musical notes follow a logarithmic frequency scale. So the deeper a tone is, the closer in the spectrogram it is to its immediate neighbors. Additionally higher pitches also exhibit significant energy in the lower frequency bins making it even harder to reliably detect low notes. Therefore local distance calculation should accommodate this fact by relatively tolerant penalizing of missing low notes in the audio feature.

A simple distance measure that combines these ideas and has yielded good results during experimentation is

$$d(h^s, h^p) = \sum_{i=0}^{N-1} diff(h_i^s, h_i^p) \tag{4}$$

with

$$diff(h_i^s, h_i^p) = \begin{cases} h_i^s * \alpha & \text{if } h_i^p = 0 \\ h_i^p * \beta & \text{if } h_i^s = 0 \\ |h_i^s - h_i^p| & \text{else} \end{cases} \quad (5)$$

where h^s represents a feature vector taken from the score and h^p represent one feature vector extracted from the recorded performance. α and β are the weights for missing and spurious notes respectively. Throughout our work 1.2 and 2.0 have proven to yield good results.

Experiments have further shown that alignments can be improved by ignoring missing notes lower than a threshhold around C3 (midi pitch 48). Also taking the square root of d turned out to be advantageous in combination with dynamic time warping as explained in Section 3.2.

3.2 Global Optimization

Using the local distance measure a similarity matrix SM comparing each frame of one sequence to each frame of the other sequence can be built. Mapping corresponding frames together is done by finding a minimal cost path through this similarity matrix. A path through SM_{ij} is then equivalent to the alignment of frame i of the score feature sequence to the performance feature sequence's frame j. Dynamic time warping (DTW) is a well-established dynamic programming based algorithm that finds such optimal paths. A detailed tutorial can be found in [9].

In order to get meaningful results a path has to meet several constraints. The constraint of continuity forces a path to proceed through adjacent cells within the similarity matrix. Jumps would be equal to skipping frames without considering the costs of this operation. The constraint of monotonicity in both dimensions guarantees that the alignment has the same temporal order of events as the reference sequence. And finally the end-point constraint forces the ends of the path to be the diagonal corners of the similarity matrix. In doing so it assures that the alignment covers the whole sequences.

The determination of the optimal path according to DTW works in two steps. The forward step starts at the point $[0, 0]$ with the cost SM_{ij} and recursively calculates the minimized path cost of any partial alignment ending with frame i of the score being aligned to frame j of the recorded performance according to

$$Accu(i, j) = min \begin{cases} Accu(i-1, j-1) + SM_{ij} * w_d \\ Accu(i-1, j) + SM_{ij} * w_s \\ Accu(i, j-1) + SM_{ij} * w_s \end{cases} \quad (6)$$

The three options correspond to a diagonal step, a step upwards, and a step to the right within the similarity matrix respectively. Accordingly w_d and w_s are weights for diagonal and straight steps. We have chosen the values 1.4 and 1.0 giving diagonal steps a preference over straight ones. In our implementation we do this calculation in place, i.e. overwriting the values SM_{ij} by $Accu(i, j)$ in order to save memory space.

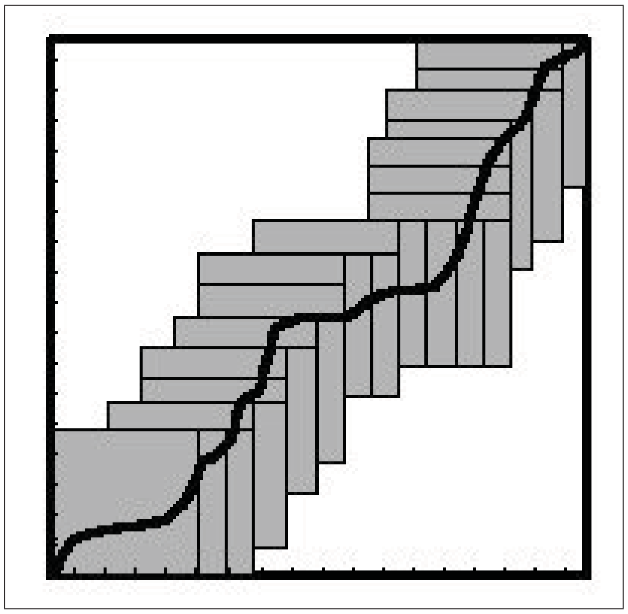


Figure 1. Refinement of the global alignment: The estimated path leads the search area through a similarity matrix computed using a higher time resolution.

Additional to the accumulated path cost a second matrix is used in order to memorize whether the last step leading to a point $[i, j]$ was diagonal, upwards, or to the right. As soon as all values $Accu(i, j)$ have been calculated this information is used to trace the complete path back from $[N-1, M-1]$ to $[0, 0]$.

3.3 Alignment of very long sequences

This algorithm is of complexity $O(n^2)$ in time as well as in space. For very long pieces of music it is impossible to keep a reasonable time resolution of features and still compute a global alignment by DTW. Several improvements have been proposed in order to reduce the complexity, including path pruning where only promising partial paths with costs below an adaptive threshold are further expanded or Shortcut Path where only the alignments of frames corresponding to note on- and offsets are stored [12]. Another approach to handle very long pieces of music is to use online algorithms like proposed in [3] instead of processing the whole piece at once.

Another approach reducing time and space complexity to $O(n)$ is multiscale DTW ([10]). Here an initial estimation of the optimal path is calculated using a low time resolution and then refined iteratively. Since each iteration increases time resolution but only considers paths near the one found during the last step there is no guarantee that the optimal path is really found. It may happen that low resolution features are misleading in such a strong way that the actual optimum is out of the search radius.

In our implementation we only use two iteration steps. In the first one a standard DTW is computed on features extracted from a spectrogram. The window as well as the hop size was chosen to be 4096 samples ($\sim$93 ms). This allows processing pieces of lengths up to more than 25 minutes. The second step is the refinement, calculated on features based on another spectrogram where the hop size was reduced to 512 samples ($\sim$12 ms). As illustrated by Figure 1 the path estimation from the first step leads the search in the second step so that only similarity values and path costs within an area of radius r frames needs to be calculated. In this way memory requirements can be kept low by just storing similarity measures of the currently processed row or column and path costs for the last $2r$ rows and columns, leading to constant space complexity of this part of the algorithm.

4 EXPERIMENTAL RESULTS

In order to evaluate the factorization based feature we test its performance on several pieces of classical piano music. The database used consists of 13 Mozart sonatas played by a professional pianist on a computer monitored Bösendorfer SE290 grand piano, giving us a precise ground truth of played notes in midi representation. The data set consists of more than 100.000 notes and represents a performance time of almost 4 hours.

The single pieces have lengths from 12 minutes up to more than 26 minutes [1] . This is longer than test pieces used in most other publications. On the one hand this leaves us with issues of computational expenses, which are handled as described in Section 3.3. On the other hand stronger deviations from a strictly diagonal alignment path are expected since the different movements of a sonata are played using different tempi, which prohibits the use of additional alignment constraints like the Itakura Parallelogram [5].

The tone models used during factorization were learned from recordings of single tones played on the same piano. Since such a recording was only available for every fourth midi pitch, the missing models were generated by means of parabolic interpolation.

4.1 Feature Evaluation

In the evaluation process an alignment was calculated for each of the test pieces using the audio recording and a midi file containing the mechanical score without any expressive timing. The resulting note onset times were compared to the ground truth data. Sections where more than 10 consecutive notes had been misplaced by more than 3 seconds were classified as 'unaligned'. Throughout the test set 31 such regions were found containing 2438 notes, which accounts for 2.4% of the overall number of notes. Further investigation showed that such regions where alignment failed are likely to be sections played very softly and with increased use of pedal. This causes the spectrogram that is used as basis for the feature calculation to blur and makes it very hard to distinguish partials belonging to a certain pitch.

For the remaining aligned notes we compute the absolute displacement relative to the ground truth data. In Table 1 we give the median difference, the third quartile and the limit covering 95% of all displacements.

The largest value found within the test set was an error of 8.631 seconds. Although we counted unaligned sections separately, sporadic values of that magnitude are plausible. Since we are dealing with whole sonatas it can happen that the alignment places single notes played at the end of one movement at the beginning of the next one and inversely. Pauses of one or more seconds between movements as well as fermatas and long sustaining of notes at the ends of movements lead to such dramatic values.

In the evaluation of onset detection algorithms, an error threshhold of 50 ms up to which a note onset is classified as correctly detected is quite common in literature. Therefore we also give the percentage of notes satisfying this criterion which is about 50% on average in Table 1.

4.2 Feature Comparison

In the context of audio matching and alignment, a comparison of several acoustic features is given by Hu et al. ([4]). They show that chroma vectors perform significantly better than the other features including variations of an MFCC based approach and Pitch Histograms. This is relevant to our work since Pitch Histograms as defined in [13] also rely on a multiple f_0-estimation. However they are computed by an algorithm based on autocorrelation.

Using the same testing environment as described above we also compare the factorization based feature against the performance of chroma vectors. We found that chroma vectors are more robust, leaving only 687 notes unaligned which accounts for 0.7% of the overall number of notes. However the evaluation of the precision of all aligned notes as given in Table 1 is comparable to the results yielded by the factorization feature. Also the value of the largest absolute error being more than 9 seconds is even worse.

The evaluation criterion used in this work was different from the one in [4]. So the results are not directly comparable. In any event, we can not confirm that chroma vectors perform significantly better than features computed by multiple f_0-estimation except for robustness, where the amount

[1] Note that we align complete sonatas. That is, the pieces were not cut into individual movements.

piece	# notes	duration	chroma vectors				factorization based			
			$50\% \leq$	$75\% \leq$	$95\% \leq$	$\% \leq 50$ms	$50\% \leq$	$75\% \leq$	$95\% \leq$	$\% \leq 50$ms
kv279	7387	16:21	35 ms	65 ms	327 ms	64.7%	32 ms	61 ms	408 ms	68.6%
kv280	6070	15:04	41 ms	75 ms	432 ms	59.7%	38 ms	69 ms	399 ms	61.6%
kv281	6395	14:37	40 ms	63 ms	193 ms	61.7%	40 ms	67 ms	169 ms	60.1%
kv282	5564	14:59	60 ms	145 ms	532 ms	41.9%	70 ms	222 ms	808 ms	38.2%
kv283	7884	17:35	38 ms	76 ms	316 ms	59.8%	40 ms	80 ms	500 ms	57.8%
kv284	12762	26:07	37 ms	65 ms	262 ms	64.1%	39 ms	65 ms	260 ms	63.2%
kv330	7589	18:36	38 ms	70 ms	262 ms	60.2%	35 ms	66 ms	358 ms	64.8%
kv331	11580	22:51	276 ms	415 ms	508 ms	9.6%	277 ms	407 ms	493 ms	10.4%
kv332	8744	18:02	51 ms	92 ms	338 ms	49.4%	52 ms	89 ms	302 ms	48.9%
kv333	8833	20:34	58 ms	89 ms	244 ms	44.1%	59 ms	93 ms	261 ms	44.0%
kv457	6915	18:22	44 ms	97 ms	525 ms	54.7%	48 ms	96 ms	919 ms	52.0%
kv475	3871	12:05	79 ms	198 ms	718 ms	32.5%	76 ms	148 ms	579 ms	32.9%
kv533	8611	22:27	46 ms	86 ms	208 ms	52.9%	47 ms	86 ms	178 ms	52.4%
all	102205	3:57:40	50 ms	106 ms	449 ms	50.0%	50 ms	104 ms	459 ms	50.2%

Table 1. Comparison between chroma vectors and the factorization based feature in combination with DTW

of aligned notes was 99.3% instead of 97.6%.

5 THE SYMBOLIC DOMAIN

Most current methods for audio to score alignment including the approaches described above work in the acoustic domain, avoiding the step of explicitly transcribing the audio data. Such a transcription would bring some benefits.

- Whereas acoustic features will result in large arrays of data, symbolic representations are much more compact, using just a small fraction of the original memory space.
- While computing alignments using DTW-like algorithms the number of frames per sequence can be dramatically reduced from a fixed ratio of frames per time unit to one frame each time a note onset or offset occurs.
- Using a transcription in midi format obvious errors of the feature extraction process can be recognized and handled prior to the actual alignment step. Examples for such obvious errors are detected notes with pitches never played during the current piece or detected chords that are never used. This might also eliminate incorrect notes played by the performer in certain cases.

We have also done experiments using the same factorization method as described above to extract an audio feature in midi-format by just setting a note on-event each time the activation energy h_i^p of pitch i becomes greater than zero and setting a note off-event each time h_i^p falls back to zero. This is not just exploiting the sparseness of the factorization result but also strong pruning since note velocities are set to a default value and the actual values of the activation patterns during the note sustain time are dropped.

	$50\% \leq$	$75\% \leq$	$95\% \leq$	$\% \leq 50$ms
acoustic	32 ms	61 ms	408 ms	68.6%
symbolic	205 ms	370 ms	905 ms	18.9%

Table 2. Comparison of alignments using the factorization based feature in its original version and pruned to a symbolic representation

Applied to the recording of Mozart's piano sonata kv279 the resulting midi representation contains 6275 notes using less than 150 kB of memory. This is a little more than 7.5% of the space needed to store the chroma vectors calculated at a time resolution of 50 frames per second. For the original acoustic representation of the factorization result this relation is even more drastic. The activation patterns of 58 pitches (concerning to the pitch range used in the kv279) require 11MB of memory which is more than 70 times the space needed for the symbolic version of the feature.

The actual alignment is again done using dynamic time warping. In doing so from the score as well as the audio feature slices containing unchanged numbers of notes are extracted (i.e. splitting the piece at each note on- and offset). The resulting number of feature vectors are comparable to those obtained in the first estimation step of our multiscale DTW implementation as described in section 3.3. For the piece kv279 there were no unaligned regions for both feature representations. But as can be seen from Table 2, the accuracy yielded by the symbolic feature can not compete with the original version. However, a maximum displacement error of 7.95 seconds indicates that stability is not decreased.

A deterioration of accuracy by a factor 7 (concerning the median displacement) may be an acceptable compromise, at least in some applications. It has to be considered that the compactness of feature representation was increased by a factor of 70 and the time of computation in the costly alignment step was reduced to about one tenth because, because no refinement step is needed.

6 CONCLUSIONS

In this paper we have explained a way to extract f_0-estimation features from spectrograms. We then used dynamic time warping in order to align such feature sequences to midi representations of the corresponding score. Since we used whole piano sonatas for our experiments a multiscale DTW approach had to be used in order to tackle complexity issues. Evaluations showed that the extracted feature can compete with other state-of-the-art features.

The actual benefit of the feature described here as well as the one proposed by [2] is that unlike others they are very sparse in nature. So they can easily be converted into a symbolic representation. Using additional pruning the accuracy is reduced significantly but on the other hand data reduction concerning the feature representation as well as during the alignment process is remarkable. Since we have demonstrated the capabilities of our original feature, the modification can be seen as a tradeoff between accuracy and computational costs.

A median displacement error of about 200 ms is too much for applications like performance analysis. But applications like content query engines might profit from such compact features. Especially in the context of huge databases fast and memory-saving routines can be of advantage over methods yielding the highest accuracy.

7 ACKNOWLEDGMENTS

This research is supported by the Austrian Fonds zur Förderung der Wissenschaftlichen Forschung (FWF) under project number P19349-N15.

8 REFERENCES

[1] Bloch, J. J. and Dannenberg, R. B. "Real-Time Accompaniment of Polyphonic Keyboard Performance", *Proceedings of the 1985 International Computer Music Conference*, pp. 279–290, Burnaby, BC, 1985.

[2] Cont, A. "Realtime Audio to Score Alignment for Polyphonic Music Instruments Using Sparse Non-negative constraints and Hierarchical HMMs", *IEEE International Conference in Acoustics and Speech Signal Processing (ICASSP)*, Toulouse, 2006.

[3] Dixon, S. "Live Tracking of Musical Performances Using On-Line Time Warping", *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx)*, Madrid, 2005.

[4] Hu, N; Dannenberg, R. B. and Tzanetakis, G. "Polyphonic Audio Matching and Alignment for Music Retrieval", *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New York, 2003.

[5] Itakura, F. "Minimum Prediction Residual Principle applied to Speech Recognition", *IEEE Transactions on Acoustics, Speech, and Signal Processing* vol. 23, pp. 52–72, 1975.

[6] Lawson, C. L. and Hanson, R. J. "Solving least squares problems", *Prentice Hall*, Lebanon, Indiana, 1974.

[7] Müller, M.; Kurth, F. and Clausen, M. "Audio Matching via Chroma-based Statistical Features", *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, 2005.

[8] Niedermayer, B. "Non-negative Matrix Division for the Automatic Transcription of Polyphonic Music", *Proceedings of the 9th International Symposium on Music Information Retrieval (ISMIR)*, Philadelphia, PA, 2008.

[9] Rabiner, L. R. and Juang, B.-H. "Fundamentals of speech recognition". Prentice Hall, Englewood Cliffs, NJ, 1993.

[10] Salvador, S. and Chan P. "FastDTW: Toward accurate Dynamic Time Warping in Linear Time and Space", *Intelligent Data Analysis*, vol. 11/5, pp. 561–580, Amsterdam, 2004.

[11] Sha, F. and Saul, L. "Real-time pitch determination of one or more voices by nonnegative matrix factorization", *Advances in Neural Information Processing Systems 17*. Saul, K.; Weiss, Y. and Bottou, L. Eds., MIT Press, Cambridge, MA, 2005.

[12] Soulez, F.; Rodet, X. and Schwarz D. "Improving Polyphonic and Poly-Instrumental Music to Score Alignment", *Proceedings of the 4th International Symposium of Music Information Retrieval (ISMIR)* Baltimore, MD, 2003.

[13] Tzanetakis, G.; Ermolinskyi, A. and Cook, P. "Pitch Histograms in Audio and Symbolic Music Information Retrieval", *Proceedings of the 3rd International Symposium of Music Information Retrieval (ISMIR)* Paris, 2002.

PREDICTING THE PERCEIVED SPACIOUSNESS OF STEREOPHONIC MUSIC RECORDINGS

Andy M. Sarroff and Juan P. Bello
New York University
andy.sarroff@nyu.edu

ABSTRACT

In a stereophonic music production, music producers seek to impart impressions of one or more virtual spaces upon the recording with two channels of audio. Our goal is to map spaciousness in stereophonic music to objective signal attributes. This is accomplished by building predictive functions by exemplar-based learning. First, spaciousness of recorded stereophonic music is parameterized by three discrete dimensions of perception—the width of the source ensemble, the extent of reverberation, and the extent of immersion. A data set of 50 song excerpts is collected and annotated by humans for each dimension of spaciousness. A verbose feature set is generated on the music recordings and correlation-based feature selection is used to reduce the feature spaces. Exemplar-based support vector regression maps the feature sets to perceived spaciousness. We show that the predictive algorithms perform well on all dimensions and that perceived spaciousness can be successfully mapped to objective attributes of the audio signal.

1 INTRODUCTION

Auditory spatial impression, or the concept of type and size of an actual or simulated space [1], helps a listener form judgements about auditory events and where those events occur. In natural acoustic settings, the relative positions of sound sources to each other, the relative positions of sound sources to a listener, the listener's and sources' relative positions to the surfaces of the listening environment, and the physical composition of the structures that form and fill the listening environment are factors that contribute to spatial impression.

In a stereophonic music production, music producers seek to impart impressions of one or more virtual spaces upon the recording with two channels of audio. Spatial cues are captured, manipulated, and added in order to provide the listener with impressions of simulated acoustic spaces, whether intentionally natural or unnatural sounding. The artful handling of these cues by producers can affect enjoyability of the listening experience.

SMC 2009, July 23-25, Porto, Portugal

Our goal is to successfully predict the spatial impression that stereophonic recorded music imparts. A robust predictive system can empower music producers, listeners, and consumers with perceptually meaningful ways to evaluate, manipulate, and manage their music. Top-down controls for spaciousness may help music makers sculpt their sound. Casual listeners may customize their experience by using "spaciousness" controls similar to the EQ controls ubiquitous in consumer reproduction systems. By giving humans such resources, the music making and listening experience will become more flexible and interactive.

We set about our task by parameterizing the concept of spaciousness with three dimensions. A data set of stereophonic music recordings is collected and subsequently annotated for each dimension of spaciousness. We then use exemplar-based learning to build functions that map objective measurements of digital audio to the annotated music recordings.

We have structured this paper as follows: Section 2 gives some background as to how others have dealt with spaciousness and describes our approach to predicting spatial impression. In Section 3, we detail the processes of music selection and annotation. Section 4 describes the learning algorithms that are used and their parameterizations. The algorithms are tested and subsequent results are discussed in Section 5. We end with concluding remarks and suggestions for future work in Section 6.

2 BACKGROUND AND APPROACH

Our approach is summarized in Figure 1. We begin with a set of musical recordings and end with three spatial dimensions, or "target concepts"—the width of the source ensemble, the extent of reverberation, and the extent of immersion (defined in Table 1). Prediction is accomplished by mapping subjective ratings to objective measurements by machine learning.

In this paper, we focus on three relations between listener and music—the source relation (width of ensemble), the environment relation (reverberation), and the global relation (immersion). They have been selected from an amal-

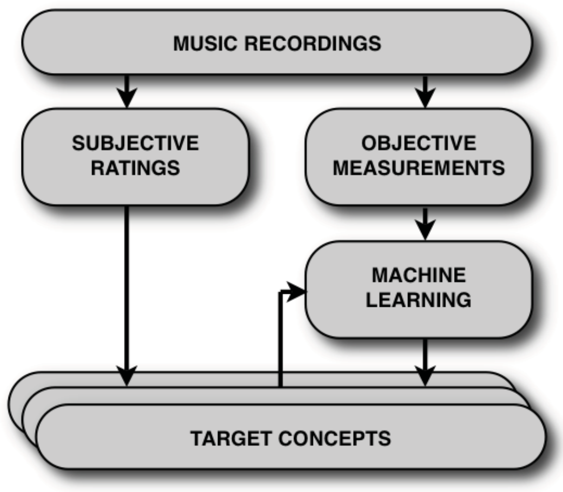


Figure 1. Framework for predicting perceived spaciousness of music recordings.

gamation of perceived attributes found in literature on natural acoustics and sound capture/reproduction. For both of these areas, there is an implicit need to rate the spatial quality of such systems. To do so, researchers must know the dimensions of spaciousness that are most salient to human listeners and to have a means of evaluating spatial quality along these dimensions.

In natural acoustics, spatial impression is divided into two primary dimensions, Apparent Source Width (ASW) and Listener Envelopment (LEV) [2, 3]. In sound capture and reproduction, the dimensionality of spaciousness is further demarcated. For example, [4] groups spatial attributes into descriptions that are related to sources, groups of sources, environments, and global scene parameters. We borrow from the literature of both fields for identifying salient spatial dimensions. ASW is defined as "the apparent auditory width of the sound field created by a performing entity" in [5] and "ensemble width" is the "overall width of a defined group of sources" in [4]. Both of these definitions connote one attribute that entails the width of the source ensemble. The extent of reverberation is directly linked to LEV in [5]. However, we use "immersion" to describe an attribute that encapsulates several kinds of envelopment, as is done in [4], and we treat immersion and reverberation independently, as is done in [6].

Once we have defined the spatial dimensions, or target concepts, we need a means of quantitatively evaluating them. In natural acoustics, objective measurements have been suggested numerous times, for example in [5] and [7]. Such measures quantify acoustic properties of physical space and relate these to perceived spaciousness. As recorded music only represents physical space virtually, measurements like

- The "width of the source ensemble" of a sound is how widely spread the ensemble of sound sources appears to be.
- The "extent of reverberation" of a sound is the overall impression of how apparent the reverberant field is.
- The "extent of immersion" is how much the sound appears to surround one's head.

Table 1. Definitions of learning concepts.

these do not serve our goals. To the best of our knowledge, the perception of spatial attributes has been addressed qualitatively, but not quantitatively, in sound capture and reproduction.

It is therefor necessary for us to newly construct a set of annotated music recordings and determine a quantitative relation. The target concepts cannot be divided into a semantically meaningful finite number of categories, so we impose a bounded arbitrary continuum and build a regression model for each concept. With the exception of listener experience, perceived attributes discussed in the literature are consistently related to sound sources or their environment, rather than personal properties like gender. These are universal in nature and therefor support a model which maps spaciousness to objective measurements of the recorded signal.

3 MUSIC SELECTION AND ANNOTATION

3.1 Music Selection and Segmentation

Fifty songs were selected from an online music database [8]. The songs were equally distributed across seven genre groups: "Alt/Punk," "Classical," "Electronic-Dance," "Hip-Hop," "R&B/Soul," and "Rock/Pop." An equal propagation of chorus, verse, and bridge segments were spread across the pool. A seven second segment was excerpted from each of the songs. The duration was chosen, by informal evaluation, to be long enough to develop concrete impressions of spaciousness, yet short enough to prevent much temporal variation in spaciousness within the excerpt. None of the recordings were from commercially available songs; it is therefore unlikely that songs would be recognizable and induce bias during human annotation.

3.2 Labeling

Human subject studies were conducted online and in a laboratory. In each, subjects were required to use headphones. First, basic demographic data was collected. Participants were mostly experienced music listeners, but varied in country of residence, age, gender, profession, and other attributes of demography. Subjects were given explicit explanations and definitions of the dimensions that they were to evaluate. For each of the terms, participants were asked to listen to a non-musical mixture of sources (a room of applause). This training phase was designed to give participants time to familiarize themselves with the concepts and focus their listening on a simple stimulus. The nonmusical recordings exhibited the spatial dimensions but, to avoid pre-biasing their judgments of spaciousness, participants were not told how spacious the recordings were to be perceived.

Subjects were asked to rate, on a bipolar 5-ordered Likert scale from "Less" to "Neutral" to "More," each of the dimensions for each song. There were a total of 98 participants providing 2,523 total ratings. Ratings were transformed from a Likert space to a numerical space by assigning the 5-ordered response categories integer values. For each song and dimension, all responses that were at least 3 standard deviations from the mean were removed as outliers. Any participant who had more than two outliers for a dimension was removed from that dimension. The responses for each dimension were standardized to zero mean and unit variance, and the mean for each dimension and song was calculated. The pairwise correlation coefficient R was calculated between ratings for the learning concepts. Width–immersion R was 0.87 and reverberation–immersion R was 0.57. Concepts width–reverberation were the least correlated, with a coefficient of 0.32, suggesting that subjects perceived differences between these dimensions unambiguously.

4 MACHINE LEARNING

A block diagram for building our objective-to-subjective mapping function is shown in Figure 2. At the beginning, we have a large feature space that objectively describes the music recordings. At the end, we have a support vector machine that needs optimization to accurately predict subjective ratings. In between, a correlation-based feature selection and subset voting scheme are used to narrow down the feature space. Then a grid search for the best parameterization of the support vector regression function is conducted. Each stage is described in detail below.

4.1 Feature Generation

A verbose set of attributes was batch-generated on the Left-Right difference signal of the data set using the MIR Toolbox [9] and two additional features. The batch-generated features include many that are widely used, like MFCCs, Spectral Centroid, and Spectral Flatness. The two additional features, which we have reported in [10], are non-standard but describe spatial characteristics of a signal. The first blindly estimates, through magnitude cancellation techniques, how widely a mixture of sources is distributed within the stereo field. The second uses the residual of a linear predictor as an indicator of how much reverberation a signal contains.

Category	**Feature**
Dynamics	RMS energy
Rhythm	Fluctuation Peak Position*, Fluctuation Peak Magnitude*, Fluctuation Spectral Centroid*, Tempo, Tempo Envelope Autocorrelation Peak Position, Tempo Envelope Autocorrelation Peak Magnitude, Attack Time, Attack Time Onset Curve Peak Position*, Attack Time Onset Peak Magnitude*, Attack Slope, Attack Slope Onset Curve Peak Position*, Attack Slope Onset Curve Peak Magnitude*
Timbre	Zero-Cross Rate, Spectral Centroid, Brightness, Spectral Spread, Spectral Skewness, Spectral Kurtosis, Roll-Off (95% threshold), Roll-Off (85% threshold), Spectral Entropy, Spectral Flatness, Roughness, Roughness Spectrum Peak Position, Roughness Spectrum Peak Magnitude, Spectral Irregularity, Irregularity Spectrum Peak Position, Irregularity Peak Magnitude, Inharmonicity, MFCCs, Δ MFCCs, $\Delta\Delta$ MFCCs, Low Energy*, Low Energy RMS, Spectral Flux
Pitch	Salient Pitch, Chromagram Peak Position, Chromagram Peak Magnitude, Chromagram Centroid, Key Clarity, Mode, Harmonic Change Detection
Spatial	Wideness Estimation*, Reverberation Estimation*
Summary Functions	Mean, Standard Deviation, Slope, Period Frequency, Period Amplitude, Period Entropy

Table 2. List of audio features, their categories, and summary functions. Features with an asterisk (*) only had their mean calculated.

For most features, the recording was frame-decomposed and feature extraction was performed on each frame. Some features, such as Fluctuation, were calculated on the entire segment. The frame-level features were summarized by their mean and standard deviation. Additionally, their periodicity was estimated by autocorrelation, and period frequency, amplitude, and entropy was calculated. The size of the final feature space extracted from the recordings was 430 dimensions. The entire set of features, which can be sub-divided into categories of Dynamics, Rhythm, Timbre, Pitch, and Spatial, is listed in Table 2.

4.2 Pre-Processing

The feature space was normalized to the range $[0, 1]$ and transformed into a principal components space. The non-principal components that accounted for the 5% least variance in the data set were discarded, and the data set was transformed back to its original symbolic attribute space.

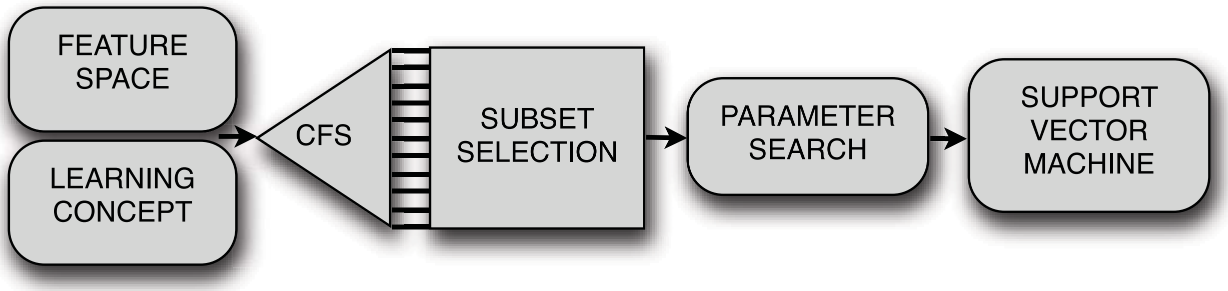


Figure 2. Block diagram for building and optimizing the mapping function.

4.3 Feature Selection

For each target concept, Correlation-Based Feature Selection (CFS) was performed with a greedy step-wise forward search heuristic. CFS chooses attributes that are well correlated to the learning target, yet exhibit low intercorrelation with each other. It has been shown to be good for filtering out irrelevant and redundant features [11].

However, supervised attribute selection can over-fit attributes to their learning concept when the same data set is used for training and testing [12]. To minimize subset selection bias, a percentile-based voting scheme with $10 \times$ 10-fold cross-validated attribute subset selection was performed. Multiple cross-validation (CV) is a robust way of estimating the predictive power of a machine when only a small data set is available. As each fold generated a different feature set, some features were selected more often than others. For each run, features were placed in a percentile bin based upon how many times that feature had been selected. Up to 11 new data sets with monotonically increasing feature spaces were generated in this way.

Each feature space was then used to learn a non-optimized support vector regression algorithm for each dimension. The subset that performed the best for each learning concept was voted as the final subset for further system optimization and training.

4.4 Regression

For each concept, a support vector regression model was implemented with the Sequential Minimal Optimization (SMO) algorithm [13]. Support vector machines have shown to generalize well to a number of classification and regression tasks. Our support vector models employed a polynomial kernel, $K(x, y) = (< x, y > +1)^p$, chosen as the best in an informal kernel search. Support vector machines perform, to some extent, similarly well independent of kernel type if the kernel's parameters are well-chosen [14]. An exhaustive grid search for the optimal values of the support vector machine complexity (C) and its kernel exponent (p) was conducted after the optimal feature space had been selected.

5 EXPERIMENTS AND RESULTS

For each dimension of spaciousness, the best feature space was found by using Multiple CV. Then we systematically searched for the support vector parameterization that yielded the lowest error for each concept. Success was evaluated by relative absolute error (RAE), which is insensitive to scale. RAE is the sum of all the errors normalized by the sum of the errors of a baseline prediction function, Zero-R. Zero-R picks the mean value of the test fold for every instance. An error of 0% would denote perfect prediction.

Figure 3 shows the results of testing for the best feature space percentile. All predictors show two local minima: Width at the 20^{th} and 50^{th} percentiles; reverberation at the 10^{th} and 40^{th} percentiles; and immersion at the 20^{th} and 70^{th} percentiles. This indicates that there might have been more than one optimal feature subset percentile to use. We have chosen the percentile that yielded the lowest RAE for the algorithm, without testing all local minima. The steepness of the error curves between the 0 and 10^{th} percentiles shows that simply using the entire feature set without any feature selection would greatly inhibit the performance of the support vector algorithm.

The final test results are depicted in Table 3. The mean absolute error (MAE), which is dependent on scale, was no more than 0.11 for any of the predictors. The average MAE for the Zero-R predictor is shown for comparison at the bottom of the table. The predictive capability of each of the machines was well above chance, as indicated by the RAE. All predictors had a correlation coefficient R of 0.73 or higher. An R value of 0.0 would denote a complete lack of correlation between the predicted and actual values. The predictor for wideness of source ensemble performed the poorest, but still well above chance. By all measurements of accuracy, the predictor for extent of reverberation performed the best.

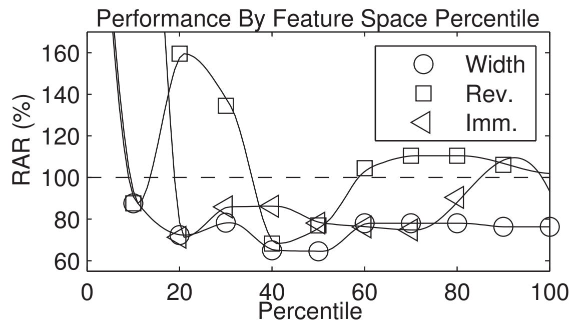


Figure 3. Performance of non-optimized machine on monotonically decreasing feature spaces.

Its coefficient of determination (R^2) indicates that the function accounted for 62% of the variance in the test set.

A summary of the final feature subset percentile used for learning each concept is shown in Table 4. While most features are probably not individually useful, the correct combination of features is. Features that were selected for more than one learning concept are shown in boldface. The width and immersion dimensions shared the most features in common; this is understandable, as these dimensions shared the highest correlation among annotations. Selected features for all three concepts were largely from the Timbre category. We find it interesting that the reverberation predictor picked three features from the Pitch category. We also note that the spatial estimators for wideness and reverberation were automatically chosen for the tasks of predicting source ensemble wideness and extent of immersion.

The error surfaces for parameterizations of each of the machines is shown in Figure 4. These surfaces show the RAE for each value in our grid search for optimal C and p values. It can be seen that the surfaces are not flat and that a globally optimal parameterization can be found for each. Yet they depict few local minima and are relatively smooth, suggesting that other parameter choices in between the grid marks would not have significantly improved results. It is worth noting that the flattest error surface, that for extent of reverberation, is also the one that performed the best, indicating robustness against parameter choices.

	Width	**Rev.**	**Imm.**
RAE(%)	62.63	67.20	64.36
MAE	0.11	0.10	0.11
R	0.73	0.79	0.76
R^2	0.53	0.62	0.58
MAE (Zero-R)	*0.19*	*0.17*	*0.18*

Table 3. The final mean absolute error (MAE), relative absolute error (RAE), correlation coefficient (R), and coefficient of determination (R^2) of the learning machines are given. The MAE for a baseline regression function, Zero-R, is given for comparison. All results are averaged from Multiple CV.

Concept (%-tile)	**Features**
Width (50 %)	**Tempo Envelope Autocorrelation Peak Magnitude Period Frequency**, **Spectral Flatness Period Amplitude**, **Wideness Estimation Mean**, **Reverb Estimation Mean**, **Δ MFCC Slope 5**, **$\Delta\Delta$ MFCC Mean 11**
Reverb. (40 %)	MFCC Mean 3, MFCC Period Entropy 3, MFCC Slope 3, $\Delta\Delta$ MFCC Period Amplitude 13, Key Clarity Slope, Chromagram Peak Magnitude Period Frequency, Harmonic Change Detection Function Period Amplitude, Spectral Flux Period Amplitude, Pitch Period Amplitude, Δ MFCC Slope 10, Δ MFCC Period Frequency 10, Δ MFCC Slope 13
Imm. (20 %)	MFCC Period Entropy 6, Spectral Centroid Period Entropy, **Tempo Envelope Autocorrelation Peak Magnitude Period Frequency**, **Spectral Flatness Period Amplitude**, Spectral Kurtosis Standard Deviation, **Wideness Estimation Mean**, **Reverb Estimation Mean**, Mode Period Entropy, Pitch Period Frequency, Δ MFCC Slope 7, **Δ MFCC Slope 5**, Δ MFCC Slope 11, Δ MFCC Mean 11, **$\Delta\Delta$MFCC Mean 11**

Table 4. Selected feature spaces after running on non-optimized machine. Features in boldface were picked for more than one learning concept.

6 CONCLUSIONS AND FUTURE WORK

We have presented a model for the automatic prediction of spaciousness in stereophonic music. We first parameterized the concept of "spaciousness" with the dimensions of source ensemble width, extent of reverberation, and extent of immersion. A verbose feature space of objective measurements was generated on a data set of human-annoted music recordings. Feature subset selection by percentile voting was used to narrow the feature space. The three target concepts were effectively learned by support vector regression with a polynomial basis function, achieving a direct mapping between signal attribute and subjective perception. Prediction for the extent of reverberation performed the best, while predictions for the wideness of the ensemble source and the extent of immersion performed slightly poorer relative to reverberation. All concept predictions exhibited RAE much better than chance.

This work is based on an assumption of independence between the learning concepts. Future work will include deeper examination of dimensional interdependency, exploration of other regressors, kernels, and feature selection algorithms, and increasing the size of our database.

The accuracies of the models suggest that objective measurements of digital audio can be successfully mapped to new dimensions of music perception. Such mappings may

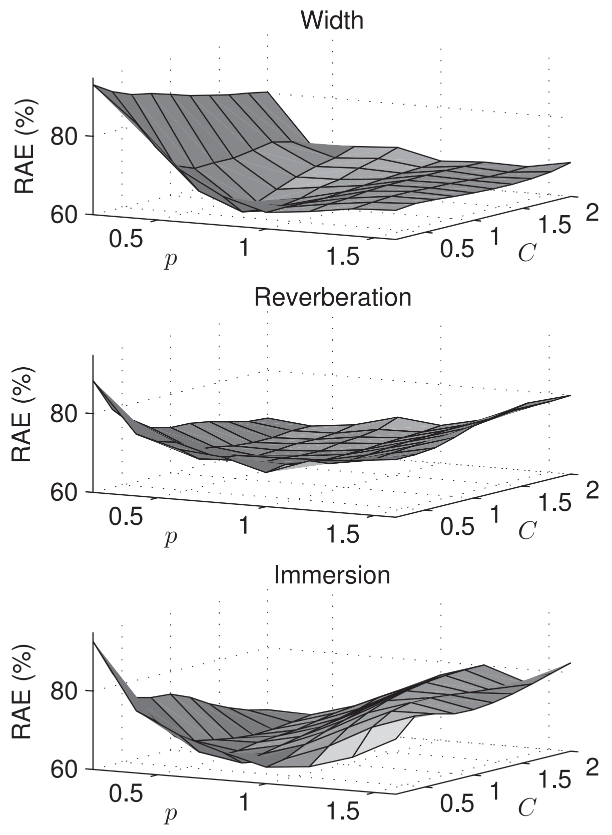


Figure 4. Relative absolute error surface for machine parameter grid search of kernel exponent p and machine complexity C.

allow music producers to have more control over their domain, including feature-driven audio synthesis and perceptually meaningful sound-sculpting. In addition, this work examines signal properties and perceptual attributes that can be tied directly to studio production of recorded music. Spaciousness is manipulated by the music engineer by applying a number of recording and signal processing techniques. By directly mapping signal attributes to the perceptual domain, music producers may gain new resources for their trade. We believe that the perceptual components of music listening that are affected by processes that occur in the production studio are a rich, yet under-exploited information stream to harvest.

7 REFERENCES

[1] J. Blauert and W. Lindemann, "Auditory spaciousness - some further psychoacoustic analyses," *Journal of the Acoustical Society of America*, vol. 80, no. 2, pp. 533–542, Aug 1986.

[2] W. Keet, "The influence of early lateral reflections on the spatial impression," in *6th International Congress on Acoustics, Tokyo, E-2-4*, 1968.

[3] M. Barron, "Late lateral energy fractions and the envelopment question in concert halls," *Applied Acoustics*, vol. 62, no. 2, pp. 185–202, 2001.

[4] F. Rumsey, "Spatial quality evaluation for reproduced sound: Terminology, meaning, and a scene-based paradigm," *Journal of the Audio Engineering Society*, vol. 50, no. 9, pp. 651–666, 2002.

[5] T. Okano, L. L. Beranek, and T. Hidaka, "Relations among interaural cross-correlation coefficient (iacc(e)), lateral fraction (lfe), and apparent source width (asw) in concert halls," *Journal of the Acoustical Society of America*, vol. 104, no. 1, pp. 255–265, Jul 1998.

[6] J. Berg and F. Rumsey, "Systematic evaluation of perceived spatial quality," in *Proceedings of AES 24th International Conference on Multichannel Audio*, Banff, Alberta, Canada, 2003. [Online]. Available: http://eprints.sics.se/888/01/aes24final.doc.pdf

[7] T. Hanyu and S. Kimura, "A new objective measure for evaluation of listener envelopment focusing on the spatial balance of reflections," *APPLIED ACOUSTICS*, vol. 62, no. 2, pp. 155–184, Feb 2001.

[8] Mp3 music downloads. [Online]. Available: http://www.mp3.com/

[9] O. Lartillot, P. Toiviainen, and T. Eerola. (2008) Mirtoolbox. [Online]. Available: http://www.jyu.fi/music/coe/materials/mirtoolbox

[10] A. Sarroff and J. Bello, "Measurements of spaciousness for stereophonic music," in *125th Convention of the Audio Engineering Society, San Francisco, USA. October, 2008*, 2008.

[11] M. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, University of Waikato, Department of Computer Science, Hamilton, New Zealand, April 1999.

[12] A. J. Miller, *Subset Selection in Regression*. CRC Press, 2002.

[13] J. Smola, Alex and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[14] B. Scholkopf and J. Smola, Alexander, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

A DYNAMIC ANALOGY BETWEEN INTEGRO-DIFFERENTIAL OPERATORS AND MUSICAL EXPRESSIVENESS

Giovanni De Poli, Antonio Rodà
Dept. Information Engineering
Padova University, Italy
giovanni.depoli@dei.unipd.it

Luca Mion
TasLab – Informatica Trentina
Trento, Italy
luca.mion@infotn.it

ABSTRACT

Music is often related to mathematics. Since Pythagoras, the focus is mainly on the relational and structural aspects of pitches described by arithmetic or geometric theories, and on the sound production and propagation described by differential equation, Fourier analysis and computer algorithms. However, music is not only score or sound; it conveys emotional and affective content. The aim of this paper is to explore a possible association between musical expressiveness and basic physical phenomena described by integro-differential operators.

1 INTRODUCTION

Intuitive awareness of the relationship between music and mathematics exists as early as Pythagoras, who made an attempt to investigate and to determine this relation through acoustics. Pythagoreans founded the quadrivium, the fourfold way of knowledge. They divided mathematical science in two parts: *how many* (i.e., discrete, quantity) and *how much* (i.e., continuous, magnitude). Each of these parts is further subdivided into either absolute (stable) or relative (in motion). Thus, the quadrivium consisted of arithmetic (discrete quantity which subsist in itself), music (discrete quantity which is related to another), geometry (continued magnitude immovable) and astronomy (continuous magnitude of self-motive nature). Since then, the link between music and mathematics has been very fruitful in defining harmony and consonance, scales and temperament (i.e., relation among pitches), or in describing the musical structures as symmetries, transpositions, etc. (i.e., a geometrical–spatial metaphor of melodic structure).

During the late sixteenth and early seventeenth centuries, music began to be recognised more as an art and to be treated as a language and analysed in expressive terms. During the same period, science was moving from theoretical to experimental. Attention to physical phenomena lead to the development of new mathematical theories to explain *change*, such as calculus and differential equations. Science started to investigate in mathematical terms the physical nature of music (i.e., sound production and acoustics). In the second half of the ninetieth century Helmholtz set the basis of scientific investigation of sound perception (i.e., psychoacoustics). If we look at the content of books on Music and Mathematics (e.g., [1]), we notice that the content includes mainly the relational and structural aspects of pitches, described using arithmetic or geometric theories, and sound production and propagation, described by differential equation, Fourier analysis and computer algorithms. Moreover recently music performance, which acts as mediation between composer and listener, is increasingly being studied in scientific terms, and mathematical models are being developed. The pioneering model [7] of musical expressiveness in music performance, based on kinematics, is particularly relevent for the aims of this paper. ,

However music is not only score or sound; it can convey also sensorial and/or affective contents. A piece of music can suggest a light or a heavy sensation, an happy or a sad emotion. Obviously, the fact that a listeners judges a piece of music as light does not mean that this is the content of that music. In fact, sensorial or affective adjectives can be considered as metaphors, by means of which a listener translates his physical and cognitive experience of music [4]. The aim of this paper is to start to explore if and how the aspects of musical expressiveness related to sensorial or affective experiences can be associated to basic physical phenomena described by integro-differential operators.

2 MATHEMATICS AND MUSIC EXPRESSIVENESS

In mathematics, an *operator* is a function which operates on (or modifies) another function. Often, an operator is a function acting on functions to produce other functions. Given a function $f(t)$, the simplest operators in mathematical analysis are the *differential* operator $\mathcal{D}$, which defines the derivative of a function,

$$\mathcal{D}f(t) = \frac{d}{dt}f(t), \tag{1}$$

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

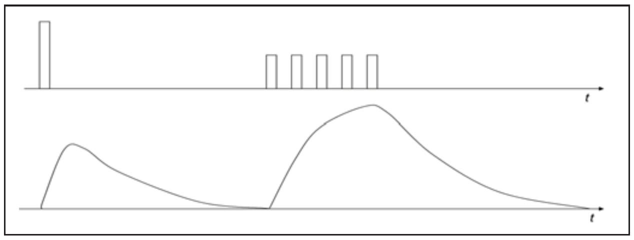


Figure 1. Analogy between time evolution of emotion and a ringing bell. Top: The pattern of strikes and intensity applied to a bell. Bottom: Intensity of the sound produced by the bell.

the *integral* operator $\mathcal{I}_0$, which defines the integral of a function with initial point $t = 0$

$$\mathcal{I}_0 f(t) = \int_0^t f(\tau) d\tau, \tag{2}$$

and the *proportional* operator, which scales the function by a constant k

$$\mathcal{K} f(t) = k f(t). \tag{3}$$

In the field of affective computing, Picard [6] outlines a signal representation for emotions and moods (emotional states lasting more than a couple of minutes) by a linear system preceded by a non-linearity. Picard uses the analogy of a ringing bell to illustrate the time course of emotion: both the bell sound and the emotional response, when triggered, have a fast rise time followed by a more gradual decay. The response sums if re-triggering occurs before the signal has completely decayed away. The intensity of the bell sound – and by analogy the intensity that a person might assign to their felt emotion – are shown in Fig. 1, when triggered by a single or repeated excitations. The influence of the person temperament is taken into account by the parameters of the linear model. Saturation and threshold effects are modelled by a smooth (sigmoid) non linear function applied to the inputs of the emotional system. The parameters of the function can be set according to the personality, mood, cognitive expectation, activation and arousal level of a person.

This analogy between time evolution of sound intensity and of emotion intensity suggested us to look for a possible generalization, in order to derive an analogy between integro-differential operators and musical expressiveness. In fact, the idea of using linear systems to model affective behaviour can be further developed. The basic building blocks of linear systems are the proportional, integral and derivative transformations, and they are described respectively by

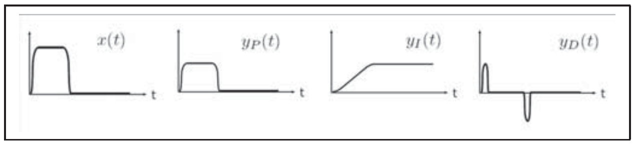


Figure 2. Functional transformation produced by proportional, integral and differential operators to a smoothed and large pulse $x(t)$ as input function.

the following equations:

$$y_P(t) = \mathcal{K}x(t) = k_P x(t) \tag{4}$$
$$y_I(t) = k_I \mathcal{I}_0 x(t) = k_I \int_0^t x(\tau) d\tau + y_I(0) \tag{5}$$
$$y_D(t) = k_D \mathcal{D} x(t) = k_D \frac{d}{dt} x(t) \tag{6}$$

where $x(t)$ is the input and $y_P(t)$, $y_I(t)$, $y_D(t)$ are the outputs of the basic blocks. Fig. 2 shows the behaviour of the proportional, integral and differential operators to a smoothed and large pulse $x(t)$ as input function.

3 EXPERIMENT

In order to verify if a metaphor based on $\mathcal{KID}$ operators can appropriately describe some expressive characteristics of music content, we conducted an experiment which investigates subject's associations between two sets of musical stimuli and three haptic attractors, that we assumed to be representatives of the three components of the $\mathcal{KID}$ metaphor.

3.1 Experiment design

Procedure. Participants were asked to listen to each musical excerpt and to associate it to one of the three attractors. Participants were allowed to listen to the excerpts and to test the attractors as many time as wished, and to change their responses until they were satisfied by their choices. The attractors induced the listeners to organize the musical excerpts on the basis of similarity criteria which depend on the characteristics of the attractors. Therefore, we can expect that different set of attractors will induce a different mental organization of musical excerpts. On the other hand, the features of the set of music excerpts can influence the results as well.

Materials. Two sets of musical stimuli were used in the experiment setup. The first set of musical excerpts comprise a subset of the performances of professional instrument players used in [5]: the theme from Händel's Sonata HWV 379 in E Minor Op. 1 No. 1 (Adagio) and the traditional song Twinkle Twinkle Little Star were played by flute and violin (Fig. 3). Each melody were played more times,

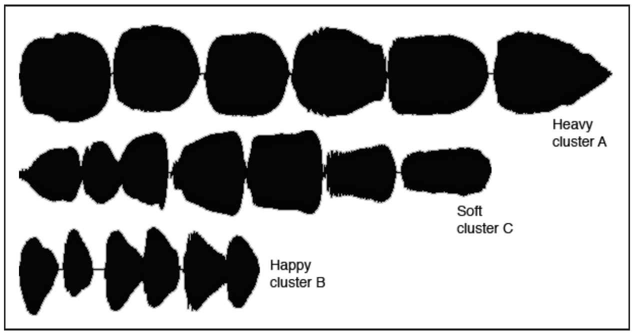


Figure 3. Simple musical stimuli belonging to different clusters: The first six notes of Twinkle Twinkle Little Star played by violin with respectively Heavy, Soft and Happy expressive intention.

with different expressive nuances, in order to convey expressive intentions Happy, Sad, Angry and Calm (affective space), Light, Heavy, Soft and Hard (sensorial space [3]) plus a Neutral performance. In total we took into account 2 (instruments) x 2 (pieces) x 9 (adjectives) = 36 examples with an average duration of 30s. These excerpts have a plain musical structure constituted by a tonal melody played by a single instrument, so we will refer to them as *simple musical stimuli*. They are oriented to study expressive aspects related to performance and they have already been analyzed by an acoustic point of view [5]. Moreover, the labels chosen to characterize the different performances allow a comparison between affective and sensorial space.

The second set is constituted by the data used in [2] to study the emotion communication in music: 27 musical stimuli extracted by recordings belonging to the Western music repertoire of the classic-romantic period. As these excerpts have a polyphonic structure with two or more instruments playing together, we will refer to them as *complex musical stimuli*. By means of a perceptual test Bigand and coworkers [2] found that these musical stimuli are organized along two dimensions, which were interpreted as associated to the emotional properties of valence and arousal, and are arranged in four groups, characterized by different levels of valence and arousal: a cluster with high arousal and high valence (HAHV); a cluster with high arousal and low valence (HALV); a cluster with low arousal and high valence (LAHV); a cluster with low arousal and low valence (LALV). These clusters roughly correspond to Happy, Angry, Calm and Sad adjectives respectively.

The set of attractors is composed by three haptic stimuli synthesized by means of a Phantom Omni haptic device [1], which simulates the basic effect of a mechanical mass–spring–damper system.

All the force feedback are omni-directional: the device reacts to the user's input in every points of the haptic sphere. Regarding the stimulus E (*elasticity*), the device generates a force feedback with intensity

$$f_E(t) = -K_{el} \cdot ||s(t) - s_0|| \tag{7}$$

where s_0 is the center of the haptic sphere, $s(t)$ is the position of the stylus at the instant t and K_{el} is the elasticity constant of the system. The stimulus F (*friction*) is characterized by a force feedback proportional to the velocity of the user's movement:

$$f_F(t) = -\eta_v \cdot v(t) \tag{8}$$

where $v(t)$ is the velocity of the stylus and η_v is the viscosity constant of the system. Finally, the stimulus I (*inertia*) simulates the interaction with an inertial mass m, moving in a field free of other (gravitas or magnetic) forces. The mass m is coupled to the stick, that we assume to have a negligible inertial mass. The intensity of the force follow the equation:

$$f_I(t) = -m \cdot a(t) \tag{9}$$

where $a(t)$ is the stylus acceleration. After several tests, we set $m = 0.5$ Kg, $K_{el} = 510$ N/m, and $\eta_v = 31.9$ Ns/m.

When using the dynamic analogy, force $f(t)$ is often subjectively considered as the cause and movement (velocity $v(t)$) as the effect. Thus, we are induced to associate Elastic attractor to $\mathcal{D}$ operator, Friction to $\mathcal{K}$ and Inertia to $\mathcal{I}_0$. This association constitutes the basis of our dynamic analogy.

A preliminary experiment was carried out with the aim of testing if the three haptic stimuli were perceived as different by the subjects. We presented in a random order nine stimuli, three equal stimuli for each type of feedback. Subjects were asked to group the stimuli according to their similarity. All the 20 subjects correctly grouped the stimuli in three categories, each one comprehending the three equal stimuli.

Apparatus. The sound files were represented on the computer screen by a visual interface implemented using the real-time sound synthesis environment PD (Pure Data). The interface consists on 3 buttons displayed on the top, associated to the 3 attractors, and on a set of buttons listed in column associated to the musical stimuli which are presented (in random order) to the participants, who were allowed to listen to the excerpts and to the attractors as many time as wished just by pressing the related button. Each of the buttons was also associated to a radio button where the participants could select one attractor only, by employing a three-alternative forced-choice (3AFC) method.

Participants. A total of 39 subjects participated to the experiment. Of these, 18 subjects (13 male, 5 female) were presented to the simple musical stimuli: 4 subjects had a professional musical training for 5 years at least and were referred as musicians (M); 4 subjects play an instrument and they are referred as amateurs (A); 10 subjects did not have any musical training and were referred as non-musicians

[1] http://www.sensable.com/haptic-phantom-omni.htm

stimuli	Friction	Elasticity	Inertia	resulting clusters
Angry	41	25	6	
Hard	40	17	15	A
Heavy	31	9	32	
Happy	10	60	2	B
Light	15	47	10	
Calm	13	5	54	
Sad	11	3	58	C
Soft	21	18	33	
Neutral	28	18	26	

Table 1. Contingency table of the subjects' responses in the experiment with the simple musical stimuli and resulting clusters.

(N). Participants were aged from 23 to 34 years (26 years average). The duration of the test was about 15 minutes. The other 21 subjects (7 male, 14 female) were presented to the complex musical stimuli: 3 subjects had a professional musical training for 5 years at least and were referred as musicians (M); 7 subjects play an instrument and they are referred as amateurs (A); 11 subjects did not have any musical training and were referred as non-musicians (N). Participants were aged from 22 to 53 years (30 years average). The duration of the test was about 20 minutes.

4 RESULTS

Simple musical stimuli. Subjects' responses were summarized into a two-way contingency table containing 9 rows (expressive intentions) and 3 columns (attractors Friction, Elasticity, and Inertia). Each cell of the table recorded the number of times (observed frequency) that each expressive intention was associated to an attractor (Tab. 1). Pearson's χ^2 test was used to compare observed frequencies with expected frequencies under the null hypothesis of independence, yielding a total value of $\chi^2 = 305.96$ ($df = 16$, $p < 0.001$), denoting strong evidence of a relation between expressive intentions and attractors. The analysis of each row, taken individually, shows that the association among expressive intentions and attractors is confirmed by Angry, Hard, Happy, Light, Sad, Calm ($\chi^2 > 21.9$, $df = 1$, $p < 0.001$), and weakly confirmed by Heavy ($\chi^2 = 5.0$, $df = 1$, $p < 0.05$). On the contrary, no significant relation among expressive intention and attractors has been found for Neutral and Soft.

We conducted two analysis to investigate the association between expressive intentions and attractors: a Simple Correspondence Analysis and a K-means clustering. The contingency table was submitted to Simple Correspondence Analysis in order to graphically represent the degree of association between expressive intentions and attractors according to their χ^2 distances. Since Simple Correspondence Analysis is applied on a 3-columns table (i.e., represented

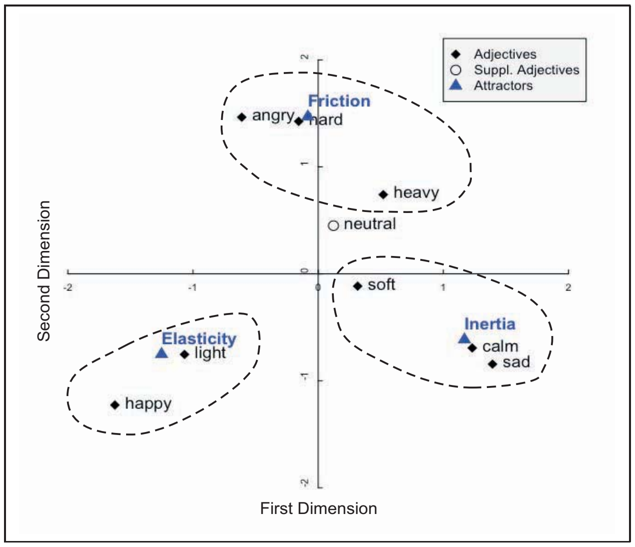


Figure 4. Bi-plot of the correspondence analysis on the experiment with simple musical stimuli.

by two degrees of freedom), we can derive two dimensions only (Fig. 4) with eigenvalues covering the total variance (first eigenvalue explains the 75.52% of the total inertia). The Neutral expression was considered as supplementary row (not used to perform the previous analysis) and its projection into the correspondence plot resulted close to the origin of the axis. We can see in Fig. 4 that the expressive intentions Angry-Hard-Heavy, Happy-Light, Calm-Sad-Soft are depicted close to attractors Friction, Elasticity and Inertia respectively. Then, we applied the K-means clustering (number of groups = 3) to the coordinates of the points in the Correspondence Plot in order to identify the stable groups. We did not take into account the Neutral intention. Three stable groups (see dashed lines in Fig. 4) were identified corresponding to the clusters: (A) Angry-Hard-Heavy, (B) Happy-Light and (C) Calm-Sad-Soft (see last column of Tab. 1). By grouping these expressive intentions according their cluster membership we found strong relation among clusters and attractors ($\chi^2 = 253.0$, $df = 4$, $p < 0.001$). Moreover, significant relation has been found between A cluster and F attractor, B cluster and E attractor and C cluster and I attractor ($\chi^2 > 65.5$, $df = 1$, $p < 0.001$).

It can be noticed that these clusters are consistent with the results of [5], where the same groups were obtained by clustering the same musical excerpts on the basis of their significant musical/acoustic features. It was found that cluster A is mainly related to Sound Pressure Level and some spectral cues, cluster B to Note Per Second and C to Attack time. These associations are confirmed by the canonical correspondence analysis on the data of our experiment.

Complex musical stimuli. Table 2 shows the subjects' responses with rows representing the 27 musical excerpts

clusters [2]	stimuli	Friction	Elasticity	Inertia
HALV	B12	7	13	1
	B16	9	8	4
	B17	4	15	2
	B18	9	8	4
	B25	9	5	7
	B26	12	5	4
	B27	5	8	8
HAHV	B10	9	9	3
	B11	12	3	6
	B13	9	12	0
	B14	7	12	2
	B15	6	14	1
	B22	7	11	3
	B23	7	11	3
	B24	6	13	2
LALV	B3	3	3	15
	B7	3	3	15
	B8	3	2	16
	B9	3	5	13
LAHV	B1	3	3	15
	B2	8	8	5
	B4	9	1	11
	B5	8	1	12
	B6	4	1	16
	B19	10	4	7
	B20	9	3	9
	B21	7	3	11

Table 2. Contingency table of the subjects' responses in the experiment with the complex musical stimuli.

and columns representing the 3 haptic attractors. The Pearson's χ^2 test denotes a strong relation between musical excerpts and attractors ($\chi^2 = 205.2, df = 52, p < 0.001$) and confirmed that subjects are able to distinguish the different haptic attractors and to use them to classify the musical excerpts.

The contingency table was submitted to Simple Correspondence Analysis in order to graphically represent the degree of association between musical stimuli and attractors (see Fig. 5). Then, we proceeded with a K-means analysis in order to identify clusters of stimuli. After several trials, we set the number of groups both to 3 and to 5 (respectively continuous and dashed lines in Fig. 5). Three of the five clusters include those stimuli which were associated to one of the haptic attractors. The other two clusters are composed by stimuli that subjects associated equally to two attractors: Elasticity - Friction and Inertia - Friction.

5 DISCUSSION

The results of the experiment with simple musical stimuli support a strong relation between the cluster Hard-Heavy-Angry and the Friction attractor, between the cluster Light-Happy and the Elasticity attractor, and between the cluster Sad-Calm-Soft and the Inertia attractor. Although from a semantic point of view these associations are not surprising (when I am angry I feel friction with someone; when I am

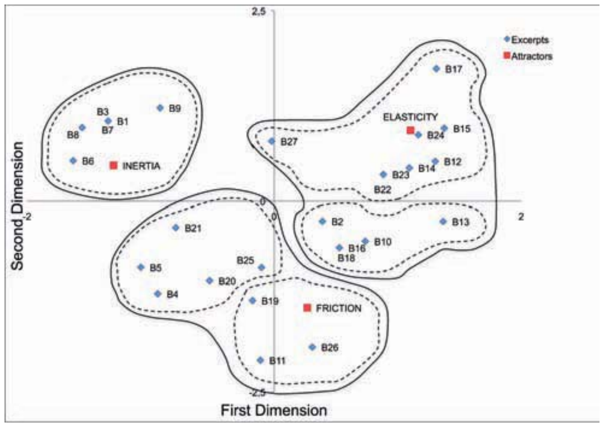


Figure 5. Correspondence analysis on experiment with complex musical stimuli.

happy I jump of the joy; when I am calm I move slow as an object with an high inertia), what is interesting is that the subjects were able to make this associations without any explicit semantic mediation, directly associating a musical stimulus to a haptic one.

In general, the subjects were able to consistently recognize common characteristics between musical stimuli and haptic attractors. Concerning the single expressive intentions, Neutral intention was not recognized as related to one single attractor, but the contingency table (Tab. 1) shows a balanced contribution of all the three attractors, as we could expect due to its meaning. It is interesting to note that, in some cases, the scores in the contingency table (Tab. 1) suggest the idea the three attractors Friction, Elasticity, and Inertia constitute a sort of basic components; the various expressive nuances can be represented as a combination of these components. E.g., Heavy performance was perceived as related not only to Friction, but also to Inertia, as we could expect. Moreover we can notice that the emotional response to a stimulus, as shown in Fig. 1, has a low pass characteristics: i.e. it can be modelled by a combination of $\mathcal{K}$ and $\mathcal{I}_0$ operators. This fact is in agreement with the projection of the Neutral expression in Fig. 4, which tends to be located between Friction and Inertia attractors.

The results of the experiment with the complex musical stimuli support a relation between the HAHV (Happy) cluster and the Elasticity attractor (confirmed by all the excerpts except for B10 and B11), and between the LALV (Sad) cluster and the Inertia attractor. On other cases, subjects responses are divided between two attractors: e.g., excerpts B4, B5, B20, and B21 are associated both to Inertia and Friction. This observation is coherent with the Fig. 5, where two clusters are composed by stimuli that subjects associated to two attractors: Elasticity-Friction and Inertia-Friction.

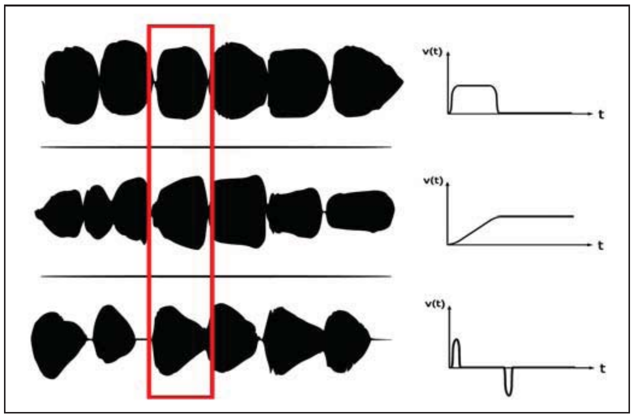


Figure 6. Comparison of sound envelope of expressive performances (Heavy, Soft and Happy) of Fig. 3 with the integro-differential operators $\mathcal{K}$, $\mathcal{I}_0$, $\mathcal{D}$ of Fig. 2.

With reference to our dynamic $\mathcal{KID}$ analogy, the cause-effect relation is represented by the mechanical admittance Y which mathematically describes the dynamic mapping and the qualitative behaviour from force $f(t)$ to velocity $v(t)$ by a linear combination of $\mathcal{K}, \mathcal{I}_0, \mathcal{D}$ operators or, physically, by a combination of friction, inertia and elasticity. We can distinguish resistive admittance which dissipates energy, from reactive impedance which stores energy. Ideal friction (operator $\mathcal{K}$) is a pure resistive admittance, while ideal inertia and elasticity (operators $\mathcal{I}_0$ and $\mathcal{D}$) are pure reactive admittances: in particular inertia stores kinematics energy and it opposes changes in movement, while elasticity stores potential energy and opposes changes in force.

In Fig. 6 we compare the sound intensity envelopes of 3 stimuli, belonging to the different clusters, with the velocity $v(t)$ resulting by applying the operator to a smoothed large force pulse $f(t)$ (see first graph in Fig. 2), or equivalently with the effect of using the force pulse as input to the corresponding haptic attractor. It can be seen that friction acts as a scaling factor of the input force and does not modify the shape of the input. The inertia (mass) tends to remain at its initial velocity (which is zero in the present example), then it grows progressively and remains constant when the input stops; the mass progressively augments its kinetics energy. The elasticity (spring) instead reacts immediately to the input variations; it stores potential energy which is used to oppose to force changes. From this qualitative description of the dynamic behaviour of the three basic elements we are induced to confirm the association of operator $\mathcal{K}$ to the cluster Hard/Heavy/Angry, $\mathcal{I}_0$ to cluster Sad/Calm/Soft, and $\mathcal{D}$ to cluster Light/Happy. Friction, elasticity and inertia are the basic properties of ideal mechanical systems and the dynamic behaviour of each real system depends by a weighted combination of friction and elasticity or of friction and inertia, where friction represents the quantitative aspect of the dynamical behaviour and elasticity/inertia represent the qualitative aspect of dynamical behaviour. The quantitative aspect can be associated to the vertical axis of Fig. 4 and 5, while the qualitative aspect can be associate to the horizontal axis.

Comparing these results we can notice that, although subjects are able to recognize the different haptic feedback, the $\mathcal{KID}$ metaphor seems to be more suitable for representing expressive cues in simple musical excerpts (where the expressive content is mainly related to performance cues) than in complex musical stimuli (where musical structure is more relevant). This result can be explained by the fact that music performance is more related to action–based aspects, whereas musical structure can involve aspects related to cognitive and/or cultural factors. Moreover, "real" music pieces usually are not characterized by a single expressive intention, but rather by a mixture of expressive nuances.

6 CONCLUSIONS

We proposed a dynamic analogy based on proportional operator $\mathcal{K}$, integral operator $\mathcal{I}_0$ and differential operator $\mathcal{D}$, and we carried out an experiment with the aim of investigating the relation between the $\mathcal{KID}$ metaphor and expressive music contents. The results let us hypothesize that relevant expressive characteristics of music can be associated to a weighted combination of quantitative and qualitative basic components described respectively by the $\mathcal{K}$ operator and by the opposition of $\mathcal{I}_0$ vs. $\mathcal{D}$ operators.

References

[1] D. Benson. *Music: A Mathematical Offering*. Cambridge University Press, 2006.

[2] E. Bigand, S. Vieillard, F. Madurell, J. Marozeau, and A. Dacquet. Multidimensional scaling of emotional responses to music: The effect of musical expertise and of the duration of the excerpts. *Cognition and Emotion*, 19(8):1113–1139, 2005.

[3] S. Canazza, G. De Poli, A. Rodà, and A. Vidolin. An abstract control space for communication of sensory expressive intentions in music performance. *Journal of the New Music Research*, 32(3):281–294, 2003.

[4] M. Leman. *Embodied Music Cognition and Mediation Technology*. The MIT Press, 2007.

[5] L. Mion and G. De Poli. Score-independent audio features for description of music expression. *IEEE Transactions on Speech, Audio and Language Processing*, 16(2):458–466, 2008.

[6] R. W: Picard. *Affective computing*. MIT Press, 1997.

[7] N. P. Todd. The kinematics of musical expression. *Journal of the Acoustical Society of America*, 97(3):1940–1949, 1995.

JACKTRIP/SOUNDWIRE MEETS SERVER FARM

Juan-Pablo Cáceres & Chris Chafe
Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University
{jcaceres,cc}@ccrma.stanford.edu

ABSTRACT

Even though bidirectional, high-quality and low-latency audio systems for network performance are available, the complexity involved in setting up remote sessions needs better tools and methods to asses and tune network parameters. We present an implementation of a system to intuitively evaluate the Quality of Service (QoS) on *best effort* networks.

In our implementation, musicians are able to connect to a multi-client server and tune the parameters of a connection using direct "auditory displays." The server can scale up to hundreds of users by taking advantage of modern multi-core machines and multi-threaded programing techniques. It also serves as a central "mixing hub" when network performance involves several participants.

1 INTRODUCTION

Systems for real-time, high-quality and low-latency audio over the Internet that take advantage of high-speed networks are available and have been used in the last several years for distributed concerts and other musical applications [12]. The difficulty of setting up one of these distributed sessions is, however, still very high. Most musicians have experienced the disheartening amount of time that can be lost in rehearsal, where most of the time is spent adjusting the connection rather than playing music.

Keeping delay to a minimum is one of the main goals when tuning network parameters. Delay is known to be disruptive in musical performance [4], so a sensible goal is to minimize it as much as possible. Often, there is a tradeoff with audio quality. The longer the latency, the better the audio (i.e., less dropouts) if facing problematic network conditions. For most users that are not familiar with TCP/IP network protocols [1] and delivery, understanding the meaning of these parameters can be daunting.

We present here a server-based application that can be of use to intuitively tune these parameters using "auditory displays" [5]. With it, musicians tune their network connection much like they do their instruments, using their ears. The implementation is part of the JackTrip application [3], a software for low-latency, high quality and multi-channel audio streaming over TCP/IP Wide Area Networks (WAN).

The design and architecture is first geared towards implementation of this QoS evaluation method. The architecture has also been extended to provides other types of service. In particular, a central "mixing hub" to control audio in a concert where multiple locations are involved.

[1] In particular, we use the User Datagram Protocol (UDP) which is part of the TCP/IP protocol suite.

2 QoS EVALUATION METRICS

Cromer gives a good definition of QoS:

> "The term *Quality of Service (QoS)* refers to statistical performance guarantees that a network system can make regarding loss, delay, throughput, and jitter." [7, p. 510]

Most of the networks available today are *best effort delivery*, i.e., don't provide any specific level of QoS. As such, this infrastructure can be problematic since sound is unforgiving in regard to packet loss and jitter; any lost data is immediately audible. In evaluating a particular connection, we want to know "instantaneous" QoS, i.e., assessing its quality at any given moment. Users should be able to adjust their settings to achieve the optimal quality given the current bandwidth and congestion conditions. This should be convenient and a conscious part of setting up, it should also be monitored with regard to longer-term changes: a connection that is perfectly clean at 1:00 a.m. can become congested at 9:00 a.m. A bad connection today can be a surprisingly good one a year from now when intermediate network upgrades are put in place, or when the user asks that their service be enhanced.

A connection is presently either tuned by trial and error, or is set automatically by an adaptive mechanism that changes the data rate depending on bandwidth availability [11]. Adaptive methods are typically found in unidirectional streaming and have a disadvantage for bidirectional high-quality audio. Latency is a parameter we want to keep constant. To accommodate changing amounts of jitter, adaptive methods can arbitrarily increase and decrease the local buffering, affecting total latency in a way that is very disruptive for musical performance.

We describe an implementation of a tool that let musicians tune a connection completely by ear. Parameters like buffer size, sampling rate, packet size, and packet redundancy among others, can be adjusted using this "auditory display" mechanism.

2.1 *Pinging* the network, acoustically

The advantages of evaluating very fine-grained jitter and packet loss using these "auditory displays" have been previously discussed in the literature [5]. The method consists of listening to a pitched sound in order to assess delay, jitter, and loss. The procedure produces a tone by recirculating audio in the network path and thus allows for fine-grained listening of the packet flow[2]. The acronym, SoundWIRE, describes the technique used in this project, sound waves on the Internet from real-time echoes. In principle, it uses the Karplus-Strong plucked string synthesis algorithm [9] and simply replaces string delay lines running in local host memory with network memory.

This technique can be extended to incorporate different-sounding auditory pings using other physical models [6], but the underlying approach is the same. In the case of, e.g., a string physical model, musicians want to tune their connection to get a sounding instrument that has the highest possible pitch (low delay) without vibrato (jitter). Users also want to minimize extraneous impulses coming from packet loss.

In the next section (Sec. 3), we present an architecture of a server that clients can use to evaluate and tune their connection solely based on auditory feedback, much like guitar players tune their instruments.

3 MULTI-CLIENT CONCURRENT SEVER

We extended the JackTrip platform to include a system for QoS evaluation. The new architecture provides a multi-client concurrent server that can be used to provide QoS evaluation service, or a central network/mixer hub, among other uses. Taking advantage of multi-core computers, it is possible to run concurrently hundreds of clients with uncompressed real-time audio and processing plugins.

3.1 Server architecture

The User Datagram Protocol (UDP) is a connection-less protocol and consequently identification of a new client's IP number has to be done on a packet-per-packet basis. Several techniques to deal with multiple clients connecting are discussed in the literature [13], but no standard exists as in the case of Transmission Control Protocol (TCP) servers (see [7] or [10] for a good description of the differences between TCP and UDP protocols.)

[2] The granularity is determined by the sampling rate and the packet size. e.g., at 48kHz and 64 samples/packet, the granularity is 1.3 milliseconds.

Two different but related techniques are implemented. The first relies on a "smart" client which can change to a new server port number after being assigned one for exclusive communication. The second uses Linux's *iptables* rules to route clients into local sockets. The former technique has the advantage of being portable (works currently on both Linux and OS X, and should be easily portable to Windows), is lightweight and doesn't require root privileges. In turn, it expects its clients to change connection ports. The latter technique (Linux only) requires *iptables* privileges but provides a mechanism whereby "dumb" clients (e.g., embedded systems) can connect to a unique IP/Port pair without any change to their behavior. It is also computationally more expensive because the kernel has to perform a port redirect by source IP number for every packet.

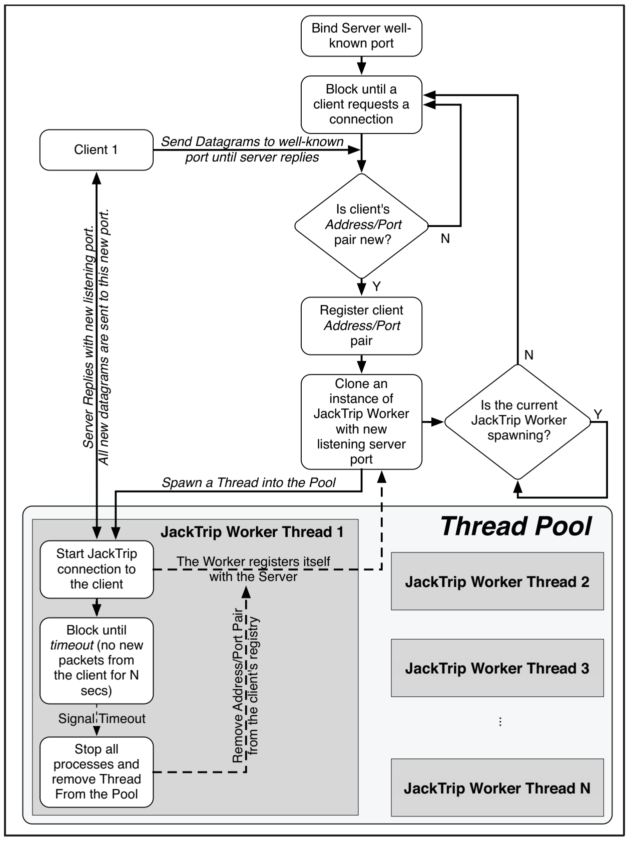

Bind Server well-known port
Block until a client requests a connection
Client 1
Send Datagrams to well-known port until server replies
Is client's Address/Port pair new?
N
Y
Register client Address/Port pair
N
Clone an instance of JackTrip Worker with new listening server port
Is the current JackTrip Worker spawning?
Y
Server Replies with new listening port. All new datagrams are sent to this new port.
Spawn a Thread into the Pool
JackTrip Worker Thread 1
Thread Pool
Start JackTrip connection to the client
The Worker registers itself with the Server
JackTrip Worker Thread 2
Block until timeout (no new packets from the client for N secs)
Remove Address/Port Pair from the client's registry
JackTrip Worker Thread 3
Signal Timeout
Stop all processes and remove Thread From the Pool
JackTrip Worker Thread N

Figure 1. Multi-client concurrent server algorithm

Figure 1 describes the architecture of the system. The server listens on a well-known port for client connection requests. For every new request, the server has to check if the originating address/port pair is new. If it is, it registers it in an array of active address/port pairs and blocks the requests of new clients while this one is being processed. It then allocates a new port to communicate exclusively with this client, and informs it of the new port. The client then stops sending packets to the well- known port and starts to send them to its own assigned one. From then on, the whole JackTrip process is sent to a thread pool and runs independently, in its own thread. The server is freed to wait for new client requests. The thread runs until the client stops sending packets (or the server doesn't receive them) for a certain amount of time. At that point a signal is emitted and the server deletes the client IP/port pair from the active clients registry and removes the process from the thread pool. The implementation is written in C++ using the Qt libraries [1] for networking and multithreading.

The architecture that uses Linux's *iptables* is similar, except that all port determination work is on the server side, and packets are redirected to a local IP/Port pair assigned exclusively to the client. It doesn't need to be notified of a new port.

4 SERVER APPLICATIONS

4.1 Quality of Service (QoS) evaluation

Each connection between the client and the server recirculates audio and implements a Karplus-Strong string model [9]. This configuration has been discussed in detailed previously [6]. Figure 2 shows a basic implementation of the algorithm. The ipsi-lateral host (which in our system corresponds to the server) generates excitations (plucks or noise bursts) that are "echoed" back from the contra-lateral host (the client) recirculating in a loop that includes a low-pass filter (LPF).

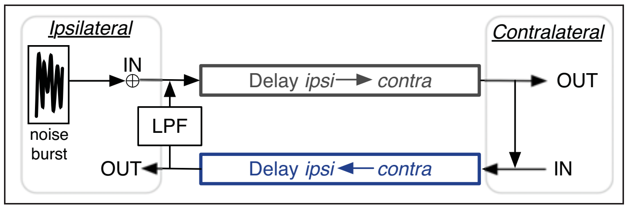


Figure 2. Karplus-Strong algorithm implemented in the network path recirculating audio.

To test a connection, a client connects against a known server IP number (e.g., CCRMA at Stanford). The path is sonified with this string model. As the network delay increases, the pitch of the sting will be lower. Variances in the latency will be perceived as vibrato of the string model. Packet losses are translated into impulsive types of sounds (for the case when the receive plays zeros when it doesn't receive a packet) or into wavetable type of sound (for the mode when the system keeps looping through the last receiving packet) [3] .

Providing this service for intuitive and quick evaluation of connection QoS is the original intended application of this technology. By connecting to the server and "listening" to the path, users can tune their connection to its optimal settings. As mentioned above, there's a trade-off between latency and sound quality. In the presence of jitter/vibrato, the local buffering has to be increased to avoid late packets, but at the same time we don't want to increase it too much (to avoid unnecessary latency). Doing this by trial and error requires experience and can be frustrating for new users. If, in turn, musicians can listen and tune the connection in the same way they tune an instrument, the setup is much faster and intuitive. Again the goal for the musician, is that they want to tune their pitch to be as high as possible (lower latency) with the smallest possible vibrato (jitter).

[3] More details on these two modes can be found in [3].

4.2 Star topology connection/mixing hub

Mixing and managing a remote connection when more than two sites are involved can be very complicated. Engineers have to deal with audio channels coming from different places (sometimes on confusingly different channels), all with different levels. They also need to make sure local audio is sent to the peer with proper gains. A solution to centrally manage these types of situations designates a master location which can mix and/or relay all the channels and send them back to the respective connected peers.

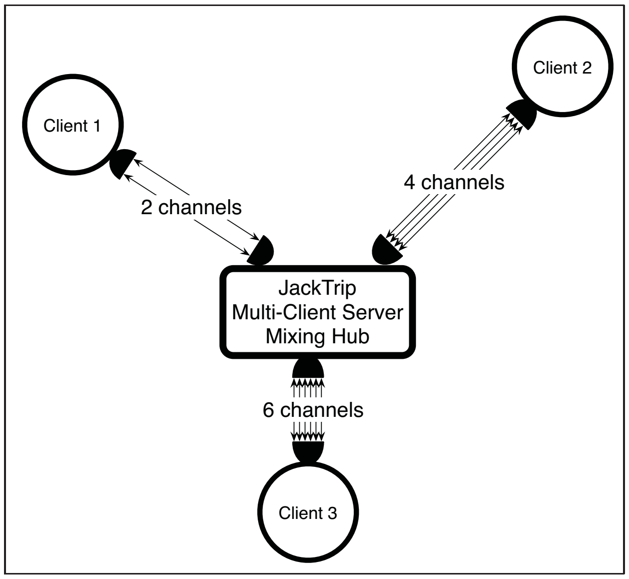


Figure 3. Multi-client server as a hub

The present sever implementation allows a server to dynamically connect and disconnect audio from different clients. Each client can have a different number of channels and different network tuning parameters. [4] In this case

[4] JackTrip presently uses Jack [2] as its audio host. This has the limitation that sampling rate and buffer are fixed at *Jack start-time* and cannot be tuned after the server has started.

the server will act as a "hub" between several locations.

Figure 3 illustrates this for an example with three clients. The server can mix and re-route all the audio channels between the clients, hence allowing a multi-site performance with one site acting as a master relay service and/or mixer.

5 CONCLUSIONS AND FUTURE WORK

The first decade of the 21st century evidenced a dramatic increase in the speed and reliability of high-speed networks. This increase is expected to continue. We have provided a system for musicians to tune and optimize their connections against a reference server in a way that lets them adapt to their given network situation. The server can also be used to interconnect multiple sites with arbitrary number of channels, and be a "mixing hub" that distributes audio to all the locations from a central place.

Scalability in network performance is a big issue that still needs to be solved. Learning how to connect hundreds or even thousands of remote locations for a global-jam session is a pending goal. Multicast at the network layer would provide a solution for a fully connected peer-to-peer mesh. Clients would select from a list of peers they want to connect with, and then send just one packet via multicast (using its underlying network layer implementation). Network routers and switches determine when a copy needs to be made. AccesGrid implements this [8] for a fixed number of audio channels, however this infrastructure is not yet ubiquitous. Furthermore, when the number of audio channels and other settings differ among the clients, a new and consistent solution is required so that they can inter-operate.

Scaling up and distributing physical models embedded in the network path can also serve to perform "global string network symphonies", where the global network becomes the instrument itself, an instrument distributed throughout the world.

6 ACKNOWLEDGMENTS

This work was carried out in cooperation with MusicianLink, Inc. and funded by National Science Foundation Award Grant No. IIP-0741278 with a sub-award to CCRMA. See the online Final Report, Technical Research Summary.[5] Fernando Lopez-Lezcano and Carr Wilkerson from CCRMA have provided continuous assistance in the implementation and server infrastructure setup.

7 REFERENCES

[1] (2008–2009) Qt Software. [Online]. Available: http://www.qtsoftware.com/

[2] (2009) JACK: Connecting a world of audio. [Online]. Available: http://jackaudio.org/

[3] Cáceres, J.-P. and Chafe, C., "JackTrip: Under the hood of an engine for network audio," in *Proceedings of International Computer Music Conference*, Montreal, 2009.

[4] Chafe, C. and Gurevich, M., "Network time delay and ensemble accuracy: Effects of latency, asymmetry," in *Proceedings of the AES 117th Convention*, San Francisco, 2004.

[5] Chafe, C. and Leistikow, R., "Levels of temporal resolution in sonification of network performance," in *Proceedings of the 2001 International Conference on Auditory Display*. Helsinki: ICAD, 2001.

[6] Chafe, C., Wilson, S., and Walling, D., "Physical model synthesis with application to internet acoustics," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Orlando, 2002.

[7] Comer, D. E., *Internetworking with TCP/IP, Vol 1*, 5th ed. Prentice Hall, Jul. 2005.

[8] Daw, M., "Advanced collaboration with the access grid," *Ariadne*, vol. 42, Jan. 2005. [Online]. Available: http://www.ariadne.ac.uk/issue42/daw/intro.html

[9] Karplus, K. and Strong, A., "Digital synthesis of Plucked-String and drum timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43–55, 1983.

[10] Peterson, L. L. and Davie, B. S., *Computer Networks: A Systems Approach, 3rd Edition*, 3rd ed. Morgan Kaufmann, May 2003.

[11] Qiao, Z., Venkatasubramanian, R., Sun, L., and Ifeachor, E., "A new buffer algorithm for speech quality improvement in VoIP systems," *Wireless Personal Communications*, vol. 45, no. 2, pp. 189–207, Apr. 2008.

[12] Renaud, A. B., Carôt, A., and Rebelo, P., "Networked music performance: State of the art," in *Proceedings of the AES 30th International Conference*, Saariselkä, Finland, 2007.

[13] Stevens, W. R., Fenner, B., and Rudoff, A. M., *Unix Network Programming, Volume 1: The Sockets Networking API (3rd Edition)*, 3rd ed. Addison-Wesley Professional, Nov. 2003.

[5] http://ccrma.stanford.edu/~cc/pub/pdf/qosServer-nsfFinalReport.pdf

PARALLELIZATION OF AUDIO APPLICATIONS WITH FAUST

Yann Orlarey
Grame
orlarey@grame.fr

Dominique Fober
Grame
fober@grame.fr

Stephane Letz
Grame
letz@grame.fr

ABSTRACT

Faust 0.9.9.6 introduces new compilation options to automatically parallelize audio applications. This paper explains how the automatic parallelization is done and presents some benchmarks.

1 INTRODUCTION

Faust is a programming language for real-time signal processing and synthesis designed from scratch to be a compiled language. Being efficiently compiled allows Faust to be complementary to existing audio languages and to provide a viable high-level alternative to C/C++ to develop high-performance signal processing applications, libraries or audio plug-ins.

Until recently the computation code generated by the compiler was organized quite traditionally as a single sample processing loop. This scheme works very well but it doesn't take advantages from multicore architectures. Moreover it can generate code that exceeds the autovectorization capabilities of current C++ compilers.

We have recently extended the compiler with two new schemes : the *vector* and the *parallel* schemes. The *vector* scheme simplifies the autovectorization work of the C++ compiler by splitting the sample processing loop into several simpler loops. The *parallel* scheme analyzes the dependencies between these loops and adds OpenMP pragmas to indicate those that can be computed in parallel.

These new schemes can produce interesting performance improvements. The goal of this paper is to present these new compilation schemes and to provide some benchmarks comparing their performances. The paper is organized as follows : next section will give a brief overview of Faust language, The third section will present the three code generation schemes and the last section will introduce the benchmarks used and the results obtained.

SMC 2009, July 23-25, Porto, Portugal

2 FAUST OVERVIEW

In this section we give a brief overview of Faust with some examples of code.

A Faust program describes a *signal processor*, something that transforms some input signals and produces some output signals. The programming model used combines a *functional programming approach* with a *block-diagram syntax*. The functional programming approach provides a natural framework for signal processing. Digital signals are modeled as discrete functions of time, and signal processors as second order functions that operate on them. Moreover Faust's block-diagram *composition operators*, used to combine signal processors together, fit in the same picture as third order functions.

The Faust compiler translates Faust programs into equivalent C++ programs. It uses several optimization techniques in order to generate the most efficient code. The resulting code can usually compete with, and sometimes outperform, DSP code directly written in C/C++. It is also self-contained and doesn't depend on any DSP runtime library.

Thanks to specific *architecture files*, a single Faust program can be used to produce code for a variety of platforms and plug-in formats. These architecture files act as wrappers and describe the interactions with the host audio and GUI system. Currently more than 10 architectures are supported (see Table 1) and new ones can be easily added.

alsa-gtk.cpp	ALSA application + GTK
alsa-qt.cpp	ALSA application + QT4
jack-gtk.cpp	JACK application + GTK
jack-qt.cpp	JACK application + QT4
ca-qt.cpp	CoreAudio application + QT4
ladspa.cpp	LADSPA plug-in
max-msp.cpp	Max MSP plug-in
supercollider.cpp	Supercollider plug-in
vst.cpp	VST plug-in
q.cpp	Q language plug-in

Table 1. Some architecture files available for Faust

In the following subsections we are giving a short and informal introduction to the language through the example of a simple noise generator. Interested readers can refer to [1] for a more complete description.

2.1 A simple noise generator

A Faust program describes a signal processor by combining primitive operations on signals (like $+, -, *, /, \sqrt{}, \sin, \cos, \ldots$) using an algebra of high level *composition operators* [2] (see Table 2). You can think of these composition operators as a generalization of mathematical function composition $f \circ g$.

$f \sim g$	recursive composition
$f \;,\; g$	parallel composition
$f : g$	sequential composition
$f <: g$	split composition
$f :> g$	merge composition

Table 2. The five high level block-diagram *composition operators* used in Faust

A Faust program is organized as a set of *definitions* with at least one for the keyword `process` (the equivalent of `main` in C).

Our noise generator example `noise.dsp` only involves three very simple definitions. But it also shows some specific aspects of the language:

```
random  = +(12345) ~ *(1103515245);
noise   = random/2147483647.0;
process = noise * vslider("noise", 0, 0,
                          100, 0.1)/100;
```

The first definition describes a (pseudo) random number generator. Each new random number is computed by multiplying the previous one by `1103515245` and by adding to the result `12345`.

The expression `+(12345)` denotes the operation of adding `12345` to a signal. It is an example of a common technique in functional programming called *partial application*: the binary operation `+` is here provided with only one of its arguments. In the same way `*(1103515245)` denotes the multiplication of a signal by `1103515245`.

The two resulting operations are *recursively composed* using the $\sim$ operator. This operator connects in a feedback loop the output of `+(12345)` to the input of `*(1103515245)` (with an implicit 1-sample delay) and the output of `*(1103515245)` to the input of `+(12345)`.

The second definition transforms the random signal into a noise signal by scaling it between -1.0 and +1.0.

Finally, the definition of process adds a simple user interface to control the production of the sound. The noise signal is multiplied by the value delivered by a slider to control its volume.

2.2 Invoking the compiler

The role of the compiler is to translate Faust programs into equivalent C++ programs. The key idea to generate efficient code is not to compile the block diagram itself, but *what it computes*.

Driven by the semantic rules of the language the compiler starts by propagating symbolic signals into the block diagram, in order to discover how each output signal can be expressed as a function of the input signals.

These resulting signal expressions are then simplified and normalized, and common subexpressions are factorized. Finally these expressions are translated into a self contained C++ class that implements all the required computation.

To compile our noise generator example we use the following command :

```
 $ faust noise.dsp
```

This command generates the following C++ code on the standard output :

```
class mydsp : public dsp {
 private:
  int   iRec0[2];
  float  fslider0;
 public:
  static void metadata(Meta* m)  {
  }

  virtual int getNumInputs() { return 0; }
  virtual int getNumOutputs() { return 1; }
  static void classInit(int samplingFreq) {
  }
  virtual void instanceInit(int samplingFreq)
  {
    fSamplingFreq = samplingFreq;
    for (int i=0; i<2; i++) iRec0[i] = 0;
    fslider0 = 0.0f;
  }
  virtual void init(int samplingFreq)
  {
    classInit(samplingFreq);
    instanceInit(samplingFreq);
  }
  virtual void buildUserInterface(UI* interface)
  {
    interface->openVerticalBox("noise");
    interface->declare(&fslider0, "style"
                                 , "knob");
    interface->addVerticalSlider("noise",
          &fslider0, 0.0f, 0.0f, 100.0f, 0.1f);
    interface->closeBox();
  }
  virtual void compute (int count,
                        float** input,
                        float** output)
  {
    float  fSlow0 = (4.656613e-12f * fslider0);
    float* output0 = output[0];
    for (int i=0; i<count; i++) {
      iRec0[0] = 12345+1103515245*iRec0[1];
      output0[i] = fSlow0*iRec0[0];
      // post processing
      iRec0[1] = iRec0[0];
    }
  }
};
```

The generated class contains seven methods. Among these methods `getNumInputs()` and `getNumOutputs()` return the number of input and output signals required by our signal processor. `init()` initializes the internal state of the signal processor. `buildUserInterface()` can be seen as a list of high level commands, independent of any toolkit, to build the user interface. The method `compute()` does the actual signal processing. It takes 3 arguments: the number of frames to compute, the addresses of the input buffers and the addresses of the output buffers, and computes the output samples according to the input samples.

2.3 Generating a full application

The `faust` command accepts several options to control the generated code. Two of them are widely used. The option `-o` *outputfile* specifies the output file to be used instead of the standard output. The option `-a` *architecturefile* defines the architecture file used to wrap the generate C++ class.

For example the command `faust -a jack-qt.cpp -o noise.cpp noise.dsp` generates a full jack application using QT4.4 as a graphic toolkit. The figure 1 is a screenshot of our noise application running.

Figure 1. Screenshot of the noise example generated with the `jack-qt.cpp` architecture

2.4 Generating a block-diagram

Another interesting option is `-svg` that generates one or more SVG graphic files that represent the block-diagram of the program as in Figure 2.

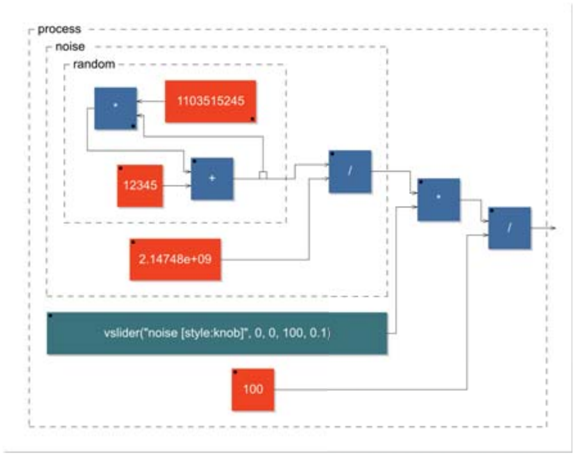


Figure 2. Graphic block-diagram of the noise generator produced with the -svg option

It is interesting to note the difference between the block diagram and the generated C++ code. The block diagram involves one addition, two multiplications and two divisions. The generated C++ program only involves one addition and two multiplications per sample. The compiler has been able to optimize the code by factorizing and reorganizing the operations.

As already said, the key idea here is not to compile the block diagram itself, but *what it computes*.

3 CODE GENERATION

In this section we describe how the Faust compiler generates its code. We will first introduce the so called *scalar* generation of code which was the only one until version 0.9.9.5. Then, we will present the *vector* generation of code where the code is organized into several loops that operates on vectors, and finally the *parallel* generation of code where these vector loops are parallelized using OpenMP directives.

3.1 Preliminary steps

Before reaching the stage of the C++ code generation, the Faust compiler have to carry on several steps that we describe briefly here.

3.1.1 Parsing source files

The first one is to recursively parse all the source files involved. Each source file contains a set of definitions and possibly some *import* directives for other source files. The result of this phase is a list of definitions: $[(name_1 = definition_1), (name_2 = definition_2), \ldots]$. This list is actually a set, as redefinitions of symbols are not allowed.

3.1.2 Evaluating block-diagrams

Among the names defined there must be *process*, the analog of main in C/C++. This definition has to be evaluated as Faust allows algorithmic block-diagram definitions.

For example the algorithmic definition:

```
foo(n) = *(10+n);
process = par(i,3, foo(i));
```

Listing 1. example of algorithmic definition

will be translated in a *flat* block-diagram description that contains only primitive blocks:

```
process = (_,10:*),(_,11:*),(_,12:*);
```

This description is said to be in *normal form*.

3.1.3 Discovering the mathematical equations

Faust doesn't compile a block-diagram directly. It uses a phase of symbolic propagation to first discover its mathematical semantic (what it computes). The principle is to propagate symbolic signals through the inputs of the block-diagram in order to get, at the other end, the mathematical equation of each output signal.

These equations are then normalized so that different block-diagrams, but computing mathematically equivalent signals, result in the same output equations.

Here is a very simple example where the input signal is divided by 2 and then delayed by 10 samples:

```
process =  /(2) : @(10);
```

This is equivalent to having the input signal first multiplied by 2, then delayed by 7 samples, then divided by 4 and then delayed by 3 samples.

```
process = *(2) : @(7) : /(4):  @(3);
```

Both lead to the following signal equation:

$$Y(t) = 0.5 * X(t - 10)$$

Faust applies several rules to simplify and normalize output signal equations. For example one of theses rules says that it is better to multiply a signal by a constant after a delay than before. It gives the compiler more opportunities to share and reuse the same delay line. Another rule says that two consecutive delays can be combined into a single one.

3.1.4 Typing the mathematical equations

The next phase is to assign *types* to the resulting signal equations. This will not only help the compiler to detect errors but also to generate the most efficient code. Several aspects are considered:

1. the *nature* of the signal: *integer* of *float*.
2. *interval of values* of the signal: the minimum and maximum values that a signal can take
3. the *computation time* of the signal: the signal can be computed at *compilation time*, at *initialization time* or at *execution time*.
4. the *speed* of the signal: *constant* signals are computed only once, *low speed* user interface signals are computed once for every block of samples, *high speed* audio signals are computed every samples.
5. parallelism of the signal: *true* if the samples of the signal can be computed in parallel, *false* when the signal has recursive dependencies requiring its samples to be computed sequentially.

3.1.5 Occurrence analysis

The role of this last preparation phase is to analyze in which context each subexpression is used and to discover common subexpressions. If an expensive common subexpression is discovered, an assignment to a *cache variable* `float fTemp = <common subexpression code>;` is generated, and the cache variable `fTemp` is used in its enclosing expressions. Otherwise the subexpression code is used inlined.

The occurrence analysis proceeds by a top-down visit of the signal expression. The first time a subexpression is visited, it is annotated with a counter. Next time, the counter will be increased and its visit skipped.

Subexpressions with several occurrences are candidates to be cached in variables. However in some circumstances expressions with a single occurrence also need to be cached if they occur in a faster context. For example, a constant expression occurring in a low speed user interface expression or a user interface expression occurring in a high speed audio expression will generally require to be cached.

Only after this phase can the generation of the C++ code start.

3.2 Scalar Code generation

The generation of the C++ code is made by populating a *klass* object (representing a C++ class), with strings representing C++ declarations and lines of code. In scalar mode these lines of code are organized in a single sample computation loop, while they can be splitted in several loops with the new *vector* and *parallel* schemes.

The code generation basically relies on two functions: a translation function $[\![\ ]\!]$ that translates a signal expression into a string of C++ code, and a cache function $\mathbf{C}()$ that checks if a variable is needed.

We don't have enough room to go in too much details but here is the translation rule for the addition of two signal expressions:

$$\frac{\begin{array}{c}[\![E_1]\!] \rightarrow \mathtt{S}_1 \\ [\![E_2]\!] \rightarrow \mathtt{S}_2\end{array}}{[\![E_1 + E_2]\!] \rightarrow \mathbf{C}(''(\mathtt{S}_1 + \mathtt{S}_2)'')}$$

It says that to compile the addition of two signals we compile each of these signals and concat the resulting strings with a + sign in between. The string obtained is passed to the cache function that will check if the expression is shared or not.

Let's say that the string passed to the cache function $\mathbf{C}()$ is `(input0[i] + input1[i])`. If the expression is shared, the cache function will allocate a fresh variable name `fTemp0`, add the line of code `float fTemp0 = (input0[i] + input1[i]);` to the *klass* object and return `fTemp0` as a string to be used when compiling enclosing expressions. If the expression is not shared it will simply return the string `(input0[i] + input1[i])` unmodified.

To illustrate this, let's take two simple examples. The first one converts a stereo signal into a mono signal by adding the two input signals:

```
process = +;
```

In this case `(input0[i] + input1[i])` is not shared and the generated C++ code is the following:

```
virtual void compute (int count,
                      float** input,
                      float** output)
{
  float* input0 = input[0];
  float* input1 = input[1];
  float* output0 = output[0];
  for (int i=0; i<count; i++) {
    output0[i] = (input0[i] + input1[i]);
  }
}
```

But when the sum of the two input signals is duplicated on two output signals as in:

```
process = + <: _,_;
```

then `(input0[i] + input1[i])` will be cached in a temporary variable:

```
virtual void compute (int count,
                      float** input,
                      float** output)
{
  float* input0 = input[0];
  float* input1 = input[1];
  float* output0 = output[0];
  float* output1 = output[1];
  for (int i=0; i<count; i++) {
    float fTemp0 = (input0[i] + input1[i]);
    output0[i] = fTemp0;
    output1[i] = fTemp0;
  }
}
```

3.3 Vector Code generation

Modern C++ compilers are able to do autovectorization, that is to use SIMD instructions to speedup the code. These instructions can typically operate in parallel on short vectors of 4 simple precision floating point numbers thus leading to a theoretical speedup of x4. Autovectorization of C/C+ programs is a difficult task. Current compilers are very sensitive to the way the code is arranged. In particular too complex loops can prevent autovectorization. The goal of the new vector code generation is to rearrange the C++ code in a way that facilitates the autovectorization job of the C++ compiler. Instead of generating a single sample computation loop, it splits the computation into several simpler loops that communicates by vectors.

The vector code generation is activated by passing the `--vectorize` (or `-vec`) option to the Faust compiler. Two additional options are available: `--vec-size <n>` controls the size of the vector (by default 32 samples) and `--loop-variant 0/1` gives some additional control on the loops.

To illustrate the difference between scalar code and vector code, let's take the computation of the RMS (Root Mean Square) value of a signal. Here is the Faust code that computes the Root Mean Square of a sliding window of 1000 samples:

```
// Root Mean Square of n consecutive samples
RMS(n) = square : mean(n) : sqrt ;
```

```
// Square of a signal
square(x) = x * x ;

// Mean of n consecutive samples of a signal
// (uses fixpoint to avoid the accumulation of
// rounding errors)
mean(n) = float2fix : integrate(n) :
          fix2float : /(n);

// Sliding sum of n consecutive samples
integrate(n,x) = x - x@n : +~_ ;

// Convertion between float and fix point
float2fix(x) = int(x*(1<<20));
fix2float(x) = float(x)/(1<<20);

// Root Mean Square of 1000 consecutive samples
process = RMS(1000) ;
```

The compute() method generated in scalar mode is the following:

```
virtual void compute (int count,
                      float** input,
                      float** output)
{
  float* input0 = input[0];
  float* output0 = output[0];
  for (int i=0; i<count; i++) {
    float fTemp0 = input0[i];
    int iTemp1 = int(1048576*fTemp0*fTemp0);
    iVec0[IOTA&1023] = iTemp1;
    iRec0[0] = ((iVec0[IOTA&1023] + iRec0[1])
                - iVec0[(IOTA-1000)&1023]);
    output0[i] = sqrtf(9.536744e-10f *
                       float(iRec0[0]));
    // post processing
    iRec0[1] = iRec0[0];
    IOTA = IOTA+1;
  }
}
```

The `-vec` option leads to the following reorganization of the code:

```
virtual void compute (int fullcount,
                      float** input,
                      float** output)
{
  int     iRec0_tmp[32+4];
  int*    iRec0 = &iRec0_tmp[4];
  for (int index=0; index<fullcount; index+=32)
  {
    int count = min (32, fullcount-index);
    float* input0 = &input[0][index];
    float* output0 = &output[0][index];
    for (int i=0; i<4; i++)
      iRec0_tmp[i]=iRec0_perm[i];
    // SECTION : 1
    for (int i=0; i<count; i++) {
      iYec0[(iYec0_idx+i)&2047] =
               int(1048576*input0[i]*input0[i]);
    }
    // SECTION : 2
    for (int i=0; i<count; i++) {
      iRec0[i] = ((iYec0[i] + iRec0[i-1]) -
               iYec0[(iYec0_idx+i-1000)&2047]);
    }
    // SECTION : 3
    for (int i=0; i<count; i++) {
      output0[i] = sqrtf((9.536744e-10f *
                   float(iRec0[i])));
    }
    // SECTION : 4
    iYec0_idx = (iYec0_idx+count)&2047;
    for (int i=0; i<4; i++)
      iRec0_perm[i]=iRec0_tmp[count+i];
  }
}
```

While the second version of the code is more complex, it turns out to be much easier to vectorize efficiently by the C++ compiler. Using Intel icc 11.0, with the exact same compilation options: `-O3 -xHost -ftz -fno-alias -fp-model fast=2`, the scalar version leads to a throughput performance of 129.144 MB/s, while the vector version achieves 359.548 MB/s, a speedup of x2.8 !

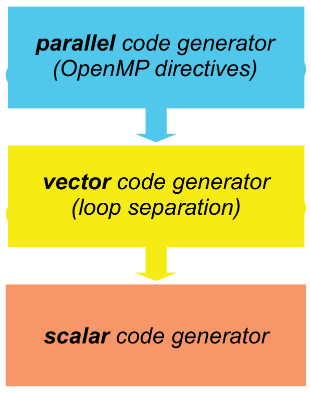


Figure 3. Faust's stack of code generators

The vector code generation is built on top of the scalar code generation (see figure 3). Every time an expression needs to be compiled, the compiler checks to see if it needs to be in a separate loop or not. It applies some simple rules for that. Expressions that are shared (and are complex enough) are good candidates to be compiled in a separate loop, as well as recursive expressions and expressions used in delay lines.

The result is a directed graph in which each node is a computation loop (see Figure 4). This graph is stored in the klass object and a topological sort is applied to it before printing the code.

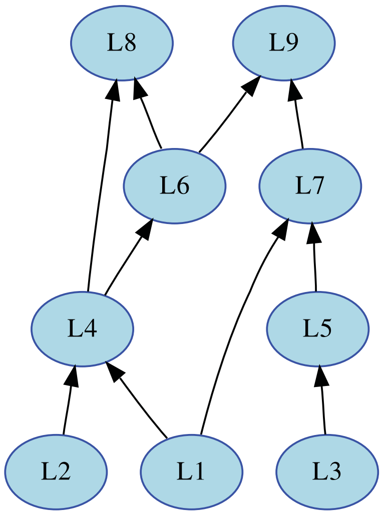


Figure 4. The result of the -vec option is a directed acyclic graph (DAG) of small computation loops

3.4 Parallel Code generation

The parallel code generation is activated by passing the `--openMP` (or `-omp`) option to the Faust compiler. It implies the `-vec` options as the parallel code generation is built on top of the vector code generation by inserting appropriate OpenMP directives in the C++ code.

3.4.1 The OpenMP API

OpenMP (http://wwww.openmp.org) is a well established API that is used to explicitly define direct multi-threaded, shared memory parallelism. It is based on a fork-join model of parallelism (see figure 5). Parallel regions are delimited by using the `#pragma omp parallel` construct. At the entrance of a parallel region a team of parallel threads is activated. The code within a parallel region is executed by each thread of the parallel team until the end of the region.

```
#pragma omp parallel
{
  // the code here is executed simultaneously by
  // every thread of the parallel team
  ...
}
```

In order not to have every thread doing redundantly the exact same work, OpemMP provides specific *work-sharing* directives. For example `#pragma omp sections` allows to break the work into separate, discrete sections. Each section being executed by one thread:

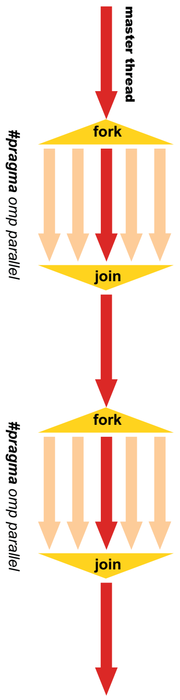


Figure 5. OpenMP is based on a fork-join model

```
#pragma omp parallel
{
  #pragma omp sections
  {
    #pragma omp section
    {
      // job 1
    }
    #pragma omp section
    {
      // job 2
    }
    ...
  }

  ...
}
```

3.4.2 Adding OpenMP directives

As said before the parallel code generation is built on top of the vector code generation. The graph of loops produced by the vector code generator is topologically sorted in order to detect the loops that can be computed in parallel. The first set S_0 (loops $L1$, $L2$ and $L3$ in the DAG of Figure 4) contains the loops that don't depend on any other loops, the set S_1 contains the loops that only depend on loops of S_0, (that is loops $L4$ and $L5$), etc..

As all the loops of a given set S_n can be computed in parallel, the compiler will generate a `sections` construct with a `section` for each loop.

```
 #pragma omp sections
 {
   #pragma omp section
   for (...) {
     // Loop 1
   }
   #pragma omp section
   for (...) {
     // Loop 2
   }
   ...
 }
```

If a given set constains only one loop, then the compiler checks to see if the loop can be parallelized (no recursive dependencies) or not. If it can be parallelized, it generates:

```
 #pragma omp for
 for (...) {
  // Loop code
 }
```

otherwise it generates a `single` construct so that only one thread will execute the loop:

```
 #pragma omp single
 for (...) {
  // Loop code
 }
```

3.4.3 *Example of parallel code*

To illustrate how Faust uses the OpenMP directives, here is a very simple example, two 1-pole filters in parallel connected to an adder (see figure 6 the corresponding block-diagram):

```
filter(c) = *(1-c) : + ~ *(c);
process = filter(0.9), filter(0.9) : +;
```

The corresponding compute() method obtained using the -omp option is the following:

```
virtual void compute (int fullcount,
                      float** input,
                      float** output)
{
  float   fRec0_tmp[32+4];
  float   fRec1_tmp[32+4];
  float*  fRec0 = &fRec0_tmp[4];
  float*  fRec1 = &fRec1_tmp[4];
  #pragma omp parallel firstprivate(fRec0,fRec1)
  {
    for (int index = 0; index < fullcount;
                                    index += 32)
    {
      int count = min (32, fullcount-index);
      float* input0 = &input[0][index];
      float* input1 = &input[1][index];
      float* output0 = &output[0][index];
      #pragma omp single
      {
        for (int i=0; i<4; i++)
          fRec0_tmp[i]=fRec0_perm[i];
        for (int i=0; i<4; i++)
          fRec1_tmp[i]=fRec1_perm[i];
      }
      // SECTION : 1
      #pragma omp sections
      {
        #pragma omp section
        for (int i=0; i<count; i++) {
          fRec0[i] = ((0.1f * input1[i])
                   + (0.9f * fRec0[i-1]));
        }
        #pragma omp section
        for (int i=0; i<count; i++) {
          fRec1[i] = ((0.1f * input0[i])
                   + (0.9f * fRec1[i-1]));
        }
      }
      // SECTION : 2
      #pragma omp for
      for (int i=0; i<count; i++) {
        output0[i] = (fRec1[i] + fRec0[i]);
      }
      // SECTION : 3
      #pragma omp single
      {
        for (int i=0; i<4; i++)
          fRec0_perm[i]=fRec0_tmp[count+i];
        for (int i=0; i<4; i++)
          fRec1_perm[i]=fRec1_tmp[count+i];
      }
    }
  }
}
```

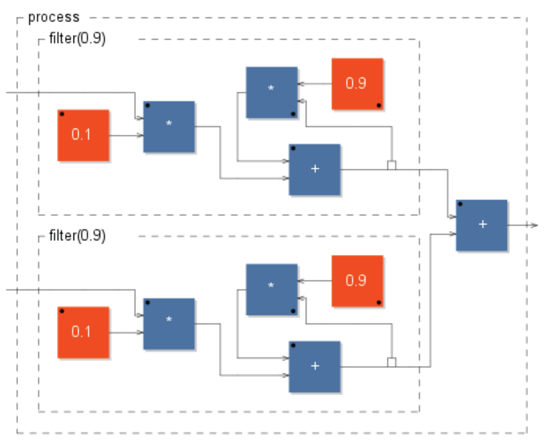


Figure 6. two filters in parallel connected to an adder

This code appeals for some comments:

1. The parallel construct `#pragma omp parallel` is the fundamental construct that starts parallel execution. The number of parallel threads is generally the number of CPU cores but it can be controlled in several ways.

2. Variables external to the parallel region are shared by default. The `firstprivate(fRec0,fRec1)` clause indicates that each thread should have its private copy of fRec0 and fRec1. The reason is that accessing shared variables requires an indirection and is quite inefficient compared to private copies.

3. The top level loop `for (int index = 0;...)...` is executed by all threads simultaneously. The subsequent work-sharing directives inside the loop will indicate how the work must be shared between the threads.

4. Please note that an implied barrier exists at the end of each work-sharing region. All threads must have executed the barrier before any of them can continue.

5. The work-sharing directive `#pragma omp single` indicates that this first section will be executed by only one thread (any of them).

6. The work-sharing directive `#pragma omp sections` indicates that each corresponding `#pragma omp section`, here our two filters, will be executed in parallel.

7. The loop construct `#pragma omp for` specifies that the iterations of the associated loop will be executed in parallel. The iterations of the loop are distributed across the parallel threads. For example, if we have two threads, the first one can compute indices between 0 and count/2 and the other between count/2 and count.

8. Finally `#pragma omp single` in section 3 indicates that this last section will be executed by only one thread (any of them).

4 BENCHMARKS

To compare the performances of these three types of code generation in a realistic situation we have implemented a special *alsa-gtk-bench.cpp* architecture file that measures the duration of the compute() method. Here is a fragment of this architecture file:

```
while(running) {
  audio.read();
  STARTMESURE
  DSP.compute(audio.buffering(),
              audio.inputSoftChannels(),
              audio.outputSoftChannels()
             );
  STOPMESURE
  audio.write();
  running = mesure <= (KMESURE + KSKIP);
}
```

The methodology is the following. The duration of the compute method is measured by reading the TSC (Time Stamp Counter) register. A total of 128+2048 measures are made by run. The first 128 measures are considered a warm-up period and are skipped. The median value of the following 2048 measures is computed. This median value, expressed in processors cycles, is first converted into a duration, and then into number of bytes produced per second considering the audio buffer size (in our test 2048) and the number of output channels.

This *throughput performance* is a good indicator. The memory bandwidth is a strong limiting factor for today's processors, and it has to be shared among the processors. In other words, on a SMP machine a realtime audio program can never go faster than the memory bandwidth. And if a sequential program already uses all the available memory bandwidth, there is no room for improvement. In this case a parallel version can only perform worth.

4.1 Machines and compilers used

In order to compare the scalar code generation with the new vector and parallel code generation, we have compiled with Faust 0.9.9.5b2 a series of tests in three different versions. The following commands were used :

- **scal** : `faust -a alsa-gtk-bench.cpp test.dsp -o test.cpp`
- **vec** : `faust -a alsa-gtk-bench.cpp -vec -vs 3968 test.dsp -o test.cpp`
- **par** : `faust -a alsa-gtk-bench.cpp -omp -vs 3968 test.dsp -o test.cpp`

We have also used two different C++ compilers, GNU GCC and Intel ICC :

- GCC version 4.3.2 with options : `-O3 -march=native -mfpmath=sse -msse -msse2 -msse3 -ffast-math -ftree-vectorize`. (`-fopenmp` added for OpenMP).

- ICC version 11.0.074 with options : `-O3 -xHost -ftz -fno-alias -fp-model fast=2`. (`-openmp` is added for OpenMP).

All the tests were run on three different machines :

- **vaio** : a Sony Vaio SZ3VP laptop, with an Intel T7400 dual core processor at 2167 MHz, 2GB of Ram, running an Ubuntu 7.10 distribution with a 2.6.22-15-generic kernel.
- **xps** : a Dell XPS machine with an Intel Q9300 quad core processor at 2500 MHz, 4GB of Ram, running an Ubuntu 8.10 distribution with a 2.6.22-15-generic kernel.
- **macpro** : an Apple Macpro with two Intel Xeon X5365 quad core processors at 3000 MHz, 2GB of Ram, running an Ubuntu 8.10 distribution with a 2.6.27-12-generic kernel

4.2 Benchmark: copy1.dsp

The goal of this first test is to measure the memory bandwidth. We use a very simple Faust program copy1.dsp that simply copies the input signal to the output signal:

```
process = _;
```

The results we have obtained are summarized figure 7. The horizontal axe corresponds to the three faust compilation schemes : *scalar* , *vector* and *parallel*, combined with the two C++ compilers : *gcc* and *icc*. The vertical axe is the throughput : how many bytes of samples each tested program is able to produce per second (higher values are the better).

It is interesting to note how catastrophic the performances of the parallel versions are. The scalar and vector versions are quite similar with a little advantage to the scalar version. The code generated by icc performs better. The memory bandwidth of the Macpro is disappointing especially considering that it has to be shared by 8 cores.

How stable are these measures ? Figure 8 compares the performances of copy1 (compiled with icc) on the Macpro on 5 different runs. As we can see the stability is reasonably good.

4.3 Benchmark: freeverb.dsp

The second test is freeverb.dsp, a Faust implementation of the Freeverb (the source can be found in the Faust distribution).

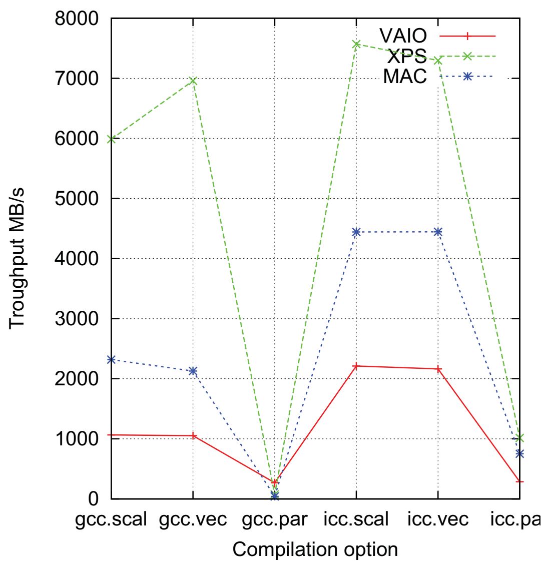


Figure 7. Copy1.dsp benchmark

The results are given figure 9. Here gcc gives very good results in scalar code and outperforms icc in 2 of the 3 cases. But the performances of gcc are still very poor on vector and parallel code.

Despite the fact that freeverb has a limited amount of parallelism, icc gives quite convincing results with a reasonable speedup on vector and parallel code on the Vaio and the XPS machines. It is also interesting to note that on parallel version the 8 3GHz cores of the macpro were slower than 4 2.5Ghz cores of the XPS !

4.4 Benchmark: karplus32.dsp

Karplus32.dsp is a generalized version of Karplus-Strong algorithm with 32 slightly detuned strings in parallel (the source can be found in the Faust distribution). Figure 10 gives the results. Again we can see excellent performances of gcc in scalar mode, good progression of the performances in vector mode as well as in parallel mode for icc.

4.5 Benchmark: mixer.dsp

This is the implementation of a simple 8 channels mixer. Each channel has a mute button, a volume control in dB, a vumeter and a stereo pan control. The mixer has also a volume control of the stereo output.

```
import("music.lib");

smooth(c) = *(1-c) : +~*(c);
```

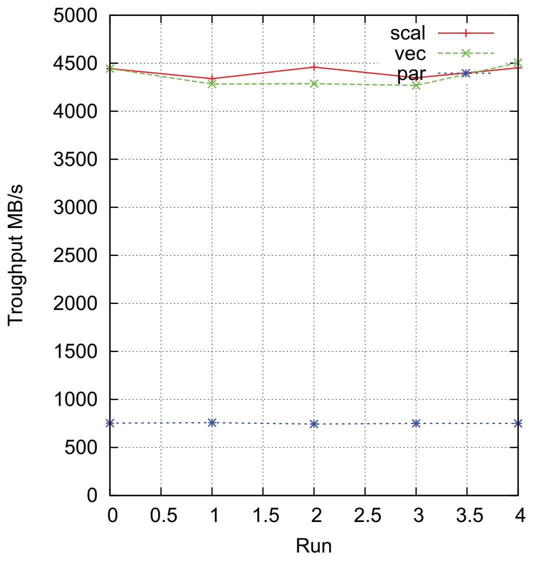


Figure 8. Stability of measures (copy1 on macpro, icc version)

```
vol    = *( vslider("fader", 0, -60, 4, 0.1)
           : db2linear : smooth(0.99) );

mute   = *(1 - checkbox("mute"));

vumeter(x)  = attach(x, env(x) :
                     vbargraph("",0,1))
  with {
        env = abs:min(0.99):max ~ -(1.0/SR);
       };

pan      = _ <: *(sqrt(1-c)), *(sqrt(c))
  with {
        c = ( nentry("pan",0,-8,8,1)-8)/-16 :
              smooth(0.99 );
       };

voice(v)   = vgroup("voice␣%v",
              mute :
              hgroup("", vol : vumeter) :
              pan );

stereo     = hgroup("stereo␣out", vol, vol);

process    = hgroup("mixer",
              par(i,8,voice(i)) :> stereo);
```

The results of figure 11 show a real benefit for the vectorized version with a speedup exceeding x2 on the 3 machines. There is also a positive impact of the parallelization even if more limited. As usual gcc delivers good scalar code but poor results on vectorized and OpenMP code.

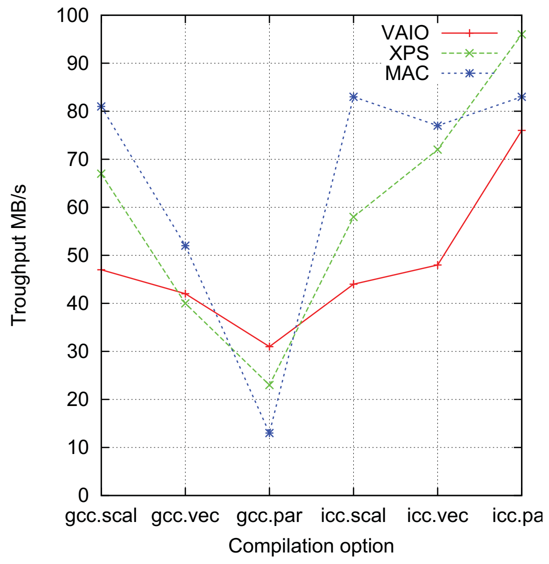


Figure 9. Freeverb.dsp benchmark

4.6 Benchmark: fdelay8.dsp

This test implements an 8-channels fractional delay. Each channel has a volume control in dB as well as a delay control in fractions of samples. The interpolation is based on a fifth-order Lagrange interpolation from Julius Smith's Faust filter library.

```
import("filter.lib");

line(i) =  vgroup("line␣%i",fdelay5(128,d):*(g))
  with{ g = vslider("gain␣(dB)",-60,-60,4,0.1)
          : db2linear : smooth(0.995);
        d = nentry("delay␣(samp)",10,10,128,0.1)
          : smooth(0.995);
      };

process = hgroup("", par(i, 8, line(i)) );
```

The results are presented figure 12. The Macpro exhibits a good speedup of x2.5 for its parallel version. The parallel speedup for the XPS machine is more limited and there is no speedup at all on the Vaio.

4.7 Benchmark: rms.dsp

The Faust source of rms.dsp was presented section 3.3. It is a purely sequential algorithm therefore the performances of the parallel versions are very bad. But, as figure 13 indicates, the vectorization gives a real boost to the performances, particularly on the vaio.

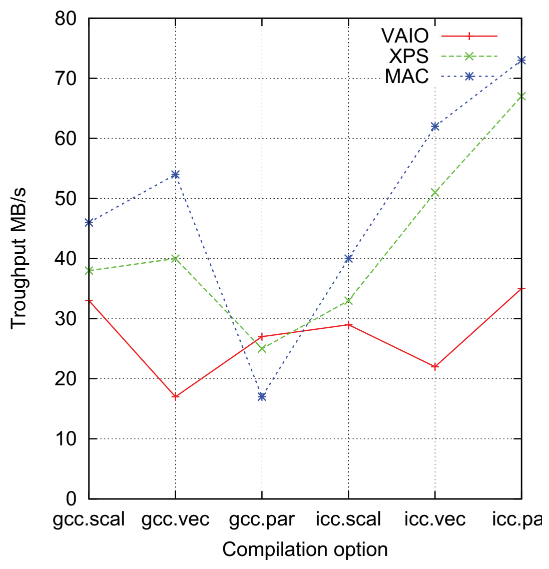


Figure 10. Karplus32.dsp benchmark

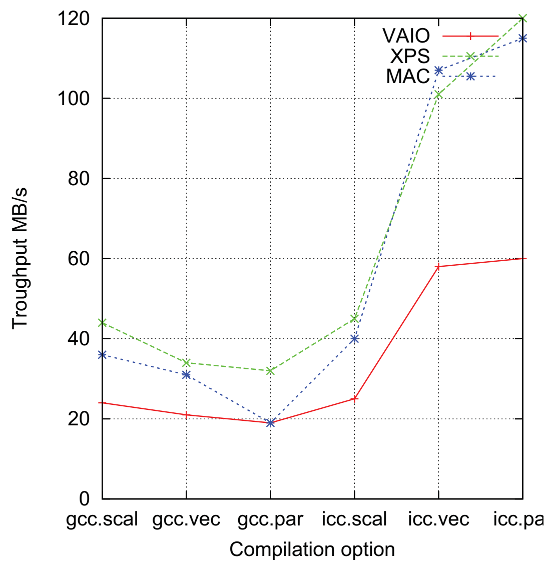


Figure 11. mixer.dsp benchmark

4.8 Benchmark: rms8.dsp

This test computes the RMS value on 8 channels in parallel. The Faust code is :

```
process = par(i,8,component("rms.dsp")) ;
```

We obviously have a good amount of parallelism here that icc is able to exploit as indicated by the results figure 14. Compared to the scalar performances, the parallel version exhibits a speedup of nearly x3 on the Mac, while the speedup for the XPS exceed x2.5. But the record is for the Vaio with a speedup of x2.2 !

5 CONCLUSION

We have presented two new compilation schemes recently introduced in the Faust compiler. The *vector* scheme simplifies the autovectorization work of the C++ compiler by splitting the sample processing loop into several simpler loops. The *parallel* scheme analyzes the dependencies between these loops and add OpenMP pragmas to indicate those that can be computed in parallel.

Figure 15 shows the speedup obtained with the vectorized code. With a good autovectorizing C++ compiler like Intel icc 11.0 we can obtain very significant improvements in many cases. On the contrary gcc 4.3.2 was not able to generate SIMD instructions, leading to a degradation of the performances. We therefore highly recommend icc to compile vectorized code, that is a pity considering the excellent results of gcc on scalar code.

Following the so called Amdahl's law, the speedup obtained with the parallelized code is highly dependent on the quantity of parallelism available (see figure 16. On purely parallel programs like fdelay8 and rms8 a speedup exceeding x2.5 was observed on the mac. This is a little bit disappointing for a 8-cores machine, but in phase with its relatively limited memory bandwidth. Here too, we recommend icc to compile OpenMP applications.

All these results are dependent on many choices and settings, in particular on compiler's options. The options we have retained were the best we could find, but the parameters space is huge and we have only explored a small part of it. It may be the case that the gcc results could be improved by changing the settings. This would be good news and the authors are interested by any suggestions on that point.

There is also a lot of possible improvements in the code generated by Faust. While it is easy to discover the whole potential parallelism of a Faust program [1], generating efficient OpenMP programs is much more difficult due to the overheads introduced and the additional pressure on the shared memory.

The trade-off between parallelism and overhead + memory pressure is something that we will have to improve in future versions. The fixed scheduling of the parallel tasks is also probably far from optimal in many cases. It will be also

[1] parallel programming is probably the chance of functional programming languages compared to imperative languages

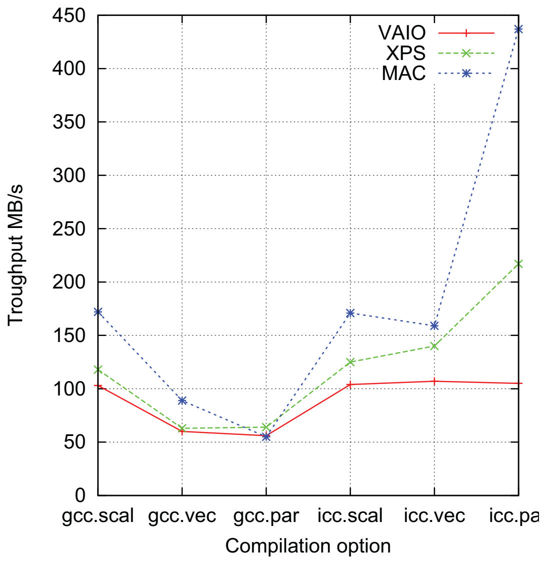


Figure 12. fdelay8.dsp benchmark

interesting to explore the possibilities of GPGPU and their high-level programming languages as an alternative to C++ and OpenMP.

Resources

1. http://openmp.org/
2. http://faust.grame.fr
3. http://www.intel.com/cd/software/products/asmona/eng/277618.htm

6 REFERENCES

[1] Yann Orlarey, Dominique Fober, and Stephane Letz. Syntactical and semantical aspects of faust. *Soft Computing*, 8(9):623–632, 2004.

[2] Y. Orlarey, D. Fober, and S. Letz. An algebra for block diagram languages. In ICMA, editor, *Proceedings of International Computer Music Conference*, pages 542–547, 2002.

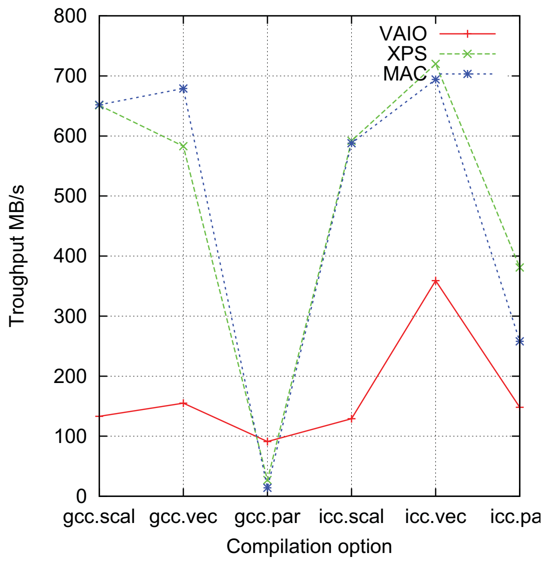


Figure 13. rms.dsp benchmark

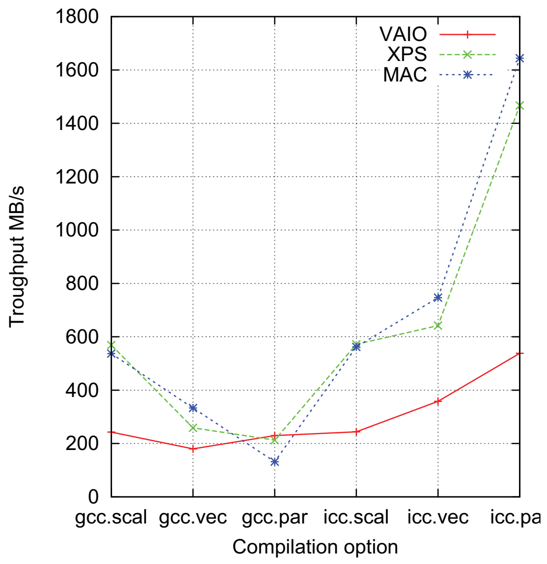


Figure 14. rms8.dsp benchmark

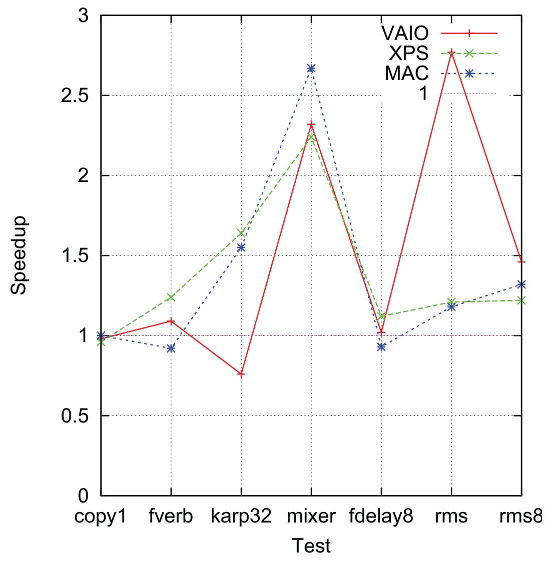


Figure 15. Speedup ratio between vector and scalar code (using icc)

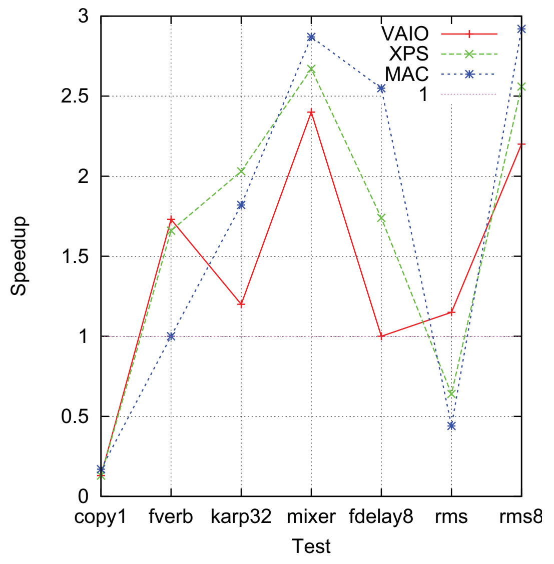


Figure 16. Speedup ratio between parallel and scalar code (using icc)

L-SYSTEMS, SCORES, AND EVOLUTIONARY TECHNIQUES

Bruno F. Lourenço, José C. L. Ralha, Márcio C. P. Brandão
Departamento de Ciência da Computação, Universidade de Brasília
{brunofigueira,brandao,ralha}@cic.unb.br

ABSTRACT

Although musical interpretation of L-Systems has not been explored as extensively as the graphical interpretation, there are many ways of creating interesting musical scores from strings generated by L-Systems. In this article we present some thoughts on this subject and propose the use of genetic operators with L-System to increase variability.

1 INTRODUCTION

L-systems are powerful tools for creating models of plants and other structures that have some degree of self-similarity. Typically, an L-system consists of a set of symbols and a set of *rewritting rules* which can be applied in a parallel basis. Figure 1 shows an L-system for the famous *dragon curve* using the syntax adopted by [1].

```
#level 11
#delta 90
#axiom FX

X -> X+YF+;
Y -> -FX-Y;
-----------------------------------
         Resulting string
0 FX
1 FX+YF+
2 FX+YF++-FX-YF+
3 FX+YF++-FX-YF++-FX+YF+--FX-YF+
```

Figure 1: L-system grammar for the dragon curve.

The usual way of extracting something interesting from strings generated by L-Systems, is to interpret each symbol as a command to a imaginary turtle, in a LOGO-like manner. For such approaches, "F" means draw a segment with length d, "+" means turn the turtle $+\delta$ degrees, "-" means turn the turtle $-\delta$ degrees. "X" and "Y" are just auxiliary symbols and do not have a graphical interpretation. After a few iterations, 11 to be precise, we derive from the L-System in Figure 1 the picture shown in Figure 2. Description of more complex graphical structures and a complete overview of different types of L-Systems can be found in [1].

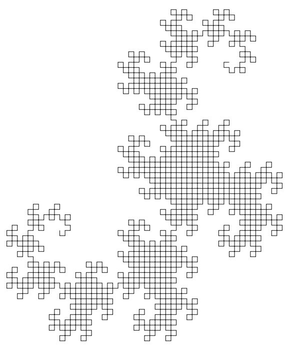


Figure 2: The dragon curve after 11 iterations on the L-system shown in Figure 1.

Of course, the graphical interpretation is not the only way to interpret the strings. Although most extensions to L-Systems focus only on the graphical interpretation, several authors described techniques to extract musical scores from strings produced by L-Systems [2], [3], [4], [5], [6]. Music has a certain fractal property [7], so it fits nicely in the context of parallel rewriting that the L-Systems provide. However, it's not a perfect fit. If the L-Systems or the rendering techniques are too simplistic, the resulting score will probably be equally simplistic, with the same theme or motif going over and over again, but starting at different points of the chosen musical scale.

In our research, we observed that the authors usually try to cope with this problem in two different ways: using more sophisticated L-Systems (stochastic or context-sensitive, for example) or using a more refined method for score generation in order to introduce variability. Keeping this idea in mind and borrowing a few operators from genetic algorithms, we have developed a method to increase variability by changing the set of rules after each iteration.

Each method of score generation has properties that make it more suitable to a certain type of L-System, but it's interesting to observe how the same L-System "behaves" under different renderings, so we have developed a program that implements three types of rendering: *spatial* [2], *sequential*

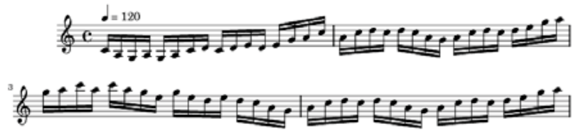


Figure 3: Score generated by projecting the graphical interpretation on a pentatonic scale

[3] and *schenkerian* [3].

Section 2 presents an overview of existing musical rendering. Section 3 introduces the notion of *Genetic L-Systems*. Section 4 describes a program written in the Python programming language that implements some musical renderings and our approach for Genetic L-Systems. Section 5 summarizes this work and suggests future directions.

2 MUSIC FROM L-SYSTEMS?

Arguably, one of the firsts articles on the subject of score generation and L-Systems was Prusinkiewicz's *Score generation with L-systems* [2], where he described a technique to extract music from the graphical interpretation of a string produced by an L-System. Each horizontal segment of the picture is interpreted as a note with a length proportional to the length of the segment. The pitch of a note is the y-th note of the chosen musical scale, where y is y-coordinate of the the segment. Figure 2 shows the first four bars of the score associated with the 9th iteration of the L-System in Figure 1.

Altough it is possible to generate interesting melodies with this *spatial rendering*, the musical rendering is still tied to the graphical rendering. So it's natural that other authors have sought to separate them. The sequential and schenkerian rendering described in [3] are examples of musical renderings that are completely independent of the graphical rendering. Both are well-suited to L-Systems that represent trees and other branched structures. The author also remarked that *"there seems to be enough information in a typical L-System to create only a short melody and still be interesting"*. To cope with this problem, Worth and Stepney suggested the use of context-sensitive and stochastic L-Systems, but some of the L-Systems built specially for the musical rendering did not have an interesting graphical interpretation, thus suggesting that it's very hard to conciliate, with aesthetic results, both the musical and the graphical interpretation.

Also worth mentioning is the LMUSe [8] program that uses map files to describe how to derive pitch, timbre and velocity from the turtle state. It has an interesting feature: a "mutate" button that randomly modifies the production rules before the first derivation step.

All the four musical renderings discussed so far are typically used with L-Systems that have sets of symbols that were originally designed for the graphical interpretation. A departure from this is the work of Jon Mccormack [4] [9], where he describes L-Systems that use notes (*A,B,C..,G*) instead of LOGO style commands (*F*,+,-).

As we have stated earlier, there is a problem with repetition and this article we try to address this issue. Even if we use stochastic rules, the set of rules is fixed, so we are still prone to hear the same fragments over and over again. The use of context-sensitive rules is a potential solution but it adds complexity to the process of building an L-System that makes sense melodically. So we want a mechanism that is simple while adding variability.

3 GENETIC L-SYSTEMS

Many authors have described techniques to "breed" L-Systems with genetic algorithms as a way of partially solving the so called *inference problem* [1] and, as a consequence, finding an L-System that produces a certain *graphical* structure. These approaches typically use genetic operators such as mutation and crossover to create new individuals (sets of production rules) and fitness functions to check if the population has a particular feature and to select the fittest individuals.

Jacob [10] described a technique to select L-Systems that produce plants with a certain branching pattern. Ashlock [11] bred populations of L-Systems to generate graphical renderings of landscapes. Mccormack [12] described an *aesthetical evolution*, where the user is asked interatively to inform the fitness of a graphical interpretation associated with a certain L-System at each step of the algorithm.

Most applications of L-Systems and evolutionary techniques are targeted to the graphical rendering. This is a surprising fact, since evolutionary techniques have been largely applied to computer music with interesting results. In [13] there is an overview of modern techniques for applying evolutionary concepts to sound synthesis and algorithmic composition. While we do not claim that we are filling the gap between the use of L-Systems and evolutionary techniques to create music, we believe that, at least, we are providing some inital steps.

There are many different techniques to mutate and to do the crossover of productions rules, which are not the hardest parts of combining genetic algorithms with L-Systems. Arguably, it's how to define the fitness function that causes the difficulties. We have taken a different aproach and decided not to use a fitness function at all. Instead, we designed an extension that allow mutations and crossover between sucessive iterations of an L-System. Consider the L-System shown on Figure 4.

[1] The inference problem asks for an axiom and set of production rules that capture a certain growth process.

```
#axiom FX

X -> X+YF+
Y -> -FX-Y,crossover(0,1)
--------------------------------------
         Resulting string
0 FX
1 FX+YF+
2 FX+YF++-FX-YF+
3 FXF++-+YFX-YF++-FX+--F+YFX-YF+
```

Figure 4: Genetic dragon curve

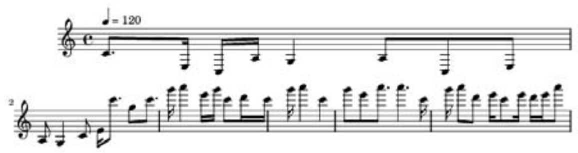


Figure 5: Spatial Rendering and a "genetic" dragon curve.

It is an extreme example, where crossover between the rules 0 and 1 is done each time the symbol Y is rewritten. But as odd as it may seem, comparing the score generated by spatial rendering with the original dragon curve shows a vast improvement, see Figures 3 and 5.

With parametric rules it's possible to mutate or crossover an L-System when certain conditions are met, thus allowing greater control, as shown in Figure 6.

After a rule is matched, we substitute the sucessor and then we proceed to evaluate the operators, if any. We take a simplistic view on the genetic operators as we are considering that they can only change the sucessor of the rules. More sophisticated descriptions of mutation and crossover between L-System rules can be found on [9], [10], [14], [15]. Our focus here is on the genetic operators, which are now intrinsic to the L-Systems, and that they can be parameters of the model and not just external agents. Now that's clear how we intend to use the genetic operators, we can discuss exactly how to mutate and/or do the crossover between rules.

3.1 Mutation of rules

The mutation can be thought as a function or a procedure that has two parameters: the number of the rule that will be mutated and the probability of mutation. The mutation operator scans each symbol of the sucessor of the chosen rule and then generates a random number between 0 and 1. If the generated number is less than the probability of mutation, it chooses randomly between the symbols in the

```
#axiom -XA(0)B(0)
#mutation_pool  F + -
#mutation_ignore  ( ) [ ]

X -> -YF+XFX+FY-
Y -> +XF-YFY-FX+
A(t) -> A(t+1),(t%2)==0:crossover(0,1)
A(t) -> A(t+1),(t%2)!=0:
B(t) -> B(t+1),t==3:mutation(0,0.5)
B(t) -> B(t+1),t!=3:
--------------------------------------
         Resulting string
0 -XA(0)B(0)
1 --YF+XFX+FY-A(1)B(1)
2 --+XF-YFY-F+XFX+F+-YFX+FY-F-YFX+FY-+F+
  XF-YFY-F+XFX+-A(2)B(2)
```

Figure 6: Genetic Hilbert Curve

```
#axiom -X

X -> -YF+XFX+FY-:
Y -> +XF-YFY-FX+:
-------------------------------------
         Resulting string
0 -X
1 --YF+XFX+FY-
2 --+XF-YFY-FX+F+-YF+XFX+FY-F-YF+XFX+FY-
  +F+XF-YFY-FX+-
```

Figure 7: Canonical Hilbert Curve

mutation pool and mutate the symbol in the sucessor. For example, *mutation(0,0.5)* in Figure 6 refers to a mutation in rule 0 with probability of 0.5 for each symbol.

There are certain symbols that should not be mutated because they would affect the consistency of the rules. Therefore we have to keep a list of ignored symbols, that must be skipped when we are scanning through the successor of a certain rule. In bracketed L-Systems, for example, the '[' pushes the turtle state on a stack and ']' pops the turtle state. As we usually have the same number of '[' and ']', we can not allow disruptions in this balance.

3.2 Crossover between rules

The crossover operator introduces variability by recombining two rules. Usually, we generate a random integer and then we split and combine both rules at that point. But since the sucessor of the rules can have different sizes, we have adopted the *two-point-crossover*. We generate two pseudo-random numbers for each rule, and then we swap the characters whose indexes are between these two numbers. Suppose we have generated the numbers 2,3 and 0,2 and that we have the following two rules:

```
X -> X+YF+
Y -> -FX-Y
```

The substring, in rule 0, delimited by the indexes 2 and 3 is `YF`, and the substring, in rule 1, delimited by the indexes 0 and 2 is `-FX`, so we swap them to get new rules:

```
X -> X+-FX+
Y -> YF-Y
```

If we allow bracketed or parametric rules, extra care must be taken to avoid creating unbalanced or syntatic incorrect rules. So before doing the crossover, we break the rule in tokens. Consider these two rules:

```
X -> F-[[X]+X]+F[+FX]-X
F -> FF+FF
```

We break the the first rule in F, -, [[X]+X], +, F, [+FX], - and X. The second rules gives us F,F,+,F,F. Suppose we have generated the numbers 1, 5, and 1,3. This would give us the following rules:

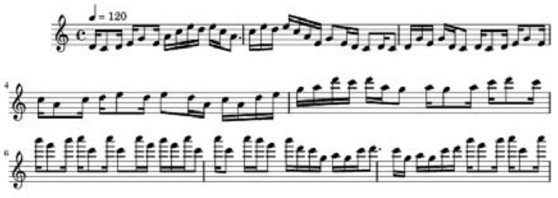
```
X -> FF+F-X
F -> F-[[X]+X]+F[+FX]F
```

3.3 Further Remarks

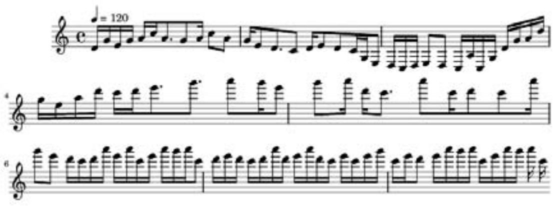
We do not describe our technique as an application of genetic algorithms because we do not have a fitness function nor a population. We pick up instead a single L-System, and we introduce the hability of mutating and to do crossover between successive iterations. Since we do not use a fitness function, the only way of evaluating the “fitness” of the musical interpretation of a certain L-System is by listening to the resulting score. In chapter 2 of *Evolutionary Computer Music* [13], John Biles argues that the most difficult part of composing music with genetic algorithms is how to specify the fitness functions, since the notion of what is right and wrong is highly subjective when we are dealing with music. Without fitness functions we open the possibility of generating scores with non-conventional musical structures, but we do not have an objective way of evaluating if the changes made by the genetic operators were positive or negative. Since our main purpose is to introduce variability, we do not impose any constraints on the operators and let the user decide himself whether a certain music piece is fit or not for his purposes.

We propose two ways of modifying an existing L-System:

- Adding a mutation or crossover operator at the end of an existing rule. The dragon curve that was discussed previously is an example of this technique.



(a) Canonical



(b) Genetic

Figure 8: Spatial rendering of the Hilbert curve

- Adding symbols to control the genetic operators. Figure 6 shows an example of this technique and Figure 7 shows the canonical Hilbert curve. Also, compare the scores produced by both L-Systems on Figure 8.

In our experiments we found out that the first technique gave the most dramatic results, since the same operator could be applied many times at each iteration thus drastically changing the set of productions rules. The second technique could be used to introduce slight deviations in the set of production rules, thus generating a score that has some traces of the original L-System.

An interesting possibility is the use of L-Systems in a similar fashion to what Mason and Saffle described on [7]. They used the spatial rendering described by Prusinkiewicz and different graphical rotations of the same L-System to build a longer musical piece while creating the feeling of counterpoint. Instead of using different rotations, we could use several realizations of the same L-System, since each realization produces a different score due to the probabilities associated with the genetic operators.

4 LSCORE

We wrote a program in Python that implements our ideas of genetic L-Systems. It is an ongoing research where we are able to generate MIDI files using different musical renderings. It’s also a parser for DOL, 2L, stochastic, bracketed and parametric L-Systems. The data flow is described on Figure 9 . It uses Python’s reflection mechanism to parse and execute the rules as Python code, allowing the user to

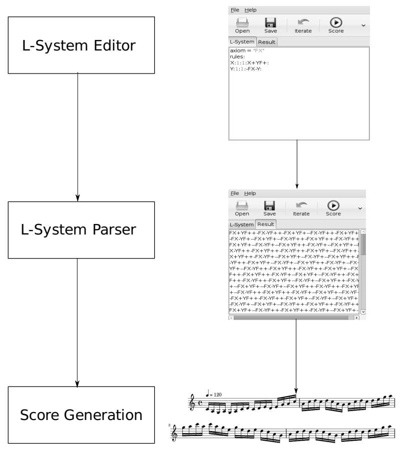


Figure 9: Data Flow in LScore

call his own routines from the production rules. The format of the rules file is shown below:

```
<Python Code>
axiom = "xy..z"
rules:
rule 1
rule 2
...
rule n
```

Each rule must use the following syntax:
predecessor:cond:prob:sucessor:code

As an example, consider the rule: `B(t):t==3:1:B(t+1):mutation(0)` The *predecessor* is `B(t)`, which is the symbol that will be replaced. The condition is `t==3`, the symbol will be replaced only if the condition is true. The probability is 1, and therefore this is a deterministic rule. The *sucessor* is `B(t+1)` and this the string that will replace the symbol `B(t)`. The *code* is `mutation(0)` and it is the procedure that will be executed after the symbol is replaced.

Figures 10 and 11, shows respectively the Genetic Hilbert and the Dragon curve written using LScore's syntax.

4.1 L-System evaluation and score generation

It is actually pretty simple to implement and evaluate L-Systems if we are dealing with non-parametric rules, since we only need to concatenate strings, check for context and generate pseudo-random numbers. But when we have parametric rules, an L-System becomes almost like a computer program of its own. Consider the following L-System:

```
axiom = "A(1)"
rules:
A(t):t==1:1:A(t*2):
```

The parametric rule `A(t):1:1:A(t*2)` matches the module `A(1)` because the letter in the production rule and in the module are the same, the number of formal parameters are also the same, and the condition (`t==1`) evaluates to true. After the rule is matched, the parameters are evaluated, the module is substituted, and we find the string `A(2)`. While it is not hard to write a parser to evaluate the arithmetic expressions that appear on parametric rules and to check if the conditions are true, a much simpler implementation is possible if we can execute statements and evaluate expressions at runtime, thus delegating the issue of expression parsing and execution to the underlying language interpreter. For that specific example, we would have two statements: `exec("t=1")` and `eval("t==1")` Since after the first statament, t is indeed equal to 1, the last statement returns `True` and then we can evaluate the sucessor of the rule: `eval("t*2")`, which returns 2. Both *exec* and *eval* are built-in statements in Python.

The genetic operators are coded in a similar way. As we stated earlier, each rule has a *code* part, which can be empty. We coded the genetic operators *crossover* and *mutation* as Python functions that change the production rules. After a symbol is replaced, we do `exec(code)` and the Python interpreter executes the *code* part of the rule.

After sucessive iterations, we can interpret the resulting string as a score. In our implementation, we generate a standard MIDI file with the score. The user has the option of choosing the instrument, the key, the musical scale, the method of rendering (sequential, schenkerian or spatial), the initial octave and a few other small tweaks.

The implementation of the rendering methods is straightforward, since we just have to scan the resulting string and interpret each symbol correctly according to the chosen method. At this moment, the MIDI capabilities of LScore are at best rudimentary, since we record the score in a single track and do not allow instrument changes during the rendering process. Nevertheless, it is possible to render interesting melodies and scores.

```
axiom = "-XA(0)B(0)"
rules:
X:1:1:-YF+XFX+FY-:
Y:1:1:+XF-YFY-FX+:
A(t): (t%2)==0:1:A(t+1):crossover(0,1)
A(t): (t%2)!=0:1:A(t+1):
B(t): t==3 :1:B(t+1):mutation(0)
B(t): t!=3:1:B(t+1):
```

Figure 10: Genetic Hilbert Curve with LScore's syntax. The *mutation_pool* and *mutation_ignore* are implicitely defined.

```
axiom = "FX"
rules:
X:1:1:X+YF+:
Y:1:1:-FX-Y:crossover(0,1)
```

Figure 11: Genetic Dragon Curve with LScore's syntax

5 CONCLUSION

In this article we presented a brief overview of existing methods for extracting musical scores from L-Systems and introduced a few ideas of our own. More specifically, we also presented the concept of Genetic L-Systems, where the set of productions rules can be changed between sucessive iterations. These changes are made by two genetic operators: *crossover* and *mutation*. We have also briefly described a program written in Python that implements our approach for Genetic L-Systems and generates MIDI files.

We believe we have succeeded in introducing variability in the musical interpretation of L-Systems, but certainly there is room for more experimentation. As we stated earlier, the genetic operators we are using are very simple in the sense that they only modify the sucessor of the production rules. A more sophisticated mutation process, for example, could further enhance the resulting musical score.

Also, the LScore program lacks some features, such as better MIDI support. So we have the intention of addressing these issues in the future.

6 ACKNOWLEDGEMENTS

The authors would like to thank Dr. Jon McCormack for promptly sending his PhD thesis.

7 REFERENCES

[1] P. Prusinkiewicz and A. Lindenmayer, *The algorithmic beauty of plants*. New York: Springer-Verlag, 1990. [Online]. Available: http://algorithmicbotany.org/papers/#abop

[2] P. Prusinkiewicz, "Score generation with L-systems," *Proceedings of the 1986 International Computer Music Conference*, pp. 455–457, 1986.

[3] P. Worth and S. Stepney, "Growing music: musical interpretations of L-systems," *Springer*, vol. 3449, pp. 545–550, 2005. [Online]. Available: http://www-users.cs.york.ac.uk/~susan/bib/ss/nonstd/eurogp05.htm

[4] J. McCormack, "Grammar-based music composition." [Online]. Available: http://journal-ci.csse.monash.edu.au/ci/vol03/mccorm/mccorm.html

[5] G. L. Nelson, "Real time transformation of musical material with fractal algorithms," *Computers & Mathematics with Applications*, vol. 32, no. 1, pp. 109–116, Jul. 1996.

[6] M. Supper, "A few remarks on algorithmic composition," *Computer Music Journal*, vol. 25, no. 1, pp. 48–53, Mar. 2001. [Online]. Available: http://dx.doi.org/10.1162/014892601300126106

[7] M. Saffle and S. Mason, "L-Systems, melodies and musical structure," *Leonardo Music Journal*, vol. 4, p. 8, 1994.

[8] D. Sharp, "Lmuse," http://www.geocities.com/athens/academy/8764/lmuse/lmuse.html, 2001.

[9] J. McCormack, "The application of L-systems and developmental models to computer art, animation, and music synthesis," Ph.D. dissertation, School of Computer Science and Software Engineering, Monash University, Clayton, Australia, 2003.

[10] C. Jacob, A. Lindenmayer, and G. Rozenberg, "Genetic L-System programming," *Parallel Problem Solving from Nature III, Lecture Notes in Computer Science*, vol. 866, pp. 334—343, 1994. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.7334

[11] D. Ashlock, S. Gent, and K. Bryden, "Evolution of l-systems for compact virtual landscape generation," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 3, 2005, pp. 2760–2767 Vol. 3.

[12] J. McCormack, "Interactive evolution of L-System grammars for computer graphics modelling," 1993. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.6763

[13] E. R. Miranda and J. A. Biles, *Evolutionary Computer Music*. Springer, April 2007.

[14] C. Roger, "On the evolution of parametric L-Systems," University of Calgary, Tech. Rep., 2000.

[15] K. Mock, "Wildwood: the evolution of L-system plants for virtual environments," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 476–480.

A FRAMEWORK FOR MUSICAL MULTIAGENT SYSTEMS

Leandro Ferrari Thomaz and Marcelo Queiroz
Department of Computer Science - University of São Paulo
{lfthomaz | mqz}@ime.usp.br

ABSTRACT

Multiagent system technology is a promising new venue for interactive musical performance. In recent works, this technology has been tailored to solve specific, limited scope musical problems, such as pulse detection, instrument simulation or automatic accompaniment. In this paper, we present a taxonomy of such musical multiagent systems, and an implementation of a computational framework that subsumes previous works and addresses general-interest low-level problems such as real-time synchronization, sound communication and spatial agent mobility. By using it, a user may develop a musical multiagent system focusing primarily in his/her musical needs, while leaving most of the technical problems to the framework. To validate this framework, we implemented and discussed two cases studies that explored several aspects of musical multiagent systems, such as MIDI and audio communication, spatial trajectories and acoustical simulation, and artificial life constructs like genetic codes and reproduction, thus indicating the usefulness of this framework in a variety of musical applications.

1 INTRODUCTION

Agent technology is particularly suited for musical applications due to the possibility of associating a computational agent with the role of a singer or instrumentalist. With these associations one can map features such as performance, perception, adaptation and improvisation on one side, and artificial processes on the other. Moreover, it is possible to define forms of social interrelationship between agents, which brings this technology even closer to collaborative musical performance. By focusing the discussion on multiagent systems we purposely leave aside any aesthetic issues specific to the choice of compositional algorithms for each agent, and concentrate on the study of communication and interaction (i.e., sociology) of musical agents in the collective musical production context [1, 2].

Most of the previous work involving computer music and multiagent systems is generally very limited in its scope, dealing with problems such as generation and evaluation of melodies [3], pulse detection [4], simulation of specific instruments [5] or automatic accompaniment and collective performance in a tonal context [6, 7]. A more ellaborate example may be found in Living Melodies [8], in which the authors build an artificial world of music players with a well-defined spatial structure (including sound propagation), as well as rules for walking, interacting and music-making. Works that aim at more general settings are the MAMA architecture [9], where agents communicate using a symbolic representation of the piece being performed, and the Andante project [10], that allows agents to migrate between machines in a distributed computer environment.

This work aims at both a definition and an implementation of a general framework for musical multiagent systems, as well as a study of their inherent problems. We extend on [9] by allowing synchronous and asynchronous communication of various sorts (audio signals, symbolic music streams, video signals and other forms), artificial life concepts (life, death and reproduction of agents) and physical simulation (sound propagation and motion of agents). Our musical agents can also benefit from the migrating abilities of Andante's agents [10], since both systems are written in Java.

A central issue in this project is the treatment of space, a musical attribute of utmost importance in contemporary musical composition, which has been superficially explored in the context of multiagent systems in previous works [8]. Thus, one of the main goals of our framework is to allow the simulation of sound propagation in a virtual environment, automatically adjusting the sound information received by each agent, depending on its position and listening conditions.

Through the observation of common elements among existing musical multiagent systems and by proposing new features and tools, we intend to put forward both a conceptual and a computational framework that eases the task of implementing a musical multiagent system that best fits a given musical application. On one hand, simulation of existing musical multiagent applications should be straightforward by using the framework with its basic components; on the other hand, extending the framework by adding new features such as compositional algorithms, music and sound analysers and synthesisers, or rules for handling artificial life aspects of the virtual musical agent society, should be made easy by freeing the user from low-level implemen-

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

tation details such as synchronization and communication protocols, and allowing him/her to concentrate on the conceptual level of the musical project.

2 A TAXONOMY FOR MUSICAL AGENTS SYSTEMS

This section presents a preliminary taxonomy of musical multiagent systems that has been generalized from previous musical multiagent works [8, 9, 10, 5, 6, 11, 7, 4], as well as other works that deal with related issues [3, 12].

The higher level categories considered in a musical multiagent system are the musical agent, the virtual environment, and interactions (both among musical agents and between musical agent and virtual environment). Figure 1 summarizes these categories and their components.

2.1 Virtual Environment

An environment in the context of a musical multiagent system can be defined as a virtual space in which computational agents are immersed and interact with it by means of sensors and actuators.

Physical Representation. An environment's physical representation includes the definition of a virtual world together with every physical information that is relevant to the musical application. These may include space representation (dimensionality, connectedness), sound representation (MIDI, audio) and propagation (including acoustical effects such as air absorption, reflection, diffraction or Doppler effect), and mechanical laws (gravity, inertia, attraction and collisions).

Ecological Representation. Systems inspired in Artificial Life [8, 3] often use some representation for energy to control a few aspects in an agent's life, such as eating, walking, growing old, among others. Energy may be defined and used to control which actions (requiring some amount of energy) an agent may perform, if it needs refueling (through food consumption, for instance) or rest, or if it is able to reproduce.

2.2 Musical Agent

The musical agent is a computational agent specialized in processing sound and musical information. Typically, this agent is capable of analysing incoming sound, performing some sort of musical reasoning, and creating a response by means of sound processing and synthesis. Figure 2 shows a musical agent and its components, described below, immersed in a virtual environment.

Knowledge Base. The knowledge base is a storage area where the agent keeps everything related to its know-how in music-making as well as its memory of past events. This includes algorithms for composition, music theories, non-musical facts about the environment and other agents, techniques for music or sound encoding and processing, linguistics, ontology, among others.

Reasoning. We denote by Reasoning the set of mechanisms an agent uses to decide its future actions, musical or otherwise. An agent's output may combine several compositional strategies, such as playback of recorded fragments, context-based reactions to musical input, or analysis and synthesis tools within a structural plan to achieve a compositional goal.

Sensors and Actuators. Sensors capture information from the environment and forward it to the cognitive center (Knowledge Base + Reasoning) of the agent. They are usually specialized in receiveing a particular type of sensorial information, for instance auditory sensors, visual sensors or tactile sensors. An agent may have several similar sensors (e.g. two ears, two eyes) to aid its cognitive capabilites (e.g. to perceive direction or distance of a sound source).

Actuators are the active counterparts to sensors, and are responsible for affecting the environment, driven by the agent's reasoning. They produce sound, perform movement, and generate events that may change the way the world is perceived by other agents.

Analysis and Synthesis. In addition to any kind of analysis automatically done by an agent's sensors, higher level analysis may be requested by the agent's reasoning in order to achieve its musical goals. These may include contextual analysis (comparing instantaneous inputs to previous data) and planning of future events (comparing possible outputs to expected outputs by other agents), and may include a number of specific techniques of signal processing such as temporal and spectral analysis, analysis of psychoacoustical phenomena (such as pitch detection or perceived loudness) and musical analyses of all sorts (rhythmic, melodic and harmonic analysis, or analysis of genre, style and expression).

Synthesis may be regarded as the most fundamental part of a musical agent, since through it an agent partakes musically in a collective performance. It includes symbolic synthesis of high-level events (as in MIDI or MusicXML), sound synthesis and signal processing techniques, but also non-musical information such as spatial trajectories or graphical output (to communicate visually with other agents or as a visual counterpart to the musical performance).

2.3 Interactions in the Environment

In a collaborative musical performance there are many kinds of interaction that may be simulated in a musical multiagent system. Besides the obvious exchange of sound information during performance (through hearing and playing together), other information may be exchanged, such as gestures for guiding the performance (for instance in slowing

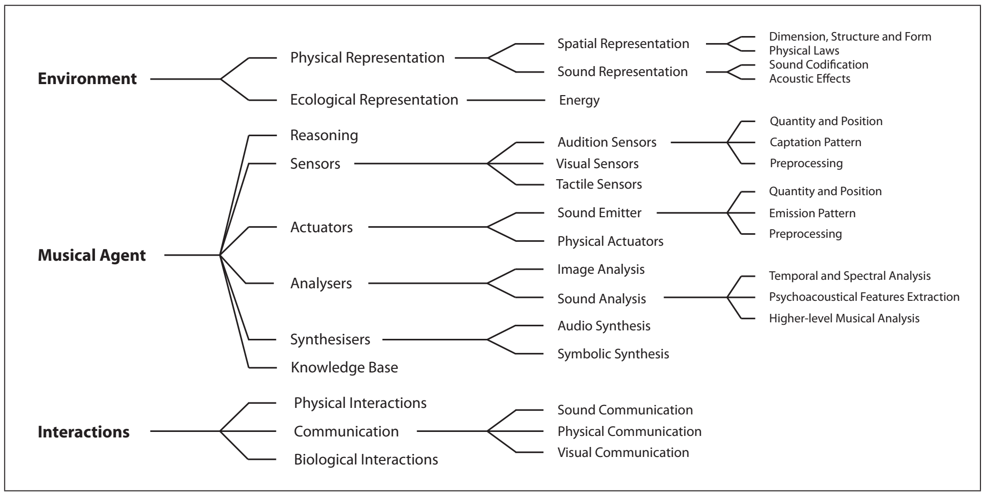


Figure 1. Taxonomy of Musical Multiagent Systems

the tempo down), or may be agreed upon before the actual performance starts (such as a musical score or a general plan for the composition).

When the simulation includes physical or biological elements, other interactions may be observed, among agents or between agent and environment, that affect their states. These include mobility of agents through external forces or free-will, feeding and reproduction, to name a few.

3 FRAMEWORK'S ARCHITECTURE AND IMPLEMENTATION

The present architecture was designed with plugability in mind, that is, it should be possible for a user to develop new components which are easily added to the original framework, and also connect component instances to an agent in execution time. For instance, a running system might consist of simple agents with one sensor for receiving sound and one actuator for emitting sound; later on, the user might want to change those into anthropomorphic musical agents, by plugging another sound sensor at a different position in the agent's body, and creating a component that analyses incoming sounds received by both sensors and infer sound source positions.

The framework presuposes the utilization of a multiagent system middleware that provides the required infrastructure for maintaining agent execution and controlling messages exchanged among agents. The current implementation uses JADE 3.6 [1] with the Java 6 programming language.

[1] available at `http://jade.tilab.com/`. 13 april 2009.

3.1 Framework's Actors

Relying on the taxonomy described in the last section, a *musical agent* is modeled as an aggregate of linked components, such as reasoning, knowledge base, sensors, actuators, analysers and synthesisers. In order to create a musical agent, one defines its components either by reusing existing ones, or by extending basic classes to create new components.

The virtual environment is represented by a unique agent called *environment agent*. This agent is composed by a physical representation of the virtual world in which the agents live, and by event server components. *Interactions* are represented by events that flow between musical agents and the environment agent, where each event type represents a particular kind of interaction, such as sound exchange, verbal messages, gestures, and so on.

An *external agent* is a human or an outside system that interacts with the framework in runtime. It is embodied in the virtual world by a regular musical agent, so that it interacts with the environment and other musical agents using the default mechanisms. For instance, a compositional algorithm implemented in Pure Data [2] might communicate with the framework using sockets, by sending a stream of notes to be played through the agent's actuator.

3.2 Virtual Time

Time in the virtual environment is controlled by a virtual clock that allows agents to obtain the current time and schedule tasks. This virtual clock may be managed in two different ways: *by the internal clock*, which means that the virtual clock is tied to the computer's internal clock and so to the

[2] available at `http://crca.ucsd.edu/~msp/software.html`. 13 april 2009.

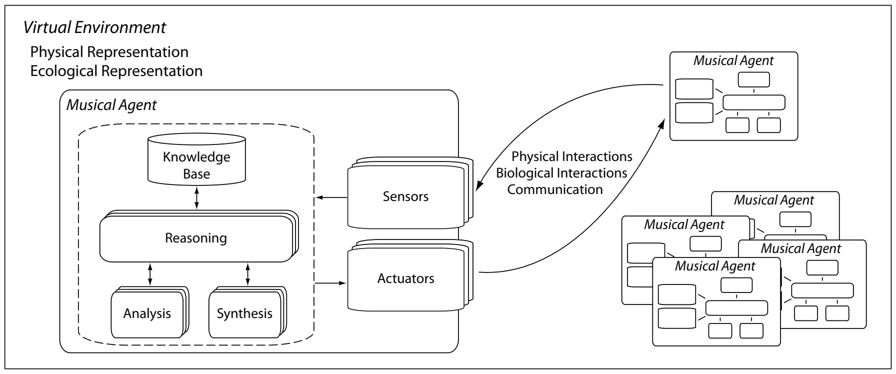


Figure 2. Musical Agents in the Virtual Environment

external flow of time; and *by the user*, which means that the user, usually through the environment agent, updates the clock, giving rise to non-homogeneous time measurements.

3.3 Communications

Communication in the multiagent middleware serves two different purposes: to enable the operation and control of the framework, and to implement the aforementioned interactions in the virtual world. The first purpose is addressed by commands, while the latter correspond to communication via events.

Commands. A command is a message passed between agents that controls the framework's internal functioning. Every agent has a dedicated asynchronous communication channel that is able to send and receive commands, and is used to pass control messages such as registering an agent, informing a change of turn (see 3.4), or updating a public fact in its knowledge base.

Events. Interactions between musical agents and the environment is done by means of exchange of events, which is always controlled by the environment's event servers. Events are used to model all sorts of interactions in the virtual world, such as exchange of audio chunks, an agent's collision with an obstacle, non-musical messages between agents, among others.

There are two exchange modes for events: *sporadic* events, such as gestures or verbal messages, can be sent at any rate and instant; and *periodic* events, such as audio chunks, which must obey a frequency of exchange that is previously agreed upon by all components that use that event type.

The periodic exchange mode is a synchronous communication process involving a set of sensors, actuators and an event server. In this case, virtual time is divided in frames of the same period as the periodic event, and processing is always done ahead of time, i.e. while an event is happening in the environment, reasonings and actuators are working

	Deadline	Description
t_0	*frameTime*	Frame start time.
t_1	*needAction*	In this instant, the actuator automatically warns its registered reasonings that an action must be taken to produce the next frame.
t_2	*agentDeadline*	Deadline for actuators to send frames to the event server.
t_3	*receiveDeadline*	Deadline for events sent by actuators to arrive at the event server; after this time, arriving events are discarded.
t_4	*sendDeadline*	Deadline for the event server to send events back to registered sensors.
t_5	*period*	End of current frame, and simultaneous start of the next one.

Table 1. Deadlines for periodic exchange mode

to send the event corresponding to the next frame. Actuators and event servers have state machines that tell them to work or to wait for some response from other agents, and their internal states are affected by scheduled tasks which are triggered at user-defined times. The deadlines for agents and event server is detailed in table 1.

The communication interface, present in every sensor, actuator and event server, is responsible for sending and receiving events by means of the callback methods *send()* and *receive()*, respectively. Two interfaces were implemented and compared: *communication by messages*, that uses the message passing system native to JADE, based on FIPA-ACL messages; it works by encapsulating an event inside a message and relaying its delivery to JADE; this kind of communication can be used in networked systems, but it may be slow and therefore not suited to time-sensitive events; and *communication by direct calls*, implemented as a single JADE service instance that can be accessed by every agent; a component must register its access point in this service, which will be used by the service to deliver the event by a direct call to the interface's *sense()* method; this implementation has the advantage of being much faster than the traditional message passing method, but on the other hand it blocks the service process while *sense()* does not return, so it must be carefully and thoughtfully used.

3.4 Execution Mode

There are two possible execution modes: *Batch mode*, where time is discrete and divided in turns, and is updated by the environment agent only after every agent responds with whatever it is supposed to produce in that turn, with no time constraints whatsoever. It can be used when there is a need to control the sequence of actions, or it can be used in computationally intense non-interative processes to create a musical output for later appreciation; and *Real-Time mode*, where time is controlled by the internal clock, and every agent is responsible for producing their outputs ahead of time, or else they will be silent, i.e. the environment agent does not wait every agent to complete its action, and only forwards events that arrived up to each deadline. Agents may rely on some fallback strategy that sends previously computed outputs whenever a certain real-time deadline cannot be met.

4 CASE STUDY

To validate the current stage of the framework development, two different systems were implemented, to test the framework for user programmability, robustness and computational performance. These two systems were chosen in order to explore the use of both symbolic and audio communication streams, both batch mode and real-time execution modes, and artificial life concepts.

4.1 Living Melodies

The Living Melodies [8] system is a complex example of musical multiagent system having various kinds of sensors and actuators that deal with sound, energy, life cycle, movement, and contact. Agents sing, walk, reproduce and die, and their physiology includes not only hearing and singing devices, but also tactile sensors, genetic codes, preference rules for reproduction and energy management. The system uses a symbolic codification of sound as musical events, which are propagated through a bidimensional discrete space in a manner similar to wave fronts.

The simulation of the Living Melodies system was based on an article [8] and a software [3]. Although not explicitly said by the authors, it seems to use a monolithic non-real-time architecture, in which the processing of agents' actions is made in a round-robin fashion by a single thread. The mapping to the framework required the use of the batch execution mode.

The re-implementation of this system within our framework allows the user to fine-tune many parameters of the simulation, including the number of agents, genetical material, agent's age limits, sound absorption by the air, world size, among many others. It also features a graphical user interface that shows the spatial distribution of the agents in the world, as well as wavefronts of sound propagation.

[3] available at `http://www.ituniv.se/~palle/projects/living-melodies/`. 13 april 2009.

4.2 Audio Exchange and Acoustical Simulation

A simple audio system was conceived to test the framework capabilites in dealing with real-time audio exchange. Musical agents are positioned in a virtual two-dimensional space, and produce audio streams that are constantly broadcasted to other agents. Each agent receive a mixture of the sound in the environment, considering the effects of delay and attenuation of every other signal according to relative distances between agents. The user may be immersed in the virtual environment, disguised as a special musical agent, who captures all sound received at its virtual position, and renders it through an audio interface. As an example of application, this system might be used for placing several human musicians in a virtual environment, and letting them play together in real-time, hearing each other as they would in the virtual acoustic space.

In real-time terminology, this kind of audio exchange is classified as soft real-time since the loss of a deadline is not catastrophic for the system. Such a loss simply implies that an agent will be mute during a time frame. On the other hand, the loss of a deadline by the event server is more serious because it will cause all agents to be mute for a frame. To test for robustness and performance of the system, we measured the number of successful fragments delivered to the event server, as a function of both the number of agents and the size of the audio chunks. Table 2 shows the results over 25 seconds of digital audio delivered by each agent [4].

Samples	**Period**	**# Musical Agents**					
	(ms)	**1**	**5**	**10**	**25**	**50**	**100**
44100	*1000*	100	100	100	100	100	100
22050	*500*	100	100	100	100	98	77
11025	*250*	100	100	100	97	75	*
4096	*90*	100	100	100	86	*	*
1024	*23*	100	100	93	*	*	*
512	*11*	100	100	67	*	*	*
256	*5*	100	90	*	*	*	*

Table 2. Percentage of fragments received by the Event Server. Stars represent simulations that could not keep a constant audio exchange rate.

As expected, there is a tradeoff between the number of agents and the size of audio chunks, in order to keep full functionality. When the number of agents increases past a certain point, the system performance lowers due to the increased number of threads and, consequently, of process time and memory. Also, very small chunks increase the

[4] Using a Intel Core 2 Quad 2.40 GHz, 4 GB of and Windows Vista.

rate of loss because there is less processing time available to agents within each time frame. Since this implementation uses a temporal resolution to schedule tasks of the order of miliseconds, and the deadlines are too close to each other, this lack of precision can sometimes desynchronize the agents' state machines.

5 DISCUSSION AND FUTURE WORK

The implementation of two systems (Living Melodies and Audio Exchange and Acoustical Simulation) has shown that the current state of framework development is capable of covering systems that have different demands, such as real-time digital audio and exchange of symbolic sound information. Features made available by the framework allow the user to concentrate in solving his/her specific musical problem without worrying about lower level problems, such as communication between agents and synchronization.

As for the exchange of periodic events, tests have shown that it is possible to accomplish real-time audio exchange between agents, keeping a constant audio rate, by carefully choosing the size of the audio chunks as a function of the number of agents needed. However, since audio chunk size determines the overall latency in agents' perception of each other's performance, it is desirable to improve the communication mechanism in order to allow for smaller audio chunks with larger number of agents.

Apart from refinements in the current implementation of the framework that might lead to a better performance, another more structural approach to this problem is to use the Real-Time Java Specification [5] , which provides a high resolution time clock (in nanoseconds), and the possibility to do real-time scheduling with this resolution. Although using a real-time operating system and a real-time Java infrastructure to execute the application should produce better results, we intend to keep both implementations available, so as not to impose complicate system requirements on the common user.

From the user interface point-of-view, we intend to provide access for users of various levels of expertise. Users with no programming experience or who only need existing components can define a musical multiagent system simply by writing a text description, which is interpreted and executed by the framework. This script only points out which components (names, quantity and parameters) belong to each kind of musical agent and environment agent, and the global parameters of the simulation. This might also be done through a graphic interface, where one manipulates existing components, their interconnections and parameters. During the execution of the system, it would be possible to visualize a representation of the virtual environment with its agents, and to plug-in new components in runtime. Advanced users with programming skills may extend the framework by programming new components with special features using Java, either adapting existing codes to their needs or writing out new components and maybe new features for the architecture.

[5] available at `http://java.sun.com/javase/technologies/realtime/index.jsp`. 13 april 2009.

6 REFERENCES

[1] H. J. Levesque, P. R. Cohen, and J. H. T. Nunes. On acting together. *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, page 9499, 1990.

[2] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(3):223–250, 1993.

[3] E. R. Miranda. Emergent sound repertoires in virtual societies. *Computer Music Journal*, 26(2):77–90, 2002.

[4] S. Dixon. A lightweight multi-agent musical beat tracking system. *PRICAI 2000: Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, page 778788, 2000.

[5] L.L. Costalonga, R.M. Vicari, and E.M. Miletto. Agent-based guitar performance simulation. *Journal of the Brazilian Computer Society*, 14:19–29, 2008.

[6] G. L. Ramalho, P. Y. Rolland, and J. G. Ganascia. An artificially intelligent jazz performer. *Journal of New Music Research*, 28(2):105–129, 1999.

[7] R. D. Wulfhorst, L. Nakayama, and R. M. Vicari. A multiagent approach for musical interactive systems. *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 584–591, 2003.

[8] P. Dahlstedt and M. G. Nordahl. Living melodies: Coevolution of sonic communication. *Leonardo*, 34(3):243–248, 2001.

[9] D. Murray-Rust, A. Smaill, and M. Edwards. Mama: An architecture for interactive musical agents. In *Ecai 2006*, 2006.

[10] L.K. Ueda and F. Kon. Mobile musical agents: the andante project. In *Conference on Object Oriented Programming Systems Languages and Applications*, pages 206–207. ACM New York, NY, USA, 2004.

[11] P.A. Sampaio, G. Ramalho, and P. Tedesco. CinBalada: a multiagent rhythm factory. *Journal of the Brazilian Computer Society*, 14:31–49, 2008.

[12] P. M. Todd and G. M. Werner. Frankensteinian methods for evolutionary music composition. *Musical Networks: Parallel Distributed Perception and Performance*, 1999.

LEARNING JAZZ GRAMMARS

Jon Gillick
Wesleyan University
Middletown, Connecticut, USA
jrgillick@wesleyan.edu

Kevin Tang
Cornell University
Ithaca, New York, USA
kt258@cornell.edu

Robert M. Keller
Harvey Mudd Collge
Claremont, California, USA
keller@cs.hmc.edu

ABSTRACT

We are interested in educational software tools that can generate novel jazz solos in a style representative of a body of performed work, such as solos by a specific artist. Our approach is to provide automated learning of a grammar from a corpus of performances. Use of a grammar is robust, in that it can provide generation of solos over novel chord changes, as well as ones used in the learning process. Automation is desired because manual creation of a grammar in a particular playing style is a labor-intensive, trial-and-error, process.

Our approach is based on unsupervised learning of a grammar from a corpus of one or more performances, using a combination of clustering and Markov chains. We first define the basic building blocks for contours of typical jazz solos, which we call "slopes", then show how these slopes may be incorporated into a grammar wherein the notes are chosen according to tonal categories relevant to jazz playing. We show that melodic contours can be accurately portrayed using slopes learned from a corpus. By reducing turn-around time for grammar creation, our method provides new flexibility for experimentation with improvisational styles. Initial experimental results are reported.

1. INTRODUCTION

Jazz improvisation is a form of composition done concurrently with the performance of the music itself. Although the ideal would have no premeditation about what will be performed, it is known that jazz musicians do work out and practice vocabulary ideas prior to the actual performance. We are interested in tools that facilitate the construction and recording of such ideas, for purposes of education as well as performance. The present contribution demonstrates that grammars for generating jazz melodies can be learned from performances, in a manner that captures stylistic aspects of the performer. The ability to generate melodies in a given style is expected to have significant tutorial value.

2. RELATED WORK

Use of grammars for creating musical structures has been investigated by Cope [6], Bel [4], Pachet [18], and others. We base our approach on the grammatical representation of Keller and Morrison [14], which seems to provide an adequate basis for generating jazz melodies.

Methods for algorithmic composition have been surveyed extensively by McCormack [17] and by Papadopoulos and Wiggins [19]. Our work combines and extends some of the ideas they discussed, including the combined use of grammars and machine learning. Dubnov, Assayag, Lartillot, and Bejerano [8] used probabilistic and statistical machine learning methods for musical style recognition. Eck and Lapamle [9] investigated automatic composition and improvisation with neural networks, and Cruz-Alcazar and Vidal-Ruiz [1] developed a method for learning grammars to model musical style.

The idea of melodic contour for abstraction has been used for analysis purposes. Kim, Chai, Garcia and Vercoe [15] used contours for musical classification and querying, and Chang and Jiau [5] investigated musical contour with applications to extracting repeating figures and themes from music. In addition, De Roure and Blackburn [7] proposed melodic pitch contours for content-based navigation of music. We incorporate the melodic abstraction of contours and slopes by utilizing them as the building blocks of a grammar for compositional purposes.

Kang, Ku, and Kim [13] used a graphical clustering algorithm for extraction of melodic themes. The clustering portion of our model serves a similar function. Verbeurgt, Dinolfo, and Fayer [20], among others, used Markov models as a means for composition by learning transition probabilities between patterns. Ames [2] dealt with different-sized Markov chains of notes. Our approach utilizes Markov chains to determine transition probabilities between *clusters* of different types of melodic themes. We believe this is helpful in providing additional flexibility and fluidity needed to generate jazz melodies.

3. ABSTRACT MELODIES

Although a given jazz performer might not be aware of how he or she does improvise, it seems reasonable to say that ideas of what one is able and willing to play can be captured in some form of grammar. At the very least, if a student has memorized a finite set of "licks" (melodic fragments), it is obvious that that set could be described by an *ad hoc* grammar. A grammar that is too *ad hoc*, however, would tend to generate only very predictable, and thus eventually uninteresting, melodies. It is important then that melodic ideas be abstracted so as to enable the replacement of certain elements with others to continually produce novel output. If the abstraction is too coarse-grain, though, the melody may lose coherence.

Our grammatical approach for jazz melodies attempts to strike a balance between novelty and coherence by augmenting the five note categories of [14] that correspond to concepts in jazz playing. These categories are instantiated probabilistically and also in observance of other constraints, such as range considerations, at generation time. Each category, as given in Table 1, has a corresponding terminal symbol in the grammar and four of them show as different note head colors on the staff, for explication purposes.

Symbol	Color	Meaning
C	black	Chord tones of the current **Chord**
L	green	**Color tones,** complementary tones sonorous with the current chord
A	blue	Tones that chromatically **approach** one of the above
——	red	Neither C, L, nor A
S	——	Tones in a **scale** that corresponds to the chord
X	——	Arbitrary tone
R	——	Rest

Table 1. Note categories used in grammar terminals

A terminal symbol of the grammar is formed by following a category symbol by a numeric duration. For example, A8 represents an approach tone of duration one eighth-note, C4 a chord tone of duration one quarter-note, L4/3 a color tone of a quarter-note *triplet*, S4. a *dotted* quarter-note, R2 a half-note rest, etc. We think of a sequence of terminal symbols in the grammar as being an *abstract melody*, in the sense that multiple melodies will fit the sequence when the categories are instantiated to corresponding tones. Another advantage of such melodic abstractions is that they can be instantiated over any chord progression, even for chords of different quality, such as major vs. minor.

We extend these individual note categories with "macro" concepts dealing with sequences of notes in certain patterns. Although more general macros are possible, our current work focuses on a single macro concept, called a *slope*. Each slope has two numeric parameters, indicating the minimum and maximum interval between notes in the sequence going in the ascending direction. Negative numbers indicate the *descending* direction. S-expressions [16] are used to provide grouping of notes in a sequence, and for hierarchy, when necessary. For brevity, we will represent "slope" by Δ in this paper. For example, (Δ 1 2 S8 S8 S8) would indicate an ascending group of three eighth notes that are scale tones, with each at least 1 semitone and at most 2 semitones pitch separation. Similarly, (Δ -3 -4 C4 L4 C4) indicates a descending series of a chord, color, and chord tone, with a minimum separation of 3 and a maximum separation of 4 semitones. More generally, it is not always possible to obey the constraints of both the slope and the note category, so sometimes we must *relax* one or the other, as described later.

Slopes may be concatenated to provide *contours*. Each note symbol in a slope indicates a direction and a range of possibilities for the interval from the previous note. We break melodic lines into strictly ascending, descending, or stationary segments and define the slope of a segment by the minimum and maximum intervals between notes in the segment. Such a definition of slope allows us to represent many common jazz idioms. For example, consider the bebop idiom of an *enclosure* [3], wherein a chord tone is approached by notes above and below.

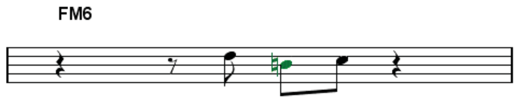


Figure 1. Example of an *enclosure*

Using slopes, we represent an abstraction of the melody in Figure 1 as the S-expression:

(R4 R8 L8 (Δ -3 -4 S8) (Δ 1 2 C8) R4).

Following two rests, we have an eighth note color tone followed by a scale tone three to four half steps down, a chord tone one to two steps up, and finally a quarter note rest. Note that we abstract only pitches, while rhythms are captured exactly.

Our notation for chords follows jazz lead sheet abbreviations, as given in Table 2.

Symbol	Meaning
M	major
m	minor
7	dominant seventh, if by itself
6	added sixth

Table 2. Jazz chord symbols used above the staff

In addition to short idioms, we can capture larger selections such as the line in Figure 2 from Red Garland's solo on "Bye Bye Blackbird" [10]. We represent an abstraction of the melody in Figure 2 with another S-expression:

```
(R8 C8  (Δ -9 -9 A16)
        (Δ 1 3 C16 C16 C16 C8)
        (Δ -12 -12 C8)
        (Δ 1 4 C8 A8)
        (Δ -4 -1 L8 C8 C8 A8 C8)
        (Δ 12 12 C8)
        (Δ -12 -2 C8 C8))
```

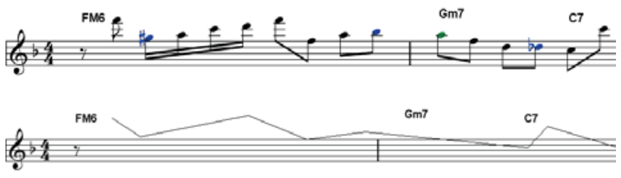
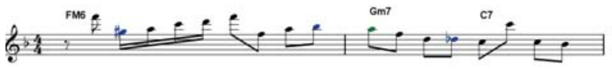


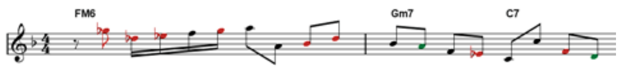
Figure 2. A melody line and its *slope* representation

Notes such as the G# in the first measure begin an ascending segment and so have only one interval from which to choose a minimum and maximum slope. In such cases, we found that relaxing the bounds by a half step in each direction yielded better results. Consequently, we relax (Δ -9 -9 A16) to (Δ -8 -10 A16) before instantiating to a melody. Since chord tones play the most significant role in shaping the melody, we weight chord tones higher than slope bounds, but for note categories other than chord tones, we do not.

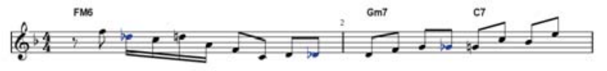
Figure 3 demonstrates several new licks generated from our representation of Red Garland's melody.



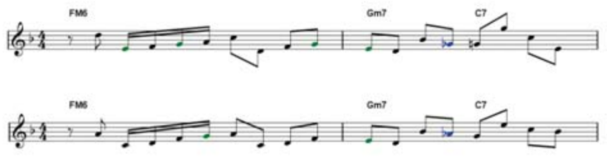
(a) Original melody



(b) Generation using only contours



(c) Generation using only note categories



(d) Two generations using both contour and categories

Figure 3. Original melody vs. melody generation methods

4. GRAMMATICAL INFERENCE

Grammatical inference (GI) algorithms attempt to define the rules of a grammar for an unknown language through analysis of a training dataset. The data can contain both positive sample sets (strings in the language) and negative sample sets (strings that should not be accepted), though we use only positive sample sets, *e.g.* performances from an artist that we want to imitate.

The idea of grammatical inference is to determine a set of rules that will produce strings similar to those in the training data. The grammar, then, should generate a set of strings containing everything in the training data and nothing that differs greatly from the data.

Cruz-Alcazar and Vidal-Ruiz [1] applied three GI algorithms to automatic composition of melodies in Gregorian, Bach, and Joplin styles, achieving the best results with the Gregorian melodies, in which they classified twenty percent of composed melodies as *very good*, which they define as able to be "taken as an original piece from the current style without being a copy or containing evident fragments from samples." We strove toward the same definition of *very good* solos.

We experimented with several methods for extracting grammar rules from training data, including extraction by phrases, with a phrase defined as a section of a solo starting after a rest and ending with a rest. Given the variable length of phrases and the difficulty of recombining phrases into a solo of a specified size, we settled on breaking melodies into *time windows* of a predefined length. After choosing two parameters, the number of beats per window and the number of beats by which to slide the window, we collect all melodic fragments of a certain length in a corpus and associate one grammar terminal for each abstract melody of the given length. We found that, among fragments of between 1 and 8 beats, 4 beat fragments achieved the best balance between originality and continuity. All songs in our training data are in 4/4 time, so we are effectively collecting abstractions for each measure in the training data.

5. MARKOV CHAINS

Once we have gathered the abstract melodies that will make up our generated solos, we combine them into full solos by implementing the equivalent of a Markov chain into the grammar. Markov chains represent a system with a sequence of states, using conditional probabilities to model the transitions between successive states. An n-gram Markov chain uses probabilities conditioned on the previous n-1 states. In our work, sets of abstract melodies serve as the states in the Markov chain; given a starting melody, we add the next phrase based on a list of transition probabilities from the first measure.

6. CLUSTERING

To construct a Markov chain with meaningful transition probabilities, we need a reasonable number of data points for each state. Before building the transition matrix, we group similar abstract measures together using the k-means clustering algorithm [11]. We then collect statistics on which clusters follow other clusters in the corpus and build our table of probabilities accordingly, using clusters as states of the Markov chain. To compose new solos, we first generate a sequence of clusters from the grammar and then randomly select representatives from clusters.

Clustering algorithms represent data as points in an n-dimensional plane and group points together through some distance metric. We base our cluster analysis as a Euclidean distance measure on 7 parameters:

1) Number of notes
2) Location of the first note struck within the window
3) Total duration of rests
4) Average maximum slope of ascending or descending groups of notes
5) Whether the window starts on or off the beat
6) Order of the contour (how many times it changes direction)
7) Consonance

We assign a "consonance" value to a measure based on the note categories. For each note, we add to the consonance value a coefficient for the note category multiplied by the length of the note. For example, typical coefficients are 0.8 for a chord note, 0.6 for an approach note, 0.4 for a color note, and 0.1 for other notes.

Given a parameter *k* for the number of clusters, we use the *k-means algorithm*, which selects *k* points as cluster centers and then begins an iterative process given by the following two steps:

1) Assign each data point to the nearest cluster center.
2) Re-compute the new cluster centers.

These steps are repeated for some number of iterations or until few enough data points switch clusters between iterations. Figure 4 shows three representative 1-measure melodies that the algorithm clustered together in a corpus of Charlie Parker solos.

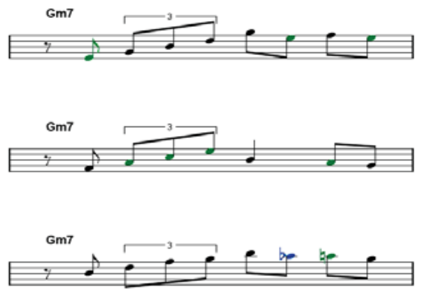


Figure 4. Three representatives from the same cluster

The top line of Figure 5 shows a 2-measure melody and the result of choosing two measures, one from the cluster for the first measure, and one for the cluster for the second.

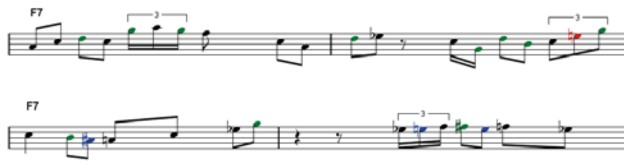


Figure 5. A 2-measure melody (top) and a melody synthesized from clusters corresponding to each of the measures (bottom)

7. IMPLEMENTATION

Our ideas and methods were implemented as a learning extension of the solo generation functionality of the open-source software tool Impro-Visor [12], implemented in Java. Both the executable and source code for the results presented here will be made available publicly.

Table 3 shows a simple grammar with 3 clusters inferred from a corpus of Charlie Parker solos. Each non-terminal symbol has an integer argument indicating the number of measures to be filled. Non-terminals C0, C1, and C2 represent the states of the Markov chain. Due to space limitations, we have only included one of the rules for each of Q0, Q1, and Q2. We have transcribed our implementation's notation from S-expressions to more conventional grammar rules for readability.

Production Rule	π
P(0) → ()	1
P(Y) → Start(32) P(Y-32)	1
Start(Z)→ C0(Z)	0.23
Start(Z) → C1(Z)	0.25
Start(Z) → C2(Z)	0.52
C0(0) → ()	1
C1(0) → ()	1
C2(0) → ()	1
C0(Z) → Q0 C0(Z-1)	0.24
C0(Z) → Q0 C1(Z-1)	0.24
C0(Z) → Q0 C2(Z-1)	0.52
C1(Z) → Q1 C0(Z-1)	0.18
C1(Z) → Q1 C1(Z-1)	0.28
C1(Z) → Q1 C2(Z-1)	0.54
C2(Z) → Q2 C0(Z-1)	0.25
C2(Z) → Q2 C1(Z-1)	0.24
C2(Z) → Q2 C2(Z-1)	0.51
Q0→((Δ 0 0 R2 R4 R8 C16/3)(Δ1 1 A16/3 L16/3)	0.33
Q1→((Δ 0 0 C8)(Δ -9 -9 C8)(Δ 2 3 C8 G4+8 R4))	0.33
Q2→((Δ 0 0 C4/3)(Δ 1 2 L4/3 A4/3)(Δ-7 -1 C4/3 G4 C8/3))	0.33

Table 3. Probabilistic grammar embedding a Markov chain, with π being the probability of using the rule, given the left-hand side

For brevity, the rules for expanding the individual clusters are not shown. These expand into sequences of slope specifications of the sort that were described in section 3. This grammar generates 32-measure abstract melodies, but we can specify any number of measures to generate, and the grammar will adjust accordingly. Once we have an abstract melody, we then generate a real melody by randomly selecting an initial note of the specified note category and then filling in the rest from slope and note category constraints. Figures 7 and 8 show two solos created by our approach, intended to be in the style of John Coltrane and Charlie Parker respectively. Each is based on a grammar learned from a corpus of solos from the respective artists.

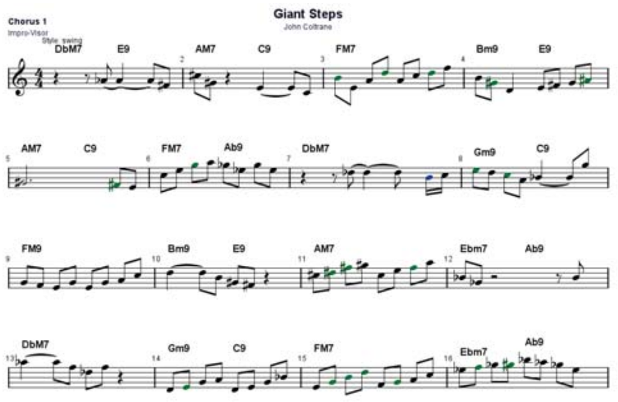


Figure 7. Generated Coltrane-style solo for "Giant Steps"

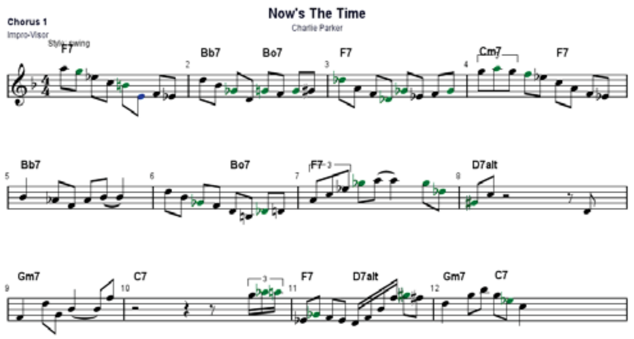


Figure 8. Generated Parker-style solo for "Now's the Time"

8. QUALITATIVE RESULTS

For short solos, we found that our algorithm's compositions usually sound like a capable jazz soloist and occasionally like a convincing imitation of an artist. For short solos of 4 to 8 bars, results were regularly very good and could be easily mistaken for the original artist, but longer solos tended to lack a sense of direction. Four-gram Markov chains produced longer coherent passages than bigram and trigram models and were able to generate very good 12 or 16 measure solos about 25 percent of the time. Given our lack of a large data set (our largest set was about 400 measures of Charlie Parker solos), higher order n-grams gave no more information than the 4-gram model. We hypothesize that with a much larger dataset, higher n-gram models would yield more coherent solos of longer duration.

9. EXPERIMENTAL RESULTS

To measure our method's effectiveness at style emulation, we set up an experiment to determine whether or not test subjects could match the styles of three prominent jazz trumpet players with solos composed in the style of each player. We inferred grammars for Clifford Brown, Miles Davis, and Freddie Hubbard from 72 bars of solos from each. We then played for the subjects one clip from each artist and one clip generated from each grammar, with each computer solo generated over the same tune ("Bye Bye Blackbird"). Without revealing the names of the artists, we asked the subjects to match the artists from the computer-composed solos with the human players. We also asked subjects to qualitatively indicate how close the resemblance was by "Not close," "Somewhat close," "Quite close," or "Remarkably close."

Out of 20 test subjects, 95 percent correctly identified Clifford Brown, 90 percent identified Miles Davis, and 85 percent identified Freddie Hubbard. Of the same subjects, 85 percent correctly matched all 3 solos. All subjects characterized the resemblance to the original artists as either "Somewhat close" or "Quite close," with 9 votes to "Somewhat close," 10 to "Quite close" and 1 unable to decide. On a scale of 1 to 10, 50 percent ranked their own musical knowledge between 2 and 5, and 50 percent between 6 and 9.

10. FUTURE WORK

A more convincing test of our method would be an experiment to determine whether listeners, particularly jazz musicians, can tell the difference between a solo generated by a learned grammar and a human composed solo. Improvements to both our musical representation and our algorithm could be made to achieve good results in such a test.

In terms of musical representation, we currently determine appropriate scales (and note categories) only by one chord. Some chords fit into several keys though, so we could make better scale choices by looking at adjacent chords. Also, examining more specific information about notes in conjunction with the note categories, such as interval from the root of a chord, could prove to be beneficial.

The greatest weakness of our generated long solos is their lack of global structure. A more conclusive evaluation of the effectiveness of n-grams for global structure could be done given a sufficiently large data set. Another approach that could be explored is to infer the high level structure of a generated solo on a particular solo from the training set. We could represent an outline for a solo with a sequence of clusters and create new solos based on the outline by choosing different cluster representatives than were in the original.

11. CONCLUSIONS

The ability of our method to generate solos that sound similar to the artist from the training data, yet distinct from any particular solo, shows that our method of data abstraction is effective. The combination of contours and note categories seems to balance similarity and novelty sufficiently well to be characterized as jazz. In addition, clustering appears to be a workable algorithm for grouping fragments of melodies. Markov chains were effective in structuring solos, however, additional global structure is desirable for providing intra-solo coherence.

12. ACKNOWLEDGMENT

This work was supported by grant 0753306 from the National Science Foundation.

13. REFERENCES

[1] Pedro P. Alcazar and Enrique Vidal-Ruiz, “Learning Regular Grammars to Model Musical Style: Comparing Different Coding Schemes”, *Proc. 4th ICGI*, 211-222, 1998.

[2] Charles Ames, “The Markov Process as a Compositional Model: A Survey and Tutorial”, *Leonardo*, **22** (2), 175-187, 1989.

[3] David Baker, *How To Play Bepop 1*, Alfred Publishing Co., Inc., Van Nuys, CA, 1988.

[4] Bernard Bel, “Pattern Grammars in Formal Representations of Musical Structures”, *Proc. 11th Int’l Joint Conference on Artificial Intelligence, Workshop on AI & Music*, August, 1989.

[5] Chuan-Weng Chang and Hewijin Christine Jiau, “Extracting Significant Repeating Figures in Music by Using Quantized Melody Contour”, *Proc. Eighth IEEE Int’l Symposium on Computers and Communication* (ISCC’03), 2003.

[6] David Cope, “Computer Modeling of Musical Intelligence in EMI”. *Computer Music Journal*, **16** (2), 1992.

[7] David C. De Roure and Steven G. Blackburn, “Content-based navigation of music using melodic pitch contours”, *Multimedia Systems*, **8** (3), October, 2000.

[8] Shlomo Dubnov, Gerard Assayag, Olivier Lartillot, and Gill Bejerano, “Using Machine-Learning Methods for Musical Style Modeling”, *Computer*, **36** (10), 73-80, October, 2003.

[9] Douglas Eck and Jasmin Lapalme, “Learning Musical Structure Directly from Sequences of Music”, *Tech. Rept. 1300, Universite de Montreal DIRO*, 2008.

[10] Red Garland, on “Miles Davis in Person, Friday Night at the Blackhawk, San Francisco”, Columbia Records, 1961.

[11] J. A. Hartigan and M. A. Wong, “Algorithm AS 136: A *k*-Means Clustering Algorithm", *Applied Statistics*, **28** (1), 100–108, 1979.

[12] Impro-Visor, Jazz Improvisation Advisor, `http://www.impro-visor.com`.

[13] Yong-Kyoon Kang, Kyong-I Ku, Yoo-Sung Kim, “Extracting Theme Melodies by Using a Graphical Clustering Algorithm for Content-Based Music Information Retrieval”, *Proc. 5th East European Conference on Advances in Databases and Information Systems*, 84-97, 2001.

[14] Robert M. Keller and David R. Morrison, “A Grammatical Approach to Automatic Improvisation”, *Proc. Fourth Sound and Music Conference*, Lefkada, Greece, July, 2007.

[15] Youngmoo E. Kim, Wei Chai, Ricardo Garcia, and Barry Vercoe, “Analysis of a Contour-based representation for Melody”, *Proc. Int’l Symposium on Music Information Retrieval*, 2000.

[16] John McCarthy, “Recursive functions of symbolic expressions and their computation by machine”, *Comm. ACM*, **3** (1), 184-195, 1960.

[17] Jon McCormack, “Grammar-Based Music Composition”. In Stocker et al, eds. *Complex Systems 96: from local interactions to global phenomena*, 321-336. IOS Press, 1996.

[18] François Pachet, “Surprising Harmonies”, *Int’l Journal on Computing Anticipatory Systems*, 1999.

[19] George Papadopoulos and Geraint Wiggins, “AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects”, in *AISB Symposium on Musical Creativity*, 1999.

[20] Karsten Verbeurgt, Michael Dinolfo, and Mikhail Fayer, “Extracting patterns in music for composition via Markov chains”, *Proc. 17th International Conference on Innovations in Applied Artificial Intelligence*, 1123-1132, 2004.

A SYSTEM FOR MUSICAL IMPROVISATION COMBINING SONIC GESTURE RECOGNITION AND GENETIC ALGORITHMS

Doug Van Nort, Jonas Braasch, Pauline Oliveros
Rensselaer Polytechnic Institute
{vannod2,braasj,olivep}@rpi.edu

ABSTRACT

This paper describes a novel system that combines machine listening with evolutionary algorithms. The focus is on free improvisation, wherein the interaction between player, sound recognition and the evolutionary process provides an overall framework that guides the improvisation. The project is also distinguished by the close attention paid to the nature of the sound features, and the influence of their dynamics on the resultant sound output. The particular features for sound analysis were chosen in order to focus on timbral and textural sound elements, while the notion of "sonic gesture" is used as a framework for the note-level recognition of performer's sound output, using a Hidden Markov Model based approach. The paper discusses the design of the system, the underlying musical philosophy that led to its construction as well as the boundary between system and composition, citing a recent composition as an example application.

1 INTRODUCTION

In the context of free improvisation, the language that performers speak to one another and to the audience is developed throughout the course of a performance as well as rehearsal, listening to all facets of the sound that each player produces. Timbral and textural sound features become strong indicators of the musical form, and further it is the *shape* and direction of these qualities through which performer's speak to one another, expressing their *intention* for the future as much as their creation of the present moment or reaction to the past.

With this in mind, we have developed an interactive system for musical improvisation that analyzes the sonic content of performers, recognizes the nature of the sonic contours being produced in real time, and uses this information to drive a genetic algorithm. The output of this algorithm may be mapped to sonic or visual processes, creating a feedback loop and interplay between performer, machine recognition and a directed evolutionary process.

THIS WORK WAS SUPPORTED BY GRANT NUMBER 0757454 FROM THE NATIONAL SCIENCE FOUNDATION (NSF).

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

This particular choice of system design has arisen from our personal experience as improvisers performing together and with various other musicians, and observations made on this mode of musical creation in general. In terms of design constraints or what one might call "demands" of the system, we built around the following features

1. *A focus on timbral as well as textural information*.
The former is clearly a strong building block for improvisers defining their own performance language in concert. The latter is more separable in time than timbre, and in more sound-focused musics such as free improvisation becomes a strong structural element that interplays with larger sonic contours.
2. *Sonic gestural undestanding*.
We define note-to-phrase level sonic shapes as "sonic gestures", and feel that these convey the fundamental sense of musical intention and direction in improvised music. In this way, the refinement of the system focuses on the interplay between the system's response to this "immediate" information and the nature of the output, rather than building recognition of large-scale structural information that is less important in this context. In other words, the focus is shifted in our system to the immediacy of sound awareness.
3. *Continuous, on-line recognition with measure of certainty/uncertainty*.
While many systems exists for recognition of musical timbre, often the interest lies in the out-of-time acts of classifying musical notes, excerpts, passages or pieces in a way that is categorical. In contrast, our work builds an understanding or likely scenario of the type of sonic gesture that is being played, with a continuous degree of certainty about this understanding. In a sense, we are less interested in a musical retrieval than using the *process* of musical retrieval in a way that an improvising performer does, continually updating their expectation.
4. *Novel output from the system that is continuously influenced by performer's sonic gestures*.
Our goal was a system that would continuously produce spontaneous, novel, and what one might call "creative" events at the same relatively low level on which we focus for analysis and recognition. Therefore we explored the use of pro-

cesses that were directed while being under the influence of randomness.

With these design constraints in mind, we have arrived at a system in which continuous recognition of textural and timbrally-focused sonic gestures are recognized with varying degree of probability and confidence. This understanding then directs an evolutionary process that is finally mapped to an appropriately-defined system output. The sound analysis, gestural recognition and search process together are considered as an *agent* that reacts to the musical situation, influencing it by some output which is treated as an application of the agent rather than an inherent part of it.

2 RELATED WORK

There are several well-known examples of interactive systems for improvisation. One of the most prominent and musically successful is George Lewis' Voyager system [8], which converts pitch to MIDI data in order for various internal processes to make decisions depending on the harmonic, melodic and rhythmic content before producing symbolic output to interact with the human performer. Similar work can be see in Robert Rowe's Cypher system [11], which explores musical cognition and theory to form structuring principles based again on analysis of MIDI data. There exist other examples of MIDI/symbolic music content analysis systems, and the reader is directed to [11] as one point of reference.

As noted, our current system differs in that we examine continuous signal-level timbral/textural features to drive system output so that the system adapts to the changing audio content – as in [7] or [9] – with added layers of complexity from sonic gesture recognition and evolutionary processes. Another approach to analysis of sonic gestures was taken in [6] in which parameter curves for pitch, loudness, noisiness, roughness, etc. were extracted, with captured sequences being stored in a database in order to drive synthesized gestures having similar timbral contours. In this way the system is similar to the musical gesture-driven processing of [10], with the added layer that out-of-time gestural inflection drives the system rather than direct online parameters. Our system shares the conviction that sonic gestures are important in human-machine interaction for improvisation. However we differ in that the recognition itself is on-line with our system, and continual adaptation to the *anticipated* gesture is used as an element of the machine intelligence.

Finally, while the evolutionary paradigm has been widely used for algorithmic composition and sound design, there have also been several approaches to using evolutionary algorithms in an improvisational context. Biles' Genjam system [2] used an interactive genetic algorithm (GA) to evolve a system that learns to play jazz music along with a solo human player. The goal is to use the GA as a means to evolve the final state, while our interest is in the GA *process itself* as engaging with performer in improvisation. In a similar spirit is the work of [3], in which pitch values become centers of attraction for a swarm intelligence algorithm, producing a melodic stream that moves about these input values. Our system shares the interest of mutual influence between evolutionary/biological process and performer's sound output, while we focus on dynamics of recognition as an added layer to help guide this process.

3 SYSTEM OVERVIEW

The overall system, depicted in Figure 1, was written in Max/MSP utilizing several custom externals as well as the FTM, Gabor and MnM packages from IRCAM. In the first step, the system continually extracts spectral and temporal sound features. At the same time, onsets and offsets are tracked on a filtered version of the signal, which act as discrete cues for the system to begin recognizing sonic gestures. When such a cue is received, a set of parallel Hidden Markov Model (HMM) based gesture recognizers follow the audio, with the specific number of these being chosen as a product of needed resolution as well as processing power. The recognition continually provides a vector of probabilities relative to a "dictionary" of reference gestures. Processing on this vector extracts features related to maximum likelihood and confidence, and this information drives the fitness, crossover, mutation and evolution rate of a GA process acting on the parameter output space.

3.1 Sound Feature Analysis

The goal of the system is not to recognize a given sound quality absolutely, but rather to differentiate between sounds made by a performer along a continuum in several dimensions. In particular we believe that in the free improvisation context that is our focus, that global spectral features related to timbre are important for an immediate parsing of sound, with further qualitative differences coming from textures that are more separable in time and acting over a larger time scale (e.g. less than 20ms vs. 20-1000ms). Similarly, rather than the specificity of pitch values, the relative strength or "pitchness" of a note becomes important, as well as its register. In light of this we employed features that can be broken down into *global spectral, pitch strength and textural*.

For the first group we extract *Spectral Centroid* and *Spectral Deviation*. The first is a commonly-used feature that has proven to be a strong perceptual correlate of timbral brightness in distinguishing between sounds, while the second provides a useful means of differentiating between spectrally dense or sparse sounds. Deviation is calculated from the second order central moment of the power spectrum.

For the pitch strength features we use the robust Yin model [4], extracting *Frequency*, *Energy*, *Periodicity* and *AC Ratio*.

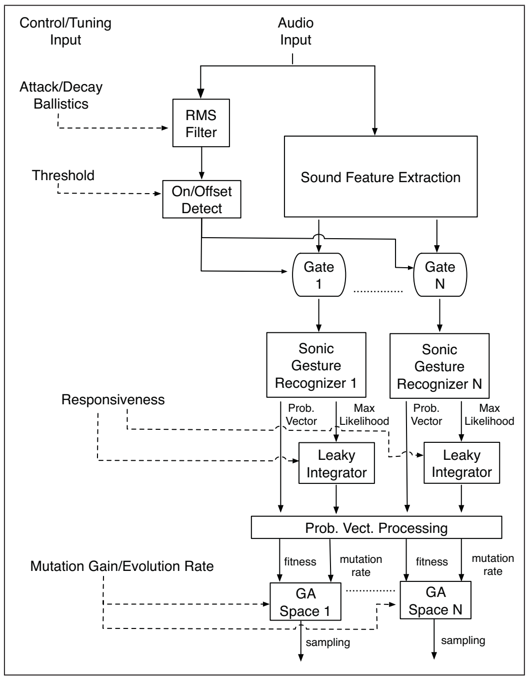


Figure 1. System Overview including feature extraction, recognition and mapping to GA process. The input parameters are used to tune the temporal response of system, and can be used to update this in real-time.

Periodicity provides a degree of "pitchiness", used as a measure of the confidence in the pitch estimate, while AC ratio is a qualitatively different measure of regularity, coming from the ratio of the first two autocorrelation coefficients.

The textural quality of the gestures is examined using the *3rd and 4th Order Moments of LPC Residual*. As was discussed in [5], higher order moments from the excitation describe jitter properties of this signal, which relate to nonlinear frequency modulations of sustained partials and so to textural phenomena. Therefore we extract the residual value by way of LPC analysis, and compute 3rd and 4th order moments on these values in order to differentiate between disparate musical textures. We have found that this measure is very useful in separating voiced and unvoiced content.

3.2 Onset/Offset Detection

Our system uses onsets and offsets as cues to indicate that a relevant event may have begun/ended, with the final decision of whether an event is relevant being determined in the recognition stage. Rather than model onset detection for particular types of events, we employ a straightforward approach that uses tuning parameters for thresholds and response time. Specifically, we utilize the levelmeter object from Max/MSP - which models a VU-style meter - to produce an RMS value of the input sound. The purpose of using this object is that allows for tuning the ballistics of attack/decay times, which strongly influences the onset detection. After extracting the smoothed value, the difference of successive values is taken, and if this difference is greater than a given threshold an onset is considered to have occurred. The same is done in the opposite direction with offset detection. This is represented in Figure 1 as the first two stages of the left-most signal path. When an onset has been detected, it opens up a gate which causes the system to begin searching for sonic gestures that it may recognize from the audio stream. The way that the individual gates close depends either on an offset cue or on other factors related to the recognition as we will discuss in section 3.3.4.

3.3 Sonic Gesture Recognition

Our sonic gesture recognizer is built on the efficient gesture-follower [1] modules from the MnM library developed at IRCAM. This implementation uses an HMM and dynamic time warping to follow as well as recognize gestures in real-time. While there are some trade-offs made with this implementation for efficiency and to allow use with a low number of training examples, we have found it to work well in light of our requirements 2 and 3 as stated in the introduction. These modules work on any data that one can represent in matrix form, leading us to adapt them for our sonic recognition stage. We use all eight of the employed sound features as a singular multidimensional gestural representation, producing a vector that will represent one state in the underlying HMM model, defined using a left-to-right state topology as is standard in applications such as speech recognition.

3.3.1 Gestural Dictionary

The gestural follower requires training examples as a basis for future comparisons. Our interest is not to provide exemplary gestures that performers must later try to recreate - and in this sense perhaps our approach is an outlier for HMM-based recognition. Rather, our goal is to populate a space of gestures that represent a general playing style, in disparate parts of "gesture space". That is, these sonic gestures should be orthogonal in some musical sense, and this is regarded as part of the composition for the system. From experience of using the follower implementation, however, two important considerations arise: the gestures should be roughly the same length and there is a complexity limit (total of number of states in the database) beyond which adding new gestures makes recognition impossible.

3.3.2 Continuous, Dynamic Attention

After the recognizer is trained on a set of gestures, it is ready to accept vectors of the same type for comparison, providing a probability for each member of the gestural dictio-

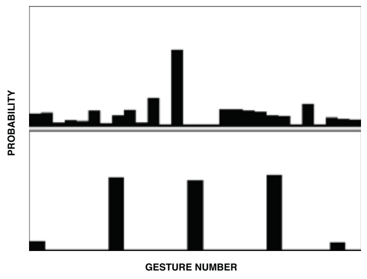


Figure 2. Output probability vectors. Top value shows strong certainty for one gesture while bottom shows confusion over three possible gestures.

nary given the current input. In order to define the temporal boundaries of a gesture from the current input, the recognizer must be explicitly started and stopped, which we initiate with onsets. Most importantly, once a start message has been given the probabilities are updated in real-time for each member of the gesture space, providing a form of *dynamic attention*. This is important in the context of improvisation, where *one's expectation of what a sonic gesture is at any given moment is continually being updated*. We use this information to drive system output, thereby mapping this dynamic attention into action as an engaged improviser would.

3.3.3 Probability Dynamics Processing

From the raw probability values for each gesture, we extract the normalized probability, the maximally-likely gesture and the deviation between the maximum and the few highest values. These latter values each give some indication of how strongly the system believes that the performed gesture is one from the system. Both values are needed to know uniqueness as well as strength of recognition, as illustrated by Figure 2.

At the same time, instantaneous recognition values are not enough in order to usefully map the dynamics of recognition, as the HMM produces sudden changes in the probability vector. For stable gestures this is normally a slow oscillation between perceived values, but occasionally the recognizer will change course abruptly. Therefore, a leaky integrator is applied to the extracted maximum $m_{k,i}$ and deviation values $d_{k,i}$ to create a confidence value defined as

$$C_{n,i} = \delta(m_{n,i} - m_{n-1,i}) \sum_{k=0}^{n} 2^{\frac{-1}{\lambda_{k,i}}}(m_{k,i} d_{k,i}).$$

This represents a building of confidence in a given gesture's likelihood over time n for gesture space i. If the maximum probability value changes abruptly between two members (i.e. if a strong "change of mind occurs") then

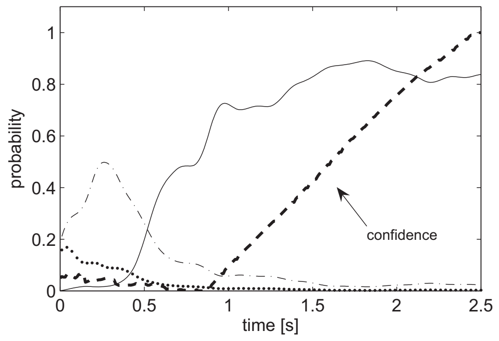


Figure 3. Probability dynamics for three most-likely gestures and related confidence level.

the integrator is cleared by the binary δ function. Otherwise, the value decays smoothly as determined by the response time $\lambda_{n,i}$. Figure 3 illustrates a situation in which uncertain movement keeps the confidence value low until this subsides, when the confidence begins to build.

3.3.4 Parallel Gestural Spaces

Although the follower does use dynamic time warping in order to provide a best guess of the gestural scale, the implementation is limited by the need to have similarly-sized gestures in a given dictionary. Further, it is not trivial to track the beginning and ending of gestures along differing temporal scales in one analysis, as well as to make decisions on what is considered a meaningful gesture as exceptional players often embed one type of gesture within another. In order to examine these different levels of granularity, we create gestural spaces that act on different time scales in parallel, as noted in the diagram of Figure 1. The generality of the diagram reflects the fact that the number may vary depending on computing power and musical context, while we have found that using three different temporal scales has been adequate for our own purposes thus far.

As noted we use onsets as a way to cue the recognition process. When an onset is detected, recognition is triggered using the shortest database of sonic gestures. If the smoothed maximum value stays below a given threshold, then the recognition stop after μ_i seconds, which represents half of the average length of gestures from the ith set. Otherwise, recognition ends after M_i, the maximum time over all gestures in i. If no offset is detected, then the next $N-1$ levels of recognizer immediately begin searching their databases. If the accumulated confidence value $C_{n,i}$ for space i is not above a given threshold by μ_i, then the recognizer is re-set to the beginning. Otherwise it is reset when M_i is reached. For example, Figure 3 represents gestures from a database where the average gestures length is 5 seconds, and μ_i = 2.5. In the initial portion from 0-1 seconds there is deviation between the three main gestures so that confidence $C_{n,i}$ remains low. However it increases rapidly after one gesture asserts itself, easily passing any threshold the user may set. If μ_i were instead 1 second, then the system would likely be reset as confidence would be below any threshold value.

3.4 Gesture-Driven Evolutionary Process

While sonic gesture recognition is an important part of our system, as noted it is ultimately the *process* of recognition and understanding that is central to its musical behavior. The goal is to have a continuous interplay between this process and an output that guides performers in a feedback loop as they in-turn guide this system. We have utilized genetic algorithms as a goal-directed process that moves in a globally predictable direction while maintaining random elements on a local scale. Rather than set the goal a priori as one commonly would in a GA used for optimization purposes, the "goal" changes as a product of the system's gestural recognition and confidence.

The underlying parameter space for our GA implementation is tied to the size of the gestural spaces employed. As noted there is a limit to the size of each gestural space, which we have found to be between 20-30 depending on the time scale of the gestures. At the same time, the required population size for a GA implementation is a product of the problem complexity for optimization purposes. For our application to real-time improvisation, we have found spaces as small as 20 members to be effective in moving towards a perceptible goal.

The reason that we constrain the population size to that of the gestural space is that each member of the gene pool is treated as an ideal output that should arise when a given sonic gesture is believed to be present. Therefore, if a performer "plays into" a certain known gestural type, the system will strongly recognize this and move the GA towards an output that is intended for this type of playing. The way that we achieve this is by mapping the probability for each gesture in a dictionary into the fitness for the corresponding member of the GA population. Thus, for example, in figure 2 if the top probability vector were in steady-state, then the GA would converge towards the member associated with the highly-probable member located in the center of the image.

While belief in a particular gesture causes output to converge towards a particular parameter set, the dynamics of this convergence are determined by the confidence value. As the confidence raises, the probability of mutation (randomization of output parameters at crossover step) as well as the depth of mutation (degree of randomization) decrease. Taking the example from figure 3, the fitness value would oscillate as a function of the three probability curves while the mutation rate would be high due to the low confidence. This would cause the members selected for breeding to move into new areas of the parameter space. After the inflection point – when one gesture begins to dominate and the confidence level starts to rise – the space would move towards the highly-probable level, with less mutation applied at each new generation. The rate of each generation ("rate of evolution"") is context sensitive, being controlled by the confidence level for large-scale gestures or by onsets for small-scale gestures. The relative nature and dividing line of "small" vs. "large" gestures is a product of musical context. The reason for making this distinction is that we have found that short, attack-focused gestures that occur with higher frequency can evolve the space at a reasonable rate, and appear more musical as the change in output is tied to musical events. Longer-scale gestures do not drive the space at a fast enough rate. Further, improvisation and other sound-focused music tends to listen for internal developments "inside" a given sound gesture as it unfolds, so that evolution of system output should not be tied to the initial moment of attack.

As with the gesture-follower, the GA implementation is built up from abstractions written in Max/MSP, while the core GA itself is a C external programmed with operators that are unique to our application. This external is instantiated with messages for population and member size. Each member is a string of float values in the range 0-1. A list of fitness values may be input – one for each member – and a bang message causes a random sampling of members of the population (with selection probability proportional to fitness) for mating/crossover. A simple one-point crossover occurs between members at a random location in the parameter list. This GA implementation is categorical in that each member is tied to a particular member in the corresponding gestural dictionary. Therefore, rather than using a random replacement operator, care must be taken in order to replace the proper parent from a previous generation with its children, where the children inherit the fitness of the parent until the recognition assigns a new value.as

4 CASE STUDY: ACOUSTICS/ELECTRONICS TRIO

The premiere of our system in concert was in the context of a new piece written by the first author for the New York City Electroacoustic Music Festival (NYCEMF) [1] . The piece was written for saxophone, accordion and laptop performer. The electronics capture the sound of the acoustic performers in real-time and transform them in order to define new sonic gestures having their own timbre and texture. The software system is a granular feedback-delay system written by the first author as a performance tool, where input sound may be scrubbed (via gestural control), time-stretched and novel transformations applied through per-grain processing and feedback-delay coupled with larger-scale (e.g. 5-15 sec-

[1] http://www.nycemf.org/ - last accessed on June 8th, 2009.

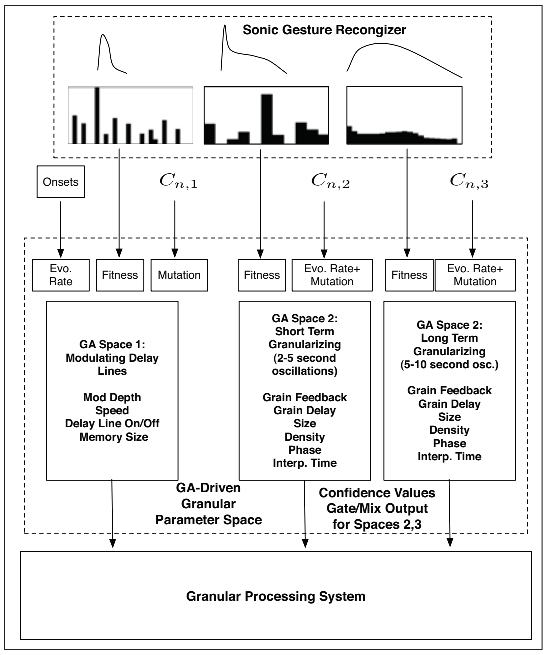


Figure 4. Gesture recognition to GA mapping. Compositional choices are reflected, such as onsets to drive space 1, and choice to mix or gate (depending on section) space 2 and 3 output depending on confidence values.

onds) delay modulations that return as independent gestures rather than transformations on previous material. The structure of the piece was centered around how the sonic gestures and sound processing should co-evolve over time, and how much influence the agent exerted over the human electronics performer. As such, "composing" took on several meanings.

The choice of sonic gestures with which to train the agent was one of the strongest compositional choices. This defined the central gestural types that the performer could then choose to "play into"" or to "play around". The system used 1, 5 and 10-second gestural spaces in parallel. The short gestures focused on a variety in regards to brightness and pitch material, while the 5-second gestures defined different textural values in terms of voiced vs. unvoiced qualities and rough vs. smooth tones. The 10-second gestures defined forms that one might call phrases: differentiating between fast, stunted patterns and slower drones, having different timbral qualities.

A second compositional choice was made in terms of the type of processing to map to each gesture space. The desire was for shorter gestures having sudden attack to lead to a variable number of output gestures that related timbrally to the input while having their own unique gestural character. This was achieved by mapping this smallest gestural space to sound parameters that controlled an array of modulating delay lines each with a unique modulation function, wherein the number of active delay lines, their modulation rate and depth, and memory size (i.e. how far back in time to look for input) were controlled by the agent. Meanwhile, medium and large-scale gestures were mapped into the granular parameters related to grain size, rate, inter-grain phasing, per-grain feedback gain and delay time as well as interpolation time between parameter changes. In this way, the extended gestures with slower attack could be scrubbed by the laptop performer or time-stretched automatically depending on the section, while the internal characteristics of the granular processing evolved at a rate that depended on the anticipated length of the gesture (i.e. whether driven from the 5 or 10 second gesture spaces). This application illustrates one of the great strengths of our system: that a particular gestural type can be mapped into a sound processing parameter set that is tailored to its dynamic sonic character. The nature of the sound processing can then be changed for different temporal scales of sonic gesture. Therefore, rather than content-based processing where the audio quality directly determines the transformation type (as in e.g. [10]), we have added the layer in which the processing type is determined by the audio feature content (indirectly) and type of gestural dynamics (directly) that the system *believes* is occurring, creating an appropriate interplay for improvisation.

5 REFERENCES

[1] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. Wireless sensor interface and gesture-follower for music pedagogy. In *Proceedings of the 7th Int. Conf. on New interfaces for musical expression*, pages 124–129, 2007.

[2] J. A. Biles. Improvising with genetic algorithms: Genjam. In E. R. Miranda and J. A. Biles, editors, *Evolutionary Computer Music*, pages 137–169. Springer, 2007.

[3] T. Blackwell and M. Young. Self-organised music. *Organised Sound*, 9(2):123–136, 2004.

[4] A. de Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *J. Acoust. Soc. Am.*, 111:1917–1930, 2002.

[5] S. Dubnov, N. Tishby, and D. Cohen. Influence of frequency modulating jitter on higher order moments of sound residual with applications to synthesis and classification. In *Proc. 1996 Int. Comp. Music Conference (ICMC 96)*, pages 378–385, 1996.

[6] W. Hsu. Managing gesture and timbre for analysis and instrument control in an interactive environment. In *Proceedings of the 2006 Int. Conf. on New Interfaces for Musical Expression*, pages 376–379, 2006.

[7] T. Jehan and B. Schoner. An audio-driven perceptually meaningful timbre synthesizer. In *Proceedings of the 2001 International Computer Music Conference*, 2001.

[8] G. Lewis. Too many notes: Computers, complexity and culture in voyage. *Leonardo Music Journal*, 10:33–39, 2000.

[9] C. Lippe. A composition for clarinet and real-time signal processing: Using max on the ircam signal processing workstation. In *Proceedings of the 10th Italian Colloquium on Computer Music*, pages 428–432, 1993.

[10] E. Metois. Musical gestures and audio effects processing. In *Proc. of 1998 Int. Conf. on Digital Audio Efects (DAFx 98)*, 1998.

[11] R. Rowe. *Interactive Music Systems: Machine Listening and Composing*. The MIT Press, 1992.

NON-NEGATIVE MATRIX FACTORIZATION WITH SELECTIVE SPARSITY CONSTRAINTS FOR TRANSCRIPTION OF BELL CHIMING RECORDINGS

Matija Marolt
University of Ljubljana
Faculty of Computer and Information Science
matija.marolt@fri.uni-lj.si

ABSTRACT

The paper presents a method for automatic transcription of recordings of bell chiming performances. Bell chiming is a Slovenian folk music tradition involving performers playing tunes on church bells by holding the clapper and striking the rim of a stationary bell. The tunes played consist of repeated rhythmic patterns into which various changes are included. Because the sounds of bells are inharmonic and their tuning not known in advance, we propose a two step approach to transcription. First, by analyzing the covariance matrix of the time-frequency representation of a recording, we estimate the number of bells and their approximate spectra using prior knowledge of church bell acoustics and bell chiming performance rules. We then propose a non-negative matrix factorization algorithm with selective sparsity constraints that learns the basis vectors that approximate the previously estimated bell spectra. The algorithm also adapts the number of basis vectors during learning. We show how to apply the proposed method to bell chiming transcription and present results on a set of field recordings.

1. INTRODUCTION

Bell chiming is a Slovenian folk music tradition that still exists in its original context today. It takes place in the church tower and its original role is strongly connected to Christian religious contexts. Bell chiming combines the signaling, ritual, and musical functions, because it is most often used to call the faithful to mass in a musical way, and at the same time to mark important church holidays. This is how the difference between conventional bell ringing and bell chiming as a more solemn form of playing the bells is established [1].

Slovenian-style bell chiming is performed by the musicians holding the clapper and striking the rim of the stationary bell at regular intervals. The sound is thus not produced by a swinging bell hitting the clapper, but by the clapper, typically held close to the rim, hitting the bell's rim. This gives musicians more control in altering the rhythm, speed, dynamics and accents of individual strikes, as well as leaving out strikes if desired. In the so called "Flying" tunes, one of the bells (usually the largest) is swung with a rope or electronically, and all the other bells, which are stationary, are played by striking the clapper. As a rule, each musician is responsible for playing one bell, and should strike the bell only with its clapper (touching the bell's rim with hands or other tools is not allowed). Another important rule in bell chiming is that two tones can never be played at the same time, but exceptions do occur.

Bell-chiming tunes contrast one another in the method of playing, the number of bells used, and their rhythmic and metric structure. Tunes themselves consist of repeated rhythmic patterns into which various changes, typically dynamic and agogic are included to enliven the performance. Since musicians perform in groups, without the group's consent, only small changes are possible within the time limits allocated to the bell chimer for performing his role. These changes are usually expressed as double strikes, triplets, or pauses [1].

Pioneering work in analysis of bell chiming practices was made by Ivan Mercina in the late 19th and early 20th century, who introduced a numerical notation system and published a repertoire of 243 bell chiming tunes. His work is carried on by researchers of the Institute of Ethnomusicology of the Scientific Research Centre of the Slovenian Academy of Sciences and Arts, who are still actively researching bell chiming practices. Their digital archive of Slovenian folk music and dances Ethnomuse [2] holds a large collection of bell chiming recordings, collected from the 1950s onwards. Only parts of the archive have been manually transcribed and annotated.

In this paper, we present a method for automatic transcription of bell chiming recordings. Automatic music transcription is a difficult problem to solve, although methods are improving constantly; Klapuri and Davy provide an extensive overview of the current state of the art [3]. Unsupervised learning techniques have been used by several authors to perform transcription. Abdallah and Plumbley [4] used sparse coding for transcription of synthesized harpsichord music, while Virtanen used it to transcribe drums [5]. A number of authors use variants of

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

non-negative matrix factorization to transcribe polyphonic music [6-9]. Their methods, however, were devised for music composed of harmonic sounds and are thus difficult to apply to our domain, because the sounds of bells are inharmonic and their tuning not known in advance.

To solve this problem, we propose a two step approach to bell chiming transcription. We first present an algorithm that analyzes a bell chiming recording and estimates the number of church bells and their approximate spectra by using prior knowledge of church bell acoustics and bell chiming performance rules. We then show how non-negative matrix factorization (NMF) can be used for transcription by introducing two extensions to the standard NMF learning algorithm: selective sparsity constraints that take prior knowledge of approximate bell spectra into account, and adaptation of the number of basis vectors during NMF learning.

2. ESTIMATING THE NUMBER OF BELLS AND THEIR SPECTRA

The shape or profile of a bell determines the relative frequencies of its vibrations. The conventional western shape of bells, which stems from the middle ages, tends to give the bell a single dominant pitch. Figure 1 shows the magnitude spectrum of a bell, whose dominant pitch lies at 412 Hz. The names of significant partials of the bell are also shown. These partials are usually the strongest, although (as can be seen) many others exist. The dominant pitch of the bell is defined by relations of three of its significant partials: nominal, superquint and octave nominal [10]. These form a near harmonic series with ratios 2/2, 3/2 and 4/2 resulting in a perceived virtual pitch at approximately half the nominal frequency. Most of the other partials, including the strongest for this bell (tierce), do not belong to this harmonic series.

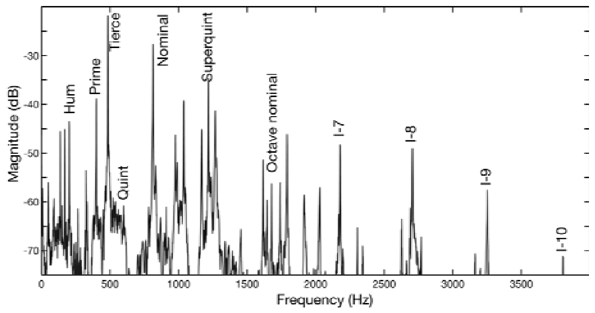


Figure 1. Magnitude spectrum of a bell

An extensive analysis of acoustics of church bells of western shape was made by Hibbert [10]. He showed that a relationship exists between the position of significant partials above the nominal and the ratio of octave nominal to nominal frequency. Hence, we can quite accurately infer the frequencies of these partials if we know the frequencies of the nominal and octave nominal. Frequencies of partials below the nominal do not exhibit strong relationships with the nominal. To assess their positions relative to the nominal, we analyzed a set of 318 church bells and used the means and standard deviations of partial frequencies in this collection.

We estimate the number of bells in a given recording and their spectra by analyzing the covariance matrix **C** of frequency bins of the time-frequency magnitude spectrogram. The elements of the matrix are calculated as:

$$c_{ij} = \frac{1}{n-1}\sum_{k=1}^{n}\left(f_{ik}-\mu_i\right)\left(f_{jk}-\mu_j\right), \qquad (1)$$

where n is the length (number of frames) of the spectrogram, f_{ij} its elements and μ_i the average magnitude of the i-th frequency bin. We exploit the fact that in a bell chiming performance, bells are usually struck many times, but typically not at the same time. Therefore, the amplitude envelope of a bell's partial will be correlated to amplitude envelopes of other partials of the bell, but not to amplitude envelopes of partials of other bells. Groups of bells are usually tuned so that some of their strong partials overlap, but then these will also be at least partially correlated with non-overlapping partials. Figure 2 shows two rows of the covariance matrix of a bell chiming performance including three bells. The top row is placed at the tierce frequency (488 Hz) of the bell from Figure 1 (B1). The bottom row is placed at the nominal frequency of B1 (824 Hz), which coincides with the superquint of another bell with nominal frequency 548 Hz (B2).

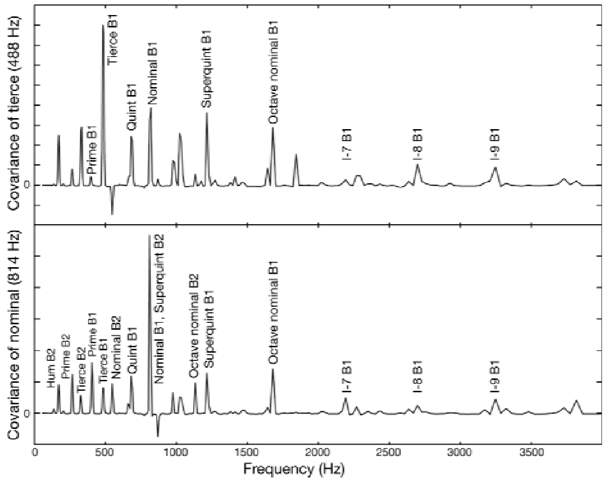


Figure 2. Two rows of the covariance matrix.

The top row clearly shows that the amplitude envelope of the tierce partial of B1 correlates well with amplitude envelopes of other partials of B1. Even though two more bells are present in the recording, amplitude envelopes of their partials are quite different and therefore not correlated with B1's tierce. The bottom row shows that even though partials belonging to two bells share the frequency of 814Hz, partial series of both bells can be discerned from this row of the covariance matrix.

These findings led us to the following algorithm for estimation of the number of bells and their approximate spectra from the covariance matrix:

1. given a time-frequency representation **F** of an audio signal, we calculate its covariance matrix **C**, which contains covariances of amplitude envelopes of all frequency bins;
2. for each row $\mathbf{c}_i$ of the covariance matrix, we find all pairs of peaks (c_{ij} , c_{ik}) that may form a nominal - octave nominal pair of partials. Our analysis of bells showed that the octave nominal typically lies in the range of 1050 to 1350 cents above the nominal, so all pairs within this range are taken into consideration;
3. for each pair of peaks (c_{ij} , c_{ik}), we construct a spectral template B_{ijk}={ (μ_p, σ_p), p=1,...,l } of a bell using the Hibbert's model [10] for calculating the frequencies of partials above the nominal and our own analysis of bells for partials below the nominal. The template contains estimates of frequencies of significant bell partials and their standard deviations;
4. for each template B_{ijk} we calculate its correlation to the row $\mathbf{c}_i$ of the covariance matrix. If it exceeds a threshold T_1, we include the template in the set of all bell templates $\mathfrak{B}$.

To test the bell finding algorithm, we collected a set of 22 bell chiming recordings performed on three to five church bells and manually labeled the nominal frequencies of bells used in performances. We calculated the Constant-Q magnitude spectrogram of each recording by using a maximum window size of 125ms, a step size of 31.25 ms and 20 cent spacing between adjacent frequency bins. To flatten the spectral energy distribution, we scaled the bark scale sub-bands inversely proportional to their variance, as suggested by Klapuri [11]. This especially enhances the amplitudes of higher frequency partials, which is helpful for finding the correct number and spectra of bells with the algorithm described previously, as well as for subsequent NMF learning, which tends to be more sensitive to high-energy observations. Comparable methods for weighting the spectrum were also used by other authors. Virtanen [12] used a weighted cost function in which the observations were weighted so that the quantitative significance of the signal within each critical band was equal to its contribution to the total loudness. Similarly, Vincent [7] used perceptual weights to improve the transcription of low energy notes.

The whitened power spectrum was used as input to the previously described algorithm. We evaluated the algorithm by calculating the precision and recall scores describing how the nominal frequencies of the found bell templates match the manually annotated nominal frequencies of bells. The mean precision-recall curve for all 22 recordings, calculated by varying the threshold T_1, is shown in Figure 3. For our further experiments, we chose to set the threshold T_1 at 0.35. In this way, a high recall value of 0.97 was obtained, meaning that virtually all bells in all recordings were correctly identified, while precision of 0.68 yielded an average of two false positives (superfluous bell templates) per recording.

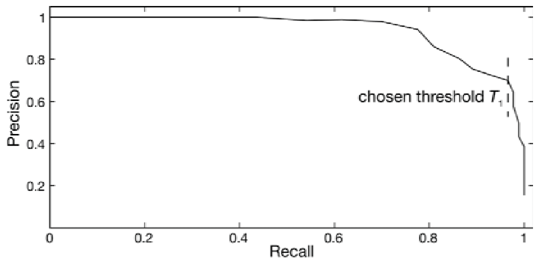


Figure 3. Precision-recall curve of the bell finding algorithm

3. TRANSCRIBING BELL-CHIMING RECORDINGS

When non-negative matrix factorization (NMF) is applied to transcription of polyphonic music, a time-frequency transform is first used to transform the time-domain audio signal into time-frequency space, thus obtaining a time-frequency representation **A**. NMF approximates **A** by two non-negative matrices **W** and **H**, so that:

$$\mathbf{A} \cong \mathbf{WH} \,, \tag{2}$$

where **W** is a matrix of basis vectors and **H** a matrix of coefficients. For music applications, the columns of **W** corresponds most naturally to individual music events (spectra of bells in our case), while the rows of **H** explain how amplitudes of these events change over time. Several efficient implementations of NMF exist in the literature. Our experiments are based on a recent algorithm introduced by Kim and Park [13] that allows sparsity constraints on matrices **W** or **H**.

The naive approach to applying non-negative matrix factorization to transcription of music signals simply by factorizing the magnitude or power spectrum has several shortcomings. As music events overlap in time, there is no guarantee that NMF will separate individual music events into separate basis vectors. A single vector may end up containing partials of several events or only a subset of partials of an individual event. We also have to estimate the number of basis vectors in advance; using too few basis vectors will result in vectors containing several events, on the other hand, too many vectors may result in fragmentation of events over several vectors. Authors have addressed these issues in the past by constraining basis vectors to predefined harmonic templates. Niedermayer [9] used a preset number of fixed basis vectors learned from recordings of individual piano notes and only adapted the matrix of coefficients **H** during learning. Raczynski et al. [8] used a preset number of basis vectors corresponding to individual piano notes, initialized and constrained the vectors to non-zero values only for frequency bins corresponding to perfectly harmonic partial series and learned the weights of these harmonics, as well as the matrix of coefficients. The idea was extended by Vincent [7], who learned the weights of predefined hamonic narrowband partial series belonging to different notes. Vincent also allowed for inharmonicity of the partial series.

The described approaches all assume that the transcribed music signal is composed of a number of individual events (notes) that are composed of (almost) harmonic partials series and thus spectral templates of these events can be constructed in advance. While this works well for piano music, on which all of the mentioned approaches were tested, it cannot be directly applied to our domain. The sound of church bells is inharmonic. We do not know what frequencies the bells are tuned to and even when we do, frequencies of significant bell partials can only be coarsely approximated. Our analysis of bell partials below the nominal showed that their frequencies can vary by as much as 150 cents from an estimated average. In addition, we simply have no data to model all of the bell partials (see unlabelled partials in Figure 1).

We therefore devised an NMF learning algorithm with selective sparsity constraints that uses the spectral templates found by the bell finding algorithm presented in section 2 to initialize and guide the learning process, so that the learned basis vectors approximate the actual bell spectra and as a result, the matrix of coefficients describes the amplitude envelopes of individual bells.

3.1. NMF Learning with Selective Sparsity Constraints

As described previously, non-negative matrix factorization can be used to factor the spectrum into two non-negative matrices **W** and **H**, so that columns **W** corresponds to individual music events, while rows of **H** explain how the amplitudes of these events change over time. We wish to use the set of bell templates $\mathfrak{B}$ obtained by the algorithm described in the section 2 to guide NMF learning, so that the basis vectors of **W** will approximate the found bell templates, while at the same time allowing the algorithm to find the best fit to the actual bell spectra.

To this end, we introduce selective sparsity constraints into NMF learning. Our algorithm is derived from Kim and Park's SNMF/L learning algorithm, which is based on alternating non-negativity constrained least squares and active set method [13]. The algorithm already supports sparsity constraints by imposing L_1-norm based constraints on **H** or **W**. Factorization is calculated by solving:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\left(\left\|\mathbf{A}-\mathbf{WH}\right\|_F^2 + \zeta\left\|\mathbf{H}\right\|_F^2 + \alpha\sum_{i=1}^{m}\left\|\mathbf{w}_i^T\right\|_1^2 \right), \quad \mathbf{W},\mathbf{H} \geq 0, \quad (3)$$

where $\mathbf{w}_i^T$ is the i-*th* row vector of **W**, m the number of rows of **W**, $\zeta \geq 0$ a parameter that suppresses the growth of **H**, while $\alpha \geq 0$ balances the trade-off between accuracy of approximation and sparsity of **W**. Eq. (3) is minimized by iteratively solving two sub-problems using an active set based fast algorithm for non-negativity constrained least squares with multiple right hand side vectors:

$$\min_{\mathbf{W}} \left\| \begin{bmatrix} \mathbf{H}^T \\ \sqrt{\alpha}\mathbf{e}_{1\times k} \end{bmatrix} W^T - \begin{bmatrix} \mathbf{A}^T \\ \mathbf{0}_{1\times m} \end{bmatrix} \right\|_F^2, \ \mathbf{W} \geq 0, \quad (4)$$

and

$$\min_{\mathbf{H}} \left\| \begin{bmatrix} \mathbf{W} \\ \sqrt{\zeta}\mathbf{I}_k \end{bmatrix} \mathbf{H} - \begin{bmatrix} \mathbf{A} \\ \mathbf{0}_{k\times n} \end{bmatrix} \right\|_F^2, \ \mathbf{H} \geq 0, \quad (5)$$

where k is the number of basis vectors, m the number of rows of **W**, n the number of columns of **H**, $\mathbf{e}_{1\times k}\in\mathbb{R}^{1\times k}$ a vector of ones, $\mathbf{0}_{k\times n}\in\mathbb{R}^{k\times n}$ a matrix of zeros and $\mathbf{I}_k$ a $k\times k$ identity matrix. Minimization of equation (4) involves L_1-norm minimization of each row of **W**, thus imposing sparsity on **W**. The strength of this constraint is controlled by the parameter α.

To introduce prior knowledge into NMF learning, we propose a modification of the above approach that selectively enables sparsity constraints only for parts of **W** where no partials are expected. The goal is to constrain **W** to approximate the spectral templates derived from the covariance matrix, while still allowing NMF to learn the best match to the actual bell spectra. Learning improves the estimated partial frequencies, adds partials not included in spectral templates and estimates partial amplitudes.

The bell finding algorithm described in section 2 can estimate the number of church bells in a given recording, as well as their approximate spectral templates ($B_i\in\mathfrak{B}$, i=1,...,k) by analyzing the covariance matrix of the time-frequency representation. The templates are represented as a set of partial frequencies and their standard deviations: B_i={ (μ_{ip}, σ_{ip}), p=1,...,l }. We can thus construct a selectivity matrix **V** as:

$$\mathbf{V} = [v_{ij}]_{m\times k}, \ v_{ij} = \begin{cases} 1, \ \exists p : \left| f_i - \mu_{jp} \right| < T_2\sigma_{jp}, p = 1,...,l \\ 0, \ elsewhere \end{cases}, \quad (6)$$

where f_i is the center frequency of i-th row of the time-frequency representation **A** and T_2 a threshold determining the amount of allowed deviation of a partial from its estimate in $\mathfrak{B}$. The number of columns in the selectivity matrix is equal to the number of bell templates found. Each column of the matrix corresponds to one template and contains ones in places where we expect partials to occur in the time-frequency representation (according to the corresponding template), and zeros elsewhere. The matrix can be used to selectively apply sparsity constraints only for components of **W** where no partials are expected (**V** contains zeros), and leave components where partials should occur unconstrained. To incorporate selective sparsity constraints, we modify the first part of the SNMF/L learning iteration (Equation (4)) to obtain the SSNMF/L algorithm:

$$\min_{\mathbf{w}_i^T} \left\| \begin{bmatrix} \mathbf{H}^T \\ \sqrt{\alpha}\left(1-(\mathbf{v}_i^T)^T\right) \end{bmatrix} \mathbf{w}_i^T - \begin{bmatrix} \mathbf{a}_i^T \\ 0 \end{bmatrix} \right\|_F^2, \mathbf{w}_i^T \geq 0, i = 1,...,m, \quad (7)$$

where $\mathbf{w}_i^T, \mathbf{v}_i^T$ and $\mathbf{a}_i^T$ are i-*th* rows of matrices **W**, **V** and **A** respectively. The matrix **V** acts as a selector that enables or disables sparsity constraints with regard to corresponding bell templates. Equation (7) should be minimized for each row of **W**; for efficiency, we can group together all rows with the same values of $\mathbf{v}_i^T$ and perform non-negativity constrained least squares calculation once for each group.

3.2. Adapting the number of basis vectors during learning

To initialize NMF learning, we must decide on the number of basis vectors to use. We set this number to the number of bells found by the algorithm presented in section 2. The algorithm is tuned to correctly find most of the bells in a recording, with an average of two additional false positives (bells not present in the recording). These false positives are problematic, as partials may be incorrectly attributed to the false-positive vectors during learning. We observed that this usually happens with only a small number of partials, so that the false-positive vectors are a poor match to their corresponding spectral templates. We therefore introduce an additional step to the SSNMF/L learning algorithm, which removes the basis vectors that do not match the templates. The entire learning algorithm is as follows:

1. calculate a set of bell templates $\mathfrak{B}$ using the algorithm described in section 2;
2. set the number of NMF basis vectors k to the number of bell templates found and initialize SSNMF/L learning;
3. repeat minimization of equations (7) and (5) until the change in the normalized Karush-Kuhn-Tucker residual Δ, as defined in [13], falls below a set threshold ε_1;
4. remove all basis vectors that do not match any of the bell templates in $\mathfrak{B}$;
5. repeat points 3. and 4. until the change in Δ falls below a threshold ε_2.

We tested the algorithm on the same set of manually annotated bell chiming recordings used for evaluation of the bell finding algorithm. We also used the same time-frequency representation as previously described. The values of other parameters were set to: $\alpha = 0.0005n$, $T_2 = 2$, ε_1=0.01, ε_2=10^{-9}, $\zeta = 0.01$. For comparison, Table 1 displays results achieved with the standard NMF algorithm without constraints, Kim and Park's SNMF/L with sparsity constraints on **W**, the proposed SSNMF/L with selective sparsity constraints on **W** and finally SSNMF/L with adaptation of the number of basis vectors. The listed precision and recall scores show how well the found basis vectors correspond to the actual bells. Gradual improvement is attained when sparsity and selective sparsity constraints are introduced, while a major boost is achieved by adapting the number of basis vectors during learning. Adaptation is especially helpful in boosting precision, because it removes irrelevant basis vectors, which in turn also boosts recall by correctly distributing the partials amongst the remaining basis vectors.

	precision	recall
standard NMF	0.61	0.83
SNMF/L	0.65	0.82
SSNMF/L without adaptation	0.66	0.86
SSNMF/L with adaptation of the number of basis vectors	0.89	0.89

Table 1. Evaluation of basis vectors learned by different NMF methods

3.3. Transcribing bell chiming recordings

The presented SSNMF/L algorithm factors the time-frequency representation of a bell chiming recording into two non-negative matrices **W** and **H**, so that **W** corresponds to the spectra of bells, while **H** explains how the amplitudes of bells change over time. The transcription algorithm we propose is straightforward. Onsets are first detected with the complex domain algorithm devised by Bello et al. [14]. The matrix **H** is filtered with a fourth order low-pass Butterworth filter with cutoff frequency at 0.25π to smooth the amplitude changes and remove small irregularities. Then, for each onset found, we analyze the 150 ms section of the filtered matrix **H** around the onset and assign the bell with the strongest increase in **H** to the onset.

	num bells	bells found prec. / rec.	onset precision/recall	transcription precision/recall
r1	3	0.75 / 1	0.94 / 0.96	0.93 / 0.95
r2	3	1 / 1	0.91 / 0.96	0.78 / 0.83
r3	4	1 / 1	0.88 / 1	0.79 / 0.90
r4	4	0.75 / 0.75	0.99 / 1	0.71 / 0.72
r5	5	1 / 1	0.87 / 0.92	0.66 / 0.69
r6	5	0.8 / 0.8	0.75 / 0.98	0.45 / 0.59

Table 2. Evaluation of transcriptions of six bell chiming recordings

To test the algorithm, we manually transcribed 20 second excerpts of 6 bell chiming recordings containing three to five bells and evaluated the number of correctly transcribed bells. Table 2 lists results for the 6 recordings. Its columns contain: the number of bells in each recording (2), precision and recall of the match between the basis vectors and the actual bells (3), precision and recall of onset detection (4) and precision and recall of transcription (5).

When all bells are correctly represented by the basis vectors, the average recall is around 0.84. We are satisfied with this result, because we are transcribing real field recordings, which are affected by factors such as poor microphone placement, weather conditions, bell tower acoustics etc. Even though onset detection itself works very well, most of the errors are still made due to indistinctive bell onsets, which may occur because of the aforementioned factors, long bell decay times or change of dynamics by performers. Weak onsets make it hard to determine which bell actually sounded at a given onset, resulting in ignored onsets or incorrectly labeled bells. Because of long bell decay times, which cause most bells to sound throughout a performance, as well as bell tower acoustics, partials may be attenuated or amplified during a performance, which may also lead to false positives or incorrect bell labeling. Accuracy drops sharply when bells are not accurately identified, as wrong bells are assigned to the found onsets.

4. CONCLUSION

The proposed approach to transcription of bell chiming recordings is a good first step into making this part of Slovenian cultural heritage more accessible to interested researchers. There is plenty of room for improvement, especially with the transcription algorithm, but we also plan to extend our researches into automatic extraction of bell chiming patterns, as well as the development of a retrieval system for queries based on bell chiming patterns and recording excerpts.

Acknowledgments. This work was supported in part by the Slovenian Government-Founded R&D project EthnoCatalogue: creating semantic descriptions of Slovene folk song and music.

5. REFERENCES

[1] M. Kovačič "New Contexts, Esthetics, and Transfer in Bell-chiming Tradition," *Slovene studies,* vol. 29, pp. 19-34, 2007.

[2] M. Marolt*, et al.*, "Ethnomuse: Archiving Folk Music and Dance Culture," in *Eurocon 2009*, St. Petersburg, Russia, 2009.

[3] A. Klapuri and M. Davy, Eds., *Signal Processing Methods for Music Transcription*. New York: Springer, 2006, p.^pp. Pages.

[4] S. A. Abdallah and M. D. Plumbley, "Unsupervised analysis of polyphonic music by sparse coding," *Neural Networks, IEEE Transactions on,* vol. 17, pp. 179-196, 2006.

[5] T. Virtanen, "Monaural Sound Source Separation by Nonnegative Matrix Factorization With Temporal Continuity and Sparseness Criteria," *Audio, Speech, and Language Processing, IEEE Transactions on,* vol. 15, pp. 1066-1074, 2007.

[6] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, 2003, pp. 177-180.

[7] E. Vincent*, et al.*, "Harmonic and inharmonic Nonnegative Matrix Factorization for Polyphonic Pitch transcription," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, 2008, pp. 109-112.

[8] S. A. Raczynski*, et al.*, "Multipitch Analysis with Harmonic Nonnegative Matrix Approximation," in *ISMIR 2007, 8th International Conference on Music Information Retrieval*, Vienna, Austria, 2007, pp. 381-386.

[9] B. Niedermayer, "Non-negative Matrix Division for the Automatic Transcription of Polyphonic Music," in *ISMIR 2008, 9th International Conference on Music Information Retrieval*, Philadelphia, USA, 2008, pp. 544-545.

[10] W. A. Hibbert, "The Quantification of Strike Pitch and Pitch Shifts in Church Bells," Ph.D., Faculty of Mathematics, Computing and Technology, The Open University, Milton Keynes, UK, 2008.

[11] A. Klapuri, "A perceptually motivated multiple-FO estimation method for polyphonic music signals," presented at the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics,, New Paltz, USA, 2005.

[12] T. Virtanen, "Separation of sound sources by convolutive sparse coding," in *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, Jeju, Korea, 2004.

[13] H. Kim and H. Park, "Nonnegative Matrix Factorization Based on Alternating Nonnegativity Constrained Least Squares and Active Set Method," *SIAM Journal on Matrix Analysis and Applications,* vol. 30, pp. 713-730, 2008.

[14] J. P. Bello*, et al.*, "On the use of phase and energy for musical onset detection in the complex domain," *Signal Processing Letters, IEEE,* vol. 11, pp. 553-556, 2004.

SHAPE-BASED SPECTRAL CONTRAST DESCRIPTOR

Vincent Akkermans
Faculty of Art, Media & Technology
Utrecht School of the Arts
Utrecht, the Netherlands
vincent.akkermans@student-kmt.hku.nl

Joan Serrà, Perfecto Herrera
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain
{joan.serraj, perfecto.herrera}@upf.edu

ABSTRACT

Mel-frequency cepstral coefficients are used as an abstract representation of the spectral envelope of a given signal. Although they have been shown to be a powerful descriptor for speech and music signals, more accurate and easily interpretable options can be devised. In this study, we present and evaluate the shape-based spectral contrast descriptor, which is build up from the previously proposed octave-based spectral contrast descriptor. We compare the three aforementioned descriptors with regard to their discriminative power and MP3 compression robustness. Discriminative power is evaluated within a prototypical genre classification task. MP3 compression robustness is measured by determining the descriptor values' change between different encodings. We show that the proposed shape-based spectral contrast descriptor yields a significant increase in accuracy, robustness, and applicability over the octave-based spectral contrast descriptor. Our results also corroborate initial findings regarding the accuracy improvement of the octave-based spectral contrast descriptor over Mel-frequency cepstral coefficients for the genre classification task.

1 Introduction

Music information retrieval (MIR) studies processes, systems and contexts for automatically acquiring information about music from large collections [8]. It plays an increasingly important role in a society that moves towards a freely accessible abundance of recorded music. The main audiences benefiting from MIR research are end-users, industry bodies and academics. Users have easier and personalized access to their collections, the industry employs these methods in the production process from creation to distribution, and researchers are able to discover new patterns in large corpora of data [3].

Content-based MIR methods extract information from the music itself rather than from any supplied meta-data. One of the prototypical tasks in content-based MIR is the automatic classification of a song, in the form of an audio signal, into a music genre [1, 3, 11, 12]. The octave-based spectral contrast (OBSC) is a descriptor specifically designed for this task [6, 14]. The spectral contrast of a sub-band in a signal can be seen as a measure of the signal's difference to white noise [10, 14]. In addition, the concept of spectral contrast is also used to enhance sound for hearing impaired people [16]. Because the spectral contrast of an audio signal is based on its timbre it is related to descriptors like spectral centroid, roll-off, flatness, skewness, spread and Mel-frequency cepstral coefficients (MFCCs) [10, 11, 12]. MFCCs were originally developed for use in speech recognition applications and later on proved to be useful for music information retrieval [10, 12]. They provide good discriminative power but can be hard to interpret [9].

In this study the shape-based spectral contrast (SBSC) descriptor is presented. SBSC yields a significant increase in accuracy, robustness, and applicability over OBSC. For evaluation, SBSC is compared to OBSC and MFCCs in terms of discriminative power and MP3 compression robustness. Discriminative power is evaluated by measuring their accuracy on different combinations of data sets and classifiers for the automatic genre classification task [1, 12]. We study the robustness of the descriptors at different MP3 encodings as it is not so common in the literature to test MP3 robustness [5, 13]. A descriptor is considered to be robust when its values do not change significantly while the audio it describes is encoded at different levels of MP3 compression. Finally, information overlap between MFCC and spectral contrast is briefly investigated.

The structure of this document is as follows. In section 2 we briefly summarize OBSC. Section 3 presents the SBSC descriptor and section 4 the evaluation methodology. In section 5 the results are presented and we conclude our study in section 6.

2 Octave-based Spectral Contrast

The OBSC descriptor was introduced by Jiang et al. in [6]. It describes the ratio between the magnitudes of the peaks and valleys within sub-bands of the frequency spectrum. This way, the relation of harmonic to non-harmonic frequency components of each sub-band is

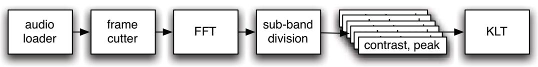


Figure 1. Spectral contrast descriptor general block diagram

reflected.

This feature has been proven useful for genre classification [6, 14]. The two aforementioned studies explain in detail how the descriptor is calculated. In short, the audio signal is loaded and cut into frames with an overlap of 50% (figure 1). Then, the spectral data resulting from an FFT of every frame is divided into 6 octave-scaled sub-bands (with boundaries at 0, 200Hz, 400Hz, 800Hz, 1.6kHz, 3.2kHz, and 8kHz). For each band k, the magnitudes of the FFT bins x are sorted into descending order and the peak P_k and valley V_k values are subsequently calculated by averaging a percentage of the highest and lowest magnitudes. The ratio between P_k and V_k, defined as contrast C_k, and V_k itself comprise the feature vector $OBSC_k$ of a single frame, with a dimensionality of twice the number of sub-bands:

$$OBSC_k = [C_k, V_k]\,, \tag{1}$$

where

$$C_k = \log \frac{P_k}{V_k}, \tag{2}$$

$$P_k = \frac{1}{\alpha N_k} \sum_{i=1}^{\alpha N_k} x_{k,i}, \tag{3}$$

and

$$V_k = \frac{1}{\alpha N_k} \sum_{i=1}^{\alpha N_k} x_{k,N_k-i+1}. \tag{4}$$

Here α corresponds to the part of all bins in the band to average over ($0 < \alpha \leq 1$), N_k to the total number of bins in the sub-band, and i denotes the sorted bin index. In [6], α is set empirically to 0.02. Finally, the dimensions of the feature vector of OBSC are decorrelated for each frame by a Karhunen-Loève Transform (KLT) to increase the accuracy. The orthogonal base vectors for the KLT are generated from the averaged covariance matrices of all classes involved in the problem [6].

3 Shape-based Spectral Contrast

The SBSC descriptor is a modification of the OBSC descriptor intended to improve accuracy, robustness and applicability. It does so by employing a diferent sub-band division scheme, an improved notion of contrast, and a different use of the KLT transform.

3.1 Accuracy

OBSC calculates a sub-band's contrast by regarding its valley and peak. However, by including information about the shape of the band's sorted spectrum, a better estimation of spectral contrast can be made. Figure 2 shows two possible shapes of sorted sub-bands. Both possibilities would yield the same C_k value, as P_k and V_k would be the same. However, if we consider noise to be the equal presence of all frequencies, figure 2A intuitively corresponds to a more noisy sound than figure 2B. Accordingly, the shape in figure 2B should result in a higher spectral contrast.

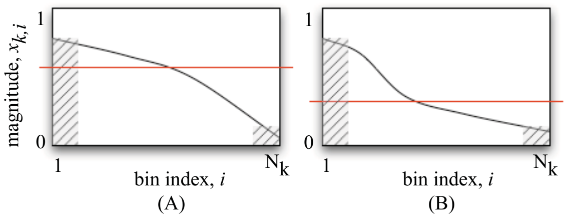


Figure 2. Two possible shapes of sub-bands sorted by magnitude. The horizontal red line indicates the average magnitude of the sub-band.

When we look at the location of the average magnitude in the sub-band relative to the peak and the valley we can better distinguish between both shapes. Accordingly, the contrast C_k (equation 2) could be calculated as:

$$C'_k = \frac{log(P_k/V_k)}{log\mu_k}, \tag{5}$$

where

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} x_{k,i} \tag{6}$$

Equation 5 is equivalent to $log_{\mu_k}(P_k/V_k)$ and expresses the spectral contrast in base μ_k. Through trial and error, the following equation was determined to have similar characteristics but a slightly better accuracy:

$$C'_k = \left(\frac{P_k}{V_k}\right)^{1/log\mu_k} \tag{7}$$

3.2 Robustness

Because the spectral contrast is calculated from the peaks and valleys, MP3 compression, which eliminates masked frequencies, might have a large effect on the robustness of OBSC [4]. When peaks and valleys are averaged over a larger number of bins, eliminated frequencies have a smaller impact and the spectral contrast can be expected to be more robust. The neighbourhood ratio α and the total number of bins in a sub-band N_k are therefore looked at for increasing robustness.

For flexible adjustment of N_k and the number of sub-bands K, a different sub-band division scheme is used in SBSC instead of the original octave-based scheme. A portion of the bins of the FFT analysis is distributed equally among sub-bands while the rest is distributed exponentially:

$$B_k = \left(\frac{(1-s)U}{L}\right)^{k/K} L + \frac{(U-L)sk}{K}, \qquad (8)$$

where B_k denotes the upper boundary of the k-th band, s the portion of the spectrum to distribute equally ($0 \leq s \leq 1$), and U and L the upper and lower bound (in Hz) of the spectrum, respectively. This way, all sub-bands contain enough bins to be stable and the distribution still mimics the non-linear frequency response of the human ear [7]. In all SBSC tests in this study, $s = 0.15$ and $L = 20Hz$ (see section 4.1). For $U = 11kHz$ and $K = 6$, boundaries are 20 Hz, 330 Hz, 704 Hz, 1256 Hz, 2303 Hz, 4729 Hz, and 11 kHz. Parameters K, and U vary from test to test and are set manually.

3.3 Applicability

Instead of applying a KLT based on the averaged covariance matrices of all classes [6] (see section 2), SBSC applies the KLT based on the covariance matrix of each individual song. This does not perform significantly different and has two additional advantages: (a) when either the instances in the data set or the number of categories changes, the average covariance matrix will not have to be recalculated and (b) the descriptor can be used in other applications where no training data set is required (e.g., song similarity, audio fingerprinting, etc.).

4 Evaluation Methodology

We here detail the methods employed for determining genre classification accuracy, evaluating MP3 compression robustness, and studying MFCC and SBSC information overlap.

4.1 Genre Classification Accuracy

The discriminative power of both OBSC and SBSC is compared by measuring their accuracy in a genre classification problem. In addition, we evaluate MFCC (12 coefficients with the zeroth coefficient included) as a baseline. The means and variances of the feature vectors of all frames are used for classification.

Three distinct classifiers are trained and evaluated on two data sets. The classifiers used are decision trees, support vector machines (SVM), and linear logistic regression models (LLR) as implemented by the WEKA software [1] [15]. All classification results are computed on a 10-fold cross-validation scheme and averaged over 10 runs. The 2 data sets used are the following. Data set A is the same data set that was used by Tzanetakis in [12]. It consists of 10 genres with 100 song-excerpts per genre and has a sample rate of 22050 Hz. The genres include blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. Data set B was developed in-house, has a sample rate of 44100 Hz, and consists of 55 full songs for each of 8 genres, which are classical, dance, hip-hop, jazz, pop, rhythm and blues, rock, and speech.

The data set employed by Jiang et al. in [6] had a frequency range of 8 kHz. To check the effect of a bigger frequency range, OBSC is compared to MFCCs at two different frequency ranges, 8 kHz and 11 kHz. Because MFCCs are more accurate at a frequency range of 11 kHz, SBSC is only tested at this frequency range. The SBSC descriptor is also tested with the spectrum divided into both 6 and 9 bands. Finally, MFCCs, OBSC, and SBSC are tested with the delta coefficients included. The descriptors are evaluated at the settings summarized in table 1. For all SBSC tests, $\alpha = 0.4$ and for all OBSC tests $\alpha = 0.1$ (the α value in [6] is 0.02, and is said to perform the same as $\alpha = 0.1$).

Test	Descriptor	Freq. range	Bands
MFCC-A	MFCC	8 kHz	12
MFCC-B	MFCC	11 kHz	12
MFCC-C	MFCC + ΔMFCC	11 kHz	12
OBSC-A	OBSC	8 kHz	6
OBSC-B	OBSC	11 kHz	6
OBSC-C	OBSC + ΔOBSC	11 kHz	6
SBSC-A	SBSC	11 kHz	6
SBSC-B	SBSC	11 kHz	9
SBSC-C	SBSC + ΔSBSC	11 kHz	6

Table 1. Test settings for descriptors.

4.2 MP3 Compression Robustness

The two spectral contrast descriptors and MFCCs are tested for MP3 robustness on a song by song basis. OBSC+ΔOBSC is not tested for robustness due to the expectancy of it to be highly unstable and the considerable time it takes to run the test. The wave files of data set A are compressed at two different compression rates, 192 kb and 64 kb by the Lame MP3 encoder [2]. The descriptors for all three versions of all songs in the data set are extracted. In order to make the descriptors from the compressed and original data comparable in terms of distributions and ranges, we adapted Box-Cox's transformation [2]

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] http://lame.sourceforge.net/

for obtaining uniform distributions for each descriptor attribute between 0 and 1. The transformation is calculated from the uncompressed descriptors and applied to both the uncompressed and compressed descriptors. For each dimension of each song, the absolute difference between the values of the uncompressed and compressed descriptor sets is calculated:

$$D_{h,d,e} = |R_{h,d,e} - Q_{h,d}|, \tag{9}$$

where $D_{h,d,e}$ is the robustness value of the h-th song for the d-th dimension of the e-th encoding, and R and Q denote the descriptor values of the encoded version and the uncompressed version, respectively. We say that the descriptor fails for that particular song, dimension and encoding when $D_{h,d,e} > 0.1$. The percentage of songs that fail per dimension is calculated and the average and maximum are kept as the final robustness error measures.

4.3 MFCC and SBSC Information Overlap

We follow different paths with the aim to obtain converging evidence of the amount of overlap of information between MFCC and SBSC. First, we use several attribute selection techniques on the combined feature vectors MFCC-B and SBSC-A of data set A in order to check the ranking of features or the near-optimal subset that is chosen. The selection methods used are correlation based feature subset selection (CFS), SVM ranking, and individual attribute rankers based on the most frequently preferred indices (infogain, gain ratio, and chi-square test) [15]. The percentage of SBSC attributes, as opposed to MFCC attributes, present in the first and last quartile of the combined ranked attribute list is used as a measure of information overlap. If there is no clear majority of one of the features in the quartiles, this can be interpreted as evidence of overlap.

Secondly, we perform a one-way analysis of variance (ANOVA) on the ranks obtained by the feature selection indices mentioned above, in order to check if there is an effect of the subset type (MFCC versus SBSC) on the index value. If the subset proves not to be significant, then we can consider it as evidence of overlap between them.

5 Results

5.1 Genre Classification Accuracy

Our results of analyzing and testing the OBSC descriptor corroborate the findings of Jiang et al. [6] and West and Cox [14]. In our tests (table 2), for both data sets and all three classifiers, the OBSC descriptor performs better than MFCCs, although the increase in accuracy is not as high as previously reported in [6]. There, on a different data set and using Gaussian mixture models, OBSC performs at 82.3% and MFCCs at 74.1%. The accuracy of our MFCC implementation is tested with a naive Bayes single Gaussian classifier (47.6% for 13 coefficients) and is similar to the one achieved on the same data set in [12] (47% for 10 coefficients).

We can see in table 2 that SBSC's accuracy is higher than that of OBSC and MFCCs for SVM and LLR. For these two classifiers the average relative increase for both data sets is 6.5%. Only with trees OBSC performs better than SBSC, but these accuracies are significantly lower than those achieved with SVM or LLR.

We also see that an increased frequency range has a small and slightly irregular effect on the accuracy (MFCC-A,B and OBSC-A,B, table 2). It raises MFCCs' accuracy for almost every combination of classifier and data set, while OBSC's accuracy only increases for LLR on data set A and trees on data set B.

Test	Data set A			Data set B		
	Trees	SVM	LLR	Trees	SVM	LLR
MFCC-A	41.3	60.0	**61.3**	56.6	77.6	**78.6**
MFCC-B	41.9	60.6	**63.6**	59.7	76.8	**78.6**
MFCC-C	48.2	**71.6**	71.1	63.1	**81.4**	80.1
OBSC-A	47.4	61.6	**62.4**	58.7	82.8	**83.8**
OBSC-B	46.4	61.4	**64.4**	64.2	**81.4**	81.0
OBSC-C	49.0	67.3	**69.0**	61.4	**82.2**	80.7
SBSC-A	45.5	67.3	**68.1**	63.1	**85.7**	85.5
SBSC-B	48.5	67.0	**68.9**	62.3	**86.8**	85.5
SBSC-C	49.9	72.5	**72.7**	65.1	**86.2**	**86.2**

Table 2. Genre classification accuracy (%) for the descriptors tested.

Looking at the SBSC-A and SBSC-B tests we can see that using more than six sub-bands for SBSC does on average only provide a slightly better performance. Including the delta coefficients for SBSC increases accuracy but does not provide the same improvement as the delta coefficients do for MFCCs (an average of 9.8% for MFCC+ΔMFCC and 4.8% for SBSC+ΔSBSC). Including the delta coefficients for OBSC provides a modest average accuracy increase of 3.1%.

In addition to the results presented in table 2, we also studied the effect of different KLT application variants (sections 2 and 3). Decorrelating the dimensions of the feature vector using KLT for each song separately results in a small decrease in accuracy of 0.3%. Not applying KLT results in a significant performance drop (e.g. from 67.3% to 59.7% for SBSC when using SVM).

5.2 MP3 Compression Robustness

In table 3 it can be seen how unstable OBSC is, and how robust the MFCC implementation is in comparison. MFCCs only fail for 0.7% of the songs, while OBSC fails for 13.6% (MFCC-B and OBSC-B, 64 kb, table 3). A higher neighbourhood ratio α, together with the application of equation 8 for sub-band division, results in an increased robustness for the SBSC in respect to OBSC, as it only fails for 1.6% of the songs. The robustness of the most unstable dimensions of SBSC improves significantly from 61.4% to 12.4% of failed songs when compared to OBSC (SBSC-A, 64 kb, table 3).

When the frequency spectrum is divided into 9 bands, SBSC is more unstable (SBSC-B, table 3). This can be attributed to a smaller number of bins in each sub-band and thus a smaller number of bins over which the peaks and valleys are averaged. Apart from the results shown in table 3, it is worth to mention that the delta coefficients for both SBSC and the MFCCs are very unstable with an average failure rate of 75% for 192 kb. One might hypothesize that this is due to the large effect that small changes in the audio have on the delta coefficients. Also, we find that the valley dimensions are more unstable than the contrast dimensions as these are affected most by the MP3 compression.

Test	Error rate at 192kb		Error rate at 64kb	
	mean	max	mean	max
MFCC-A	0.7	8.7	**0.4**	**4.9**
MFCC-B	**0.4**	**3.0**	0.7	6.7
MFCC-C	22.1	87.9	21.8	83.0
OBSC-A	6.5	45.3	9.9	58.6
OBSC-B	5.4	23.2	13.6	61.4
SBSC-A	1.4	4.9	1.6	12.4
SBSC-B	2.2	13.5	3.1	13.4
SBSC-C	19.7	83.7	19.7	79.9

Table 3. MP3 compression robustness error rates (%) for the descriptors tested.

5.3 MFCC and SBSC Information Overlap

The highest scoring subset of SBSC and MFCC attributes is achieved with CFS and has the same accuracy as a subset of only SBSC attributes: 67.3% for SVM (both subsets consist of 24 attributes). We also find that SBSC attributes are predominant in the first quartile of all ranked attributes and that this pattern is reversed in the last quartile, where the predominant attributes are MFCCs (table 4). Table 5 shows that for every ranking method the average SBSC rank is higher than the average MFCC rank. According to ANOVA, SBSC's attributes also rank statistically significantly higher than MFCCs' attributes.

The possible synergistic effect of the combination of both sets is addressed in figure 3, which shows the step by step increase in accuracy when adding more attributes to the selected subset for SVM classification (the order of addition is provided by the chi-square test ranking). MFCCs' accuracy quickly climbs when adding more attributes but increases only slightly after the 10th attribute. SBSC's accuracy increases at more regular intervals and is higher than MFCCs' when using 10 or more dimensions. Combining both sets and using 10 or more coefficients increases the performance above that of MFCCs, but never reaches that of the SBSC attributes. This can be taken as evidence that the description provided by SBSC subsumes and improves classification over that of MFCCs.

With all this accumulated evidence, it seems safe to conclude that the two subsets of attributes are capturing similar aspects of the sound spectra and, as their combination does not increase the classification performance beyond the level attainable by SBSC attributes alone, we should prefer them over MFCCs. However, we cannot say there is a clear overlap of information as SBSC is preferred by all attribute selection methods.

Rank method	SBSC presence in first quartile	SBSC presence in last quartile
SVM	75%	25%
Chi-square	67%	17%
info gain	75%	17%
gain ratio	75%	17%

Table 4. Percentual presence of SBSC attributes in the first and last quartile of a ranked list containing MFCC and SBSC attributes.

Rank method	F	Probability	Average MFCC rank	Average SBSC rank
SVM	5.8	0.0204	29.1	19.9
Chi-square	7.0	0.0110	29.5	19.5
info gain	8.8	0.0048	30.0	19.0
gain ratio	10.4	0.0023	30.5	18.5

Table 5. ANOVA results of attributes' rank number and origin, degrees of freedom is always 1.

6 Conclusion

In this paper, the shape-based spectral contrast descriptor is presented and evaluated. It is based on the octave-based spectral contrast, but takes the mean magnitude of a band into account in order to calculate a more descriptive measure of spectral contrast. Also, it divides the spectrum and applies KLT differently for increased robustness and applicability. SBSC is compared to both OBSC and MFCCs in terms of genre classification accuracy, to test discriminative power, and MP3 compression robustness. OBSC is shown to achieve higher accuracies than MFCCs in the genre classification task, corroborating initial findings of Jiang et al. Moreover, SBSC's outperforms OBSC and MFCCs. When it comes to MP3 compression robustness,

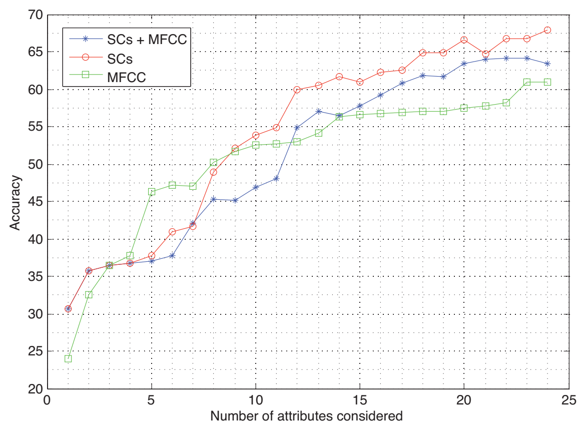


Figure 3. Increase in accuracy while considering an increasing number of attributes.

MFCCs provide the most robust option. However, SBSC represents a significant increase in robustness over OBSC. Including the delta coefficients results in higher accuracies for all descriptors but yields very unstable descriptors. Results obtained from testing information overlap between SBSC and MFCC indicate they capture similar aspects of the sound spectra. However, we found no clear evidence of overlapping information.

7 Acknowledgements

The authors want to thank their colleagues and staff at the Music Technology Group (UPF) for their support and work, specially O. Meyers and N. Wack. The authors also thank G. Tzanetakis for kindly sharing his music genre database. This research has been partially funded by the EU-IP project PHAROS [3] (IST-2006-045035).

8 References

[1] J. Aucouturier and F. Pachet. Representing musical genre: A state of the art. *Journal of New Music Research*, 32(1):83–93, 2003.

[2] G.E.P. Box and D.R. Cox. An analysis of transformations. *Journal of the royal Statistical Society*, 26(2):211–252, 1964.

[3] M.A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. In *Proceedings of the IEEE*, volume 96, pages 668–696, April 2008.

[4] N. Jayant, J. Johnston, and R. Safranek. Signal compression based on models of human perception. In *Proceedings of the IEEE*, volume 81, pages 1385–1422, 1993.

[5] J.H. Jensen, M.G. Christensen, D.P.W. Ellis, and S.H. Jensen. Quantitative analysis of a common audio similarity measure. In *IEEE Transactions on Audio, Speech, and Language Processing*, volume 17, pages 693–703, May 2009.

[6] D. Jiang, L. Lu, H. Zhang, J. Tao, and L. Cai. Music type classification by spectral contrast feature. *Proceedings of the IEEE International conference on Multimedia and Expo (ICME)*, 1:113–116, 2002.

[7] Brian C. Moore. *An Introduction to the Psychology of Hearing, Fifth Edition*. Academic Press, April 2003.

[8] Nicola Orio. Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, 1(1):1–90, 2006.

[9] D.D. O'Shaughnessy. Automatic speech recognition: History, methods and challenges. *Pattern Recognition*, 41(10):2965–2979, 2008.

[10] E. Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Technischen Universität Wien, 2006.

[11] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. *IEEE Signal Processing Magazine*, 23(2):133–141, March 2006.

[12] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, July 2002.

[13] S. Wegener, M. Haller, J.J. Burred, T. Sikora, S. Essid, and G. Richard. On the robustness of audio features for musical instrument classification. In *Proceedings of the European Signal Processing Conference*, 2008.

[14] K. West and S. Cox. Features and classifiers for the automatic classification of musical audio signals. In *Proceedings of the ISMIR conference*, page 531, 2004.

[15] I.H. Witten and E. Frank. *Data Mining, Practical Machine Learning Tools and Techniques, 2nd edition*. Elsevier, 2 edition, 2005.

[16] J. Yang, F. Luo, and A. Nehorai. Spectral contrast enhancement: algorithms and comparisons. *Speech Communication*, 39(1-2):33–46, 2003.

[3] http://www.pharos-audiovisual-search.eu

INTUNE: A MUSICIAN'S INTONATION VISUALIZATION SYSTEM

Kyung Ae Lim
School of Informatics
Indiana University
kalim@indiana.edu

Christopher Raphael
School of Informatics
Indiana University
craphael@indiana.edu

ABSTRACT

We present a freely downloadable program, *InTune*, designed to help musicians better hear and improve their intonation. The program uses the musical score from which the musician plays, assisting our approach in two ways. First, we use score following to automatically align the player's audio signal to the musical score, thus allowing better and more flexible estimation of pitch. Second, we use the score match to present the tuning analysis in ways that are natural and intuitive for musicians. One representation presents the player with a marked-up musical score showing notes whose pitch characteristics deserve closer attention. Two other visual representations of audio overlay a musical time grid on the music data and allow random access to the audio, keyed by musical time. We present a user study involving 20 highly educated instrumentalists and vocalists.

1. INTRODUCTION

One of our most beloved music teachers was a forceful advocate for "facing the music," by which he meant listening to recordings of our playing. As with the first hearing of one's voice on recording, we each wondered "do I really sound like that!?" While sometimes revealing more than we were ready to hear, the long-term effect of this exercise helped us to "hear ourselves as others hear us." Thus armed, we initiate practice habits that, perhaps over many years, move our music-making toward a state we might admire in others.

The "face the music" approach begins by accepting that most of us are not born able to judge ourselves objectively, but can learn to do so when given the proper external perspective. We adopt this approach here, still in the service of music education, though we use visual, in addition to aural, feedback. While a visual representation of audio is necessarily an abstraction, it has the advantage that the observer can "visit" the image according to her will. For instance, she may see a note having a certain undesirable (or desirable) property; find the same trait in another note of the same pitch; formulate a hypothesis of systematic error (or accuracy); and validate or refute this theory on subsequent notes. In contrast, audio data must be digested nearly at the rate it comes into the ear.

SMC 2009, July 23-25, Porto, Portugal

We apply the "face the music" approach to the practice of intonation — the precise tuning of frequencies corresponding to different musical pitches. While good intonation, "playing in tune," is often neglected in the earliest years of musical practice, it is as essential a part of technique as the playing of fast notes or the control of emphasis. Intonation is also central to what some see as the illusion of tonal beauty — that is, for a sound to be beautiful it must commit itself clearly to the "correct" pitch. We introduce a system that allows musicians to visualize pitch in ways that leverage the centuries-long tradition of music notation, and are intuitive to the non-scientist.

The electronic tuner is, without doubt, one of the most widely used practice tools for the classically-oriented musician, thus justifying efforts to improve this tool. The tuner provides an objective measurement of the pitch or frequency with which a musician plays a note, which can be judged in relation to some standard of correctness (say equal tempered tuning at A=440 Hz.). Though the tuner has been embraced by a large contingent of performing musicians, it does have its weaknesses, as follows. The tuner gives only real-time feedback, requiring the user to synthesize its output as it is generated. The tuner takes time to respond to each individual note, making it nearly impossible to get useful feedback with only moderately fast notes. The tuner cannot handle simultaneous notes, such as double stops — this is actually part of the reason the tuner fails on fast notes, since past notes linger in the air, thus confusing the instrument. Perhaps most significantly, the tuner does not relate its output through the usual conventions of notated music, thus hiding tendencies and patterns that show themselves more clearly when presented as part of a musical score. Our program, *InTune* seeks to overcome these weaknesses by presenting its observations in an intuitive and readily appreciated format.

In what follows, we present our system, *InTune*, describing the three different views of audio the program allows, as well as the backbone of score-following that distinguishes our approach from others. We consider other approaches to this problem and place ours appropriately in this context. Finally we present a user study, giving reactions to our effort from a highly sophisticated collection of users. The program was developed in close consultation with two professors emeriti of music in the School of Music at our university, and is freely available for download at http://(removed for review, program and files can be provided upon request).

2. PAST WORK

We know of two recent examples addressing computer-assisted music instruction on intonation from the computer music community: [12], [13], though these efforts address several other performance aspects, including dynamics, rhythm, articulation, etc. Of these two, [13] shares our use of score following, though their use is based on on-line recognition, and thus is somewhat limited in its ability to relate its measurements to the musical score. We make analogous use of on-line recognition for real-time feedback, but focus mostly on off-line alignment, due to its greater accuracy and appropriateness for the off-line nature of the performer's analysis of a performance [12] shares some of the basic kinds of displays as our work, though the effort is restricted to the playing of long tones, rather dramatically restricting its reach. This work also does not relate the results to a musical score, thus shifting the burden of interpretation to the musician.

There has been significantly more commercial interest along these lines as exemplified by [1], [2], [3], [7], [14], [15]. The basic kinds of visual music display we use are found in the cited examples as well. [2] [15] uses the spectrogram, [3] use pitch trace representation, and [7], [14] annotates a musical score to reflect a specific performance. However, there are some important ways in which we differ from these efforts. Most important is our use automated score alignment, which allows us to relate the music data directly to our score representation. While [7] and [14] use traditional music score display, they relate the music data to the score by requiring the player to play along with a rigid accompaniment. While this "solves" the alignment problem, it imposes a foreign constraint on the musician for typical intonation practice. Other cited efforts either prompt the musician to play specific notes, or try to estimate the notes of the musical score from audio.

The most significant difference between our work and these cited is our use of score alignment as the fundamental means of relating our measurements to the music itself. Using score alignment we can link our three representations, thus allowing the user to move freely between them while retaining focus on the current position or note. An additional difference between our work and [7], [13], [14] is our deliberate effort not to grade the musician's performance, but rather to give them the objective feedback needed by the musician in reaching independent conclusions.

3. ESTIMATING PITCH

The backbone of *InTune* is a score-following system that aligns the audio input with a musical score. Thus we assume the musician plays from a known score. We base our approach on score following since the quality of blind (no score) music recognition degrades rapidly as complexity increases — we know of no blind recognition approaches, including our own, that produce good enough results for our task at hand. Furthermore, we wish to present our feedback in the context of the musical score. Since the score must be known for this to happen, we might as well put this knowledge to good use.

Our score following is based on a hidden Markov model, as documented in [10]. This model views the audio input as a sequence of overlapping frames, with about 30 frames per second, which form the observable part of the HMM, $y = y_1, y_2, \ldots, y_N$. We construct small (10-or-so-state) Markov models for each score note modeling, among other things, the distribution of the number of frames devoted to the note. These sub-models are concatenated together, in "left to right" fashion, to form our state graph. The hidden Markov chain, $x = x_1, x_2, \ldots, x_N$, corresponds to the path taken through this state space. Given audio data and a musical score, we perform alignment by computing the onset time of each score note, $\hat{n}_i$ as

$$\hat{n}_i = \arg\max_n P(x_n = \text{start}_i \mid y_i, \ldots y_N)$$

where start_i is the unique state that begins the *i*th note model. This approach performs well when confronted with the inevitable performance errors, distortions of timing, and other surprises that frequently occur in musical practice, and has been the basis for a long-standing effort in musical accompaniment systems [11].

Our score following approach tells us when the various notes of the musical score occur, thus giving us the approximate pitch for each frame of audio. That is, if the score match designates a frame to belong to a score

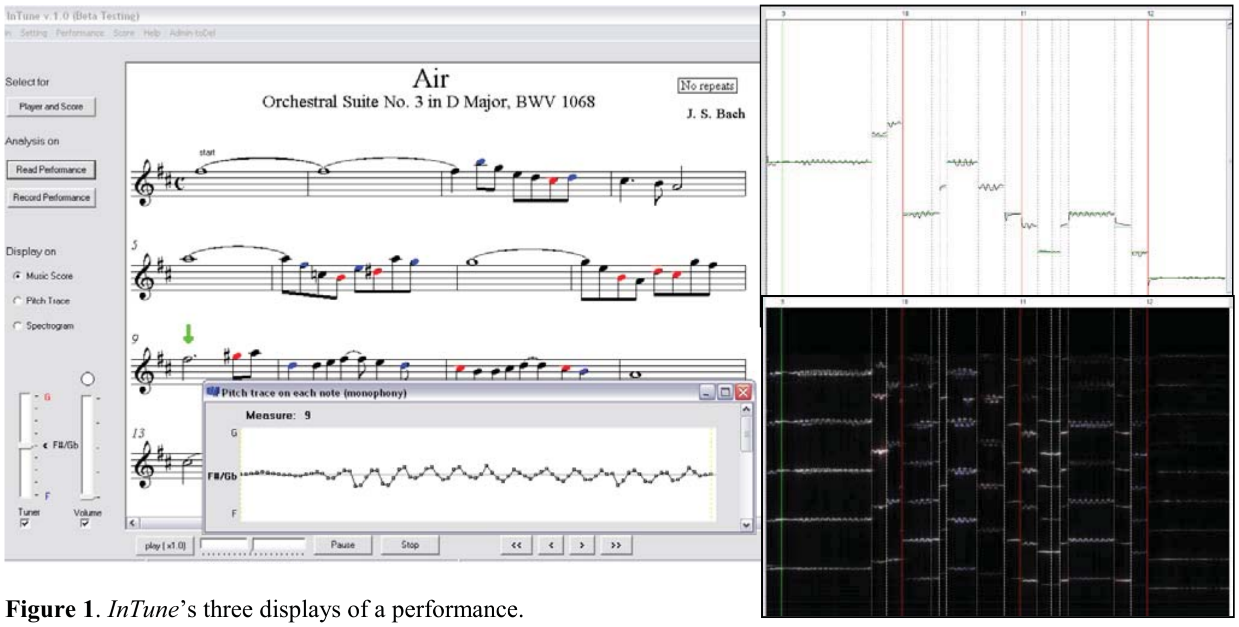


Figure 1. *InTune*'s three displays of a performance.
The score image (left) colors the heads of "suspicious" notes. The pitch trace (upper right) shows precise pitch evolving over time. The spectrogram (lower right) is the traditional display of frequency content evolving over time. The vertical lines show the measure or beat boundaries (red), the note boundaries (white), and the current position (green).

event having MIDI pitch m, then the pitch of the note is approximately

$$f(m) = 440 \times 2^{(m-69)/12} \tag{1}$$

Hz. Such knowledge makes it much easier to estimate the pitch more precisely. In fact, many pitch estimation and tracking approaches suffer more from coarse pitch errors on the octave level (misestimating by a factor of two), than from the fine tuning of pitch [6].

The frequency is defined here as the instantaneous rate of change of phase; we estimate the frequency for a particular frame by approximating this derivative. This is a timehonored and intuitive approach dating back to [8]. If $Y_n(k)$ is the windowed finite Fourier transform of y_n at frequency k, we estimate the frequency as

$$\hat{f}_n = \frac{k/r + [\phi(Y_{n+1}(k)) - E(k) - \phi(Y_n(k))]/2\pi}{\Delta t}$$

where r is the frame overlap rate, $\phi()$ is the argument or angle of a complex number, $E(k)=\frac{2k\pi}{r}$ mod 2π is the deterministic phase advance of frequency k between frames, and Δt is the time, in seconds between frames. The numerator in this calculation simply computes the fractional number of cycles that have elapsed for frequency k, which is then divided by the elapsed time to get cycles per second.

Since we know the nominal score pitch of the current note from our score match, our choice of k is not too difficult. If there is sufficient energy around the fundamental frequency we take k to be the frequency "bin" in the neighborhood of the fundamental having greatest energy. Otherwise we scan the neighborhoods of the lowest 4 or 5 harmonics seeking the bin having the greatest amplitude. If this bin corresponds to the hth harmonic, we must divide our frequency estimate by h to estimate the fundamental frequency. Thus our pitch estimation algorithm functions well when several of the lowest harmonics have little or no energy. When no harmonic seems to have any significant amount of energy, we assume the player is not generating any sound at the moment, and do not estimate frequency in this case.

4. THE THREE VIEWS OF INTUNE

On bringing up the program, the musician begins by choosing a piece to work on, at which point standard music notation is displayed. While *InTune* begins with a small collection of ready-made pieces, MIDI files can be imported, thus extending the program's range to nearly anything playable by a single instrument. The player then selects a range or excerpt from the piece and records a performance. The audio is then automatically aligned to the score, followed by pitch estimation, as described above. This information is then displayed to the musician in a collection of three linked views, as shown in Figure 1.

All three views use the notion of equal tempered tuning as reference point. For instance, if we choose A = 440 Hz as our pitch level, then the reference frequency of MIDI pitch *m* would be as given in Eqn. 1, (69 is the MIDI pitch for the "tuning" A). The location of the tuning A is adjustable by the user. We acknowledge here that there is no single "correct" view of tuning. For example, in many situations it is common to prefer tuning based on simple integer ratios, such as 3:2 for a perfect 5th. In addition, some players advocate various kinds of "expressive tuning" such as the raising of leading tones, or bending pitches in the direction of future notes. We choose equal temperament as our reference due to its simplicity and wide acceptance—not to assert its correctness. Users of the program can easily make their own judgments of the desirability or accuracy of the tuning based on this reference point without necessarily "buying in" to equal temperament. In fact, the importance of displaying, rather than judging, the tuning results was a basic tenet of ours, due to the lack of any single agreed-upon yardstick.

The *score view* is immediately presented by the program after a recording is made. This view employs a mark-up of the music notation, coloring notes whose mean frequency differs by more than a (user-adjustable) threshold from the equal tempered standard. We use red for high or "sharp" notes and blue for low or "flat" ones, due to their implications of hot and cold. The coloring of notes gives an easy-to-assimilate overall view of the performance that may show tendencies of particular notes or parts of phrases, such as the undesirable change in pitch that can accompany a change in loudness on some wind instruments. Clicking on any note in the score view opens a window that graphs the pitch trajectory over the life of the note. This aspect gives higher-resolution pitch detail, allowing one to see the tuning characteristics of vibrato, as well as variation associated with the attack or release of a note. Visualization of vibrato was of particular interest to the music faculty with which we developed this project.

Of course, one cannot appreciate the most important dimension of the performance without sound, so the score view (as well as the others) allows audio playback that is mirrored as a moving pointer in the image display. Variable-rate playback through phase-vocoding [5] allows the truly brave user to hear details of the performance often lost at the original speed. Since we have aligned the audio to our musical score the user can play back the performance beginning with any note, and at any speed, thus allowing random access to the audio and enabling more focused listening than normally possible with audio.

A second view of the audio data is called the *pitch trace* (top right of Figure 1). This representation is analogous to a piano roll graph in which notes are represented as horizontal lines whose height describes the note's pitch and whose horizontal extent shows the time interval where the note sounds. Typically, one uses a log scaling of frequency in a piano roll graph so that each octave (or any other interval) corresponds to a constant amount of vertical distance. We modify this graph simply by allowing the lines to "wiggle" with changing pitch. To make the graph more intelligible we mark measures, beats, or some other musical unit of the user's choosing, with vertical lines, courtesy of the score alignment. As with the score view, the user is free to page through the notes and to play the audio starting from the current location.
The final view (bottom right of Figure 1) is a traditional *spectrogram*, in which we show frequency energy on the vertical axis evolving over time on the horizontal axis. Except for the use of color to denote notes with suspicious tuning, and vertical lines to mark musical time units, this view presents an uninterpreted view of the raw data. To some extent, one can make judgments about timbre by the proportions of energy in the various "harmonics" of a note (integral multiples of the fundamental frequency). In user tests we have found a number of musicians to be particularly fascinated with this data view, since it seems to support concrete assertions about the seemingly intangible world of timbre.

5. USER STUDY

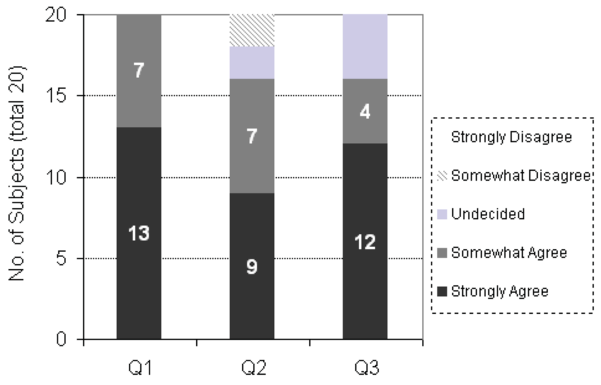


Figure 2. User study response.

We performed a small 20-subject user study using undergraduate and graduate performance majors from the School of Music. The school is one of the very best music conservatories in the US, yielding a musically sophisticated collection of subjects. The subjects consisted of a mixture of woodwind, brass, and string players, as well as two vocalists.

The students were directed to perform some simple tasks using the program, involving playing their instruments, recording, as well as visualizing and hearing their audio data. The students then responded to a questionnaire assessing their belief about the usefulness of *InTune* and their interest in incorporating the program into their practice. Overall, the students we quite positive about the the program, with most saying they would incorporate *InTune* into their practice if it were available. Figure 2 summarizes the response in the most illuminating questions:

Q1 Did *InTune* help you recognize inaccuracies you did not hear?
Q2 Is *InTune*'s sense of intonation consistent with your own?
Q3 Would you use *InTune* with your practice when it is available?

We were most pleased with the musician's willingness to use the program in actual practice, and hope that professed willingness holds true.

Several themes emerged through the written and voiced comments that accompanied the study. Virtually all perceived the program as an improvement over the tuner, though acknowledging the difficulty of carrying a laptop to the practice room. This improvement was primarily due to the possibility of scanning and studying past pitch histories while making these data accessible by relating them to the musical score. Players also commented on the program's facile and informative handling of fast notes. Some players found the program gave especially useful feedback on vibrato, by allowing one to clearly see the width of pitch excursions.

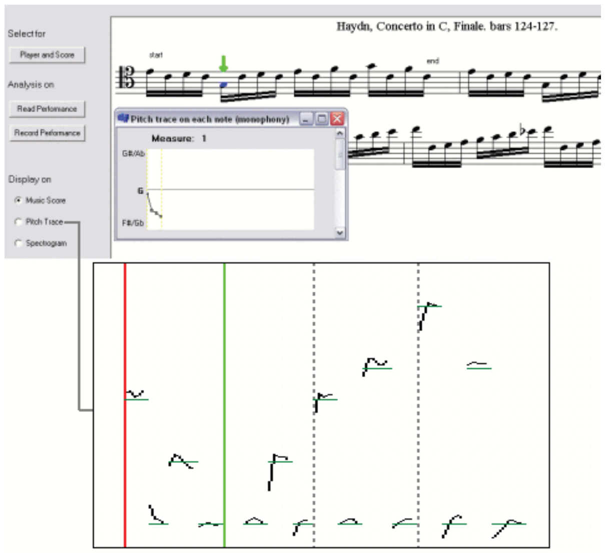


Figure 3. Example showing pitch estimation for fast notes.

The program did not perform as well with the vocalists who generally sing with accompaniment, thus giving an external pitch reference. The singers' overall pitch level tended to drift and all agreed that they should not be "marked down" for this. Another criticism repeated by several musicians addressed the note-oriented view of pitch. They observed that musicians often spend time "between" notes, and found the program's pitch estimation wanting in this scenario. We admit that our view of pitch estimation simply did not take this phenomenon into account, most common with singers and string players. In essence, we gain a significant advantage by assuming the player's pitch is close to the notated pitch, allowing accurate handling of otherwise difficult situations, though this gain does not come without cost. Several players argued for the value of expressive *context-dependent* tuning, not recognized by the program. In spite of our efforts no prescribe the correct answers, it seems inevitable that some may interpret the program's output this way.

Figure 3 shows an example from the user study of a horn playing a section of the Haydn Cello Concerto. The notes here are fast enough so that a tuner would provide little use, while accurate recognition from pure audio would be challenging and, likely, unreliable.

One especially interesting example occurred with a graduate flute major whose pitch data are shown in Figure 4 on the 2nd movement of the Mozart Clarinet Quintet, K. 581. In these data she observed a rising pitch trend in the early life of many notes. Our teacher's "face the music" maxim seemed to reverberate when she commented that the program had pointed out a tendency that she was unaware of, but could now hear. This example demonstrates the utility of automatic score alignment. The audio for this and all other examples can be heard at http:// (removed for review, files can be provided upon request).

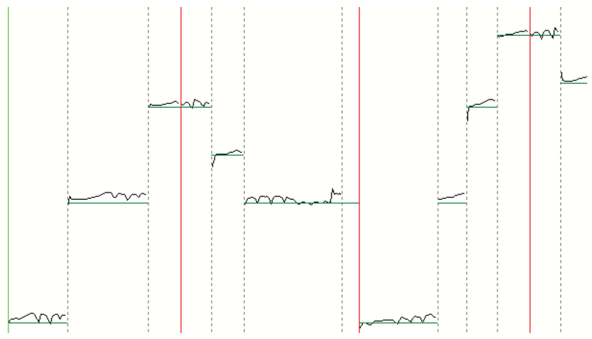


Figure 4. Example showing the rising pitch tendency, first made clear to the player by the program.

6. REFERENCES

[1] Agin, Gerald J. Intonia, http://intonia.com/index.shtml, 2008-2009.

[2] CantOvation *Sing&See*, http://www.singandsee.com/, 2008.

[3] ChaumetSoftware *Canta*, http://www.singintune.org/.

[4] Removed for review

[5] Flanagan, J. L. and Golden, R. M. *Phase Vocoder*. Bell System Technical Journal, November 1966.

[6] Kootsookos, Peter J. *A Review of the Frequency Estimation and Tracking Problems*. 1991.

[7] MakeMusic, Inc. *SmartMusic*, http://www.smartmusic.com/, 2008.

[8] McMahon, D. R. A. and Barrett, R. F. *Generalization of the Method for the Estimation of the Frequencies of Tones in Noise from the Phasese of Discrete Fourier Transforms.* Signal Processing Vol.12, 1987.

[9] Pygraphics *Interactive Pyware Assessment System*, http://www.pyware.com/ipas/, 2008.

[10] removed for review

[11] removed for review

[12] Robine, Matthias and Percival, Graham and LaGrange, M. “Analysis of Saxophone Performance for Computer-Assisted Tutoring”, *Proceedings of the International Computer Music Conference (ICMC07)*, Copenhagen, Denmark, 2007.

[13] Schoonderwaldt, E. and Askenfelt, A. and Hansen, K. F. “Design and implementation of automatic evaluation of recorder performance in IMUTUS”, *Proceedings of the International Computer Music Conference (ICMC05)*, Barcelona, Spain, 2005.

[14] StarPlayit *StarPlay*, http://www.starplaymusic.com/index.php, 2000.

[15] VoiceVista *VoiceVista* http://www.vocevista.com/, 2007.

EXPRESSIVE PERFORMANCE RENDERING: INTRODUCING PERFORMANCE CONTEXT

Sebastian Flossmann, Maarten Grachten
Dept. Computational Perception
Johannes-Kepler-University, Linz
{sebastian.flossmann,maarten.grachten}@jku.at

Gerhard Widmer
Dept. Computational Perception
Johannes-Kepler-University, Linz.
The Austrian Research Institute
for Artificial Intelligence (OFAI)
gerhard.widmer@jku.at

ABSTRACT

We present a performance rendering system that uses a probabilistic network to model dependencies between score and performance. The score context of a note is used to predict the corresponding performance characteristics. Two extensions to the system are presented, which aim at incorporating the current performance context into the prediction, which should result in more stable and consistent predictions. In particular we generalise the Viterbi-algorithm, which works on discrete-state Hidden Markov Models, to continuous distributions and use it to calculate the overall most probable sequence of performance predictions. The algorithms are evaluated and compared on two very large data-sets of human piano performances: 13 complete Mozart Sonatas and the complete works for solo piano by Chopin.

1 INTRODUCTION

Research on performance modelling and rendering constantly faces the problem of evaluation. The RENCON-Project [2] addresses this by providing a platform to present rendering systems to a public audience and in the process rate and judge the 'naturalness' and expressiveness of the rendered performances.

In the following we present YQX, the system that won all three awards in the autonomous category of the RENCON08 that was hosted by the ICMPC in September 2008 [1] [2]. However successful, the system tended to sometimes produce unstable, 'nervous' sounding performances. In response to this we present two extensions to the system that both aim for smoother variations in the performance, while ideally increasing the similarity to human performances. This is in concordance with [7] who states that an average of several performances can sound more aesthetically appealing than the actual performances going into the average. To achieve this we incorporate the *performance context*, information about the performance currently rendered, into the decisions. In contrast to the original system, which makes ad hoc decisions based only on the score context at hand, this adds a dynamic component.

In the first extension we use the additional information locally: The prediction for the previous note influences the current prediction according to the relations found in the training data. In the second extension we use the additional information to calculate the sequence of predictions that is globally the most probable of all, given the probabilities learned. In a series of experiments we test whether and to what extent the renderings of both extensions are smoother and more consistent than the renderings of the original system.

2 RELATED WORK

Much research has been done on the modelling and synthesis of expressive music. Since e.g. [10] gives a very detailed overview, only a few more recent approaches shall be mentioned here. Grindlay and Helmbold [1] use hierarchical Hidden Markov Models to generate expressive tempo values based on score characteristics. The different levels of hierarchy are used to represent different phrases of the piece. Due to the sophisticated learning algorithm and the intuitive structure the learned model is easy to interpret. They report good generalisation to new scores and positive evaluation in listening tests. A more recent approach [8], also submitted to RENCON08, uses the technique of Gaussian Processes [6] to automatically learn a mapping between score and performance characteristics. The model aims at predicting a sequence of performance parameters that is optimal with re-

[1] Given the specific context of this paper - published results of RENCON08, existing system YQX, availability of demo videos - we consider it pointless to try to keep this submission strictly anonymous.

[2] Videos of YQX in action at RENCON08 can be found at `http://www.cp.jku.at/projects/yqx`. The test pieces were composed specifically for the contest by Prof. Tadahiro Murao.

SMC 2009, July 23-25, Porto, Portugal

gard to the whole piece. Although the approach differs from ours, the intended effect is similar to our second extension to YQX. However the authors report a rather weak generalisation to new scores (perhaps due to the lack of high-quality training data).

3 THE DATA

In Spring 1989, Nikita Magaloff, a Russian-Georgian pianist famous for his Chopin interpretations, performed the entire work of Chopin for solo piano that was published during Chopin's lifetime (op. 1 - op. 64) in six public appearances at the Vienna Konzerthaus. These concerts were recorded with a Bösendorfer computer-controlled grand piano. The data set comprises over 150 pieces with over 320.000 performed notes. The MIDI data were manually matched to symbolic scores derived from scanned sheet music. The result is a unique corpus containing precisely measured performance data for almost all notes Chopin has ever written for solo piano.

The second data collection we use for the evaluation of our models are 13 complete Mozart Piano Sonatas played by the Viennese pianist R. Batik, likewise recorded on a Bösendorfer computer piano and matched to symbolic scores. This data set contains roughly 106.000 performed notes.

4 FEATURES AND TARGETS

We aim at learning a mapping between the score notes with their local score context and the human performance in our corpus. The prediction, the application of the learned mapping to unknown music, will be note-wise: each note of the melody of the score will be assigned three numeric values, the *targets*, determining the performance of the note. The targets are the dimensions with which we describe a piano performance: *loudness, articulation* and *tempo*. In the following instead of tempo we will actually use *ioi ratio*, which is directly related. The characteristics of a note and its local score context are described by the *features* extracted from the score.

One of these features (IR-Arch, see below) is based on E. Narmour's Implication-Realization model of melodic expectation [5]. The theory constitutes an alternative to Schenkerian analysis, focused more on cognitive aspects of expectation than on musical analysis. The model analyses the musical progression of a piece and the expectations aroused in the listener's mind. One of the main claims of Narmour is that sequences of intervals, harmonies etc. either produce further expectations, a situation of *non-closure*, or not, a situation of musical *closure*. Calculating the distance of a melody note to the nearest point of closure provides clues about whether a note represents a phrase boundary or not.

In our experiments we use the following (very small) set of score features:

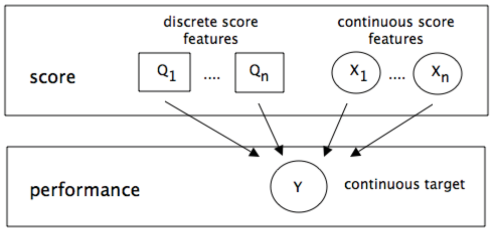


Figure 1. The probabilistic network forming the YQX system

Pitch Interval The interval between a melody note and its successor, measured in semitones.

Duration Ratio The logarithmic ratio between the score duration of a melody note and its successor.

I-R Arch The distance from the nearest point of closure, calculated from the Implication-Realization analysis.

The targets to be predicted are defined as follows:

IOI Ratio The logarithmic ratio of the length between two successive played notes and the length between the two corresponding score notes. A positive value indicates that the time between two notes is longer than notated, resulting in a decreased performance tempo.

Loudness The logarithmic ratio of the midi velocity of a note and the mean velocity of the performance. Thus positive values are louder than average, negative values softer.

Articulation This measures the amount of legato that is applied by a quotient of the gap between two successive played notes and the notated gap between the two corresponding score notes.

5 YQX - THE SIMPLE MODEL

YQX models the dependencies between score features and performance targets by means of a probabilistic network. The network consists of several interacting nodes representing the different features and targets. Each node is associated with a probability distribution over the values of the corresponding feature or target. A connection between two nodes in the graph implies a conditioning of one feature or target distribution on the other. Discrete score features (the set of which we call $\mathbf{Q}$) are associated with discrete probability tables, while continuous score features ($\mathbf{X}$) are modelled by gaussian distributions. The predicted performance characteristics, the targets ($\mathbf{Y}$), are continuously valued and conditioned on the set of discrete and continuous features. Figure 1 shows the general layout. The semantics is that of a linear gaussian model [4]. This implies that the case of a

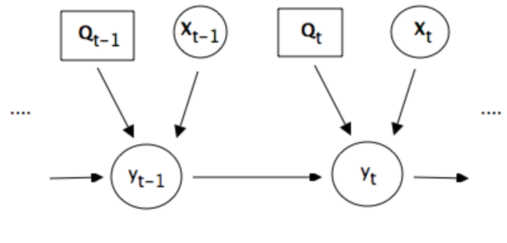


Figure 2. The network unfolded in time

continuous distribution parenting a continuous distribution is implemented by making the mean of the child distribution depend linearly on the value of the condition. In the following sets are marked with bold letters, vectors with an arrow super-scribed over the variable name.

Mathematically speaking this models a target y as a conditional distribution $P(y|\mathbf{Q}, \mathbf{X})$. Following the linear Gaussian model this is a gaussian distribution $\mathcal{N}(y; \mu, \sigma^2)$ with μ varying linearly with $\mathbf{X}$: given $\mathbf{Q} = \mathbf{q}$ and $\mathbf{X} = \overrightarrow{x}$ [3]

$$\mu = d_{\mathbf{q}} + \overrightarrow{k}_{\mathbf{q}} \cdot \overrightarrow{x},$$

where $d_{\mathbf{q}}$ and $\overrightarrow{k}_{\mathbf{q}}$ are estimated from the data by least squares linear regression. The average residual error of the regression is the σ^2 of the distribution. In effect we collect all instances in the data that share the same combination of discrete feature values and build a joint probability distribution of the continuous features and targets of those instances. This implements the conditioning on the discrete features $\mathbf{Q}$. The linear dependency of the mean of the target distribution on the values of the continuous features introduces the conditioning on $\mathbf{X}$. This constitutes the training phase of the model.

The prediction of the performance is done note by note. The score features of a note are entered into the network as evidence $\overrightarrow{x}$ and $\mathbf{q}$. The instantiation of the discrete features selects the appropriate probability table and the parameterisation $d_{\mathbf{q}}$ and $\overrightarrow{k}_{\mathbf{q}}$, the continuous features are used for calculating the mean of the target distribution μ. This value is used as the prediction for the specific note. As the targets are independent we create model and prediction for each target separately.

6 YQX - THE ENHANCED DYNAMIC MODEL

In the following we present two extensions of the system that both introduce a dynamic component by incorporating the prediction made for the preceding score note into the prediction of the current score note. Graphically this corresponds to first unfolding the network in time and then adding an arc from the target in time-step $t-1$ to the target in time-step t. Figure 2 shows the unfolded network. This should lead to smoother and more consistent performances with less abrupt changes and ideally to an increase of the overall prediction quality.

6.1 YQX with local maximisation

The first method is rather straight forward: We stick to the linear gaussian model and treat the additional parent (the target y_{t-1}) to the target y_t like an additional feature that we calculate from the performance data. In the training process the joint distribution of the continuous features, the target y_t and the target in the previous time-step y_{t-1} given the discrete score features, in mathematical terms $P(y_{t-1}, y_t, \overrightarrow{x}_t|\mathbf{q}_t)$, is estimated. That slightly alters the conditional distribution of the target y_t to $P(y_t|\mathbf{Q}, \mathbf{X}, y_{t-1}) = \mathcal{N}(y; \mu, \sigma^2)$ with [4]

$$\mu = d_{\mathbf{q}, y_{t-1}} + \overrightarrow{k}_{\mathbf{q}, y_{t-1}} \cdot (\overrightarrow{x}, y_{t-1}).$$

The prediction phase is equally straight forward. Just as in the simple model, the mean of $P(y_t|\mathbf{q}_t, \overrightarrow{x}_t, y_{t-1})$ is used as the prediction for the score note in time-step t. This is the value with the locally highest probability.

6.2 YQX with global maximisation

The second approach drops the concept of a linear gaussian model completely. In the training phase the joint conditional distributions $P(y_{t-1}, y_t, \overrightarrow{x}_t|\mathbf{q}_t)$ are estimated as before, but no linear regression parameters need to be calculated. The aim is to construct a sequence of predictions that maximises the conditional probability of the performance given the score features with respect to the complete history of predictions made up to that point.

This is calculated in analogy to the Viterbi-decoding in Hidden Markov Models (HMMs), where one tries to find the best explanation for the observed data [3]. Apart from the fact that the roles of evidence nodes and query nodes are switched, the main conceptual difference is that we have to deal with continuous instead of tabular distributions as used in the standard HMM setup. This rules out the dynamic programming algorithm usually applied but calls for an analytical solution, which we present in the following. Like the Viterbi algorithm the calculation is done in two steps: a forward and a backward sweep. In the forward movement the most probable target is calculated relative to the previous time-step. In the backward movement, knowing the final point of the optimal path, the sequence of predictions is found via backtracking through all time-steps.

6.2.1 *The forward calculation*

Let $\overrightarrow{x}_t, \mathbf{q}_t$ be the sets of continuous and discrete features at time t and N be the number of data points in piece. Let fur-

[3] We treat the real valued set of continuous score features like a vector

[4] The construct $(\overrightarrow{x}, y_{t-1})$ is a concatenation of the vector $\overrightarrow{x}$ and the value y_{t-1} leading to a new vector with a dimension $dim(\overrightarrow{x}) + 1$.

ther be α_t the probability distribution over the target values y_t to conclude the optimal path from time-steps 1 to $t-1$. By means of a recursive formula $\alpha(y_t)$ can be calculated for all time-steps of the unfolded network [5] :

$$\begin{aligned} \alpha(y_1) &= p(y_1|\mathbf{x}_1, \mathbf{q}_1) && (1)\\ \alpha(y_t) &= \max_{y_{t-1}\in\mathbb{R}} [p(y_t, y_{t-1}|\overrightarrow{x}_t, \mathbf{q}_t)\cdot\alpha(y_{t-1})] && (2)\end{aligned}$$

This formula can be interpreted as follows: assuming that we know for all the target values y_{t-1} in time-step $t-1$ the probability of being part of the optimal path, we can calculate for each target value y_t in time-step t the predecessor that yields the highest probability for each specific y_t of being on the optimal path. In the backward movement we will start with the most probable final point of the path (the mean of the last α) and then backtrack to the beginning by choosing the best predecessors. As we cannot calculate the maximum over all $y_{t-1}\in\mathbb{R}$ directly, we need an analytical way to calculate $\alpha(y_t)$ from $\alpha(y_{t-1})$, which we will derive in the following. We will also show that $\alpha(y_t)$ remains gaussian through all time-steps.

In the following we will use the distribution $p(y_{t-1}|y_t, \overrightarrow{x}_t, \mathbf{q}_t) \propto \mathcal{N}(y_{t-1}; \mu_{t-1}, \sigma^2_{t-1})$ that can be calculated via conditioning from the joint conditional distribution $p(y_{t-1}, y_t, \overrightarrow{x}_t|\mathbf{q}_t)$ that is estimated in the training of the model. For details as to how this is done see e.g. [6]. As we will prove that the $\alpha(y_t)$ are gaussian, we will refer to the mean and variance by $\mu_{\alpha,t}$ and $\sigma^2_{\alpha,t}$.

The inductive definition of α (eq. 2) can be rewritten (the conditioning on $\mathbf{q}_t$, $\overrightarrow{x}_t$ is omitted for simplicity):

$$\alpha(y_t) = \max_{y_{t-1}\in\mathbb{R}}[p(y_{t-1}|y_t)\cdot\alpha(y_{t-1})]\cdot p(y_t) \quad (3)$$

Assuming that $\alpha(y_{t-1})$ is gaussian, the result of the product in brackets is gaussian $\mathcal{N}(y_{t-1}; \mu_*, \sigma^2_*)$ with a normalising constant z, that also is gaussian in either of the means of the factors:

$$\begin{aligned} \sigma^2_* &= \frac{\sigma^2_{t-1} * \sigma^2_{\alpha,t-1}}{\sigma^2_{t-1} + \sigma_{\alpha,t-1}} && (4)\\ \mu_* &= \sigma^2_*\left(\frac{\mu_{t-1}}{\sigma^2_{t-1}} + \frac{\mu_{\alpha,t-1}}{\sigma^2_{\alpha,t-1}}\right) && (5)\\ z &= \frac{1}{\sqrt{2\pi|\sigma^2_{1-1} + \sigma^2_{\alpha,t-1}|}} e^{\left(\frac{-(\mu_{t-1}-\mu_{\alpha,t-1})^2}{2(\sigma^2_{t-1}+\sigma^2_{\alpha,t-1})}\right)} && (6)\end{aligned}$$

Later on z will be multiplied with a gaussian distribution over y_t, hence z has to be transformed to a distribution over the same variable. By finding a y_t, such that the exponent in eq. 6 equals 0, we can construct a proper μ_z and σ^2_z. Note that the variable μ_{t-1} is dependent on y_t due to the conditioning of $p(y_{t-1}|y_t)$ on y_t.

$$\begin{aligned} z &\propto \mathcal{N}(y_t; \mu_z, \sigma^2_z) && (7)\\ \mu_z &= -\frac{\sigma^2_t\cdot(\mu_{t-1}+\mu_{\alpha,t-1}) + \mu_t\cdot\sigma^2_{t,t-1}}{\sigma^2_{t,t-1}} && (8)\\ \sigma^2_z &= \sigma^2_{t-1} + \sigma^2_{\alpha,t-1} && (9)\end{aligned}$$

As z is independent of y_{t-1} it is not affected by the calculation of the maximum:

$$\begin{aligned} \alpha(y_t) &\propto \max_{y_{t-1}\in\mathbb{R}}[\mathcal{N}(y_{t-1}; \mu_*, \sigma^2_*)]\cdot && (10)\\ &\quad \mathcal{N}(y_t; \mu_z, \sigma^2_z)\cdot p(y_t)\\ &= \frac{1}{\sqrt{2\pi\sigma^2}}\cdot\mathcal{N}(y_t; \mu_z, \sigma^2_z)\cdot p(y_t) && (11)\end{aligned}$$

The factor $\frac{1}{\sqrt{2\pi\sigma^2}}$ can be neglected as it does not affect the parameters of the final distribution of $\alpha(y_t)$. The distribution $P(y_t)$ is gaussian by design and hence the remaining product again results in a gaussian and a normalising constant. As the means of both factores are fixed, the normalising constant this time is a single factor. The mean $\mu_{\alpha,t}$ and variance $\sigma^2_{\alpha,t}$ of $\alpha(y_t)$ follow:

$$\begin{aligned} \alpha(y_t) &\propto \mathcal{N}(y_t; \mu_{\alpha,t}, \sigma^2_{\alpha,t}) && (12)\\ \sigma_{\alpha,t} &= \frac{\sigma^2_t\cdot\sigma^2_z}{\sigma^2_t + \sigma^2_z} && (13)\\ \mu_{\alpha,t} &= \sigma_{\alpha,t}\left(\frac{\mu_z}{\sigma^2_z} + \frac{\mu_t}{\sigma^2_t}\right). && (14)\end{aligned}$$

Thus $\alpha(y_t)$ is gaussian in y_t, assuming $\alpha(y_{t-1})$ is gaussian. Since $\alpha(y_1)$ is gaussian, it follows that $\alpha(y_t)$ is gaussian for $1 \le t \le N$. This equation shows that the mean and variance of $\alpha(y_t)$ can be computed recursively over the mean $\mu_{\alpha,t-1}$ and variance $\sigma^2_{\alpha,t-1}$ of $\alpha(y_{t-1})$. The parameters of α_{y_1} equal μ_{y_1} and $\sigma^2_{y_1}$, which are the mean and variance of the distribution $p(y_1|\overrightarrow{x}_1, \mathbf{q}_1)$, and are estimated from the data.

6.2.2 *The backward calculation*

Once the mean and variance μ_t, σ^2_t of $\alpha(y_t)$ are known for $1 \le t \le N$, the optimal sequence $y_1, ..., y_N$ can be calculated:

$$\begin{aligned} y_N &= \mu_{\alpha,N} && (15)\\ y_{t-1} &= \operatorname*{argmax}_{y_{t-1}}\left[\mathcal{N}(y_{t-1}; \mu_*, \sigma^2_*\right] && (16)\\ &= \mu_* && (17)\end{aligned}$$

6.3 The Problem of Flatness

Both extensions presented above are designed to eliminate fast fluctuations from the predicted curves that, though small

[5] We use $\alpha(y_t)$ and $p(y_t)$ as an abbreviation of $\alpha(Y_t = y_t)$ and $p(Y_t = y_t)$, respectively

in amplitude, lead to unmusical irregularities in the rendered performances. The predicted curves are smooth, as we will show below, and suitable for rendering a consistent and principally acceptable performance. On the other hand, the flatter a curve, the more mechanical and unexpressive will the rendered performance sound. Based on the predictions we have, we can now try to counter this and superimpose peaks at selected points in the curves. To do this, a small set of note-level rules, automatically extracted by Widmer [9] from real performance data, is applied:

1. Staccato Rule: If two successive notes have the same pitch and the second of the two is longer, then the first note is played staccato.
2. Delay Next Rule: If two notes of the same length are followed by a longer note, the last note is played slightly delayed.
3. Trill Rule: If a trill is indicated in the score, the duration of the trill note is slightly prolonged.

7 RESULTS

We evaluated the algorithms on three data sets: The complete Chopin piano works, played by N. Magaloff and 13 complete Mozart Piano Sonatas, played by R. Batik, which were split into fast movements and slow movements. We first present the results of three-fold crossvalidations on the separate data sets and then take a detailed look at the predictions for an exemplary Mozart Sonata, and at the effects of the note-level rules. The quality of a predicted performance is measured by Pearson's correlation coefficient between the predicted curve and the curve calculated from the training data.

The crossvalidations, the results of which are given in table 1, show lower correlations on the Chopin data, implying that these data are much harder to predict than the Mozart pieces. This is probably due to the much higher variation that the limited information in the score features must account for. Testing with different sets of features shows that the prediction quality of a particular target depends greatly on the choice of features. As the goal of this paper is to compare different methods for introducing performance context, we restrict ourselves to one particular set of features and thereby refrain from choosing the best set for each target.

On the Mozart Sonatas the globally optimised algorithm shows a slight quality increase in predictive accuracy of the ioi ratios (12.5% for the fast movements and 10.9% for the slow movements) and loudness (10.0% and 7.0%). The slight decrease in average correlation for the articulation is not too surprising, as articulation is a rather local phenomenon that does not profit from long-term dependencies. For the Chopin data only a minor improvement in the prediction of the ioi ratios was registered (2.5%). The loudness in particular yielded low correlations. This is a problem we already encountered with the original YQX and that will need to be analysed in more detail in the future.

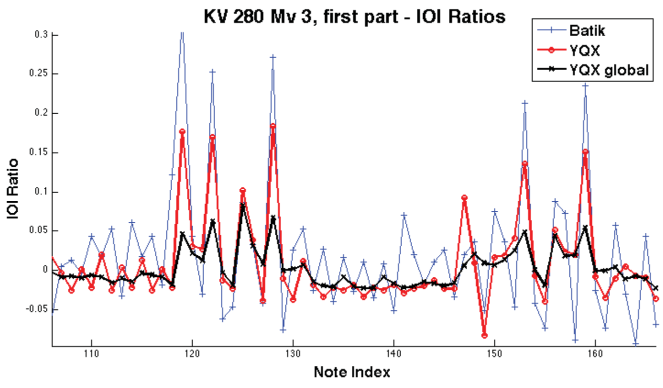


Figure 3. IOI Ratios predicted for bars 31 - 54 of K. 280, Mv.3

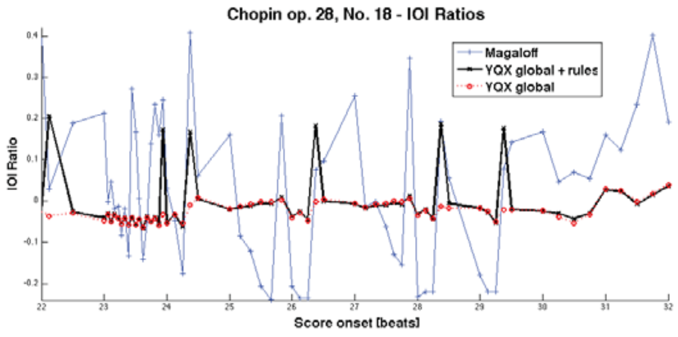


Figure 4. Effect of the note level rules applied to Chopin Prelude op.28 No. 18, bars 12 - 17

Figure 3 shows the ioi ratio predictions for bars 31 to 54 in the third movement of the Mozart Sonata K. 280. The original YQX algorithm exhibits small fluctuations that are rather uncorrelated with the human performance. This results in small, but noticeable irregularities in the rendered performance. In contrast to the human performance, which is anything but a flat curve, those make the result sound inhomogeneous instead of lively and natural. The globally optimised YQX eliminates those while still showing some of the peaks present in the human performance. The correlation for the movement was improved by 57.2% from 0.29 (YQX) to 0.456 (YQX global).

Figure 4 shows the effect of the note level rules described in section 6.3 on the ioi ratios predicted for Chopin Prelude op.28 No.18, bars 12 - 17. Instances of the Delay Next Rule occur at beats 24, 24.5, 26.5, 28.5 and 29.5, all of which coincide with great contrasts in Magaloff's performance.

	Mozart fast			Mozart slow			Chopin		
	ioi	loud	art	ioi	loud	art	ioi	loud	art
YQX	0.233	0.171	0.323	0.339	0.217	0.200	0.160	0.108	0.323
YQX local	0.238	0.160	0.296	0.345	0.196	0.161	0.169	0.053	0.313
YQX global	0.262	0.188	0.319	0.376	0.232	0.190	0.164	0.075	0.316

Table 1. Results of the crossvalidations. The values shown are correlations of the predicted performance with a human performance

8 CONCLUSION

The automatic synthesis of expressive music is a very challenging task, especially regarding the evaluation of a system, as one cannot really judge the aesthetic qualities of a performance by numbers. An adequate measure of quality can only be provided by human judgement. The rendering system we present passed this hurdle in the RENCON 2008 and therefore poses a baseline for our current research. The two extensions we devised address the problem of unsteady performances by incorporating the current performance context into the predictions. This proved to be a tightrope walk: Finding a way to restrain the predicted curves on the one hand but not losing (ideally increasing) similarity to the original curves on the other hand.

Of the data we tested our algorithms on, the Mozart Sonatas form a simpler task than the Chopin data. We registered a considerable increase in similarity to the real performances while achieving our goal of smoother predictions. The Chopin data pose a harder nut to crack. Due to the vast amount of highly heterogeneous data that has to be accounted for by a very limited set of features we were not able to increase the prediction quality significantly.

We consider this a work in progress. There is still a long way to go to a machine-generated performance that sounds profoundly musical. The main goal in the near future will be to define a set of features that is capable of explaining data with a high degree of interpretational freedom, like the Chopin data. This will raise the problem of how to balance the predicted performances against the peaks superimposed by the note level rules. We also have to solve the problem of big tempo or loudness differences within pieces that affect the global mean, as this is the reference point for our predictions. A promising approach is to calculate the ioi ratios and loudness relative to a current mean and incorporate the mean tempo and loudness curves into the prediction process. The biggest challenge, however, will be to combine the model with phrase level predictions, as e.g. are made in [11].

9 ACKNOWLEDGEMENTS

This work is funded by the Austrian National Science Fund (FWF) under project number P19349-N15.

References

[1] GRINDLAY, G., AND HELMBOLD, D. Modeling, analyzing, and synthesizing expressive piano performance with graphical models. *Machine Learning 65*, 2-3 (June 2006), 361–387.

[2] HASHIDA, M. Rencon - performance rendering contest for computer systems. http://www.renconmusic.org/, September 2008.

[3] JUANG, B. H., AND RABINER, L. R. Hidden markov models for speech recognition. *Technometrics, 33*, 3 (August 1991), 251–272.

[4] MURPHY, K. *Dynamic Bayesian Networks: Presentation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

[5] NARMOUR, E. *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*. University of Chicago Press, 1990.

[6] RASMUSSEN, C. E., AND WILLIAMS, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[7] REPP, B. The aesthetic quality of a quantitatively average music performance: Two preliminary experiments. *Music Perception 14*, 4 (1997), 419–444.

[8] TERAMURA, K., OKUMA, H., AND ET AL. Gaussian process regression for rendering music performance. In *Proceedings of the 10th International Conference on Music Perception and Cognition (ICMPC)* (2008).

[9] WIDMER, G. Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research 31*, 1 (2002), 37–50.

[10] WIDMER, G., AND GOEBL, W. Computational models of expressive music performance: The state of the art. *Journal of New Music Research 33*, 3 (2004), 203–216.

[11] WIDMER, G., AND TOBUDIC, A. Playing mozart by analogy: Learning multi-level timing and dynamics strategies. *Journal of New Music Research 32*, 3 (2003), 259–268.

FM4 SOUNDPARK AUDIO-BASED MUSIC RECOMMENDATION IN EVERYDAY USE

Martin Gasser
Austrian Research Institute
for Artificial Intelligence (OFAI)
martin.gasser@ofai.at

Arthur Flexer
Austrian Research Institute
for Artificial Intelligence (OFAI)
arthur.flexer@ofai.at

ABSTRACT

We present an application of content-based music recommendation techniques within an online community platform targeted at an audience interested mainly in independent and alternative music. The web platform's goals will be described, the problems of content management approaches based on daily publishing of new music tracks will be discussed, and we will give an overview of the user interfaces that have been developed to simplify access to the music collection. Finally, the adoption of content-based music recommendation tools and new user interfaces to improve user acceptance and recommendation quality will be justified by detailed user access analyses.

1 INTRODUCTION

The FM4 Soundpark is a web platform run by the Austrian public radio station FM4, a subsidiary of the Austrian Broadcasting Corperation (ORF). The FM4 Soundpark was launched in 2001 and gained significant public perception since then.

Registered artists can upload and present their music free of any charge. After a short editorial review period, new tracks are published on the frontpage of the website. Older tracks remain accessible in the order of their publication date. Visitors of the website can listen to and download all the music at no cost. Nowadays, the FM4 Soundpark attracts a large and lively community interested in up and coming music, and the radio station FM4 also picks out selected artists and plays them on terrestrial radio. At the time of writing this paper, there are 9577 tracks by 4689 artists enlisted in the online catalogue.

Whereas chronological publishing is suitable to promote new releases, older releases tend to disappear from the users' attention. In the case of the FM4 Soundpark, this has the effect of users mostly listening to music that is advertised on the frontpage, and therefore missing the full musical bandwith. To improve the accessibility of music in the FM4 Soundpark database, we proposed (1) a music recommendation system and (2) new user interfaces/visualizations that simplify accessing unknown music. Because the music available in the FM4 Soundpark was only very sparsely equipped with structured metadata (artist records can – but are not required to – be tagged with two genre labels out of a set of six), and because the active FM4 Soundpark community was considered of being yet too small for an approach based on collaborative filtering [4], we decided to implement a recommendation system utilizing a content-based music similarity measure.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

2 RELATED WORK

While many research prototypes of recommendation systems/visualizations of music collections that use content-based audio similarity have been described in the literature (e.g., [5, 11, 7, 10, 6], to name just a few), very little has been reported about successful adoption of such approaches to real-life scenarios. *Mufin* [1] is advertised as a music discovery engine that uses purely content-based methods. MusicIP [2] offers the *Mixer* application that uses a combination of content-based methods and metadata to generate playlists. Bang&Olufsen recently released the Beosound 5 [3] home entertainment center, which integrates content-based audio similarity with a simple "More Of The Same Music"-user interface, that allows users to create playlists by choosing an arbitrary seed song.

3 SYSTEM OVERVIEW

As we had to integrate our system with an existing infrastructure, we placed emphasis on a decoupled design and gradual integrability of our software into the system of our industrial partner. The FM4 Soundpark recommender was implemented as a web service that offers the following functionalities: (1) synchronize the recommender with the main

[1] http://www.mufin.com/
[2] http://www.musicip.com/
[3] http://www.beosound5.com/

database (import/delete tracks), (2) return similar songs to a given seed song, (3) return metadata (artist/track name, artist description if available), (4) return specialized data structures for the visualizations. Most client-server communication was done with proprietary XML- or text-based protocols. The high-level web service interface was implemented in Python (using the CherryPy [4] framework), while the low-level functionality (feature extraction, database management, similarity calculation) was implemented natively in C++.

The system frontend consists of two main user interfaces: (1) an Adobe Flash MP3 player with integrated recommendation of similar songs based on pure acoustic similarity, and (2) a downloadable 3D visualization application that uses a combined similarity measure (acoustic similarity + user-defined genre labels).

3.1 Main components

As already mentioned, our software is only loosely coupled with our partner's infrastructure. Therefore, we had to define a communication protocol that can be used to trigger the necessary synchronization operations, and we had to design algorithms that are able to work on a large and constantly changing database. More precisely, the following requirements had to be met: (1) the calculation of acoustic similarities between songs must be fast and memory efficient (see 3.3) and (2) for the map generation procedure (see 3.4) a tradeoff had to be found between quality and performance.

3.2 Feature extraction

From the 22050Hz mono audio signals two minutes from the center of each song are used for further analysis. We divide the raw audio data into overlapping frames of short duration and transform them to Mel Frequency Cepstrum Coefficients (MFCC), resulting in a spectrally smoothed and perceptually meaningful representation of the audio signal. MFCCs are now a standard technique for computation of spectral similarity in music analysis (see e.g. [8]). The frame size for computation of MFCCs for our experiments was $46.4ms$ (1024 samples). We used the first 20 MFCCs for all our experiments.

3.3 Computing spectral similarity of songs

We use the following approach to music similarity based on spectral similarity. For a given music collection of songs, it consists of the following steps:

1. for each song, compute MFCCs for short overlapping frames
2. train a single Gaussian (SG) to model each of the songs
3. compute distances between pairs of songs using the Kullback-Leibler divergence between respective SG models

We use a single Gaussian with full covariance per song [9] and calculate the acoustic similarity between two song models p and q as the symmetric KL divergence $D_{KL}(p,q)$

$$D_{KL}(p,q) = \frac{KL_N(p\|q) + KL_N(q\|p)}{2} \tag{1}$$

where

$$\begin{aligned} KL_N(p\|q) = 0.5 \log\left(\frac{\det(\Sigma_p)}{\det(\Sigma_q)}\right) + 0.5Tr\left(\Sigma_p^{-1}\Sigma_q\right) \\ + 0.5\left(\mu_p - \mu_q\right)' \Sigma_p^{-1}\left(\mu_q - \mu_p\right) - \frac{d}{2} \end{aligned} \tag{2}$$

$Tr(M)$ denotes the trace of the matrix M, $Tr(M) = \Sigma_{i=1..n} m_{i,i}$.

Because we tried to keep memory requirements low, we decided not to store the full distance matrix, but compute song-to-song similarities online. In the current implementation, the calculation of a full distance matrix row for 9500 songs takes around 50ms on a standard PC-based system.

3.4 Map generation

For the 3D visualization, we refined the main ideas from *Islands of Music* [11] and *Neptune* [5], implemented as an interactive 3D virtual landscape walkthrough. Central to this type of visualization is a terrain heightmap, which is generated automatically from the database of available tracks. The visualization uses an islands metaphor, where islands represent regions of similar-sounding music.

The heightmap profile was derived from a 2D MDS [2] solution that approximates distances derived from a combination of acoustic and metadata similarity between tracks as closely as possible. We chose not to use Self Organizing Maps because they inherently rely on an embedding of data points in a high dimensional vector space, which clearly is not the case when dealing with pure proximity data. One common way to construct vector spaces from proximity data – interpreting the distances of all data points to a data (sub)set – did not seem feasible because the database changes continuously (per day, on average 15 tracks are added, while others are deleted), and we needed a way to integrate new data without a complete recalculation of the map.

Although calculating an MDS solution from pure audio similarity tends to preserve distances as well as the local neighborhood of songs, it did not not produce visually discriminable clusters in 2D space. Each artist in the Soundpark carries two genre labels, which he or she can select

[4] http://www.cherrypy.org

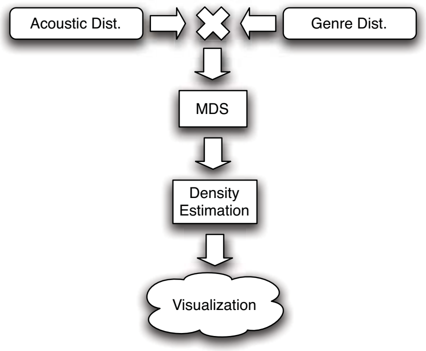


Figure 1. Map generation&visualization

freely from a set of 6 (Electronica, Funk/Soul, HipHop, Pop, Rock, Reggae). The desirable result from the map generation procedure was a layout where (1) tracks with equal genre labels are placed close to each other (2) the distances between tracks with equal genre labels still reflect mutual audio distances. Therefore, we decided to modify the audio similarity space by constructing a combined distance measure from the audio distance $\mathcal{D}_a$ and the genre distance $\mathcal{D}_g$:

$$\begin{aligned}
\mathcal{D}_a(t_1,t_2) &= \overline{\mathcal{D}_{KL}} + w_a \cdot (\mathcal{D}_{KL}(t_1,t_2) - \overline{\mathcal{D}_{KL}}) \\
\mathcal{D}_{scaled}(t_1,t_2) &= 1 - \exp(-D_a(t_1,t_2)/100)) \\
\mathcal{G}_k(t_i) &= 1 \text{ iff } t_i \text{ has genre k, 0 otherwise} \\
\mathbf{G}(t_i) &= (\mathcal{G}_0(t_i) ... \mathcal{G}_5(t_i)) \\
\mathcal{D}_g(t_1,t_2) &= 1 - \frac{\mathbf{G}(t_1) \cdot \mathbf{G}(t_2)}{min(\Sigma_k(\mathbf{G}_k(t_1)), \Sigma_k(\mathbf{G}_k(t_2)))} \\
\mathcal{D}_c(t_1,t_2) &= \mathcal{D}_{scaled}(t_1,t_2) \cdot ((1-w_g) + w_g \cdot \mathcal{D}_g(t_1,t_2)) \\
0 \leq & \; w \; < 1
\end{aligned}$$

$\overline{\mathcal{D}_{KL}}$ denotes the average audio distance in the data set. The weight factors w_a and w_g determine the influence of the tracks' audio distance and artists genre distance, respectively, on the overall distance measure. We chose values $w_a = 0.6, w_g = 0.9$ in order to enforce strong discrimination between tracks with no genre overlap. The rescaling step to calculate $\mathcal{D}_{scaled}(t_1, t_2)$ was inspired by [12].

Combined track-to-track distances are fed into the MDS module, which iteratively calculates a 2D layout, whereupon 2D positions are determined such that their respective distances approximate the original distances as closely as possible. We used Chalmers' [1] optimized spring model-based implementation. The complexity of the force calculation in this algorithm are reduced to $O(1)$ (by stochastically sampling the dataset and incrementally building a nearest-neighbor set for each point in the MDS problem by keeping the V nearest neighbors to each point over iterations), therefore its overall complexity is in $O(N^2)$, which is a crucial factor when dealing with data sets in the order of 10000 items and more (it should be noted that we found it necessary to increase the size of the sampling set to 20 in order to avoid falling into local minima during the solution of the MDS task, see [1] for a detailed discussion of the peculiarities of the algorithm). After $2N$ iterations (where N is the number of music tracks) the layout was assumed to be stable and the calculation was aborted.

From the low-dimensional track positions, we calculated a heightmap profile by applying a kernel density estimation [3] algorithm to the 2D point cloud, interpreting the estimated densities at points (x_i, y_i) as height values. The heightmap profiles are written to a binary file, which is then made available to the visualization client via a web server.

Because it is not feasible to execute the MDS procedure on each track import (which happens several times per day and requires the extraction of a full distance matrix), we place a newly imported track by calculating a weighted average position from its 5 nearest neighbors and do the MDS recalculation only during low-traffic times.

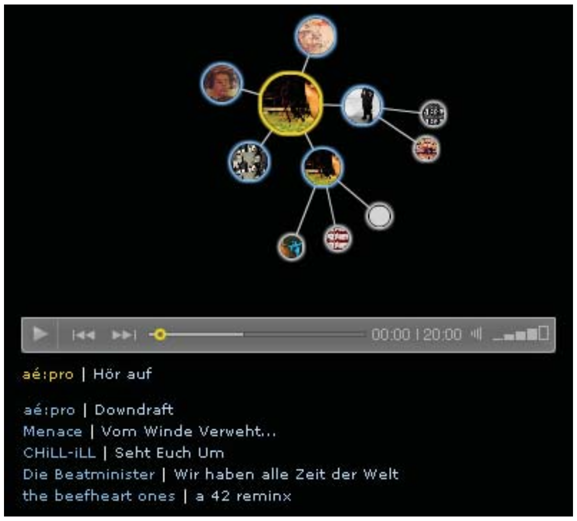
4 USER INTERFACE

Currently, two user interfaces to the recommendation system have been implemented: (1) A more traditional, Adobe Flash-based MP3 player interface with a small integrated visualization of similar tracks to the currently played one (see 4.1) and a downloadable, fully interactive, 3D visualization client (see 4.2).

4.1 Web player

The web player can be launched from within an artist's web page on the Soundpark website by clicking on one of the artist's tracks. Additionally to offering the usual player interface (start, stop, skipping forward/backward) it shows similar songs to the currently playing one in a text list and in a graph-based visualization (see figure 2).

The graph visualization displays an incrementally constructed nearest neighbor graph (number of nearest neighbors = 5), whereas nodes having an edge distance greater than two from the central node are blinded out. Figure 3 demonstrates the dynamic behavior of visualization (to simplify things, we have chosen a nearest neighbor number of 3 for this sketch): (1) User clicks on a track, visualizaton displays track (red) and the 3 nearest neighbors (green), (2) user selects track 4 by clicking on it, the visualization shows the track and its 3 nearest neighbors; note that track 2 – who is amongst the nearest neighbors to track 1 – is also in the NN set of track 4. (3) user selects track 5 as the new center, track 1 – which was nearest neighbor to track 4 in the preceding step – is also nearest neighbor to track 5. In the



aé:pro | Hör auf

aé:pro | Downdraft
Menace | Vom Winde Verweht...
CHiLL-iLL | Seht Euch Um
Die Beatminister | Wir haben alle Zeit der Welt
the beefheart ones | a 42 reminx

Figure 2. Player and SoundGraph visualization

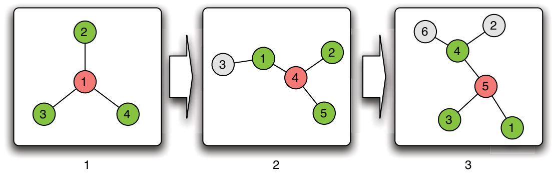


Figure 3. Interaction sequence in SoundGraph

long run, the re-occurrence of tracks in the NN sets indicates the existence of several connected components in the nearest neighbor graph.

4.2 3D visualization

The 3D visualization was implemented as a cross platform Java WebStart[5] application. The decision against a direct integration with the webpage had several reasons: (1) Flash cannot take full advantage of accelerated graphics hardware, which is ubiquitous nowadays, (2) Flash and Java applets both cannot access the local harddrive without quirks, which was necessary for local data caching.

The application was implemented using the JMonkey Engine[6] 3D framework. It implements an animated walk-through of the island landscape; the user is put in a first-person perspective, and she can control the application by the well-known WASD+mouse method (W/A/S/D keys for movements, mouse for controlling the viewing/walking direction). Tracks are represented as images that are positioned at their appropriate positions and heights. By focussing a track image and clicking on it with the mouse,

[5] http://java.sun.com/javase/technologies/desktop/javawebstart/index.jsp
[6] http://www.jmonkeyengine.com/

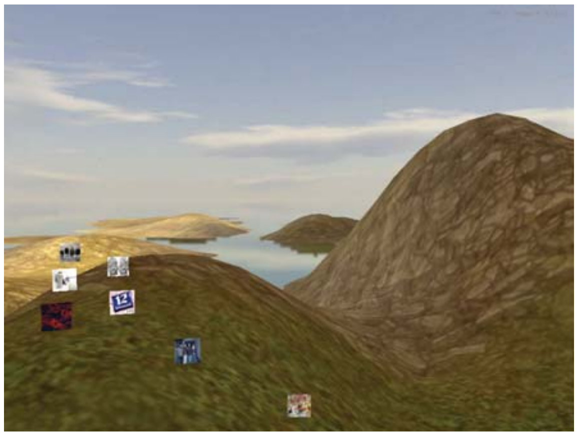


Figure 4. The 3D application

the user can start playback of a track. Figure 4 shows a screenshot of the running application.

By using optimized rendering algorithms (space subdivision with quadtrees), we were able to maintain interactive framerates ($\sim$35 frames per second), even for large scenes consisting of $\sim$9500 tracks.

5 EVALUATION

5.1 Graph theoretical considerations

Our analysis of the incrementally constructed nearest neighbor graphs concentrates on how likely it is that individual songs are reached when users browse through the graph. The analysis is done on an evaluation data base of 7665 songs. With the number of nearest neighbours nn equal 3 (5), 56.79% (65.28%) of the songs can be reached in principle. The other songs are never part of any of the nearest neighbour lists. Next we constructed the full nearest neighbour graphs emanating from all of the songs by incrementally expanding all subgraphs. As soon as an already visited song is reached again, the corresponding subgraph is fully constructed. The size of the full nearest neighbour subgraphs ($nn = 3$) for 96.20% of the songs is between 597 and 957, for the remaining 3.8% it is only between 4 and 46. For $nn = 5$ there is a similar behavior with more songs being reached: the size of the full nearest neighbour subgraphs for 97.91% of the songs is between 2306 and 2669, for the remaining 2.09% it is only between 6 and 50. Looking at the strongly connected graphs that exists in our data set helps to explain this surprising behavior. For our incrementally constructed nearest neighbor graph, a strongly connected component (SCC) is a subgraph where every song is connected to all other songs traveling along the nearest neighbour connections. Using Tarjan's algorithm [13] to find all SCC-graphs in our nearest neighbour graph with $nn = 3$

($nn = 5$), we find that there is one single large SCC with the size of 543 (2231) songs. The remaining 5913 (4674) SCC have a size of only 1.2 (1.16) on average. All our results seem to indicate, that there exists one large tightly connected subgraph that all other songs lead to when travelling along the nearest neighbour connections.

5.2 Quality of map visualization

To quantify the success of mapping the high dimensional distance space to the low dimensional map visualization, we computed the following measures of neighborhood preservation. We obtained an MDS solution D_a^{MDS} using only the audio distances $\mathcal{D}_a$ as input. We obtained another MDS solution D_c^{MDS} using the combined distances $\mathcal{D}_c$ as input. For every song, we compute sets of n nearest neighbours N_a, N_a^{MDS} and N_c^{MDS} using $\mathcal{D}_a$, D_a^{MDS} and D_c^{MDS} as distance measure. We also compute the set of n nearest neighbours Ng_a within genres using $\mathcal{D}_a$ as distance measure, i.e. only songs with identical genres are allowed to be part of the nearest neighbour set. We next compute the percentage of common (overlapping) nearest neighbours $O(N_a, N_a^{MDS})$ between neighbour sets N_a and N_a^{MDS}, as well as $O(Ng_a, N_c^{MDS})$ between Ng_a and N_c^{MDS}. Whereas $O(N_a, N_a^{MDS})$ measures the neighbourhood preservation for a mapping based on audio alone, $O(Ng_a, N_c^{MDS})$ measures how well local audio similarities within genres are preserved in mappings based on the combined distances. This is done for a random subset of 1000 songs with varying size of n and the average overlaps are depicted in Fig. 5. As can be seen, the preservation of local neighbourhoods within genres in the combined map visualization is even better than the preservation of neighbourhoods based on audio information only. Combining audio and genre information not only allows for more interesting map designs, but also respects audio similarity at a local level.

5.3 Usage statistics

In this section, we will present analyses that prove the acceptance of music recommendation technology in everyday use. Note that we did not evaluate usage behavior of the 3D visualization yet, as this part of the system was very recently finished.

By analyzing webserver log files, we were able to verify the following hypotheses about how it will be possible to change the music consumption behaviour of FM4 Soundpark users by utilizing music recommendation technology:
(1) The new technology is used by a significant number of users
(2) while the absolute number of track accesses might stay constant, the number of *distinct* track accesses increases,
(3) track accesses are more evenly distributed across the entire track catalogue, that is, the age distribution of downloaded tracks is becoming flatter.

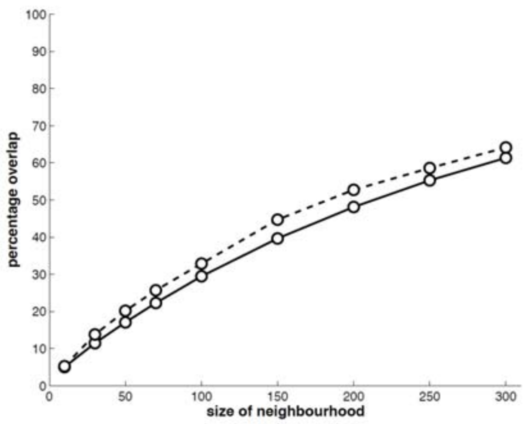


Figure 5. Percentage of overlap (y-axis) of neighbourhood sets $O(N_a, N_a^{MDS})$ (solid) and $O(Ng_a, N_c^{MDS})$ (dashed) for varying size of neighbourhood (x-axis).

Figure 6(a) plots the number of distinct tracks that have been downloaded per day between 2008-03-23 and 2009-03-29. Our recommendation service was launched on 2008-05-06. The peak at this date is clearly visible, and although the numbers turn down again during the following days, they clearly remain at a higher average level than they were before. The distinct track access numbers before/after the launch were distributed according to

	min	median	mean	max	stddev
before	17	338	359	781	137
after	41	593	672	4355	394

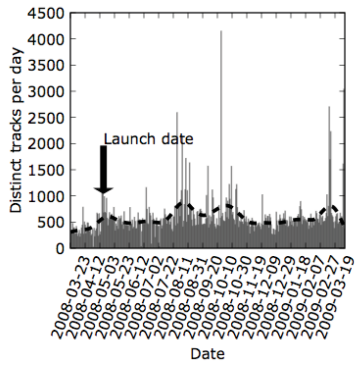
The ages of accessed tracks in days were distributed according to

	min	median	mean	max	stddev
before	0	200	483	2383	613
after	0	476	766	2714	795

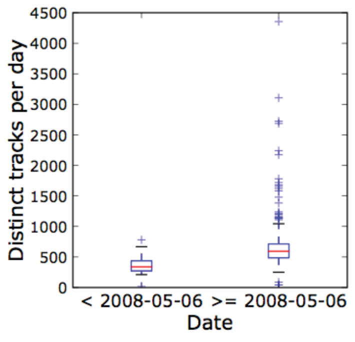
The boxplots in fig. 6(b) and fig. 6(c) give a better visual impression of the effect of the recommendation service.

6 CONCLUSIONS

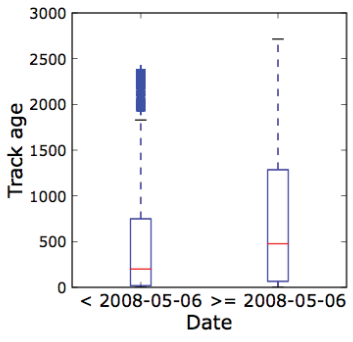
We have presented a real-life implementation of a music recommendation system that incorporates (1) purely content-based recommendations based on a seed track, (2) a 2D visualization based on pure audio similarity, and (3) an interactive 3D visualization based on a combined (audio and metadata-based) distance measure. We showed that recommendations based on a k nearest neighbor approach are



(a)



(b)



(c)

Figure 6. Usage statistics

likely to run into cycles if k is too small. We have presented a combined audio-and-metadata distance measure, whereas audio- and metadata-contributions can be weighted as required, and we have shown that a 2D MDS projection of a data set based on this measure respects audio similarity on a local level, while the coarse structure reflects distances calculated from the metadata. To check the usefulness of our system, we analyzed nearest-neighbor graphs calculated from pure audio similarity with graph theoretical methods, we analyzed MDS solutions with respect to neighborhood- and distance-preserving properties, and we performed statistical analysis of web server logfiles to analyze usage behavior. The results indicate that the approach we chose works reasonably well for our specific problem area.

7 REFERENCES

[1] Matthew Chalmers. A linear iteration time layout algorithm for visualising high-dimensional data. In *Proceedings of IEEE Visualization*, 1996.

[2] M.F. Cox and M.A.A Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

[3] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[4] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.

[5] Peter Knees, Markus Schedl, Tim Pohle, and Gerhard Widmer. An innovative three-dimensional user interface for exploring music collections enriched with meta-information from the web. In *Proceedings of the ACM Multimedia*, 2006.

[6] Paul Lamere and Douglas Eck. Using 3d visualizations to explore and discover music. In *Proceedings of the 8th International Conference on Music Information Retrieval*, Vienna, Austria, September 2007.

[7] Stefan Leitich and Martin Topf. Globe of music - music library visualization using geosom. In *Proceedings of the 8th International Conference on Music Information Retrieval*, 2007.

[8] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the 1st International Conference on Music Information Retrieval*, Plymouth, Massachusetts, 2000.

[9] Michael Mandel and Dan Ellis. Song-level features and support vector machines for music classification. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, United Kingdom, 2005.

[10] Robert Neumayer, Michael Dittenbach, and Andreas Rauber. Playsom and pocketsomplayer: Alternative interfaces to large music collections. In *Proceedings of the Sixth International Conference on Music Information Retrieval*, 2005.

[11] Elias Pampalk. Islands of music: Analysis, organization, and visualization of music archives. Master's thesis, Vienna University of Technology, 2001.

[12] Elias Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Vienna, Austria, March 2006.

[13] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

SENSORY THREADS: SONIFYING IMPERCEPTIBLE PHENOMENA IN THE WILD

Robin Fencott
Interaction, Media, and Communication Group
Queen Mary University of London
robinfencott@dcs.qmul.ac.uk

Nick Bryan-Kinns
Centre for Digital Music
Queen Mary University of London
nickbk@dcs.qmul.ac.uk

ABSTRACT

Sensory Threads is a pervasive multi-person interactive experience in which sensors monitor phenomena that are imperceptible or periphery to our everyday senses. These phenomena include light temperature, heart-rate and spatial-density. Participants each wear a sensor as they move around an urban environment, and the sensor data is mapped in real-time to an interactive soundscape which is transmitted wirelessly back to the participants.

This paper discusses the design requirements for the Sensory Threads soundscape. These requirements include intuitive mappings between sensor data and audible representation and the ability for participants to identify individual sensor representations within the overall soundscape mix. Our solutions to these requirements draw upon work from divergent research fields such as musical interface design, data sonification, auditory scene analysis, and the theory of electroacoustic music. We discuss mapping strategies between sensor data and audible representation, our decisions about sound design and issues surrounding the concurrent presentation of multiple data sets. We also explore the synergy and tension between functional and aesthetic design, considering in particular how affective design can provide solutions to functional requirements.

1 INTRODUCTION

Our everyday surroundings contain countless phenomena to which we are oblivious. Our eyes and ears can only detect within a limited range, while within our own bodies there are countless rhythms and processes which we cannot feel or sense.

Sensory Threads is an interdiserplinary research project that combines the expertise of several institutions, and builds upon technologies developed for *Snout* [1] and *Feral Robots* [8]. The aim of the project is to create a collective and shared sensing experience which highlights imperceptible phenomena in our everyday surroundings. In the Sensory Threads experience, participants wear mobile sensing technology (see figure 1, with discussion in section 4) and go on an expedition through the urban environment. During their expedition, a generative soundscape translates the real-time sensor data into an audible representation, allowing the participants to explore and experience the hidden quantities and patterns of imperceptible phenomena that surround them.

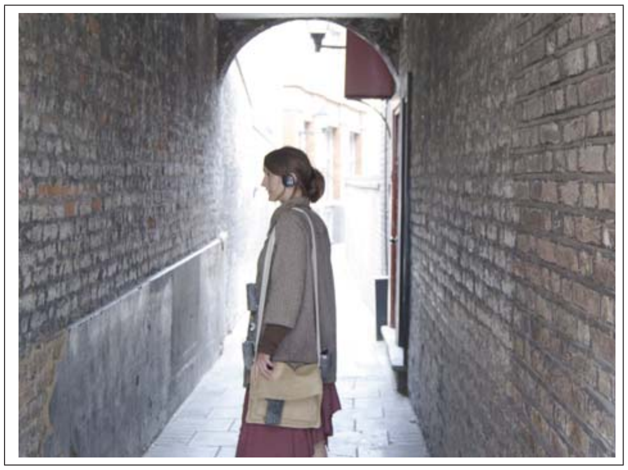


Figure 1. Participant wearing a mobile sensor.

A key requirement for the soundscape is the audible presentation of sensor data in a form which is intuitively perceivable, thus allowing a participant to isolate the contributions made by their own sensor or those of others in the group. The soundscape also needs to be pleasant to listen to, so as to retain participant attention during the expedition. Solutions to these design challenges have been informed by research from a range of fields, including data sonification, electro-acoustic music and Auditory Scene Analysis.

The rest of the paper is structured as follows. Section 2 surveys related work and introduces research which informed our approach to sound design, section 3 summarises the requirements for the soundscape, stressing design problems posed by these requirements. Section 4 documents the implementation process, giving examples of how the-

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

ory from the above-mentioned fields was applied to satisfy the design requirements. Section 5 outlines future work, and the paper is concluded in 6.

2 RELATED WORK AND SOUND DESIGN INFLUENCES

2.1 Related Work

Pervasive games are a form of interactive experience which attempt to blur the boundaries between the physical world and digital representations [16]. A good example is *Can You See Me Now* [2], a game which involves real-world 'runners' in an urban environment chasing online players who are navigating a virtual model of the same physical location. While using similar technologies (e.g., Global Positioning System) to these pervasive games, the Sensory Threads experience is not game-like or task-based in design; emphasis is instead placed on the reflective experience of listening to changes in the soundscape while exploring the urban environment. Sensory Threads therefore shares more commonality with mobile experiences such as *Sonic City* [10] and *'Ere be Dragons* [7]. *Sonic City* is a mobile automatic music creation system that uses mobile sensors to control the manipulation of environmental sounds surrounding the participant as they move through the city. *'Ere be Dragons* is a pervasive experience which uses the participant's heart rate to control the real-time generation of a virtual environment.

Sensory Threads is also similar to Christina Kubisch's 'Electrical Walks' [6], in which participants walk around a city while wearing headphones that are designed to amplify the electro-magnetic interference caused by wireless networks, electrical lighting and suchlike. However unlike Kubisch's walks, which give direct access to the phenomena under observation, the sonification process in Sensory Threads places a layer of abstraction between the raw sensor data and the audible representation.

Although [10] [7] and [6] generate real-time audio in response to live sensor data, they are solitary experiences, and are therefore distinct from Sensory Threads, which is a collective sound experience. An example of a collaboratively generated, sensor controlled soundscape is presented in [13], however the emphasis is on encouraging spontaneous performance activities, rather than reflection on the sensor data itself.

While we do not view Sensory Threads as an instrument or musical interface, it is important to mention collaborative interactive music systems. Blane and Fells [3] present a comparison of many such projects, however in the majority of cases, these projects are based around a fixed location such as a computer system or physical interaface(s). A notable exception to this is [17], who present a large scale mobile collaborative music system using GPS and wireless technologies. Despite differences in the style and presentation of these multi-party experiences, we can view them, and Sensory Threads, within the framework of creative mutual engagement [5].

2.2 Sound Design Influences

This section introduces fields of research which informed the development of the sound design for the Sensory Threads soundscape. Concrete examples of their application are given in subsequent sections.

2.2.1 Sonification

Sonification is the process of representing data in an audible format, using speech or non-speech sounds. Within the field of sonification, we are particularly interested in approaches to presenting several pieces of information concurrently. McGookin [12] notes that the larger the quantity of information presented simultaneously, the more problematic it is for a user to interpret it correctly. [12] suggests several design guidelines for addressing this problem, such as using a range pitch registers and timbres for different information sets, and de-correlating the temporal onset of concurrent events.

While the field of data sonification provides guidelines for the audible representation of data, there have been few attempts to address the aesthetic quality of the sonification. One exception is Vickers [15], who notes that the approaches to organising sound used by electro-acoustic music composers could be employed to enhance the aesthetic appeal of sonified data.

2.2.2 Auditory Scene Analysis

Auditory Scene Analysis (ASA) investigates the means by which humans interpret auditory signals. A key theory in ASA is that of segregation; the process which allows us to pick individual components, (e.g., voices or music) from the noisy mix of sounds in our environment; a phenomenon commonly known as the 'cocktail party effect'. ASA attempts to uncover the strategies we apply when segregating signals into discrete channels (referred to as streams), and the factors which affect our ability to perform this segregation. ASA also investigates the perceptual fusion of auditory components; where sounds from different sources are perceived as part of a single auditory stream [4].

ASA is useful in our context because it provides suggestions for how to design a soundscape in which the constituent parts are individually identifiable.

2.2.3 Spectromorphology

Spectromorphology, proposed by electroacoustic music composer Dennis Smalley, is a theoretical framework for understanding the activity of listening, and the process of recognising sounds or sound-types [14]. Spectromorphology refers to the spectral qualities of a sound (the *spectro*) and the ways these qualities change over time (the *morphology*). In his theory, Smalley argues that listeners have a 'natural' ability to associate sounds with possible causes, a process he calls 'source-bonding'. One implication of the theory of spectromorphology is that composers can try to suggest the cause or nature of imaginary or synthetic sounds, by designing sounds which may appeal to a listener's innate tendency to perform source-bonding.

In the Sensory Threads soundscape we try to leverage the idea of source-bonding, by giving each audio stream distinct spectromorphological characteristics, each intended to mirror the properties of the phenomena they represent. This also helps us to separate the different sensor representations within the overall mix of the soundscape, as each representation is distinct.

2.2.4 Summary

Sonification and spectromorphology are both concerned with the use of sound as a medium for description or representation. Sonification seeks to present some external source of information in an audible format, using speech or synthesised sound. Spectromorphology on the other hand is a theoretical framework for classifying and explaining sound types, and can be used as a tool for describing specific features *within* a sound event. Underlying both sonification and spectromorphology we find the field of Auditory Scene Analysis, which investigates the human perception of sound.

3 SOUNDSCAPE DESIGN REQUIREMENTS

This section gives more detail about the specific design requirements and features we identified for the Sensory Threads soundscape. Most of the aesthetic and experience related requirements were drawn from consultations with Proboscis, the arts organisation co-ordinating the Sensory Threads project. Other requirements were drawn from our previous experience in creating interactive sound experiences (e.g., [9] [13]) and our reflections on related research. Solutions to our design requirements are presented in the subsequent section, where we also consider the issue of balancing aesthetic concerns and functionality.

3.1 Perceivable mapping between sensor data and sound

The soundscape must provide feedback to the participants about the state of the environment and their own physiological systems. Therefore one of the most important requirements is that incoming sensor data is mapped to an audible representation in a direct and perceivable fashion, thus allowing participants to listen to the contribution their own sensor is making to the soundscape as a whole.

To facilitate this, we decided that each sensor data stream should be matched to a specific voice or instrument within the soundscape. To avoid confusion while listening, we require these voices to be sonically distinct, so that each voice is perceived within it's own auditory stream, and voices do not interfere with, or mask one another. We also argue that the sound design for the sensor representation should reflect the phenomena being monitored, so participants can intuitively interpret the meaning of the soundscape.

While providing identifiable sensor readings is integral to the functionality of the soundscape, we also argue that it is important for promoting group interaction and mutual engagement. We believe participants should be able to identify changes caused by their sensor, and also become aware of the way their fellow explorers are shaping the auditory experience.

In summary, the mapping and sound design for the soundscape should serve the dual purpose of representing the data in an easily readable format and promoting a sense of self identify and group awareness within the participants.

3.2 Longitudinal Requirements

As the Sensory Threads experience lasts for up to an hour, we recognise the need to incorporate temporal development into the soundscape, to retain interest, counter fatigue and avoid desensitisation to changes in the sensor data. We therefore considered additional, indirect sensor mappings to control longer term development of the soundscape, and also 'composed' changes within the soundscape, such as musical progressions or sounds which gradually evolve.

4 IMPLEMENTATION

4.1 Development Process

We developed the soundscape iteratively, starting with a primitive prototype, and making modifications based on informal testing and feedback from meetings at Proboscis.

4.2 Sensing Platform

Our collaborators at Birkbeck College have developed a portable wireless sensing platform using embedded Linux Gumstix computers. The current version of Sensory Threads uses three of these devices, each of which transmits sensor readings via Bluetooth to a small laptop computer, where the soundscape is generated. Each participant carries a single sensor. The sensors used are a light sensor, a noise sensor and a 'spatial density' sensor, which uses four ultrasound range finders placed on the front, back and sides of a participant to give an estimation of how cramped or constricted the participant is within their current environment. A heart-rate monitor, GPS receiver, 3G internet uplink and web-camera are also connected to the laptop.

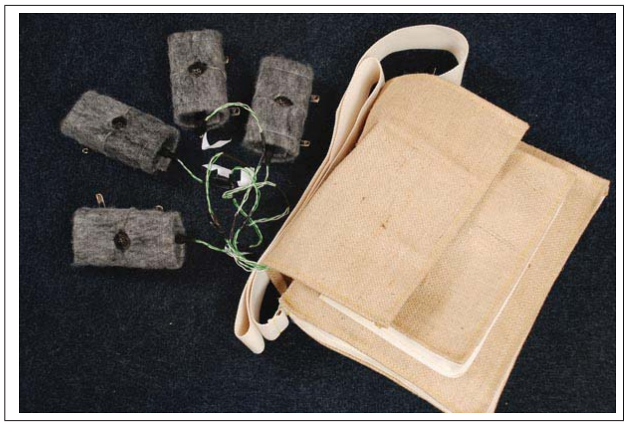


Figure 2. Spatial Density sensors and shoulder bag

A Java program (written by the first author) interfaces with the sensor hardware using Bluetooth and serial port connections, and passes relevant sensor data to the soundscape using Open Sound Control [18]. This Java program also uploads the GPS and sensor data to the web-server for use in a gallery-based listening environment. The program attempts to re-establish Bluetooth or internet connections if they are lost, and all sensor data is logged locally.

A second Java application takes still images using the attached web-camera. Images are taken at points of significant change in the sensor data streams.

Audio is generated in the soundscape program (see below) and transmitted to the four participants using Bluetooth headphones with external Bluetooth audio transmitters. These transmitters are used instead of the laptop's onboard Bluetooth interface, making the system more robust, as headphones can disconnect and reconnect without causing interference to the laptop or soundscape program.

The whole system runs on battery power and is worn in fabric satchels created by a member of Proboscis (see Figure 2).

4.3 Soundscape Program

The soundscape program is written in SuperCollider [11]. All sounds are synthesised in real-time, rather than being based on sampled audio. The main advantage of this approach is that the synthesis parameters can be modulated directly by the incoming sensor data, making the soundscape flexible and sonically varied. Another advantage of this approach is that the synthesis models can be quickly created, modified and tested during development.

4.4 Data Representation Streams

We developed four sound synthesis models, which we refer to as the 'representation streams'. These representation streams are collections of synthesis graphs coupled with control and mapping algorithms. Each representation stream was designed for a specific sensor.

4.4.1 Heart-Rate Stream

This heart-rate sensor is represented as a high-pitch pulse-train, whose rate is determined by the heart-rate of the participant wearing the heart-rate monitor. Reverberation is applied to the pulse, to make it sound less harsh. Over time, the pitch of the individual pulses begins to modulate, gradually transforming the simplistic tones into a melody.

4.4.2 Spatial Density Stream

The spatial density sensor is represented as a persistent drone, to which frequency and amplitude modulation are applied. As the participant wearing the sensor becomes more tightly constricted in their current environment, the rate of modulation is increased, causing the timbre of the drone to become intense and foreboding. In a low-density environment the drone becomes lighter and less prominent.

4.4.3 Noise Level Stream

The noise level sensor is represented as filtered white noise, where the sensor reading is mapped to the cut-off frequency of a low-pass filter. Some reverberation is applied to make the noise sound less harsh.

4.4.4 Light Stream

The light sensor modulates the filter frequency of a rich synthesised tone. Over time, additional musical notes are introduced to this tone, creating a rich harmonic backdrop to the soundscape. Delay and reverberation effects emphasise the modulations caused by variations in the sensor data.

4.5 Sound Design

We designed the soundscape to stand out against the acoustic environment, using entirely non-speech sounds, and employing synthetic timbres rather than emulations of acoustic instruments. The musical genre of the soundscape could be described as ambient or minimal electronica.

Our sound design decisions for the representation streams were not arbitrary. Rather, we tried to apply ideas from Smalley's theory of spectromorphology to make each representation stream an intuitive reflection of the phenomena under observation. For instance as the wearer of the spatial density sensor becomes more tightly constricted, the increasing modulation is intended to suggest the sensation of claustrophobia. Another example is the light sensor representation stream, which becomes sonically brighter and more energetic as the ambient light level increases.

We also use the idea of source-bonding, as with the variably spaced pulses of the heart rate representation, which imply a source that is emitting discrete events at a constantly changing speed. Here, we suggest that the pulsation becomes a metaphor for heart rate, as the pulses are not directly aligned with the heart beats of the sensor wearer.

4.6 Concurrent Data Representation

Using a distinct sonic identity for each sensor representation also means that streams can be distinguished from one-another within the overall mix. To ensure that each representation stream is sufficiently distinct, we followed principles outlined by [12] for the concurrent presentation of auditory information. These guidelines include the use of different timbres and pitch registers for each data element. We therefore used a variety of sound types, including high pitched tones, filtered bands of noise, and harmonic textures.

We also applied lessons learnt from the principles of auditory scene analysis (ASA), such as trying to ensure that each sensory representation gets assigned to it's own perceptual stream. One approach to this was using a mixture of continuous and discrete sounds, which can be viewed in gestalt terms as *figures* and *grounds*. Where possible we also de-correlated the onset of sound events, and used spatialisation to segregate elements, by placing them at different positions within the stereo field.

To further avoid conflict between the representation streams, we chose not to map sensor data directly to pitch, but instead opted to map the incoming data to temporal and spectral parameters. This approach allows us to place each sensor representation within a specific pitch region, with the assurance that the live sensor data will not cause sensor representations to gravitate towards a common frequency range and become difficult to distinguish between.

4.7 Functionality and Aesthetics

During the development of the soundscape, we became engaged in the tensions and synergy between functional and aesthetic design concerns. Functional problems included providing accurate representation of the data, and allowing for the concurrent presentation of the four data sets. The main aesthetic concerns we faced were developing representation streams which were pleasant to listen to, ensuring that the soundscape is clear and coherent despite multiple layers of audio, and incorporating temporal development.

In some cases the functional and aesthetic requirements were easily satisfied. For instance by assigning each representation stream unique characteristics, we found that the requirement for identifiable concurrent data presentation was met, however in the process we also arrived at a set of interesting and complimentary sonic voices, which work to create a coherent and musical soundscape.

The requirement for pleasant sounding sonification was met by using harmonic and musical sounds where possible, and through applying reverberation to create a more 'polished' presentation. In creating a soundscape that is pleasant to listen to, we also go some way to satisfying the functional requirement of supporting and encouraging engagement over a prolonged duration.

However, functional and aesthetic concerns were not always neatly resolved. As an example, we are conscious that temporal developments of the soundscape (as outlined above) might be confusing or misleading to the participants, who could potentially accredit time-based changes to the incoming sensor data. This is still an unresolved issue.

5 FUTURE WORK

5.1 Soundscape Development

A primary concern is to enhance the temporal evolution of the soundscape over the course of a Sensory Threads expedition. We see this work as important because if the soundscape remains relatively static participants may become bored or fatigued by the sounds. This may involve developing a context sensitive model of the experience, which is maintained as an expedition takes place. This approach is a departure from the current state-machine approach to temporal development, where new behaviour is introduced as certain time thresholds are crossed.

Although the system can recover from communication loss, we plan to introduce audible feedback to indicate when participants stray too far from one-another. As well as being functionally useful, these signals could also become an aspect of the group experience.

5.2 Formal Evaluation

We intend to carry out an evaluation of Sensory Threads, investigating not only the degree to which participants are able to interpret the sensor data, but also social factors such as the characteristics of the group interaction that takes place during the Sensory Threads experience.

We would also also like to assess the degree to which our implementation matches our design requirements, and we will use the evaluation to improve the soundscape design, in accordance with the principles of user-centric design.

Sensory Threads will be shown at various events during the summer of 2009, and we aim to use these events as a platform for performing our evaluations.

6 CONCLUSION

Sensory Threads is a pervasive multi-party experience in which four participants listen to an interactive soundscape as they move around an urban environment. Each participant wears a mobile sensor, which controls a particular element within the soundscape. The soundscape has been designed according to a specific set of requirements, and our solutions to these requirements have been made through the application of ideas and techniques from a variety of research fields and artistic disciplines.

7 ACKNOWLEDGEMENTS

Sensory Threads was also developed by Demetrios Airantzis, Frederik Lesage, George Roussos, Jenson Taylor (Birkbeck, University of London), Joe Marshall (University of Nottingham), Alice Angus, Giles Lane, Karen Martin, Orlagh Woods (Proboscis) and Lorraine Warren (University of Southampton).

Sensory Threads is part of EPSRC Creator (EP/G002088/1): New Research Processes and Business Models for the Creative Industries, http://www.creatorproject.org/.

Robin Fencott is funded by an EPSRC Doctoral Training Account Award.

8 REFERENCES

[1] Airantzis, D., Martin, K., Angus, A., Roussos, G., Lane, G. and Woods, O. “Snout: A mobile sensing platform”. Technical report, *Proboscis inIVA and Birkbeck College London* http://socialtapestries.net/snout/snout documentation.pdf (last accessed 20 Jan 2009), 2007.

[2] Benford, S., Crabtree, A., Flintham, M., Drozd, A., Anastasi, R., Paxton, M., Tandavanitj, N., Adams, M. and Row-Farr, J. “Can You See Me Now?” *ACM Transactions on Computer-Human Interaction* Vol. 13, No. 1, Pages 100133, 2006.

[3] Blaine, T and Fels, S. “Contexts of collaborative musical experiences” *Proceedings of the 2003 conference on New interfaces for musical expression* Montreal, Quebec, Canada, 2003.

[4] Bregman, A. *Auditory Scene Analysis: The Perceptual Organisation of Sound*. MIT Press, Cambridge, 1990.

[5] Bryan-Kinns, N. and Healey, P. G. T. “Exploring Mutual Engagement in Creative Collaborations” *Proceedings of Creativity and Cognition 2007* Washington, USA, 2007.

[6] Cox, C. “Invisible Cities: An Interview with Christina Kubisch” *Cabinet Magazine*, Issue 21 Spring 2006, ”Electricity”. http://www.christinakubisch.de/pdf/Kubisch_Interview.pdf (last accessed 1 April 2009), 2006.

[7] Davis, S. B., Moar, M., Cox, J., Riddoch, C., Cooke, K., Jacobs, R., Watkins, M., Hull, R. and Tom Melamed. “ere be dragons: an interactive artwork.” *Proceedings of the 13th annual ACM international conference on Multimedia* New York, USA, 2005.

[8] Diall, D. and Airantzis, D. “Urban tapestries - feral robot client architecture: Technical report of a feral robot prototype, 2nd generation. [Technical Report]” *Proboscis and Birkbeck College* http://socialtapestries.net/feralrobots/docs/feral-robot-client-arch.pdf (last accessed 20 Jan 2009), 2006.

[9] Fencott, R. “Interactive Music Using a Multi-Touch Cellular Automata” *Proceedings of (re)Actor3 the third international conference on digital live art* Liverpool, UK, 2008.

[10] Gaye, L., Mazé R. and Holmquist, L. E. “Sonic city: the urban environment as a musical interface.” *Proceedings of the 2003 conference on New interfaces for musical expression* Singapore, Singapore, 2003.

[11] McCartney, J. “Rethinking the Computer Music Language: Super Collider” *Computer Music Journal*. Winter 2002, Vol. 26, No. 4. MIT Press, 2002.

[12] McGookin, D. K. and Brewster, S. A. “Understanding Concurrent Earcons: Applying Auditory Scene Analysis Principles to Concurrent Earcon Recognition” *ACM Transactions on Applied Perceptions* Vol. 1, No. 2, October 2004.

[13] Sheridan, J. G. and Bryan-Kinns, N. “Designing for Performative Tangible Interaction” *International Journal of Arts and Technology. Special Issue on Tangible and Embedded Interaction* 2008.

[14] Smalley, S. ‘Spectromorphology: explaining sound-shapes” *Organised Sound* Vol. 2, No. 2. Pages 107-26. Cambridge University Press, 1997.

[15] Vickers, P. “Ars Informatica - Ars Electronica: Improving Sonification Aesthetics” *Understanding and Designing for Aesthetic Experience Workshop* Edinburgh, Scotland, 2005.

[16] Walther, B. K. “Reflections On The Methodology Of Pervasive Gaming”, *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology* Valencia, Spain , 2005.

[17] Wozniewski, M., Settel, Z., Cooperstock, J. R. “Large-Scale Mobile Audio Environments for Collaborative Musical Interaction” *Proceedings of the 2008 Conference on New Interfaces for Musical Expression* Genova, Italy, 2008.

[18] Wright, M., Freed, A. and Momeni, A. “OpenSound Control: State of the Art 2003” *Proceedings of the 2003 Conference on New Interfaces for Musical Expression*

THE KINEMATIC RUBATO MODEL AS A MEANS OF STUDYING FINAL RITARDS ACROSS PIECES AND PIANISTS

Maarten Grachten
Dept. of Computational Perception
Johannes Kepler University, Linz, Austria
maarten.grachten@jku.at

Gerhard Widmer
Austrian Research Institute for Artificial Intelligence, Vienna, Austria
Dept. of Computational Perception
Johannes Kepler University, Linz, Austria
gerhard.widmer@jku.at

ABSTRACT

This paper presents an empirical study of the performance of final ritards in classical piano music by a collection of famous pianists. The particular approach taken here uses Friberg and Sundberg's kinematic rubato model in order to characterize the variability of performed ritards across pieces and pianists. The variability is studied in terms of the model parameters controlling the depth and curvature of the ritard, after the model has been fitted to the data. Apart from finding a strong positive correlation of both parameters, we derive curvature values from the current data set that are substantially higher than curvature values deemed appropriate in previous studies. Although the model is too simple to capture all meaningful fluctuations in tempo, its parameters seem to be musically relevant, since performances of the same piece tend to be strongly concentrated in the parameter space. Unsurprisingly, the model parameters are generally not discriminative for pianist identity. Still, in some cases systematic differences between pianists are observed between pianists.

1 INTRODUCTION AND RELATED WORK

One of clearest manifestations of expressive timing in music is the final ritard, the slowing down toward the end of a musical performance to conclude the piece (or a part of it) gracefully. Several models have been proposed to account for the specific form of the ritard. These models typically come in the form of a mathematical function that describes how the tempo of the performance changes with score position. For example, Repp [9] found a quadratic function of score position to adequately describe IOI's measured in 28 performances. Honing [6] proposes a different kind of model, that consists in the combination of two computational models, one for tempo tracking, and one for rhythmic categorization. This model, rather than describing a single tempo curve, predicts the upper and lower boundary of the range of acceptable tempo curves for ritards. Since the two constituent models are intended to mimic perceptual processes involved in human listening, this can be called a *perceptual* model of expressive timing.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

Another kind of models has arisen from the analogy of expressive timing with physical motion [11, 10, 3]. For example, Todd [11] describes a model for expressive timing where tempo is treated as the velocity of a particle that moves under constant acceleration or deceleration, depending on its position. The physical position of the particle is equated to score position with respect to phrase boundaries. Also lead by the analogy with physical motion, Friberg and Sundberg [4], derive a model for the velocity of human motion, when halting after running. An evaluation of the runner's stopping movement in terms of aesthetic quality yields that stopping motion with approximately constant deceleration power is rated highest. From the assumption of constant deceleration power, they derive a model of tempo as a function of score time.

As pointed out in [5], models that are dependent only on score position are incomplete in the sense that they ignore any characteristics of the musical material that is performed. Also, the physical motion metaphor ignores perceptual and production aspects of music performance that are relevant to the shaping of musical tempo [2, 5].

Nevertheless, the kinematic rubato models described above predict the evolution of tempo during the final ritard quite accurately, when matched to empirical data [4, 11]. An additional advantage of the models is their simplicity, both conceptually (they contain few parameters) and computationally (they are easy to implement).

In this paper we study the variability in the final ritards of Chopin's Nocturnes performed by multiple famous pianists, using Friberg and Sundberg's kinematic model. Rather than validating the model on empirical data, we use the model to learn about the data (as in [12]). More specifically, we investigate whether the identity of the piece or the pianist is reflected in the parameters of the model. Given the simplicity of the two-parameter model, the existence of such an effect would be surprising, but would also shed some light on the interplay of personal interpretative freedom on the one hand, and performance practice and conventions on the other.

The data used for the study is described in section 2. Section 3 deals with the kinematic model and how it is applied to the measured data. Results are presented and discussed in section 4. Finally, section 5 states conclusions and remaining work.

2 DATA

The data used here consists in measurements of timing data of musical performances taken from commercial CD recordings of Chopin's Nocturnes. The contents of the data set are specified in table 1. We have chosen Chopin's Nocturnes since they exemplify classical piano music from the romantic period, a genre which is characterized by the prominent role of expressive interpretation in terms of tempo and dynamics. Furthermore, the music is part of a well-known repertoire, performed by many pianists, facilitating large scale studies.

Tempo in music is usually estimated from the interonset intervals of successive events. A problematic aspect of this is that when a musical passage contains few events, the obtained tempo information is sparse, and possibly unreliable, thus not very suitable for studying tempo. Therefore, through inspection of the score, we selected those Nocturnes whose final passages have a relatively high note density, and are more or less homogeneous in terms of rhythm. In two cases (Op. 9 nr. 3 and Op. 48 nr. 1), the final passage consists of two clearly separated parts, both of which are performed individually with a ritard. These ritards are treated separately (see table 1). In one case (Op. 27 nr. 1), the best-suited passage is at the end of the first part, rather than at the end (so strictly speaking, it is not a *final* ritard).

The data were obtained in a semi-automated manner, using a software tool [8] for automatic transcription of the audio recordings. From the transcriptions generated in this way, the segments corresponding to the final ritards were extracted and corrected manually by the authors, using *Sonic Visualizer*, a software tool for audio annotation and analysis [1].

3 METHOD

As mentioned in section 1, we wish to establish whether the specific form of the final ritard in a musical performance is dependent on the identity of the piece being played, or the performing pianist. We address this question by fitting a model to the data, and investigating the relation between the piece/pianist identity and the parameter values of the fitted model. We employ the kinematic model by Friberg & Sundberg [4], mainly for it's simplicity.

3.1 Friberg & Sundberg's kinematic model

The model is based on the hypothesized analogy of musical tempo and physical motion, and is derived from a study of the motion of runners when slowing down. From a variety of decelerations by various runners, the decelerations judged by a jury to be most aesthetically pleasing turned out to be those where the deceleration force is held roughly constant. This implies that velocity is proportional to square root function of time, and to a cubic root function of position. Equating physical position to score position, Friberg and Sundberg use this velocity function as a model for tempo in musical ritards. Thus, the model describes the tempo $v(x)$ of a ritard as a function of score position x:

$$v(x) = (1 + (w^q - 1)x)^{1/q} \tag{1}$$

The parameter q is added to account for variation in curvature (that is, the function is not necessarily a cubic root of position). The parameter w represents the final tempo, and was added since the tempo in music cannot reach zero. The model is designed to work with normalized score position and tempo. More specifically, the ritard is assumed to span the score positions in the range $[0, 1]$, and the initial tempo is defined to be 1.

The effect of the parameters w and q is illustrated in figure 1, which shows plots of tempo curves defined by the model for different values of w and q. Note that values of $q > 1$ lead to convex tempo curves, whereas values of $q < 1$ lead to concave curves. The latter is not expected to occur under normal circumstances, since tempo curves of ritards are typically convex. Note also that w determines the vertical end position of the curve.

3.2 Fitting the model to the data

The parameters of the model allow it to be fitted to ritards performed by particular pianists. As explained above, for this it is necessary to normalize the data. When normalizing the score position, it is important to make normalized position 0 coincide with the actual start of the ritard. Although in most cases there is a ritard instruction written in the score, the ritard may start slightly before or after this instruction. A manual inspection of the data showed that the starting position of the ritards strongly tended to coincide among pianists. For each piece, the predominant starting position was determined and the normalization of score positions was done accordingly.

When normalizing tempo, it is important to notice that normalizing should be done globally for the data set, rather

Pianist	Year	Op.9 nr.3 rit1	Op.9 nr.3 rit2	Op.15 nr.1	Op.15 nr.2	Op.27 nr.1	Op.27 nr.2	Op.48 nr.1 rit1	Op.48 nr.1 rit2
Argerich	1965			X					
Arrau	1978	X	X	X	X	X	X	X	X
Ashkenazy	1985	X	X	X	X	X	X	X	X
Barenboim	1981	X	X	X	X	X	X	X	X
Biret	1991	X	X	X	X	X	X	X	X
Engerer	1993	X	X	X	X	X	X	X	X
Falvai	1997	X	X	X	X	X	X	X	X
Harasiewicz	1961	X	X	X	X	X	X	X	X
Hewitt	2003	X	X	X	X	X	X	X	X
Horowitz	1957			X		X			
Kissin	1993					X	X		
Kollar	2007	X	X	X	X		X	X	X
Leonskaja	1992	X	X	X	X	X	X	X	X
Maisenberg	1995			X					
Mertanen	2001	X	X	X	X	X	X		
Mertanen	2002							X	X
Mertanen	2003							X	X
Ohlsson	1979	X	X	X	X	X	X	X	X
Perahia	1994			X					
Pires	1996	X	X	X	X	X	X	X	X
Pollini	2005	X	X	X	X	X	X	X	X
Richter	1968			X					
Rubinstein	1937	X	X	X	X	X	X	X	X
Rubinstein	1965	X	X	X	X	X	X	X	X
Tsong	1978	X	X	X	X	X	X	X	X
Vasary	1966	X	X	X	X	X		X	X
Woodward	2006	X	X	X	X	X	X	X	X
d´Ascoli	2005	X	X	X	X	X	X	X	X

Table 1. Performances used in this study. The symbol "X" denotes the presence of the corresponding combination of pianist/piece in the data set. The additions "rit1" and "rit2" refer to two distinct ritards within the same piece

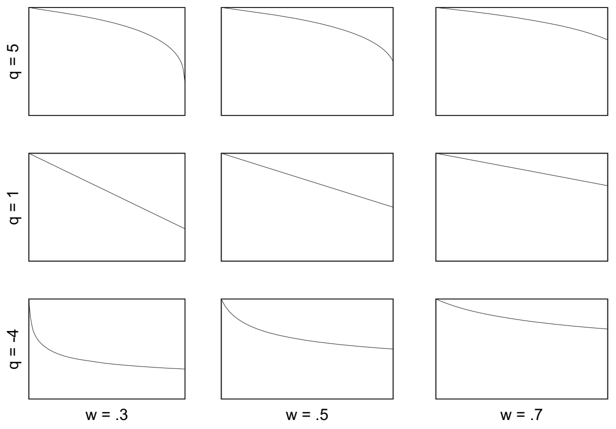


Figure 1. Ritards produced by the model using different values for the parameters w and q; In each plot, the x and y axis represent score position and tempo respectively, both in arbitrary units

than individually, since the latter would render the w parameter useless (the final tempo of every ritard would be 0). The result of global normalization is that the tempo value 1 corresponds to the highest tempo occurring in the data set, and the tempo value 0 to the lowest. Although this procedure maintains the relative scaling of the ritards, the majority of the ritards will not start with a tempo value of 1, whereas a constraint of the model is that it starts at tempo 1. An additional problem is that in some cases the first tempo value is not always the maximal value. This implies that shifting the data to make either the first value or the maximal value equal to 1, will result in a poor fit. To illustrate this, a problematic case is presented in the left plot of figure 2, where the maximal tempo value is not equal to 1. The fitted model is clearly a very poor approximation of the data. To alleviate these problems, an additional offset parameter is included added to the model while fitting. The right plot of figure 2 shows the same data with the fitted model using the offset parameter. Within its capabilities, the model now fits the data relatively well. Note that the offset parameter is only used for calibration purposes and is not regarded as a meaningful part of the model.

The model is fitted to the data by non-linear least-squares fitting through the Marquardt-Levenberg algorithm, using the *gnuplot* implementation [1] . The model fitting is applied to each performance individually, so for each combination of pianist and piece a value is obtained for w, q, and the root mean square of the error after fitting (this serves as a goodness-of-fit measure).

4 RESULTS AND DISCUSSION

The values obtained from fitting are displayed as a scatterplot on the two-dimensional parameter space q versus w, in

[1] The fitting must be done by numerical approximation since the model is non-linear in the parameters w and q.

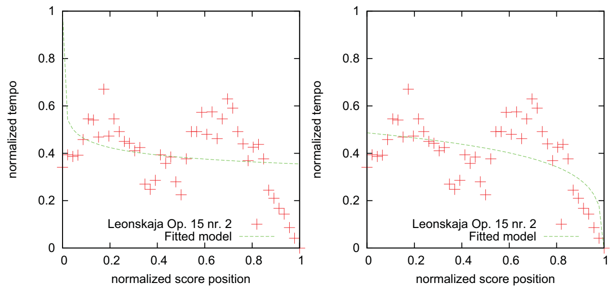


Figure 2. The necessity of an offset parameter for fitting the model (dashed line) to the data ('+' symbols). left: fitting without offset compensation; right: fitting with offset compensation (see text)

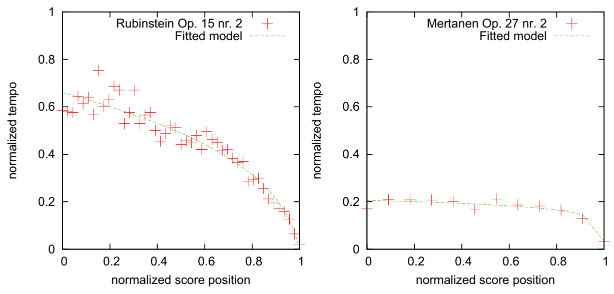


Figure 3. Two extremes of the effective parameter space; left: Rubinstein's ritard in Op. 15 nr. 2 (low w, low q); right: Mertanen's ritard in Op. 27 nr. 2 (high w, high q)

figures 4 and 5.[2] To facilitate the interpretation of points in specific locations of the plane, the reader is referred to figure 1, in which the relative location of the plots corresponds to the topology of the w-q plane.

In figure 4, the symbols group the data points by piece, such that the performances of the same piece by different pianists have identical symbols. The sizes of the symbols are proportional to the goodness-of-fit. That is, bigger symbols are represent a better fit of the model to the data, and are therefore to be considered more reliable than smaller symbols.

The scatter plot reveals a strong positive correlation between the w and the q parameter. In musical terms, this implies that the tempo decrease in deep ritards (low w) is more gradual (low q), whereas in shallow ritards (high w), the tempo decrease is more sudden, and postponed to the last notes of the ritard (high q). These two situations are illustrated by the ritards shown in figure 3.

Notable is also that virtually all performances correspond to q values above 3. This value (marked in the figure as a black horizontal line), corresponds to the model setting that mimics the motion of a physical body under constant braking power. This setting, together with $q = 2$ (constant braking force, assumed in [7] and [11]), is claimed by Friberg and Sundberg [4] to yield ritards that are aesthetically preferred by listeners. They suggest that this preference is due the fact that we are familiar with these conditions from our perception of physical motion. In contrast, the higher q values that are measured in the current study suggest model settings where braking power increases with time. Interestingly, the scatter plot shows a strong ridge close to $q = 4$, where the range above the boundary is highly populated, whereas the range below it is virtually empty. This means that, independent of the depth of the ritard, curvatures below $q = 4$ (approximately the curvature displayed in the left plot of figure 3) are very uncommon.

The distribution of the symbols indicate that, even if the data points of some pieces overlap, they are clearly clustered according to piece. For example, the performances of Op. 15 nr. 2, are all located in the lower ranges of w and q (deep and gradual ritards), whereas those of Op. 27 nr. 2 are all in the higher ranges (shallow and sudden ritards).

Another notable aspect of the results is that the ritards of Op. 48 nr. 1 rit. 2 (except for one, by Leonskaja) are played with various depths (w), but always with low curvature (q).

Figure 5 shows the same data, but labeled according to pianist. In this case clustering is less apparent from the plot. In part this may be due to the amount of different pianists (and thus symbols) displayed in the figure. However, since the data shows a considerable clustering along piece, a strong clustering along pianists is not to be expected. Still, upon more detailed inspection, the w-q plane conveys some differences between pianists.

Firstly, some pianists tend to concentrate in distinct areas of the w-q plane. This is the case for Leonskaja and Vasary. Their performances are displayed jointly in figure 6. Note that the pianists are almost separable based on their w and q coordinates.

As a second example of differences between pianists, consider the performances of Rubinstein and Pollini (figure 7). Rubinstein's w-q coordinates span a much larger part of the plane, suggesting that he plays ritards in a more diverse ways, whereas Pollini's coordinates are concentrated in a smaller area, suggesting a more uniform way of playing ritards. It is interesting to note that the relative locations of the pieces are roughly the same for Rubinstein and Pollini.

5 CONCLUSIONS AND FUTURE WORK

In this study we have used a kinematic rubato model [4] to investigate the performance of final ritards in Chopin's Nocturnes, played by 25 pianists. To our knowledge this is the first application of the model to data gathered from famous pianists. Studying the value range of model parameters that represent the measured ritards, we found that there is a strong positive correlation between the depth of ritards

[2] The figures are best viewed in color

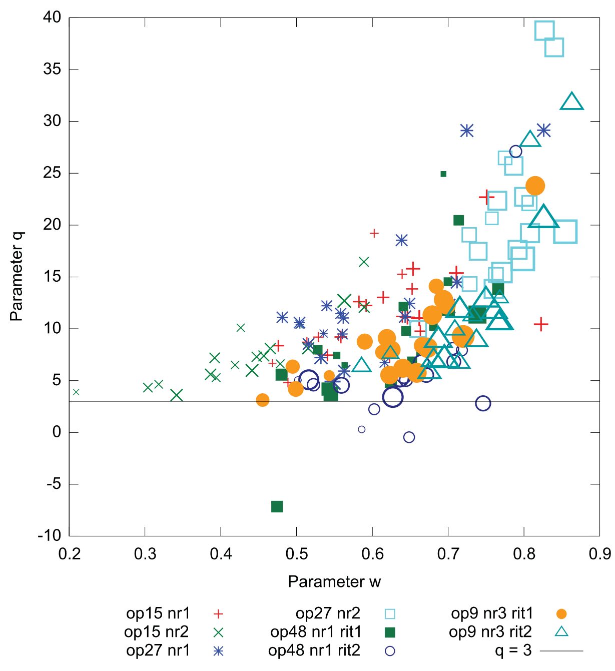


Figure 4. Distribution of rubato model parameter values over pieces; the size of the symbols is proportional to goodness of fit of the parameters to the data

(w) and their curvature (q). In addition, fitted q values are only sporadically below 4. This contrasts with earlier studies stating $q = 2$ and $q = 3$ as plausible settings [4, 7, 11].

Furthermore, ritards of the same piece by different pianists tend to be concentrated in the w-q plane, suggesting that the musical material being played is an important factor in the determination of the depth and curvature of the ritard. Although in general the model parameters are not discriminative for pianists, in some cases the differences in the parameter ranges for individual pianists are considerable. In order to make more decisive claims about pianist-specific differences however, more performances per pianist are needed.

An important issue that we have not addressed in this paper is that in many cases the structure of the ritards are more complex than the model can accommodate. More specifically, the measured tempo data in addition to a simple tempo decrease often shows internal structure that seems to be related to rhythmical patterns or motivic grouping in the music. This affects the goodness-of-fit of the model, and shows the need for a more elaborate modeling approach, either by using more sophisticated models (such as the one proposed by [6]), or by an analysis of the residual information after the model has been fitted.

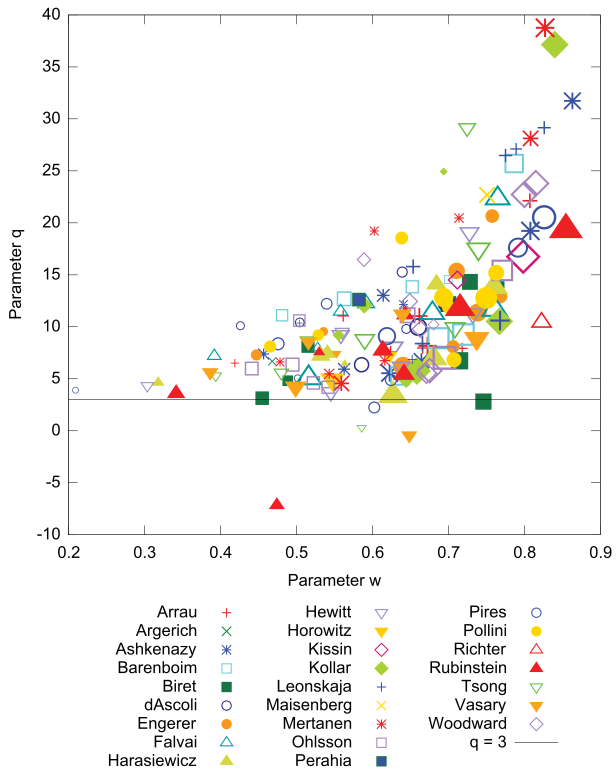


Figure 5. Distribution of rubato model parameter values over pianists; the size of the symbols is proportional to goodness of fit of the parameters to the data

Acknowledgments

We wish to thank Werner Goebl and Bernhard Niedermayer for their help in the acquisition of the timing data from the audio recordings. This work is funded by the Austrian National Research Fund (FWF) under project number P19349-N15.

6 REFERENCES

[1] CANNAM, C., SANDLER, M., AND BELLO, J. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of the 7th International Conference on Music Information Retrieval* (2006), ISMIR.

[2] DESAIN, P., AND HONING, H. Physical motion as a metaphor for timing in music: the final ritard. In *Proceedings of the 1996 International Computer Music Conference* (San Francisco, 1996), ICMA, pp. 458–460.

[3] FELDMAN, J., EPSTEIN, D., AND RICHARDS, W. Force dynamics of tempo change in music. *Music Perception 10*, 2 (1992), 185–204.

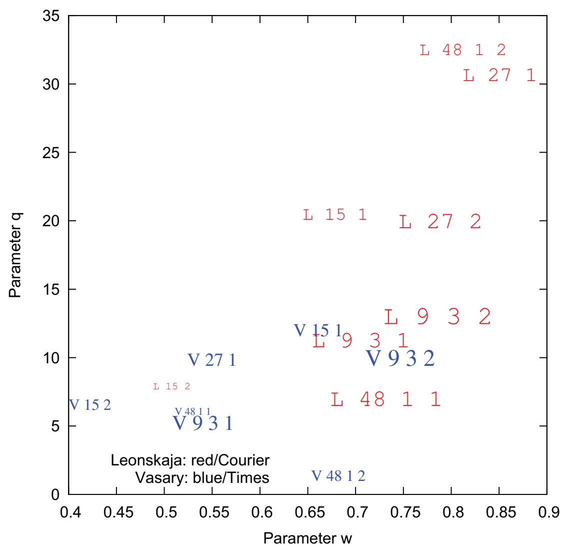


Figure 6. Two pianists occupying different areas of the parameter space; the size of the labels is proportional to goodness of fit of the parameters to the data; “L 48 1 1” = “Leonskaja, Op. 48, nr. 1, rit 1”, etc.

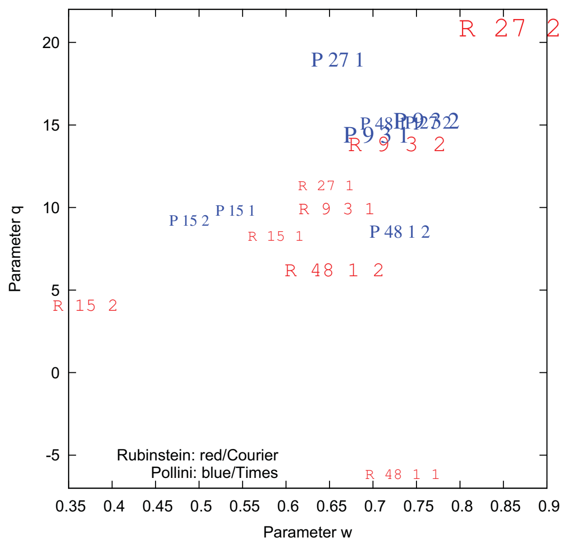


Figure 7. Two pianists spanning overlapping, but differently sized areas of the parameter space; the size of the symbols is proportional to goodness of fit of the parameters to the data

[4] FRIBERG, A., AND SUNDBERG, J. Does music performance allude to locomotion? a model of final ritardandi derived from measurements of stopping runners. *Journal of the Acoustical Society of America 105*, 3 (1999), 1469–1484.

[5] HONING, H. When a good fit is not good enough: a case study on the final ritard. In *Proceedings of the ICMPC* (2004), pp. 510–513.

[6] HONING, H. Is there a perception-based alternative to kinematic models of tempo rubato? *Music Perception 23*, 1 (2005), 79–85.

[7] KRONMAN, U., AND SUNDBERG, J. Is the musical ritard an allusion to physical motion? In *Action and Perception in Rhythm and Music*, A. Gabrielsson, Ed., no. 55. Royal Swedisch Academy of Music, 1987, pp. 57–68.

[8] NIEDERMAYER, B. Non-negative matrix division for the automatic transcription of polyphonic music. In *Proceedings of the 9th International Conference on Music Information Retrieval* (2008), ISMIR.

[9] REPP, B. H. Diversity and commonality in music performance - An analysis of timing microstructure in Schumann’s “Träumerei”. *Journal of the Acoustical Society of America 92*, 5 (1992), 2546–2568.

[10] SUNDBERG, J., AND VERRILLO, V. On the anatomy of the retard: A study of timing in music. *Journal of the Acoustical Society of America 68*, 3 (1980), 772–779.

[11] TODD, N. The dynamics of dynamics: A model of musical expression. *Journal of the Acoustical Society of America 91* (1992), 3540–3550.

[12] WINDSOR, W. L., AND CLARKE, E. F. Expressive timing and dynamics in real and artificial musical performances: using an algorithm as an analytical tool. *Music Perception 15*, 2 (1997), 127–152.

PV STOCH: A SPECTRAL STOCHASTIC SYNTHESIS GENERATOR

Luc Döbereiner
Institute of Sonology, The Hague
luc@doebereiner.org

ABSTRACT

PV Stoch is a phase vocoder (PV) unit generator (UGen) for SuperCollider. Its objective is the exploration of methods used in "non-standard synthesis", especially in Dynamic Stochastic Synthesis (Xenakis), in another domain. In contrast to their original conception, the methods are applied in the frequency domain. This paper discusses some of the compositional motivations and considerations behind the approach, it gives a description of the actual synthesis method and its implementation, as well as a summary of the results and conclusions drawn.

1 INTRODUCTION

PV Stoch is a generator for frequency domain stochastic synthesis. After having worked at the generalization of "non-standard" synthesis[4], the development of *PV Stoch* was driven by an interest in extending stochastic synthesis; an interest in testing the transferability of its principle workings and reapply them in another area, the frequency domain. In this paper, we will discuss the first result of this investigation.

1.1 Transferability

Iannis Xenakis used stochastic functions for the generation of sound after having used them on a higher-level before. They have been compositional tools to him. The step to synthesize the sounds themselves using probabilities, as well as the introduction of them in musical composition itself, follow the belief that a method which has successfully been employed on one level or one domain may successfully be transferred to another.

> Any theory or solution given on one level can be assigned to the solution of problems of another level. Thus the solutions in macrocomposition (programmed stochastic mechanisms) can engender simpler and more powerful new perspectives in the shaping of microsounds.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

1.2 Overview

Firstly, the synthesis method itself is described. The individual parameters are presented, as well as brief description of their aural effects. It shall be noted that the descriptions are somewhat simplified and most of all, the control parametrization is not congruent with their multi-layered perception. Although, the perceptible effects of each of the parameters is briefly addressed, their inter-dependencies and trans-active nature is far too complex to be properly outlined here.

Subsequently, we will give attention to *PV Stoch*'s relation to the "non-standard" synthesis approaches and thereby place it in a historical and theoretical context. Although *PV Stoch* does not fulfill all the criteria to be classified as "non-standard", we are trying to demonstrate that it does indeed comply with and even extend some fundamental notions present in these approaches.

Furthermore, some of the challenges and features we have encountered in the practical work with the generator are discussed by means of the description of a 96-channel composition by the author which was realized exclusively with *PV Stoch*.

2 IMPLEMENTATION

PV Stoch is a phase vocoder UGen for SuperCollider (J. McCartney). SuperCollider features a robust and efficient framework for the design of frequency domain operators. As the development of *PV Stoch* has been a rather experimental investigation, SuperCollider's flexibility and real-time controllability proved to be crucial. The implementation framework is straight forward and the UGen can be combined with a variety of already existing UGens and control mechanisms.

PV Stoch takes the following parameters, which are explained below. Except of **nBps** and **lambda**, which only have effect during the initialization, all parameters are dynamically controllable:

2.1 Basic Functionality

PV Stoch is a frequency domain stochastic synthesis generator. Although, it operates on a FFT buffer, it does not process an analyzed sound, but rather synthesizes sound with-

```
PV_Stoch(buffer, nBps, lambda,
         phaseSwitch, specDec,
         interpBase, range,
         offset, deviation)
```

Figure 1. The parameters of *PV Stoch*

out input source.[1] The created spectra have an envelope, or spectral contour, which is constructed of interpolated breakpoints. The positions of these breakpoints deviates from frame to frame, as the time domain breakpoints deviate from cycle to cycle in Xenakis's Dynamic Stochastic Synthesis.

When the UGen is initialized, it generates an initial spectral envelope. The distribution of the breakpoints follows controllable probabilistic laws – an exponential random distribution – and the interpolation function may vary over time. Frame by frame the positions of the breakpoints, and thereby the spectral shape, deviate. The amount of deviation is dynamically controllable. The created spectrum can also be dynamically frequency shifted or stretched, which are familiar frequency domain techniques. Furthermore, the phase spectrum generation has three states and it can be interpolated between them.

2.2 The Envelope

The initial envelope has a big effect on the resulting sound. Initially, its shape is determined by three parameters: the number of breakpoints (**nBps**), a random variable controlling the spread of an exponential random distribution which determines the horizontal (frequency) position of the breakpoints (**lambda**), and the base of the interpolation function (**interpBase**). If the base is 1, the interpolation is linear, if it is bigger or smaller than 1, the interpolation is exponential, resulting in concave and convex curves respectively.

A higher number of breakpoints (**nBps** > 20) results in more defined and more complex spectra, a lower number creates sounds similar to more simply filtered noise. If **lambda** is smaller (**lambda** $<= 1.0$), the resulting sounds are more distinguishable and the deviations are clearer, if the random variable is greater, the sound becomes more static, the changes less drastic. A more concave interpolation curve (**interpBase** < 1.0) articulates the attenuated frequency regions more clearly, whereas more convex shapes create blurrier noise regions.

The vertical (amplitude) positions of the breakpoints are determined by a beta random distribution. Additionally, the magnitudes of the whole spectrum are also scaled by an exponentially decreasing shape, whose steepness is variable (**specDec**).

[1] One exception is the phase, which is explained in 2.5.

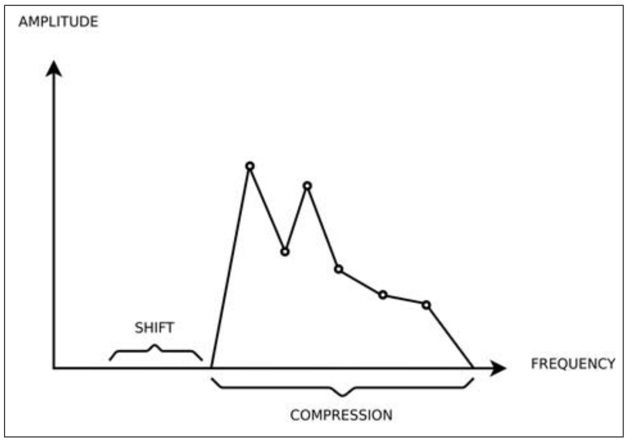


Figure 2. Shifting and compressing the created spectrum

2.3 Shifting and Stretching

Figure 2 illustrates two additional – and well known – operations which can have a drastic effect on the sound: frequency shifting and stretching/compressing. As can be seen, the entire spectral shape can be shifted (**offset**) along the frequency axis (in both directions) and stretched or compressed (**range**). Since the spectral shape is expressed by interpolated breakpoints whose position along the frequency axis does not need to coincide with the frequency grid imposed by the frame size, shifting and stretching or compressing occurs smoothly without making the frequency resolution audible. The shifting and stretching is similar to techniques presented by among others Trevor Wishart[8]. Although, Wishart's approach is regarded as "standard" synthesis, it is surely a compositionally motivated approach to sound synthesis.

2.4 Deviation

Figure 3 shows the deviation principle. The breakpoints deviate frame per frame from their previous position by a random amount, the maximum of which is controlled by the parameter **deviation**. Thus, similar to Dynamic Stochastic Synthesis', the breakpoints undergo random walks, however, only in their vertical position (amplitude).

2.5 Dealing with the Phase

There are three basic settings for the phase. It can be interpolated between them. The phases can be set zero, in which case the results are closer to additive synthesis using sine waves, the phase values can be generated randomly, which creates sounds closer to filtered noise, or they can be derived from an input source. Although *PV Stoch* has not been designed to process analyzed input sources, this phase setting was introduced in order to add "articulation" stemming

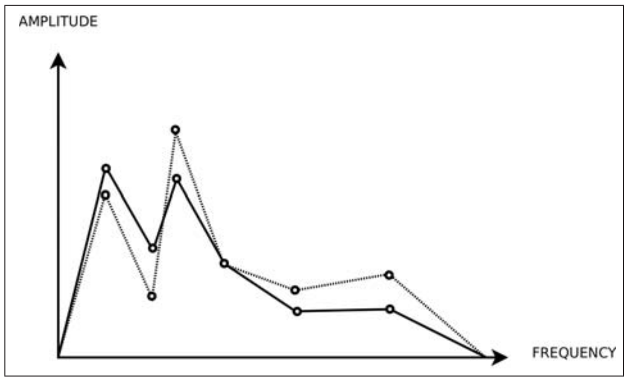


Figure 3. The deviation of breakpoints in two successive frames

from another source. It was primarily used with frequency modulated impulses.

3 *PV STOCH* AS "NON-STANDARD" SYNTHESIS

The starting points for the development of *PV Stoch* have been the so-called "non-standard" sound synthesis[6] approaches, especially Iannis Xenakis's Dynamic Stochastic Synthesis. The systems subsumed under the term "non-standard" have in common that they do not adhere to any superordinate acoustic models. [2] Instead, the models of sound are derived from compositional models. Sound synthesis is understood as the development of processes organizing the low-level units, as "microtemporal *compositional* processes."[3] *PV Stoch* takes up this idea of deriving higher level structural properties from the description of lower level processes. Here, the distinction between sound and music is blurred.

For different reasons, however, the "non-standard" approaches rejected the frequency domain. Xenakis heavily criticized the use of harmonic analysis for the synthesis of sound. The results he deemed uninteresting, the approach "inadequate". He ascribed the problems to the "synthesis by finite juxtaposed elements"-principle. "It is as though we wanted to express a sinuous mountain silhouette by using portions of circles"[9], he writes. Curiously, Xenakis's UPIC system is based on the very principle he had been criticizing so vehemently, it is a form of additive synthesis.

Perhaps, due to its mathematical nature and popularity among the more simulating sound synthesis methods, the frequency domain was considered inappropriate for a uniquely digital music. It seemed to be a concept which was not very well suited for answering the question, "what means of expression are idiomatic to computers?"[7] For Xenakis, the reason for his rejection may rather have been his association of additive synthesis with the electronic music of the Cologne studio.

[2] Oddly, the time domain is usually not considered an acoustic model in the descriptions of "non-standard" synthesis.

In contrast to the lion share of the research done in sound synthesis, the "non-standard" approaches are truly experimental. The interest does not lie in "trying to reconstruct a sound based on analytic data", but in "composing sound using musical procedures."[1] They can be seen as explorations of compositional representations of sound. In Koenig's SSP, amplitude and time values are elevated to the level of musical unit elements. Surely, problems arise from that, because their treatment "would require parameters to have a recognizable identity"[2] *PV Stoch* continues to ask the question "what is the minimum of logical constraints necessary for the construction of a musical process"[9], but it changes the underlying form of representation, also in hope of creating elements with a more recognizable identity.

In fact, the so-called "non-standard" sound synthesis approaches are all characterized by the use of concepts which are initially alien to the description of sound. With SSP, for example, G.M. Koenig uses methods which he had developed for instrumental composition for the structuring of audio sample values. Similarly, Paul Berg's programs ASP and PILE derive musical and sonic relationships from instructions present in programs for numerical computation.

There is, thus, an element of transfer, of reapplication, in "non-standard" synthesis. The sound organizing principles arise from a compositional interest, the compositional idea is embodied in the 'sound material', it is not imposed on it.

In this line of thought, *PV Stoch* can be seen as an attempt at creating frequency domain "non-standard" synthesis. Although, this may stand very much in contrast to the rejection of superordinate acoustic models, it follows Xenakis's idea of transferability of concepts.

Instead of aiming at the (re)creation of specific sounds, it is rather a search for the remains of an organization principle, for the traces the prinicple may leave in the sound and through another representation.

4 AN APPLICATION: *SPACE STUDY 1*

Space Study 1: Order From Noise is a fixed medium (tape) piece for 96 independent channels which was composed by the author in 2009. For the sound production *PV Stoch* was used exclusively. Due to the immense amount of data and coordination necessary for the independent composition of 96 tracks, it became unavoidable to automate many processes in the production of the piece. A consequence of the automation was the necessity of clear distinctions, of parametric configurations on the one hand and strategies of transitions and transformations of the other hand.

The piece consists of four sections which undergo a similar macro-level development, there can be seen as variants of a common higher-level description. For the most part, the synthesis settings are the variable element among the sections. The four sections are briefly described:

1. Impulses whose frequencies follow exponential curves, and ranging from 1 to 100 Hz thus creating rhythm and pitch, serve as the input for the phase values. *PV Stoch* initially creates ”resonances” and gradually the phases become more random, the impulses are thus replaced by noise and the ”resonances” become the central sound itself.

2. The phases alternate between noise, impulses, and zero, thus creating clearly distinguishable types of events. Instead of gradual change and slow transitions, the different timbres are clearly opposed to each other.

3. Blocks of quickly deviating bursts form gestural units. The deviation is high, lambda is low.

4. Finally, the phases are set to zero. The section is rather soft in volume and the spectra act as clusters, slowly shifted in frequency and space.

Each of the sections creates its own timbre space. When the phases are derived from impulses, the generator creates ”resonances”, when it is random, the deviation is high, and lambda is low, the random walks are most audible and the output strongly resembles time-domain stochastic synthesis. Since, all the timbre states are outcomes of the same process, they can easily be related to each other.

5 FURTHER WORK

Peter Hoffmann writes about Xenakis’s GENDYN:

> The key idea of stochastic synthesis is its non-linear waveshaping, where the waveshaping function changes stochastically from period to period. Consequently, it is not the waveform as such that defines the aural result [...] but rather the dynamic behavior of its deformation over time. [5]

PV Stoch behaves similarly. It is not the specific spectrum created but rather the way it changes from frame to frame that determines the aural quality of the result. The behavior is also what is most controllable. Since the initial envelope has a big effect on the resulting sound and since it is not completely predictable from the parameter settings, several instantiations of the same parametric configurations can result in a great variety of different sounds. The generator is thus not very well suited for the purposeful creation (simulation) of pre-conceived sounds. By controlling deviations, “spectral definition”, pitch and noisiness, types of sounds and types of sonic behaviors can be created.

Several improvements and additions suggest themselves and need to be tested regarding their musical effectiveness. The deviation may be further refined. Since the dynamic behavior of the system is the perceptibly most significant element, it should be further developed.Similarly to Xenakis’s models, second order random walks could be included and the breakpoints could move on the frequency axis as well. Furthermore, the number of breakpoints should be dynamically variable. The impact of different random distributions on the various stochastic processes should be investigated.

6 REFERENCES

[1] Berg, P., Rowe, R., Theriault, D. “SSP and Sound Description”, *Computer Music Journal*, Vol. 4, No. 1, 1979.

[2] Berg, P. “Composing Sound Structures with Rules”, *Contemporary Music Review*, Vol. 28, No. 1, 2009.

[3] Di Scipio, A., Prignano, I. “Synthesis by Functional Iteration. A Revitalization of Nonstandard Synthesis”, *Journal of New Music Research*, Vol. 25, 1996.

[4] Döbereiner, L. “CompScheme: A Language for Composition and Stochastic Synthesis”, *Proceedings of the 5th Sound and Music Computing Conference*, Berlin, Germany, 2008.

[5] Hoffmann, P. “The New GENDYN Program”, *Computer Music Journal*, Vol. 24, No. 2, 2000.

[6] Holtzman, S. R. “An Automated Digital Sound Synthesis Instrument”, *Computer Music Journal*, Vol. 3, No. 2, 1979.

[7] Holtzman, S. R. *Digital Mantras*, MIT Press, Cambridge, 1994.

[8] Wishart, T. *Audible Design*. OTP Ltd, 1994.

[9] Xenakis, I. *Formalized Music*. (Rev. ed). Pendragon Press, New York, 1992.

TOWARDS AN EXPERIMENTAL PLATFORM FOR COLLECTIVE MOBILE MUSIC PERFORMANCE

Koray Tahiroğlu
Helsinki University of Technology,
Department of Signal Processing and Acoustics,
Espoo, Finland
Koray.Tahiroglu@tkk.fi

ABSTRACT

This paper presents an experimentation of an interactive performance system that enables audience participation in an improvisational computer music performance. The design purports an improvisation tool and a mechanism by involving collective mobile interfaces. It also provides a design of an adaptive control module in parts of the system. Designing a collaborative interface for an easy to use and easy to control everyday-life communication tool allows for an audience to become more familiar with the collaboration process and experience a way of making music with a mobile device. The role of the audience is critical, not only for the design process of the system, but also for the experience of such experimental music.

1 INTRODUCTION

Mobile interfaces aid the participatory augmentation in collaborative computer music performances. They do not only make use of the technology but also provide an opportunity to emphasize the role of social communication and its relationship to interaction in human actions and reactions towards music.

Collaborative music making requires event participation in a musical context and it includes all aspects of human musical interaction - voluntary and involuntary actions [1, 2]. Voluntary actions involve participants consciously forming decisions about musical activity, both in listening and playing modes. Some human actions on the other hand, are performed without conscious comprehension, and it can be argued that it is hard to draw the line between voluntary and involuntary actions, especially in an activity like music making. In music, these actions result in exchanging musical events and gestures, which in turn bring about a shared interaction and experience framework.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

The level of interaction in a music performance can be estimated by whether the voluntary actions take an active role or they are reduced to a passive role of just being a member of the audience. In the context of experience design, interactivity is comprised of many attributes, such as feedback, control, creativity, adaptivity, productivity, communications and conversational experiences [3]. These attributes draw the line from low-level to high-level interaction in the interactivity spectra. Interactive performance systems can enable a high level of interaction by providing new practices for voluntary actions in order to achieve a high level of feedback, communication and audience control in a computer music performance. Creativity and productivity achieved in the moment of participation and the adaptive ability of the interactive system will also increase the level of interaction during the performance. This will result in a shared collective musical experience and enhance musical satisfaction.

This paper introduces the current state of experimentation of the *Control Augmented Adaptive System for Audience Participation (CAASAP)*, which is a part of the strategies developed for a collective mobile music performance. CAASAP is a facilitator of dynamic social interaction, controlling group communication and its relation to the improvisation of music. The core novelty of the system is, that it cumulates the participants' control-data in a centralized network and grounds it as a source to generate overall control parameters for improvised music. Figure I illustrates the ideas for the performance modules and interaction model of the system. In the following sections, the overview of the current work done in developing strategies for system modules is presented and the interface and adaptive module plan of action are introduced in detail. Paper concludes with a presentation of outcomes and indicates the future development of the CAASAP system.

2 BACKGROUND IN COLLECTIVE MOBILE MUSIC PERFORMANCES

CAASAP can be categorized as a small-scale system based on Weinberg's taxonomy [4]. Mobile collective interfaces

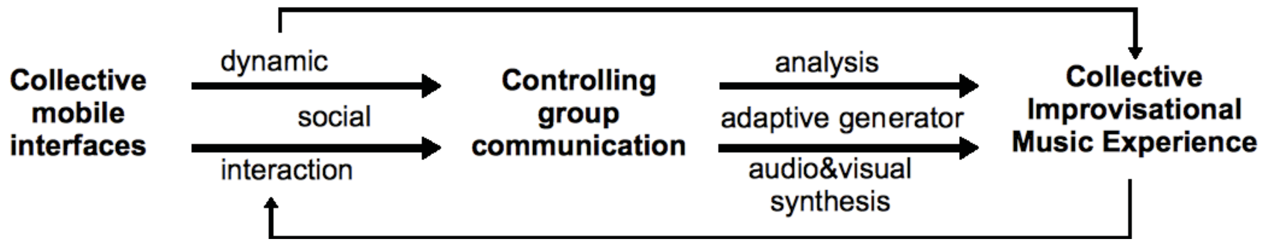

Collective mobile interfaces
dynamic
social
interaction
Controlling group communication
analysis
adaptive generator
audio&visual synthesis
Collective Improvisational Music Experience

Figure 1. Modular structure of interaction in CAASAP performances.

give more possibilities to increase the amount of participants in collaborative music performances. However, designing such a system to include audience participation requires bringing together different technical platforms and managing the complexity of their integration. In literature, the collaborative, improvisatory and technological aspects of mobile music performances are introduced as alternative approaches for this integration in mobile music systems.

Golan Levin's mobile music piece *Dialtones (A Telesymphony) (2001)* [1] is a large-scale collaborative performance that brings forth a different approach to the performative role of the audience. Low-level interaction achieved in terms of decisions made by the audience; however, using mobile phones as means of musical instruments, by hacking the dialtones of the audience's mobile phones and performing a pre-composed piece by ringing and dialing, is a remarkable event for the use of mobile devices in musical contribution.

A mobile phone, being a significant communication tool for the majority of people, gives the opportunity to interface its everyday-life practice for a collaborative musical experience. *Call in the Dark Noise (2006)* is a performance that provided the audience with a responsive environment for participating in an act of musical improvisation [1]. In this performance, the interactive performance system allowed the audience to use their mobiles phones as a musical instrument by sending SMS messages. The interactive system altered SMS messages into sound structures and created respond text messages. Using an everyday communication medium as a musical instrument can make the audience feel comfortable about participation and improvisation by sending SMS messages can create an exciting framework for collaborative music making. Today, the technology of mobile devices can support more possibilities than only receiving and responding to SMS messages during a live performance.

As the technological aspects are developed further, new capabilities and tools in mobile phone technologies have begun to provide alternative feedback mechanisms. Tapping on a touch screen, tilting the mobile device, multi-touch interfaces change the way we interact with a mobile phone. Moreover, they create a new gestural dictionary within the context of interactive gestures [2]. Nokia N-series phones, Apple iPhone and iPod touch mobile devices are the leading new alternatives that support these types of gestures. Interfacing new mobile functions as expressive musical instrument is an interesting prospect. Mobile devices provide alternative possibilities for experimenting with new music making processes. MoPhO is the Mobile Phone Orchestra of CCRMA using mobile phones as musical instruments in a larger scale performance [5]. They are using not only new feedback mechanisms that come with the new series of mobile phones, but they are also using the advantage of enriched computational possibilities. However, the computational possibilities for audio synthesis in mobile phones are still limited compared to other portable devices.

MoPhO compositions are performed through performers' actions on mobile phones guided by the conductor of the performance. These compositions can be interpreted as freely composed pieces. Conducted improvisation in music or the performance of pre-composed pieces might limit the type of free communication that could be achieved in a free or structured improvisation performance. The overall control structure in CAASAP does not scale down participants' collective activities as the performers of a pre-composed piece, instead, it supports them developing their musical ideas and activities in a real-time performance.

Malleable Mobile Music Engine in a broader scope shows similarities with CAASAP as it serves as a platform for collaborative music making through mobile wireless networks [6]. Malleable Music encourages participatory activity by facilitating a system that detects involuntary gestures and the remote geographic location of the participants. In contrast, CAASAP is a facilitator for audience participation where the improvised music content is generated through the voluntary actions of the participants.

3 OVERVIEW OF CAASAP

CAASAP is based on independently developed modules; their interaction forms the characteristics of the interactive system and its performance. The system consists of interface, registration, adaptive control, and audio & visual syn-

[1] `http://www.flong.com/`

[2] `http://intertactivegestures.com`

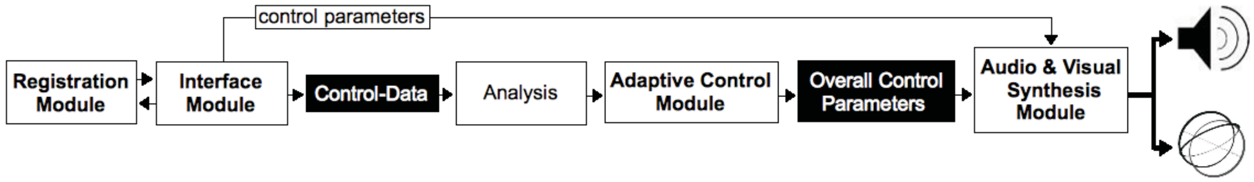


Figure 2. Block diagram of the system modules.

thesis modules. In addition to receiving and collecting control parameters, these modules also analyze control-data and perform audiovisual synthesis allowing these actions to be interfaced within a mobile device.

The modular structure of the system requires sequenced action of particular modules. Figure 2 shows the interconnected system modules. Registration module represents the initial act that alters participation into action. This module is in charge of creating a local network and maintaining the requirements for real-time connections of the participants. It includes assigning the server IP for participants to access and join the local network. When all the participants are connected, then the registration module uploads the interface module to the participants' mobile devices. The current version of the registration and interface modules are enabled by mrmr technology [3] . Mrmr is an ongoing research project to develop a standardized set of protocols and syntax conventions to control live installations and multimedia performances. This technology, based on the Open Sound Control (OSC) and other open standards, makes it possible to use mobile devices as controllers in audio-visual performances.

Interface module is based on the design of a collective mobile interface that enables interactive gestures and sends parameter changes to the audio & visual synthesis module. This module modifies main control parameters, including instrument's ID number, volume level, direction of the audio stream, reverb level, noise level and text messages. Interface module also sends the state changes of participants' interactive gestures to the adaptive control module for further analysis of the control-data.

The adaptive control module will analyze the control-data and it will generate overall control parameters for the audio synthesis module. As a result of the different modes of the participants' musical activities, this module will generate alternative improvisation models during the collective improvisation performance. Section 5 introduces the analysis and generative strategies that will be implemented in adaptive control module in detail.

Audio & visual synthesis module receives control parameters and maps them onto control values of the digital instruments. The changes of the direction of the audio stream will be also used as visualized representation of the participants' location in the performance space. Three-axis (x,y,z) accelerometer's control data will be visualized as three circles for each participant [7]. The rotation speed of each circle will represent the participant's speed of the action on the particular axis.

[3] `http://poly.share.dj/projects/#mrmr`

4 INTERFACES IN CAASAP

In the course of the development of CAASAP, several available technologies have been studied and practiced. Figure 3 illustrates the UI every participant operates on. The interface is made by using mrmr protocol and tools. The four push-buttons on the top enable/disable up to four shared instruments (see section ”Audio Synthesis” for more discussion). At this instant, the first instrument is selected and the values for reverb, noise and volume parameters are assigned. The state changes of the interface are sent through OSC protocol. The current version of mrmr technology supports one-way OSC communication, which does not enable the system to send feedback based on state changes to the interface module. On the other hand, this interface module supports text message affordance, which opens up another communication channel for participants and possible sonification strategies for the audio synthesis module.

In the process of developing the interface module, RjDj application [4] has also been experimented with. RjDj is a technology that uses sensory input to generate and process embedded scenes for iPhone and other mobile devices. RjDj technology enables Pure Data [5] for processing live data taken in mobile devices. The overall system architecture of the CAASAP has been developed by using the Pure Data environment; therefore, RjDj gives more possibilities to integrate control-data with other CAASAP modules. Accelerometer and touch screen sensor data of the mobile device is available and can be accessed with RjDj application. Moreover, it takes in the sensory input from microphone device, which makes it possible to process the audio as a sensory data in mobile devices. In order to improve CAASAP performance, some part of the analyzes can be embedded in mobile devices through RjDj application and resulted event data can be transfered for further use in adaptive and audio & visual modules. At the moment RjDj application also only supports one-way OSC data stream; however when

[4] `http://rjdj.me/`
[5] `http://www.puredata.info/`

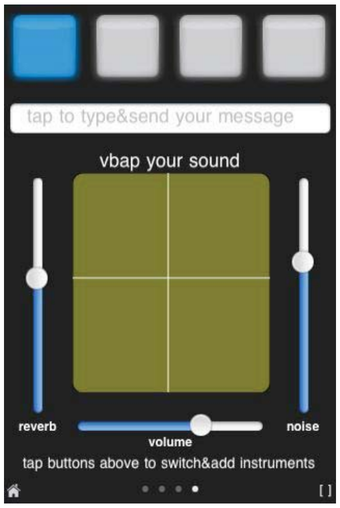

tap to type&send your message
vbap your sound
reverb
volume
noise
tap buttons above to switch&add instruments

Figure 3. The interface module enabled on mrmr technology.

RjDj will support OSC stream in both directions, then it will be possible to control audio device features of the mobile devices in more advanced level and implement new features in CAASAP.

5 CONTROL PARAMETERS FOR OVERALL IMPROVISATION

Improvisation practices lead participants to experience a way of music making, which takes form in the act of performing in a collaborative environment. In this participation, exchanged gestures play an important role together with the exchanged musical events and together they form the participant's musical activity. CAASAP will collect these exchanged events similar to the way that the control-data of the interactive gestures will be received. Accelerometer control-data of the participants will be analyzed in order to generate certain improvisation models to negotiate with the participants, moreover to support or provoke them to create new musical ideas within the group dynamics of the collective improvisation process. The overall improvisation parameters will be generated in this process to control the group communication, aiming to balance the flow of the improvisation; however there will be no pre-defined structure as there is in a pre-composed music piece.

The current strategies for analysis of the control-data is designed focusing on the performative aspects of music in the performance space. This led the design ideas to involve adaptive features based on the group dynamics of the collective improvisation. Concentrating more on the group dynamics and adaptability, swarm patterns came forward as a model for generating overall improvisation music materials as swarms in real-life organize themselves flock patterns.

The method of the control-data analyses is also structured on the basic rules that are governing the interactions between neighboring particles in swarm; *1) if apart, move closer (cohesion), 2) if too close move apart (separation), 3) attempt to match velocities (alignment)* [8]. The analysis structure will determine whether the participants' gestures move closer, move apart or attempt to match their velocities. Resulting rule will make the decision in adaptive control module and it will generate further parameters for audio synthesis. If the gestures will attempt to move closer or further apart than the system's thresholds, the overall control parameters will be generated to respond as negative feedback to the collective improvisation. If the velocities of the gestures will attempt to match, then the module will respond as positive feedback. David Borgo points out that positive feedback forms the recruitment and reinforcement activities and negative feedback keeps the balance by causing unexpected occurrences in improvisation [9]. In addition to positive and negative feedback, CAASAP will determine random choices as well not to result too much regular and repetitive moods for the improvisation. Resulting overall improvisation will be formed based on direct and indirect interactions; interaction happens among the individuals as participants and the group as a whole [9].

5.1 Audio Synthesis

The audio synthesis module in CAASAP consists of two different parts. First, the system responds to parameter changes received from interface module and maps them onto a number of musical textures. Second, the audio synthesis module will respond directly to the adaptive module parameters based on the analyzes of the control-data. The sound-action strategy in audio synthesis is focusing on main action as *tilt to change state*. Participant's tilt to change state, accelerometer sensor data, determines the similar control features of each instrument. Participant can choose, switch, add and control four different instruments. Volume, reverb and noise levels can be set through the interface module.

The first instrument, *Low3P* is based on the combination of three signal-controlled lowpass modules. In each module the resonance frequency is set by continuos sinusoidal waves, which bring changes in the bandwidth of the cutoff frequencies. Accelerometer data is mapped to control the frequency values of the sawtooth and sinusoidal waves that determine cutoff frequencies. The tilt movement also changes the width of the resonant peak. The gain of resonance frequency results in more dramatic texture when all three modules are activated within the equivalent low frequency scale.

OscMood is the second instrument and it is based on the basic frequency modulation where the carrier audio signals and modulation frequencies are controlled through accelerometer sensor data. The frequency range is set on a very limited scale and this gives more weight to the modulation frequencies to affect on resulted sounds. *BeatMove* is an instrument that creates dynamic rhythmic patterns through pulses generated by granular samples from a wavetable. The speed and the force values of the accelerometer data creates the beat-move. Sound manipulation performed with the fourth instrument, *WaveStretch*, is based on the playback use of the sampled sound source and overlap mixing of the wavetable reading points with sawtooth wave oscillator. The gain of the oscillator is controlled by the amount of samples in the sound file and changes in the noise texture.

The instruments, *Swarm1* and *Swarm2* will be controlled through the adaptive control module. These instruments will apply resulted parameter changes to the transformations of sampled sound materials, using them as a source for the synthesis of a new sound output. The transformation of the sound will use a polyphonic synthesis patch with four different parameters. Pitch, amplitude, duration and starting point in milliseconds will be all modified by variations. *Swarm1* and *Swarm2* will generate a class of sounds with multiple sonic gradations and variants.

5.2 VBAP - Direction and Communication

The spatial sound feature in CAASAP will make it possible to transform and perceive generated sounds from various directions based on speaker locations in the actual performance space. Receiving audio stream in different directions during a collective improvisation performance can cause alternative communication possibilities among the participants and it can change the flow of improvisation. CAASAP will use VBAP technology to control the direction of the audio stream in the performance [10].

5.3 Visual Representation of the Sonic Location

Spatial sound also gives possibilities to represent participant's location through the decisions made for the audio streaming directions in the performance. CAASAP will visualize sonic movement and the interactive gestures of the performers. Projecting these visual representations will support participants to recognize their sounds and their act of control within the overall improvisation.

6 AUDIENCE-ORIENTED DESIGN

CAASAP aims to develop a set of evaluation methods in order to study and examine the interactive system's features and its engagement with the audience reflecting its participatory attributes. System design strategies will be developed further based on the audience experiences. Evaluation methods will involve observations, interviews and questionnaires, which will be designed in order to achieve a certain level of comparison and assessment of the effectiveness of sound-action-gesture strategies and to reflect on the CAASAP research arguments.

CAASAP claims that besides exchanged musical events in a collaborative music performance, musical activity of the participants can be observed through participants' exchanged gestures in the moment of playing. Sound and gestures can be both regarded as an important mode of interaction in a collaborative music making process. This is a generic hypothesis of musical interaction that CAASAP research accentuates. The validity of this hypothesis is supported by a recent study [15] that categorizes the movements of musicians as sound-producing, ancillary, sound-accompanying and communicative that results in the music-related body movements.

CAASAP also argues that the dynamic social interaction in real-time performances challenges the traditional roles in music by changing the role of the audience to that of a performer of improvised music. The main building block of the modular structure of interaction in CAASAP research (Figure 1) is necessary and sufficient for wide-ranging scenarios in defining social roles and social behavior in collective music performances. Understanding the social behavior in a collective music making process will develop the adaptive ability of the CAASAP interactive systems, which will support the creativity and productivity achieved during the participation. This will result in a shared collective musical experience. Support for this hypothesis comes from the ongoing fundamental research issues in theoretical approaches to social behavior in music and in analysis of social roles in performers [11].

CAASAP research strongly claims that the level of interaction in music performances will be increased by providing audience control, feedback and communication blocks in social interaction in music. This hypothesis is supported by specifically, in the context of experience and interaction design [3]. While mediated music making and listening is usually concerned as a passive, non-interactive and non-social experience, CAASAP will facilitate audience participation by providing finely balanced mechanisms for dynamic social interaction.

7 CONCLUSION

Collective improvisation enriches the musical experience and audience participation enhances the pleasure achieved in a musical activity. Interactive performance systems can facilitate audience participation by providing finely balanced mechanisms for dynamic social interaction that expands in event participation. These include accessible and *easy use - easy control* tools that give audience control over creative acts and allow them to explore musical experience. Controlling group communication is another mechanism that can provide a flow in the event. This paper focuses on the modular form of these mechanisms in the CAASAP system and explicates the strategies in developing the interface and adaptive modules. In this process, enabled technologies, such as mrmr and RjDj, extended the research ideas in the study of interactive gestures with mobile devices.

Computer-supported collaborative work has been a traditional focus in human-computer interaction (HCI), and collaborative music making is perhaps one of the most interesting application domain. However, applying HCI methodologies directly to the interactive art [12] and computer music [13] could be problematic, as HCI relies on a task-based paradigms and graphical stimulus & response model (e.g., WIMP), whereas interactive art and music systems are based on continuous interaction and multisensory feedback. Especially, the special nature of musical interaction by the use of gestures requires special care for grounding our design decisions [14, 15]. In the next phases of the system it is apt to consider HCI methodologies such as hierarchical task analysis and interface design, as informed by interaction design [16]. In a complementary task, there will be more advanced feature analysis of audience voluntary actions in order to broaden the strategies for mapping gestures to musical output in the CAASAP system.

During a collective improvisation, participants discover expressiveness in themselves, which is supported by the easy to use interfaces. In this process of investigation and exploration, the audience can learn more and without difficulty. Therefore, alternative levels of learning curves will be implemented in CAASAP in order to reach a wider musical expressivity, varying from instant gratification to virtuosity.

The modular structure will help to extend the amount of participants in the performance and develop interface modules that will not be dependent on a certain type of mobile device as well. CAASAP is in early development stages aiming to involve audience experiences in the process of its development by organizing more collective performances.

8 ACKNOWLEDGEMENTS

I would like to acknowledge Cumhur Erkut for the discussions we had on the paper structure and Hannah Drayson for her collaboration with *Circles*. This work is supported by the Academy of Finland (pr. 120583).

9 REFERENCES

[1] Tahiroğlu, K. *Interactive Performance Systems: Experimenting with Human Musical Interaction. Doctoral dissertation*. University of Art and Design Helsinki Publications, ISBN: 978-951-558-276-8. 2008.

[2] Lippe, C. ”Real-Time Interaction Among Composers, Performers, and Computer Systems”, *Information, Processing Society of Japan SIG Notes 2002*, (123), 1-6. 2002.

[3] Shedroff, N. *Experience Design 1*. Ind.: New Riders, Indianapolis, 2001.

[4] Weinberg, G. ”Interconnected Musical Networks: Toward a Theoritical Framework”, *Computer Music Journal 29* (2), 23-39. 2005.

[5] Wang, G. - Essl, G. - Penttinen, H. "'Do Mobile Phones Dream of Electric Orchestras?", *In Proc. Int. Computer Music Conference*, Northern Ireland, 2008.

[6] Tanaka, A. "Malleable Mobile Music", *Conference on Ubiquitous Computing*, Tokyo, Japan, 2005.

[7] Tahiroğlu, K. - Drayson, H. - Erkut, C. "An Interactive Bio-Music Improvisation System", *In Proc. Int. Computer Music Conference*, Northern Ireland, 2008.

[8] Blackwell, T. *Swarming and Music*. In Eduardo Reck Miranda and Al Biles (Eds.), Evolutionary Computer Music (194-217). Springer, London, 2007.

[9] Borgo, D. *Sync or Swarm: Improvising Music in a Complex Age*. Continuum International Publishing Group, New York, 2005.

[10] Pulkki, V. - Karjalainen M. "Multichannel Audio rendering Using Amplitude Panning", *Signal Processing Magazine 25*, (3), 118-122, 2008.

[11] SBM 2009. International Workshop on Social Behavior in Music (Online). Avaliable from `http://www.infomus.org/SBM2009/` (accessed on 2009-06-04).

[12] Höök, K. - Sengers, P. - Andersson, G. "Sense and sensibility: evaluation and interactive art", *In Proc. Conf. Human Factors in Computing Systems (CHI)*, Fort Lauderdale, FL, USA, 2003.

[13] Kiefer, C - Collins, N - Fitzpatrick, G. "HCI methodology for evaluating musical controllers: A case study", *In Proc. New Inst. Musical Expression (NIME)*, Genova, Italy, 2008.

[14] Miranda, E.R. - Wanderley, M.M. *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*. AR Editions, Middleton, Wisconsin, 2006.

[15] Jensenius, A.R. *ACTION - SOUND: Developing Methods and Tools to Study Music-Related Body Movement.PhD thesis*. Dept. Musicology, Unv. Oslo, Oslo, Norway, 2007.

[16] Sharp, H. - Rogers, Y. - Preece, J. *Interaction Design: Beyond Human Computer Interaction.*. Wiley Intl., London, UK, 2007.

MUSICAL APPLICATIONS AND DESIGN TECHNIQUES FOR THE GAMETRAK TETHERED SPATIAL POSITION CONTROLLER

Adrian Freed
Devin McCutchen
Andy Schmeder

CNMAT, dept. of Music
UC Berkeley
{adrian, andy}@cnmat.berkeley.edu
devin_mccutchen@berkeley.edu

Anne-Marie Skriver Hansen, Dan Overholt

Department for Media Technology
Aalborg University
{amhansen,dano}@imi.aau.dk

Winslow Burleson
Camilla Nørgaard Jensen

School of Computing and Informatics
Arts, Media, and Engineering
Arizona State University

winslow.burleson@asu.edu
camilla.jensen@asu.edu

Alex Mesker

Department of Media, Music and Cultural Studies
Macquarie University
alex.mesker@humn.mq.edu.au

ABSTRACT

Novel Musical Applications and Design Techniques for the Gametrak tethered spatial positioning controller are described. Individual musical instrument controllers and large-scale musical and multimedia applications are discussed.

1. INTRODUCTION

Although hundreds of new controllers have been explored for musical applications, very few have emerged as sufficiently flexible and general to serve as platform technologies for a wide variety of musical instruments and interactions. One successful controller that is already widely used in musical applications is the digitizing tablet [15, 16]. In this paper we show by exploring representative examples how the Gametrak controller is emerging as another viable platform technology.

The Gametrak spatial position controller is an increasingly popular platform for experimental musical controllers, math and science manipulatives, large scale interactive installations and as a playful tangible gaming interface that promotes inter-generational creative play and discovery. Although largely displaced in its original market as a gaming controller by the Nintendo Wii, the Gametrak is attractive for music controller experiments and performance-quality instruments because of its unusually simple, cheap implementation and the ease with which it can be customized.

After introducing the peculiarities of the Gametrak and comparing it to related spatial position sensing systems we survey musical applications of the device and some of the basic design techniques discovered by the authors. The short paper format cannot do justice to the depth and breadth of such applications, so projects have been selected based on whether they represent unusual or surprising uses of the controller or because they represent fruitful starting points for future explorations. More detail on each project can be obtained from the web links included in the bibliography.

2. The Gametrak: a versatile tethered position sensing system

The Gametrak system was invented in 2000 by Elliot Myers [8]. He arrived at the basic concept while playing with a retractable washing line in a hotel. By placing potentiometers on a worm gear driven by the hubs of two retractable nylon tethers, the distance of the extension of the chords can be estimated. Passing the chords through the knobs of a pair of gaming joysticks supplies two orthogonal angle estimates for each.

This approach is cheap to implement but requires a careful mechanical design to achieve the desired precision, avoid tangling and to minimize the impact to the user of the pull of the tether. The first problem is solved by a series of smooth guiding tubes and by a clean path for the nylon chords. The basic ergonomic design was influenced by gaming applications involving the swinging of clubs (golf), bats (baseball) bowling or skiing. The two joysticks and tethered sensors are housed in a weighted box that is normally placed on the ground. Nylon clips are provided that can be attached to accompanying gloves. The unit also includes a plug-in footswitch.

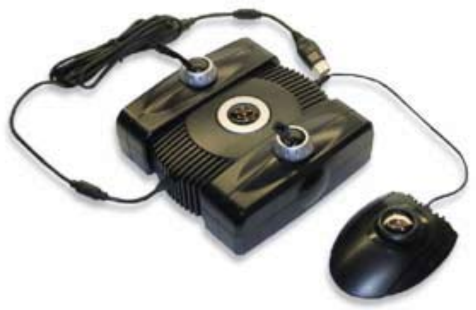


Figure 1 Gametrak and Footswitch

The control electronics implements serial protocols for USB using HID protocols for computers, a specialized USB protocol for PS2 and there is also support for XBOX. The choice of format is made by shorting solder pads under the board making it straightforward to change a Gametrak to a different platform if required.

Perhaps the most important single factor to its recent popularity as an experimental music controller is the low cost of the device that resulted when thousands of Gametraks entered the surplus wholesale channel and became available for between US$8 and US$20 at internet retailers.

The official retail price for Gametrak games bundles is listed as US$70 but MadCatz the current owner of the technology has discontinued the device. Unlike other interesting discontinued controllers such as the P5 glove and the fingerworks iGesture, the Gametrak will be readily available for many years as over 300,000 have been sold. Note that it is easy to build comparable 3-axis position sensing from readily available string pots (from Celesco or Penny and Giles, for example) and joysticks.

3. Comparison with other Spatial Positioning Technologies

The Gametrak occupies a unique niche in the rich ecosystem of devices that can be used for 3D spatial position sensing [3]. It is by far the cheapest of any absolute position-sensing device. The Gametrak has in common with 3D time-of-flight cameras and other remote optical sensing techniques of low mass at the point(s) being sensed. Most other position sensing devices require a wand or small box with associated weight and power requirements. Most IR sensing devices such as the Wii controller or Buchla Lightning only work reliably indoors. GPS on the other hand works poorly indoors and has too low resolution for gesture sensing. The Gametrak works outside but is not weather proofed for permanent installations. The Gametrak, like the Polhemus system, is insensitive to most interference in the physical environment.

The major peculiarity of the Gametrak is of course the constant pull of the tethers. So although of low mass, each tether both constrains the position of objects to be sensed (because of tangling) and requires a source of counterbalancing force to establish a controlled position. In the following applications we will see that much of the interesting work with the Gametrak comes from strategies for embracing, tackling or defeating the tether. Note that the original Gametrak patent describes a haptic feedback component to the device where the tether's response was controlled dynamically[8].

People have little difficulty compensating for the constant pull of the tether because they already master comparable interactions, i.e., lifting constant mass limbs against the force of gravity; car accelerometer pedal; high-hat pedals; bent branches and stems; retractable dog leashes, key fobs, laptop cables, and vacuum cleaner power chords; fishing lines; sailing boat "sheets" and the bell ringers "sally," etc.

4. Gametrak Design Techniques and Applications

4.1. Direct mapping: Tethered Theremin

A straightforward musical application useful for exploring calibration, mapping and scaling of Gametrak gestures is a Tethered Theremin illustrated in Figure 2. the Gametrak gloves are used as originally intended on the hands, with one hand controlling pitch on the x-axis (and an optional wave-shaping filter on the y-axis) simulating distance from a theremin's upright antenna, and the other hand controlling volume based on z-axis extension, simulating distance from the theremin's loop antenna [10, 13].

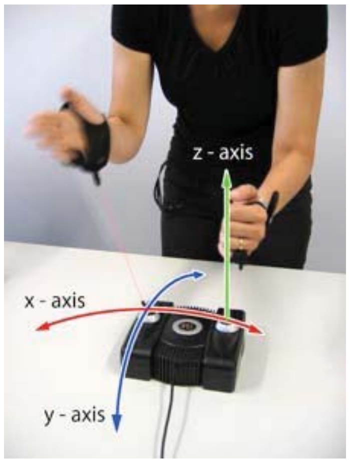
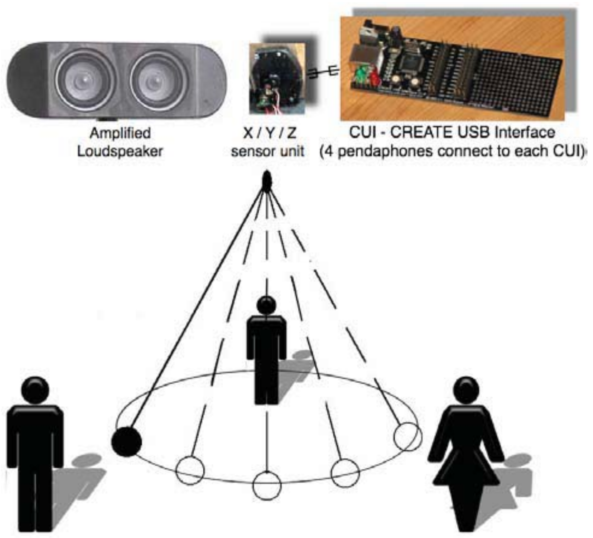


Figure 2: Tethered Theremin control axes

4.2. Counterweighting: Pendaphonics

In Pendaphonics installations Gametraks are ceiling mounted with balls or other object ("Pendaphones") attached to the tethers designed as counterweights to the spring-return force of the tether reels and to create physical dynamics that are engaging for users [5]. This effectively mimics what divers call a "neutral buoyancy" and from the users point of view (as for divers and astronauts) changes the interaction from counteracting a constant force to exercising inertial control.


Amplified Loudspeaker
X / Y / Z sensor unit
CUI - CREATE USB Interface (4 pendaphones connect to each CUI)
Figure 3. Pendaphone Components with Multiple Users

Each Pendaphone can be raised and lowered between 0–3 meters in height, and the trajectory of their swings directly controls the sounds emanating from a loudspeaker mounted above them. Multiple channels of loudspeakers are used to spatially distribute the sounds that are generated, enhancing the sense of physical immersion in the space. The physical setup is designed to be flexible and can be adapted to many different exhibition spaces and applications.

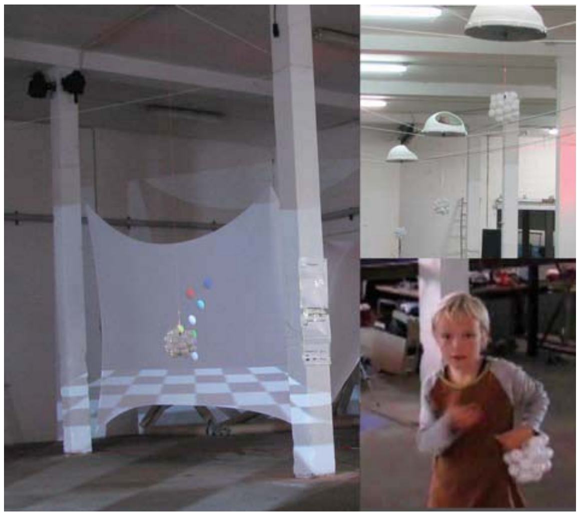


Figure 4x, y, and z. Left x), a Pendaphone bob hanging with a projected 3D environment; top right (y), three suspended Pendaphones; bottom right (z), a child plucking the string while holding the Pendaphone bob steady.

To illustrate the range of applications possible with pendaphonics we describe 3 mappings used at the Platform4 event. These were all directed towards the intuitive investigation of the interface, where exhibition visitors activate a soundscape in physical space. Familiar metaphors have been used, such as the idea of the turntable, where a rhythmical soundtrack is played back.

4.2.1. Clockwise Rotation

Clockwise rotation plays the sound forward, and counter clockwise rotation plays the soundtrack backward; the polar velocity of the swing changes the playback speed.

4.2.2. Percussion Sounds

Another sound feedback system consisted of percussion sounds that were mapped to cue points along the 360 degrees of the pendulum swing. Every thirty degrees a percussion sound was activated. The percussion sound changed pitch, depending on how high or low the pendulum was positioned in the air, and the audio frequencies percussion sound was filtered according to the amount of acceleration.

4.2.3. Musical Piece

The third sound feedback system was a musical piece composed by Mads Weitling (see http://www.kiloton.dk/). It consisted of a pre-composed soundscape, where the pendulum movement generated tones that mixed in with the soundscape and varied in texture and velocity.

Other metaphors for Pendaphonics being explored are:

- • Sound ball improvisation tool [1]
- • Sound transfers and sound traveling (locally and networked), incorporating hanging loudspeakers embedded in the bob, providing natural Doppler and "Leslie" effects
- • Diverse ways of throwing and catching sounds through physical actions with the pendaphones
- • Plucking the pendulum strings to set up future events or trigger special effects (either sounds or visuals)
- • Detection of user's direct interaction with bobs while the string is motionless, e.g. w/ embedded accelerometers
- • Detection of spatial interaction between two or more bobs
- • Diverse game/play scenarios
- • Individual instruments versus one collective instrument
- • Physical vs. virtual presence, movement, and representation

4.3. Pendaphonics + Sound Directivity Control

The Pendaphonics application of Figure 5 combines a Gametrak, an inertial sensor (Wii) and a 120-channel programmable directivity loudspeaker array [4]. In one application the tether is used to steer narrow sound beams. In another striking gestures are captured for a virtual swinging gong. These gesture parameters drive sound motion and directivity models that are synthesized in real-time and rendered on the speaker array.

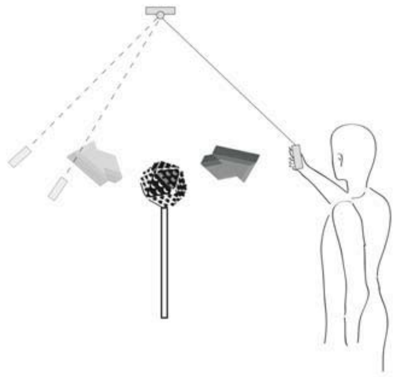


Figure 5: Spherical speaker array radiation pattern control

4.4. Combining and Duplexing Vertically

Another way of providing "neutral buoyancy" is shown in Figure 6 with two Gametraks (one floor mounted, the other ceiling mounted) tethered at the corners of a lightweight cube corresponding to the vertices of a tetrahedron. This arrangement allows for orientation and position to be computed. It also offers an interesting set of mechanical resonances to interact with as the cube twists independently around its center and this center rotates around the axis between the Gametraks.

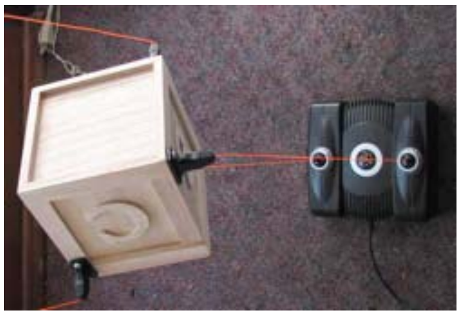


Figure 6: Tethered cube viewed from upper Gametrak

4.5. "Gearing" with the Gametrak control space

The conical bound on the Gametrak measuring region means that large spaces require a longer tether and are less precisely measured. These constraints suggest using the shortest tether that provides a large enough sensing region. It also affords mappings that let the user decide in real-time what range of motion they want to use with the radial axis being the gear factor. For example for acquiring conducting gestures the Gametrak is installed at waist height (on a robust music stand for example) and the axial parameters are mapped to the rhythmic gesture analysis. Conductors with a flamboyant use of space use long tethers, more reserved precise conductors are closer to the Gametrak. Note that the distance axis information need not be discarded for gesture analysis. It can be used to establish, for example, a direction for the conductors' leaning gestures.

4.6. Pinned Tether – The tea chest bass

In addition to the free-plucked tether of the Pendaphone stopped-string instruments can be simulated by extending a tether, pinning it down and providing an angled fingerboard. One application of this idea simulates a tea-chest or wash-tub bass.

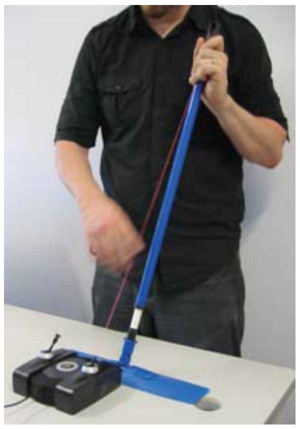


Figure 7 Gametrak Chordophone

An analysis of attack events (based on impulses on the x- and y-axes) determines whether the musician has plucked a string and to what extent the string deviated from its average position.

4.7. Specialty Tether: Tethertonium

With a conductive fingerboard and conductive thread attached to a tether the footpedal input can be used to provide accurate timing of touches on a simulated trautonium [14] as shown in Figure 7.

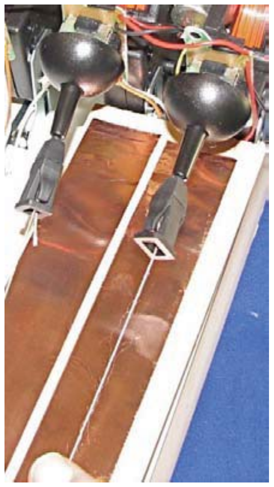


Figure 7: Tethertonium

4.8. Dividing + Dismantling: Kotrak and Ondestrak

In some applications the imposed distance between the two joysticks is too constraining. This can be easily addressed by adding a second Gametrak or by dismantling one and reassembling its two sensors at the desired distance. The Kotrak can be used to capture the string pressing and pulling gestures of Koto players.

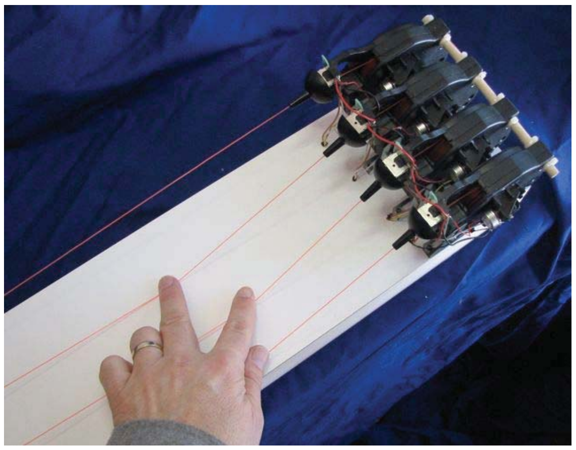


Figure 8: Kotrak

The Ondestrak [7] takes the dismantling even further as one of the two sensors is used with its spring removed. This needs to be done carefully because of the energy stored in the spring, its sharp edges and the messy lubricant.

The Ondestrak was built to explore a variant of the Ondes Martinot variable pitch interface. Instruments preceding the Ondes Martinot such as the Hellertion [6]and Trautonium [14] provided a combined amplitude and continuous pitch control for an untethered hand. The Ondestrak is a hybrid form of these instruments. A finger drags a ring attached to a loop of chord driving the Gametrak sensor without spring. The other sensor measures displacement of the board the whole system is built on as it is pressed against springs at each end. Note that the joysticks provide for a third axis of control as the ring is moved away and towards the player. This is naturally mapped to timbral parameters.

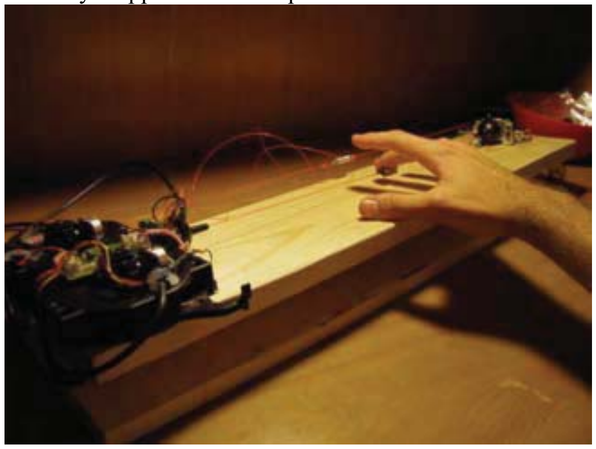


Figure 9: the Ondestrak

4.9. Controller Replacement

The HID interface using the PC USB configuration is "plug and play" using the HID objects in Max/MSP and PD and other music synthesis programs. It is common practice to rewrap the HID data as OSC messages. It is also straightforward to replace the Gametrak microcontroller with a wireless transmitting system or another interface as illustrated in Figure 10 which shows a $25 Microchip USB microcontroller board with uOSC [11] to provide a 1000Hz update rate for the data encoded directly in OSC.

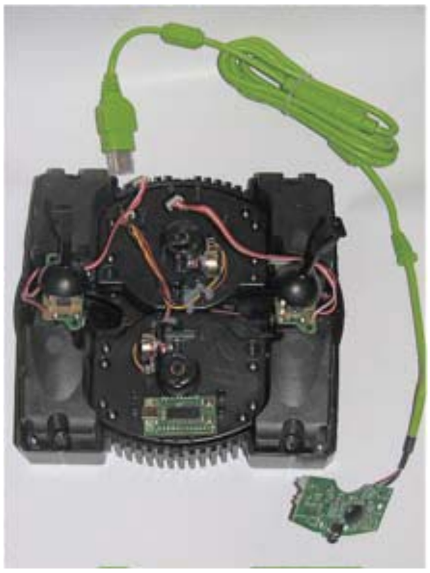


Figure 10: replacing the microcontroller

As well as providing a higher data rate and higher resolution samples the uOSC board provides control of latency and jitter to reasonable bounds for exacting performance applications [12]. Pendaphonics installations typically use CUI boards [9] to acquire the data from 2 pairs of Gametrak sensors.

5. Conclusion

Each of the applications presented invites exploration of numerous interesting mapping strategies for sound, image and motion synthesis. Ultimately development of the mappings takes longer than the physical prototyping. The Gametrak is a convenient platform to learn about mapping strategies and can facilitate rapid "sketching" of user interfaces [2] that may ultimately use untethered or inertial sensing.

An important conclusion from our explorations is that although the tethers are sometimes a nuisance they often create opportunities to more fully engage users in physical interactions beyond those originally reflected in commercial gaming applications.

6. References

[1] Belt, L. and Stockley, R. *Improvisation through theatre sports : a curriculum to improve acting skills*. Thespis Productions, Seattle, Wash., 1991.
[2] Buxton, B. *Sketching User Experiences*. Morgan Kaufmann, 2007.
[3] Freed, A. Position Sensing Technology and Product Summary. 2009. http://cnmat.berkeley.edu/Position-Sensing.
[4] Freed, A., Schmeder, A. and Zotter, F. Applications of Environmental Sensing for Spherical Loudspeaker Arrays *IASTED Signal and Image Processing*, Hawaii, 2008.
[5] Hansen, A.-M.S., Overholt, D., Burleson, W. and Jensen, C.N. Pendaphonics: a tangible pendulum-based sonic interaction experience *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, ACM, Cambridge, United Kingdom, 2009.
[6] Lertes, P. and Helberger, B. Electrical Musical Instrument 1932. USPTO # 1847119.
[7] McCutchen, D. The Ondestrack. 2008. http://www.instructables.com/id/The_Ondestrak/.
[8] Myers, E. Position transducer 2004. USPTO # 6697760.
[9] Overholt, D. Musical Interaction Design with the CREATE USB Interface: Teaching HCI with CUIs instead of GUIs *ICMC*, New Orleans, LA, USA, 2006.
[10] Sauer, M. Die Thereminvox : Konstruktion, Geschichte, WerkeThesis Thesis (doctoral)Electronic Publishing, Universität, Mainz, 2006. 2008.
[11] Schmeder, A. and Freed, A., A low-level embedded service architecture for rapid DIY design of real-time musical interfaces. in *NIME*, (2009).
[12] Schmeder, A. and Freed, A. uOSC: The Open Sound Control Reference Platform for Embedded Devices *NIME*, Genova, Italy, 2008.
[13] Theremin, l.s. Method and Apparatus for the Generation of Sounds 1928. USPTO # 1661058.
[14] Trautwein, F. Electrical Musical Instrument 1938. USPTO # 2141231.
[15] Zbyszynski, M. An Elementary Method for Tablet *New Interfaces for Musical Expression*, Genova, Italy, 2008.
[16] Zbyszynski, M., Wright, M., Momeni, A. and Cullen, D., Ten Years of Tablet Musical Interfaces at CNMAT. in *New Interfaces for Musical Expression*, (New York, 2007), 100-105.

A FRAMEWORK FOR ECOSYSTEM-BASED GENERATIVE MUSIC

Oliver Bown
Centre for Electronic Media Art, Monash University, Clayton, Australia, 3800
oliver.bown@infotech.monash.edu.au

ABSTRACT

Ecosystem-based generative music is computer-generated music that uses principles borrowed from evolution and ecosystem dynamics. These are different from traditional interactive genetic algorithms in a number of ways. The possibilities of such an approach can be explored using multi-agent systems. I discuss the background, motivations and expectations of ecosystem-based generative music and describe developments in building a software framework aimed at facilitating the design of ecosystemic sonic artworks, with examples of how such a system can be used creatively.

1 INTRODUCTION

A traditional paradigm of generative and evolutionary music is that of the interactive genetic algorithm (IGA), which is based on the notion that an artist can evolve aesthetically pleasing music by using their aesthetic judgement as the 'selective pressure' in an artificial environment. More recently, researchers in generative and evolutionary music have started to broaden their interest to more collective behaviours, such as social learning, cultural dynamics, and niche construction [9], in simulated multi-agent systems [6, 11, 8]. This approach, which I will refer to generally as *ecosystemic* [1], abandons the goal of exerting direct control over evolving systems through aesthetic selection, requiring a more complex creative interplay between software and artist. For example, McCormack [7] has described modelling processes such as niche construction as a kind of creative *design pattern* with which an artist can design complex generative dynamics in an exploratory manner. One of the major challenges in working with multi-agent systems is that complexity of design rapidly gets in the way of deeper exploratory investigation. Design patterns address this problem by helping us to break down and conceptualise systems, providing general rules that could be applicable to a range of different situations. As such, they reduce the opacity of complex systems.

The framework discussed here (also see [3]) complements this software engineering-inspired approach by offering additional ways to reduce the opacity of complex creative projects, primarily through tools that facilitate the batch processing and analysis of systems in a range of scenarios. Such challenges are already being addressed by frameworks developed for studying multi-agent systems, particularly in the social sciences and in artificial life. But none of these systems have been built to deal with the practical goals and typical methodologies of artists and musicians in mind. Multi-agent modelling experiments are of little creative value if they cannot be successfully transferred to creative domains, or be used creatively. However, they have the potential to form the basis for a new approach to works such as compositions, installations and creative software, due to the variety of behaviour that multi-agent systems exhibit that is not seen in traditional software.

This paper begins by discussing the motivations for pursuing an ecosystemic approach to artificial evolutionary music and art, as compared to existing approaches such as the IGA. I then describe a framework for developing ecosystemic artworks and music, which integrates an experimental multi-agent modelling environment with a real-time music environment. Finally, I demonstrate how this framework can be used to develop ecosystemic sonic artworks: installation works in which audio is used as the basis for an evolutionary environment.

2 BACKGROUND, MOTIVATION AND EXPECTATIONS

Despite moderate successes, the IGA approach to evolutionary art falls somewhat short of its original hopes. In principle, by analogy with natural selection, it promised to produce complex outputs that are both pleasing to, but beyond the understanding of the user, that the user wouldn't have thought of himself, and perhaps couldn't even have imagined. In practice, this goal has proven elusive.

These shortcomings can be discussed in terms of the structure of the *space* that an IGA user navigates in his search. Parameterising a generative system in order to make parts of it evolutionary inherently defines a space – the pa-

[1] Arguably, ecosystem models are a more specific subset of this area. The term is used here to refer more generally to any set of coevolving interdependent elements.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

rameter, or genotype, space of the system. This space may be thought of as a network of points (genotypes) connected by genetic mutations. The IGA user engages in a blind and relatively passive process of evaluating and selecting, which can be represented as a journey through this genotype space. Initially, an important question about the space itself is whether it is structured such that the user can steer the evolving system towards a certain target. If so, this is a potentially powerful design tool. For example, even though the target was pre-specified, it may have other interesting incidental properties.

A more ambitious expectation for IGA software is that it should also guide the user in interesting and surprising directions. This prompts a second question about the design of the genotype space: if interesting and surprising generative artefacts appear on this journey, will they be positioned *en route* to even more interesting and more surprising artefacts? These questions highlight the importance of the parameterisation of the system itself, suggesting that the art of designing such systems lies in designing genotype spaces that lead us to interesting places. Little development has been made in this area, although examples, such as the Neuroevolution of Adaptive Topologies (NEAT) project [12], have pointed to ways in which evolutionary lineages could lead in diverse aesthetic directions. NEAT specifies a procedure for neural nets to increase in topological complexity during their evolution. Here the evolved entities could be argued to have accumulated a user's preferences over time.

The aesthetic selection of computer generated art and music by individual users is also bound for practical reasons to the simple case where genotypes map directly to phenotypes that can be treated as isolated units for evaluation. Evolutionary entities with any kind of developmental process, environmental influence, or interaction between the individuals within the population, are complex enough to necessitate a different approach. One solution to this *fitness bottleneck* [1] is to employ multiple users as aesthetic selectors. Recent research explores this avenue by borrowing from areas such as social networking technology, human computing [2], and grid computing efforts such as *SETI@Home* [3], in order to find ways to distribute the process of selection (*e.g.* Draves' *Electric Sheep* [4]).

The ecosystem approach attempts to rethink evolutionary art by focusing on the design of spaces which embody coevolutionary processes such as niche construction [5]. In IGAs, the passivity of users in their interaction with the system is at odds with the supposed aesthetic influence they have over it. Ecosystem models are primarily non-interactive, but still face many of the same challenges as the IGA approach: designing evolutionary spaces that lead 'naturally' to a diverse array of interesting states.

Nevertheless, sonic ecosystems also have great potential for interactivity in more or less direct ways. In the interactive audio visual installation *Eden*, McCormack [6] used audience presence as a resource for a population of evolving agents, such that agents might evolve to draw the attention of the audience.

In other forms, interaction with sonic ecosystems could correspond more closely to the original goals of the IGA approach. This follows the reasoning that the real world of artistic and musical creativity itself resembles an ecosystem, exhibiting heterogeneity, stability, interdependence and the capacity for the components of the system to influence the entire system's future evolution in unpredictable ways. The value of an artwork or piece of music is strongly tied to its situatedness in a cultural context, which determines its relevance [10]. If culturally determined relevance is a significant factor in determining the evaluation of artefacts, then creative cultural domains can be said to involve a strong degree of feedback. Prior experience affects our evaluation of new music, and new music is discovered and consumed via channels of authority that precede pure content-based evaluation. Evaluation of creative artefacts is constantly shifting and heavily influenced by these factors. Accordingly, in a future inhabited by fully fledged creative computational systems, we would expect them not only to be adapted to the cultural factors affecting value judgement, but also to actively manipulate these factors through cultural interaction. In short, it may be a mistake to assume that systems designed to adapt to our preferences will actually produce the most pleasing results.

This discussion highlights the extensibility of ecosystem models towards more interactive and evaluative forms. However, our present focus is on the more manageable problem of creative ecosystems which are either closed or loosely interactive. In this area, an obstacle to good design is that the behaviour of a multi-agent system is typically complex and requires detailed analysis of its macroscopic properties to be clearly understood. Put differently, an ecosystemic approach is particularly interested in the creative exploration and use of exactly those models that are complex and not immediately obvious, and the methodology presented here is aimed at maximising the potential to explore interesting complex systems. In the sonic domain, this might involve generative sonic works that continue to develop and transform indefinitely, but with consistent structural and aesthetic properties.

The issues of complexity suggests the need for an iterative development process based on the analysis of behaviours in successive versions of a model. Similarly, dis-

[2] The use of human brains for data processing, such as in *reCAPTCHA* [13]

[3] SETI stands for the Search for Extra Terrestrial Intelligence. This group maintains an *ad hoc* processing grid of subscribers' home computers for the brute-force analysis of cosmic radio data [5].

[4] http://electricsheep.org

[5] Coevolution typically refers to the mutual influence between the evolution of two species, but can apply to a multitude of interacting evolving entities, including the set of individuals within a species.

cussing approaches to the relationship between artificial life models and real-world complex systems, di Paolo *et al* [4] propose a three-stage development process of exploration, experimentation and explanation. The exploratory phase sets out to see what is of interest and relevance in a model and to record information about the model. In the experimental phase, hypotheses are tested about the behaviour of the model. In the explanatory phase, the understanding gained from the experimental phase is applied to the natural world. The different goals of creative model building mean that there is no explanatory phase, and a more open-ended flexible approach to exploration and experimentation (which includes de-bugging and learning the relationship between a model's logical behaviour and its aesthetic outcomes). A more detailed study of working practices in creative computing would reveal how this kind of methodology could be developed more explicitly in creative domains.

3 FRAMEWORK DESIGN

In [3], I discuss a number of design requirements for a framework for creative ecosystemic models:

- Analysing system behaviours (e.g., comparing many different parameters)
- Integrating multi-agent and sound components in one development environment
- Facilitating the design of live algorithms [2] using a multi-agent approach
- Creating flexible agents and configurations for multiple contexts
- Probing and editing models interactively
- Facilitating new forms of software extensibility, such as being able to embed models inside other models.

A preliminary design for a framework is introduced in [3], with an example of a sonic ecosystem. Here I discuss how the framework aims to integrate multi-agent modelling principles and a computer music library. The framework also offers libraries for more specific ecosystemic and evolutionary functionality like genetic variation, resource management, and handling dynamically changing populations.

One of the main practical considerations for exploring new types of creative generative and evolutionary methodologies is for models to be easily adapted for multiple target applications, such as batch processing for analysis, interactive exploration and real-time performance. For this reason, and for the goal of achieving useful extensibility, popular object oriented programming languages were seen as providing the most power and flexibility. Java was chosen for the current framework, primarily because of its greater user-friendliness. Extendable general-purpose development environments, such as Eclipse [6], also provide core project management functionality that is desirable for the kinds of complex projects that are likely to be needed.

An audio library, Beads [7], was developed by the author in pure Java with this framework and a number of other computer music applications, including live performance, in mind. This audio library follows the principle of good integration; that is, it is advantageous to work in a single development environment when working experimentally. Popular real-time music environments such as MaxMSP [8], SuperCollider [9] and JSyn [10], use separate formats for high-level and low-level (digital signal processing or DSP) elements, requiring users to skip between different environments. For example, users can write *externals* for MaxMSP, but must compile them in a development environment and then launch them in MaxMSP (at least in the case of native C externals). A complete integrated environment allows developers to add DSP routines straight into their code. With a multi-agent modelling environment and sound environment closely integrated, it is also easy to have modelling processes triggered by audio-rate processes, as well as vice-versa.

The current instantiation of the ecosystems framework focuses on the basic requirements listed above. A hierarchical scheduling mechanism provides the core configurability and addresses the requirement of extensibility by allowing schedulers to be easily chained together. Using an XML configuration file, an arrangement of schedulers and listeners can be set up, with a master scheduler at the root. In the most basic multi-agent scenario, this would consist of one scheduler and a population of agents that are updated by the scheduler. Additional listeners, such as a visualiser or sonifier or tools for collecting simulation data from the population, can be specified in the configuration file or added interactively. Thus different configuration files can specify different use cases such as batch processing and live installation. A scheduler can also take its timing from an audio-rate clock instead of the master simulation scheduler, meaning that audio rate processes and higher level scheduling processes run in the same ratio under different circumstances (this is useful in the case of running real-time systems offline). As well as allowing different configurations, the hierarchical structuring of schedulers also facilitates interesting experiments in the extensibility of existing systems, such as the integration of two populations of agents in a common environment, or the embedding of agents and their environments as subsets of bigger environments.

[6] http://www.eclipse.org
[7] http://www.beadsproject.net
[8] http://www.cycling74.com
[9] http://www.audiosynth.com
[10] http://www.softsynth.com/jsyn

For logging data, a data logger class exists which uses the Java Reflect API to probe agent classes. Thus if agents have the field `size`, the logger can be told to log the value "agent:size". This will log all of the size values for all of the agents at each time step. This syntax works for nested classes and data structures. If agent's have a field `memory` which in turn stores a variable length array of known pitches, then telling the logger to log "agent:memory:pitches" will log all of the pitches for each agent at each time step, and so on. Loggers can be switched on and off at different time steps. The logged data can then be formatted to work with a plotting program such as Gnuplot[11] or Mathematica[12]. No extra code has to be inserted into the agent except 'getter' methods for these fields, which the data logger automatically discovers. For interactive probing of data, another class exists which unpacks the entire configuration of schedulers and agents into a tree view. The elements of the tree view can then be edited manually (assuming the specified fields have 'setter' methods). Alternatively, for any elements that are being visualsed, it is easy to configure the system such that double clicking on that element brings up a similar tree rooted at that element.

Such features are in an early stage of development and the ultimate goal of this project is to develop the system and its associated working methodology to the stage where the framework can be used as a general purpose creative toolkit for sonic ecosystems.

4 USING THE FRAMEWORK

An ecosystemic approach to evolutionary computer music attempts to ground the design of an evolving artificial multi-agent system in an environment that is intrinsically related to the world of sound and music that we are familiar with. As discussed in Section 2, this provides, at least in principle, a kind of *coupling* between the agents in the evolving system and our experience, capable of guiding an artist's development of their work.

An ecosystem model typically consists of an environment containing resources (or also environmental conditions), a population of evolvable agents, and a set of rules defining the relationship between the resources and the agents' dynamics of survival and reproduction. This relationship is not the same as a specific fitness function for individual isolated agents, typical of standard genetic algorithms, since the population is involved in myriad interactions amongst each other (directly, or via manipulation of the environment), from which the exact conditions for survival emerge. This invites a variety of evolved relationships of both co-operative and competitive natures. For example, since parents and their offspring are closely related, their behaviour, locality, and their dependence on resources is similar and can often be in conflict. Such emergent relationships can be understood in terms of the theory of evolution, or more specifically in terms of known artificial life models. Often, models are unpredictable in the sense that the best way to know what they will do is to run them. But it is possible to work out the underlying behaviour and adjust the model accordingly (helped by the framework to the extent that it satisfies the design requirements).

In order to embed the system in a sonic context, the environmental resources should reflect acoustic features of sound that is created or affected by the agents in the population, but may also be mixed with sound coming into the model from outside (there is no reason why this couldn't be done at a more abstract level, such as in a MIDI domain, but sound is preferred as it makes fewer musical assumptions). An example is to take a spectrogram of the sound and to treat the level of each band as the literal quantity of a specific resource. Another is to take the amount of 'space' left in each band (e.g., a max value minus the level of the band) as the quantity of the resource, and rewarding health gains to agents in proportion to the relative level of sound they contributed to that band. In the latter case, the environment becomes less inhabitable the more sound is being made, but agents have to make sound to get fit. This contradiction invites the possibility of an *evolutionarily stable state* or an unstable dynamic process (see [3]).

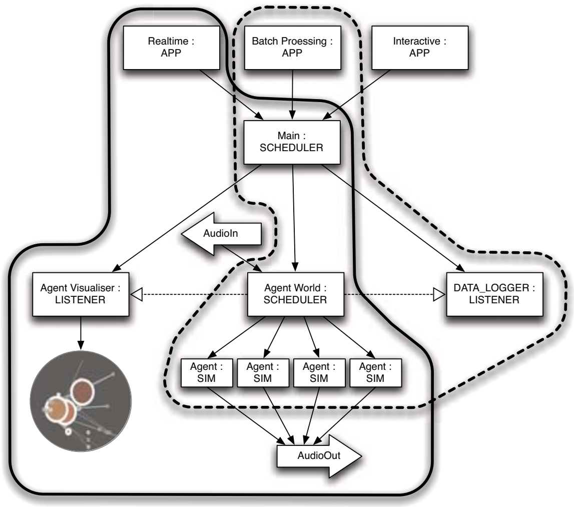


Figure 1. Different model configurations. In the real-time configuration (contained by solid line), the simulation is run with real-time audio and visualisation but no analysis. In the batch processing configuration (contained by dashed line), the simulation is run with non-real-time audio and analysis but no visualisation.

[11] http://www.gnuplot.info/
[12] http://www.wolfram.com/

A basic sonic ecosystem is given by the structure in Figure 1. At the top, three different executable applications are available, each of which sets up a master scheduler and runs it. A configuration file specifying the set-up of the environment and agents can be fed into one of the applications, depending on the required context. The application reads the configuration file and attaches the specified objects to the master scheduler. In the batch processing case, additional configuration data is required (e.g., what parameters to sweep over), but since the applications themselves are only a few lines of code, it is easy to write dedicated applications for specific batch processes. Alternatively, a set of configuration files, one for each parameter set, can be fed into the batch processing application.

There are a number of ways that schedulers, agents and scheduler listeners can be built to reflect the basic design of environment and multi-agent population. In Figure 1, this is achieved by making the environment itself a scheduler (labelled 'agent world'), which sets up the audio context, and then sets itself to be updated from an audio-rate clock, instead of from the main scheduler (it is still *controlled* from the main scheduler). That way we know that environment and agents are always updated at the same ratio with respect to the audio processing. When initialised, the environment generates the initial population of agents from the configuration file, and each agent is attached to the environment's scheduler, which triggers updates in the agents. The scheduler can be configured to perform synchronous or asynchronous updates of the population. After the initial set-up, agents take care of automatically removing themselves from the environment when they die, and adding their offspring to the environment when they reproduce.

In the case of the region enclosed in the dotted line in Figure 1, the population is run in a batch process without real-time audio input and output (determined by the configuration file, but overridden by the batch processing application). A data logger is also attached, which records information about the agents. In the case of the region enclosed in the solid line, the simulation is run in real-time with an additional visualisation unit attached.

As constructed, the creative challenge for such models is to work out how to manipulate aspects of the overall design, such as the resource model, to achieve interesting dynamical behaviour, and also satisfying sonic behaviour, without letting one of these goals eclipse the other. It is assumed that the dynamics of an evolutionary process *could* form the basis for an interesting musical and harmonic structure: one could construct a metaphor in which mass extinction and adaptive radiation act as mechanisms of sudden change, and gradual coevolution leads to complex and harmonic relationships between components. Having glimpsed this possibility, the role of the framework is to facilitate the exploration of the model's behaviour with these combined goals in mind. Since the environment of agents consists largely of other agents, there is fair reason to assume that some variation of such a model exists in which the population continues to evolve over a long period of time, without settling into a stable state, and perhaps even increasing in complexity (this can be see as following a *coevolution* design pattern).

I discuss an example of such a model, in the form of a sonic installation artwork, in [3], which uses the 'available space' resource model discussed above, and demonstrate an example of running a parameter sweep across a number of simulation settings in order to find configurations in which the population of agents divides into two or more species over time. Where such configurations were found, the speciation behaviour is also clear from the audio recording of these runs. In other configurations it was common for separate species to emerge for brief periods, often collapsing back to a single population. The ultimate victory of a single population is a likely outcome of the 'available space' resource model, because agents gain fitness most easily by being more noisy and hogging the audio spectrum: a species that made the noisiest sound would ultimately out-compete less noisy species. This still produced interesting dynamics as the population evolved through various phases of noisiness, often moving through a complex sequence of phases before discovering the noise strategy.

5 SUMMARY

This paper begins by discussing the motivations behind developing new approaches to evolutionary computer music, following an ecosystemic paradigm. Beyond the traditional notion of aesthetic selection, I consider ways to integrate tools from multi-agent evolutionary systems into creative practices. I describe the design of a framework which addresses the design goals established by such an approach. This framework integrates computer music and multi-agent modelling tools into an existing development environment and allows different model configurations to be interactively explored and analysed in a flexible manner. Future work will focus on explicitly formulating a stronger methodology to back this kind of creative exploration, and continuing to develop the framework with this in mind.

6 ACKNOWLEDGEMENTS

I wish to thank Jon McCormack for valuable discussions that contributed to this paper. This research was funded by the Australian Research Council under Discovery Project grant DP0877320.

7 REFERENCES

[1] J. A. Biles, "Autonomous genjam: Eliminating the fitness bottleneck by elim-

inating fitness," 2001, available from http://www.it.rit.edu/j̃ab/GenJam.htmljab/GenJam.html.

[2] T. Blackwell and M. Young, "Live algorithms," http://www.timblackwell.com/, 2005.

[3] O. Bown, "Ecosystem models for real-time generative music: A methodology and framework," in *Proceedings of the 2009 International Computer Music Conference (ICMC 2009)*, Montreal, Canada, forthcoming 2009.

[4] E. Di Paolo, J. Noble, and S. Bullock, "Simulation models as opaque thought experiments," in *Articial Life VII: Proceedings of the Seventh International Conference on Articial Life*, M. A.Bedau, J. S. McCaskill, N. H. Packard, and S. Rasmussen, Eds. Cambridge, MA: MIT Press, 2000, pp. 497–506.

[5] E. Korpela, D. Werthimer, J. Kobb, and M. Lebofsky, "Seti@home – massively distributed computing for seti," *Computing in Science and Engineering*, vol. 3, pp. 78–83, 2001.

[6] J. McCormack, "Artificial ecosystems for creative discovery," in *Proceedings of the 2007 Genetic and Evolutionary Computation Conference*, D. Thierens *et al.*, Eds. ACM, New York, 2007, pp. 301–307.

[7] J. McCormack and O. Bown, "Life's what you make: Niche construction and evolutionary art," in *Applications of Evolutionary Computing: EvoWorkshops 2009*, 2009.

[8] E. R. Miranda and P. M. Todd, "A-life and musical composition: A brief survey," in *IX Brazilian Symposium on Computer Music: Music as Emeregnt Behaviour*, 2003, pp. 59–65.

[9] F. J. Odling-Smee, "Niche construction, evolution and culture," in *Companion Encyclopedia of Antrhopology: Humanity, Culture and Social Life*, T. Ingold, Ed. Oxford, UK: Routledge, 1994.

[10] V. S. Ramachandran, "The artful brain," Talk given at the 2003 BBC Reith Lectures, available from http://www.bbc.co.uk/radio4/reith2003/lecture3.shtml, 2003.

[11] J. Romero and P. Machado, Eds., *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Springer-Verlag: Heidelberg, Germany, 2008.

[12] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[13] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "recaptcha: Human-based character recognition via web security measures," *Science*, vol. 321, pp. 1465–1468, 2008.

REAL-TIME BINAURAL AUDIO RENDERING IN THE NEAR FIELD

Simone Spagnol
University of Padova
spagnols@dei.unipd.it

Federico Avanzini
University of Padova
avanzini@dei.unipd.it

ABSTRACT

This paper considers the problem of 3-D sound rendering in the near field through a low-order HRTF model. Here we concentrate on diffraction effects caused by the human head which we model as a rigid sphere. For relatively close source distances there already exists an algorithm that gives a good approximation to analytical spherical HRTF curves; yet, due to excessive computational cost, it turns out to be impractical in a real-time dynamic context. For this reason the adoption of a further approximation based on principal component analysis, which can significantly speed up spherical HRTF computation, is proposed. The model resulting from such an approach is suitable for future integration in a structural HRTF model and parameterization over anthropometrical measurements of a wide range of subjects.

1 INTRODUCTION

The history of binaural 3-D sound rendering dates back to Lord Rayleigh's well known diffraction formula which approximates the behaviour of a sound wave produced by an infinite point source around the listener's head, thus providing a first crude sketch of what we today call a head-related transfer function (HRTF). On the other hand, most of the relevant issues in this field appeared only recently.

HRTF-based spatial audio rendering can be achieved in multiple ways. Approximations based on low-order rational functions (see e.g. [4]) and series expansions of HRTFs [5, 9] were proposed, resulting in simple yet valuable tools for diffraction modeling. Nevertheless, significant computation is required from both techniques when real-time constraints are introduced, due to the complexity of filter coefficients and weights respectively. This is why structural modeling [2] seems nowadays to be an attractive alternative approach. Within this framework, the contribution of the listener's head, ears and torso to the HRTF can be isolated in several subcomponents, each accounting for some well defined physical phenomenon. Due to linearity of all these effects, they can be later combined meaningfully and realistically in an additive fashion to result in a global HRTF. Such a decomposition yields a model which is both economical and well suited to real-time implementations.

In this paper we will conceptually isolate the earless head of the listener and treat it as a rigid sphere, trying to find a suitable way to represent its contribution to the HRTF. Henceforward we will relate to its transfer function by calling it a spherical HRTF. Furthermore, we will concentrate on sources located in the so-called near field – namely within a few meters from the center of the head – for which real-time computation of HRTFs turns to be more troublesome. Section 2 briefly introduces the theory lying behind the problem. Then, Section 3 presents a PCA-based approach for spherical HRTF modeling. Section 4 deals with the problem of efficient filter modeling. Finally, Section 5 concludes with a discussion on the further work to be done in this direction.

SMC 2009, July 23-25, Porto, Portugal

2 THE SPHERICAL HRTF

2.1 Analytical background

Within the assumption of an infinitely distant source from the center of the head, we can describe the response related to a fixed observation point on the sphere's surface by means of the following transfer function, based on Lord Rayleigh's diffraction formula [1] :

$$H(\mu,\theta) = \frac{1}{\mu^2}\sum_{m=0}^{\infty}\frac{(-i)^{m-1}(2m+1)P_m(cos\theta)}{h'_m(\mu)}, \quad (1)$$

where θ is the incidence angle, the angle between the ray from the center of the sphere to the source and the ray to the observation point, and μ is the normalized frequency, defined as [2]

$$\mu = f\frac{2\pi a}{c}, \quad (2)$$

which is directly proportional to the sphere radius a. Figure 1 shows the magnitude of the transfer function on a dB scale against normalized frequency for 19 different values of incidence angle. When we remove the assumption of an infinitely distant source and consider source positions in

[1] Here P_m and h_m represent, respectively, the *Legendre polynomial* of degree *m* and the *m*th-order *spherical Hankel function*. h'_m is the derivative of h_m with respect to its argument.

[2] Parameter *c* is the ambient speed of sound.

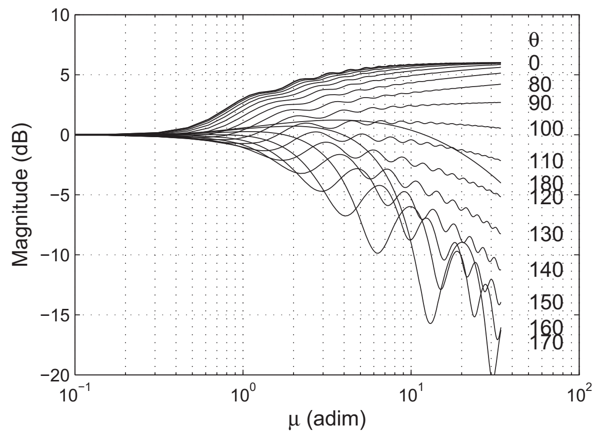


Figure 1. Magnitude response for an infinitely distant source.

the near field, the distance dependence can no longer be ignored. Having defined the normalized distance to the source ρ as the ratio between the absolute distance from the center of the sphere and the sphere radius

$$\rho = \frac{r}{a}, \tag{3}$$

the spherical HRTF can be evaluated by means of the following function [11]:

$$H(\rho, \mu, \theta) = -\frac{\rho}{\mu} e^{-i\mu\rho} \sum_{m=0}^{\infty} (2m+1) P_m(cos\theta) \frac{h_m(\mu\rho)}{h'_m(\mu)}, \tag{4}$$

for each $\rho > 1$. From the analysis of this function we can state a fundamental characteristic of spherical HRTFs: as the source approaches the sphere (ρ tends to 1) the response on the ipsilateral side increases, while the response on the contralateral side decreases [3]. A description of the evaluation algorithm, based on recursion relations, can be found in [8].

2.2 Real-time computation

Let us consider a scenario where the listener is free to move his head with respect to the virtual source to be rendered, and vice versa. It is clear that real-time computation of HRTFs is needed in order to track these movements with enough reactivity, possibly avoiding any discontinuity in the resulting sound. Furthermore, we have to take into account the possibility of having to simulate a complex acoustic environment that includes several independent sound sources, and/or reflections coming from the environment.

Relatively simple HRTF filter structures for sources in the far field have been proposed to date (e.g., Duda's first-order filter in [2]). These turn out to be impracticable in the near field, having no parameterization on source distance. Point-to-point real-time evaluation of Eq. (4) using the algorithm in [8] is computationally still too expensive. Moreover, even if a suitable parameterized filter model is found each source has to be processed with a separate filter. Thus we need to introduce a proper HRTF approximation to speed up the computation. In the next section we discuss such an approximation, which makes use of Principal Component Analysis (PCA) to represent a collection of sample analytical HRTFs.

3 A PCA-BASED APPROACH

3.1 Principal Component Analysis

Principal Component Analysis is used in a number of problems to reduce the dimensionality of an input data set. Its main goal is to provide an efficient representation of a set of correlated measures - in this instance, a set of vectors.

Without delving into deep technicalities (which can be found in [7]), suffice it to say that given a set of n real-valued vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, each of dimension d, and defining its *covariance matrix* $\mathbf{S}$ as

$$\mathbf{S} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}_k \mathbf{x}_k^t, \tag{5}$$

it can be seen that the best p-dimensional representation (with $p \leq d$) of the data set is obtained by taking as basis vectors the p eigenvectors of $\mathbf{S}$ that correspond to the p largest eigenvalues. [3] Each vector $\mathbf{x}_k$ is then projected onto the space defined by the basis vectors as follows:

$$\mathbf{a}_k = \mathbf{C}^t \mathbf{x}_k, \tag{6}$$

where $\mathbf{C}$ is a matrix, the columns of which are the basis vectors. We call principal components the set of weights $\{a_{ki}\}$, $k = 1, \ldots, n$, associated to basis vector i. Now given the set of p-dimensional vectors $\mathbf{a}_k$, $k = 1, \ldots, n$, we can reconstruct an estimate of each original data vector by the inverse equation:

$$\mathbf{x}_k = \mathbf{C}\mathbf{a}_k. \tag{7}$$

Clearly, by increasing the dimension p of the representation the approximation improves. Thus, when dealing with PCA, the main design goal is to extrapolate the value p for which the trade-off between accuracy and data dimensionality is maximized.

PCA has already been used in previous works concerning HRTF modeling [5, 9], with the vectors $\mathbf{x}_k$ representing

[3] An alternative formulation of PCA requires the mean of all vectors in the data set to be subtracted from each one of them before constructing the covariance matrix. However, as the data set we will take into consideration is already well-centered, inclusion of the mean turns out to be quite unnecessary.

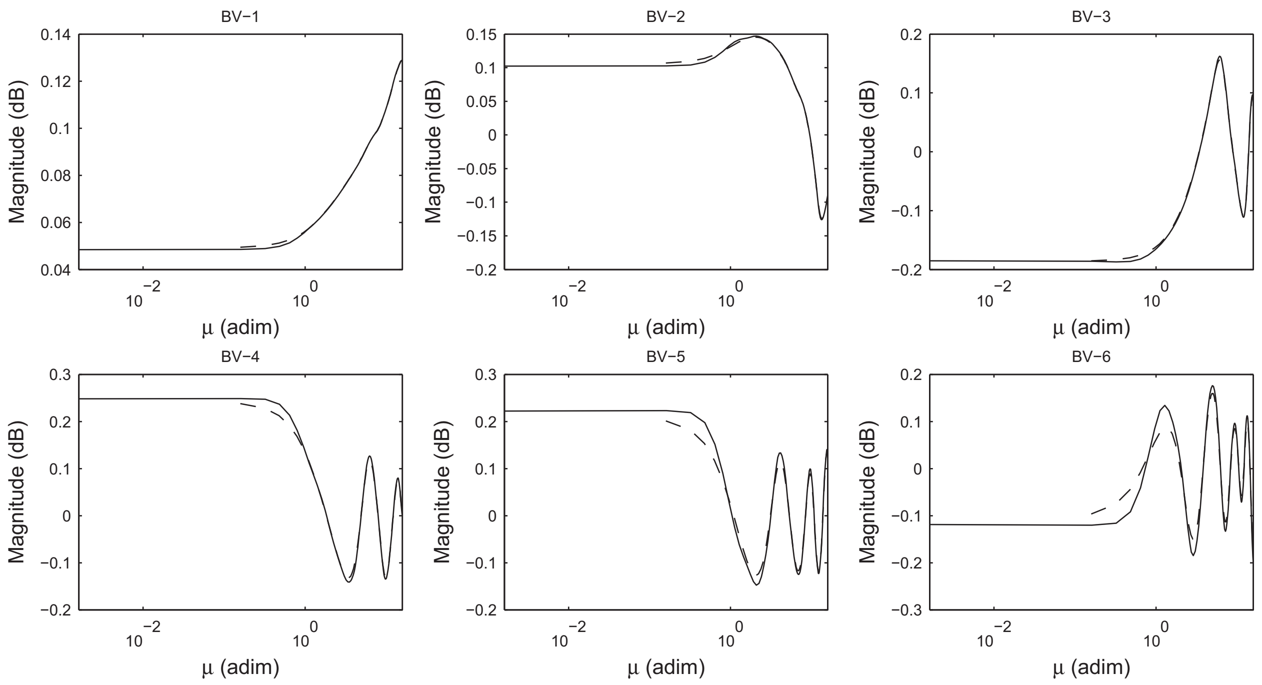


Figure 2. The first six basis vectors (solid lines) and the corresponding least-squares fit 8-th order IIR filters (dashed lines).

magnitude responses of a set of measured HRTFs. However, instead of applying the technique to experimental data, we will exploit it to approximate a collection of spherical HRTF magnitudes sampled from Eq. (4) on a discrete set of frequencies. We will show that, thanks to the decoupling of spatial variables from frequency created by PCA, this approach provides significant computational and storage advantages in the modeling of spherical HRTFs.

3.2 Design choices

We choose to collect a set of spherical HRTFs for sound sorces located at different distances and incidence angles with respect to the ear canal. Being Eq. (4) dependent on only two spatial parameters, in our polar coordinate system we do not consider elevation and restrict these locations to points lying on the horizontal plane. We conventionally assume θ to be the incidence angle at the right ear canal. Therefore $\theta = 0°$, $\theta = 90°$, and $\theta = 180°$ corresponds to a sound source facing the right ear, in front of the head, and facing the left ear, respectively. The set of spherical HRTFs is sampled by fixing the head radius to the standard value $a = 8.75$ cm and varying the following parameters:

- 19 linearly spaced θ values, from $0°$ to $180°$, with $10°$ angle increments;
- 7 exponentially spaced distance values, $\rho = 1.25, 1.5, 2, 4, 8, 16, 32$ (with the last one approximating far field);
- 100 linearly spaced frequency points from 100Hz to 10 kHz, with 100 Hz increments.

We obtain a set of $19 \times 7 = 133$ spherical HRTFs, of which we consider only the dB magnitude responses. Indeed, the HRTF for an ideal sphere appears to be minimum phase for all ranges and incidence angles [8]. In addition, when considering interaural differences for binaural hearing, approximated ITD models (e.g. the Woodworth's formula) can be used to simulate phase lag between right and left ear canal as a simple delay line. Interaural Time Difference (ITD) effects can therefore be cascaded to the HRTF synthesis process.

3.3 Application of PCA

At this point we apply PCA to the set of $n = 133$ real-valued vectors $\mathbf{x}_1,...,\mathbf{x}_n$, each of dimension $d = 100$. The first 6 basis vectors of the analysis are sketched in Figure 2. As we can see, after the first one which accounts for the general slope of the majority of HRTFs (with a positive weight for ipsilateral sources and a negative weight for contralateral ones – see Figure 3), each successive basis vector introduces more and more ripples in the frequency response, starting from the most prominent at $\theta = 170°$.

By investigating the trend of principal components 2 to 6 with the varying of distance and incidence angle we obtain a deeper insight of the analysis. As expected from the observations reported in Section 2.1, weights' moduli are am-

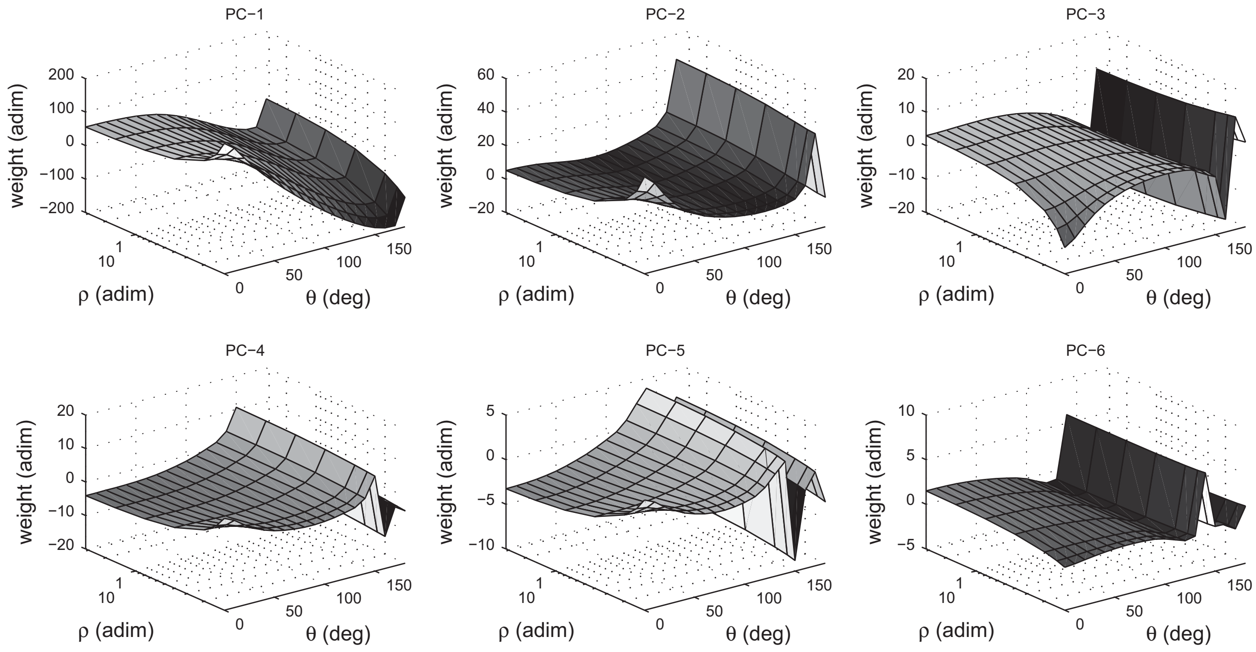


Figure 3. The first six principal components.

plified by decreasing distance; furthermore, Figure 3 shows that each component emphasises its corresponding basis vector only for a limited range of incidence angles, regardless of the distance. This means that the first basis vector retains most of the common variation, while those from the second onwards provide particularized description of the rippled high-frequency behaviour of spherical HRTFs, which varies according to the incidence angle.

3.4 Theoretical optimality

The number of principal components (parameter p) to be included in our model is crucial: as a matter of fact, it denotes the number of filters required to approximate the spherical HRTF by means of the new representation. We need then a proper principle to theoretically quantify the maximum tolerable error, so to extract the minimum p that meets its constraints.

Mills [10] presents a psychoacoustical result which can be used to this purpose. In particular the Interaural Level Difference (ILD) jnd curve as a function of frequency in Figure 4 represents a safe upper bound on the approximation error, owing to insensibility of human hearing apparatus to small changes in ILD (which remarkably denotes the main feature for discriminating source location together with ITD). After having checked that the absolute error between all ILDs derived from a complementary pair of original HRTFs (same distance parameter and sum of incidence angles equal to 180 degrees, assuming diametrically opposite ear canals) and those reconstructed after PCA approximation turns out to lie under the jnd function, we can state there is no significant information loss in our approximation. Note that the jnd function has not been defined for very low frequencies; nevertheless, the dominant localization feature in this frequency range being ITD, ILD information appears to be relevant just for detecting very close distances.

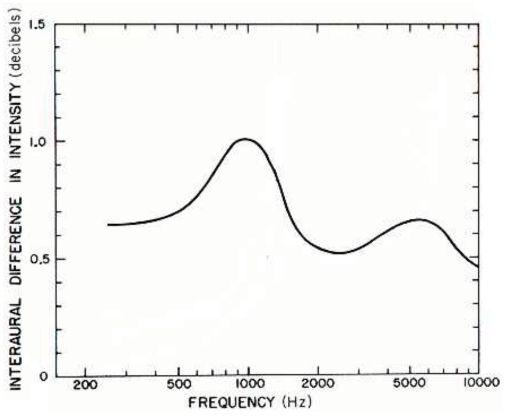


Figure 4. ILD jnd as a function of frequency (figure reproduced from [10]).

As we can see from Figure 5 the minimum value p for which the total error introduced by the PCA approximation

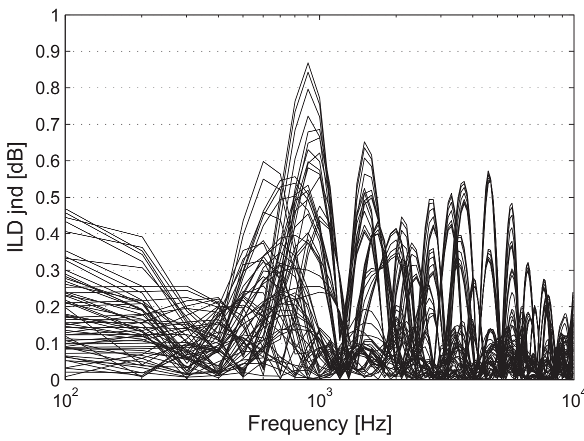


Figure 5. ILD error functions with $p = 7$.

remains below the jnd curve is $p = 7$. In Section 4.2 we will repeat this kind of analysis by including errors due to filter approximation of basis vectors.

4 FILTER REALIZATION

4.1 Filter modeling and interpolation

Each basis vector provides the magnitude response of a filter that has to be realized numerically. Using the least-squares fit procedure provided by the Yule-Walker approach, we design for each basis vector an IIR filter of a desired order, such that it approximates the corresponding magnitude response. In order for the function to work properly we assign a fictional value for zero frequency (we choose this to be the same value as $f = 100$ Hz, as the low-frequency magnitude response is essentially flat) and assume a 20 kHz sampling rate (so that the Nyquist frequency coincides with our 10 kHz limit). Filter coefficients may later be rescaled in case of different sampling rates and different head radii.

It can be seen from Figure 2 that eighth-order filters provide accurate matching of the target magnitude responses. It has to be noted that procedure does not take into account phase requirements. However the resulting filter structures have poles and zeros all inside the unit circle, and are therefore minimum-phase filters.

Having HRTF frequency dependence (now incorporated inside filters characterization) been decoupled from spatial variables dependence, interpolation of spherical HRTFs over spatial points which are not included in the analysis process involves only interpolation of principal components in the form of scalars. To this end, the components plotted in Fig. 3 can be interpolated over distance and incidence angle using simple techniques, e.g 2-dimensional spline interpolation. In particular, in this way any distance value can be rendered (with the upper distance bound in the analysis $\rho = 32$ corresponding to the far field).

Frequency decoupling from spatial variables gives another fundamental advantage. Specifically, the simulation of N independent sound sources located at different positions around the listener head does not require N different filter sets. Instead the set of filters derived above is used for all the sources, with only the components a_i varying for each source. This can be seen in the following equation:

$$\begin{aligned} Y(\mu) &= \sum_{k=1}^{N}\sum_{i=1}^{p} H_{ki}(\rho_k,\mu,\theta_k)X_k(\mu) \\ &= \sum_{k=1}^{N}\sum_{i=1}^{p} H_i(\mu)a_i(\theta_k,\rho_k)X_k(\mu) \qquad (8) \\ &= \sum_{i=1}^{p} H_i(\mu)\sum_{k=1}^{N} a_i(\theta_k,\rho_k)X_k(\mu), \end{aligned}$$

where the N input signals, each with frequency response X_k, are linearly combined through spatial coefficients a_i and filtered by the H_i's, resulting in the output signal with frequency response Y. This result, together with the inclusion of distance dependence and near-field effects in the spherical HRTF, represents the main advantage of the proposed approach with respect to the model described in [2].

4.2 Optimality considerations

The filter realization described in the previous section introduces further error between the real-time model and analytical spherical HRTF curves. Hence, in addition to parameter p, choosing the adequate filter orders $o_1,\dots,o_p$ turns out to be pivotal. To this end, we reapply the ILD jnd criterion in order to determine minimum parameters p and $o_1,\dots,o_p$ that satisfy the forementioned psychoacoustical constraint.

The analysis must be targeted at finding a satisfactory trade-off between accuracy and efficiency. By keeping the minimum value $p = 7$ determined in Section 3.4, it is verified that eighth-order filters ($o_1 = \ldots = o_p = 8$) provide an error which is still below the jnd curve, while seventh-order filters cause 1 dB low-frequency errors. If p is increased by one or more units, using filters of lower order (e.g., 7) still results in errors which are above the psychoacoustical threshold. Intuitively, this circumstance can be explained as follows. Considering that the very first principal components capture the largest part of variance in the data set and have the corresponding basis functions being multiplied by a relatively high coefficient, adding new principal components does not affect the accuracy of the representation as much as properly designing the filters representing each basis vector. Further inspection shows that, since the magnitude responses H_i become increasingly rippled as i grows, the psychoacoustical threshold is satisfied even by choosing filter orders that increase accordingly, i.e. $o_i = i + 1$ $(i = 1 \ldots 7)$.

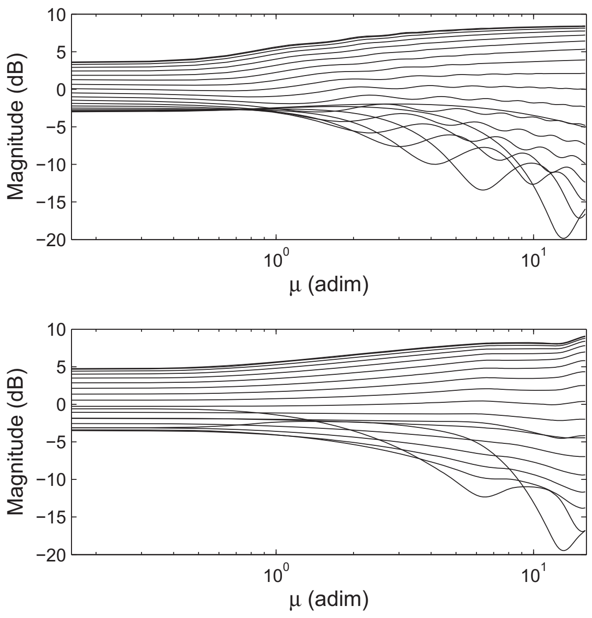


Figure 6. Analytical (top panel) and approximated (bottom panel) spherical HRTF magnitude curves for $p = 3$, $o_1 = o_2 = o_3 = 3$, and $\rho = 4$.

4.3 A low-cost realization

The above discussion is based on purely theoretical assumptions which are very strict. Moreover, the realization proposed in the previous section may have exceedingly high computational costs for real-time applications. In light of this, a more efficient approximation of the spherical HRTF based on a lower number of components and lower-order filters can still be usable even if it does not satisfy the psychoacustical criterion discussed above.

By choosing $p = 3$ and $o_1 = o_2 = o_3 = 3$, the gross magnitude characteristics of the spherical HRTF are still matched, even though the ILD error can be as large as 3 dB at low frequencies. This statement can be verified by looking at Figure 6, which represents reconstructed spherical HRTF magnitude responses for $\rho = 4$ and varying incidence angle. Comparison of the top and bottom panels of the figure confirms that three basis vectors represented with third-order filters already provide a satisfactory approximation.

5 CONCLUSIONS AND FUTURE WORK

In this paper we have presented a PCA-based approach for approximating spherical HRTFs in the near field. We proved that a description in terms of seven eighth-order filters and a set of coefficients turns out to be psychoacustically robust. Much work is still needed in this direction. First, we shall reproduce the analysis in Subsection 3.4 for spatial points that were not included in the synthesis step. Second, the low-cost realization described in Subsection 4.3, possibly along with alternative descriptions, needs to be experimentally evaluated. Third, we need a strong criterion for the personalization of HRTFs based on anthropometrical measurements, analogously to the approach presented in [1]. Finally, we should take into consideration alternative and more realistic head models, like the elliptical one [6].

6 REFERENCES

[1] Algazi, V. R., Avendano, C. and Duda, R. O. “Estimation of a spherical-head model from anthropometry”, *JAES Volume 49, Issue 6, 472-479*, 2001.

[2] Brown, C. P. and Duda, R. O. “A structural model for binaural sound synthesis”, *IEEE Transactions on Speech and Audio Processing, Vol 6, No. 5, 476-488*, 1998.

[3] Brungart, D. S. “Near-field virtual audio displays”, *Presence Vol. 11, No. 1, 93-106*, 2002.

[4] Durant, E. C. and Wakefield, G. H. “Efficient model fitting using a genetic algorithm: pole-zero approximations of HRTFs”. *IEEE Trans. Speech Audio Process., Vol 10, No. 1, 18-27*, 2002.

[5] Chen, J., Van Veen, B. D. and Hecox, K. E. “A spatial feature extraction and regularization model for the head-related transfer function”, *J. Acoust. Soc. Am. 97 (1), 439-452*, 1995.

[6] Duda, R. O., Avendano, C. and Algazi, V. R. “An adaptable ellipsoidal head model for the interaural time difference”, *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing, Phoenix, AZ, 965-968*, 1999.

[7] Duda, R. O., Hart, P. E. and Stork, D. G. *Pattern Classification: Second Edition*. John Wiley & Sons, New York, 2001.

[8] Duda, R. O. and Martens, W. L. “Range dependence of the response of a spherical head model”, *J. Acoust. Soc. Am. 104 (5), 3048-3058*, 1998.

[9] Kistler, D. J. and Wightman, F. L. “A model of head-related transfer functions based on principal components analysis and minimum-phase reconstruction”, *J. Acoust. Soc. Am. 91 (3), 1637-1647*, 1992.

[10] Mills, A. W. “Lateralization of High-Frequency Tones”, *J. Acoust. Soc. Am. 32 (1), 132-135*, 1960.

[11] Rabinowitz, W. M., Maxwell, J., Shao, Y. and Wei, M. “Sound Localization Cues for a Magnified Head: Implications from Sound Diffraction about a Rigid Sphere”, *Presence Vol. 2, 125-129*, 1993.

DOES A "NATURAL" SONIC FEEDBACK AFFECT PERCEIVED USABILITY AND EMOTION IN THE CONTEXT OF USE OF AN ATM?

P. Susini IRCAM susini@ircam.fr

N. Misdariis IRCAM misdarii@ircam.fr

O. Houix IRCAM houix@ircam.fr

G. Lemaitre IRCAM lemaitre@ircam.fr

ABSTRACT

The present study[1] examines the question of a "natural" sonic feedback associated with keys of a numerical keyboard - in the context of use of an Automatic Teller Machine (ATM). "Natural" is defined here as an obvious sound feedback with regards to the action made by a user on a device. The aim is then to study how "naturalness" is related to the perceived usability and the perceived emotion of the sonic feedback before and after participants perform several tasks with the keyboard. Three levels of "naturalness" are defined: causal, iconic, and abstract. In addition, two levels of controlled usability of the system are used: a low level and a high one. Results show that pre-experimental ratings of perceived "naturalness" and perceived usability were highly correlated. This relationship held after the participants interacted with the keyboard. "Naturalness" and emotional aspects were less dependant, revealing that "naturalness" and usability represent a special type of relation. However, results are affected by the level of controlled usability of the system. Indeed, the positive change at the high level of controlled usability for the iconic sounds (medium level of naturalness) obtained after the performance task failes at the low level of controlled usability.

1. INTRODUCTION

Designing new artefacts that may be used in everyday life situations reveals several questions. One of those is related to the interaction between the user and the artefact: is a "natural" relation well appropriated to favour an interaction? By natural, it is meant a causal relation instead of an arbitrary one that tends to be perceived as an obvious display of the interaction based on our everyday experiences (referring to the ecological viewpoint by J.J. Gibson [1], a perceived "natural" interaction could be considered as a perceived affordance). In their recent study Rath et al. [2] defined a "natural" interaction as a 'spontaneous understanding of interaction principles on the side of a user'. Within the framework of the present study, we specify the previous question by examining if a "natural" interaction favours the perceived usability of a device? An intuitive answer to this question is provided by D. Norman [3]: "I believe that our reliance on abstract representations and actions is a mistake and that people would be better served if we would return to control through physical objects, to real knobs, sliders, buttons, to simpler, more concrete objects and actions". In the realm of sound perception, recent trends have focused on everyday listening [4, 5] engaging the cause of the sound event rather than selected specific aspects of the sound signal. Briefly, most of the common listeners focus on the cause of the sound, identifying properties of the source (like the size or the material) and the action made by or to the object. Based on the latter assumption, that we are good at identifying sound events, the design of sound for interactive devices has been proposed using a causal display, rather than abstract one. The hypothesis is that "natural" sonic interactions with virtual objects should be perceived as more intuitive. Thus the question in the present study is to test if a "natural" sonic feedback affects the degree of the perceived usability: 1) before interacting with a device, and 2) after users interact with it, in order to examine if the initial impression holds after a period during which the device is used. In addition, relation between "naturalness" and emotion will be examined. Natural sonic feedback was tested by comparison with designed and with arbitrary sonic feedback on a naturalness dimension.

2. EXPERIMENT

The procedure of the present experiment was partly based on the procedure proposed by Tractinsky et al. in [6]. The experiment was a one between-subjects, one within-subjects full factorial design, with the naturalness as the within-subjects factor, and the level of controlled usability as the between-subject factor. Factor 1 was the naturalness of the sonic feedback, with 3 levels (Low / Medium / High). Factor 2 was the controlled usability of the device, with two levels of usability: low and high. For the high level, the sonic feedback operated each time a key was pressed, and for the low level, the sonic feedback did not operate each time a key was pressed

1 The development of this study was supported by the EC Project CLOSED, FET-NEST no. 29085.

Participants
Two groups of 45 participants performed the main experiment. Their ages varied from 15 to 58 years old. Each group performed step 1 to 4 in one usability condition. No subject reported having any hearing problems.

Apparatus
The keyboard used was a Mobile Numeric USB Keyboard.

Stimuli
The sound corpus was calibrated along a perceptual scale defining three discrete levels of naturalness: i) *high* – causal – corresponded to various keyboard sound recordings ii) *medium* – iconic or designed - corresponded to synthetic sounds having a causal morphological aspect (impacts) superimposed with non natural timbres iii) *low* – abstract – corresponded to sounds having no relationship with the action made on the keyboard (bicycle ring, piano chord, etc. …)

Procedure
Figure 1 presents the different steps of the present procedure.
Step 0: eighty-one sounds were rated by 20 participants on a scale between the labels “not natural at all” and “very natural” in terms of relation between the sound display and the action of pressing a key on a keyboard. Finally, 9 sounds were selected: three at high level, three at an intermediate one, and three at the lower level of naturalness (see details above)
Step 1: the 9 selected sounds were rated on a nine-points scale by two groups of 45 participants on five scales (Naturalness / N, Usability 1 / U1, Usability 2 / U2, Pleasantness / P and Stimulating / S). U1 and U2 were described to the participants as scales related to the a priori perceived usability of the sound (before using the keyboard in step 2). For example, for scale Usability 1 / U1, participants were asked to rate the assertion: “I find that the sound is well associated with the keys”. P and S were associated to the two usual emotional dimensions used in the realm of research in emotions [7]. For example, for scale Pleasantness / P, participants were asked to rate the assertion: “I find that the sound is pleasant”. For each scale, the mark 1 was associated with “I don’t agree at all” and the mark 9 with “I completely agree”.
Step 2: based on their evaluations in step 1, each participant was assigned at one of the three level of “naturalness” (H/M/L), and then performed step 2 using one sound that she/he evaluated at the corresponding level of “naturalness”. In step 2, using the numerical keyboard, different tasks were performed several times like withdrawing cash and transferring an amount of money between two bank accounts.
Step 3: the sound used during the performance task was rated on the same five scales as in step 1.
Step 4: participants were asked to rate directly if the sound was finally worse or better compared to their initial impression on each scale, which corresponds to the direct difference between post and pre-experimental perception. For example, for the pleasantness, the assertion proposed was: “I find that the sound is *more* pleasant”. Evaluations were made on the scale [-4, +4]; a negative mark meant that the participant did not agree, and a positive one that she/he agreed.

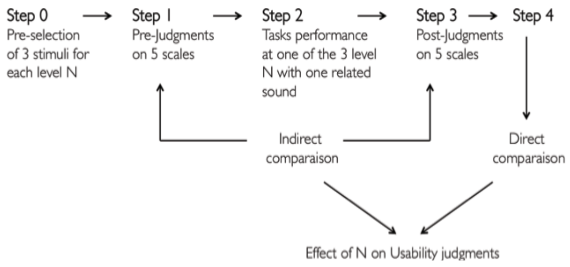

Step 0
Pre-selection of 3 stimuli for each level N
Step 1
Pre-Judgments on 5 scales
Step 2
Tasks performance at one of the 3 level N with one related sound
Step 3
Post-Judgments on 5 scales
Step 4
Indirect comparaison
Direct comparaison
Effect of N on Usability judgments

Figure 1. Step 0 to 4 of the experimental procedure

3. RESULTS

3.1 Manipulation check

Figure 2.a and 2.b displays the pre-experimental mean ratings for the naturalness scale in order to check that the 9 sounds were correctly judged with the expected level of naturalness. The mean value for each sound corresponds to the average of the 45 evaluations on this scale respectively for group 1 and group 2. As it can be observed on figures 2.a and 2.b, pre-selected sounds from step 0 were judged in step 1 with the same level of naturalness as it was expected. On the figures, the first three sounds are labelled respectively H1, H2 and H3 as they were expected to be perceived with the highest level of naturalness, the three next sounds are labelled respectively M1, M2 and M3 for the medium level of naturalness, and the last three ones are labelled L1, L2 and L3 respectively for the lowest level of naturalness. A repeated-measures analysis of variance (ANOVA) was performed on pre-experimental ratings obtained on the naturalness scale with one within-subject factor, sound (9 levels), and with one between-subject factor, group of participants (2 levels). The analysis reveals a strong effect of the sound factor (F(8, 704)=167.8, $p < 0.001$) and no effect of the group factor, as well as no interaction between the two main factors which means that ratings did not depend on the group factor. As it was expected that the first three sounds [H1, H2, H3] will be judged with a higher level of naturalness compared to [M1, M2, M3] and [L1, L2, L3], and [M1, M2, M3] to be judged with a higher compared to [L1, L2, L3], contrast analyses were performed in order to test if perceived naturalness ratings were different between the three groups of sounds. Results showed that perceived naturalness was significantly different between [H1, H2, H3] and the two other groups of sounds ($p<0.001$), and between [M1, M2, M3] and [L1, L2, L3] ($p<0.001$)

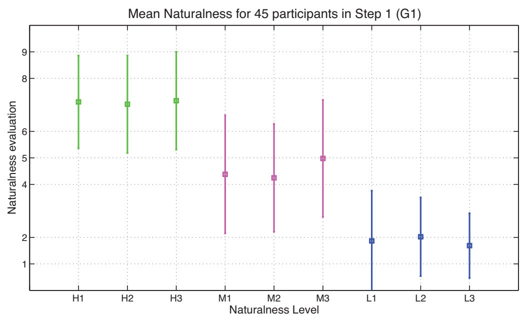


Figure 2.a Naturalness pre-experimental mean ratings on 45 participants from Group 1

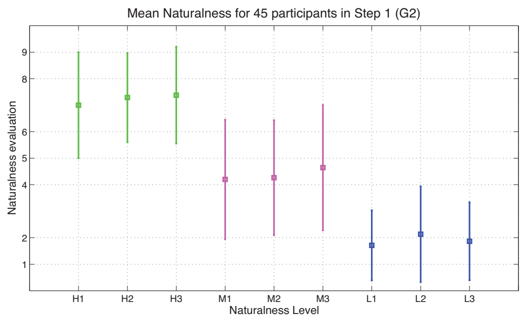


Figure 2.b Naturalness pre-experimental mean ratings on 45 participants from Group 2

3.2 Analysis of the pre-experimental ratings

3.2.1 Reliability

For the two groups of subjects, the repetition (test and retest) factor is examined considering individual ratings on each of the five scales. Test-retest reliability obtained is r=0.83, r=0.78, r=0.71, r=0.69 and r=0.60 ($p < 0.001$) respectively for the five scales (Naturalness / N, Usability 1 / U1, Usability 2 / U2, Pleasantness / P and Stimulating / S). Since the correlation values for several scales were not very high, even if they were statistically significant, datasets were not aggregated. In addition, participants reported to have been more confident in their second ratings. Based on participants' comments, only the ratings from the second set were kept for further analysis.

3.2.2 Results presentation

Figure 3.a and 3.b displays the pre-experimental mean ratings of the five scales (Naturalness, Usability 1, Usability 2, Pleasantness, Stimulating) for respectively group 1 and 2 (high and low usability level). The mean values for each level of naturalness for the five scales are also presented in Table 1.

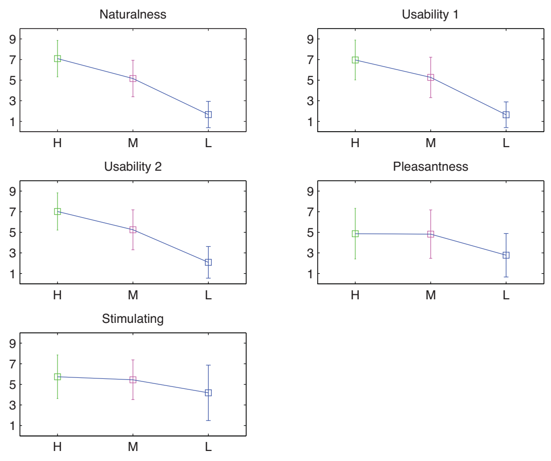


Figure 3.a Ratings on the five scales for the sounds assigned to one of the three level of naturalness (High, Medium, Low) and for the **high** level of the controlled usability.

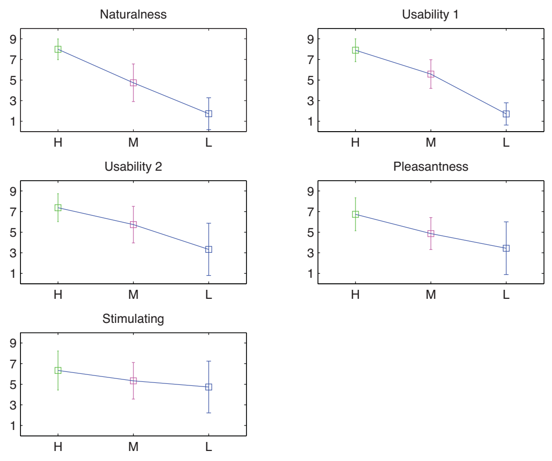


Figure 3.b Ratings on the five scales for the sounds assigned to one of the three level of naturalness (High, Medium, Low) and for the **low** level of the controlled usability.

Naturalness level	Pre-experimental scale	Controlled Usability	
		High	Low
High	Naturalness	7.08 (1.76)	7.97 (1.01)
	Usability 1	6.95 (1.93)	7.88 (1.11)
	Usability 2	7.02 (1.80)	7.37 (1.35)
	Pleasantness	4.86 (2.45)	6.73 (1.60)
	Stimulation	5.73 (2.11)	6.33 (1.89)
	N	*15*	*15*
Medium	Naturalness	5.15 (1.77)	4.73 (1.82)
	Usability 1	5.26 (1.95)	5.57 (1.38)
	Usability 2	5.24 (1.93)	5.73 (1.77)
	Pleasantness	4.82 (2.34)	4.86 (1.54)
	Stimulation	5.44 (1.92)	5.33 (1.77)
	N	*15*	*15*
Low	Naturalness	1.66 (1.27)	1.73 (1.54)
	Usability 1	1.64 (1.24)	1.71 (1.07)
	Usability 2	2.08 (1.53)	3.33 (2.53)
	Pleasantness	2.77 (2.10)	3.44 (2.55)
	Stimulation	4.17 (2.69)	4.73 (2.50)
	N	*15*	*15*

Table 1. Pre-experimental mean ratings (standard deviations in brackets) on the five scales and for both level of usability

Results show clearly that the degree of perceived usability U1 and U2 decreased with the same amount as the perceived naturalness N, whereas the amount of change of the perceived pleasantness P is slightly less important. Finally, the level of stimulation (S) seems independent of the three level of naturalness. This result indicates that participants perceived a stronger relation between naturalness and usability rather than between naturalness and emotional aspects.

3.2.3 Analysis of variance

A multivariate analysis of variance (MANOVA) with repeated measures was conducted on the five dependent variables (Naturalness, Usability 1, Usability 2, Pleasantness, Stimulating), using a full-factorial design, with the following two between-subject factors: level of naturalness (3 levels, H, M and L) and group of participants (2 levels, G1 and G2). A MANOVA was performed instead of an ANOVA to take into account correlations between ratings on similar scales such as U1 (Usability 1) and U2 (Usability 2), for example.
The main interest here is to determine whether the naturalness factor has had a global effect on ratings for the five scales. Results show, as it was expected, that the factor group neither had an effect on ratings nor interacted with the factor naturalness. On the other hand, the multivariate analysis of variance (MANOVA) reveals an overall significant effect of the naturalness factor (Wilks' lambda value, F=32.0, $p<0.001$). One-way ANOVAs show that the effect is significant for each scale. The percentage of total variance accounted for by each effect is indicated by the R^2 coefficient. The main effect of naturalness accounts for about 81, 58, 32 and only 15% of the total variance respectively for scales U1, U2, P, and S. Thus the strongest effect of the naturalness factor is obtained for ratings on U1 and U2. This result corroborates descriptions provided in the previous section.

3.3 Comparison between Pre and Post-experimental ratings

3.3.1 Correlation analysis

Inter-correlations among the perceived measures are presented in table 2.a and 2.b. Pre experimental ratings of perceived Naturalness and perceived Usability 1&2 were highly correlated (respectively r=0.9 and r=0.8). The same results were obtained for Post experimental ratings (respectively r=0.85 and r=0.71) meaning that the correlations between perceived Naturalness and Usability 1&2 remained high even after the performance task. On the other hand, Pre-experimental ratings of perceived Naturalness were less correlated with the scale Pleasantness (r=0.62) and weakly with the scale Stimulating (r=0.44). The weakest correlations were obtained for ratings on the Stimulating scale and the other scales for both Pre and Post experimental ratings (respectively $0.44 \leq r \leq 0.56$ and $0.36 \leq r \leq 0.42$). This indicates that the level of correlation with the Naturalness scale depends on the type of scale. Pre and Post-experimental correlations of Naturalness were relatively high (r=0.78), and Pre and Post-experimental correlation of perceived Usability 1&2 were lower (respectively r=0.65 and r=0.55), as well as Pre and Post-experimental correlation for scales Pleasantness and Stimulating (respectively r=0.34 and r=0.41). There was a slightly difference between the two usability groups related to the perceived Pleasantness; the Pre and Post-experimental correlation were 0.39 and 0.29 respectively for the high and low-usability groups.

	Pre-U1	Pre-U2	Post-N	Post-U1	Post-U2
Pre-N	0.90***	0.80***	0.78***	0.69***	0.54***
Pre-U1		0.79***	0.72***	0.65***	0.52***
Pre-U2			0.65***	0.59***	0.55***
Post-N				0.85***	0.71***
Post-U1					0.71***

Table 2.a Correlation matrix of pre and post-experimental measures (N=90) for the Naturalness (N), the Usability 1 (U1) and the Usability 2 (U2) scales (***$p<0.0001$, **$p<0.005$, *$p<0.05$)

	Pre-P	Pre-S	Post-N	Post-P	Post-S
Pre-N	0.62***	0.44***	0.78***	0.54***	0.32**
Pre-P		0.56***	0.47***	0.34**	0.28**
Pre-S			0.28*	0.23*	0.41**
Pots-N				0.72***	0.36**
Post-P					0.42***

Table 2.b Correlation matrix of pre and post-experimental measures (N=90) for the Naturalness (N), Pleasantness (P) and Stimulating (S) scales (***$p<0.0001$, **$p<0.005$, *$p<0.05$)

3.3.2 Results presentation

Figure 4.a and 4.b display the average ratings for the two groups of participants (respectively the two levels of usability) obtained in step 4 showing how each level of naturalness was perceived on the five scales before and after interacting with the keyboard in step 2. A positive value indicates that positive change in ratings between before and after the performance task.

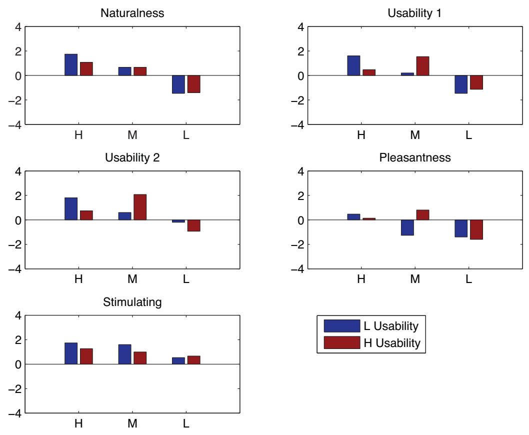


Figure 4. Direct estimation of the difference between post and pre-experimental perception on the five scales and for both level of usability

Analysis of variance
A multivariate analysis of variance (MANOVA) with repeated measures was conducted on the direct estimation of the difference between post and pre-experimental perception on the five scales. The main interest here is to compare ratings obtained for the two usability levels in order to examine effect of the controlled usability on perceptive ratings. Thus, the main null hypothesis tested is: "the experimental condition, controlled usability, does not have any effect on perceptive ratings". The MANOVA reveals an overall significant effect of the naturalness factor (Wilks' lambda value, F=4.42, $p<0.001$) but no effect of the controlled usability factor. On the other hand, the analysis reveals a significant interaction between these two factors. It thus appears that the effect of the controlled usability (between-subjects factor) is present but it varies as a function of naturalness (within-subjects factor). One-way ANOVAs reveal that the naturalness factor is significant for all the scales except for the scale Stimulating. In addition, the analyses reveal that the significant interaction is obtained only for perceived Usability 1&2 (F(2, 84)=15.7, $p<0.001$ and F(2, 84)=7.8, $p<0.001$, respectively) and for Pleasantness (F(2, 84)=6.64, $p<0.01$). Contrast analyses show that the interactions obtained for these three scales are based on significant difference only for the medium level (M) of Naturalness, while there is no significant difference between the two other levels of Naturalness. These analyses reveal that the controlled usability affect only the medium level of naturalness that corresponds to the iconic (designed) sounds. As it can be seen in figure 4, for the low level of controlled usability (L Usability), iconic sounds were judged after the performance task to be less usable and pleasant.

4. CONCLUSION

This study examined whether a “natural” sonic feedback affects the degree of the perceived usability and emotion in a simple interaction. Results show that:
- the naturalness scale related to the three groups of sounds was perceived as expected in the context of use of the keyboard.
- initial impression of naturalness affects the initial impression of usability and holds after participants interact with the keyboard. On the other hand, ratings on the emotional scales are less affected.
- the level of controlled usability interacts mainly with the medium level of naturalness. Sounds at this level were designed sounds (synthetic impact sounds), well perceived prior to the performance task (pre-judgment) and finally perceived to be more useful in a high level of usability, but this impression failed in the low usability condition.
- whatever the situation, an abstract sound, corresponding to a low level of naturalness, is not perceived to be useful and pleasant, and even worse after performing with the keyboard.

To summarize the major findings of this study, the results suggest that it looks like the acceptance of an iconic (designed) sound is weaker when the sound is not efficient for the expected feedback (dysfunctioning system). However, a causal sound is still accepted when the system does not work correctly.

5. REFERENCES

[1] Gibson, J. J. (**1966**). *The Senses Considered as Perceptual Systems* (Houghtin Mifflin, Boston).

[2] Rath, M., and Schleicher, R. (**2008**). "On the Relevance of Auditory Feedback for Quality of Control in a Balancing Task," Acta Acustica united with Acustia 94, 12-20.

[3] Norman, D. A. (**1999**). " Affordance, conventions, and design," Interactions 6 6, 38-43.

[4] Gaver, W. W. (**1993**). "How do we hear in the world? Explorations in ecological acoustics," Ecological Psychology 5, 285-313.

[5] Gaver, W. W. (**1993**). "What do we hear in the world? An ecological approach to auditory event perception," Ecological Psychology 5, 1-29.

[6] Tractinsky, N., Adi, S.-K., and Ikar, D. (**2000**). "What is Beautiful is Usable," Interacting with Computers 13, 127-145.

[7] Russell J. A., “A circumplex model of affect”, Journal of personality and social psychology, vol. 39(6), pp. 1161-1178, 1980.

FREE CLASSIFICATION OF VOCAL IMITATIONS OF EVERYDAY SOUNDS

Arnaud Dessein, Guillaume Lemaitre
dessein@ircam.fr, lemaitre@ircam.fr

IRCAM – 1, place Igor Stravinsky – 75004 Paris

ABSTRACT

This paper reports on the analysis of a free classification of vocal imitations of everyday sounds. The goal is to highlight the acoustical properties that have allowed the listeners to classify these imitations into categories that are closely related to the categories of the imitated sound sources. We present several specific techniques that have been developed to this end. First, the descriptions provided by the participants suggest that they have used different kinds of similarities to group together the imitations. A method to assess the individual strategies is therefore proposed and allows to detect an outlier participant. Second, the participants' classifications are submitted to a hierarchical clustering analysis, and clusters are created using the inconsistency coefficient, rather than the height of fusion. The relevance of the clusters is discussed and seven of them are chosen for further analysis. These clusters are predicted perfectly with a few pertinent acoustic descriptors, and using very simple binary decision rules. This suggests that the acoustic similarities overlap with the similarities used by the participants to perform the classification. However, several issues need to be considered to extend these results to the imitated sounds.

1 INTRODUCTION

1.1 Framework

Vocal imitations are very commonly and spontaneously used in everyday conversations when trying to describe a sound. Two kinds of imitations have to be distinguished: standardized imitations (i.e. onomatopoeias) and non-standardized ones. Onomatopoeias are words, the spelling of which is conventional, and the meaning shared by a given population. "Cock-a-doodle-doo" is an example of onomatopoeia in English: every English listener knows that this word labels the cry of a rooster, but its pronunciation might be somehow different from the rooster's cry. On the contrary, non-standardized imitations occur when a speaker tries to imitate a sound with any means of vocal production, without using standardized words. Whereas there is a finite number of onomatopoeias in a language, the variety of vocal imitations potentially occurring in conversations is virtually infinite.

SMC 2009, July 23-25, Porto, Portugal

The study reported here focuses on non-standardized vocal imitations. The assumption is made that such imitations are simplifications of the imitated sounds, which still allow the recognition of what has been imitated. Therefore, the production of vocal imitations is believed to provide a relevant paradigm for the study of how human listeners identify sound sources. Sound source identification and the perception of everyday sounds have become an important domain of research since the 90's [6, 7], the potential applications of which are manyfold in audio content analysis or sound synthesis. More specifically, studying sound source identification and vocal imitations is expected to inform the development of *cartoonification*, a particular method of sound synthesis that consists in exaggerating some acoustic features while discarding some others [18]. The advantages of such a technique are that it renders the information clearer, more effective, while reducing the computational cost.

1.2 State-of-the-art

Vocal imitations have been studied from different perspectives. Laas et al. [11,12] showed that listeners could identify fairly well human-imitated animal sounds, and that the identification performances were sometimes even better with imitations than with real animal sounds. Nevertheless, the authors do not explicitly mention whether participants listened to the sounds to imitate or were given the names of the animals to imitate, and whether they could used onomatopoeias or not. Therefore, the successful identification might be accounted for the conventionality and symbolism in the imitations. Other studies have reported systematic patterns of associations between phonetic properties of the imitations and acoustical properties of the imitated sounds [2,4,21,22]. For example, plosives are very commonly used to imitate short sounds or sounds with brutal onsets, such as impacts, explosions. Fricatives are used to imitate sounds with smooth onsets, such as the wind, a breath. The length of imitations is related to the duration of the sounds, or to the number of distinct elements composing the sounds. This shows that vocal imitations can mimic various temporal aspects of sounds. However, there are always imitations that do not verify these rules, and some imitations are better than others. Other studies were interested in vocal imitations of tabla drumming sounds [15], strange machine sounds [14, 20], impulse sounds [8, 9], various sounds [10], flue organ pipe sounds [17], sounds of laser printers and copy machines [19]. They show that many spectro-temporal properties of the sounds are reproduced in the imitations: duration, range of frequency, spectral centroid, transients, etc.

Overall, this review of the literature points out several issues. Not all imitations allow perfect identification. The quality of an imitation can be related to the capacities of the vocal apparatus, the performance of the imitators, the difficulty to imitate a given sound. The degree of conventionality and symbolism of the imitations is variable. This can be linked with the nature of the imitations (onomatopoeias, non-word phonetic imitations or non-phonetic imitations).

1.3 Outlines of the study

The analysis reported in this article is based on the results of an experimental study described in [3]. This study provides a set of vocal imitations of everyday sounds that have been categorized by a group of listeners. These imitations are non-standardized, and some of them are even difficult to transcribe phonetically. The goal of this paper is to highlight the acoustical properties that have allowed the listeners to classify the imitations into categories that are closely related to the categories of the imitated sound sources. This paper reports on the specific techniques that have been developed to this end, as well as the results of the analyses.

The experimental study is reported in Section 2. The participants' strategies are analyzed in Section 3, with a specific technique to detect the outliers, using the R_V coefficient. The categories provided by the participants are analyzed in Section 4, with hierarchical clustering and the inconsistency coefficient. The acoustical properties accounting for the categories of imitations are finally highlighted in Section 5.

2 FREE CLASSIFICATION OF IMITATIONS

2.1 Recordings

Vocal imitations were recorded for a set of environmental sounds. The imitated sounds were selected from a corpus of sounds recorded in a kitchen. These sounds had already been used in other experimental studies reported in [13]. Particularly, they had been used in a free classification task. Therefore, the perceptual organization of these sounds into categories of sound sources is available (the 4 main categories are liquid, solid, gas, electric). During the recording session, the participants listened to each sound to imitate and had three trials to record an imitation. They were explicitly asked not to use words, in particular onomatopoeias.

2.2 Method

Twelve sounds were chosen: 3 liquid sounds L_1, L_2, L_3, 3 solid sounds S_1, S_2, S_3, 3 gas sounds G_1, G_2, G_3, and 3 electric sounds E_1, E_2, E_3. Six imitators were chosen: 3 women W_1, W_2, W_3, and 3 men M_1, M_2, M_3. The 6 imitations of each of the 12 sounds gave a corpus of $n = 72$ vocal imitations that were used in a free classification experiment. Participants had first to group together the imitations so as to form different classes. They could create as many classes as they wished, and did not receive any specific instruction on how to form the classes. Then, they had to freely describe each class they had made. For each participant p, the results of the classification were encoded in a $n \times n$ matrix $\mathbf{D}_p$, called *distance matrix*, such that:

$$d_{ij}^{(p)} = \begin{cases} 0 & \text{if sounds } i \text{ and } j \text{ were grouped together;} \\ 1 & \text{else.} \end{cases} \tag{1}$$

3 THE PARTICIPANTS' STRATEGIES

3.1 Descriptions of the categories

The descriptions of the categories provided by the participants are not systematically analyzed here. They suggest however that the participants have used different kinds of similarities to group together the sounds (according to the typology defined in [13]). Indeed, most of the descriptions mention *causal* and *semantic* similarities (i.e. the cause and the meaning associated with the identified sources). But other similarities were also used: acoustical properties of the sounds, feelings (called here *hedonic* properties), means of vocal production (see Table 1). For some participants, the description of a given class sometimes mentions several kinds of similarities (e.g. "Continuous sounds, with a kind of vibration, with the lips, the throat, there is something spinning, noises of machines"). Furthermore the descriptions provided by a participant suggest that he made classes in a rather random fashion.

Similarity	Examples of descriptions
Causal / Semantic	"Mechanical actions of slicing" "Water dripping" "All kinds of drilling machines, food processors"
Acoustic	"Loud and rhythmic sounds" "Repeated, percussive sounds" "Continuous sounds"
Hedonic	"Very aggressive, catches attention" "Suffering" "Mentions the comfort"
Vocal production	"Throat noises" "Expiration with a whistle on the tongue" "With the lips"

Table 1. Examples of descriptions given by the participants, sorted into different kinds of similarities.

3.2 Individual classifications

The descriptions of the classes suggest different strategies across the participants, and even an outlier behaving randomly. There is however no widespread method to analyze individual differences in classification experiments. We used here a method inspired from [1]. It consists in computing a measure of pairwise similarity between the individual classifications. It is also possible to add random individual classifications in order to detect potential outliers.

3.2.1 A measure of pairwise similarity

The R_V coefficient [5] is a measure of similarity between two symmetric matrices $\mathbf{X}$ and $\mathbf{Y}$ and is given by:

$$R_V(\mathbf{X}, \mathbf{Y}) = \frac{\operatorname{trace}(\mathbf{X}\mathbf{Y}^T)}{\sqrt{\operatorname{trace}(\mathbf{X}\mathbf{X}^T)\operatorname{trace}(\mathbf{Y}\mathbf{Y}^T)}} \qquad (2)$$

Therefore, it can be used as a measure of pairwise similarity between individual classifications. Following [1], the R_V coefficient is not computed here directly between the distance matrices, but between the individual normalized (with respect to the spectral radius) *cross-product matrices*. The cross-product matrix $\tilde{\mathbf{S}}_p$ for participant p is given by:

$$\tilde{\mathbf{S}}_p = -\frac{1}{2}\,\mathbf{C}\mathbf{D}_p\mathbf{C}^T \qquad (3)$$

where $\mathbf{D}_p$ is the distance matrix of participant p. The $n \times n$ matrix $\mathbf{C}$ is called a *centering matrix* and is given by:

$$\mathbf{C} = \mathbf{I} - \mathbf{1} \cdot \mathbf{m}^T \qquad (4)$$

where $\mathbf{I}$ is the $n \times n$ identity matrix, $\mathbf{1}$ is a column vector of length n filled with ones, and $\mathbf{m}$ a column vector of length n called *mass vector* and composed of positive numbers whose sum is equal to 1. Here, all observations are of equal importance so we set each element of $\mathbf{m}$ equal to $\frac{1}{n}$.

3.2.2 Distances between participants

The *between-participant similarity matrix* $\mathbf{R_V}$, whose coefficients $[\mathbf{R_V}]_{ij} = R_V(\mathbf{S}_i, \mathbf{S}_j)$ are the R_V coefficients between the normalized cross-product matrices $\mathbf{S}_i$ and $\mathbf{S}_j$, is then constructed. A principal component analysis (PCA) is applied on $\mathbf{R_V}$ and is represented in Figure 1 using the two principal components. In this map, a kind of distance between the participants is represented since the proximity between two points reflects their similarity. We also added random normalized cross product matrices in order to simulate random individual results. The participant P_{13} that we suspected to be an outlier is closer to the random participants than the other real participants. He was therefore excluded for the rest of the analysis. However, the presented technique did not allow to highlight different strategies across the participants, even when using more dimensions and removing the random individual results.

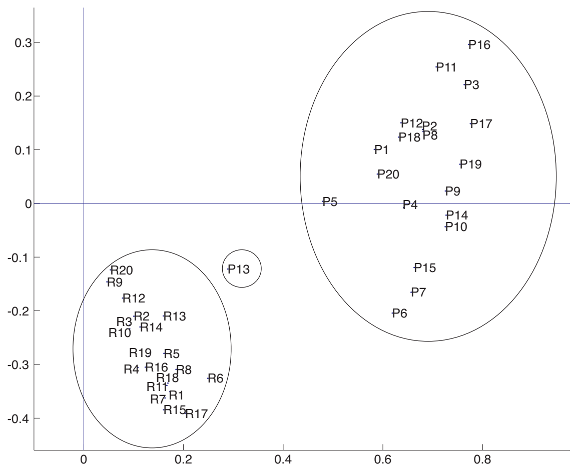


Figure 1. Representation of the distances between the participants using the two principal components of the PCA applied on the between-participant similarity matrix $\mathbf{R_V}$, with the real participants (P) and random participants (R).

4 ANALYSIS OF THE CLASSIFICATION

4.1 Hierarchical clustering

The average distance matrix $\mathbf{D}$ across the individual matrices $\mathbf{D}_p$ was submitted to a *hierarchical clustering* analysis, which represents the average distances in $\mathbf{D}$ with a tree called *dendrogram*. In this tree, the distance between two items (here vocal imitations) is represented by their *height of fusion* (i.e. the height of the node linking the two items).

To identify significant clusters of items, the dendrogram is usually cut at a given height of fusion. As an alternative clustering method, we propose here to use a threshold of *inconsistency*. The advantage of the inconsistency is to emphasize compact subclasses that would not be revealed using the height of fusion. The inconsistency coefficient characterizes a given node by comparing its height of fusion with the respective heights of fusion of its non-leaf subnodes:

$$\text{inconsistency} = \frac{\text{height of fusion} - \mu_d}{\sigma_d} \qquad (5)$$

where μ_d and σ_d are respectively the mean and the standard deviation of the height of fusion of the d highest non-leaf subnodes. The depth d specifies the maximum number of non-leaf subnodes to include in the calculation. The maximum number is used if there are enough non-leaf subnodes, otherwise all non-leaf subnodes are included. The inconsistency coefficient of a given node is positive, having a value set to 0 for leaf nodes, and increasing with the inner dissimilarity of the objects merged by that node.

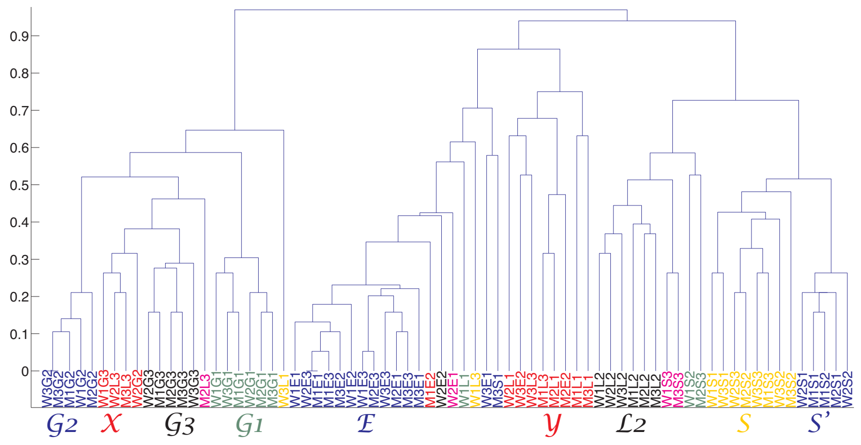


Figure 2. Dendrogram of the vocal imitations (labeled using the imitator's label followed by the label of the imitated sound).

4.2 Dendrogram of the vocal imitations

The dendrogram of the imitations is represented in Figure 2 (using the unweighted average linkage method). We created the clusters with a threshold of inconsistency equal to 1.45 (using a maximal depth so that for each node, all its non-leaf subnodes are included in the calculation). We chose this threshold by decreasing the inconsistency, and so increasing the number of clusters, until the created clusters did not seem coherent to us anymore. It is important to note that the 6 imitations of a given sound are not systematically in the same cluster — in fact, only the sounds G_1 and L_2 have their 6 imitations clustered together. Our hypothesis is that it is related to the quality of the vocal imitations of a given sound, or at least to the agreement between participants on the way to imitate a given sound. As we want to highlight common acoustic invariants in the imitated sounds, we focused on 7 clusters that seemed relevant to us:

(1) $\mathcal{G}_1$ made up of 6 imitations of the gas G_1;

(2) $\mathcal{G}_2$ made up of 5 imitations of the gas G_2;

(3) $\mathcal{G}_3$ made up of 5 imitations of the gas G_3;

(4) $\mathcal{E}$ made up of 12 imitations of electric sounds;

(5) $\mathcal{L}_2$ made up of 6 imitations of the liquid L_2;

(6) $\mathcal{S}$ made up of 8 imitations of solids;

(7) $\mathcal{S}'$ made up of 5 imitations of solids.

We rejected $\mathcal{X}$ because it contains 2 imitations of a liquid and 1 imitation of two gases. We also rejected $\mathcal{Y}$ because although it contains 4 imitations of the same liquid L_1, it also contains 2 imitations of another liquid and 2 imitations of an electric sound, and because its node of fusion is quite high. Finally, we did not consider the clusters with 1 or 2 imitations (the other clusters gather at least 4 imitations).

5 ACOUSTIC PROPERTIES OF THE IMITATIONS

The goal of the analysis reported in this section is to predict the classification of the 7 clusters described above from the acoustical properties of the sounds. We used binary decision trees with a few relevant acoustic descriptors. The descriptors were computed with the IrcamDescriptor toolbox [16].

5.1 First level of the hierarchy

As a first step, we considered the first 3 classes in term of height of fusion: $\mathcal{G}$ composed of $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_3$, $\mathcal{E}$ as described previously, and $\mathcal{R}$ composed of $\mathcal{L}_2$, $\mathcal{S}$ and $\mathcal{S}'$. To explain these classes, we chose 2 descriptors: (MA) the modulation amplitude of the energy envelope to discriminate between the sounds with a repetitive pattern in $\mathcal{R}$ and the one-block sounds in $\mathcal{G}$ and $\mathcal{E}$, and (MSC) the loudness weighted mean of the perceptual spectral centroid to discriminate between the unvoiced imitations with a high-frequency noisy part in $\mathcal{G}$ and the voiced imitations with a relatively low fundamental frequency in $\mathcal{E}$. The classes are perfectly discriminated (see Figure 3) with the following rules:

(1) $\mathcal{G}$: MA < 0.301208 and MSC ≥ -0.0697998;

(2) $\mathcal{E}$: MA < 0.301208 and MSC < -0.0697998;

(3) $\mathcal{R}$: MA ≥ 0.301208.

One may wonder if these rules generalize well if considering the first 3 classes with the 72 imitations, instead of the first 3 classes with the 47 imitations of the 7 clusters. The answer is rather positive even if there are 7 errors of classification (see Figure 4). These errors are in part due to the fact that 2 of the 3 classes have imitations with a repetitive pattern, whereas only $\mathcal{R}$ has such imitations within the 7 clusters considered. Thus MA is not sufficient anymore to discriminate between 2 of the 3 classes.

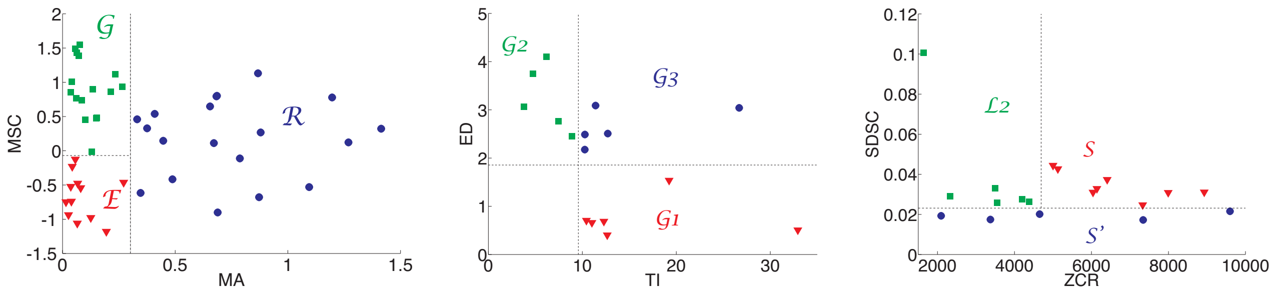


Figure 3. Discrimination between (1) $\mathcal{G}$, $\mathcal{E}$, $\mathcal{R}$, (2) $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3 \subset \mathcal{G}$, and (3) $\mathcal{L}_2$, $\mathcal{S}$, $\mathcal{S}' \subset \mathcal{R}$ (from left to right) with binary decision rules and a few relevant acoustic descriptors.

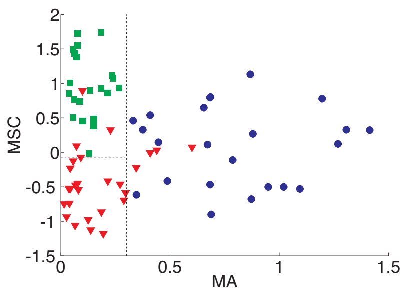


Figure 4. Generalization of the binary decision rules for the discrimination between $\mathcal{G}$, $\mathcal{E}$ and $\mathcal{R}$, to the discrimination between the first 3 classes with the 72 vocal imitations.

5.2 Second level of the hierarchy

We then focused on $\mathcal{G}$ and explained the 3 subclasses $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_3$ with 2 descriptors: (TI) the temporal increase of the energy envelope, and (ED) the effective duration of the energy envelope, because we remarked that the sounds in $\mathcal{G}_1$ are short with a brutal attack, the sounds in $\mathcal{G}_2$ are long with a smooth attack, and the sounds in $\mathcal{G}_3$ are long with a brutal attack. The classes are perfectly discriminated (see Figure 3) with the following rules:

(1) $\mathcal{G}_1:$ ED < 1.85578;

(2) $\mathcal{G}_2:$ ED ≥ 1.85578 and TI < 9.57589;

(3) $\mathcal{G}_3:$ ED ≥ 1.85578 and TI ≥ 9.57589.

We also focused on $\mathcal{R}$ and explained the 3 subclasses $\mathcal{L}_2$, $\mathcal{S}$ and $\mathcal{S}'$ with 2 descriptors: (SDSC) the loudness weighted standard deviation of the perceptual spectral centroid to discriminate between the sounds with a varying timbre in $\mathcal{L}_2$ and $\mathcal{S}$ and the sounds with a constant timbre in $\mathcal{S}'$, and (ZCR) the loudness weighted mean of the zero-crossing rate to discriminate between the sounds with a quite low pitch in $\mathcal{L}_2$ and the sounds with a higher pitch and some kind of noise and roughness in $\mathcal{S}$. The classes are perfectly discriminated (see Figure 3) with the following rules:

(1) $\mathcal{L}_2:$ SDSC ≥ 0.0231424 and ZCR < 4692.32;

(2) $\mathcal{S}:$ SDSC ≥ 0.0231424 and ZCR ≥ 4692.32;

(3) $\mathcal{S}':$ SDSC < 0.0231424.

6 DISCUSSION AND PERSPECTIVES

This paper reports on the analysis of a free classification experiment with vocal imitations of environmental sounds. The descriptions provided by the participants suggest that they used different kinds of similarities to group together the imitations: causal, semantic, acoustic, hedonic, types of vocal production. We have therefore proposed a method to assess the individual strategies. Using the R_V coefficient, we computed a measure of pairwise similarity between the participants. Although we detected an outlier, we were not able to highlight different strategies. This may be due to the method, or to the fact that the different kinds of similarities might actually overlap. This method must therefore be tested on synthetic data and other results from classification experiments to assess its robustness and reliability. Other measures of pairwise similarity could alternatively be used.

The participants' classifications were submitted to a hierarchical clustering analysis. We created clusters using the inconsistency coefficient, instead of the height of fusion. We chose a relevant threshold of inconsistency and created 7 clusters, which seemed interesting for finding acoustic invariants involved in the recognition of the imitated sources. However, a more systematic method to select the threshold of inconsistency may be preferred. A potential technique based on bootstrap is currently being developed.

It was finally possible to predict the 7 clusters by using binary decision rules with a few acoustic descriptors. With only 6 relevant descriptors, we discriminated the clusters perfectly. This suggests that the acoustic similarities overlap with the similarities used by the participants to perform the classification. Several issues need to be considered to extend these results to the imitated sounds. We worked with non-onomatopoeic imitations so as to emphasize their acoustic properties, and we chose clusters with respect to their relative quality. But we should now assess the quality of the imitations and their symbolic aspect, to ensure that the acoustic invariants found in the imitations can be generalized to the real sounds. Indeed, the imitations may allow the discrimination between the perceptual categories but not the recognition of these classes. Further experiments are required to address this issue.

7 ACKNOWLEDGMENTS

This work was funded by the EU project *CLOSED* FP6-NEST-PATH no. 29085.

8 REFERENCES

[1] H. Abdi, D. Valentine, S. Chollet, and C. Chrea. Analyzing assessors and products in sorting tasks: DISTATIS, theory and applications. *Food quality and preference*, 18(4):627–640, 2002.

[2] A. Akiyama. Analysis of onomatopoeia in French. Undergraduate thesis, Tokyo University of Foreign Studies, Tokyo, Japan, 2001.

[3] K. Aura. Imitation et catégorisation des sons dans le développement normal [Sound imitation and categorization in normal development]. Master's thesis, Université de Toulouse le Mirail, Toulouse, France, 2007.

[4] R. A. W. Bladon. Approaching onomatopoeia. *Archivum Linguisticum Leeds*, 8(2):158–166, 1977.

[5] Y. Escoufier. Le traitement des variables vectorielles [The treatment of vector variables]. *Biometrics*, 29:751–760, 1973.

[6] W. W. Gaver. How do we hear in the world? Explorations in ecological acoustics. *Ecological Psychology*, 5(4):285–313, 1993.

[7] W. W. Gaver. What in the world do we hear? An ecological approach to auditory event perception. *Ecological Psychology*, 5(1):1–29, 1993.

[8] K. Hiyane, N. Sawabe, and J. Iio. Impulse sound recognition system based on onomatopoeia. In *Proceedings of Acoustical Society of Japan*, pages 135–136, 1998.

[9] J. Iio and K. Hiyane. Onomatopoeia cluster for non-speech recognition. In *Proceedings of IEICE*, 1999.

[10] S. Iwamiya and M. Nakagawa. Classification of audio signals using onomatopoeia. *Journal of Soundscape Association of Japan*, 2000.

[11] N. J. Laas, S. K. Eastham, T. L. Wright, A. R. Hinzman, K. J. Mills, and A. L. Hefferin. Listeners' identification of human-imitated animal sounds. *Perceptual and Motor Skills*, 57:995–998, 1983.

[12] N. J. Laas, A. R. Hinzman, S. K. Eastham, T. L. Wright, K. J. Mills, B. S. Bartlett, and P. A. Summers. Listeners' discrimination of real and human-imitated animal sounds. *Perceptual and Motor Skills*, 58(2):453–454, April 1984.

[13] G. Lemaitre, O. Houix, N. Misdariis, and P. Susini. Listener expertise and sound identification influence the categorization of environmental sounds. Submission to *Journal of Experimental Psychology: Applied*, 2009.

[14] M. Ono, T. Sato, and K. Tanaka. Basic study on applying onomatopoeia to evaluating strange sound (Second report: Study on uttered sound of onomatopoeia). *Symposium on Evaluation and Diagnosis*, 3:29–32, 2004.

[15] A. D. Patel and J. R. Iversen. Acoustic and perceptual comparison of speech and drum sounds in the north indian tabla tradition: an empirical study of sound symbolism. In *Proceedings of the 15th International Congress of Phonetic Sciences*, Barcelona, August 2003.

[16] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical report, IRCAM, 2004.

[17] V. Rioux. *Sound quality of flue organ pipes. An interdisciplinary study on the art of voicing*. PhD thesis, Chalmers University of Technology, Sweden, 2001.

[18] D. Rocchesso, R. Bresin, and M. Fernström. Sounding objects. *IEEE Multimedia*, 10(2):42–52, 2003.

[19] M. Takada, K. Tanaka, S. Iwamiya, K. Kawahara, A. Takanashi, and A. Mori. Onomatopoeic features of sounds emitted from laser printers and copy machines and their contribution to product image. *Journal of INCE/J*, 26:264–272, 2002.

[20] K. Tanaka. Study of onomatopoeia expressing strange sounds (case if impulse sounds and beat sounds). *Transactions of the Japan Society of Mechanical Engineers*, 1995.

[21] H. Wissemann. *Untersuchungen zur Onomatopoiie [Study on onomatopoeia]*. Winter Verlag, 1954.

[22] R. Zuchowski. Stops and other sound-symbolic devices expressing the relative length of referent sounds in onomatopoeia. *Studia Anglica Posnaniensia*, 33:475–485, 1998.

A STRATIFIED APPROACH FOR SOUND SPATIALIZATION

Nils Peters[a], Trond Lossius[b], Jan Schacher[c], Pascal Baltazar[d], Charles Bascou[e], Timothy Place[f]
[a] CIRMMT, McGill University, Montréal, nils.peters@mcgill.ca
[b] BEK - Bergen Center for Electronic Arts, trond.lossius@bek.no
[c] ICST, Zurich University of the Arts, jan.schacher@zhdk.ch
[d] GMEA - National Centre for Musical Creation of Albi, pb@gmea.net
[e] GMEM - National Centre for Musical Creation of Marseille, charles.bascou@gmem.org
[f] Cycling '74, tim@cycling74.com

ABSTRACT

We propose a multi-layer structure to mediate essential components in sound spatialization. This approach will facilitate artistic work with spatialization systems, a process which currently lacks structure, flexibility, and interoperability.

1 INTRODUCTION

The improvements in computer and audio equipment in recent years make it possible to experiment more freely with resource-demanding sound synthesis techniques such as spatial sound synthesis, also known as spatialization. For seeking new means of expression, different spatialization applications should be readily combined and accessible for both programmatic and user interfaces. Furthermore, quantitative studies on spatial music (e.g. [12]) remind us that there are great individual and context-related differences in the compositional use of spatialization and that there is no one spatialization system that could satisfy every artist. In an interactive art installation, the real-time quality of a spatial rendering system in combination with the possibility to control spatial processes through a multi-touch screen can be of great importance. In contrast, the paramount features in a performance of a fixed-media composition may be multichannel playback and the compensation of non-equidistant loudspeakers (in terms of sound pressure and time delays). Additional scenarios may require binaural rendering for headphone listening, multichannel recording, up and down mixing, or a visual representation of a sound scene. Moreover, even during the creation of one spatial art work, the importance of these requirements may change throughout different stages of the creative processes.

Guaranteeing efficient workflow for sound spatialization requires structure, flexibility, and interoperability across all involved components. As reviewed in the following section, common spatialization systems too often give no consideration to these requirements.

SMC 2009, July 23-25, Porto, Portugal

2 REVIEW OF CURRENT PARADIGMS

2.1 Digital Audio Workstations - DAW

Many composers and sound designers use DAWs for designing their sound spatialization primarily in the context of fixed media, tape-music, and consumer media production. A number of DAWs are mature and offer a systematic user interface, good project and sound file management, and extendability through plug-ins to fulfill different needs.

DAWs mainly work with common consumer channel configurations; mono, stereo and 5.1. However, through focusing on consumer media products, multichannel capabilities are limited. ITU 5.1 [6], a surround sound format with equidistant loudspeakers around an ideal located listener, is the most common multichannel format. Its artistic use may be limited because 5.1 favors the frontal direction and has reduced capabilities for localizing virtual sources from the sides and back. Recent extensions up to 10.2 are available [1], but are insufficient for emerging reproduction techniques such as Wave Field Synthesis or Higher Order Ambisonics. Also, in art installations or concert hall environments, non-standard loudspeaker setups are common due to artistic or practical reasons, varying in number and arrangements of loudspeakers. These configurations are typically unaccounted in DAWs and therefore often difficult to use.

DAW surround panners often comprise a parameter named *blur*, *divergence*, or *spread* that controls the apparent source width through modifying the distributed sound energy among loudspeakers. Although this parameter enriches the creative possibilities, it is often either missing or only indirectly accessible, e.g. through changing the distance of the sound source.

[1] A comparison of DAWs concerning their multichannel audio capabilities can be seen on `http://acousmodules.free.fr/hosts.htm`.

2.2 Media programming environments

Various media programming environments exist that are capable of spatial sound synthesis, e.g. SuperCollider, Pure Data, OpenMusic, and Max/MSP. In order to support individual approaches and to meet the specific needs of computer music and mixed media art, these environments enable the user to combine music making with computer programming. While aspiring to complete flexibility, they end up lacking structured solutions for the specific requirements of spatial music as outlined in section 1. Consequently, numerous self-contained spatialization libraries and toolboxes have been created by artists and researchers to generate virtual sound sources and artificial spaces, such as Space Unit Generator [26], Spatialisateur [7], or ViMiC [3]. Also toolboxes dedicated to sound diffusion practice has been developed, e.g. BEASTmulch System [2], ICAST [1]. Each tool, however, may only provide solutions for a subset of compositional viewpoints. The development of new aesthetics through combining these tools is difficult or limited due to their specific designs.

2.3 Stand-alone Applications

A variety of powerful stand-alone spatialization systems are in development, ranging from directional based spatialization frameworks, e.g. SSR [4], Zirkonium [19], and Auditory Virtual Environments (AVE), e.g. tinyAVE [2] to sound diffusion and particle oriented approaches, e.g. Scatter [9]. Although these applications usually promote their graphical user interfaces as the primary method to access their embedded DSP-algorithms, a few strategies to allow communication from outside through self-contained XML, MIDI or OSC [25] protocols can be found.

3 A STRATIFIED APPROACH TO THE SPATIALIZATION WORKFLOW

When dealing with spatialization in electroacoustic composition or linear sound editing, the workflow comprises a number of steps in order to construct, shape and realize the spatial qualities of the work. The creative workflow might appear to be different when working on audio installations or interactive/multimedia work. Still, we identified underlying common elements when spatialization is used. For this reason a stratified approach, where the required processes are organized according to levels of abstraction is proposed.

This model is inspired by the Open Systems Interconnection network model (OSI) [3], which is an abstract description for layered communications and computer network protocol design. OSI divides network architecture into seven layers that range from top to bottom between the Application and Physical Layers. Each OSI-layer contains a collection of conceptually similar functionalities that provide services to the layer above it and receives services from the layer below it.

As depicted in Figure 1, six layers have been defined in our model.

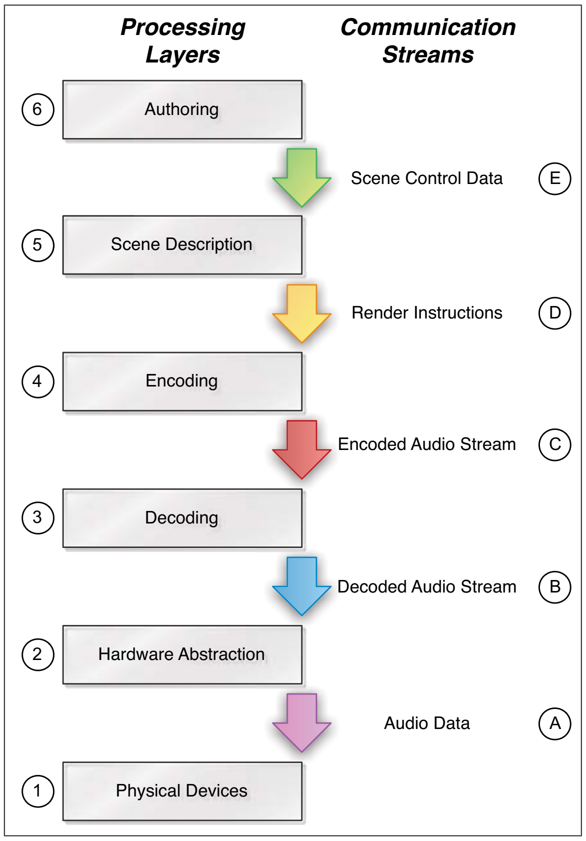


Figure 1. Layers and streams in sound spatialization

3.1 Physical Device Layer

The major functionality of this layer is to establish the acoustical connection between computer and listener. It defines the electrical and physical specifications of devices that create the acoustical signals, such as soundcards, amplifiers, loudspeakers, and headphones.

3.2 Hardware Abstraction Layer

This layer contains the audio services that run in the background of a computer OS and manages multichannel audio data between the physical devices and higher layers. Examples are Core Audio, ALSA, or PortAudio. Extensions such as JACK, Soundflower, Rewire and networked audio

[2] `http://www.beast.bham.ac.uk/research`
[3] `http://en.wikipedia.org/wiki/OSI_model`

streaming can be used for more complex distributions of audio signals among different audio clients.

3.3 Encoding and Decoding Layers

In the proposed model the spatial rendering is considered to consist of two layers. The Encoding Layer produces encoded signals containing spatial information while remaining independent of and unaware of the speaker layout. The Decoding Layer interprets the encoded signal and decodes it for the speaker layout at hand. According to [24, p. 99] this makes the creative process and the created piece more portable and future-proof because different speaker layouts can be used as long as a decoder is available. Examples of such hierarchical rendering methods are Ambisonics B-Format, Higher Order Ambisonics, DIRAC [18], MPEG Surround, AC-3, or DTS.

Not every rendering technique generates intermediate encoded signals, but instead can be considered to encapsulate the Encoding and Decoding Layers in one process. Some examples of such renderers are VBAP [16], DBAP [8], ViMiC [3] and Ambisonics equivalent panning [11]. Processing of sources to create an impression of distance, such as Doppler effect, gain attenuation and air absorption filters, are considered to belong to the Encoding Layer, as does the synthesis of early reflections and reverberation, i.e. as demonstrated by surround effects that employ B-format impulse responses convolution.

3.4 Scene Description Layer

This layer mediates between the Authoring Layer above and the Decoding Layer below through an abstract and independent description about the spatial scene. This description can range from a simple static scene with one virtual sound source up to complex dynamic audio scenes including multiple virtual spaces. This data could also be stored to recreate spatial scenes in a different context. Specific (lower-level) render instructions are communicated to the Encoding Layer beneath. Examples are ASDF [4], OpenAL [5] or SpatDIF [13].

3.5 Authoring Layer

This layer contains all software tools for the end-user to create spatial audio content without the need to directly control underlying processes. Although these tools may remarkably differ from each other through functionality and interface design to serve the requirements for varicolored approaches to spatialization, the communication to the Scene Description Layer must be standardized. Examples are symbolic authoring tools, generative algorithms, and simulations of emergent behaviors (swarms or flock-of-birds); or, more specifically as discussed below, Holo-Edit, and ambimonitor/ambicontrol.

3.6 Concluding remarks

OSI provided the idea that each layer has a particular role to play. The stratified model does not enforce one particular method for each layer; rather, a layer offers a collection of conceptually similar functions. This is analogue to how the TCP and UDP are alternative protocols working at the Transport Layer of the OSI model.

Spatialization processes should be modularized according to the layered model when feasible. With standardized communication to and from the layers, one method for a layer can easily be substituted for another, enhancing a flexible workflow that can rapidly adapt to varying practical situations and needs.

4 STRATIFIED TOOLS

Following, the authors discus several of their developments which strive to establish and evaluate the proposed stratified concept.

4.1 SpatDIF

The goal of the Spatial Sound Description Interchange Format (SpatDIF) is to develop a system-independent language for describing spatial audio [13] that can be applied around the Scene Description Layer to communicate between authoring tools down to the Encoding/Decoding Layers.

Formats that integrate spatial audio descriptors, such as MPEG-4 [23] or OpenAL, did not fully succeed in the music or fine arts community because they are primarily tailored to multimedia or gaming applications and don't necessarily consider the special requirements of spatial music, performances in concert venues, and site-specific media installations. To account for these specific requirements, the SpatDIF development is consequently a collaborative effort that jointly involves researchers and artists.

A database [4] has been created to gather information about syntax and functionalities of common spatialization tools and to identify the lowest common denominator, the "Auditory Spatial Gist", for describing spatialized sound. Beside these essential Core Descriptors, a number of extensions have been proposed to systematically account for enhanced features, e.g. the Directivity Extension, which deals with directivity information of a virtual sound source; the Acoustic Spaces Extension that contains acoustical properties of virtual rooms, or the Ambisonics Extension that handles ambisonics-only parameters. The latter is an example where SpatDIF mediates between the processing layers, starting from Layer 3 to Layer 6.

Although SpatDIF does not imply a specific communication protocol or storing format, at present, OSC for streaming and SDIF [22] as a storing solution are used for piloting.

[4] `http://redmine.spatdif.org/wiki/spatdif/SpatBASE`

4.2 ICST Ambisonics

The ICST Ambisonics Tools is a set of externals for Max/MSP [21]. The DSP externals ambiencode∼ and ambidecode∼ generate and decode Higher Order Ambisonics and are part of the Encoding and Decoding Layer.

Ambimonitor and ambicontrol complete the set as control tools for the Authoring Layer. Ambimonitor generates coordinate information for the DSP objects, presents the user with a GUI displaying point sources in an abstract 2D or 3D space and is equipped with various key commands, snapshot and file I/O capabilities. Ambicontrol provides a number of methods that control motion of points in the ambimonitor's dataset. Automated motions, such as rotation or random motion, optionally constrained in bounding volumes and user defined trajectories can be applied to single or grouped points. Trajectories and state snapshots can be imported/exported as an XML file, which will be replaced by a SpatDIF compliant formatting in a next release.

A novel panning algorithm [11] was derived from in-phase ambisonics decoding and implemented as a Max/MSP external entitled ambipanning∼. It encapsulates the Encoding and Decoding Layer by transcoding a set of mono sources in one process onto an ideally circular speaker setup with an arbitrary number of speakers. The algorithm works with a continuous order factor, permitting the use of individually varying directivity responses.

4.3 Jamoma

Jamoma[5] is a framework [14] for structuring and controlling modules in Max/MSP. Work on spatialization has been of strong interest to several of the developers, and solutions for spatialization in Jamoma have a stratified approach in accordance with the proposed model.

The Max/MSP signal processing chain only passes mono signals, and for multichannel spatial processing the patch has to be tailored to the number of sources and speakers. If Max/MSP is considered a programming environment and the patch is the program, a change in the number of sources or speakers requires a rewrite of the program, not just a change to one or more configuration parameters. Jamoma addresses this by introducing multichannel audio signals between modules with all channels wrapped onto a single patch cord. Jamoma Multicore[6] is being developed as a more robust solution than the current approach for handling multichannel signals which are also used between the Encoding, Decoding and Hardware Abstraction Layers.

Jamoma modules have been developed to convert multichannel signals, play and record multichannel sound files, perform level metering and pass multichannel signals on to the sound card or virtual auxiliary bus. These are supplemented by modules compensating for sound-pressure and time-delay differences in non-equidistant loudspeaker arrangements.

Ambisonics is the only spatialization method implemented in Jamoma that separates spatial encoding and decoding. 1st to 3rd order B-format encoding of mono sources is implemented using the ICST externals[21]. Other modules are available to encode recordings made with the Zoom H2 and to encode UHJ signals. Encoded signals can be manipulated, i.e. the balance between the encoded channels can be adjusted, or the encoded signal can be rotated, tilted and tumbled. The decoding module for up to 3rd order B-format signals uses the ICST externals while a module for binaural decoding uses Spatialisateur [7]. B-format signals can also be decoded to UHJ.

Several other popular spatialization algorithms are available as Jamoma modules: VBAP [17], ViMiC [3] and DBAP [8]. Consequently, one rendering technique can easily be substituted for another, or several rendering techniques might be used in tandem for a variety of spatial expressions, analogues to how an artist will use many different brushes in one artwork.

Prior to rendering, additional modules offer Doppler, air absorption and distance attenuation source pre-prosessing. All modules operating at the Encoding Layer are SpatDIF-compliant and hence provide the same interface to controlling modules operating at higher layers.

At the Scene Description Layer, a module provides a simple interface for defining the position of sources. The same module can be used to set loudspeaker positions for the Decoding Layer.

At present, two modules operate at the Authoring Layer; Boids simulation of co-ordinated animal motion and a scene manipulator allows geometric transformations (e.g. scaling, skewing, rotation) and stochastically driven manipulations of the whole scene in three dimensions. In addition, Jamoma can be bridged to Holo-Edit as discussed in the next section.

4.4 GMEM Holo-Edit

Initiated by L. Pottier [15], Holo-Edit is part of the GMEM Holophon project and conceptualized as an authoring tool for spatialization.

This standalone application uses the timeline paradigm found in traditional DAWs to record, edit, and play back control data. The data is manipulated in the form of trajectories or sequences of time-tagged points in a 3D space, and the trajectories can be generated or modified by a set of tools allowing specific spatial and temporal behaviors including symmetry, proportion, translation, acceleration, and local exaggeration. Different scene representation windows allow the user to modify data from different (compositional) viewpoints: *Room* shows a top view of the virtual space, the *Time Editor* shows the traditional DAW automation curve view

[5] `http://www.jamoma.org`
[6] `http://code.google.com/p/jamulticore/`

and, finally, the *Score Window* represents the whole composition in a multi-track block-based view. Holo-Edit's space and time representations are generic and can be adapted to any renderer at the Encoding Layer. To allow precise alignment of sound cues to desired spatial movements, waveform representations of sounds and associated trajectories are displayed and can be edited together.

Holo-Edit uses OSC for communicating with the desired spatial sound renderer. Here, the main challenge is to adapt and format the data stream that fits the specific rendering algorithm syntax (e.g. coordinate system, dimensions, units). To overcome this challenge, a Holo-Edit communication interface that handles sound file playback and position data of loudspeakers and sound sources through its standardized OSC-namespace was developed for the Jamoma environment. Therefore, Holo-Edit can be used as the main authoring tool for spatialization, while all DSP audio processes are executed in Jamoma (Figure 2). The communication between Holo-Edit and Jamoma is full-duplex, thus also enables the recording of trajectories in Holo-Edit from any real-time control interface addressable through Jamoma.

5 DISCUSSION & CONCLUSION

The examples from the previous section illustrate that a stratified model can be fruitful for development within media programming environments. The modular framework TANGA [20] for interactive audio applications reveals a related separation of tasks.

A few stand-alone applications are designed with a similar layered approach that allows control of different spatial rendering algorithms from one common interface, e.g. [4]. Artists and researchers would benefit greatly if all these "local solutions" could be accessed by any desired authoring tool and integrated into existing environments.

After an ICMC 2008 panel discussion on interchange formats for spatial audio scenes [7] and informal discussion showed that adequate spatialization tools for working in DAWs are missing, but strongly desired. The proposed stratified approach would be more flexible than the current DAW architecture where tools for spatialization are tied to a number of consumer channel configurations. The object oriented mixer approach proposed in [10] suggests that stratification can be employed in DAWs. A potential limitation might be imposed by the fact that automation in DAWs generally is represented as time-tagged streams of one-dimensional values while spatial information is generally multi-dimensional.

One keystone may be to define and agree on a meaningful communication format for spatialization. Therefore SpatDIF needs to be further developed which will culminates in an API that easily integrates in any spatialization software.

[7] http://redmine.spatdif.org/wiki/spatdif/Belfast_2008

6 ACKNOWLEDGMENT

The concepts proposed in this paper were to a large degree developed during a workshop at GMEA Centre National de Création Musicale as part of the Virage research platform funded by the French National Agency for Research. This work is also partly funded by the Canadian Natural Sciences and Engineering Research Council (NSERC), the Canada Council for the Arts, The COST IC0601 Action on Sonic Interaction Design (SID) and the Municipality of Bergen.

7 REFERENCES

[1] S. D. Beck, J. Patrick, B. Willkie, and K. Malveaux. The Immersive Computer-controlled Audio Sound Theater: Experiments in multi-mode sound diffusion systems for electroacoustic music performance. In *Proceedings of International Computer Music Conference 2006*, New Orleans, US, 2006.

[2] C. Borß and R. Martin. An improved parametric model for perception-based design of virtual acoustics. In *AES 35th Int. Conference*, London, UK, 2009.

[3] J. Braasch, N. Peters, and D. L. Valente. A loudspeaker-based projection technique for spatial music applications using virtual microphone control. *Computer Music Journal*, 32(3):55 – 71, 2008.

[4] M. Geier, J. Ahrens, and S. Spors. The SoundScape Renderer: A Unified Spatial Audio Reproduction Framework for Arbitrary Rendering Methods. In *124th AES Convention, Preprint 7330*, Amsterdam, The Netherlands, May 2008.

[5] G. Hiebert. *OpenAL 1.1 Specification and Reference*, 2005.

[6] ITU. Recommendation BS. 775: Multi-channel stereophonic sound system with or without accompanying picture, International Telecommunications Union, 1993.

[7] J.-M. Jot. *Etude et Réalisation d'un Spatialisateur de Sons par Modèles Physiques et Perceptifs*. PhD thesis, France Telecom, Paris 92 E 019, 1992.

[8] T. Lossius, P. Baltazar, and T. de la Hogue. DBAP - Distance-Based Amplitude Panning. In *Proceedings of 2009 International Computer Music Conference*, Montreal, Canada, 2009.

[9] A. McLeran, C. Roads, B. L. Sturm, and J. J. Shynk. Granular sound spatialization using dictionary-based methods. In *Proceedings of the 5th Sound and Music Computing Conference*, Berlin, Germany, 2008.

[10] S. Meltzer, L. Altmann, A. Gräfe, and J.-O. Fischer. An object oriented mixing approach for the design of spatial audio scenes. In *Proc. of the 25th Tonmeistertagung*, Leipzig, Germany, 2008.

[11] M. Neukom and J. Schacher. Ambisonics equivalent panning. In *Proceedings of the 2008 International Computer Music Conference*, Belfast, UK, 2008.

[12] F. Otondo. Contemporary trends in the use of space in electroacoustic music. *Organised Sound*, 13(01):77–81, 2008.

[13] N. Peters. Proposing SpatDIF - the spatial sound description interchange format. In *Proceedings of the 2008 International Computer Music Conference*, Belfast, UK, 2008.

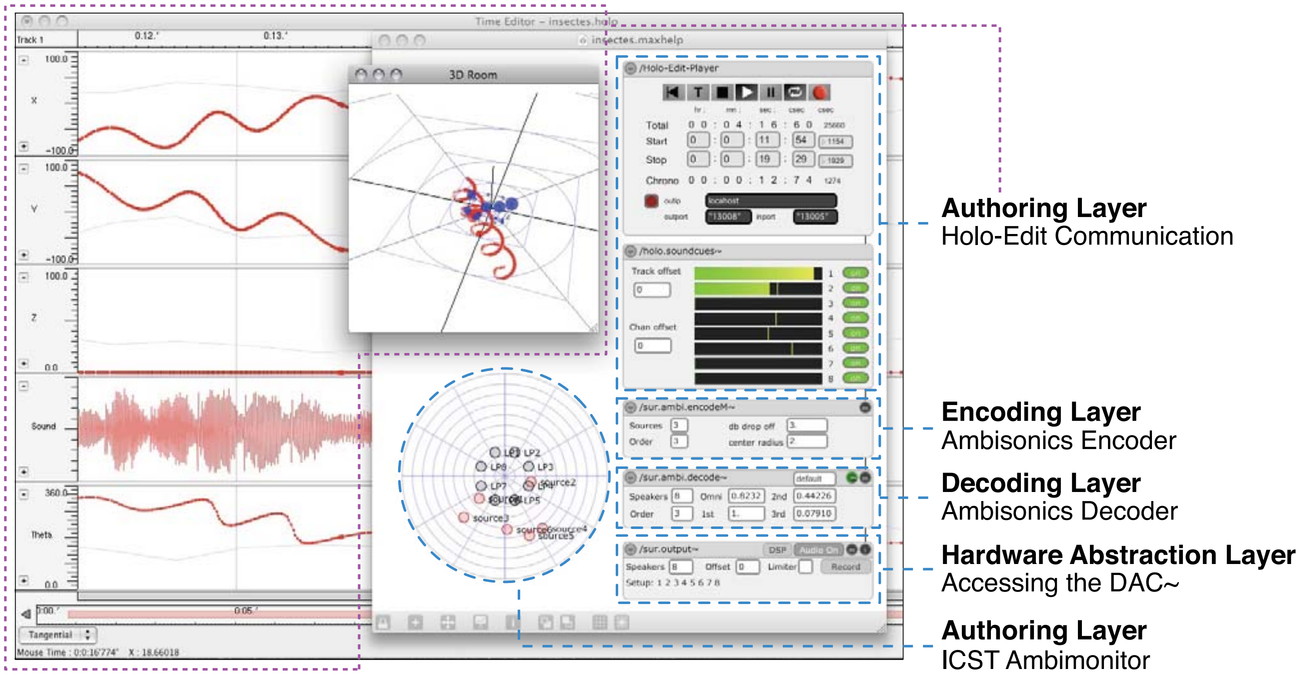


Figure 2. Holo-Edit, Jamoma and ICST Ambisonics Tools unified

[14] T. Place and T. Lossius. Jamoma: A modular standard for structuring patches in max. In *Proceedings of the 2006 International Computer Music Conference*, New Orleans, US, 2006.

[15] L. Pottier. Dynamical spatialisation of sound. holophon: a graphical and algorithmical editor for sigma 1. In *Proceedings of the International Conference on Digital Audio Effects, DAFX98*, Barcelona, Spain, 1998.

[16] V. Pulkki. Virtual sound source positioning using vector base amplitude panning. *J. Audio Eng. Soc.*, 45(6):456–466, 1997.

[17] V. Pulkki. Generic panning tools for MAX/MSP. In *Proceedings of 2000 International Computer Music Conference*, pages 304–307, Berlin, Germany, 2000.

[18] V. Pulkki. Spatial sound reproduction with directional audio coding. *J. Audio Eng. Soc.*, 55(6):503–516, June 2007.

[19] C. Ramakrishnan, J. Goßmann, and L. Brümmer. The ZKM Klangdom. In *Proc. of the 2006 conference on New Interfaces for Musical Expression*, pages 140–143, Paris, France, 2006.

[20] U. Reiter. TANGA - an interactive object-based real time audio engine. In *Proceedings of the 2nd Audio Mostly conference*, Ilmenau, Germany, 2007.

[21] J. C. Schacher and P. Kocher. Ambisonics Spatialization Tools for Max/MSP. In *Proc. of the 2006 International Computer Music Conference*, pages 274–277, New Orleans, US, 2006.

[22] D. Schwarz and M. Wright. Extensions and Applications of the SDIF Sound Description Interchange Format. In *Proceedings of the 2000 International Computer Music Conference*, pages 481–484, Berlin, Germany, 2000.

[23] R. Vaananen and J. Huopaniemi. Advanced AudioBIFS: virtual acoustics modeling in MPEG-4 scene description. *Multimedia, IEEE Transactions on*, 6(5):661–675, 2004.

[24] B. Wiggins. *An investigation into the real-time manipulation and control of three-dimensional sound fields*. PhD thesis, University of Derby, Derby, UK., 2004.

[25] M. Wright and A. Freed. Open Sound Control: A New Protocol for Communicating with Sound Synthesizers. In *Proceedings of the 1997 International Computer Music Conference*, pages 101–104, Thessaloniki, Greece, 1997.

[26] S. Yadegari, F. R. Moore, H. Castle, A. Burr, and T. Apel. Real-time implementation of a general model for spatial processing of sounds. In *Proceedings of the 2002 International Computer Music Conference*, pages 244–247, Goteborg, Sweden, 2002.

A CLASSIFICATION APPROACH TO MULTIPITCH ANALYSIS

Anssi Klapuri
Department of Signal Processing, Tampere University of Technology
klap@cs.tut.fi

ABSTRACT

This paper proposes a pattern classification approach to detecting the pitches of multiple simultaneous sounds. In order to deal with the octave ambiguity in pitch estimation, a statistical classifier is trained which observes the value of a detection function both at the position of a candidate pitch period and at its integer multiples and submultiples, in order to decide whether the candidate period should be accepted or rejected. The method improved significantly over a reference method in simulations.

1 INTRODUCTION

A fundamental problem of basically all pitch detection functions (such as the autocorrelation function) is that they do not show a peak only at the position of the true pitch, but also at twice and half the correct pitch, and often at all multiples and submultiples of it. This ambiguity is particularly challenging in multipitch detection where the detection function easily becomes congested with spurious peaks due to the ambiguity associated with each component sound.

To tackle the problem, multipitch estimation methods typically search for a set of pitch frequencies that best explain all the peaks in the detection function. Both joint estimation of multiple pitches and iterative detection and cancellation have been proposed (see [1, 2] for a review). A limitation of many of these techniques is that they produce a discrete set of detected pitch values, not a continuous function which would show the likelihoods of all pitch candidates within a given range. The latter would be more desirable for feature extraction purposes, where the actual detection stage is postponed to processes that look at a larger time scale and may include musicological constraints.

In this paper, we investigate a classification approach to pitch analysis. This approach has been previously investigated by Ellis and Poliner in [3], but they considered the multipitch analysis for a specific instrument (piano) and the applied technique was different from the present one.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

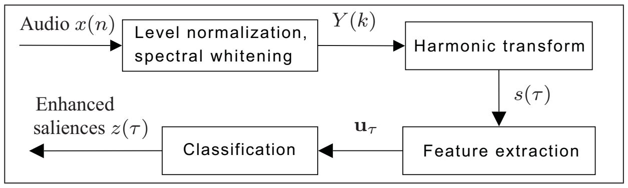


Figure 1. Overview of the method. See text for details.

2 METHOD

Figure 1 shows an overview of the proposed method. The first two steps, spectral whitening and harmonic transform, can be seen as preprocessing to distill information that is relevant for pitch detection. The steps are similar to the front-end used in [4] and produce a pitch salience function $s(\tau)$ where peaks indicate potential pitch periods in the input.

The latter two steps, feature extraction and classification, constitute the core of the method proposed here. They produce an enhanced salience function $z(\tau)$, where the peaks that correspond to correct periods are emphasized and extraneous ones are suppressed.

2.1 Level normalization and spectral whitening

The input audio signal $x(n)$ is blocked into 93 ms analysis frames that are processed independently. The signal within each frame is Hamming windowed, level-normalized to unity variance, zero-padded to twice its length, and then discrete Fourier transformed to obtain spectrum $X(k)$.

Spectral whitening, or flattening, is applied on $X(k)$ in order to suppress timbral information and thereby make the subsequent pitch analysis more robust to various sound sources. This is achieved by calculating power σ_c^2 of the signal within critical-band subbands c and by scaling the signal within each band by $\gamma_c = \sigma_c^{\nu-1}$, where $\nu = 0.16$ is a parameter determining the amount of whitening. The resulting whitened magnitude spectrum is denoted by $Y(k)$.

2.2 Harmonic transform

A harmonic transform is applied on the spectrum $Y(k)$ in order to calculate the saliences $s(\tau)$ of pitch period candidates τ:

$$s(\tau) = \sum_{h=1}^{H} g(\tau, h) \max_{k \in \kappa_{\tau,h}} Y(k), \tag{1}$$

where the set $\kappa_{\tau,h}$ defines a range of frequency bins in the vicinity of the h:th overtone partial of the pitch candidate f_s/τ (f_s denoting the sampling rate) and $H = 20$. More exactly, $\kappa_{\tau,h} = \{\lfloor hK/(\tau + \Delta\tau/2)\rceil, \ldots, \lfloor hK/(\tau - \Delta\tau/2)\rceil\}$, where $\lfloor\cdot\rceil$ denotes rounding to the nearest integer, K is the length of the Fourier transform, and $\Delta\tau = 0.5$ denotes the spacing between successive period candidates τ.

The weights $g(\tau, h)$ are defined after [4] and are of the form $g(\tau, h) = (f_\mathrm{s}/\tau + \epsilon_1)/(hf_\mathrm{s}/\tau + \epsilon_2)$, where $\epsilon_1 = 52$ Hz and $\epsilon_2 = 320$ Hz. Note that the weights reduce to $1/h$ if the moderation terms ϵ_1 and ϵ_2 are omitted.

2.3 Feature extraction

Peaks in the salience function $s(\tau)$ are useful for indicating potential fundamental frequencies in the input signal. However, a pitched sound in the input does not only produce a peak at the corresponding pitch period τ, but also at multiples and submultiples of τ, complicating the pitch detection.

In order to do the detection more robustly, we observe $s(\tau)$ at the candidate period τ, but also at its multiples and submultiples. Let us define a vector $\mathbf{a}_\tau$:

$$\mathbf{a}_\tau = [1, s(\tau), s(2\tau), \ldots, s(J\tau), s(\tau/2), s(\tau/3), \ldots, s(\tau/J)]^\mathsf{T}$$

where $J = 5$ is the maximum (sub)multiple of τ considered. The length of $\mathbf{a}_\tau$ is $2J$. We then form a feature vector

$$\mathbf{v}_\tau = \begin{bmatrix} b_0(\tau)\mathbf{a}_\tau \\ b_1(\tau)\mathbf{a}_\tau \\ \vdots \\ b_{M-1}(\tau)\mathbf{a}_\tau \end{bmatrix}$$

where $b_m(\tau) = [\log(\tau + 1)]^m$, $m = 1, \ldots, M$, are basis functions that depend on the period τ and allow the subsequent statistical model to treat short and long periods differently. The length of $\mathbf{v}_\tau$ is $2JM$.

From here on, we consider data from different analysis frames and use $\mathbf{v}_{i,\tau}$ to denote the feature vector corresponding to period candidate τ in frame i. For the purpose of training, we collect $\mathbf{v}_{i,\tau}$ corresponding to the true periods in each frame, plus those corresponding to the 20 next-highest "false" peaks in $s(\tau)$. The vectors $\mathbf{v}_{i,\tau}$ in different frames and for different τ are stored as columns in a large matrix $\mathbf{V}$. The matrix is then processed by removing the uppermost row which is $b_0(\tau) \cdot 1 \equiv 1$ at all columns. The rest of the rows are normalized to zero mean and unity variance. The resulting normalized matrix is denoted by $\mathbf{W}$. The columns of $\mathbf{W}$ correspond to individual feature vectors, $\mathbf{w}_{i,\tau}$.

Finally, a linear transform is employed to decorrelate the features and to reduce their dimensionality. We tested principal component analysis (PCA) and linear discriminant analysis (LDA) for this purpose. They both produce a transform matrix $\mathbf{A}$ of size $((2JM-1) \times D)$. The transformed feature vectors $\mathbf{u}_{i,\tau}$ with dimensionality D are obtained by

$$\mathbf{u}_{i,\tau} = \mathbf{A}^\mathsf{T}\mathbf{w}_{i,\tau}. \tag{2}$$

2.4 Classification

Gaussian mixture models (GMMs) are used to classify the peaks in $s(\tau)$ either as "true" or "false" pitch periods. A GMM is defined as

$$\mathsf{p}(\mathbf{u}_{i,\tau}|\theta) = \sum_{j=1}^{J} \beta_j \mathcal{N}(\mathbf{u}_{i,\tau}; \mu_j, \mathbf{\Sigma}_j), \tag{3}$$

where $\mathcal{N}(\mathbf{u}; \mu, \mathbf{\Sigma})$ denotes Gaussian distribution with mean μ and covariance $\mathbf{\Sigma}$. The shorthand $\theta = \{\beta_j, \mu_j, \mathbf{\Sigma}_j\}$ is used to refer to all the parameters of a GMM.

Two GMM models are trained, using the feature vectors corresponding to the "true" and "false" periods, respectively. The resulting model parameters are denoted by θ_T and θ_F, respectively.

Enhanced salience $z_i(\tau)$ of period candidate τ in frame i is then defined as

$$z_i(\tau) = \log \mathsf{p}(\mathbf{u}_{i,\tau}|\theta_\mathrm{T}) - \log \mathsf{p}(\mathbf{u}_{i,\tau}|\theta_\mathrm{F}). \tag{4}$$

The above formula calculates salience as the difference of the log-likelihoods for the two models. It is important to use the model for the false peaks as a "background" model in (4): including only the first term on the right-hand side of (4) would give a low salience for an exceptionally strong peak since it does not fit ideally to the model of true peaks. Calculating the salience as the difference between the two models corrects this problem, since these exceptionally strong cases are even less likely in the background model.

3 RESULTS

The proposed method was tested on mixtures of 1, 2, 4, and 6 simultaneous sounds, randomly mixing sounds from 32 different musical instruments. Half of the data was contaminated with random drum sounds using 0 dB SNR. The models were trained using 2600 sound mixtures and tested using a set of 1300 different mixtures. The results are averaged over the test cases.

Instrument samples were obtained from the McGill University Master Samples collection, the University of Iowa website, IRCAM Studio Online, and by making independent recordings for the acoustic guitar. Instruments represented are the piano, the guitar, mallet percussions (marimba, vibraphone), brass and reed instruments, strings, and flutes.

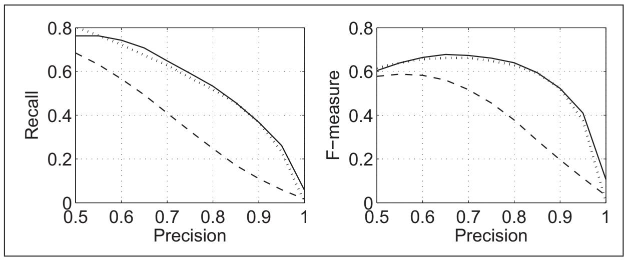


Figure 2. The left panel shows precision and recall for the proposed method with LDA (solid line), with PCA (dotted line) and for the baseline method (dashed line). The right panel shows F-measure as a function of precision.

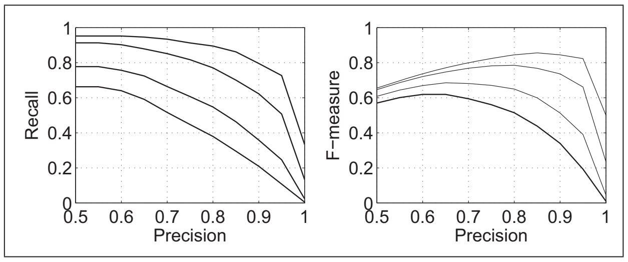


Figure 3. Recall and F-measure as a function of precision for the proposed method with LDA. The four curves from top to down correspond to polyphonies 1, 2, 4, and 6.

Figure 2 shows precision, recall, and F-measure for the proposed enhanced salience function $z(\tau)$ and, for comparison, for the raw salience function $s(\tau)$ used in [4] (here the iterative pitch detection and cancellation was not employed). The results were obtained by fixing a threshold value T, picking all the peaks above the threshold from all the frames, and then calculating the resulting precision π, recall ρ, and F-measure $\varphi = 2\pi\rho/(\pi + \rho)$. As can be seen, the proposed method improves significantly over the baseline method.

Figure 3 shows how the recall and F-measure behave in different polyphonies, varying the number of concurrent sounds from 1 to 6. The number of concurrent sounds in the mixtures was not given, but a single threshold value was again used (common to all polyphonies) and peaks above the threshold were picked from the enhanced salience function.

4 CONCLUSIONS

The proposed method for calculating pitch salience improved significantly over the baseline method in simulations. Furthermore, LDA reduces the feature vector dimensionality to one and does not require more than one Gaussian in the GMM. This means that the proposed method is computationally efficient and can be applied at all points of the raw salience function $s(\tau)$, not only at the positions of the main peaks. This is particularly useful for smooth pitch content visualization and feature extraction purposes.

5 REFERENCES

[1] C. Yeh, *Multiple fundamental frequency estimation of polyphonic recordings*, Ph.D. thesis, University of Paris VI, 2008.

[2] A. de Cheveigné, “Multiple F0 estimation,” in *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*, D. Wang and G. J. Brown, Eds. Wiley–IEEE Press, 2006.

[3] D. P. W. Ellis and G. Poliner, “Classification-based melody transcription,” *Machine Learning*, vol. 65, no. 2–3, pp. 439–456, 2006.

[4] A. Klapuri, “Multiple fundamental frequency estimation by summing harmonic amplitudes,” in *Intl. Conf. on Music Information Retrieval*, Victoria, Canada, 2006.

THE EFFECT OF VISUAL CUES ON MELODY SEGREGATION

Jeremy Marozeau
The Bionic Ear Institute,
Melbourne, Australia
jmarozeau@bionicear.org

David B. Grayden
University of Melbourne
Melbourne, Australia
grayden@unimelb.edu.au

Hamish Innes-Brown
The Bionic Ear Institute,
Melbourne, Australia
hinnes-brown@bionicear.org

Anthony N. Burkitt
University of Melbourne
Melbourne, Australia
aburkitt@unimelb.edu.au

ABSTRACT

Music often contains many different auditory streams and one of its great interests is the relationship between these streams (melody vs. counterpoint vs. harmony). As these different streams reach our ears at the same time, it is up to the auditory system to segregate them. Auditory stream segregation is based mainly on our ability to group different streams according to their overall auditory perceptual differences (such as pitch or timbre). People with impaired hearing have great difficulty separating auditory streams, including those in music. It has been suggested that attention can influence auditory streaming and, by extension, visual information. A psychoacoustics experiment was run on eight musically trained listeners to test whether visual cues could influence the segregation of a 4-note repeating melody from interleaved pseudo-random notes. Results showed that the amount of overall segregation was significantly improved when visual information related to the 4-note melody is provided. This result suggests that music perception for people with impaired hearing could be enhanced using appropriate visual information.

1. INTRODUCTION

While listening to music, listeners with normal hearing are generally able to hear the melody played by each instrument separately. This ability is commonly know as auditory streaming and refers to the process by which the human auditory system organises sounds from different sources into perceptually meaningful elements [1]. This ability is crucial to appreciate music, as its main interest relies on the relationship between voices (melody vs. counterpoint vs. harmony).

In a typical auditory streaming experiment [2], listeners are exposed to a sequence of alternating high and low notes – the sounds may be grouped together and perceived as coming from a single source (termed fusion) or perceived as streams from separate sources (termed fission). In the fusion case, the single stream is perceived as a "gallop". In the fission case, the sequence is perceived as two separate streams, or as a "Morse code" (Figure **1**).

Any salient perceptual changes, such as pitch and timbre [3], can influence auditory streaming [4]. If the same melody is played simultaneously by two different instruments, the perceptual segregation of the two auditory streams will be based mainly on their timbre differences. If the same instrument, such as a piano, plays two melodies simultaneously, the timbre of each melody is relatively similar [5]; therefore, the perceptual segregation is based mainly on the pitch difference between the two melodies.

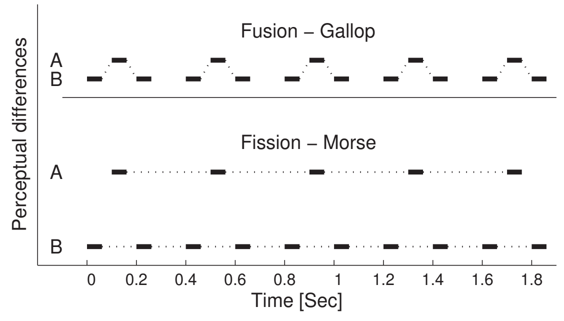


Figure 1. Illustration of typical auditory streaming experiment [Upper panel] The stimuli A are B perceptually close: the sequence is perceived as fused. [Lower panel] The stimuli A are B perceptually dissimilar: the sequence is perceived as two streams.

The mechanism of auditory streaming and the parts of the auditory pathway involved are still unclear. However, some evidence has shown peripheral cochlear [6] and central cortical components [2]. The "Peripheral Channelling" theory suggests that streaming depends primarily on the amount of overlap in the excitation pattern on the basilar membrane induced by the two stimuli, the more the two stimulus excitation patterns overlap, the more likely they are to be perceived as a single stream. On the other hand, streaming can be strongly affected by attention. Carlyon [2] showed that the minimum duration for two streams to be perceived as segregated depends on the amount of time the listener has paid attention to the sequence. As auditory attention can be affected by vision [7], this result suggests that a visual cue can have an influence on auditory streaming.

In normal hearing adults, coincident auditory and visual signals from the same object or event are combined in a process commonly known as multisensory integration. Multisensory integration can alter perception [8, 9] and facilitates information processing [10]. In adults with good hearing, stream segregation is also improved by multisensory integration. For example, studies have shown that the ability to segregate auditory speech from background noise is facilitated when visual cues (lip-movements) are available [11]. More recently, visual sequences have been shown to influence stream segregation of two ambiguous auditory sequences without linguistic components [12]. However, if incongruent information is presented to each sensory modality, the senses can also interfere with each other, causing altered or illusory percepts [8, 13].

Two hypotheses can be formulated:

1] A visual cue can enhance segregation of two streams by focusing the attention of the listener on one specific stream.

2] A visual cue will interfere with the auditory process and, therefore, weaken the segregation ability.

One of the most common complaints of people with hearing impairment, and especially cochlear implant recipients, is that music perception is very poor. A hearing impairment not only reduces the loudness of sounds, it also decreases the perceptual differences between sounds. This affects the ability of people with hearing impairment to segregate auditory streams and, therefore, weakens their appreciation of music. If hypothesis 1] is supported, this will indicate that visual support might be beneficial to improve music perception in people with hearing impairment.

The following experiment has been designed to extract baseline data for a study dedicated to better understanding auditory streaming ability in people with hearing impairment, and whether a visual cue may help them to improve their appreciation of music. The first step of this study is to test whether hypothesis 1] or 2] is supported for people with normal hearing and musical training.

2. EXPERIMENT

2.1. Methods

2.1.1. Stimuli

The auditory stimuli were all pure tones with a duration of 180 ms including 10 ms rise and fall raised cosine windows. The frequency of each tone ranged from midinote 45 (A2 or 110 Hz) to midinote 72 (C5 or 523.25 Hz). All tones were equalized in loudness at 70 phons according to the model of Moore [14]. A delay of 20 ms was introduced between each note. Stimuli were generated using Matlab 7.5 at a sampling rate of 44.1 kHz. The melody was composed of a repeating 4-note sequence (the target) interleaved with pseudo-random notes (the masker). The four notes of the target melody were middle C, F, D, and G (midinotes 60, 65, 62, 67). Figure **2(A)** shows the melody. Between each note of the target, a masker note was introduced, selected randomly within a range of 10 semi-tones. The note range of the masker varied across each block. Figure **2(B-D)** shows three possible sequences of notes depending on the note range of the masker. Sounds were presented through AKG K601 headphones at a comfortable level.

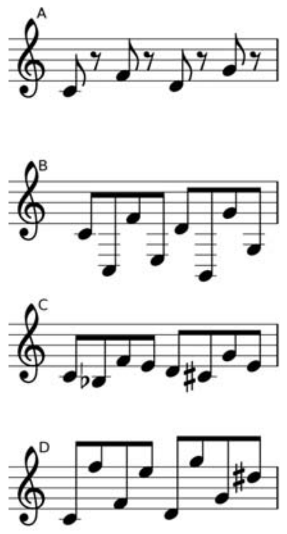
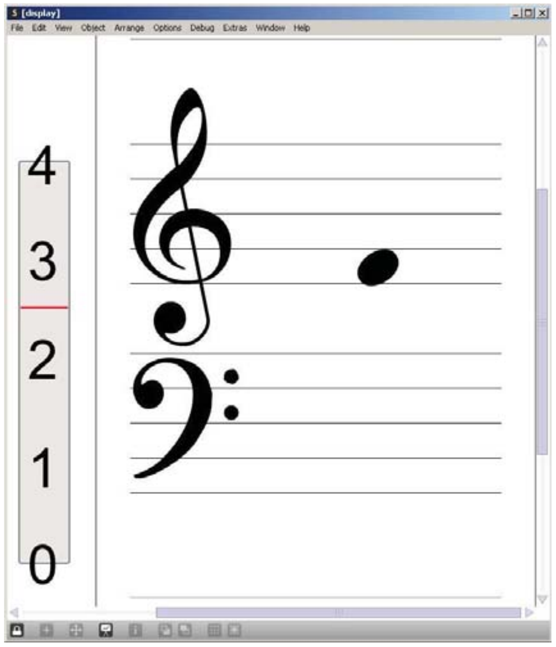


Figure 2. Musical sequence presented. The measure A] represents the target 4-note melody; B] shows a possible sequence when the note range of the masker is well below the target; C] a possible sequence when the note range of the masker overlaps the target; D] when the masker is well above the target.

Visual stimuli were presented in synchrony with the target and consisted of a piano staff and a quarter note. Each time a note from the target was played, it appeared as a quarter-note on the staff and disappeared when the note was turned off. The visual cues were presented on an LCD monitor and were generated through the software MAX/MSP 5.0. Figure **3** shows a screen shot of the visual cue.



[display]
File Edit View Object Arrange Options Debug Extras Window Help
4
3
2
1
0

Figure 3: Screen shot of the visual cue. On the left the rating scale and on the right the staff with note indicated.

2.1.2. Procedure

The experiment was divided into four blocks. As the experiment was designed to minimize the duration of testing, no training was provided. In order to reduce any possible bias due to training effects, the order of the four blocks was randomized across listeners. At the beginning of one block, the notes of the masker ranged between midinotes 45 (A2) to 54 (G3). After 12 seconds (60 notes), the note range of the masker increased by one semi-tone (from midinotes 46-55). The block stopped when the note range reached midinotes 72-81 (after 5.6 minutes). In another block, the masker notes range started at midinote 72-81, and then decreased at the same rate as the previous block, and stopped at midinotes 45-54 . These two blocks were each performed once with a visual cue, and once without. The overall experiment lasted less than half an hour.

The listeners were asked to focus their attention only on the target and to indicate their response by moving in real time the cursor of a midi controller. The response was a rating of how well the target melody was perceived. The cursor was graded from 0 to 4. A score of 4 indicated that the melody was easily perceived and a score of 0 indicated that the melody was not perceived at all. If the listeners perceived only half of the melody, or if they perceived the melody every other presentation, the listener was instructed to move the cursor to the position 2. Other positions of the cursor indicated any other intermediate percepts respectively. The midi controller was linked to a MAX/MSP patch that managed the experiment.

When a visual cue was presented, the listener was asked to keep their visual attention on the screen.

2.1.3. Listeners

Eight listeners participated, with ages ranging from 27 to 40 years of age. They all had some musical training and were able to read music. No one was paid for their participation.

2.2. Results

Figure **4** shows the results averaged across listeners and direction of the note range of the masker (from low to high register, or high to low). The data have been normalized to represent the probability of the fusion of the target and the masker into a single non-repeating melody (*i.e.* a score of 4 on the cursor was mapped to a value of 0 on the y-axis, and a score of 0 on the cursor to 1 on the y-axis). The x-axis represents the centre midinote range of the masker for any given range. The thin lines represent the raw data averaged over the eight listeners and the thick line a Gaussian-like fit:

$$Y_e = a * e^{-0.5*(\frac{b-N}{c})^2}, \qquad (1)$$

where Y_e is the estimation of the model, N is the midinote, and a, b, and c are the free parameters of the model. This model was fit separately to the data with and without visual cues. The parameter a indicates the peak value of the model on the normalized scale, the parameter b is the location of the peak, and the parameter c is the bandwidth of the model. In order to assess the streaming ability as a function of the note range overlap between the masker and the target, an indicator, P50, was derived from both models. It represents the note range where the model was above a probability of 50% of fusion.

The overall shape of the functions with and without visual cues indicates as expected, that when the masker note range is well separated from the target, the two streams are clearly segregated, and when they overlap, the two streams are fused.

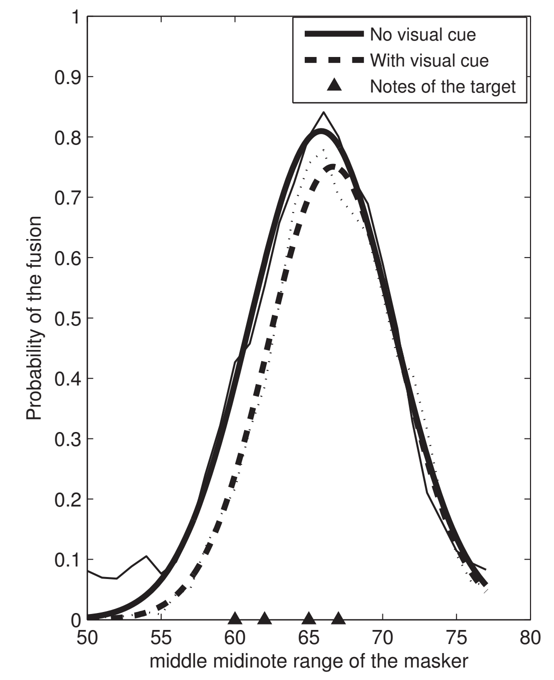


Figure 4: Average results (thin line) and Gaussian-like fit (thick line) of the results averaged across listeners and masker direction.

Student's t-tests were performed to compare the likelihood of fusion between the conditions (with and without visual cues). The analyses revealed that:

Parameter *a*]: The difference of amplitude observed in figure **4** between the conditions was not significant (t(7) = 1.17, p = 0.1397). This indicates that at the point of maximum difficulty, visual cues did not contribute significantly to auditory streaming.

Parameter *b*]: The average location of the peak was shifted to the right (higher note range) for the condition with visual cues. T-tests revealed a trend (t(7) = 1.53, p = 0.0850) and 7 out of 8 eight listeners showed a shifted peak. Thus, the visual cues showed a greater effect when the masker range was lower than the target range than when it was higher.

Parameter P50]: This parameter was highly significantly smaller for the condition with the visual cues (t(7) = 3.02, p = 0.0097), indicating that visual cue can allow a bigger note overlap between the target and the masker while still being perceived as streamed.

Overall, the results suggest that visual cues had a significant effect on auditory streaming.

3. DISCUSIONS

3.1. The effect of visual cues on auditory streaming

The results from this experiment clearly indicate that visual cues can have an effect on auditory streaming. Listeners, when presented with visual cues, could segregate a melody from an interleaved masker with a larger note range overlap than without visual cues.

This result strongly supports hypothesis 1] and refutes hypothesis 2]. This indicates that visual information might help people with impaired hearing to restore some of their appreciation of music.

3.2. The asymmetry of the effect.

The parameter *b* indicates that the effect of the visual cue was greater when the two lower notes were masked. Figure **4** shows no effect of the visual cues when the two higher note were masked. As the melody was tonal, and in a C key, the C note of the melody appeared clearly as the down beat. Listeners reported that if the C note was perceived, it was easier to extract the rest of the melody. The results suggest that the visual cues should emphasise the down beat. Further research is needed to validate this hypothesis.

3.3. The mechanism of auditory streaming

In conditions with and without visual cues, the auditory information was the same. Therefore, if streaming depends primarily on the excitation pattern on the basilar membrane, as suggested by the "Peripheral Channelling" theory, both conditions should lead to the same results. However, our data showed an effect on streaming of attention driven by visual cues. This result, therefore, supports the theory that streaming is at least partially supported by some cortical components.

3.4. The effect of musical training

All the listeners that participated had some musical training. It might be possible that the results could be different with people with normal hearing and no musical training. However, if this is the case, it implies that training could improve the weight of visual information in auditory segregation. Further testing is needed to validate this hypothesis.

4. ACKNOWLEDGMENT

The authors wish to thanks Ayla Barutchu and Prof. Peter Blamey for helpful comments on the early version of the paper and Aimee Clague for proofreading the document. Financial support was provided by the Jack Brockhoff Foundation, Goldman Sachs JBWere Foundation, Soma Health Pty Ltd, Mr Robert Albert AO RFD RD, Miss Betty Amsden OAM, Bruce Parncutt & Robin Campbell, and The Frederick and Winnifred Grassick Memorial Fund. The Bionic Ear Institute acknowledges the support it receives from the Victorian Government through its Operational Infrastructure Support Program.

REFERENCES

[1]. Bregman, A. S., *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA: The MIT Press, 1994.

[2]. Carlyon, R. P., "How the brain separates sounds". *Trends Cogn Sci*. 8: p. 465-71, 2004.

[3]. Wessel, D., "Timbre Space as a Musical Control Structure". *Comput Music J*. 3: p. 45-52, 1979.

[4]. Moore, B. C. and Gockel, H., "Factors Influencing Sequential Stream Segregation". *Act. Acustica*. 88: p. 320 – 332, 2002.

[5]. Marozeau, J., de Cheveigne, A., McAdams, S. and Winsberg, S., "The dependency of timbre on fundamental frequency". *J Acoust Soc Am*. 114: p. 2946-57, 2003.

[6]. Hartmann, W. M. and Johnson, D., "Stream segregation and peripheral channeling". *Music Percept*. 9: p. 155-184, 1991.

[7]. Driver, J. and Spence, C., "Crossmodal attention." *Curr Opin Neurobiol*. 8: p. 245-253, 1998.

[8]. McGurk, H. and MacDonald, J., "Hearing lips and seeing voices". *Nature*. 264: p. 746-8, 1976.

[9]. Barutchu, A., Crewther, G. S., Kiely, P., Murphy, M. J. and Crewther, D. P., "When /b/ill with /g/ill become /d/ill: Evidence for a lexical effect in audiovisual speech perception". *Eur J Cogn Psychol*. 20: p. 1-11, 2008.

[10]. Miller, J., "Divided attention: evidence for coactivation with redundant signals". *Cogn Psychol*. 14: p. 247-79, 1982.

[11]. Sumby, W. H. and Pollack, I., "Visual contribution to speech intelligibility in noise." *J Acoust Soc Am*. 26: p. 212-215, 1956.

[12]. Rahne, T., Bockmann, M., von Specht, H. and Sussman, E. S., "Visual cues can modulate integration and segregation of objects in auditory scene analysis". *Brain Res*. 1144: p. 127-35, 2007.

[13]. Shams, L., Kamitani, Y. and Shimojo, S., "Visual illusion induced by sound". *Brain Res Cogn Brain Res*. 14: p. 147-152, 2002.

[14]. American National Standard. "Procedure for the Computation of Loudness of Steady Sounds" *ANSI S3.4-2007*, 2007.

THE SONIFIED MUSIC STAND – AN INTERACTIVE SONIFICATION SYSTEM FOR MUSICIANS

Tobias Grosshauser
Ambient Intelligence Group
CITEC – Center of Excellence in Cognitive Interaction Technology
Bielefeld University, Bielefeld, Germany
tgrossha@techfak.uni-bielefeld.de

Thomas Hermann
Ambient Intelligence Group
CITEC – Center of Excellence in Cognitive Interaction Technology
Bielefeld University, Bielefeld, Germany
thermann@techfak.uni-bielefeld.de

ABSTRACT

This paper presents *the sonified music stand*, a novel interface for musicians that provides real-time feedback for professional musicians in an auditory form by means of interactive sonification. Sonifications convey information by using non-speech sound and are a promising means for musicians since they (a) leave the visual sense unoccupied, (b) address the sense of hearing which is already used and in this way further trained, (c) allow to relate feedback information in the same acoustic medium as the musical output, so that dependencies between action and reaction can be better understood. This paper presents a prototype system together with demonstrations of applications that support violinists during musical instrument learning. For that a pair of portable active loudspeaker has been designed for the music stand and a small motion sensor box has been developed to be attached to the bow, hand or hand wrist. The data are sonified in real-time according to different training objectives. We sketch several sonification ideas with sound examples and give a qualitative description of using the system.

1. INTRODUCTION

Musical instrument learning is a complex multi-modal real-time activity that involves processes from low-level coordinated motor control, auditory perception up to automation and complex cognitive processes such as understanding and learning. It is representative for the larger class of human activity where expression and behavior shape and develop during practice towards a specific goal, such as in dance and sports. Due to its richness and complexity, novices tend to allocate their attention on the closed-loop interaction so that they comply with a coarse level of control, for example to produce the accurate frequency or to generate the accurate rhythm, and this strong focus on primary objectives induces a neglecting of other important aspects such as a good body posture and alike that become relevant at later stages. Particularly, wrong coordination and posture can even cause physical problems for musicians and lead to a lot of effort to be relearned. Therefore techniques that can actively shift the player's focus of attention during practice in a early learning phase are highly motivated. As second aspect, the training of the hearing abilities and the reaction to acoustic events is one of the core abilities in learning musical instruments and singing. Sonification supports learning, by providing additional information in real-time acoustically and helps the students to use and train their ears with less visual distraction, compared to visual feedback.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

In this paper we present an approach that uses sonification[1], the non-speech auditory display of information as real-time feedback for the musician. Sonification, as described in [1] addresses our highly developed yet often neglected sense of listening. Indeed, compared to visual display, sound does not demand the user to visually attend a specific display location, sound is processed over a larger range of frequencies (typically the useful pitch range 50Hz to 5000Hz exceeds the visual 'one octave' from red to blue, and in the range from 0.1 to 10 Hz we perceive temporal structure and rhythm), and is highly capable to direct and alter the human's focus of attention. Furthermore, we are capable to attend to even subtle cues in complex sounds simultaneously and perceive the sound as a whole at the same time. Concerning coordinated rhythmical activity, by listening we are capable of discovering even faint changes in rhythm as well as coordination problems.
With this motivation, the idea is to measure the player's motor activity and to reflect specific properties of his/her

[1]See www.sonification.de/main-def.shtml for a definition.

performance as an task-specific and unobtrusive interactive sonification, so that on the one hand, the musician can still focus on the musical sounds but, without change of media, receives additional information to keep awareness on relevant aspects of the physical execution. To be successful, a careful selection of sensing technology, an appropriate and well-defined task, a suitable feature extraction process by means of data mining techniques and a sonification design which combines well with the musical signal need to come together, so that a system will be acceptable to musicians and helpful for long-term use.

The *sonified music stand* is our approach to integrate the essential technology into a tool that is typically in use anyway for musicians. It represents a first principled approach towards better closed-loop auditory interaction systems, here developed and optimized for a specific user group and application, but conceptually reaching beyond this case towards general sonification-based interaction support. The paper continues with a description of design aspects and a presentation of the technology. This is followed by a section on the selection of movements and requirements for the application of violin learning support. In turn, sensor data of bow strokes are shown and features are extracted. Section 5 presents our first interactive sonifications implemented in SuperCollider and projected via the sonified music stand. Finally we discuss our first experiences and our plans for continuation of this research.

2. THE SONIFIED MUSIC STAND – IDEA, OBJECTIVES AND DESIGN

The closed-loop audio feedback supports students in learning and performing situations, similar to the AcouMotion system in [2]. Complex movements are divided into several simple ones. Audio feedback supports the active learning phase by significant acoustic events for instruction in real-time. In this paper, scenarios of violin learning and teaching are shown, but other possibilities in the area of instrumental teaching in general and learning of movements, gestures and postures are possible.
A pair of self-made and developed three-cornered active stereo speakers is assembled on the left and right side of a conventional music stand. It is connected with the audio out of a laptop computer. The computer receives data from a 5 DOF sensor and generates audio out via SuperCollider. The sensor data are processed and several levels of difficulty allow user-adapted feedback, new experimental learning scenarios and a controlled learning effect. The evaluation process includes sensor-based motion capturing, evaluated on music instrument learning based scenarios and audio feedback.

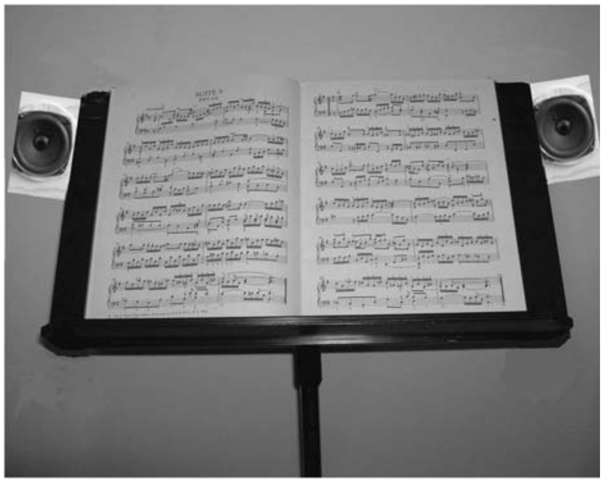


Figure 1. The sonified music stand

2.1. Sensor Hardware

Similar to the carbon K-Bow form Keith McMillen [3] and the used technologies from [4] and [5], acceleration and gyroscope sensor data were measured.
In our exemplary use cases, 5 degrees of freedom, acceleration sensors for x-, y-, and z-axis and 2 gyroscopes are analyzed. The data from the sensor are transmitted via radio frequency. A small Lithium polymer (Lipo) battery is directly attached for power supply. This small and light-weight, about 20 gr. sensor-module can be used as a standalone tool, just for movement learning or clipped to a bow of a string instrument.

2.1.1. The Gyroscope

A IDG-300 dual-axis angular rate sensor from InvenSense is used. This allows the measurement of the rotation of the x- and y-axis of the bow stroke (see fig. 2). The x-axis rotation is an additional compensating motion for e.g. soft bowing starts. The y-axis rotation is besides other functions relevant for pressure transfer onto the bow and to balance and change articulation and volume.

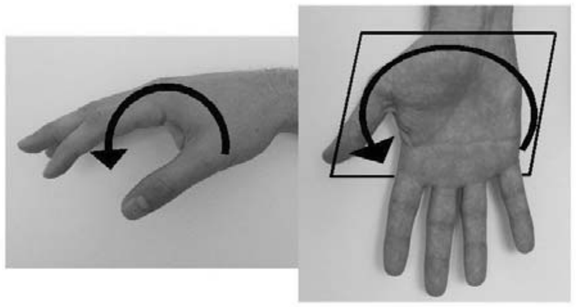


Figure 2. X-axis (left) and y-axis (right) rotation of the hand

2.1.2. The Accelerometer

The ADXL330 from InvenSense sensor is used, a small, thin, low power, complete x-, y-, and z-axis accelerometer. According to the description of the test cases in Sec. 3.1, every axis is important and has it's own defined plane, in which the movement is performed. Thinking in planes and rotations helps to learn complex movements, especially when the movement takes place beside your body and you the player hardly see it or control it visually.

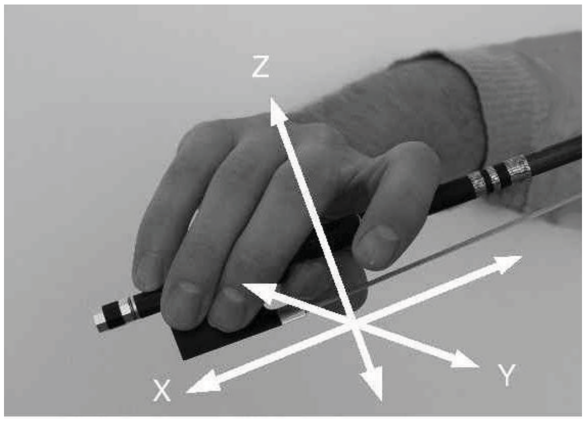


Figure 3. Motion coordinate system, x-, y-, and z-axis

2.2. Loudspeaker Design

For the sonified music stand, we developed a new design of an active loudspeaker, which offers many advantages:
the new developed active loudspeaker is easy to use and can be attached very flexible to manifold things and surfaces. The exceptional shape, the good sounding and energy-efficient amplifier allows many other uses. The introduced cases below will show simple sonification methods, where the loudspeaker can be used fixed to the music stand or stand-alone on a table or other surface.

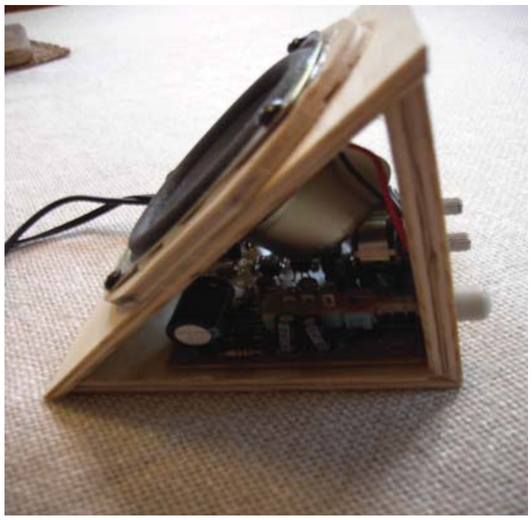


Figure 4. Loudspeaker without lateral housing

2.2.1. The Three-Cornered Design

The side view of the speaker shows the three-cornered shape. It allows an 45° angular, right angular and a parallel mounting of the loudspeakers. In the case of just putting them on a table, the sound 45° propagation is easy adaptable to the listener's ears for best audio quality results. In the other case of mounting them parallel to a screen or the music stand, the sound propagation is right-angled and in the direction of the user sitting or standing in front of a monitor or the music stand.
The three-cornered design diminishes standing waves compared to quadratic boxes because of the non-parallel walls. Furthermore, the robust construction of the cabinet reduces the transmission of the inter-casing wave propagation and vibration. These two advantages, besides the integrated amplifier, lead to good sound quality. The practical form-factor allows safe transportation of speaker pairs in cube-form putting them together membrane to membrane.

2.2.2. Amplification, Low Energy Consumption and Battery

The active loudspeaker design allows simple usage with all audio sources such as laptop computers and mp3 players. For amplification, a digital chip is used, which provides high energy-efficiency and good sound quality.

The mounting of the loudspeaker is done with clamps or hook-and-loop fasteners, depending on the surface and thickness of the music stand, monitors or other surfaces. Only a power and stereo cable is attached to the stereo amplifier housed in one loudspeaker, the other one is directly powered via one cable. This design allows the use of only one power cable for two speakers.

3. PERFORMANCE ASPECTS IN VIOLIN PLAYING

3.1. Methodical and systematical learning scenarios for complex motor skills

The following scenarios are basic extractions of beginners' violin lessons. Depending on the age of the pupil or student, different approaches exist. One of these is the breakdown and fragmentation of a movement into several simpler action units, based on the ideas of Conrad von der Goltz [6] . In our scenarios, a simple bow-stroke is decomposed in 4 cases, where the last one is the "recomposition" of the stroke. This is not only a beginner's problem, this is even trained from time to time by advanced students and professionals to develop their skills and physical awareness. The sensor and the real-time sonification gives us the possibility to train these simplified movements and adding step by step more and more complexity. In other words, this means the combination of simplified movements to more complex ones. The single and combined movements in the following cases can be performed simultaneously or successively, with or without instrument.

Case 1:
Problem: Movement of the hand in the x/y-plane (see fig. 4), with zero deviation in the z-axis.
Pedagogical aspect: Understanding the different planes of the bowing movement.
Idea: Drawing a line on a plane, for example on a virtual table. The pencil would draw a curve, accelerating in the x- and y-axis, but the plane surface of the table gives a constant zero-acceleration of the z-axis. The x-plane itself has different horizontal angles, but they don't change during one stroke. This is the so-called "string-plane", the elbow and upper arm don't change their height, just the forearm and hand move sidewards.
Result: The student gets an idea of the arm movement in one plane and the so-called "string-planes".

Case 2:
Problem: Adding a second plane, the y-plane with zero deviation of the y-axis to the exercise, drawing a virtual straight line.
Pedagogical aspect: Understanding the "virtual straight line" of bowing movement.
Idea: If you move your hand exactly along one direction so that you draw a perfect line into the air beside your body, complex compensating movements of the hands and arms are necessary. If you try this with a pupil the first time, it is not only hard to understand the movement without seeing your hands, also practicing in front of a mirror is difficult, because every change has to be side-inverted.
Result: Students learn to move the hand on defined straight lines, without looking to it.

Case 3:
Problem: Starting the movement of case 2 with a short acceleration phase, followed by constant speed without any acceleration and deceleration at the end of the movement.
Pedagogical aspect: The recognition of a constant low or high acceleration and constant speed of a movement in 3d-space is very difficult.
Idea:. Acceleration, the constant moving speed and deceleration in one single move influence the produced sound. These three aspects are realized, learned and trained in this exercise, supported by sonification.
Result: The student gets an idea of the 3 phases of a bow stroke and its' influence to sound generation, the acceleration phase, the constant bow speed phase and the deceleration phase.

Case 4:
Problem: Adding a defined rotation of some degrees at the beginning of the movement, as described in the cases before to the x-axis.
Pedagogical aspect: Learning and understanding the movement according to playing in reality with more or less bow-hairs influencing the attack and volume.
Idea: The data from the gyroscope and sonification allow an easier evaluation of this matter. Practicing in front of a mirror is difficult, because the posture and movement changes are very small and hard to see. The fine grain solution of the sensors allows detecting them and augmenting them for rendering sonified feedback.

3.2. Sonification for short- and long-term monitoring

Sonification allows short and long-term unobtrusive monitoring and feedback of many parameters to the musical instrument player in real-time. The above described cases 1 - 4 demonstrate a short-term observation. But long-term use allows the recognition of mistakes or symptoms of fatigue while exercising a completely other problem or playing a long piece en bloc. In these cases, attention is concentrated to other demands and sometimes even basic skills are neglected and cause a significant loss in sound or performance quality. The sonified music stand is easy to set up and on this account it can support students in every day practicing.

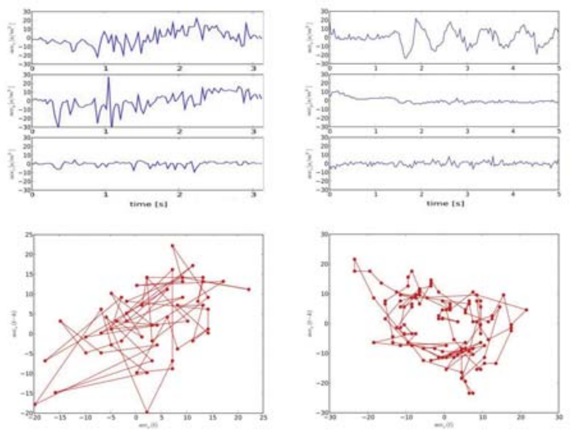


Figure 5. Left column incorrect bowing, right column correct bowing, time series of accelerations along the , x-, y-, z-axis and below Takens embedding, plotting accelerations against their delayed values.

4. FEATURE EXTRACTION FOR BOW MOVEMENT SONIFICATION

There are many sonification designs that use the raw measured sensor readings for mapping sonification. Sometimes (see for example [7] and [8]) also the audio signal of the instrument, played by the musician is used to render and calculate the audio-feedback and visualization with defined audio descriptors. In this paper, we use the sensor data for the sonifications, which allows a flexible set-up and learning scenarios with and without a musical instrument. However, we believe that sonification can profit a lot from the definition of task-oriented defined features which emphasize the movement structure of interest. For that we here present our first steps towards using data mining techniques as plug-in for mapping-based sonification. Since, however, the described cases are already quite straight-forward characterized by the cartesian axes of the sensors, a mixture of such raw sensors and derived features comes ideally to application. We depict the sensor readings and a Takens-embedding (plot of y(t) against y(t-k)) for the case 2 in fig. 5. It can be seen that the correct execution is a circle-shaped figure, so that the deviation of that circle radian as well as the phase are promising features to be used for parameter mapping sonification.

5. INTERACTIVE SONIFICATION DESIGN AND EXAMPLES

The sonifications are programmed in SuperCollider. For the first versions, the sensor data are mapped to different acoustic synthesis parameters and used to create acoustic events, as explained in the following. This 4 cases can all be exercised with and without instrument and bow and are at beginners' level. An example video can be seen at: http://www.sonification.de/publications/GrosshauserHermann2009-TSM/ . The first video shows 4 strokes of a bow, where the first two are executions within the x/y-plane, so that only the violin sound can be heard, in the following two strokes a salient noise sound can be heard that is the result of the deviation from the ideal movement. Obviously, the noisy sonification does not or only marginally interfere with the perception of the musical sound signal, yet it efficiently keeps the player's attention aligned to the task.

Concerning case 1, the frequency of a certain tone changes according to the deviation of the z-plane. If the deviation is bigger than a given gain, a second sound appears, according to the contact of another string in real life scenarios.

In the 2. case, first, the spatial panning in the stereo position changes according to the deviation of the given y-plane. Then the panorama and a second sound is played, the latter according to the deviation of the z-plane.

In the 3. case, more or less attack or crunchiness of the sound is rendered according to different high acceleration changes. Changes in the “constant speed” phase are augmented with overdone volume changes. Also the ending of the bow, the deceleration phase is overdone with abrupt volume decrease or even combined with a crunchiness sound, to point out the importance of this matter.

Simple ticking sounds point to a missing or too strong rotation of the hand. In a certain range of rotation, similar to the 1. case, the volume is raised or reduced.

6. DISCUSSION

The possibilities of sonification will show in an intuitive way, that every change of the movement induces and influences a sound or changes certain parameters of an existing sound. This helps to understand intuitively, how a special movement, in this case the bowing on a stringed instrument works. The closed-loop auditory feedback supports the learning and the optimization of the movement, only by hearing. Hearing, the most important feedback channel for musicians, can't be trained enough by music students and pupils.

Our first impression is that the continuous mapping sonifications described above for case 1 and 2 work great and are quite efficient to direct the attention to improper executions. A comparison of strategies in their ability to induce better execution needs psychophysical studies. Also the age-related reactions and adapted sonifications are tested.

7. CONCLUSION AND NEXT STEPS

This paper has introduced the *sonified music stand* as a portable, integrated, versatile, interactive sonification system for musicians. The system combines sensor technology, real-time sonification, and new ideas on integrating multi-channel audio projection into a standard music stand into a usable every day system. The presented application has been specifically selected and optimized for the task of violin learning and the sonification examples demonstrate that the sound conveys useful information. This prototype system is still in a quite early state and we plan to conduct long-term user studies after we arrived at a couple of competitive and useful sonification approaches. We hope that our sonified music stand can make a positive contribution to better pedagogic approaches and methodical understanding, exercising-productivity rising methods and ultimately to the development of more healthy practices for musicians.

Future scenarios provide a highly compact multi-channel loudspeaker array, which can also be fixed to other displays such as computer monitors or other surfaces.

The next step is the development of a at least 8 + 1channel portable speaker setup for better sound localization in 3D sound scenarios and for complete loudspeaker arrays around surfaces, monitors, displays etc.. A high-power battery supply for off-the-line usage and several attachment features allow simple usage and fixing on and beside nearly every surface, monitors, stands, walls etc.. Also audio radio frequency transmission from the computer to the speaker is considered.

Finally, we are very convinced that we can easily adapt the system to other musical instrument playing problems and even to other fields such as movement training in sports and dance, where sonification can help to better learn and perform complex movements.

8.ACKNOWLEDGEMENT

The work has been supported by CITEC – Center of Excellence in Cognitive Interaction Technology. Thanks to Till Bovermann and Florian Grond for their help with SuperCollider programming.

9. REFERENCES

[1] Hermann, T. A*n introduction to interactive sonification* (guest editors' introduction). *IEEE MultiMedia*, 12(2):20–24, 04 2005

[2] T. Hermann, T. *AcouMotion - an interactive sonification system for acoustic motion control.* In S. Gibet, N. Courty, and J.-F. Kamp, editors, Gesture in Human-Computer Interaction and Simulation: 6th International Gesture Workshop, GW 2005, Berder Island, France, May 18-20, 2005, Revised Selected Papers, volume 3881/2006 of Lecture Notes in Computer Science, pages 312–323, Springer, Berlin, Heidelberg, 2006

[3] McMillen, K. *Stage-Worthy Sensor Bows for Stringed Instruments*, Casa Paganini, NIME 2008

[4] Bevilacqua, F. *The augmented violin project: research, composition and performance report*, NIME06, Paris, France, 2006

[5] Young, D. *Wireless sensor system for measurment of violin bowing parameters*, Music Acoustics Conference, Stockholm, 2003

[6] Goltz, v. d. C. *Orientierungsmodelle fur den Instrumentalunterricht*, G. Bosse Verlag, ISBN 978-3764925826, 1974

[7] Ferguson, S. *Learning Musical Instrument Skills Through Interactive Sonification,* Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06), Paris, France, 2006

[8] Grosshauser, T. *Seeing the Inaudible. Descriptors Used for Generating Objective and Reproducible Data in Real-Time for Musical Instrument Playing Standard Situations*, AES Convention, Amsterdam, Netherlands, 2008

MUSICAL GROOVE IS CORRELATED WITH PROPERTIES OF THE AUDIO SIGNAL AS REVEALED BY COMPUTATIONAL MODELLING, DEPENDING ON MUSICAL STYLE

Guy Madison
Umeå University
guy.madison@psy.umu.se

Fabien Gouyon
INESC Porto
fgouyon@inescporto.pt

Fredrik Ullén
Karolinska Institutet
fredrik.ullen@ki.se

ABSTRACT

With groove we mean the subjective experience of wanting to move rhythmically when listening to music. Previous research has indicated that physical properties of the sound signal contribute to groove - as opposed to mere association due to previous exposure, for example. Here, a number of quantitative descriptors of rhythmic and temporal properties were derived from the audio signal by means of computational modeling methods. The music examples were 100 samples from 5 distinct music styles, which were all unfamiliar to the listeners. Listeners' ratings of groove were correlated with aspects of rhythmic patterning for Greek, Indian, Samba, and West African music. Microtiming was positively correlated with groove for Samba and negatively correlated with groove for Greek, but had very small unique contributions in addition to the rhythmical properties. For Jazz, none of the measured properties had any significant contributions to groove ratings.

1. INTRODUCTION

People often find themselves spontaneously nodding or tapping their feet when they hear music, and much music is indeed intended for synchronised movement in the form of dance, drill, and ritual behaviours. There are indications that the connection between movement and the rhythmic component of music is biologically determined [1,2]. In order to examine these potential connections, it is important to find out which physical properties in the musical signal - if any - are correlated with listeners' experience of wanting to move to the music.

Music features several forms of temporal structure. A priori, one could assume that two of these are particularly relevant for groove: rhythmic patterning and systematic microtiming. Rhythmic patterning is the assignment of different sound events to certain canonical time values, i.e. to certain positions in the metrical grid. It seems likely that the degree of repetitive rhythmical patterning may be related to the experience of groove. Microtiming refers to deviations from canonical time values that are often found in real music performances. Systematic microtiming patterns are those that covary, either within [3] or across performers [4].

SMC 2009, July 23-25, Porto, Portugal

Here, we examine the correlation between ratings of groove and numerical descriptors of the sound signal that correspond to (a) the salience of periodic events close to preferred tempo; the relative salience of sound events that are faster than the beat level, both (b) those that repeatedly occur at small-integer (metrical) subdivisions of the beat level and (c) those that do not; and (d) recurrent micro-timing structuring of events. In addition to this, we also considered (e) unsystematic micro-timing [5].

2. METHODS

Nineteen non-musicians, ranged from 19 to 32 years in age, were paid for acting as listeners. Twenty music examples (ME) were selected by the authors as representative of each of five traditional folk music traditions from a certain region, here referred to as Greek, Indian, Jazz, Samba, and West African. The 100 MEs were copied from commercially available CDs and were copied from any position within the original sound track that constituted a representative and musically meaningful example of that track, containing at least one rhythmic phrase with a prominent steady beat. MEs were 9.06-14.55 s in duration, were adjusted to equal amplitude, and their tempi ranged from 81 to 181 BPM. The dependent variable was the participants' experience of Groove, defined as "evokes the sensation of wanting to move some part of the body" and rated on an 11-point scale. The scale appeared as a horizontal line anchored "not at all appropriate" (0) and "very appropriate" (10). Each session began with 10 MEs, two from each of the five styles but different from the experimental MEs, whose purpose was to orient participants about the range of properties to rate in the experiment. Ratings from the first block were not included in the analysis. The sound descriptors measured the magnitude of rhythmical and temporal properties corresponding to the psychological properties outlined in the introduction, namely Event density, Beat salience, Fast metrical levels, Systematic microtiming (MT), and Unsystematic MT. Details of the computational approach are the object of a forthcoming paper.

3. RESULTS AND DISCUSSION

The strongest correlations between Groove and the descriptors were found for Beat Salience and Event Density, and these correlations were also stronger among the Greek, Indian, and Samba styles than among West African and Jazz. Jazz exhibited very small correlations overall. Rhythmical descriptors seem to play a substantially greater role than do microtiming descriptors across all styles, and they also generally exhibit larger correlations within each style. There was also an interaction between descriptor property and music style, in that Systematic MT seems to play no role at all for Indian, Jazz, and West African, but a substantial role for Samba. Finally, Unsystematic MT seems not to play any role for groove. It is likely that the descriptors are to some extent intrinsically dependent and that the measured properties in the music examples covary to some extent. To assess the unique contributions of each descriptor, a stepwise multiple regression was performed for each style and for all styles together, the results of which are summarised in Fig. 1.

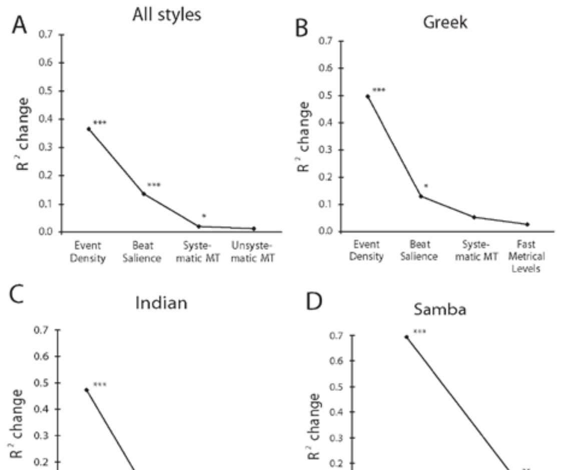


Figure 1. Changes in explained variance according to stepwise linear regression of groove ratings on the five descriptors. Note. *** = p < .001, ** = p < .01, * = p < .05.

Only one descriptor passed the entry criterion (F > 1.0) for Jazz and West African, respectively, and these are therefore not shown in the figure. Removing Fast Metrical Levels from the model subtracted 6.43 percent of the total explained variance by all descriptors (14.2%) for Jazz, and removing Beat Salience likewise subtracted 25.65 percent from the total 28.8 percent for West African. The changes in explained variance for Greek (total R^2 = 0.709; df = 5, 14), Indian (0.631), Samba (0.841), and for all styles pooled (0.537; df = 5, 94) are shown in the figure.

Jazz is commonly associated with expressive performance. In particular has groove been attributed to the swing ratio, the relative duration of the two "swung" notes in the rhythmic ostinato so characteristic for much classical jazz music. Not all jazz features swing in this sense, but 15 of the present Jazz examples did. Yet did not the variation in the swing ratio, nor any other microtiming pattern, exhibit even a tendency for being related to groove. Although contrary to popular belief, this corresponds perfectly with the finding that the swing ratio is trivially related to tempo [6], suggesting that its purpose is merely to make the two intervals discriminable, in effect to maintain a rhythmical pattern. Samba was characterised by high correlations with no less than four descriptors, which naturally proved to be highly redundant. This suggests that Samba employs all possible means to induce groove, as might be expected by music dedicated to dancing, including a strongly accentuated and continuously repeated beat provided by the *surdo* and accentuated Fast Metrical Levels by the *pandeiro*.

In conclusion, these results indicate a ubiquitous relation between the rhythmical descriptors and groove for all styles but Jazz, whereas relations with microtiming are in fact negative whenever they remain significant (for Greek and Indian). This provides further empirical evidence that periodicity is a key component for movement induction trhough music, and strengthens the position of groove as a perceptually salient dimension of music.

4. REFERENCES

[1] Madison G. Experiencing groove induced by music: consistency and phenomenology. Music Perception 2006; 24: 201-208.

[2] Merker B, Madison G, Eckerdal P. On the role and origin of isochrony in human rhythmic entrainment. Cortex 2009; 45: 4-17.

[3] Shaffer LH, Todd NPM. The interpretative component in musical performance. In: Aiello R (ed.), Musical perceptions. New York: Oxford University Press; 1994: 258-270.

[4] Repp BH. A microcosm of musical expression I: Quantitative analysis of pianists' timing in the initial measures of Chopin's Etude in E major. JASA 1998; 104: 1085-1100.

[5] Keil C. The theory of participatory discrepancies: a progress report. Ethnomusicology 1995; 39: 1-19.

[6] Friberg A, Sundström A. Swing ratios and ensemble timing in jazz performance: Evidence for a common rhythmic pattern. Music Perception 2002; 19: 333-349.

ENHANCING EXPRESSIVE AND TECHNICAL PERFORMANCE IN MUSICAL VIDEO GAMES

R. Marczak, M. Robine, M. Desainte-Catherine, A. Allombert, P. Hanna and G. Kurtag
LaBRI - SCRIME - University of Bordeaux
{marczak, robine, myriam, allomber, hanna}@labri.fr
gyorgy.kurtag@gmail.com

ABSTRACT

Musical video games are best sellers games. One of their selling point is based on the improvement of players' musical abilities. But interviews made with gamers and musicians show that if the former feel enough freedom in musical expression, the latter are more sceptical and feel more limited in the possibility of express themselves when they are playing. In parallel with the games development, some research works propose to improve control and meta-control on music to allow expressive performance without the user being a virtuoso. The goal of this article is first to present interviews made for knowing users opinions. Some research works about improving musical expressive performance are then presented. Finally, we propose games enhancing the expressive and technical musical performance, linking current game-play with current research.

1 INTRODUCTION

For more than ten years, *Bemani* [1] has developed rhythm video games. In the beginning, this unusual game-play attracted very few players, mainly in Japan. But the generalization of intuitive interfaces, *e.g.* the *Wii Remote*, and the appearance of musical games with Western soundtrack-lists, make this game-play now attractive to a wide audience. More and more editors therefore propose some musical video games, and *Guitar Hero* or *Singstar* are nowadays best-selling games.

Like the famous *Brain Age* game [2], basing all its selling point on the improvement of extra game capacities (players' intelligence and memorization), these games are partially basing their selling point on the improvement of player's musical abilities. A Guitar Center Survey [3] indicates that 67 percent of rhythm gamers will certainly buy a real instrument in mid-term. So playing musical video games could make players wanted to play real instruments. But do they really learn music playing this kind of game? Do they have enough freedom to express themselves musically? And what about the sensation of playing as a band when several players are allowed?

This article starts with the description of musical video games main characteristics in section 2. Section 3 presents the main categories of musical video games with some famous examples. Interviews with different gamers and musicians illustrate the main assets and limits of this kind of games in section 4. We then present in section 5 some hardware and software developed in a computer music research environment. A way of pooling musical video games and this research in innovative games is finally proposed in section 6.

This work is part of the SCRIME project (Studio de Création et de Recherche en Informatique et Musique électroacoustique: http://scrime.labri.fr), funded by the DMDTS of the French Culture Ministry, the Aquitaine Regional Council. It is also a part of SIMBALS project (JC07-188930) and VIRAGE project (07 AM 011 01), funded by the French National Research Agency.

[1] `http://www.konami.jp/bemani/`
[2] `http://www.brainage.com/`
[3] `http://www.1up.com/do/newsStory?cId=3171507`

SMC 2009, July 23-25, Porto, Portugal

2 PROPERTIES OF MUSICAL VIDEO GAMES

We distinguish two main components of the musical performance in the games: physical-technical and expressive parts [8]. The physical element is the gamer morphology influence and the contr oller shape used for the performance. The technical element is the gamer technical level influence, considering the interaction quality between the gamer and the controller to reach an given objective. We will group these two elements together in the technique description. The expressive element is defined by what is neither physical, nor technical. It can be considered as the intentional deviations from a reference, *i.e.* deviation in rhythm, articulation, dynamics or the adding of effects such as vibrato or timbre changes.

2.1 Technique

We propose to evaluate musical game technique as the difficulty to reach the proposed goal, or a certain level, with a given controller. We can distinguish two technical parts: the musical technique, *e.g.* the capacity to play in rhythm, and the controller technique, *i.e.* the capacity to control the hardware.

The main technical characteristic of a musical video games is linked to the specific **hardware**. It can be a joystick, or it can more or less imitate a real instrument. Hardware can induce some latency in the game for the sound production.

The **accessibility** of the game is then an interesting parameter. Depending on the controller, the game may be **tiring**, *e.g.* with a drum controller. The technical **difficulty** may be appreciated considering the time needed to pass a level or to end the game. It could be broken down into the difficulty of controlling the hardware and the difficulty relative to the musical technique.

2.2 Evaluation

The evaluation is the feedback provided to the gamer. It may be **global**, with a rank for the whole performance, **semi-global** with a rank for each part of the music, *e.g.* verse1, chorus, verse2. The feedback may be more precise, **local**, to indicate the wrong notes played or the value of a time deviation. It may also be given in **real-time**, with a video environment changing with real-time evaluation during the performance.

Most of the time an **absolute** evaluation is provided, *e.g.* the percentage of matching notes. The player could also be evaluated considering another performance, from another player: it is a **relative** evaluation. Some games propose a **self-evaluation**: the gamer evaluates his own performance.

The **relevance** of the feedback is a very interesting game parameter. Is the evaluation useful to improve the performance or to reach a level of the game ? Is the feedback more precise according to the technical level played ?

2.3 Immersion

The environment is very important in musical video games. A game can be boring or **exciting** regarding to the chance to reach the next level .The **quality of the soundtracks** or the sound synthesis is also relevant. Editors pay a particular attention in the **3D-environment** and 3D-effects for recent games, *e.g.* including virtual avatars. The **controller shape** may be fun too, *e.g.* by imitate famous guitars.

2.4 Expressiveness

The expressiveness is allowed in certain games by giving **musical freedom**, *e.g.* playing notes when no match is proposed or changing the music tempo in real-time. The **level of control** indicates the elements controlled by the player during the game: it may be simply the notes production but also the phrasing or the global tempo. When a **multi-players** mode is available, the degree of interaction between the players is a expressiveness parameter of a musical game.

3 MUSICAL VIDEO GAMES

This section presents best-selling games considering their main goal.

3.1 Scrolling score games

These games, e.g. *Pop'N'Music* [4] , *Guitar Hero World Tour* [5] or *Rock Band* [6] are based on the *Bemani* game-play: a scrolling score is displayed and every time a note crosses a line located at the bottom of the screen, the player must hit the matching buttons (by color and position) on his controller. For *Pop'N'Music*, the controller is specifically designed, and do not look like any real instrument nor joystick. For *Guitar Hero* and *Rock Band*, controllers looks like real instruments, but are significantly simplified since the guitar cords are replaced by buttons for example. After each song played, a rank is given. For *Pop'N'Music*, it is a number between 0 and 100000. For *Guitar Hero* and *Rock Band*, it is a percentage of matching notes.

3.2 Karaoke games

These games, e.g. *Singstar* [7] , are based on singing performance. A pitch-score is displayed with lyrics, and the player knows in real-time if he sings correctly. Rank is given after each song, *e.g.* a grade between "casserole" to "singstar" in *Singstar*.

3.3 Performance games

Wii Music - Conductor Game [8] : this game puts the user in the position of a conductor. By moving the Wii remote, players give the tempo to the orchestra. The nuance and the articulation of the synthesized music are in accordance with moves speed and amplitude. At the end of the performance, a rank between 0 and 100 is given.

Wii Music - Performance Game: this game offers the ability to play music by moving the Wii remote. After the selection of a song and an instrument, the player generates sound by reproducing a gesture associated with the corresponding instrument using the Wii remote. A reference score is displayed, but the player can add or remove notes, but he can not change the tempo. At the end of the performance, the player must evaluate himself.

3.4 Memory games

These games, for example *Space Channel 5* [9] , are based on the *Simon* electronic game: the player must repeat a proposed sequence of notes, in rhythm, with a joystick. A rank is given after each level, and the environment is more and more consistent as the player succeed.

4 INTERVIEWS

We interviewed the *DDR Bordeaux Association* [10] . Its aim is to present musical video games to general public as well

[4] `http://www.konami.jp/bemani/popn/`
[5] `http://worldtour.guitarhero.com/`
[6] `http://www.rockband.com/`
[7] `http://www.singstargame.com/`
[8] `http://www.wiimusic.com/`
[9] `http://www.mobygames.com/game/space-channel-5`
[10] `http://www.ddrbordeaux.com/`

as gathering some gamers for practicing. We also interviewed musicians who accepted to give their impressions after a first contact with this kind of games.

4.1 Gamers

Several gamers from the association accepted to give us their opinion on their favorite musical video games.

Technique
All presented video games are really accessible. If the main goal is to perfectly read score like in scrolling score games, many levels of difficulties exist to please beginners and expert. If the main goal is more focus on the musical performance, like in *Wii Music* or *Singstar*, levels of difficulties are nonexistent or not really pertinent.

Because scrolling score games are hard over a certain level of difficulties, a practice mode is provided. For *Guitar Hero* or *Rock Band*, the most difficult level could be reached after some months ; after some years for *Pop'N'Music*. To find his technical level in a game, a player will often try a hard level at first, and if he failed, he will try levels that are more and more easy, until he succeeds.

When players are used to scrolling score, the difficulty is mainly because of the hardware. While asking them if they would like to play using real instruments with captors, they say that it is a possibility, but only if this will not decrease the fun of the game.

Evaluation
Every game provides an evaluation. For *Singstar*, the remark was done that this game is not made to be more and more skilled, but for having fun. So the global evaluation is funny but not very precise. For *Pop'N'Music* and *Rock Band*, the evaluation is global too. For *Guitar Hero*, the evaluation is semi-global, based on verses and chorus, and the public reaction in the game helps to know the player level in real-time. For *Space Channel 5*, the evaluation is in real-time too, based on the environment more and more dense if the player succeeds. *Wii Music - Conductor Game* is the only game to propose a relative evaluation: the second player must follow the first one to have a good rank. *Wii Music - Performance Game* is really specific because the player must provide a self-evaluation: he choose a rank between 0 and 100, according to the pleasure he takes listening his own performance.

Immersion
All interviewees insist on the importance of extra-musical environment. It gives a style to games, *e.g.* in *Pop'M'Music* where colored graphics puts gamers in a happy mood. It is also a way for immersing players, *e.g.* in *Guitar Hero* where players could create avatars that look like them. They also like having specialized controller, because they add originality to the game.

About the sound quality, it is noteworthy that when more freedom is given in music performance, it is often at the expense of sound quality. And this quality is important, some players buy Original SoundTrack (OST) of games. All games are considered exciting to play, notably because players want to unlock the next song. The only ones that are physically difficult to play are *Rock Band* and *Guitar Hero* with the drums hardware that emulates the real instrument.

Performance
Except for *Wii Music*, a little freedom is given for the user to express musically. Scrolling score games impose to follow the score, without any agogic modification, *i.e.* dates modification of notes beginning and ending. *Pop'N'Music* allows adding notes even if it is not advised, *Guitar Hero* allows vibrato with guitar and *Rock Band* allows some drums improvisation parts. However, none of the interviewees tell that they lack freedom for these games. They are accustomed to *Bemani* way of playing, and they have competence in it.

They confide that scrolling games give them reading competences but no real musical competences. More surprising, they say that it is possible to play muting music, but it is impossible to play without graphics. It is noteworthy that game with more musical interest, like *Wii Music* or *Space Channel 5*, could be played without watching screen, but they seem less attractive to hardcore gamers.

Gamers tell that the multi-players mode is a value-added for games, but they often play in the same way as when they are alone. The interest is more in being one another instrumentalist or avatar than playing interactively with others.

4.2 Musicians

We asked musicians who have learned music for 10 to 20 years to give an opinion on musical video games after a first experience, and many criticisms were made about performance limits. In *Rock Band* or *Guitar Hero*, the multi-players mode has no real interest because players do not really play with others, and so they do not listen to them. The hardware is really restrictive, except some drums controllers. They had trouble when playing easy mode, because they habitually trend to add notes. Since the scrolling score manages the time, they have the feeling of being too constrained. Contrary to hardcore gamers, they are more comfortable with the games from *Wii Music*, because they have more space to express themselves.

When they are playing musical video games, gamers would appreciate to learn music or a particular instrument, and musicians would be delighted by more musical freedom. Some researches on musical performance could certainly provide such characteristics to this kind of games.

5 RESEARCH ON MUSICAL PERFORMANCE

This section presents existing hardware and software that help the technical element abstraction to focus more on expressiveness, by decreasing the human action frequency. They give the possibility for the user to be expressive without being a virtuoso, by offering controls and meta-controls. Works on music similarity show that a pertinent feedback on the performance could be provided.

5.1 Meta-Control Instruments

The *BAO PAO* is an electronic instrument developed by Jean Schmutz in collaboration with Jean Haury at the *La Puce A L'Oreille* association [11]. This instrument was firstly designed for handicapped people. The user does not directly produce sound neither the pitch: all is generated by a computer linked to the *BAO PAO*. A score is loaded into the computer, and the user could move in it note by note by cutting a laser beam with a stick. If stick speed is fast, the sound is *forte*. If the speed is slow, the sound is *piano*. This device allows musician to focus on expressive interpretation. It also permits a group of handicapped to play together. It is now used as an instrument in many secondary schools for music education.

The *Meta Piano* [12] [4] is an instrument developed by Jean Haury which mainly inspired the *BAO PAO*. It looks like a simplified piano, with just 9 notes. The mechanism is similar to the *BAO PAO*. A score is loaded into a computer, and the user could move in the score by pressing any note on the *Meta Piano*. It proposes more freedom in the performance than the *BAO PAO* thanks to its several notes: the musician could make expressive performance, because it allows working on rhythm, duration, articulation, intensity, phrased and all agogic deformations.

5.2 The Continuator

The *Continuator* [7] is a MIDI device, developed by François Pachet, which listens to MIDI events, and proposes to the musician new notes to follow the sequence by analyzing the previous ones played by the user. In real time, the system can determine the performance style. The continuator is a powerful device that pleases to confirmed musicians and as well as young children. It is mostly used in improvisation, but the way it computes the next notes to be played could be useful for guiding musician in their performance.

5.3 I-Score

I-Score [1] is a system for composing and interpreting interactive scores. It relies on a formal musical scores representation which includes the way a performer can interact with them. The possibilities of interaction with a score are based on an interpretation formalisation of instrumental pieces of music. In the current version of *I-Score*, these possibilities consist of interactively triggering some discrete events of the score (beginnings and endings of notes) during the performance. This liberty for the performer comes with the possibility for the composer to define temporal constraints on the events to fix a partial order between them.

During the execution, the performer is allowed to trigger an interactive event only when the events of the score that *must* appear before him (according to the temporal constraints) have occurred. Symmetrically, if the performer triggers an interactive event at a different date from the date written in the score, the system will automatically adapt the date of the events that *must* appear after it, to respect the order imposed by the constraints.

The ability of the system to automatically adapt the rest of the score according to the triggering of an interactive event, makes sure that the execution will reach the end of the score and respect the limits wanted by the composer. As a consequence, the performer can give his own interpretation of the musical piece while he is guaranteed that the score will be played until its end.

[11] `http://lapucealoreille.free.fr/`

[12] `http://dept-info.labri.u-bordeaux.fr/~marczak/MetaPiano/`

5.4 Classification overview

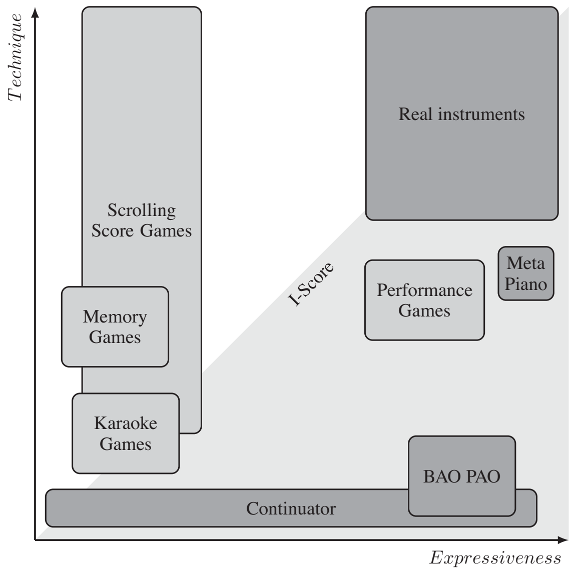


Figure 1. Classification Overview

Figure 1 makes an overview and attempts a classification by technique and expressiveness.

Memory games, even if they permit musical learning, do not gives freedom for expressiveness. The pattern must be perfectly repeated. They require technique because practice is needed to correctly memorize and play in rhythm. **Karaoke games** are very accessible, but the singer must always sing respecting the original pitch. For **scrolling score games**, the technique required could be variable, thanks to multiple control levels. *Guitar Hero* has a "novice" mode that allows the musician to play by pressing any note on time, and *Pop'N'Music* required years of practice to succeeded the higher level. The expressiveness is limited in theses games, but each one gives a little freedom in different manner. **Performance games** required practice to understand the link between the controller and generated music. They give freedom, but it is either on tempo or on notes addition, not both.

Real instruments like guitar, drums, bass... are the most expressive, but also the most technique. The **BAO PAO** is very simple to use, notably by handicapped people and young children, and allow agogic deformation. The **Meta-Piano** is more expressive, notably by allowing expressiveness on transition, but it requires some habit for playing notes. The **Continuator** is very accessible, and the expressiveness depends on the expressiveness given by the musician. Finally, **I-Score** allows any levels of expressiveness with any level of technique (depending on the number of trigger points, and on the controls or meta-controls given to the musician).

5.5 Musical Similarity

Estimating the similarity between musical performances is a difficult open problem. From a computational point of view, it consists in determining algorithms calculating a measure which indicates the degree of similarity between two musical segments. Each musical segment may be represented by a sequence of symbols, related to musical properties such as timbre, rhythm, melody, etc. Several techniques for evaluating similarities between symbolically encoded music have been introduced during the last few years. Geometric algorithms consider geometric representations and compute the distance between these objects [5]. Algorithms adapted from string matching domain and based on N-grams techniques are proposed in [11].

Other methods, generally applied in computational biology, compute a similarity measure between two strings of symbols as the maximum score sequence of elementary operations (insertion, deletion, substitution, etc.) needed to transform one of the strings into the other. For example, *local alignment* algorithm finds and extracts a pair of regions, one from each of the two given strings, that exhibits high similarity [10]. Applications to the estimation of the musical similarity show that such approaches are very accurate [6], since adaptations specific to the musical context have been proposed [3].

6 PROPOSITION OF GAMES ENHANCING TECHNICAL OR EXPRESSIVE PERFORMANCE

The main goal of this section is to link current research with musical video games, by proposing a timing improvement and two video games that enhanced respectively technical and expressive performance.

6.1 Improvement for timing

In music, time must never be completely delegated to computer. The principal risk is to have gestures precipitated and not felt. In order to have a personal expressive performance, the player must operate following his **own internal rhythm**. Then, the player could express himself by changing the speed of the gesture. When a score is presented as a notes scrolling, the player looses his thinking and feeling because he totally focuses on the current note.

Thanks to presented research (section 5), a player can have the freedom to **modify the tempo** of the music. Instead of giving him note onset, a score can be made with

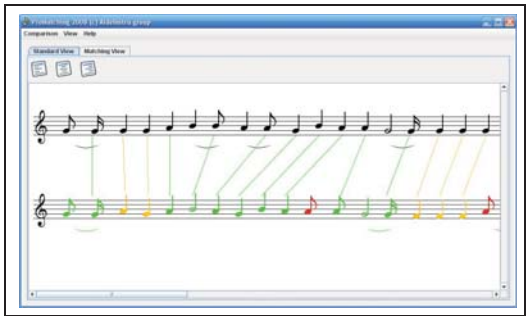


Figure 2. Example of local feedback. The upper line is the score to perform. Below, the performance contains green matching notes, red notes which have been inserted and yellow notes with the bad pitch.

bounds in which he could play the note. Using the Continuator mechanism (subsection 5.2), the score would propose **bounds more and more restrictive and precise** to respect the player way of performing.

We can also imagine a behavior like *BAO PAO*, *Meta Piano* (subsection 5.1) or *I-Score* (subsection 5.3): the score stops as long as players do not hit the good note. It would give time to players to understand how to make this note. This is a good improvement for practicing a difficult score.

6.2 Learning game

We make here propositions for a learning game, i.e. a musical video game that makes the player progress in musical technique. First, if the player wants to acquire the technique of a particular instrument, **the controller must be this instrument**, or a very good copy of it, like some existing drums controllers for example. The choice of a controller is important since the player acquire naturally a technique depending on the controller he used. MIDI instruments are particularly well adapted, but we can imagine to use real monophonic instrument with sound analysis, like with existing voice games.

A very important point is also to propose a **pertinent evaluation** to the player. If he knows exactly what was wrong, and if the feedback is adapted to his level, he would know how to perform better. One application from researches in music similarity is presented by figure 2. Integrating this kind of feedback could provide for a musical game player a local evaluation: he would know where he was not in tune, or where the vibrato was not regular. He can **identify his weakest points** and can choose to focus on particular technical exercises that could be provided in a practice mode.

Another point is to propose a **relative evaluation** by comparing different performances thanks to similarity methods again. A player could be evaluated by comparing his performance to a score, but also to his last performance, or to the performance of a professional musician or other players[9]. If the data are centralized on a server, a player could then evaluate himself to a particular group of people, with the same age or from a conservatory of music for example.

6.3 Band game

This parts is a game proposition that would enhance the impression of playing as a band.

Controllers would look like **real instruments**. Works with handicapped people point out the importance of "social instruments". Moreover, the sensation of playing real instrument is a key point to enhance the motivation.

Concerning the evaluation, the final rank would mostly take into account **relative evaluation**. The drummer would **influence the tempo** of the performance (subsection 6.1), and the evaluation would be based on how other players respect drummer's tempo. Because the tempo could be different from the original, a similarity test (subsection 5.5) would be made on the drummer performance. Since absolute evaluation is important for hardcore gamers, it must also be provided, but it would have a minor influence on ranking.

This game must give the impression to play together, as a band. Firstly, **watching other players** instead of the screen is primordial. For this, two modes would be available: **filming players** (*e.g.* with web-cams) and displaying them on the screen with the scores; or **using augmented reality**[2], and displaying scores on other players or their instruments.

Then, this game would be based on **patterns**, *i.e.* macrostructure of notes, instead of single notes. Working with children show that is more effective to talk about gestures or patterns rather than notes. A musician-player could then focus more on expressiveness. For enhancing band play, a player would also have an **action over other players' score**. This would be done using *I-Score* (subsection 5.3) in which scenarios could send information to other scenarios for triggering an interactive event. Thanks to this, if a player has difficulty to make a pattern, the game will give him more time by repeating other players' patterns as long as he does not succeed. When he succeeds, an action on others players scores would unlock the next pattern. If all players are lost, a trick could be used to again synchronize all scores.

Finally, a good way to play together is to **learn the score**. The video game would then propose a mode in which the score slowly disappear if the player succeed. It is important for this mode to be a real part of the game, because hardcore gamers like trying to reach high score in difficult modes. If a player knows the score, he could play together without being distracted by the screen, and can focus on the interpretation. Another mode proposed would be a dialog mode, like in *Space Channel 5*, but in multi-players: a player plays a parts, and the other player try to repeat it.

7 CONCLUSION AND PERSPECTIVES

Nowadays, gamers get a strong enthusiasm from musical video games, but these games do not give significant freedom for expressive and technical performance.

By applying current research, developed in the community of computer music, both parts could take benefits. Games would became more expressive and technique, and research could use games for analysing feedback, *e.g.*, to evaluate the relevance of a ranking algorithm.

In relation with the games proposed in this paper, we could consider the following perspective. Because sound quality is a key point for enhancing motivation, we would like to use original versions of songs. When significant freedom is given, music is mainly synthesized to allow effect application, *e.g.* time-stretched or repetition. To use original versions of songs, we must apply high-quality audio transformations [12].

8 REFERENCES

[1] A. Allombert, G. Assayag, and M. Desainte-Catherine. A model of interactive scores based on petri nets. In *Pr. of the 4th Sound Music Computing Conference*, 2007.

[2] O. Bimber and R. Raskar. In L. A K Peters, editor, *Spatial Augmented Reality, Merging Real and Virual Worlds*. 2005.

[3] P. Hanna, P. Ferraro, and M. Robine. On Optimizing the Editing Algorithms for Evaluating Similarity Between Monophonic Musical Sequences. *Journal of New Music Research*, 36(4):267–279, 2007.

[4] J. Haury. La pianotechnie ou notage des partitions musicales pour une interpretation immediate sur le metapiano. In *Pr. of the 14th Journees d'Informatique Musicale, Grenoble, France*, 2009.

[5] K. Lemström and A. Pienimäki. Approaches for Content-Based Retrieval of Symbolically Encoded Polyphonic Music. In *Pr. of the 9th International Conference on Music Perception and Cognition*, 2006.

[6] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24(3):161–175, 1990.

[7] F. Pachet. The continuator: Musical interaction with style. In ICMA, editor, *Proceedings of ICMC*, pages 211–218. ICMA, September 2002. best paper award.

[8] M. Robine. *Analyse de la performance musicale et synthese sonore rapide*. PhD thesis, Ecole doctorale de mathematiques et d'informatique, Bordeaux I, 2006.

[9] M. Robine and M. Lagrange. Evaluation of the technical level of saxophone performers by considering the evolution of spectral parameters of the sound. In *Pr. of the International Conference on Music Information Retrieval*, pages 79–84, 2006.

[10] T. Smith and M. Waterman. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[11] A. L. Uitdenbogerd. *Music Information Retrieval Technology*. PhD thesis, RMIT University, 2002.

[12] V. Verfaille. *Effets audionumérique adaptatifs : théorie, mise en œuvre et usage en création musicale numérique*. PhD thesis, Université Aix-Marseille II, 2003.

PROTOTYPING MUSICAL EXPERIMENTS FOR TANGISENSE, A TANGIBLE AND TRACEABLE TABLE

Daniel Arfib
Laboratoire d'Informatique de Grenoble (LIG)
Daniel.Arfib@imag.fr

Jean-Julien Filatriau
Université Catholique de Louvain la Neuve (UCL)
Jehan-Julien.Filatriau@uclouvain.be

Loïc Kessous
Université Paris6 (ISIR-UPMC)
loic.kessous@gmail.com

ABSTRACT

We describe two musical experiments that are designed for the interaction with a new tangible interface named by us Tangisense, based on a set of antennas and RFIDs. These experiments (classification, game of) use different kind of time schedule, and are now simulated using Max-Msp and Java programs and common computer-human interfaces. They are developed in such a way that they can be ported on a specific tangible interface using RFID tags in its heart. Details about this portage are given. These experiments will in the future serve as user-centered applications of this interactive table, be it for musical practice or sonic interaction design.

1. INTRODUCTION

We will in this article do a prospective plunge in the use of a tangible interface in the artistic or pedagogic domain. We will try as far as possible to bring a framework rather than just experiences. Our study is sound oriented rather than device oriented, which means that the goal of this interaction is the sound. Levels of interaction depend upon the making of the sound, and we will see that the timing itself of the sound production (out of real time, events and curves in real time) drives the interaction style.

Touchable and tangible tables are now largely used in music production. Touchable devices (including touch screens) are popularized by the Ipod and Iphone series, but belong to a long tradition, which includes multitouch tables. Tangible interfaces by definition use objects, and are also named graspable. Links can be found on this topic [1]. Musical applications have been popularized by the Reactable (see [2] for an overview on this subject), which uses special tags named “fiducials”, and NIME conferences (New Interfaces for Musical Expression) are full of articles and demos on such subjects[1].

[1] http://www.nime.org

SMC 2009, July 23-25, Porto, Portugal

2. TANGISENSE, A TANGIBLE AND TRACEABLE TABLE

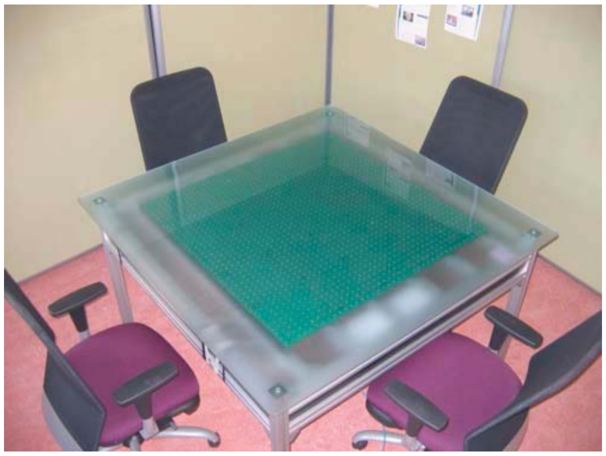


Figure 1. the first protoptype of the Tangisense table

Our Tangisense table is part of a user-centered interface, named Tangisense in the TTT project (Interactive table with tangible and traceable object). The hardware itself can be described as a retina of antennas, which can track RFID tags (Figure 1). The table is composed of 25 tiles, each of them containing 64 antennas. Each tile contains a DSP processor that reads the antennas, an antenna multiplexer, and a communication processor. The table is connected via a control interface to the host computer by an Ethernet bus. From the computer point of view, every time a tag is detected one gets its specific identification number and coordinates. This Tangisense table is part of a French national project named TTT, which involves two laboratories ((LAMIH, LIG) and two companies (CEA, RFIdées). It is now in a prototype stage, and the definitive version should be available mid-2009.

More than the hardware itself, the software structure is an important feature of this table. An architecture in 3 layers has been defined (Figure 2). These layers are written in Java.

1-the capture and interface layer handles tangible objects with tags (one or may per object) and creates Java objects associated to forms.

2-.the traceability layer handles events associated to these objects. It communicates the information to the next layer, but also traces the information and can be asked information about these objects-moves

3- the applicative layer is the one where the specific use of the table is done, in our case the musical experiments.

We usually design this musical application layer in two parts: one, written in Java, communicates with the capture and traceability levels, gathering information and transforming it into events or curves; the second one activates the sound. Communication between these two parts use a bidirectional OSC protocol (using the Java-OSC library).This second part is most of the time a Max-Msp patch, specific for each experiment.

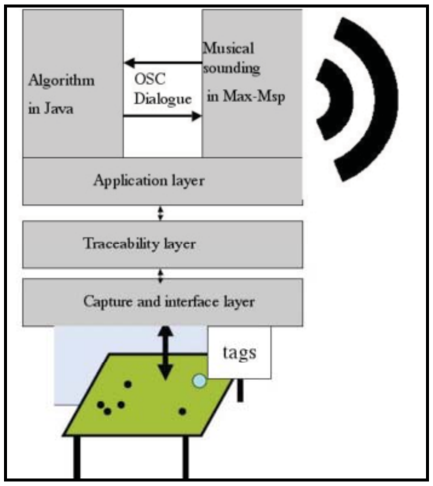


Figure 2. Software layers

Some musical experiments will now be described, with the final goal of having them working on the Tangisense Table. These experiments have been devised in two steps: a simulation one, where tags are simulated as objects on a computer window, and moved by mouse or tablet interaction. They serve as prototypes (hence the title of the article) for an implementation on Tangisense. The feasibility of the implementation of musical experiments with Tangisense is done, however as the definitive version of this table is not yet available, experiments have been conducted with individual tiles but large scale experimentation of the musical content and moreover evaluation of the sonic interaction design is definitely out of the scope of this article.

Two experiments clearly illustrate different ways where interaction with sounds may happen and are described in the core of this article, A third one is only evoked in the future directions section:

1 the first application is a classification of sounds using “baskets”. Depending upon the judgment of different users, a map can be made which corresponds to evaluated distances between sounds. The goal here is to provide navigation or selection spaces where a musician can get oriented palettes of sounds. It is especially intended for the use of textures, a musical domain hard to map.

2 the second experiment is a “MIDI-oriented” application where a cellular automata, activated by a metronome is made audible by triggering events linked to its cells.

3. ORIENTING A SONIC SPACE: THE BASKET TECHNIQUE AND ITS IMPLICATIONS.

The goal here is to provide a user interface that can help the musician in the navigation of sounds.

Two main approaches can be followed when dealing with the classification of musical sounds. The first one, referred as Music Information Retrieval (MIR), relies on an objective description of the sound based on a series of features computed on the audio signal [3][4]. The MIR community has been rapidly growing for a tens of years and a number of MIR tools are now available for experimentation[2].

The second strategy, which we adopt here, is that sound classification can be a perception-based approach. In this case, the classification does not rely anymore on mathematical descriptors but rather on results of psychoacoustic experiments. Typically in such experiments human listeners are asked to evaluate similarity between pair of sounds, group sounds in categories or assess distance between several groups of sounds [5][6]. These two complementary approaches for sound classification aims at addressing specific questions and give slightly different results.

3.1. baskets

Among the many possibilities of classifying sounds, there exist a technique that really fits a tangible paradigm: the basket (or packets) style which can be described as this: given n samples to be classified, one ask to m experimenters to separate them in different packets. The metaphor clearly is here to separate a set of objects (cards for example) in different packets. The main goal is to be able to create an estimated distance between pairs of objects, so that a multidimensional or cluster analysis can be applied.

[2]Marsyas http://marsyas.sness.net
CLAM http://clam-project.org
MirToolbox http://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

For one experimenter, one can say that the estimated distance is zero if 2 objects are put in the same basket, and one if put in different packets. If we use many experimenters we can sum up these individual experiments this way: we build a matrix where each cell represents the number of times objects have been put in different baskets (divided by the number of experimenters. This is the estimated mean distance between two objects.

MDS analysis is sometimes defined as a way to retrieve a geographical map from estimated distances. It is a dimension reduction very often used in sociological issues. The input is a matrix of dissimilarity; the output is a N-dimensional map that gives information on the proximity of these samples. We have used this technique to provide a classification of 10 sonic textures (textures can be defined as such: their long term appearance is uniform, while their short term revelation can be quite different according to time)., but the technique is the same for images, such as visual textures or even psychological cards. The goal is to find a classification where no a priori statement on the nature of the textures has been done.

3.2. The simulation

First we have devised a simulation written in Java, where a set of circles represent a sonic texture, and three spaces where the user has the task to put similar textures (Figure 3). There is a surrounding space, which can be considered as a place of uncertainty. The application is built in such a way that it is mouse-driven: when the mouse goes over a circle, this one turns from green to blue. When clicked this circle gets red and can be moved anywhere on the graphic plane. A last circle when moved indicates the end of the user session and a file is created which indicates for each circle its belonging to a box (1-3) or a free one (0).

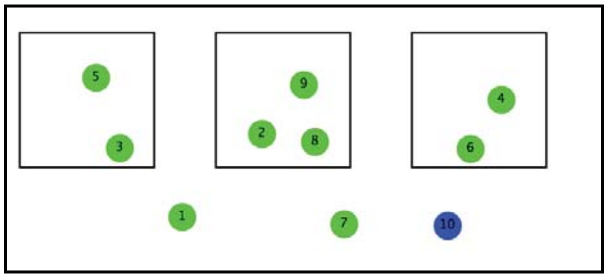


Figure 3. The computer screen as it appears to the user. Green circles become blue when browsed and red when clicked and dragged. Three baskets are symbolized by squares.

A Matlab program has been written that takes all the files corresponding to different users, and builds the dissemblance matrix by summing individual matrix, and then performs a multidimensional analysis (MDS). A MDS analysis requires the choice of the dimension of the reduction. Two is an obvious choice for a graphical representation but there are different techniques in exploration of data, which allow to estimate the best order [7]. The output of such a program (Figure 4) is a valuable representation of a set of sounds, especially when it comes to navigation in a sonic database. Moreover it can be used in performance, using Max patches similar to the SYTER interpolator), see references on the subject of 2D mapping in Bencina [8].

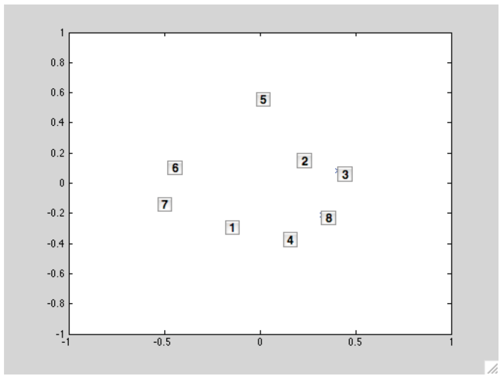


Figure 4. A map resulting to the reduction of data in a 2D space of a sonic texture database.

Another way to analyze dissemblance matrixes is cluster analysis: this way sounds are separated according to branches, giving place to clusters.

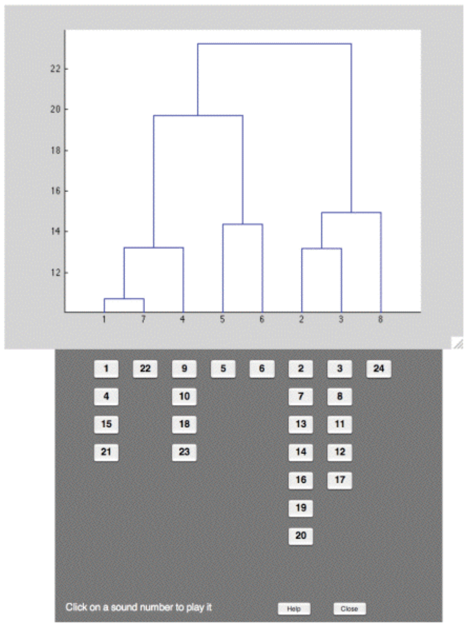


Figure 5. A tree resulting from a cluster analysis and the associated garland of sounds

To do this, we use a classical algorithm issued from the Mathworks Statistics toolbox, but we complement it by a special presentation of the results written by ourselves: for each branch of the clustering, we draw grapes of sounds corresponding to this branch (Figure 5). This way we have a clickable presentation of a sonic database where one can explore different branches, for example to select in an auditory way the best representatives of each category.

3.3. The tangible experiment

An obvious interest of tangible interfaces is to make the computer (and its mouse/screen components) disappear. We have put on the table pads that symbolise sounds (throwing them on the table in random order) plus one which gives the end signal and ask users to put them in three baskets, with the possibility for them not to use ones too hard to classify. As every object has a personal RFID, including the ultimate pad that when moved gives the signal for the end of one classification, the program is straightforward: instead of being mouse-driven, virtual objects are driven by the tags position and the program stays the same for what matters the writing of the file with positions, the MDS, cluster analysis and the garland of sounds.

One difference between the simulation and the tangible experience is the fact that we need to find a situation for actuating the sound that is represented by the pad. We have chosen the strategy of the sonic wall, or alternatively the sonic pillar: whenever we approach the pad near a wall, or inside a circle (the pillar) the object reveals its sound. Another way we are thinking of is to add another RFID on the pad, which can be activated by a switch. The "click" metaphor is back, but integrated in a "graspable" object.

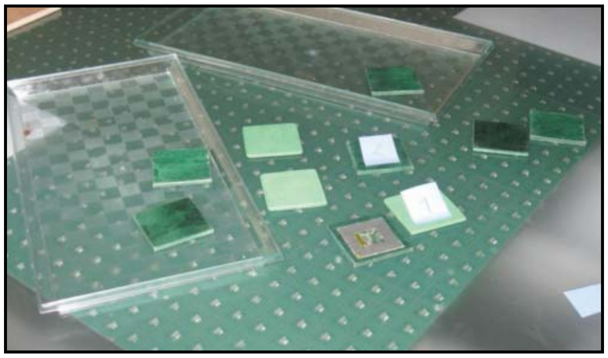


Figure 6. The Tangisense table with objects and baskets

4. INTERACTING WITH THE GAME OF LIFE.

We have previously seen a musical situation that can be considered as "out of real time". But most musical interactions, especially the art of performance are concerned with real-time experiments. One subset of it is the event driven musical flow, popularised by MIDI synthesisers and note oriented music.

We have decided to find an experiment that can be a good example of "user centred" application, which in our terms means that the computer is disappearing: here we will have a table with blinking lamps, and will bring the metaphor of a "sonic stethoscope" to drive a percussion generator. This is in the vein of cellular automata music [9][10], with a specific sonic interaction design flavour.

4.1. the parts

the game of life algorithm: this is a classical in "artificial life intelligence": typing "game of life" on a browser will give thousands of links, one of them being the original article from Gardner. [11] Many implementations are also available, but we have programmed an easy Java program for our own purpose.

Starting from an arbitrary (or not) matrix of ones and zeros (Figure 7), we consider an evolution process where the rules are taken from Conway. Conway's genetic laws are simple. First note that each cell of the checkerboard (assumed to be an infinite plane) has eight neighbouring cells, four adjacent orthogonally, four adjacent diagonally. The rules are:

1.Survivals. Every counter with two or three neighbouring counters survives for the next generation.

2. Deaths. Each counter with four or more neighbours dies (is removed) from overpopulation. Every counter with one neighbour or none dies from isolation.

3. Births. Each empty cell adjacent to exactly three neighbours is a birth cell.

This way we have an evolution process, where populations live, grows and eventually die (which is normally the end of the game). The interesting part is that depending upon the initial configuration, we may find gliders, oscillators with different rates possible, quasi-fractal patterns, and so on. Though not particularly new, this game is still exciting generations of students, mathematicians and curious people.

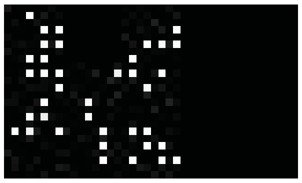


Figure 7. An instant in the game of life

The stethoscope: in this experiment every point of the game can be checked by a stethoscope-akin pad (Figure 8) and whenever a cell is activated, we will trigger a sound. As this process develops sounds related to an evolving data set, it is related to data sonification (the sonic equivalent of data visualisation). Obviously we may have two strategies: only use one trigger only when going from zero to one, or retriggering events when cells continue to light.

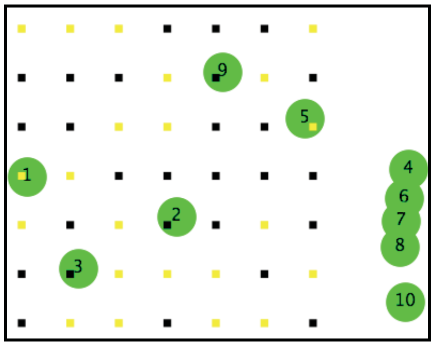


Figure 8. the “game of life”screen with the virtual pads

The sonic process: rhythm boxes are widely used, with samples or synthetic sounds, and clearly show us a path for a polyphonic stethoscope, where the individual sounds would refer to a specific pad number. Many Max patches can be used for this purpose. We particularly like the CNMAT implementations of modal synthesis[3].

4.2. The simulation.

On one part, we have a musical generator including a metronome and a sonic polyphonic generator. This is written in Max-Msp. The metronome part is indeed a basic one, while the polyphonic generator must take care that retriggering can occur. The communication between the musical process and the interaction one is bidirectional: as soon as one metronome tick is coming, an OSC message is sent, and the interaction scheme immediately sends OSC messages corresponding to note triggering.

The game of life itself is written in Java, and is an engine that runs for every click of the metronome. It must be initially given a size for the matrix, as well as the definition of the initial cells.

We have devised some tags, be them virtual or real, which must first inherit from some sonic properties. Hence we have to assign to each of the tags a sonic identity. This can be done in two ways: either we name the tags, and put an icon on them, or let them free of any assignment, in which case we must at some points get to a place where they are ”recharged” with a sound.

Then we can freely move these tags in places where cell will “pulse” the sonic material. This is of course a game, where many participants can play. It is also a big way to study the process of creativity, because each of these tags are traceable, so it is not only the result that is important, but also the process itself.

[3]Musical Applications of New Filter Extensions to Max/MSP http://web.media.mit.edu/~tristan/Papers/ICMC99_MaxFilters.pdf

4.3. The tangible experiment

The main point about using a tangible and traceable table is to establish a link between what the user’s action and what he gets as sounds. Basically the metaphor that is used is really important, especially if a visual feedback is provided. As leds are part of the TT Tangisense table, the game of life really takes its manifestation with light and the coordination with sound events is easy to draw (Figure 9).

Tags are put on table and the resulting sound corresponds to the lighting of a particular cell in a way similar to the simulation. Two domains are concerned by this experiment.
- Human-computer interaction. As an example the addition of LEDs on the table immediately connects the user on the experience itself: rather than having a displaced display, here we have a direct interconnection between one what sees and what one does and hear.
- Human-human interaction through technology. When computers disappear, the human being becomes free to share with others: this is a collaborative experience and the musical result, and its evolution greatly depends upon the “presence” of the performers, their reactivity, the sense of surprise they can bring: in a word they are musical performers.

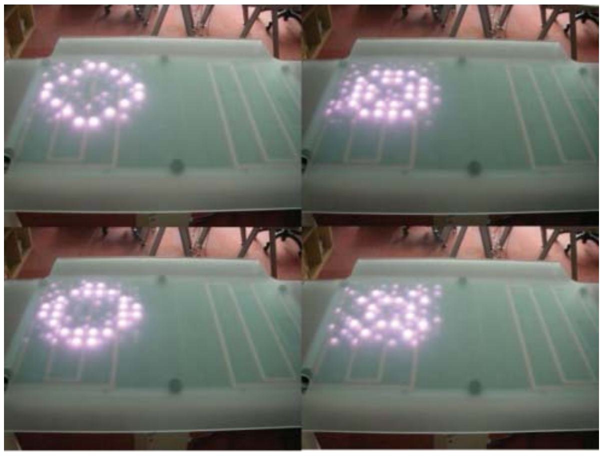


Figure 9. four ticks of the metronome drive 4 successive instants in the game of life on the Tangisense table.

5. FUTURE DIRECTIONS.

As we did see, it is possible to prepare musical experiments that can take benefit from the tangible and traceable aspect of our table. Nevertheless, musical performance largely uses gestures that are rapid, and it is interesting to see how far we can go with fast gestures.

We did modify a set of Max-Msp patches devised by Arfib et al [12] for the interactive use with graphical tablets, in order to make them controllable by a tag on the Tangisense interactive table.

We have chosen as a start to use a synthesis reminiscent to the photosonic one, where the X axis is mapped on a digital mixing of specific waves. In a way, it is navigation in a database, except that sounds are synthetic and governed by a single low frequency pitch. The Y axis is mapped on the coefficients of three filters, thereby providing a subtractive filtering to the preceding sound source. As an example filters vary from "i" to "oo' depending upon the Y axis.

With one tag we are able to drive this synthesis in real time, and delays are quite short and not noticeable. Of course the resolution of the table is less than the one of a Wacom tablet, but this works fine if one smoothes a little bit the flow of coordinates. The problem that remains is the way to trigger sounds without removing the tags, an easy game with graphic pen, an ergonomic question with RFIDs. We will in the future try many different experiments on that style of interaction.

The main future direction is the one of usability and we know this can lead to a vast discussion [13]. As stated in the article, we have demonstrated the feasibility of sonic interaction with a tangible and traceable table, but also situated the early stage of the experiments in a musical situation. These "user-centered" experiments and evaluation will follow, with a main concern on two things: the ergonomic aspect of the objects put on the table (until now this table has been used by the authors in an intuitive way, every experiment will have its own design, instructions and procedure); the evaluation aspect will be taken in account to measure the usability and improve the design and use of such a combination of software and hardware.

6. CONCLUSION

Many tangible tables rely on a video system and tags that are drawn on the surface of objects. Here we have a totally different technology, and a table initially not devoted to sonic applications. We have seen that it is possible to prototype musical experiments that can be linked to this table and make them run. More research has to follow to make comparisons with existing systems, but we also feel that our way of thinking sound, for example creating maps that can help the navigation or even the performance are innovative and can be a good framework for a community of musicians wanting to try the tangible and traceable aspect of interaction.

The present research work is supported by the "Agence Nationale de la Recherche, France" and the classification study is partly supported by Numediart, a research program funded by the "Région Wallonne, Belgium". Authors are national delegates for the SID (Sonic Interaction Design) IC0601 COST action.

7. REFERENCES

[1] Tangible Interfaces and Graspable Interfaces http://www.ehornecker.de/Tangibles.html

[2] Jordà, S., Geiger, G., Alonso, M. and Kaltenbrunner, M., "The reacTable: Exploring the Synergy between Live Music Performance and Tabletop Tangible Interfaces", in Proc. of the first international conference on "Tangible and Embedded Interaction" (TEI07). Baton Rouge, Louisiana http://mtg.upf.edu/reactable/publications

[3] G. Tzanetakis and P. Cook "Musical Genre Classification of Audio Signals", IEEE Transactions on Speech and Audio Processing , 10(5), July 2002

[4] G. Peeters., "A Large Set of Audio Features for Sound Description (Similarity and Classification)", in the CUIDADO Project. Tech. rep. IRCAM, 2004. P. 68.

[5] Saint-Arnaud N., Popat K., Analysis and classification of sound textures, in AAJCAI workshop on Computational Auditory Scene Analysis, 1995

[6] Vieillard S., Bigand E., Madurell F., Marozeau J., "The temporal processing of musical emotion in a free categorisation task", in Proceedings of the 5th Triennial ESCOM Conference, Hanover, Germany, 2003

[7] Martinez, W L, (2004) "Exploratory Data Analysis With Matlab," Chapman & Hall/CRC, 2004

[8] Bencina, R. (2005), "The Metasurface – Applying Natural Neighbour Interpolation to Two to Many Mapping," Proceedings of the 2005 Conference on New Instruments for Musical Expression (NIME'05), Vancouver.

[9] Miranda, ER. (1993) Cellular Automata Music; an interdisciplinary music project. Interface, Journal of new music research, 22:1

[10] Burraston, D., Edmonds, E., Livingstone, D. and Miranda, E. R. (2004). "Cellular Automata in MIDI based Computer Music", Proceedings of the International Computer Music Conference, Miami (USA).http://cmr.soc.plymouth.ac.uk/publications.htm

[11] Gardner, M. "The fantastic combinations of John Conway's new solitaire game "life"", Scientific American 223 (October 1970): 120-123 http://www.bitstorm.org/gameoflife/

[12] Arfib D., Couturier JM, Kessous L: "Design and use of some digital musical instruments". In Camurri A et Volpe G. (eds), Gesture-Based Communication in Human-Computer Interaction ; Lectures notes in Artificial Intelligence , volume LNAI 2915, pp 509-518. Springer-Verlag, 2004.

[13] Ljungblad S: Beyond Users: Grounding Technology in Experience http://www.viktoria.se/~saral/thesis.html

SOUND SEARCH BY CONTENT-BASED NAVIGATION IN LARGE DATABASES

Diemo Schwarz, Norbert Schnell
Ircam – CNRS STMS
Paris, France

ABSTRACT

We propose to apply the principle of interactive real-time corpus-based concatenative synthesis to search in effects or instrument sound databases, which becomes content-based navigation in a space of descriptors and categories. This surpasses existing approaches of presenting the sound database first in a hierarchy given by metadata, and then letting the user listen to the remaining list of responses. It is based on three scalable algorithms and novel concepts for efficient visualisation and interaction: Fast similarity-based search by a *k*D-tree in the high-dimensional descriptor space, a mass–spring model for layout, efficient dimensionality reduction for visualisation by hybrid multi-dimensional scaling, and novel modes for interaction in a 2D representation of the descriptor space such as filtering, tiling, and fluent navigation by zoom and pan, supported by an efficient 3-tier visualisation architecture. The algorithms are implemented and tested as C-libraries and Max/MSP externals within a prototype sound exploration application.

1 INTRODUCTION

For sound design, film and multi-media production, and musical creation, databases with instrumental or environmental sounds and sound effects are a vital resource. The number and size of commercially available sound effects databases, such as *Hollywood Edge*, *Sound Ideas*, loops collections, or community-driven on-line collections like *freesound* are growing steadily, [1] with rising network bandwith, growing harddisk capacity, and falling prices for storage and distribution media accelerating this growth even further.

From a certain scale onwards, the practical problem in the exploitation of these databases is no longer the question if a specific sound exists in the database, but how to find it. In a user survey conducted within the *SampleOrchestrator* project, professional film sound designers reported about their practice of collecting the soundtracks of the *rushes*, i.e. the raw, unedited footage shot during the making of a film, to augment their collection of ambiences, reaching the mark of 1 TB of sound data, and their difficulty of finding one special event in many long recordings with tools not adapted to such large sizes. They get by with the disciplined use of manually edited metadata, and orient themselves by the audio waveform displayed in a sound browsing application.

Our contribution to alleviating the problems of finding the right sound in a mass of unstructured recordings is interactive navigation with immediate audio feedback in a space of sound descriptors populated by sound segments. This approach greatly speeds up the usual workflow of hierarchical menu or search mask, result list, and play/stop buttons that put many mouseclicks between the user's idea of the sound and listening to appropriate contents of the database.

We propose to replace the menu- and list-driven interface with a 2D representation of a sound and category space. While navigating through the space, the sound segments close to the current position are immediately played. Playing is layered if movement is fast, so that large parts of the sound space can be explored rapidly. The strong interactivity enables the user to quickly understand the dimensions and areas of the presented space by probing sound snippets that are played as they are passed by.

This principle of navigation poses tougher requirements on the efficiency of the underlying algorithms, and on their scalability to very large databases. An accompanying article [18] concentrates on fast similarity-based search and the efficiency of low-dimensional embedding of the high-dimensional descriptor and category space with special attention to scalability. There, we chose and improved three algorithms that are also briefly described in section 3: the *k*D-tree search algorithm, the simulation of a mass–spring–damper (MSD) system, and the hybrid multi-dimensional scaling algorithm for dimensionality reduction.

We will then treat the aspects of efficient visualisation and novel methods of interaction with large sound databases for sound search by navigation, based on these improvements. To allow an efficient exploration of the space of sounds defined by the sound descriptors, we developed an architecture of the graphic model based on three coordinate spaces (model, world, device coordinate spaces) and on affine transformations or non-linear mapping between spaces (section 4.1). This architecture allows an easy integration of functionalities like zoom & pan and a novel mode *tile* formed by the subdivision of display according to categorial descriptors (section 4.2).

The algorithms and visualisation components are implemented and tested as C-libraries and Max/MSP patches de-

1 . Since its start in 2005, http://freesound.org almost doubled every year to 62701 sounds, 681 hours, 252 GB in Februray 2009.

scribed in section 5. The 229 sound descriptors used in the prototype application are analysed by an external program [9, 10] and loaded from SDIF files [2]. A limited subset of 24 descriptors can also be analysed in the prototype system itself, as described in [17].

2 RELATED WORK

The navigational approach to sound search has been inspired by interactive real-time *corpus-based concatenative synthesis* for musical creation [15] [13, 15, 14] as implemented in the CataRT system [16, 17]. This recent method permits to create music by selecting snippets of a large database of pre-recorded sound by navigating through a two- or higher-dimensional space where each snippet takes up a place according to its sonic character, such as pitch, loudness, brilliance. This allows to use a corpus of sounds as an instrument, or to compose the path through the corpus, or to resynthesise an audio file or live input with the source sounds, and thus to create novel harmonic, melodic and timbral structures. The selected units are concatenated and played, after possibly some transformations. The method can be seen as a content-based extension to granular synthesis providing direct access to specific sound characteristics.

Evidently, the high interactivity of the corpus as an instrument to create music could be immediately applied to the exploration of the sounds in the corpus with the aim of searching sounds. The existing 2D interface had to be extended to allow zoom&pan, and categories and class hierarchies had to be represented.

Related work in Music Information Retrieval start to apply spatial interfaces to content-based audio searches [3, 21], inspired by our work [16, 17], or independently [5, 6], or are concerned with the efficiency of nearest neighbour search [11] or the recent method of *locality-sensitive hashing* (LSH) [20].

3 SCALABLE ALGORITHMS

In order to optimize the exploration of large effects or instrument sound databases, the three algorithms briefly presented here solve recurrent problems in retrieval and visualisation in an efficient and scalable manner. Their functioning, improvements over the state of the art, and evaluation are described in detail in [18].

3.1 Efficient Nearest Neighbour Search with *k*D-Trees

While navigating the database, the problem of finding the sound segment closest to a target point x^t in the multi-dimensional descriptor space is solved efficiently by a *branch and bound* search algorithm based on the tree-structured index provided by the *k*D-tree.

The *k*D-tree represents a hierarchical decomposition of the descriptor space, and during search, whole subtrees are discarded from the search, by application of an elimination rule based on the farthest neighbour found so far. This removes a large amount of the distance calculations between vectors, resulting in a sublinear time complexity. Several variants of the algorithm are compared in [4], and it is argued that the best decomposition is along the hyperplanes orthogonal to the principal components, since it maximises the distance among the points in different subtrees and thus the probability that a subtree can be pruned.

The elimination of nodes is achieved by calculating the *split plane* of a node n defined by an orthogonal vector s_n and going through a point μ_n that is the mean of the node's elements. This plane is used in the vector-to-node distance function based on the dot product:

$$dist_{V2N}(x,n) = (x-\mu_n)/\sigma \cdot s_n \tag{1}$$

Ideally, s_n is the principal component vector of the node, but choosing it orthogonal to the axis of the dimension with the greatest variability results in an almost equally efficient search with less overhead for the decomposition.

The search algorithm uses a stack of nodes to be visited and the distance d of the target point x^t to the node's split plane in order for the elimination rule to prune child nodes when no vector closer than the current nearest neighbours can be found. An additional radius parameter r limits the returned nearest neighbours to lie within distance r from x^t. If $r=\infty$ all k nearest neighbours are returned. Note that both distance functions dist_{V2N} and the vector-to-vector distance

$$\mathrm{dist}_{V2V}(x,y) = (x-y)/\sigma \tag{2}$$

can include per-descriptor-weights in σ that balance the influence of each dimension in the search, even after the tree index is built.

The performance measurements in [18] show the logarithmic time complexity of search, linear complexity for building the tree, and the exponential influence of the number of dimensions. However, the highest single search time is just 2.2 ms for a database size of 10^6 10-dimensional points, and an initial overhead over linear search is quickly passed by with data sizes over 100. What's more, using PCA-based decomposition would reduce the dimensionality to the intrinsic number of dimensions, i.e. linearly dependent dimensions would not contribute to the complexity of the search.

3.2 Mass–Spring–Damper–Repulsion Model

Turning to the visualisation of sounds as points in a graphical interface, a useful model is the simulation of a system of masses connected by springs (or, more generally, by links). This model is the heart of the dimensionality reduction algorithm explained in section 3.3, but it can already be applied to an existing projection on two or three dimensions of a sound database for two purposes: First, it allows to interactively move displayed sounds with neighbouring sounds following, in order to organise the sound space. Second, the repulsion force that has been added in our implementation avoids overlapping points by pushing them apart.

The model is following MSD [7], implemented for MAX/MSP and PUREDATA. We chose an inert model for faster convergence and to avoid self-oscillating systems of masses. For the differential equations of the physics model and their implicit discretisation, see [7].

The available parameters are the nominal length of the links L, the stiffness parameter K, friction damping η, and viscosity damping μ. Repulsion takes place when the mass distance is lower than a threshold L_R and rises linearly up to R. Its effect can be seen in figure 1, where a cluster of overlapping points is distributed in space to reveal all sounds.

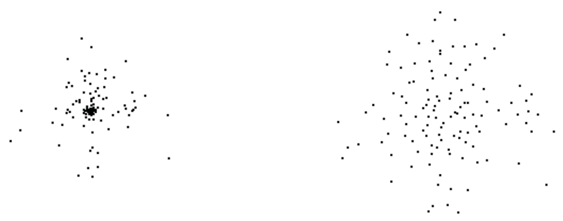


Figure 1. Effect of repulsion (right) on a cluster (left).

3.3 Hybrid Multi-Dimensional Scaling

For the low-dimensional visualisation of a high-dimensional space, the Chalmers algorithm [8] uses a mass–spring model, where the nominal spring lengths are given by the distance in the high-dimensional data space. The basic assumption is that the final minimal stress configuration, that the model will converge to, corresponds to a good layout, where points that are close in data space are also close in layout space. An additional advantage is that the algorithm is iterative such that the current configuration can be displayed to the user while the system converges.

Our improved hybrid algorithm first lays out a random sample of $n_s = \sqrt{N}$ points with a fully-connected mass–spring model to provide a good starting layout for faster convergence, then places the remaining points around their nearest neighbour from data space, taking advantage of the kD-tree, and finally lays all points out by running a mass–spring model with constant numbers of links to the nearest neighbours, and random links that change at each iteration.

The choice of n_s means that each iteration in the initialisation phase is linear, since a fully connected system takes $O(n_s^2) = O(N)$, while the placement is of complexity $O(N \log N)$ and the final iterations constant. Only few iterations are necessary until the total stress reaches a minimum.

4 INTERFACE

The search and musical synthesis of sounds from a large database of sounds and descriptors is similar to the exploration of data manipulating a graphical representation. This task is well described and much research on it has been done in the field of information visualization. Shneiderman and Plaisant [19] define the *mantra of information visualization* as:

Overview, Zoom and Filter, Details on Demand.

Interactive navigation in multi-dimensional data spaces requires either an exploration of individual descriptor dimensions, or a reduction of dimensionality to two or three to allow them being displayed: Methods of dimensionality reduction such as multi-dimensional scaling (MDS), principal component analysis (PCA) with the interactive integration of weights per dimension, linear mapping-by-example, and, if class labels are available, linear discriminant analysis (LDA), can all help to make high dimensional spaces available for interactive navigation.

We will concentrate in the following on new strategies for visualisation of sound databases for efficient search, including the integration of MDS and PCA, zoom & pan, a new display mode *tile*, and filtering options for categories. All these improvements rely on an efficient visualisation architecture that will be described first:

4.1 Visualisation Architecture

The graphics model implements a 3-tier architecture using three separate coordinate spaces (model, world, and device coordinates) and mappings and transformations between them as explained in the following:

4.1.1 Tier 1: Model Coordinates

The model coordinate space is the N-dimensional space of the descriptor data in raw descriptor coordinates (units like Hz, dB, linear amplitude, etc), populated by M units. The choice of D descriptors to use for display takes place in this space, as well as a preselection or inclusion/exclusion of U units to draw.

The model is thus represented as a matrix $model(U,D)$ which is a subset of the original unit data matrix $ud(M,N)$.

4.1.2 Tier 2: World Coordinates

The world coordinate space consists of the descriptor data projected to 2D (or 3D) and a colour scale index, normalised to $[0,1]$. It is represented as the matrix $world(U,D)$ and lists of labels for symbolic descriptors.

The selection of the unit to play being closest to the mouse pointer position takes place in world coordinates, and not in the full descriptor space of *model*.

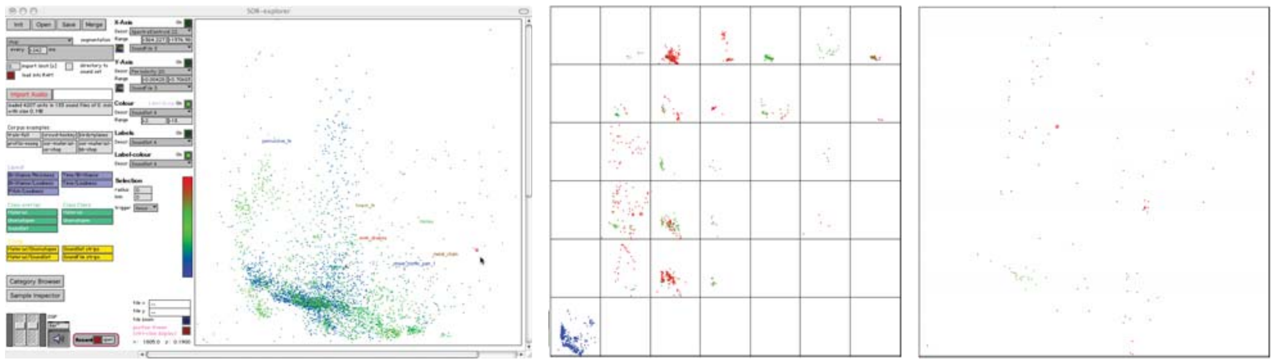


Figure 2. Screenshot of the sound navigator prototype in full view (left), the display canvas in *tile mode* (middle), and *tile zoom* (right), where the tile under the mouse is zoomed in fully.

4.1.3 *Tier 3: Device Coordinates*

This space is constituted of the raw coordinates of the output device in $device(U,D)$, i.e. screen canvas pixels and RGB colour values, or OpenGL screen coordinates, or other 3D space coordinates.

4.1.4 *Mapping A: Model $\rightarrow$ World*

Mapping A is only applied when the descriptor choice or the data changes. Usually, this mapping is a simple projection to the two selected descriptors. More advanced methods of dimensionality reduction, like PCA or MDS can be applied here. The inverse mapping A^{-1} converts a world coordinate $p(p_x, p_y)$ back into model coordinates, i.e. descriptor values.

The different modes of mapping A listed in the following, allow the transformation of the model space to the world coordinate space, or the inclusion of additional dimensions in the visualisation.

Standard Projection
Projection to two descriptor axes d_x, d_y of *model* and a colour scale d_c.

Transformed Projection
Dimensionality reduction by MDS or PCA from a mix of descriptors, d_x = principal component, d_y = secondary component, d_c = tertiary. However, any axis can also be a descriptor.

Tiled View
Can be used to display a view tiled into columns, rows, or cells by a categorical descriptor d^c such as class ID, sound set. Within each cell, a displacement descriptor d^d, scaled to $[0,1]$ is added to the category coordinate.

Pivot View *(not yet implemented)*
Pivot view uses one marked unit as the pivot p, which adds the distance between p and ud as an additional descriptor to choose from. The distance can be derived as a linear combination of descriptor distances, or be given by a distance matrix defined for a categorical descriptor. A further possibility is to place the pivot in the center of the plot, and display the remaining units in polar coordinates, with the distance mapped to the radius, and a selectable descriptor mapped to angle.

4.1.5 *Mapping B: World $\rightarrow$ Device*

The mapping from world to device coordinates produces the matrices in device coordinates (pixels and RGB colour values), label positions and colours.

The world-to-device mapping B keeps track of the current *view*: the rectangle $v(ll_x, ll_y, ur_x, ur_y)$ in world coordinates that is displayed (default: $v = (0,0,1,1)$), which is converted to a transformation matrix B including a scale factor s and a displacement vector $t(t_x, t_y)$, such that $device = world \cdot B^T$, or:

$$\begin{pmatrix} x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots \\ x'_n & y'_n & 1 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{pmatrix} \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ t_x & t_y & 1 \end{pmatrix} \quad (3)$$

This results, however, in 7 unused multiplications and 4 unused additions per point $p(x,y)$, and the additional memory for the expansion of the coordinate matrix by one column, so that the direct form is more efficient.

Mapping B provides the following functions that modify the 2D layout of the data points to ease browsing and inspection :

Zoom & Pan: Scaling and moving in the world coordinate space by changing the view v.

Magnifying glass using a sigmoid function mapping which is steepest around the current position.

The inverse mapping B^{-1} maps the input from controllers from device coordinates (pixel position of the mouse pointer) back to world coordinates (taking into account the current view), in order to perform selection.

4.2 Presentation and Interaction

The interface supporting sound search by navigation is shown in figure 2 (left). It allows to choose the descriptors used for the X- and Y-axes of the 2D space, which descriptor is displayed as a label for each sound or group of sounds, and the colour scale of the points representing a sound segment and the labels. Presets exist for the most useful settings of these choices. When moving the mouse pointer through the space, the sound segment close to the current position is immediately played. Playing is layered if movement is fast, so that large parts of the sound space can be explored rapidly. Because the input sounds would typically be segmented in rather short snippets (200–500 ms), the played sounds stop quickly, and thus also long recordings can be inspected by navigation. Other trigger modes exist that continue playing the segments of a given sound when the mouse does not move, in order to hear a recording entirely. Note that the time of a segment in the recording is also part of the selectable descriptors, allowing time-based browsing.

This strong interactivity enables the user to quickly understand the dimensions and areas of the presented space by an initial traversal, probing sound snippets that are played as they are passed by. Zooming in and out of the space and moving the view, together with the layout improvements of the MSDR model (section 3.2), allows to inspect the sound space of continuous descriptors in detail.

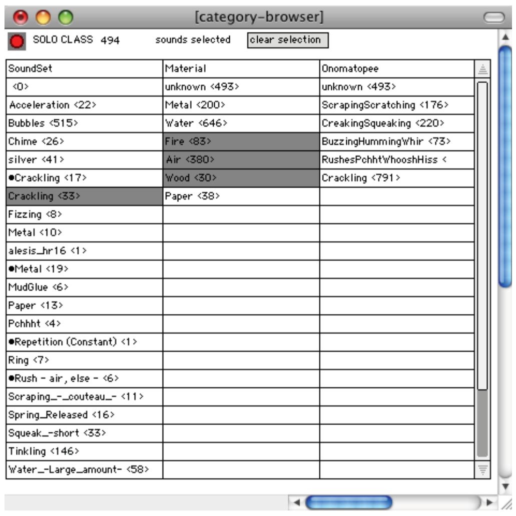
In order to explore the category descriptors and classes resulting from automatic classification, or groups of sounds defined by the user, two methods have been implemented: *tile* mode and the category browser.

Figure 2 (middle) shows the display tiled by two categories along the axes. Each tile contains a scaled version of the original descriptor space, but only with the units that are members of both of the two categories. The user can jump between the overall view and the view of a tile zoomed in fully, as illustrated in figure 2 (right).

The *category browser* (figure 3) allows to choose which units are active. To keep the overview of the whole sound distribution, inactive units are still displayed but are greyed out and not selectable. Combinations of categories can either be muted or soloed.

5 IMPLEMENTATION

The algorithms and interfaces described here are implemented as C-libraries and as externals within the FTM&Co extension library [12] at http://ftm.ircam.fr for Max/MSP and PureData, taking advantage of FTM&Co's advanced data structures such as matrices and dictionaries, and the algebraic, mapping, and statistical operators that work on these [1]. This allowed the rapid building of the prototype application to test the search-by-interaction paradigm.



SoundSet	Material	Onomatopee
<0>	unknown <493>	unknown <493>
Acceleration <22>	Metal <200>	ScrapingScratching <176>
Bubbles <515>	Water <646>	CreakingSqueaking <220>
Chime <26>	Fire <83>	BuzzingHummingWhir <73>
silver <41>	Air <380>	RushesPchhtWhooshHiss <
●Crackling <17>	Wood <30>	Crackling <791>
Crackling <33>	Paper <38>	
Fizzing <8>		
Metal <10>		
alesis_hr16 <1>		
●Metal <19>		
MudGlue <6>		
Paper <13>		
Pchhht <4>		
●Repetition (Constant) <1>		
Ring <7>		
●Rush - air, else - <6>		
Scraping_-_couteau_- <11>		
Spring_Released <16>		
Squeak_-short <33>		
Tinkling <146>		
Water_-Large_amount- <58>		

Figure 3. *Category Browser:* choose categories to view.

Differing from the CataRT system for real-time corpus-based concatenative sound synthesis [17], the sound explorer prototype does not need, and indeed can not keep all the sounds in memory. Instead, they are played back directly from disk, with a latency of about 15 ms for the harddisk access, which does not perturb navigation.

6 CONCLUSION

We described a system for efficient interactive sound search by navigation in databases of sounds and descriptors, based on three algorithms that are crucial for scalability to large databases, their improvements, performance and implementation, and novel concepts and methods for efficient visualisation and interaction in a 2D interface. First, the efficient logarithmic-time *k*D-tree search algorithm, where we added the limitation to a search radius, and weights for the descriptors. Second, the mass–spring–damper model for intuitive layout optimisation of points in a 2D interface, where we added repulsion. Third, the hybrid multi-dimensional scaling algorithm for dimensionality reduction for visualisation, based on the MSD model, where the use of the *k*D-tree speeds up the initialisation, allows more precise pre-placement, and thus faster convergence.

All three algorithms together make the paradigm of interactive sound search by navigation scalable to very large sound databases.

Then, we described a prototype application based on these algorithms and an efficient visualisation architecture, that allowed us to experiment a number of innovations and facilities in the user interface, such as class filters and a multi-grid visualisation, to organise search by navigation in large databases.

7 ACKNOWLEDGEMENTS

The research presented here is partially funded by the French National Agency of Research ANR within the RIAM project *Sample Orchestrator*. The authors would like to thank the project partners for their fruitful collaboration, and Joel Bensoam and Bram de Jong for invaluable assistance.

8 REFERENCES

[1] F. Bevilacqua, R. Muller, and N. Schnell, “MnM: a Max/MSP mapping toolbox,” in *New Interfaces for Musical Expression*, Vancouver, May 2005, pp. 85–88. [Online]. Available: http://mediatheque.ircam.fr/articles/textes/Bevilacqua05a/

[2] J. J. Burred, C. E. Cella, G. Peeters, A. Röbel, and D. Schwarz, “Using the SDIF sound description interchange format for audio features,” in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Philadelphia, USA, 2008.

[3] G. Coleman, “Mused: Navigating the personal sample library,” in *Proc. ICMC*, Copenhagen, Denmark, 2007.

[4] W. D’haes, D. van Dyck, and X. Rodet, “PCA-based branch and bound search algorithms for computing *K* nearest neighbors,” *Pattern Recognition Letters*, vol. 24, no. 9–10, pp. 1437–1451, 2003.

[5] S. Heise, M. Hlatky, and J. Loviscach, “SoundTorch: Quick browsing in large audio collections,” in *AES Convention 125*, San Francisco, CA, USA, Oct. 2008.

[6] ——, “Aurally and visually enhanced audio search with SoundTorch,” in *CHI EA ’09: Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems*. Boston, MA, USA: ACM, Apr. 2009, pp. 3241–3246.

[7] N. Montgermont, “Modèles physiques particulaires en environnement temps-réel : Application au contrôle des paramètres de synthèse,” MSc Thesis (DEA ATIAM), University of Paris 6, 2005.

[8] A. Morrison and M. Chalmers, “Improving hybrid MDS with pivot-based searching,” in *IEEE Symposium on Information Visualization*, 2003, p. 11.

[9] G. Peeters, “A large set of audio features for sound description (similarity and classification) in the Cuidado project,” Ircam – Centre Pompidou, Paris, France, Tech. Rep. version 1.0, Apr. 2004. [Online]. Available: http://www.ircam.fr/anasyn/peeters/ARTICLES/Peeters_2003_cuidadoaudiofeatures.pdf

[10] G. Peeters and E. Deruty, “Automatic morphological description of sounds,” in *Acoustics 08*, Paris, France, Jun. 2008.

[11] P. Roy, J.-J. Aucouturier, F. Pachet, and A. Beurivé, “Exploiting the Tradeoff Between Precision and CPU-time to Speed up Nearest Neighbor Search,” in *ISMIR*, London, UK, 2005.

[12] N. Schnell, R. Borghesi, D. Schwarz, F. Bevilacqua, and R. Müller, “FTM—Complex Data Structures for Max,” in *Proceedings of the International Computer Music Conference (ICMC)*, Barcelona, Spain, Sep. 2005.

[13] D. Schwarz, “Data-driven concatenative sound synthesis,” Thèse de doctorat, Université Paris 6 – Pierre et Marie Curie, Paris, 2004. [Online]. Available: http://mediatheque.ircam.fr/articles/textes/Schwarz04a

[14] ——, “Concatenative sound synthesis: The early years,” *Journal of New Music Research*, vol. 35, no. 1, pp. 3–22, Mar. 2006, special Issue on Audio Mosaicing.

[15] ——, “Corpus-based concatenative synthesis,” *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 92–104, Mar. 2007, special Section: Signal Processing for Sound Synthesis.

[16] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton, “Real-Time Corpus-Based Concatenative Synthesis with CataRT,” in *Digital Audio Effects (DAFx)*, Montreal, Canada, Sep. 2006.

[17] D. Schwarz, R. Cahen, and S. Britton, “Principles and applications of interactive corpus-based concatenative synthesis,” in *Journées d’Informatique Musicale (JIM)*, GMEA, Albi, France, Mar. 2008. [Online]. Available: http://mediatheque.ircam.fr/articles/textes/Schwarz08a

[18] D. Schwarz, N. Schnell, and S. Gulluni, “Scalability in content-based navigation of sound databases,” in *Proceedings of the International Computer Music Conference (ICMC)*, Montreal, Canada, Aug. 2009.

[19] B. Shneiderman and C. Plaisant, *Designing the User Interface*. Boston, USA: Pearson, 2005, ch. Information visualization, pp. 580–603.

[20] M. Slaney and M. Casey, “Locality-sensitive hashing for finding nearest neighbors,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 128–131, Mar. 2008.

[21] S. Streich and B. S. Ong, “A music loop explorer system,” in *Proceedings of the International Computer Music Conference (ICMC)*, Belfast, Northern Ireland, Aug. 2008.

ACCESSING STRUCTURE OF SAMBA RHYTHMS THROUGH CULTURAL PRACTICES OF VOCAL PERCUSSION

Luiz Naveda
IPEM – Ghent University
luiznaveda@gmail.com

Marc Leman
IPEM – Ghent University
marc.leman@ugent.be

ABSTRACT

In the field of computer music, melodic based forms of vocalizations have often been used as channels to access subject's queries and retrieve information from music databases. In this study, we look at percussive forms of vocalizations in order to retrieve rhythmic models entrained by subjects in Samba culture. By analyzing recordings of vocal percussions collected from randomly selected Brazilian subjects, we aim at comparing emergent rhythmic structures with the current knowledge about Samba music forms. The database of recordings was processed using a psychoacoustically inspired auditory model and further displayed on loudness and onset images. The analyses of emergent rhythmic patterns show intriguing similarities with the findings in previous studies in the field and put different perspectives on the use of vocal forms in music information retrieval and musicology.

1. INTRODUCTION

Beatboxing, Puirt-a-beul or *bols* are some of the examples of vocal percussion forms found in different cultural backgrounds. These practices generally make use of non-meaningful phonemes, which imitate instruments and often rely on onomatopoeia [1 , 2]. The information available on forms of vocal percussion account for examples distributed over different musical cultures and practices, which range from simple devices for musical learning to elaborated forms of performing art. Vocal percussion may have differentiated from melodic singing due to the necessity for more freedom in expressing rhythmical ideas. It is also possible that it originated from the combination of a vocal apparatus in human species and the necessity of rhythmical expression in all human cultures. However, what makes these forms of vocal percussion relevant to our study is the use of the voice as a seamless link between musical intentionality and acoustic energy. In this study, we use this link to access rhythmic intentionality and analyze rhythmic structure within Brazilian samba.

SMC 2009, July 23-25, Porto, Portugal

1.1. Vocal percussion

Cultural forms of vocal percussion have been rarely mentioned in traditional musicological research. More recently, the emergence of *hip-hop* in the cultural sector has shed light on the *beatboxing* form, which is only a modern and localized form of vocal percussion. Other examples found in the bibliography describe music forms in India (*bols*), song genres in Ireland (*Puirt-a-beul)*, pedagogic devices for conga teaching in Cuba [1] and verbal art in Africa [3]. So far, it seems that vocal percussion assumes diverse socio-cultural roles and importance, although only a small number of dispersed scholarly and non-scholarly records have showed this in detail.

In computer music some attention has been devoted to the potential use of vocalizations, in special pitch based vocalizations in western music contexts. The easy assessment of user's musical intentionality for music retrieval applications appears to be a central element in the "query-by-humming" approach. During the last years, a large number of publications and implementations have been produced in the field [see 4, for a review of algorithms]. Less noticeably, a small number of studies approached vocal percussion from this perspective. Kapur et al. [5] used the *beatboxing* as a mechanism to retrieve and analyze drum loops and their rhythmical structures. Nakano et al. [6] developed a similar approach by using native Japanese speakers as subjects, which demonstrates the application of the approach in phonetically different backgrounds. Kang & Kim [7] used vocal percussion for real-time animation of motion clips (dance animations). Although most of these studies aim at understanding how vocal queries relate with musical databases, in very few academic work the opposite is shown, namely, how vocal queries are related with subjects' conceptions of musical forms.

Heylen et al. [8] provides a study that uses spontaneous vocalizations to access subjects' musical conceptions or models. In this study, subjects were asked to sing along to several music pieces in different tonal contexts. The results show emergent major and minor tonal structures that resulted from spontaneous vocalizations. The use of vocal apparatus to retrieve the tonal evaluation from subjects is understood as a *corporeal articulation* in response to different tonal stimulus. The insightful turn of vocalizations into corporeal articulations is also a crucial concept in our study. It opens channels to less formalized responses to music, which include not only vocalizations but also body movement. The framework of embodied music cognition [see 9], in which this concept is developed, seem to adapt more naturally to the problems of musicological investigation developed in less-formalized music cultures. The music and dance traditions of the samba culture are examples of cultural artifacts dominated by informal learning and practices, in which our universe of study is delimited.

1.2. Vocal percussion in Samba

Samba music is generally described as having a binary meter music form accentuated in the second beat, and a rhythmic texture that is characterized by syncopated rhythms [10-13]. The music is only one component of an intricate complex, in which forms of dance, music, poetry, rituals and social relations develop mostly through in an informal context [14, 15].

Ziriguidum, balacobaco and *telecoteco* are some of the very common onomatopoeic expressions used in Brazil. They are not easily found in dictionaries, but they are intuitively linked with samba concepts, behaviors and culture. Expressions like these appear in thousands of internet references (mostly in Brazilian Portuguese): blog posts, magazines, books, little enterprises, dance clubs, restaurants and others. Surprisingly, there are almost no references on the use of vocal percussion as a common practice. To what extent is this practice common in Brazilian society? Which characteristics of this form of vocal percussion are consistently aligned with the musical repertoire? How do they reflect the conceptions of samba within the acculturated population?

In this preliminary study, we concentrate on the last two questions. By analyzing a database of vocal percussions recorded from randomly selected Brazilian subjects, we aim at providing the first images about this aspect of samba culture. The database of audio recordings was analyzed using a psychoacoustically inspired auditory model and further processed into loudness and onset images as described in the next sections. Section 2 describes the procedures used in the recordings and dataset. Section 3 explains the methods used to analyze and produce images of rhythmical content. In the Section 4, we show and discuss the results, rhythmical structures derived from the analysis and compare them with results from previous studies about samba music.

2. DATASET

The dataset used in this study is a growing database of vocal percussions recorded in Brazil, between 2008 and 2009. This database will be further complemented with questionnaires in order to provide better information about socio-cultural profiles of the subjects (not analyzed in this study).

In order to create conditions to access practical measures of how the practice is present in the universe of study, we opted to randomly choose subjects (passers-bys) in public spaces. The sessions took place in four different locations in Belo Horizonte (Brazil), in relatively quiet spaces (classrooms). The recordings were done with a professional digital recorder and high-quality microphones, using a sample definition of 44100 b/s at 16 bits (stereo), stored in SD cards.

First, the subjects filled out a brief contact form. No information about the study was provided before the recordings. During the second part of the experiment, the subjects were invited to perform samba rhythms, using their voice in spontaneously organized sequences. These sequences were registered in one single take. If the subject refused to perform or declared her/himself unable to perform the task, this was registered in a form. No training sessions or repeated takes were used in this experimental model.

3. METHODS AND ANALYSIS

3.1. Segmentation and normalization

The audio excerpts were segmented manually. The criteria for segmentation were selecting and extracting homogeneous excerpts that last a minimum of 4 or 8 beats. Each excerpt was then normalized at 0 dB (amplitude) and the channels merged into mono aural WAV files. Although female and male voice differences may have an influence on the overall auditory images, we opted to avoid any kind of spectral normalization or further processing aimed at normalizing these differences.

3.2. Analysis

3.2.1. Loudness images

During the first stage of our analysis, the excerpts were processed with a psychoacoustic inspired auditory model based on [16] and implemented as a windows executable. The auditory model simulates the auditory decomposition in the periphery of the auditory system, which results in 40 channels of loudness patterns obtained from a simulation of neural rate activity distributed over the audible spectrum [for more details see 16, p. 3514]. The loudness patterns were processed by the auditory model implementation at sample rate of 100 frames/second. All images were further resampled and normalized to a sample rate of 98 frames per musical beat. This procedure allows comparing and accumulating groups of auditory images and onset images, which originally had different absolute time durations.

In this study, we display auditory spectrum images with 4 or 8 beats lengths, depending on which illustration was used. When the auditory images are displayed with 8-beat length, 4 beat images will duplicate. When 4-beat images are displayed, only the first half of the 8-beat images is displayed.

3.2.2. Onset images

To avoid imprecise onset accumulations derived from a sum or a mean of loudness patterns, we applied a method for onset detection to each auditory image using an integrate-and-fire neural net based on an approach developed in [17] and implemented in IPEMToolbox [18][1]. In the sequence, onset images were quantized in 256 sample images (128 samples for 4-beat images), which means that onset attacks were integrated in 32 sample "slots" per musical beat. This procedure helps to slightly integrate deviated onsets in a single position when concatenating onsets in mean images. Mean onset images were produced from the integration of onsets in the same "slot" position (1/32 beat). Figure 2a show mean images of loudness, Figure 2b show mean image of onsets.

In order to visualize a single rhythm profile in the high portion of the auditory images, we summed up the high channels of the auditory images, as displayed in the Figure 3 (basically half of the channel distribution: 21:40 for high-frequency channels), which seem to be sufficient enough to demonstrate the propositions of section 4.2.3.

4. RESULTS AND DISCUSSION

The results consist of information regarding the overall profile of the database and a discussion about the selected images of the dataset, analyzed with the methods mentioned above.

4.1. Overview

We collected recordings from 55 subjects, which produced a database of 80 excerpts (1.5 excerpts/subject). A percentage of 5.4% subjects (3 subjects) was unable to perform the task. A percentage of 41% of the recordings was performed by female subjects and 59% by male ones. The actual database is composed of 45 excerpts with 4-beat length and 35 excerpts with 8-beat excerpts. We extracted the mean BPM values for all excerpts manually. The BPM list has a normal distribution (Kolmogorov-Sminov test, alfa = 0.05) with mean 102.09 and standard deviation 18.5. The minimum BPM found was 59.6 while the maximum was 184.4.

4.2. Analysis of auditory and onset images

4.2.1. First images

Figure 1 (a, b, c), displays 3 phases of the generation of loudness/onset images. Firstly, (1a) the first loudness images are extracted and resampled. They are followed by an (1b) onset image derived from the loudness images and (1c) the same onset image quantized to 32 samples/beat. The quantization of onset attacks provides a better integration of onsets for each 1/32nd segment of the musical beat.

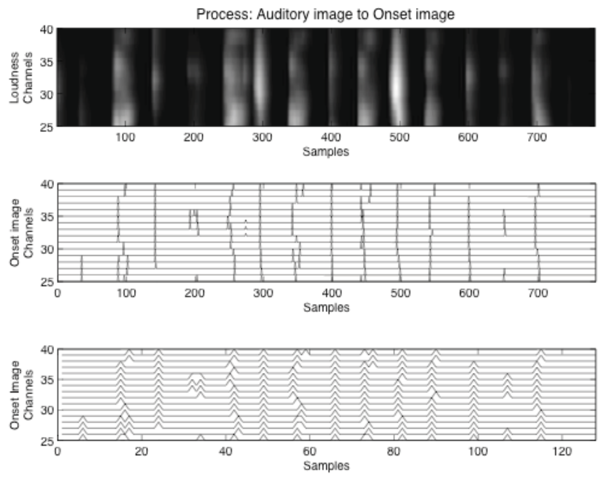


Figure 1(a) - First loudness image of an excerpt of vocal percussion performed by a female subject. Degrees of gray represent normalized loudness (white:black=0:1); **(b)** - onset image at 98 samples/beat; **(c)** - quantized onset image at 32 samples/beat.

4.2.2. Mean loudness and onset images

We calculated the mean loudness image and a mean onset image for all excerpts. Figure 2a and 2b summarize loudness and onset channels for all vocalizations.

1 IPEM toolbox is available at http://www.ipem.ugent.be

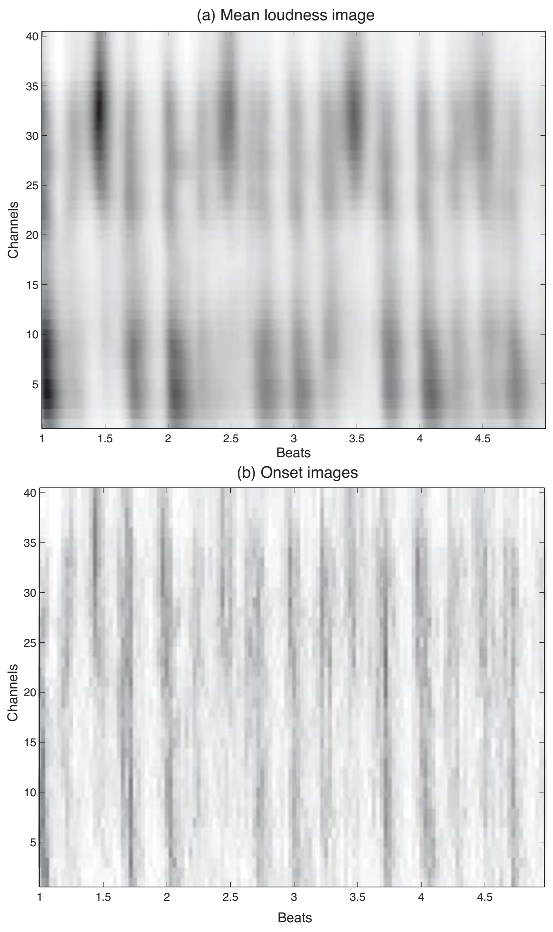
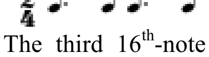


Figure 2. 4-beat mean **(a)** loudness image and **(b)** onset image for all excerpts.

In both Figures 2a and 2b we observe the isolation of low and high-pitched onsets, denoted by a separation between high and low frequency patterns. These two distinct layers seem to exhibit periodic rhythmic patterns along time. The high portion of the spectra shows a characteristic tatum layer of samba, composed of ¼-beat onset profiles (16th notes in a 2/4 bar). The low portion of the spectra exhibits a punctuated rhythm (1) that stresses each beat mark, which can also be found in several samba recordings (such as the one detailed in Figure 3). An auditory inspection of the sound database confirms the relevance of these observations and that low-frequency patterns are usually performed with consonants *t* or *d*, followed by a very low sound. Of course, transients of these consonants spread over the high spectrum, which makes the separation of onsets between spectral regions questionable (see discussion in section 4.2.4).

(1) 

The third 16th-note onsets of each beat (at the 0.5 position of the beat) are often accentuated in the high-spectra. The second 16th-note onsets are often less clearly defined in both high and low portions of the auditory spectra. Double onset peaks verified some of the 2nd, 3rd and 4th 16th-notes in the onset image (Figure 2b) seem to indicate the presence of different groups of microtiming behaviors, not evident in the loudness images.

4.2.3. *Tatum and microtiming*

The tatum layer is defined as the lowest level of musical metrical hierarchy, and normally detected through the fastest rhythmical figures observed in the rhythmical texture. This musical layer is found in high-pitched patterns of instrumental samba ensembles and commonly represented in musical notation as isochronous 16th note figures. Figure 3, extracted from [19], displays a similar onset analysis applied to a commercial recording of samba music (*Ela veio do lado de lá* – Benito Di Paula, 1975). In this case, our results seem to confirm the same structure observed in commercial samba recordings.

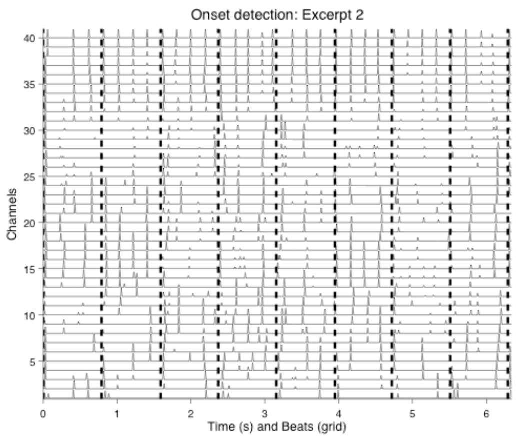


Figure 3. Onset analysis of a commercial samba recording. Traced lines indicate beat points.

Recent studies have examined the role of microtiming in the induction of “groove” in Brazilian music. Gouyon [20] studied the microtiming in the samba using a database of commercial recordings. Lindsay & Nordquist [21] analyzed microtiming using commercial recordings and recording of individual samba performers. Gerischer [22] studied the groove and microtiming using field recordings of Brazilian percussion groups. All these studies demonstrated the existence of a systematic anticipation of the 4th and 3rd onsets in groups of four 16th-notes (1 beat).

Figure 4 displays the analysis of microtiming between the peaks of the onset images (high auditory spectrum, channels 21:40). The top graph shows the accumulated onset image (1/32 beat) and the bottom graph displays the sample difference between each onset and the following one.

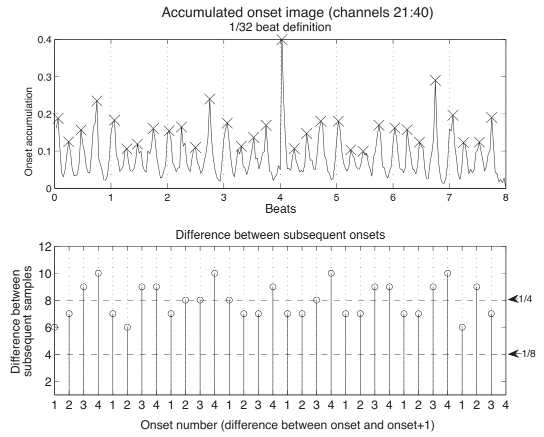


Figure 4. Microtiming analysis of the higher onset image (1/32 beat samples). The bottom graph shows the distance (in samples) between onsets. Traced lines represent 1/8 and 1/4 isochronous sample divisions of the beat.

The patterns displayed in the bottom graph of Figure 4 show that the intervals between 1st and 2nd, 2nd and 3rd 16th-notes (second graph, numbers 1 and 2, horizontal axis) are normally shorter than others. 3rd and especially 4th intervals (numbers 3 and 4, horizontal axis) are often larger than the mathematical isochronous ¼ rule of the tatum layer. The pattern of deviations verified by our database of vocal percussions is strikingly consistent with studies mentioned above, especially with the results displayed in [22, p. 105], [20, p. 200] and [21, p. 26]. Although such an observation seems to be a trivial confirmation of findings of previous studies, the fact that untrained subjects could re-enact precise microtiming structures verified in musical stimuli is worth paying attention to.

4.2.4. *Rhythmical hypotheses*

By subsuming the hypothetical voicing possibilities of our observations, we can create a collection of rhythmical motives that may provide explanations for the rhythmic possibilities found in the results. Figure 5 demonstrates these possibilities in traditional music notation:

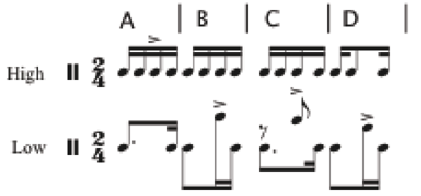


Figure 5: Hypothetical rhythmic motives (A, B, C, D) extracted from our observations. This collection of motives is not exhaustive and other variations may occur.

The motive A is the most obvious structure found in the loudness/onset images because it directly mirrors the high-low spectral layers. If we accept that the third 16th-note is linked with the low voice, a low-high-low pattern (B) will appear in the low voice, while tatum layer is still maintained. However, this strong accent may also represent a third mid-frequency voice (C) due its slight distribution over the mid frequencies. In this case, the mixture of channels is supposed to be an inevitable consequence of the monophonic condition of human voice. Finally, the same condition in B may be exclusively pertinent to the low voice, which gives rise to a syncopated pattern (D) in the high spectra. The last hypothesis of course implies that the existence of a constant tatum layer in the higher portion of the auditory spectrum must be rejected. After all, these hypotheses seem to indicate that syncopated or contrametric lines are not strongly represented by the models that lie underneath the vocal percussion practices. It is also possible that syncopation onsets may exist, but they are so dispersed and distributed over the time span that their presence is masked by the commetric forces or by the average procedure in the calculation of mean images.

The results raise very intriguing questions. Samba musicians, composers dancers, researchers and expert listeners seem to agree that syncopation, polyrhythmic and contrametric content are the most salient characteristic of Samba music [14, 23, 24]. It was expected that subjects would likely perceive and store traces of highly syncopated stimuli and, therefore, be able to perform entrained patterns with the same characteristics. However, what we observed is that vocal percussion patterns do not show observable syncopation at relatively accessible macro time and mid spectrum, but demonstrate a surprising consistency at microtiming level. In which part of the chain stimuli-subject-vocalization was the syncope filtered?

The answer to this question will take us further than this preliminary study. Improved methodologies and the incorporation of other modalities may be necessary to better represent the samba complex as a systemic structure. Nevertheless, the results seem to demonstrate the richness of the vocal practices in providing elements for both information retrieval and cognitive modeling investigation.

5. CONCLUSION

In this paper we analyzed a database of vocal percussion in samba culture. We aimed at understanding how rhythmic models emerged from the vocalizations and how they articulate with models of the samba music. The study shows that the ability to perform this practice seem to be spread over the group of subjects of this study, and that rhythmic models are relatively consistent throughout mean images of the loudness and onset patterns. Rhythmic patterns derived from these images show 3 important observations: that (1) rhythmic patterns are similar with overall samba music characteristics and (2) strikingly entrained at microtiming level. The syncopation priority of samba music (3) seems to be absent in vocal percussion representations but a better methodology may clarify the presence of syncopated onsets.

In terms of perspectives for the SMC and MIR fields, this study shows that vocal queries are not simple copies of models of musical intentionality, but may emerge from complex interactions between acoustic stimuli/environment and other modalities such as corporeal engagement with music (dance). However, the precision of vocal queries may reach levels of performance comparable with professional musicians, with the advantage that it situated within the context of the ubiquitous and normalized medium of human vocal emissions. Future work must include a better pattern detection methodology, a robust statistical verification, analyses of cross-modal interactions and a more systematic investigation over the profile of subjects, metrical content and microtiming.

6. ACKNOWLEDGEMENTS

The authors wish to thank the subjects who contributed to our study and the anonymous reviewers for the suggestions. This research is supported by a grant from Ghent University (Belgium), and partially support by CAPES (Brazil).

7. REFERENCES

[1] M. Atherton, "Rhythm-Speak: Mnemonic, Language play or Song?," in *International Conference on Music Communication Science*, Sidney, 2007, p. 15.

[2] D. Stowell and M. Plumbley, "Characteristics of the beatboxing vocal style," Tech. Rep. C4DM-TR-08-01, Dept. of Electronic Engineering, Queen Mary, University of London, 2008.

[3] L. Hunter, "Transformation in African Verbal Art: Voice, Speech, Language," *Journal of American Folklore,* vol. 109, pp. 178-192, 1996.

[4] R. Typke, F. Wiering, and R. Veltkamp, "MIREX symbolic melodic similarity and query by singing/humming," *MIREX 2006,* p. 19, 2006.

[5] A. Kapur, M. Benning, and G. Tzanetakis, "Query-by-beat-boxing: Music retrieval for the DJ," 2004, pp. 170–177.

[6] T. Nakano, J. Ogata, M. Goto, and Y. Hiraga, "A Drum Pattern Retrieval Method by Voice Percussion," *Joho Shori Gakkai Kenkyu Hokoku,* vol. 2004, pp. 45-50, 2004.

[7] K. Kang and D. Kim, "Synthesis of Dancing Character Motion from Beatboxing Sounds," 2007, pp. 216-219.

[8] E. Heylen, D. Moelants, and M. Leman, "Singing along with music to explore tonality," in *9th International Conference on Music Perception and Cognition*, Bologna, 2006.

[9] M. Leman, *Embodied Music Cognition and Mediation Technology*: Mit Press, 2007.

[10] J. Chasteen, "The prehistory of Samba: Carnival Dancing in Rio de Janeiro, 1840-1917," *Journal of Latin American Studies,* vol. 28, pp. 29-47, Feb. 1996.

[11] M. E. Mariani, "African influences in Brazilian Dance," in *African Dance*, K. W. Asante, Ed. Asmara: Africa World Press, 1998, pp. 79-98.

[12] M. Salazar, "Batucadas de Samba," *Rio de Janeiro: Lumiar Editora,* 1991.

[13] C. Sandroni, *Feitiço decente: transformações do samba no Rio de Janeiro, 1917-1933*: Jorge Zahar Editor: Editora UFRJ, 2001.

[14] M. Sodré, "Samba, O Dono do Corpo," *Rio de Janeiro: Codecri,* 1979.

[15] P. Fryer, *Rhythms of Resistance: African Musical Heritage in Brazil*: Pluto, 2000.

[16] L. Van Immerseel and J. Martens, "Pitch and voiced/unvoiced determination with an auditory model," *The Journal of the Acoustical Society of America,* vol. 91, p. 3511, 1992.

[17] L. Smith, "Onset-based sound segmentation," *Advances in neural information processing systems,* pp. 729-735, 1996.

[18] M. Leman, M. Lesaffre, and K. Tanghe, "A toolbox for perception-based music analyses," in *IPEM-Dept. Of Musicology, Ghent University,* Ghent, 2001.

[19] L. Naveda and M. Leman, " A cross-modal heuristic for periodic pattern analysis of Samba music and dance," *Journal of New Music Research (in press),* 2009.

[20] F. Gouyon, "Microtiming in "Samba de Roda"—Preliminary experiments with polyphonic audio," in *XII Simpósio da Sociedade Brasileira de Computação Musical* São Paulo, 2007.

[21] K. Lindsay and P. Nordquist, "Pulse and swing: Quantitative analysis of hierarchical structure in swing rhythm," *The Journal of the Acoustical Society of America,* vol. 122, p. 2945, 2007.

[22] C. Gerischer, "O suingue baiano: Rhythmic feeling and microrhythmic phenomena in Brazilian percussion," *Ethnomusicology,* vol. 50, pp. 99-119, 2006.

[23] M. E. Mariani, "A Portrayal of the Brazilian Samba Dance with the Use of Lab Analysis as a Tool for Movement Analysis," University of Wisconsin, Madison, 1986.

[24] B. Browning, *Samba: Resistance in Motion*: Indiana University Press, 1995.

AUTOMATIC MANIPULATION OF MUSIC TO EXPRESS DESIRED EMOTIONS

António Pedro Oliveira
Centre for Informatics and Systems, University of Coimbra
apsimoes@student.dei.uc.pt

Amílcar Cardoso
Centre for Informatics and Systems, University of Coimbra
amilcar@dei.uc.pt

ABSTRACT

We are developing a computational system that produces music expressing desired emotions. This paper is focused on the automatic transformation of 2 emotional dimensions of music (valence and arousal) by changing musical features: tempo, pitch register, musical scales, instruments and articulation. Transformation is supported by 2 regression models, each with weighted mappings between an emotional dimension and music features. We also present 2 algorithms used to sequence segments.

We made an experiment with 37 listeners who were asked to label online 2 emotional dimensions of 132 musical segments. Data coming from this experiment was used to test the effectiveness of the transformation algorithms and to update the weights of features of the regression models. Tempo and pitch register proved to be relevant on both valence and arousal. Musical scales and instruments were also relevant for both emotional dimensions but with a lower impact. Staccato articulation influenced only valence.

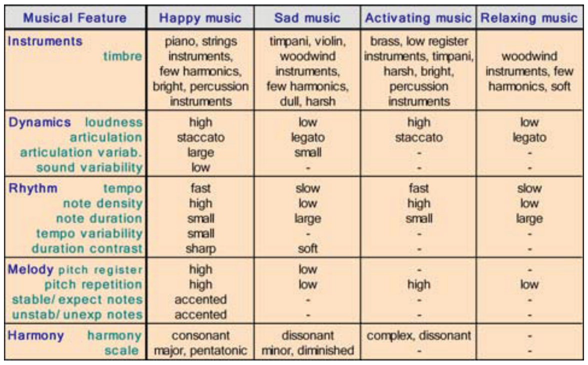
1. INTRODUCTION

The automatic production of music that expresses desired emotions is a problem with a large spectrum for improvements. The importance of developing systems with such a capability is evident to the society. Every context with a particular emotional need can use systems of this kind to accomplish its objectives. However, only recently there has been a great improvement in this area. Scientists have tried to quantify and explain how music expresses certain emotions [3, 4, 11]. Engineers have developed systems with the capability of producing music conveying specific emotions [7, 16, 17] by using the knowledge acquired by scientists.

We are developing a computational system used to produce music expressing desired emotions (section 3), grounded on research of Music Psychology and Music Computing (section 2). In this work we are focused on the transformation of music and improvement of the weights of features of the regression models used in the control of the emotional content of music; we also present sequencing algorithms (section 4). We made an experiment with 37 listeners that emotionally labeled 132 musical segments: 63 of transformed and 69 of non-transformed music (section 5). The analysis of the data obtained from this experiment and the update of the regression models are present in section 6. Section 7 makes some final remarks.

2. RELATED WORK

Our work involves research done in Music Psychology and Music Computing. The comprehension of the influence of musical features in emotional states has contributed to bridge the semantic gap between emotions and music. We are interested in the effect of structural and performance features on the experienced emotion [10]. We analyzed several works [2, 3, 4, 6, 7, 11, 15] and made a systematization of the relevant characteristics to this work that are common to four types of music: happy, sad, activating and relaxing (Figure 1).



Musical Feature		Happy music	Sad music	Activating music	Relaxing music
Instruments	timbre	piano, strings instruments, few harmonics, bright, percussion instruments	timpani, violin, woodwind instruments, few harmonics, dull, harsh	brass, low register instruments, timpani, harsh, bright, percussion instruments	woodwind instruments, few harmonics, soft
Dynamics	loudness	high	low	high	low
	articulation	staccato	legato	staccato	legato
	articulation variab.	large	small	-	-
	sound variability	low	-	-	-
Rhythm	tempo	fast	slow	fast	slow
	note density	high	low	high	low
	note duration	small	large	small	large
	tempo variability	small	-	-	-
	duration contrast	sharp	soft	-	-
Melody	pitch register	high	low	-	-
	pitch repetition	high	low	high	low
	stable/ expect notes	accented	-	-	-
	unstab/ unexp notes	accented	-	-	-
Harmony	harmony	consonant	dissonant	complex, dissonant	-
	scale	major, pentatonic	minor, diminished	-	-

Figure 1. Characteristics of happy, sad, activating and relaxing music

This systematized knowledge is used by works aiming to transform the emotional content of music. These works have developed computational systems with a knowledge-based control of structural and performance features of pre-composed musical scores [2, 7, 16, 17]. Winter [17] built a real-time application to control structural factors of a composition. Pre-composed scores were manipulated through the application of rules with control values for different features: mode, instrumentation, rhythm and harmony. REMUPP [16] is a system that allows real-time manipulation of features like tonality, mode, tempo and instrumentation. Pre-composed music is given to a music player and specific music features are used to control the sequencer (e.g., tempo); to employ filters and effects (e.g., rhythmic complexity); and to control synthesizers (e.g., instrumentation). Livingstone and Brown [7] implemented a rule-based system to affect perceived emotions by modifying the musical structure. This system is grounded on a list of performance and structural features, and their emotional effect. The KTH rule-based system for music performance [2] relates performance features to emotional expression. This system is grounded on studies of music psychology.

3. COMPUTATIONAL SYSTEM

The work presented in this paper is part of a project that intends to develop a system that produces music expressing a desired emotion. This objective is accomplished in 3 main stages: segmentation, selection and transformation; and 3 secondary stages: features extraction, sequencing and synthesis. We are using 2 auxiliary structures: a music base and a knowledge base. The music base has pre-composed MIDI music tagged with music features. The knowledge base is implemented as 2 regression models that consist of relations between each emotional dimension and music features.

Aided by Figure 2 we will describe with more detail each of these stages. Pre-composed music of the music base is input to a segmentation module that produces fragments. These fragments must as much as possible have a musical sense of its own and express a single emotion. Segmentation consists in a process of discovery of fragments. This process occurs from the beginning of the piece by looking to each note onset with the higher weights. An adaptation of LBDM [1] is used to attribute these weights according to the importance and degree of variation of five features: pitch, rhythm, silence, loudness and instrumentation). Resulting fragments are input to the module of features extraction that obtains music features used to label these fragments which are then stored in the music base.

Selection and transformation are supported by the same knowledge base. Selection module intends to obtain musical pieces with an emotional content similar to the desired emotion. These pieces are obtained from the music base, according to similarity metrics between desired emotion and music emotional content. This emotional content is calculated through a weighted sum of the music features, with the help of a vector of weights defined in the knowledge base for each emotional dimension. Selected pieces can then be transformed to come even closer to the desired emotion. Transformation is applied in 6 features (section 4). The knowledge base has weights that control the degree of transformation for each feature. Produced pieces from the transformation module are sequenced in the sequencing module. This module changes musical features with the objective of obtaining a smooth sequence of segments with similar emotional content. This sequence is given to a synthesis module, which uses information about the MIDI instruments and timbral features to guide the selection of sounds from a library of sounds.

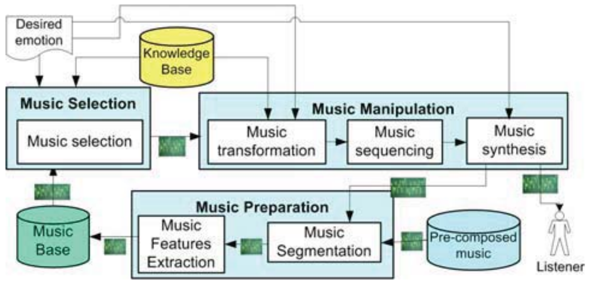


Figure 2. Diagram of our computational system

4. METHODS

This section presents the methods being used to transform music, sequence music and improve regression models.

4.1. Transformation of musical segments

Music transformation algorithms have the objective to approximate the emotional content of selected music to the desired emotion. By knowing the characteristics common to different types of music (section 2) we developed six algorithms that transform different features: tempo, pitch register, musical scale, instruments, articulation and contrast of the duration of notes. These algorithms start by calculating the emotional distance between the emotional content of the selected music and the desired emotion. The value of this distance is divided by the value of the weight of the feature being transformed. The value that results from this division corresponds to the amount of increase/decrease we need to make on the feature to approximate music emotional content to the desired emotion. Next paragraphs explain how this increase/decrease is made by each algorithm on the MIDI file.

The algorithm used to transform tempo obtains the original tempo of the music (in beats per minute) and then increases/decreases the note onsets and duration of notes.

The algorithm that transforms pitch register transposes up/down music by a specific number of octaves to increase/decrease valence/arousal. We choose octaves, because they are the intervallic transformation more consonant [14] with audible repercussion in the frequency spectrum. This is done by adding positive/negative multiples of 12 to the pitch of all the notes.

The algorithm that transforms musical scales finds the original scale of the MIDI file and selects a target scale according to emotional tags to be defined for each scale. Then, it finds the pitch distance relative to the tonic for each note in the original scale. If this distance is not found in the target scale, it finds the closer pitch distance to this distance that is present in the target scale and changes the pitch of the note, according to the new distance. For instance, suppose we want to transform from a ragha madhuri (pitch distances of 4, 5, 7, 9, 10 and 11 semitones to the tonic) to a minor gipsy scale (pitch distances of 2, 3, 6, 7, 8 and 11 semitones to the tonic). A note distant 4 semitones from the tonic in the ragha madhuri scale would have its pitch decreased by one semitone to be distant 3 semitones from the tonic in the gipsy scale. This happens because the interval of 4 semitones is not present in the minor gipsy scale. We used a group of 27 twelve-tone scales[1]. We chose this group and not others because it has a higher variety of number of notes and intervals: scales have between 2 and 7 notes and intervals vary from 1 to 7 semitones.

The algorithm used to transform instruments obtains original MIDI instruments and selects new instruments according to the emotional tags of each timbre. These tags are calculated through a weighted sum of audio features (e.g., spectral dissonance and spectral sharpness), with the help of a vector of weights defined in the knowledge base for each emotional dimension.

The algorithm that transforms normal to staccato articulation decreases the duration of all notes by a specific percentage. If we consider 75%, notes with a duration of *X* would have a new duration of *X*-*X**0.75.

We also have an algorithm that increases the contrast between the duration of notes. It increases/decreases the duration of longer/shorter notes according to a degree of transformation (*k*) and the duration of notes expressed in beats:

```
if (beat < 1 && beat > 0)
    adjustment = -(11/170) * (170 - beat*170)
else if (beat < 2 && beat > 1)
    adjustment = (4/200) * (beat*200)
else if (beat > 2)
    adjustment = (2/200) * ((beat-1)*200) +  6
newDuration = oldDuration + k*adjustment /400
```

This algorithm is based on an equivalent algorithm of KTH [2]. It was not yet tested.

4.2. Sequencing of musical segments

Music sequencing algorithms have the objective to obtain a smooth sequence of segments with similar emotional content. To date we only have algorithms for rhythmic matching and to do fade in and fade out of volume.

The algorithm of rhythmic matching intends to match the rhythm of previous segment(s) to the rhythm of next segment. This objective is accomplished by matching the mean of the interonset intervals (IOI) of the notes of the Nth segment (IOI_N) with the mean of the IOI of the notes of the N-1 (IOI_N-1) segments according to the pseudo-code:

```
adjustment = IOI_N-1 / IOI_N
if (IOI_N-1 > IOI_N)
adjustment = -IOI_N / IOI_N-1
change = 0
for firstNote to pnultimateNote
  change = change + (onsetNextNote – onsetThisNote) –
  (onsetNextNote – onsetThisNote)*adjustment
  onsetNextNote = onsetNextNote – change
end for
```

The algorithms of fade in and fade out are used to smooth the transitions between segments, respectively, by gradually increasing the volume of the starting segment and decreasing the volume of the finishing segment.

4.3. Improvement of the regression models

We intend to improve the regression models in order to control transformation algorithms, as well as to allow a better control of selection algorithms. Updated weights take into account new weights obtained in this experiment and weights obtained in previous experiments [8, 9] and are calculated according to the following formula: 0.1*Weights [8] + 0.6*Weights [9] + 0.3*NewWeights. The values of 0.1, 0.6 and 0.3 for each experiment were obtained according to the number of music files and listeners used in each of the experiments.

5. EXPERIMENT

In our experiment we started by using our system to randomly select 14 MIDI files of film and pop/rock music expressing different emotions. These MIDI files were subject to a segmentation process that produced a set of 746 segments, from which a set of features were automatically extracted. From the set of segments and with the help of the set of features, our system selected 132 segments that were expected to express different emotions and last between 10 and 15 seconds. We used small segments to try to have only one emotion expressed in each segment and to allow a fine-grained correlation between musical features and emotions. 63 of the 132 segments were changed in only one of the following features: 12 in tempo, 10 in pitch register, 27 in musical scale and 14 in articulation. The other 69 segments were not subject to transformations.

We made an online questionnaire[2] to allow anonymous people to emotionally classify the 132 segments. 37 different listeners labeled 2 emotional dimensions for each

[1] http://papersao.googlepages.com/musicalscales

[2] http://student.dei.uc.pt/~apsimoes/PhD/Music/smc09/

segment with values selected from the integer interval between 0 and 10. We obtained standard deviations of 1.76 and 1.85, respectively, for the answers for valence and arousal. These deviations were computed first between listeners, and then averaged over segments. Obtained labels were related with the extracted features for both transformed and non-transformed music. Feature selection algorithms were used (best-first and genetic search [18]) over the 476 extracted features from non-transformed music to select the features emotionally more relevant.

6. RESULTS

This section presents the results of the emotional impact of transformed (tempo, pitch register, musical scales, instruments and articulation) and non-transformed features in order to update the relations and their weights for each feature of the regression models

6.1. Tempo

We transformed 6 segments by accelerating in 50% and slowing down in 30% their tempo, obtaining 3 versions for each one: fast, normal and slow tempo. For each of the resulting 6 groups of 3 segments, we correlated the tempo of each version with the emotional data obtained in the experiment. Table 1 presents the correlation coefficients.

Group	1	2	3	4	5	6	Mean
Valence(%)	94	96	91	-7	62	100	75
Arousal(%)	100	98	98	-16	97	-36	74

Table 1. Correlation coefficients between tempo and valence and arousal for the 6 groups of segments

The expected high positive coefficients were confirmed by most of the results. However, the fourth group of segments obtained small negative coefficients for both valence and arousal, and the sixth group for arousal. This may be explained by the presence of an imperceptible transformation, because of the presence of very long notes (> 4seconds) on the original segment. A higher percentage of acceleration and slowing down of the original segment would be needed. The result of 100% for valence in the sixth group is not very reliable because the answers were very close: 3.5, 3.3 and 3.2. Emotional transformations contributed to an increase of 0.4/0.2 in valence/arousal with changes from low to normal tempo, and an increase of 1/0.8 in valence/arousal with changes from normal to high tempo.

6.2. Pitch register

We transformed 5 segments by transposing them up and down two octaves, obtaining 3 versions for each one: high, normal and low register. For each of the resulting 5 groups of 3 segments, we correlated the register of each version with the emotional data obtained in the experiment. Table 2 presents the correlation coefficients.

Group	1	2	3	4	5	Mean
Valence(%)	79	100	15	60	33	57
Arousal(%)	-39	-63	-98	-92	-62	71

Table 2. Correlation coefficients between pitch register and valence and arousal for the 5 groups of segments

Generally speaking, the increase of register correlates positively with valence and negatively with arousal. A more detailed analysis of the results in groups 3 and 5 showed lower correlation for valence, which revealed that the change from normal to high register contributes to a decrease in valence. From the analysis of the mean pitch of the segments, we can observe that the increase in register affects valence positively only till we have mean values of MIDI pitch around 80, whilst higher values contribute to a decrease in valence. We assisted to a similar situation in this first group for arousal: values of MIDI pitch higher than 80 seem to do not affect the arousal of music. Emotional transformations contributed to an increase of 2/-0.6 in valence/arousal with changes from low to normal register, and an increase of 0.7/-0.4 in valence/arousal with changes from normal to high register.

6.3. Musical scales

We transformed 1 segment by changing the original major scale to other 27 musical scales (subsection 4.2). We used feature selection algorithms in the process of finding the features that best characterize the emotional variation when changing the scale. The number of semitones in scale, the difference between successive intervals of the scale, the spectral dissonance and spectral sharpness with, respectively, the weights: -0.17,-0.15, 0.18 and -0.14 were important for valence. The number of semitones in scale, the difference between successive intervals of the scale, the spectral dissonance and stepwise motion with, respectively, the weights:-0.19, -0.07, 0.14 and 0.24 were important for arousal. Table 3 presents the correlation coefficients between the most discriminant features and emotional dimensions for the considered 27 versions of the segment.

Musical feature	**Valence(%)**	**Arousal(%)**
Spectral dissonance	46	31
Tonal dissonance	28	-
Timbral width	-32	-
Sharpness	-34	-20
Stepwise motion	24	33
Melodic thirds	-34	-18
Number of semitones in scale	-40	-23
Difference of successive intervals in scale	-28	-16
Correlation coefficient	**61**	**45**

Table 3. Correlation coefficients between musical features and valence and arousal for the 27 versions of the segment

6.4. Instruments

We changed the instruments of the 69 segments not subject to other types of transformations: tempo, register, scales and articulation. We tried to have each of the General Midi 1 (GM1) instruments [12] present in, at least, 1 of these 69 segments, in order to analyze the emotional impact of every GM1 instrument.

Table 4 presents the correlation coefficients between audio features and the valence and arousal for the 69 segments. From Table 4, we can infer that instruments are essentially relevant to the arousal, being spectral dissonance, timbral width and spectral sharpness relevant features in the emotional analysis of the sound/timbre of instruments. We found that violin, string ensembles, choirs and piccolo contribute to low valence; and percussion instruments contribute to high valence/arousal.

Musical feature	Valence(%)	Arousal(%)
Spectral dissonance	28	72
Timbral width	-	54
Tonal dissonance	19	27
Sharpness	-	44

Table 4. Correlation coefficients between musical features and valence and arousal for the 69 segments

6.5. Articulation

We transformed 14 segments by changing their articulation to staccato, obtaining 2 versions for each one: normal and staccato. We correlated the articulation of the 28 versions with the emotional data obtained in our experiment and found that the change from normal to staccato articulation is 40% correlated with the increase of valence and has no impact in arousal.

6.6. Emotional impact of non-transformed features

We used experimental data from the 69 segments not subject to transformations to analyze the emotional impact of non-transformed features. Tables 5 and 6 present the features emotionally more discriminant for each emotional dimension. We obtained correlation coefficients of 79% and 85% (tables 7 and 8), respectively, for valence and arousal, using these features.

6.7. Update of the regression models

After the analysis of the emotional effect of tempo, pitch register, musical scales, instruments and articulation on the transformed music and of the emotional effect of the more important features on the non-transformed music we updated the weights of features of the regression models according to the formula present in subsection 4.3.

In tables 7 and 8, we compare the weights as well as the correlation coefficients of previous experiments [8, 9] with the weights and correlation coefficient obtained in this experiment. The fifth features emotionally more discriminant are present with a bold font with higher size.

7. CONCLUSION

We successfully tested the effectiveness of algorithms of music transformation. Change of tempo was positively related to both valence and arousal. Change of pitch register was positively related to valence and negatively related to arousal. The presence of semitones in musical scales was found to be an important feature negatively related to valence. Spectral dissonance, timbral width and spectral sharpness were found to be important features for instruments and are positively related to arousal. Staccato articulation was found to be positively related to valence.

These results and correlation coefficients of features emotionally more relevant served to update the weights of features of the regression models that have been used to control the emotional changes made by the transformation algorithms. This experiment was grounded on previous experiments [8, 9].

Musical feature	Corr. Coeff.(%)
Staccato incidence	57
Number of unpitched instruments	53
Note density	52
Average note duration	-50
Average time between attacks	-50
Overall dynamic range	48
Variability of note duration	-46
Melodic fifths	45
Pitch variety	43
Note prevalence of closed hi-hat	42
Rhythmic looseness	41
Percussion prevalence	40

Table 5. Features emotionally more discriminant for valence

Musical feature	Corr. Coeff.(%)
Variability of note prevalence of unpitched instruments	70
Percussion prevalence	69
Note density	66
Number of unpitched instruments	58
Staccato incidence	56
Importance of loudest voice	55
Variation of dynamics	48
Note prevalence of snare drum	47
Overall dynamic range	46
Variability of note prevalence of pitched instruments	45
Note prevalence of bass drum	45
Note prevalence of closed hi-hat	43

Table 6. Features emotionally more discriminant for arousal

Musical feature	***W* [7] 16files 53list.**	***W* [8] 96files 80list.**	***NewW* 69files 37list.**	***Update W***
Average note duration	**0**	**-0.30**	**0**	**-0.18**
Chromatic motion	0	-0.16	-0.12	-0.13
Importance bass register	**-0.35**	**-0.17**	**-0.24**	**-0.20**
Initial tempo	**0**	**0.60**	**0.4**	**0.48**
Muted guitar fraction	**0**	**0.27**	**0**	**0.16**
Note density	0	0.09	0	0.05
Note prevalence of marimba	0	0.17	0	0.10
Num. unpitched instruments	0.20	0.11	0	0.08
Orchestral strings fraction	**0**	**-0.43**	**-0.14**	**-0.30**
Polyrhythms	-0.26	0	-0.16	-0.08
Saxophone fraction	0.26	0.16	0	0.13
Staccato incidence	0	0	0.17	0.05
String ensemble fraction	-0.46	0	-0.19	-0.10
Variability of note duration	-0.26	0	-0.37	-0.14
Key mode	0	-0.13	0	-0.08
Spectral sharpness	0	0	0.24	0.07
Spectral volume	0	0	0.28	0.08
Correlation coefficient	**97%**	**81%**	**79%**	**-**

Table 7. Weights and correlation coefficients for features emotionally more discriminant for valence

Musical feature	***W* [7] 16files 53list.**	***W* [8] 96files 80list.**	***NewW* 69files 37list.**	***Update W***
Average note duration	**-0.57**	**-0.44**	**0**	**-0.32**
Avg. time between attacks	0	-0.12	0	-0.07
Importance bass register	-0.22	0	-0.21	-0.08
Importance of high register	-0.57	0	0	-0.06
Initial tempo	**0**	**0.31**	**0.4**	**0.31**
Note density	**0.21**	**0.47**	**0.22**	**0.36**
Number of common pitches	0	0	0.19	0.06
Percussion prevalence	0.27	-0.26	0	-0.12
Primary register	0	-0.12	-0.25	-0.16
Repeated notes	0.16	0.11	0	0.08
Strength strong. rhythm. pulse	0	0	0.22	0.07
Variability of note duration	-0.26	-0.17	0	-0.13
Variability note prevalence of unpitched instruments	0	0.20	0.12	0.15
Spectral dissonance	0	0	0.27	0.08
Spectral sharpness	**0**	**0.30**	**0.22**	**0.25**
Spectral similarity	**0**	**-0.35**	**-0.21**	**-0.27**
Average dynamics	0	0.08	0	0.05
Correlation coefficient	**99%**	**88%**	**85%**	**-**

Table 8. Weights and correlation coefficients for features emotionally more discriminant for arousal

8. REFERENCES

[1] Cambouropoulos, E.. ”The local boundary detection model (LBDM) and its application in the study of expressive timing”, *International Computer Music Conference*, 17-22, 2001.

[2] Friberg, A., Bresin, R. and Sundberg, J., “Overview Of The KTH Rule System For Musical Performance”, *Advances in Cognitive Psychology*, 2:145-161, 2006.

[3] Gabrielsson, A. and Lindstrom, E., “The Influence Of Musical Structure On Emotional Expression“, *Music and emotion: Theory and research*. Oxford University Press, 223–248, 2001.

[4] Juslin, P., “Communicating emotion in music performance: A review and a theoretical framework” *Music and Emotion: Theory and Research,* 309-337, 2001.

[5] Leman, M., Vermeulen, V., De Voogdt, L., Moelants, D. and Lesaffre, M., “Prediction of musical affect using a combination of acoustic structural cues”, *Journal of New Music Research*, 34(1):39-67, 2005.

[6] Lindstrom, E., *A Dynamic View of Melodic Organization and Performance*, PhD thesis, Acta Universitatis Upsaliensis Uppsala, 2004.

[7] Livingstone, S. and Brown, A., “Dynamic response: real-time adaptation for music emotion.” *Australasian Conf. On Interactive Entertainment*, 105–111, 2005.

[8] Oliveira, A. and Cardoso, A., ”Towards bi-dimensional classification of symbolic music by affective content”, *Int. Computer Music Conf.*, 2008.

[9] Oliveira, A. and Cardoso, A., ”Modeling Affective Content of Music: A Knowledge Base Approach”, *Sound and Music Computing Conference*, 2008.

[10] Scherer, K. and Zentner, M., “Emotional effects of music: Production rules”, *Music and emotion: Theory and research*, 361–392, 2001.

[11] Schubert, E., *Measurement and Time Series Analysis of Emotion in Music*, PhD thesis, University of New South Wales, 1999.

[12] Selfridge-Field, E., *Beyond MIDI: the handbook of musical codes*, MIT Press, 1997.

[13] Sethares, W., *Tuning, timbre, spectrum, scale*, Springer, 2005.

[14] Vassilakis, P., Auditory Roughness as a Means of Musical Expression, *Perspectives in Systematic Musicology*, 12:119-144, 2005.

[15] Wassermann, K., Eng, K., Verschure, P and Manzolli, J., “Live soundscape composition based on synthetic emotions”, *IEEE Multimedia,* 10:82-90, 2003.

[16] Wingstedt, J., Liljedahl, M., Lindberg, S. and Berg, J. “Remupp: An interactive tool for investigating musical properties and relations”, *New Interfaces For Musical Expression*, University of British Columbia, 232–235, 2005.

[17] Winter, R., *Interactive music: Compositional techniques for communicating different emotional qualities*, Master’s thesis, University of York, 2005.

[18] Witten, I., Frank, E., Trigg, L., Hall, M., Holmes, G., Cunningham, S. ”Weka: Practical machine learning tools and techniques with java implementations.”, *Int. Conf. on Neural Info. Processing*, 192-196, 1999.

NEW TENDENCIES IN THE DIGITAL MUSIC INSTRUMENT DESIGN : PROGRESS REPORT

Paulo Ferreira-Lopes
UCP - E.Artes / CITAR
Porto - Portugal
Karlsruhe University of Music
Karlsruhe – Germany
pfl@zkm.de

Florian Vitez
Karlsruhe University of Music
Karlsruhe – Germany
florian.vitez@t-online.de

Daniel Dominguez
Karlsruhe University of Music
Karlsruhe – Germany
daniel_dt@web.de

Vincent Wikström
Karlsruhe University of Music
Karlsruhe – Germany
evererterez@gmx.de

ABSTRACT

This paper is a progress report from a workgroup of the University of Music Karlsruhe studying Music Technology at the *Institut für Musikwissenschaft und Musikinformatik* (Institute for Musicology and Music Technology). The group activity is focused on the development and design of computer-controlled instruments – digital music instruments [5].
We will describe three digital music instruments, havedeveloped at the *Computer Studio*. These instruments are mostly unified by the idea of human gesture and human interaction using new technologies to control the interaction processes. At the same time they were built upon the consciousness of musical tradition taking a fresh approach on everyday objects.

1. META_SONIC.IN PLACE

meta_sonic.in place is an interactive sound installation in which sounds can be triggered by color recognition. The aim of the work is to create a new, heightened experience of sound and space [10]. This installation was first used in the courtyard at Wedinghausen Monastery (www.kloster-wedinghausen.de) as part of the 12th *Internationaler Kunstsommer Arnsberg* (Arnsberg International Summer of Art - [www.kunstsommer-arnsberg.de].

Figure 1. picture of the opening[1]

Visitors are able to interact and influence the sounds directly; they can decide on how to combine the sound material and can change its characteristics. They do this by using either a colored scarf or a glow stick as they walk around the courtyard, exploring and discovering what the place has to reveal (Fig. 2).

meta_sonic.in place is structured in two levels : a hardware and a software level. The hardware part consists of a PA (5 loudspeakers, subwoofer, mixer), which provides for the acoustic irradiation, an audio interface, a MacBook Pro and a camera. The camera detects the movement of the visitors as part of the interaction process. The software level is a standalone application developed in Max/Msp and Jitter [7]. With the software application we can manage the sound material and control the live electronic sound processing.

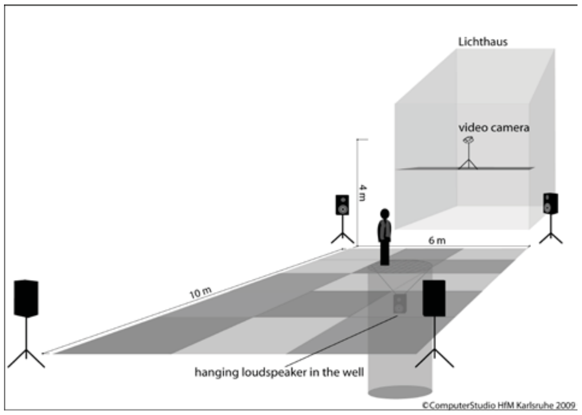

Lichthaus
video camera
4 m
6 m
10 m
hanging loudspeaker in the well
©ComputerStudio HfM Karlsruhe 2009

Figure 2. schema of the realization in the courtyard at Wedinghausen Monastery

[1] - Picture by Julian Stratenschulte

1.1. Interaction Process

In the monastery courtyard, there was a raster comprising 15 fields (Fig.3), defined by the given structure of the venue. Nine of these were active at one time. The order of the fields changed every 30 minutes (inside of the application) as predefined rule. In addition, audio effects could be triggered in four of the nine active fields. Three central fields, one of which was the over the shaft of the former courtyard well, were always active.
It is also possible to use a different number of fields and a different raster. The key factor is that the user is provided with a matrix, which is easily accessible.

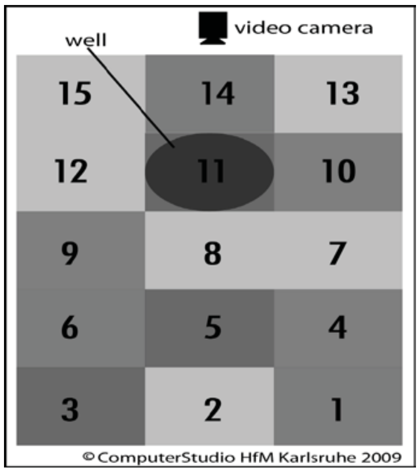


Figure 3. instance of possible array of fields

1.2. Color Recognition

The location is surveyed by a camera positioned several meters above the area. The video frame is then divided into different fields, which are programmed to detect different colors. A color filter programmed into Jitter reads the video signal and identifies the colors red, blue and green in defined image sections (fields). If one of the defined colors is filtered in the image section, a sound is activated or an existing sound is modified.

1.3. Audio

The sound library establishes a connection between the past and the present of the place where the installation is applied. Around 100 samples in stereo and mono were produced for the installation at Wedinghausen Monastery, including religious music, which had been changed to unfamiliar sound, ambient noises and synthetically generated sounds. One of the most important objectives in this work consists on the spatial and historical integration between the real space and the sound art object – the installation.
Various stereo pairs could be created using four loud speakers. So the spatialization was effected by using a combination of individual loud speakers, while the fifth loud speaker in the well only played back mono sound files. Each field was assigned nine sounds chosen for compositional reasons, which were played back in a predefined order. Once a sound was activated, it could be modified in four different ways:

- ***drunK71:*** *which consists of the effects pitch and stereo-delay. Short delay-times (ca. 200ms) give the pitched sound different nuances.*
- ***FREQdelay**: comprises two different effects that work independently of each other: by random-process, either a down and upward glissando or a stereo-delay is applied. The signal thus produced is played parallel to the original: if the original signal comes from one side, the modified sound is heard from the other side.*
- ***ALLout**: is an effect especially for the field above the well. If a color is identified here, the sample triggered comes to the fore while the outer loudspeakers are muted.*
- ***MONroom**: like ALLout, it is especially designed for the "well field". If this effect is activated, the signal is also routed to the outer loudspeakers.*

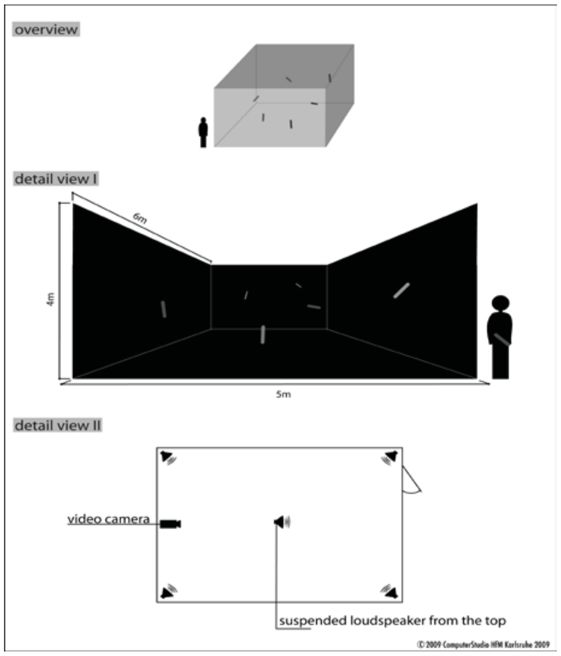

overview
detail view I
6m
4m
5m
detail view II
video camera
suspended loudspeaker from the top
© 2009 ComputerStudio HfM Karlsruhe 2009

Figure 4. Possible indoor realization

1.4. Results

The concept aims to create an installation which is as ambivalent as it possibly can be, the core of which is like an instrument which sounds different depending on where and by whom it is played, but which is always played in a similar way. The sound material, the spatialization and the type of effects can be adjusted to fit each individual location where the installation is applied. However, the visitor should always take on the role of interpreter [2] in addition to his or her classic function as listener.

The scene created in Wedinghausen transformed the monastery courtyard into a stage with the old well as its center. As this was the first location where the concept was realized, the well became the inspiration for the title *meta_sonic.in place*. The project is named after the Ancient Roman Meta Sudans, a fountain with a conical 'meta' in its center, a construction, which also marked the place where racing chariots would turn in a Roman circus.

2. CYCLEONIUM

The *cycleonium* is a computer based music instrument [5][8]. Every day objects like a bicycle and a bottle, and also a propeller are put in a new context and form the base of the sound design. They lose their conventional function consequently and have to be considered as indispensable parameters of this instrument. The physical and mechanical work that is required to make this sound-machine sound basically doesn't differ from traditional musical instruments. For instance, a guitar: it sounds, respectively do we perceive it acoustically, if the strings are excited or the corpus is oscillated somehow or other. So the player expends energy and translates it into the instrument. The user and viewer of the *cycleonium* is to be made aware of that, in principle, musical instruments are nothing else than objects that only operate with a certain energy expenditure. The energy flow that is emerged thereby plays here a primary role. Therefore we use the word fluxus, but more in a semantically significance (lat. flux, fluidum = flow) than artistic aesthetics [4]. Any kind of sounds require energy to sound. They have self-energy indeed, but it first has to be excited. Otherwise every sound is only a sounding abstraction (they aren't a perpetual motion machine) . The player of the *cycleonium* produces, amongst others, kinetic energy that excites a propeller that simultaneously blows air on the aperture of a bottle: a pitched sound is audible.

2.1. Description

The *cycleonium* consists basically of four pieces: a bicycle (Fig. 5 and Fig. 6), a bottle and a propeller that form the hardware and a computer for software-based sound processing (live-electronics).

The bicycle is rebuilt in such a way that it resembles a stationary bicycle. In this way the propeller can be powered whereby the emerging air flow hits the angle of the aperture of the bottle. Likewise in a flute, a periodical oscillating air column that is perceivable as a tone pitch is produced in the corpus of the bottle [1]. The intensity of the sound depends on the speed of the propeller and consequently on the expended force of the player. The blades of the propeller are fixed up in a way that they turn down and cause a rattling noise if the speed is too low. Thus the interplay of the expended force of the player with the speed of the propeller and thereby with the sound of the bottle is directly audible and explicit. The breath of the player and the sound of the chain and arbor of the bicycle are as equally important as the sound of the bottle itself. The produced sounds are amplified and real-time processed.

The live-electronics take place in Max/Msp [7] on a Laptop. The sounds can be pitched, transposed, filtered, recorded and played and processed with delay or reverb. Each Parameter can be controlled with a MIDI-controller. The following microphones are used:

bottle:	*electret microphone*
breath:	*dynamic microphone*
chain/arbor:	*condenser microphone (cardioid)*

For instance, over the keynote of the bottle can be formed an overtone scale with 15 partials or more that are controllable individually. The spatialization is realized with delays, which can be filtered, pitched and assigned to the respective audio-outputs discretely in real-time [3]. To bring the bottle in the best possible position to the airflow one can adjust it in height, distance and angle to the propeller.

The *cycleonium* was built with the support of the apprenticeship workshop of Chiron-Werke GmbH & Co. KG (www.chiron.de).

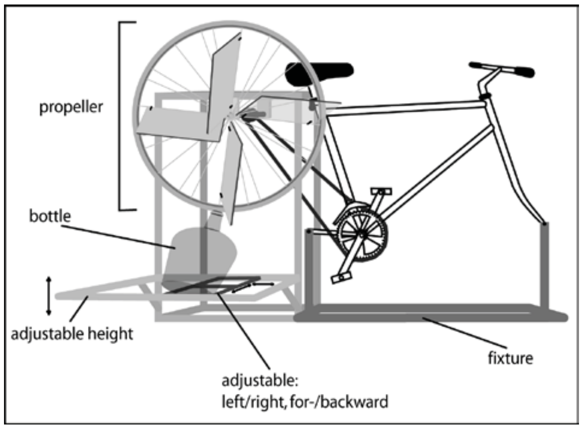


Figure 5. Technical schema

2.2. Purpose

The player perches on it like on a stationary bicycle and starts to pedal. The midi-controller and the Laptop can be put on a tray that is at the height of the handlebar. By using a dynamic microphone that is at the height of the mouth he can experiment with his voice respectively breath and modify it per MIDI in real-time. The sound of the arbor at the propeller and the chain is recorded with a condenser microphone (cardioid). Through reverse turns of the pedals the sound of chain and arbor becomes more concisely and is at the same time a variation of sound. By alternating forward and reverse turns a percussive rhythmical complement is built to the laminary sound of the bottle that mainly can be modified in the pitch. Dense and complex structures of sound can be achieved by adding overtones, glissandi and delays.
With this prototype of the *cycleonium* a performance was realized at the Kubus of the ZKM in 2009.

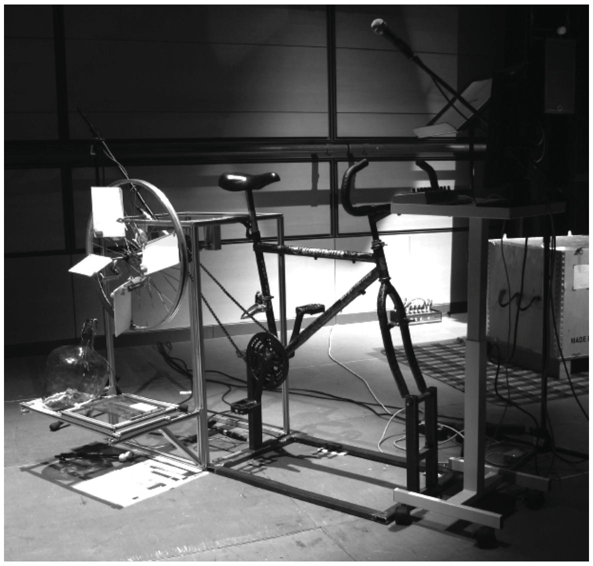


Figure 6. picture of the *cycleonium*[1]

3. PULSE GUITAR

3.1. Introduction

The *PulseGuitar* is a digital musical instrument [5]. It combines an electric guitar with a computer-controlled interface. While the strings of a conventional guitar are to be plucked by the player's fretting hand, the strings of the *PulseGuitar* are excited by micro loudspeakers (voice coils) that are attached near them (Fig. 7). The force transmission is mechanical. The instrument is played by moving a joystick which is fixed on its body with one hand, and, as usual, fretting the notes with the other hand. The data from the joystick controls a software synthesizer whose signal is amplified and finally routed to the voice coils. Dependent on the deflection of the joystick a program decides which coil the signal is routed to. The *PulseGuitar* can produce other timbres and rhythmic gestures than a common electric guitar.

Virtual Musical Instruments are based on modules. We can clearly separate the controller module from how the sound was produced (sound module). In this context, the instrument is controlled by sending data, such as MIDI, to some synthesizer or sampler that outputs sound [6]. At the very most, traditional acoustic/electric/electro-mechanical instruments do not fit into the scheme of modularity.
Since the electric guitar is an electric instrument, the controller and the sounding part cannot be separated into two independent components. The strings represent the controller module and also the origin of the sound and the major part of the sound module at the same time.
Changing any part of the controller would also affect the timbre of the instrument whereas replacing the controller of a virtual instrument will not cause any impact on the sound module.
The *PulseGuitar* revisits the electric guitar and takes a new approach in terms of how to play/control it.
It is fair to mention that several tools which aim to widen the musical diversity of the electric guitar have been developed in the past. Prominent devices are the Ebow (www.e-bow.com) and also the *Moog Guitar* (www.moogmusic.com/moogguitar). Both work, unlike the *PulseGuitar*, with electromagnetic fields that affect the state of the strings.
The PulseGuitar stays with the idea of plucking strings mechanically but expands the possibilities by passing the task to a machine.

3.2. Experimental Procedures

The synthesizer was built in the Max/MSP 5 programming environment [7]. It only produces a sawtooth wave, which is variable in frequency domain. Its signal is routed to little voice coils that are attached on the body of the guitar near the strings. Each string is equipped with its own voice coil. The mechanical movements of the coils are transferred to the strings. It is crucial to understand that the oscillator is not used as a musical instrument. Rather the physical deflection of the voice coils is needed. The strings are not plucked but hit.

[1] Picture by Kai Hanneken, ZKM 2009

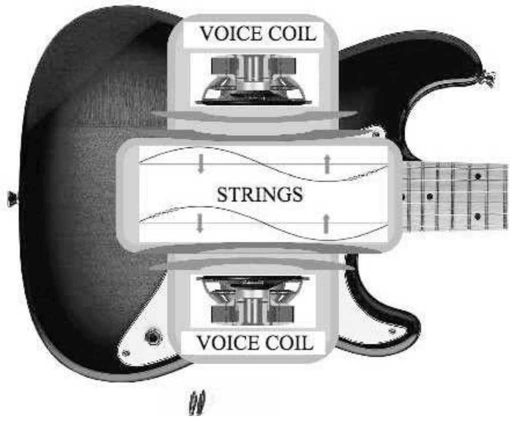


Figure 7. Schematic illustration

As a result, the synthesizer controls both the excitation of the strings and takes over the task of the player's right hand.
The synthesizer is controlled by a joystick (Fig. 8), which is also attached to the body. Turning the joystick up and down, the string to be played is selected, moving the stick sideways defines the frequency of the oscillator (Of course, other controllers such as a touchpad could be used). The range of the frequency is variable from 0.5 Hz up to 200 Hz so the strings can be hit 0.5 to 200 times per second.
In the end, the vibrations of the strings are captured by conventional pickups.

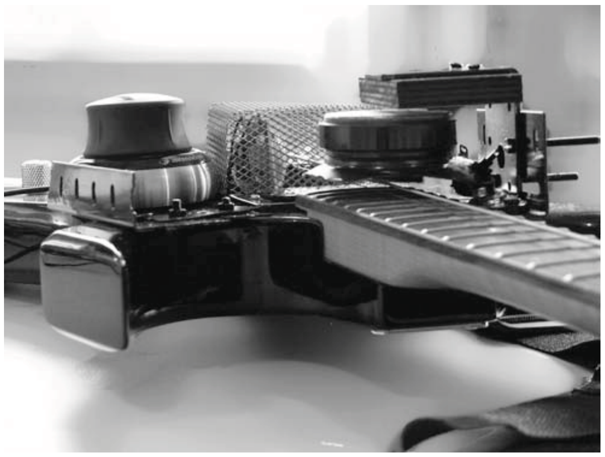


Figure 8. Picture of the *PulseGuitar*

At the moment, the *PulseGuitar* supports only two strings because a two channel audio interface was used. However, the string number can be easily increased using a multi channel interface.

The sound of the instrument depends on the frequency of the oscillator (and of course on the amplifier). Below about 20 Hz the otherwise continuous perception of oscillations (tone) changes to a perception of single events (rhythm). So if the strings are being plucked between 0.5 to 20 times per second the sound resembles a common electric guitar because a guitarist can't help playing in that time domain due to motoric limitations. If the frequency is increased though, it becomes more interesting. Above 20 Hz single excitations are not perceived as single events anymore, they unite with the sound of the strings.
The interesting part is not the fact that with the *PulseGuitar* one can play really fast (even though you could if you wanted to). Rather the timbre is interesting. It is percussive and harmonic in equal shares. The first live presentation of the *PulseGuitar* was done January 21, 2009 at ZKM Karlsruhe January 27, 2009 at Centro Cultural Belem Lisbon with the Portuguese Contemporary Music Ensemble OrchestrUtopica.

4. ACKNOWLEDGMENTS

Part of this article is a project PRICES contribute, supported by the FCT - Foundation for Science and Technology, Portugal (POCI 2010).

5. REFERENCES

[1] **DODGE, C. and JERSE, T.A**. ; *Computer Music – SynthesisComposition and Performance*, New York : Schirmer Books ; 1985

[2] **FERREIRA-LOPES, P.** ; *Étude de modèles interactifs et d'interfaces de contrôle en temps réel pour la composition musicale.* Thèse de Doctorat ; Paris ; Université de Saint Denis - Paris VIII - Dép. de Sciences et Technologies des Arts, 2004.

[3] **FERREIRA-LOPES, P. ;** *Traditional and digital music instruments : a relationship based on a interactive model* , SMC08 - Sound Music Computing International Conference, Berlin 2008.

[4] **HIGGINS, D. :** *Modernism Since Postmodernism: Essays on Intermédia*, San Diego State University Press , 1997.

[5] **JORDÀ, S.** , *Digital Lutherie Crafting musical computers for new musics' performance and improvisation*, Barcelona, 2005.

[6] **JORDA, S. ;** "Multi-user Instruments: Models, Examples and Promises", in Proceedings of 2005 International Conference on New Interfaces for Musical Expression NIME05 , Vancouver, BC, Canada, 2005.

[7] www.cycling74.com

[8] **MACHOVER, T. and CHUNG, J**. ; "*Hyperinstruments : Musically intelligent and interactive performance and creativity systems*" *in* Proceedings of the Computer Music Conference (ICMC89) ; Ohio : 1989.

[9] **MARSHALL, M.T. and WANDERLEY, M**.; "*A survey of sensor use in digital musical instruments"* www.music.mcgill.ca/musictech/idmil/projects/SensorSurvey

[10] **WINOGRAD, T.** : *Interaction Spaces for 21st Century Computing* in John Carroll (ed.), *Human-Computer Interaction in the New Millennium,* Addison-Wesley, 2001.

MAKING AN ORCHESTRA SPEAK

Gilbert Nouno, Arshia Cont, and Grégoire Carpentier
Ircam-Centre Pompidou
{nouno,cont,carpentier}@ircam.fr

Jonathan Harvey
Composer
jharvey@solutions-inc.co.uk

ABSTRACT

This paper reports various aspects of the computer music realization of "Speakings" for live electronic and large orchestra by composer Jonathan Harvey, with the artistic aim of making an orchestra *speak* through computer music processes. The underlying project asks for various computer music techniques: whether through computer aided compositions as an aid for composer's writing of instrumental scores, or real-time computer music techniques for electronic music realization and performance issues on the stage with the presence of an orchestra. Besides the realization techniques, the problem itself brings in challenges for existing computer music techniques that required the authors to pursue further research and studies in various fields. The current paper thus documents this collaborative process and introduce technical aspects of the proposed methods in each area with an emphasis on the artistic aim of the project.

1 INTRODUCTION

This paper documents a collaborative work surrounding the production and realization of "Speakings" for large orchestra and live electronics, between the composer Jonathan Harvey and several researchers and computer music designers at IRCAM [1] . The central idea behind the project is the composer's aim to bring in speech and music structures together through live electronics and orchestral writings and with the aid of computer music formalizations and realizations. We therefore devote this introduction to the clarification of artistic goals of the project.
Speech and music are very close and yet also distant. The musical quality of speech has long been known to composers and artists. Today, sound analysis tools show us that speech signals not only contain melodic information but also harmonic and inner rhythmic and dynamic qualities pertained to complex musical structures. Recent research has uncovered common trajectories between the evolutionary roots of music and speech [9]. The central idea of this project is to bring together orchestral music and human speech but not merely through realistic speech synthesis and semantic contents of the speech, but to emphasize non-verbal aspects of speech structures in music composition. It is as if the orchestra is learning to speak, like a baby with its mother, or like first man, or like listening to a highly expressive language we don't understand. The rhythms and emotional content of speech are not only formed by semantics, but also (or probably more) formed by specific spectral dynamics of speech signals despite the semantic context. Therefore, making an orchestra speak in this sense is not to reach the semantic values of speech through computer music processes, but to emphasize the non-verbal structures in speech and realize them through instrumental writing as well as live electronic processes. Starting from baby screaming, cooing and babbling, an evolution of speech consciousness through frenzied chatter to mantric serenity becomes the basic metaphor of the half-hour work's trajectory.

With this respect, speech structures are introduced into musical patterns through two distinct processes: (1) With the use of computer-aided composition techniques to enhance instrumental writing. Through this process, the orchestral instruments - soli and ensembles - would imitate speech patterns, full of glissandi, fast, and a mixture of percussive consonants and sliding vowels. Computer music techniques that allow such a passage are automatic transcription of speech signals to symbolic music notation, as well as a novel automatic orchestration technique introduced later in the paper. And (2) using of real-time analysis/resynthesis techniques to reshape the orchestral audio signals to speech structures, through which the orchestral discourse, itself inflected by speech structures, is electro-acoustically shaped by the envelopes of speech taken from largely random recordings. The vowel and consonant spectra-shapes flicker in the rapid rhythms and colors of speech across the orchestral textures.

This paper is organized as follows: In the incoming section, we discuss previous works and their relation to the presented paper. Section 3 details the first phase of this work or *computer-aided composition* techniques as means to provide musical material for instrumental writing. Section 4 details real-time audio processing techniques employed for the computer music realizations and their relations to the instrumental writing both at the score level and during performance. Section 5.1 discusses performance and synchro-

[1] http://www.ircam.fr/

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

nization issues between live electronics and the orchestra, and we conclude the paper by remarks and discussions on further developments of techniques introduced in the paper.

2 PREVIOUS WORKS

The idea of bringing music and speech together in orchestral composition is not new and has a long history that goes beyond the scope of this paper. For example, the russian composer Modest Mussorgsky's orchestral writing was highly influenced by the relations between speech and music. He went further to claim that the aim of musical art as the reproduction in musical sounds not only of modes of feeling but mainly of the reproduction of modes of human speech [7]. Another more recent example in computer music, is Clarence Barlow's *Synthrumentation* by spectral analysis of speech and their resynthesis to acoustic instruments [2]. Another similar but more recent work is Claudy Malherbe's piece *Locus* for real and virtual voice (1997). Malherbe's work make use of voice analysis techniques to deduce symbolic music materials used during composition and through a formal development compromising voice, speech and noisy structures (see [8] for details and documentations). Both Malherbe and Barlow's attempts to deduce speech structures in music could be categorized within the realm of *Computer-Aided Composition*, where music materials emerge out of composers' formalizations and offline treatment of music materials and/or eventually sound synthesis. The work presented here partially adopts both approaches in [2] and [8], see section 3.1, but takes one step further by enhancing hidden speech structures through real-time analysis of orchestral sounds and their timbre-stamped synthesis using known speech structures, detailed in section 4. This addition is not only due to artistic aims of the project, but for limitations of formal analysis techniques where non-verbal structures and inner rhythmic content of speech are often lost.

3 COMPUTER-AIDED COMPOSITION TECHNIQUES

This section details the first phase of our realization, to provide preliminary musical materials taken out of analysis of voice and speech structures, and an aid to orchestral writing. Such activities are generally referred to as *Computer-aided Composition (CAC)*. The output of this phase are raw symbolic score materials that help the composer realize the orchestral score. Speech samples used for this phase are partially random and chosen recordings of radio interviews, natural baby babbles and sounds, and poetry readings chosen by the composer. We discuss this phase within two steps: In the first, we simply transcribe melodic and harmonic structures of speech and voice (if any) through sound analysis and symbolic music score. Afterwards, we incorporate non-harmonic speech structure such as formants and timbral dynamics to provide an aid for orchestration.

3.1 Melodic and Harmonic Voice Transcription

The simplest way to extract musical information out of voice (or any audio) signal is to transcribe the melodic and harmonic structures into symbolic scores. For voice and speech, this information does not illustrate most interesting internal structures (such as formants) but is nevertheless important as a first insight. Extracting the melodic part of any speech signal amounts to running a simple pitch detector on the audio available in most computer music systems. However, a better way to extract melodic pitch information on speech, and in order to be coherent with the inner-structure, is to extract melodic contours on the level of syllabic segmentations. To this end, we chose the commercially available *Melodyne* editor [2] that automatically performs syllabic segmentations and allows further refinement of results through its intelligent graphical user interface. The results of this melodic transcription are then saved as MIDI files which will be mixed later with the harmonic transcriptions.

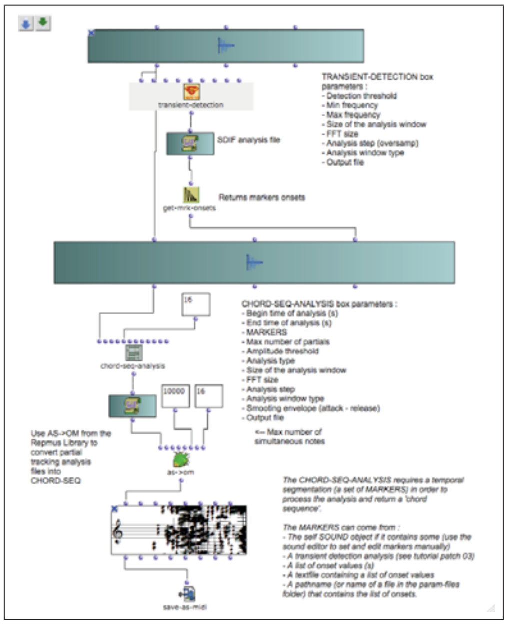
By harmonic transcription of speech and voice signals, we aim at transcribing a partial tracking analysis of the audio spectrum into polyphonic music scores. For this aim, we pass the audio recording to a transient detector and partial tracking module based on [11], and then translate "loud" enough partials into symbolic notation followed by rhythmic quantization. The whole procedure is done in one shot and in the *OpenMusic* programming environment, and using its default libraries [1]. Figure 1 shows a snapshot of the patcher used for this procedure, starting at the top (the audio) to the bottom (symbolic transcriptions).

Combining both melodic and harmonic transcription results would result into a combined score that reveals the harmonic structures of a speech signal through symbolic music notation. Figure 2 shows a sample score result of this process. Note again that this process reveals only elementary harmonic structures of the signal through pitched notes, and does not reveal any interesting timbral or formant structures. This latter is the goal of the next section.

3.2 Automatic Orchestration

Among all techniques of musical composition, orchestration has never gone further than an empirical activity. Practicing and teaching orchestration – the art of blending instrument timbres together – involve hard-to-formalize knowledge and experience that computer music and composition systems have for years stayed away from. Although several recent attempts to design computer-aided orchestration tools should be mentioned (see [3] for a review), those systems only offer little ability to finely capture musical timbre. Moreover, they

[2] http://www.celemony.com/



TRANSIENT-DETECTION box parameters :
- Detection threshold
- Min frequency
- Max frequency
- Size of the analysis window
- FFT size
- Analysis step (oversamp)
- Analysis window type
- Output file

CHORD-SEQ-ANALYSIS box parameters :
- Begin time of analysis (s)
- End time of analysis (s)
- MARKERS
- Max number of partials
- Amplitude threshold
- Analysis type
- Size of the analysis window
- FFT size
- Analysis step
- Analysis window type
- Smooting envelope (attack - release)
- Output file

<-- Max number of simultaneous notes

Use AS->OM from the Repmus Library to convert partial tracking analysis files into CHORD-SEQ

The CHORD-SEQ-ANALYSIS requires a temporal segmentation (a set of MARKERS) in order to process the analysis and return a 'chord sequence'.

The MARKERS can come from :
- The self SOUND object if it contains some (use the sound editor to set and edit markers manually)
- A transient detection analysis (see tutorial patch 03)
- A list of onset values (s)
- A textfile containing a list of onset values
- A pathname (or name of a file in the param-files folder) that contains the list of onsets.

Figure 1. OpenMusic Snapshot of partial analysis tracking for harmonic transcription

Figure 2. Resulting score sample

are often limited to small-size problems due to the cobinatorial complexity of orchestration (the set of playable sound mixtures in a large orchestra is virtually infinite).

The piece of music presented in this paper is the first to benefit from the most recent advances in automatic orchestration research [3]. With the aim of composing instrument textures that imitate the timbre of sung vowels we used the `Orchidée` [4] orchestration tool. `Orchidée` is a MATLAB-based server application that communicates with traditional computer-aided composition environments through OSC [13] messages (see figure 3). `Orchidée` embeds both a representation of instrument capabilities – obtained from prior analysis and indexation of large instrument sound sample databases – and a set of efficient orchestration algorithms. Given an input target sound, `Orchidée` outputs a musical score for imitating this sound with a mixture of traditional instruments.

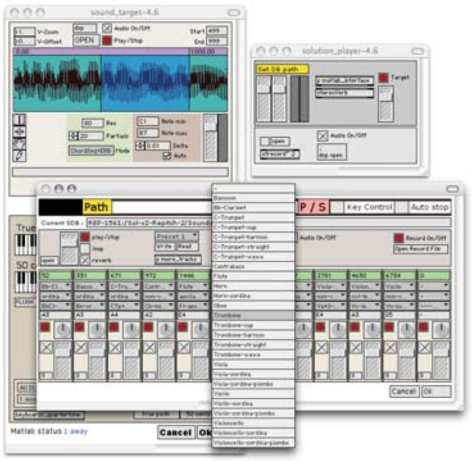


Figure 3. Controlling `Orchidée` with Max/MSP

Compared to its predecessors `Orchidée` offers many innovative features. First, the instrumental knowledge in `Orchidée` is represented by a sound description database in which each item is a an instrument sound sample associated with *musical attributes* (musical variables such as instruments, dynamics etc.) and *perceptual features* (such as brightness, roughness etc.). Second, `Orchidée` embeds a *timbre model* that efficiently estimates the joint features of any instrument sound mixture. The resulting features may then be compared to the target's features and a similarity estimate along each perceptual dimension may be computed. Last, `Orchidée` explicitly addresses combinatorial issues and tackles the orchestration problem in its inner complexity. The system comes with a time-efficient evolutionary orchestration algorithm allowing the exploration of non-intuitive sound mixtures and the fast discovery of nearly-optimal orchestration proposals. By iteratively capturing and refining users' implicit preferences, the algorithm may quickly identifies the most relevant configurations for a given orchestration situation. For more details see [4].

For the realization of the piece discussed here, `Orchidée` was used to write orchestral background textures that imitate sung vowels. The starting point was a simple three notes mantra sung and recorded by the composer. To each note corresponded a given vowel: *Oh/Ah/Hum* (see Fig. 4). The goal was to imitate the sound of the sung mantra with

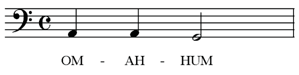


Figure 4. Mantra used as an input for `Orchidée`

a ensemble of 13 musicians. The composer wanted the orchestra to sing the mantra 22 times, and wished the resulting timbre to evolve along the ostinato in the following manner: The sound was to become louder and louder, brighter, and closer over time to the target vowel. The orchestration was to use progressive pitches with harmonic richness. Feature optimization and constraints specification and handling techniques provided by `Orchidée` were jointly used to generate a continuously evolving orchestration. Figure 5 shows an excerpt of the overall result.

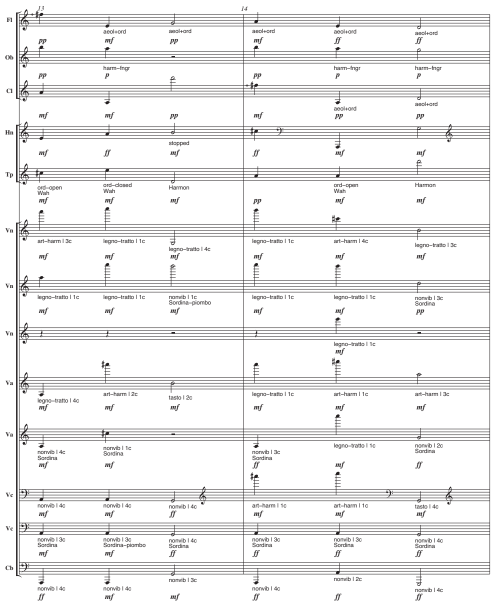


Figure 5. Excerpt of mantra orchestration by `Orchidée`

4 REAL-TIME ANALYSIS/SYNTHESIS TECHNIQUES

In the previous section, we showed how some inherent speech structures such as harmonic and formant structures could be translated to raw musical material and used during orchestral writing. Despite the significance of the information provided in this phase of work, there seem to be a lot of interesting structural information that are seemingly lost in this process, or would be lost during a realistic orchestral performance. To overcome this, we propose integrating the spectral dynamics of speech structures directly onto the orchestral spectrum through real-time analysis and resynthesis, and without passing through any formalization scheme as was the case in section 3. This way, we hope to inherit directly the inner-rhythmical and dynamic formant structures of speech onto the transformed orchestral sound.
To this aim, we couple the real-time orchestral audio, as the audio result of the transcription and orchestration process of section 3 out of given speech samples, with the speech sample itself through an analysis/synthesis scheme. The analysis/resynthesis used for this proposal is based on formant envelope computation of the speech signal using Linear Predictive Coding (LPC) and enforcing it onto the orchestral's natural spectrum envelopes arriving in real-time. The real-time implementation of this process is done using the *Gabor* libraries [12] in the *MaxMSP* programming environment[3]. Figure 6 shows a snapshot of the realtime process with visualizations of deduced formant envelopes, orchestral input and formantized orchestra on one analysis frame.

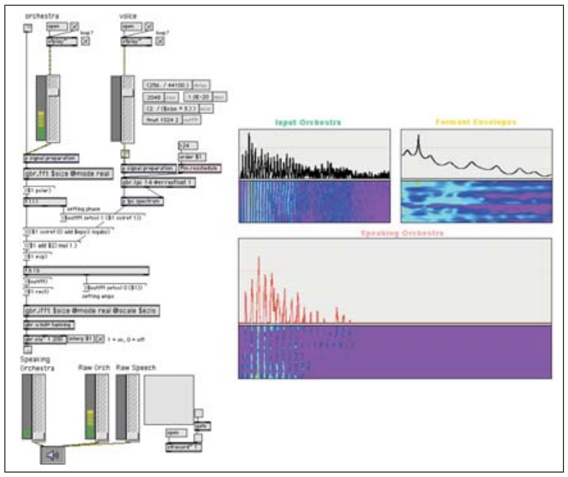


Figure 6. Realtime Formantization MaxMSP patcher

Using this process, the inner-rhythmical structures of speech are stamped into the orchestra and could be diffused as live electronics through a spatialization process. This process, hence introduces a second phase of composition involving the coupling of pre-recorded speech with sections in the orchestra and parametrizing the live electronics for diffusion.

5 PERFORMANCE ISSUES

5.1 Live electronic Synchronization

In an orchestral setting, human musicians' temporal synchronization is assured during the live performance through constant and active coordination between themselves, the music score and a conductor. Adding live or fixed electronics into the equation should not undermine the importance of this synchronization process, which is one of the main responsible factors for musical expressivity.
Traditionally, synchronization between electronics and instrumental components of a mixed piece of music has been done either by human performers through adhoc cueing, by

[3] `http://www.cycling74.com/`

score following paradigms, or by a combination of both. Either way, this type of synchronization usually assures the starting point (or correct triggering) of processes in time but not necessarily their temporal life-span. The following simple example can illustrate this important problem: Imagine an instrumental score accompanied by a fixed electronic (audio file) over three measures. Assuming that the score has a time-signature of $4/4$ with a tempo of $60BPM$, the corresponding audio should ideally have an initial length of 12 seconds. During live performance, the human player might for many reasons vary the initial tempo of 60 from the very beginning to the end of the third measure. In such situations, although the onset trigger could be easily made accurate, synchronization of the overall electronics could not be assured unless the electronic process detects and undertakes the same temporal dynamics of the human performance over the electronic score, as if two humans were interacting.

Given the temporal nature of the analysis/resynthesis technique described in section 4 we are in the schema of the example above: The speech samples behind the live processing should not only be triggered on-the-fly, but also temporally aligned to the orchestra during their life-span. To achieve this, we use `Antescofo`, a score follower that aligns score positions and also decodes an anticipatory tempo of the performance [5]. Using this information each sound file is then played back using `SuperVP`[4] advanced phase vocoder technology [11] to preserve their quality upon temporal adaptation. To achieve this, a keyboard player in the orchestra plays its own score along with the orchestra and synchronous to the tempo given by the conductor. The `Antescofo` score of the keyboard part contains not only the instrumental score, but also electronic commands written in relative beat-time, translated to clock-time at each tempo change. This way, getting back to the simple example described above, we can be certain that upon continuous tempo change of the instrumental section, the audio playback is assured to change time span during live performance and up to an acceptable precision. This procedure is applied to all live treatments and sound diffusions.

5.2 Live Electronics Interpretation

The coexistence of orchestral and computer music parts emerge from specific artistic purposes. Composer's initial idea in employing live electronics was to enhance the sonic space of the piece with a network of relations between electronic and instrumental sounds. Such internal or intimate sound relations emerge not only through compositions but also through interpretation of the electronics during performance. With this respect, live electronics is considered as an additional *instrument* in the piece whose performed gestures during any executation contribute to the wholeness of the sonic space created within the piece. We have used the `Lemur`[5] multi-touch screen interface to easily map hand gestures onto the sound processes, and to control the amounts of sound processing like a musician would control the amount of a muted sound listening to what she is producing in interactions of his movements with what she hears and expects. This feedback behavior goes farther than a simple mixing of the audio signals as it enables a direct interpretation of the musical choices. Figure 7 shows the main control screen designed for live performance as about the size of two hands.

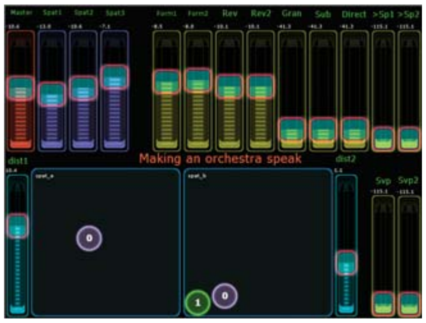


Figure 7. The *Lemur* multi-touch device interface

Human intervention of this kind for interpretation of live electronics is in no way in contradiction to the automaticity of cue-lists or the use of score following paradigms, but rather complimentary. The passage from any automatic process to "music" is beyond the process itself and depends on our hearing abilities and interaction with the sound itself as musicians. This is inherently the purpose of the simple interactive framework using the `Lemur` interface setup of figure 7. The act of making live electronic music design is then to find the balance between automatism and interference to make interactions more musical.

5.3 Enhancing Rhythm with Space

The interactive surface display of the `Lemur` in figure 7 offers extended possibilities rather than mere mixing of effects, to control spatial movements either automatically or manually through a virtual 2D space. Sound spatialization is achieved using dedicated intensity panning of `Spat`[6] real-time modules [6]. Using spatialization, we bring the sound into the audience space and put the emphasis on some theatrical characters of the sound, thus reinforcing and converging toward the voice quality of the sound processing. It is also a mean to musically enhance the rhythmic perception through fast movements, either within a rhythmic counterpoint with the orchestra or in phase with what it is

[4] `http://forumnet.ircam.fr/708.html?&L=1`

[5] `http://www.jazzmutant.com`

[6] `http://forumnet.ircam.fr/692.html`

being performed. It is also an extension of the score writing techniques as space is considered as a compositional parameter [10]. An advantage of employing Spat modules for this 8-channel work is its ability to *perceptually control* the spatial quality rather than mere positioning of sources in space and in realtime. Spat movements are algorithmically composed using modules offered by the ICST [7] interface tools originally developed for ambisonics spatialisation techniques, here adapted to the Spat modules. Figure 8 shows the diagram of the real-time performance setup implemented in the MaxMSP programming environment based on the considerations discussed in this section.

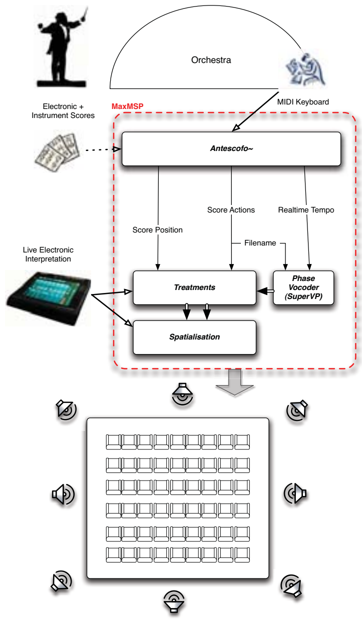


Figure 8. Real-time Performance diagram

6 CONCLUSION

In this paper, we documented a collaborative work between a composer and researchers to create an orchestral piece with live electronics with the aim of enforcing speech structures over an orchestra. "Speakings" was first performed by the BBC Scottish Orchestra and IRCAM in the BBC Proms festival on August 2008 and has since had several more performances. An audio recording is under publication.

7 REFERENCES

[1] Gérard Assayag, Camilo Rueda, Mikael Laurson, Carlos Agon, and O. Delerue. Computer Assisted Composition at Ircam: From PatchWork to OpenMusic. *Computer Music Journal*, 23(3), 1999.

[2] Clarence Barlow. On the spectral analysis of speech for subsequent resynthesis by acoustic instruments. *Forum phoneticum*, 66:183–190, 1998.

[3] Grégoire Carpentier. *Approche computationnelle de l'orchestration musicale – Optimisation multicritère sous contraintes de combinaisons instrumentales dans de grandes banques de sons*. PhD thesis, UPMC Paris 6, Paris, 2008.

[4] Grégoire Carpentier and Jean Bresson. Interacting with Symbolic, Sound and Feature Spaces in Orchidee, a Computer-Aided Orchestration Environment (accepted for publication). *Computer Music Journal*, 2009.

[5] Arshia Cont. Antescofo: Anticipatory synchronization and control of interactive parameters in computer music. In *International Computer Music Conference*, North Irland, Belfast, Août 2008.

[6] Jean-Marc Jot and Olivier Warusfel. A real-time spatial sound processor for music and virtual reality applications. In *ICMC: International Computer Music Conference*, pages 294–295, Banff, Canada, Septembre 1995.

[7] Leslie Kearney. *Linguistic and musical structure in Musorgsky's vocal music*. PhD thesis, Yale University, 1992.

[8] Claudy Malherbe. *The OM Composer's Book*, volume 2, chapter Locus: *rien n'aura eu lieu que le lieu*. Editions Delatour France, 2008.

[9] Steve Mithen. *The Singing Neanderthals: The Origins of Music, Language, Mind and Body*. Weidenfeld & Nicolson, 2005.

[10] Gilbert Nouno and Carlos Agon. Contrôle de la spatialisation comme paramètre musical. In *Actes des Journées d'Informatique Musicale*, pages 115–119, Marseille, France., 2002.

[11] Axel Roebel. Adaptive additive modeling with continuous parameter trajectories. *IEEE Transactions on Speech and Audio Processing*, 14-4:1440–1453, 2006.

[12] Norbert Schnell and Diemo Schwarz. Gabor, multi-representation real-time analysis/synthesis. In *COST-G6 Conference on Digital Audio Effects (DAFx)*, pages 122–126, Madrid, Spain, Septembre 2005.

[13] Matthew Wright, Adrian Freed, and Ali Momeni. OpenSound Control: State of the Art 2003. In *Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03)*, Montreal, Canada, 2003.

[7] http://www.icst.net

EXPLORATIONS IN CONVOLUTIONAL SYNTHESIS

Tamara Smyth, Andrew R. Elmore
School of Computing Science, Simon Fraser University
tamaras@cs.sfu.ca, aelmore@sfu.ca

ABSTRACT

In this work we further explore a previously proposed technique, used in the context of physical modeling synthesis, whereby a waveguide structure is replaced by a low-latency convolution operation with an impulse response. By doing so, there is no longer the constraint that successive arrivals be uniformly spaced, nor need they decay exponentially as they must in a waveguide structure. The structure of an impulse response corresponding to an acoustic tube is discussed, with possible synthesis parameters identified. Suggestions are made for departing from a physically-constrained structure, looking in particular at impulse responses that are mathematically-based and/or that correspond to hybrid or multi-phonic instruments by interleaving two or more impulse responses. The result is an exploration of virtual musical instruments that are either based on physical instruments, completely imagined, or somewhere in between.

1 INTRODUCTION

It is common practice for contemporary musicians to explore playing techniques by producing sounds on their instruments that may be unusual, surprising or perhaps completely atypical of the instrument. In the case of wind instruments, extended playing techniques such as side and flutter tonguing, false fingering, split tones, circular breathing, are typically practiced by very accomplished musicians as who have highly proficient and virtuosic control over their airflow and embouchure. By employing unusual playing techniques, the player achieves a sound that seemingly extends or redefines the instrument itself.

In addition to extending playing techniques, it has become increasingly common for modern musicians to *effectively* modify the instrument itself, both offline and during real-time performance. Some notable examples include recent performances by clarinetist Francois Houle creating a polyphonic effect by blowing into two different instruments simultaneously, extending the clarinet by directing the radiated sound onto the strings and soundboard of an open grand piano, or removing parts of the clarinet bore during performance. Modern trumpet players, such as slide trumpet player Steve Bernstein, often use a technique where the microphone is *swallowed* by the trumpet, effectively altering the horn's frequency response by removing the radiation transfer function of the flare, producing a low-pass sound similar to a rubber mallet dragging along the taut skin of a drum. Many musicians will also explore custom built instruments, with slight modification to the shape of the bore, the bell or the mouthpiece.

Some such instrument extensions can be awkward, costly or impractical, and limiting if the musician doesn't want the change to be permanent or damage the instrument. In any event, no matter what physical modifications are made, the instrument will always be constrained by the acoustics governing the system, which in the case of wind instruments, will yield a periodic impulse response with uniformly spaced arrivals (or echos) corresponding to the length of the instrument, with each consecutive arrival decaying over time due to reflection and propagation losses. Changing or removing components of the instrument will alter the frequency response of these losses, but they will always have a low-pass characteristic and an exponential decay in amplitude, and the impulse response will always be periodic. Convolutional synthesis would allow for greater possibilities related to the practice of instrument extension since, unlike the impulse responses corresponding to a waveguide synthesis model (which aims to model a physical structure), the employed impulse responses are not necessarily physically constrained.

In this work we further explore this previously proposed technique of convolutional synthesis [5], and propose a solution whereby the performer may enhance or replace their instrument all together with a synthesized parametric impulse response. That is, if the signal generated by the reed may be estimated from the signal recorded at the bell during a real-time performance of the instrument [6], it may be convolved directly with a new parametric impulse response corresponding to a new instrument, either physics-based, completely imagined, or perhaps somewhere in between. Alternatively, rather than using the estimated reed pulse directly, it may be used to extract key playing parameters that can then be remapped to the control parameters of a synthesized source, perhaps a parametric pulse train or a more rigorous physics-based model of a generalized-reed [7], which may then be convolved with a new parametric impulse response.

SMC 2009, July 23-25, Porto, Portugal

In both cases, using a low-latency convolution operation [1] as described in Section 4, will allow for a level of real-time interactive control comparable to a waveguide model.

The term *Convolutional Synthesis* borrows from the term *Convolutional Reverb*, a technique used for simulating reverberation of a physical or virtual space by using its characteristic impulse response. In this work, measurements of acoustic tubes, and musical instrument bores, are used as a base from which a parametric impulse response may be synthesized, corresponding to possible wind instrument bores, either existing, or imagined.

2 PARAMETRIC IMPULSE RESPONSES

2.1 Synthesizing one-dimensional propagating waves

A physics-based model of a wind instrument usually consists of a source (the reed pulse) and a filter (the bore, bell and possibly the mouthpiece). Depending on the reed configuration, the flow from the reed may have a strong or weak coupling to the connected acoustic tube, that is, the resonance of the bore and bell may have a varying degree of influence over the resonance or the reed pulse. The oscillating reed provides an excitation to the bore by modulating the air flowing through an aperture with a time-varying cross-sectional area, providing a pressure input. This input is effectively convolved with the instrument's impulse response (as measured at the reed end) to produce the pressure at the base of the bore (or mouthpiece).

Pressure waves travelling along the instrument bore are frequently modeled using the digital waveguide [3] structure seen in Figure 1. This enables use of appropriate waveguide elements to account for propagation losses, $\lambda(\omega)$, reflection losses at the tube ends $R_{cl}(\omega)$ and $R_{op}(\omega)$, and a change of tube length corresponding to the delay z^{-L}.

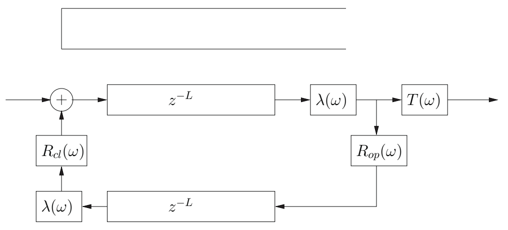


Figure 1. A waveguide model of a cylinder closed at one end with reflection $R_{cl}(\omega)$ and open at the other with reflection $R_{op}(\omega)$.

The impulse corresponding to the waveguide structure in Figure 1, as driven and measured at the closed end, is fairly straightforward to synthesize by following the signal flow of the propagating wave in response to an applied impulse: if an impulse enters the tube at the (not perfectly) closed end, the pulse would be measured at that location at that point in time. After the impulse travels round-trip to the bell and back, a second arrival, A_2, would be measured $2L$ samples later at the bore base, and would consist of the initial pulse filtered according to

$$A_2 = \lambda^2(\omega)R_{op}(\omega)(1 + R_{cl}(\omega)), \tag{1}$$

where the term $(1+R_{cl}(\omega))$ corresponds to the need to sum the left and reflected right going traveling waves to obtain the actual pressure as recorded at the *closed* end. The arrival A_2 would also travel down to the bell and back, and produce a third arrival, consisting of A_2 with additional filtering according to

$$A_3 = \lambda^4(\omega)R_{op}^2(\omega)R_{cl}(\omega)(1 + R_{cl}(\omega)). \tag{2}$$

Every subsequent arrival would be the result of the previous arrival having been subjected to another round of wall and reflection losses.

Figure 2 shows the impulse response corresponding to the closed-open tube and waveguide structure shown in Figure 1. The impulse response has a very clear periodic structure, with uniformly spaced sequence of arrivals corresponding to the length of the tube. The losses in the system have a low-pass characteristic, causing an overall exponential decay, with each arrival being increasingly smeared in time.

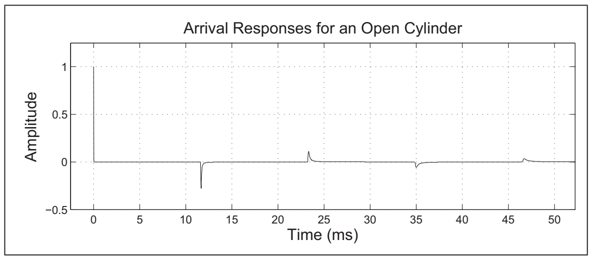


Figure 2. A synthesized impulse response corresponding to a cylindrical acoustic tube closed at one end and open at the other.

This impulse response may be synthesized by creating a sequence of arrivals, each one containing the appropriate filtering corresponding to (1) and (2), up to the number of desired arrivals. One parameter that may be made available to the user in this case is the spacing between the arrivals, which may be stretched or contracted (which would correspond to a change in delay z^L in in Fig. **??**) depending on the desired pitch. Alternate filters for R_{op} (theoretical, measured, or imagined) may be swapped in to the impulse response during performance, corresponding to a real-time change of the bell shape. Similarly, the transfer function R_{cl} may be modified to simulate a change in reed: a lip reed for example, would have more significant loss than would a woodwind reed.

2.2 Other Possible Impulse Responses

Regardless of the filter transfer functions or the delay length L, the impulse response corresponding to Figure 1 will always have uniformly spaced arrivals, and if stable, an overall amplitude envelope that decays over time. Yet when synthesizing the impulse response directly, that is, by not using a waveguide model, there is no need to adhere to these physical constraints. The period between pulse arrivals need not be uniform but may have some other structure, perhaps randomly distributed, or according to some mathematical model such as the golden ratio as in Figure 5. It may also be possible to create hybrid or multi-phonic instruments by interleaving two separate impulse responses (as in Figure 3). Also, the sequences need not decay exponentially; they may grow and then suddenly drop, with rates specified parametrically if so desired.

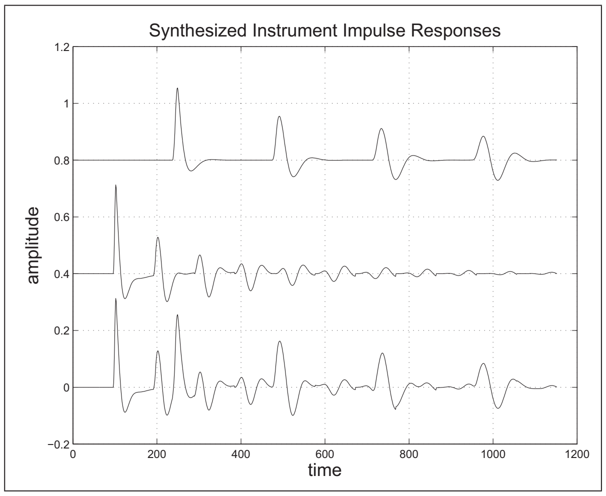


Figure 3. An impulse response is synthesized by interleaving two separate impulse response (top and middle) to produce a multi-phonic impulse response (bottom).

It is also possible to use an entire impulse response as measured from an instrument, such as the one shown for a trumpet in Figure 4. Though periodicity is less visible here, the impulse response is still composed of a sequence of arrivals, each having a transfer function corresponding to any losses in the system, most notably the wall losses and the reflection at the bell. This impulse response could be used as is, or by changing identified parameters much like was done in [2], but instead of replacing leading zeros of a measured reflection function with a delay line, we propose simply making the number of zeros in the impulse response variable, and use it to create a periodic impulse response. Another possibility is to convolve this measured impulse response with one that has been synthesized, creating a hybrid instrument.

3 CONVOLUTIONAL SYNTHESIS

Once a parametric impulse response is in place, the system needs an excitation mechanism, that is, a signal to be convolved with the impulse response. Here we consider three possibilities: a parametric pulse train, a physics-based model of a reed, and a signal corresponding to a reed pulse, estimated be inverse filtering during a live performance.

3.1 A parametric pulse train

In this case, results are explored using a sequence of bell-curve shaped functions (such as gaussian), the width, shape (symmetry) and periodicity of which are controllable parameters. Since it may be desirable to have the source strongly coupled with the acoustic filter, the periodicity of the pulse train may be set automatically to a value corresponding to the fundamental frequency, or a harmonic, of the the parametric impulse response.

A graphical user interface, as shown in Figure 5, was developed to allow for modification of both source (pulse train) and impulse response parameters, and to perform the convolutional synthesis.

3.2 A physics-based reed model

In this case, we explore the use of a physics-based model of a generalized reed as a source. The reed model depends on a bore model for obtaining the downstream (bore) pressure, which is used for establishing the pressure difference across the reed, a requirement for oscillation.

The problem that may be encountered here is that certain reed configurations will not oscillate under all bore conditions (this is an expected limitation of a physics-based model). Rather than limit the producible sounds by limiting the possible impulse responses, we propose the use of two impulse responses (as seen in Figure 6), whereby one impulse response, $h(t)$, is convolved with the bore input pressure (the product of the airflow from the reed U and the bore impedance Z_0) to produce the bore pressure required to achieve reed oscillation, and the other impulse response, $g(t)$, is convolved with the input pressure to obtain the produced sound. It should be noted that if keeping a physical structure is desirable, $g(t)$ may be inferred from $h(t)$, with the assumption that a transmission is always amplitude complementary to its corresponding reflection (that is, it has transfer function $G(z) = 1 + H(z)$ for pressure waves [4]).

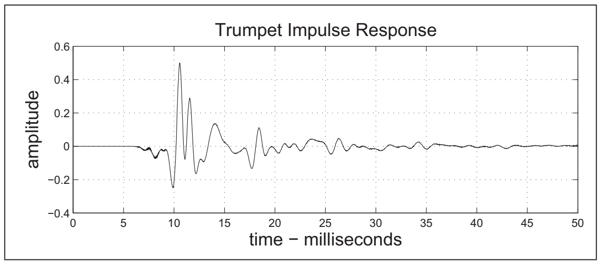


Figure 4. An impulse response from a trumpet.

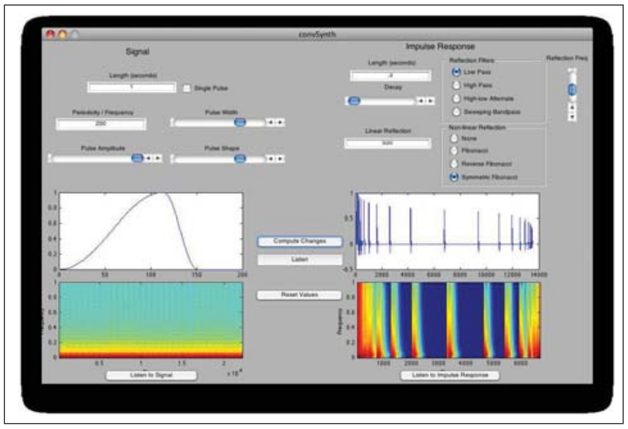


Figure 5. A graphical user interface allows for parametric synthesis of both input source and impulse response. For the impulse response, the user may specify the spacing of the arrivals, the filters of which they are composed, and the individual arrival and overall amplitude envelopes. For the source, the user may specify periodicity, shape and width of pulses.

It may also be possible however, to use completely unrelated impulse responses.

3.3 An estimated reed pulse

In this case, the bore impulse response is not only used for synthesis, but for analysis and estimation of the reed pulse, during an actual performance. If the reed pulse may be estimated using an inverse filter corresponding to the correct impulse response (which will change during performance), the reed pulse may be directly convolved with an impulse response similar or entirely different from the one corresponding to the instrument being played. Alternatively, features corresponding to different playing techniques may be extracted from the estimated reed pulse, and then remapped to control parameters of the pulse train or reed model described above. This application is the ultimate aim of the convolutional synthesis technique and relies on successfully estimating the reed pulse, a research goal that has obtained several results [6].

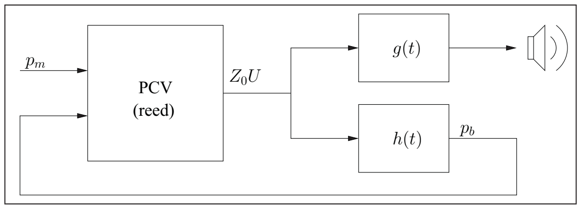


Figure 6. A signal flow diagram of convolutional waveguide synthesis in the context of a reed instrument. The bore pressure p_b is obtained by convolving the bore input pressure Z_0U with the impulse response $h(t)$, and the model output is obtained by a convolution with the impulse response $g(t)$.

4 LOW-LATENCY CONVOLUTION

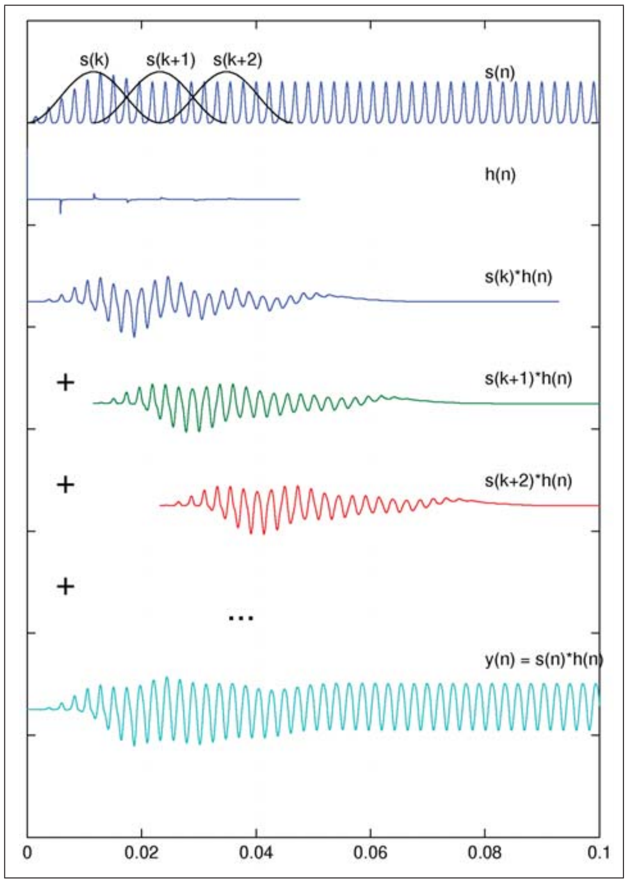


Figure 7. Fast convolution with latency

Though the waveguide structure, as seen in Figure 1, could have been used to filter all three source/excitation examples described in Section 3, we propose the use of a low-latency convolution, as described in [1], with a complete impulse response (i.e. not just a reflection function corresponding to a single round-trip down the bore to the bell and back).

If latency is not a concern, a simple implementation of fast convolution as illustrated in Figure 7 may be used, whereby a blocksize of samples $s(k)$ is convolved with the entire impulse response $h(n)$ and stored. One hopsize later, the next blocksize of samples $s(k+1)$ is also convolved with $h(n)$ and added to the previously stored result. This process continues until the end of the input signal $s(n)$ is reached. The problem with this implementation is that there is a latency corresponding to the length of the impulse response. That is, if parametric changes are made to the impulse response, the effects of one or more previous impulse response state(s) would continue to be heard for a time, in the worst case, corresponding to the sum of the lengths of the impulse response and blocksize (see the lengths of the component signals $s(k)*h(n)$, $s(k+1)*h(n)$ etc. in Figure 7 as compared to counterpart components in the low-latency convolution illustrate in Figure 8).

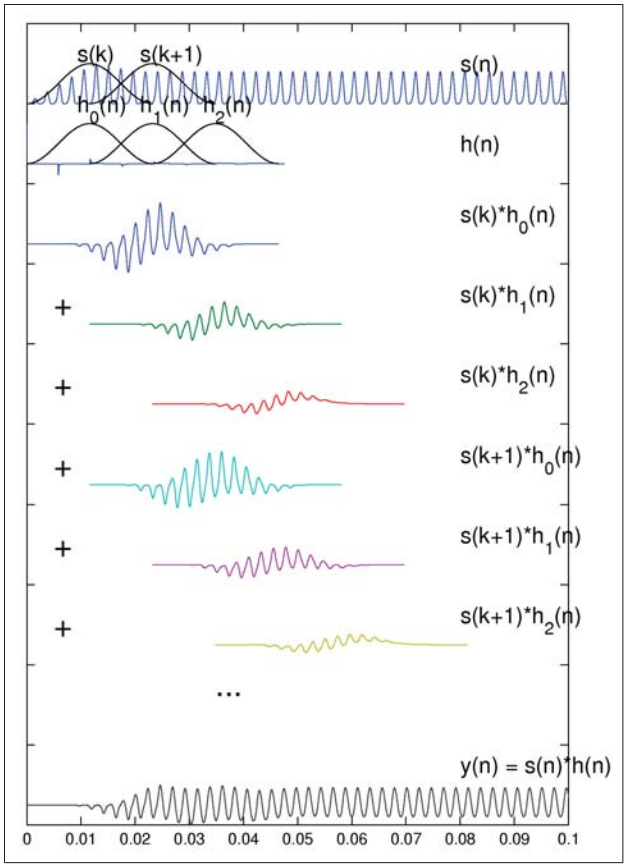


Figure 8. Fast convolution with low-latency.

In a low-latency convolution operation as described by [1] however, *both* signal $s(n)$ and impulse response $h(n)$ are windowed into a sequence of blocks (as shown in Figure 8). The convolution operation of a signal frame with an impulse response frame need only occur at the time it is needed for continued sound output. Because the result of a previous lengthy convolution is not stored, the user may modify the impulse response every blocksize of samples, where the blocksize is significantly shorter than the entire length of the impulse response. If the impulse response is parametric, the user's modifications may be heard in real-time, yielding the same level of interactive control as the waveguide model.

5 CONCLUSION

At the expense of computational complexity, the low-latency convolution affords the user some additional advantages to the waveguide model. For example, it is possible to build impulse responses less typical of a physical system, which would be difficult to obtain within a waveguide structure. There is no longer the physical constraint of having impulse response arrival uniformly spaced. Rather, they may be spaced according to some other structure, or interleaved in patterns that create multiple tones, creating multi-phonic or hybrid instruments. The impulse responses are also not constrained by stability in that arrivals must not decay.

This work is ultimately intended to be used in real-time performance, affording modern wind instrument players the ability to extend their instruments in addition to extending their playing technique. The proposed convolutional techniques permits sound exploration that is no longer limited by the physical structure of the instrument being played.

6 REFERENCES

[1] W. G. Gardner, "Efficient convolution without input-output delay," *Journal of the Audio Engineering Society*, vol. 43, no. 3, pp. 127–136, March 1995.

[2] X. Rodet and C. Vergez, "Phyical models of trumpet-like instruments: Detailed behavior and model improvements," in *Proceedings of ICMC 1996*. Clear Water Bay, Hong-Kong: International Computer Music Conference, August 1996.

[3] J. O. Smith, "A basic introduction to digital waveguide synthesis," http://ccrma.stanford.edu/˜jos/swgt/, February 2006.

[4] ——, "Physical audio signal processing for virtual musical instruments and audio effects," http://ccrma.stanford.edu/˜jos/pasp/, December 2008.

[5] T. Smyth and J. Abel, "Convolutional synthesis of wind instruments," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'07)*, New Paltz, New York, October 2007.

[6] ——, "Estimating the reed pulse from clarinet recordings," in *Proceedings of ICMC 2009*, Montreal, Canada, August 2009, forthcoming.

[7] T. Smyth, J. Abel, and J. O. Smith, "A generalized parametric reed model for virtual musical instruments," in *Proceedings of ICMC 2005*. Barcelona, Spain: International Computer Music Conference, September 2005, pp. 347–350.

POLYNOMIAL EXTRAPOLATION FOR PREDICTION OF SURPRISE BASED ON LOUDNESS - A PRELIMINARY STUDY

Hendrik Purwins^{*+},**Piotr Holonowicz***,**Perfecto Herrera***
*Music Technology Group,Universitat Pompeu Fabra, Barcelona
$^{+}$Neural Information Processing Group, Berlin Institute of Technology
hendrik.purwins,piotr.holonowicz,perfecto.herrera@upf.edu

ABSTRACT

The phenomenon of music surprise can be evoked by various musical features, such as intensity, melody, harmony, and rhythm. In this preliminary study we concentrate on the aspect of intensity. We formulate surprise as a critical derivation from the predicted next intensity value, based on the "immediate" past ($\sim$ 7 s), slightly longer than the short-term memory. Higher level cognition, processing the long range structure of the piece and general stylistic knowledge, is not considered by the model. The model consists of a intensity calculation step and a prediction function. As a preprocessing method we compare instantaneous energy (root mean square), loudness, and relative specific loudness. This processing stage is followed by a prediction function for which the following alternative implementations are compared with each other: 1) discrete temporal difference of intensity functions, 2) FIR filter, and 3) polynomial extrapolation. In addition, we experimented with different analysis window length, sampling rate and hop size of the intensity curve. Good results are obtained for loudness and polynomial extrapolation based on an analysis frame of 7 s, a sampling rate of the loudness measures of 1.2 s, and a hop size of 0.6 s. In the polynomial extrapolation a polynomial of degree 2 is fitted to the loudness curve in the analysis window. The absolute difference between the extrapolated next loudness value and the actual value is then calculated and divided by the standard deviation within the analysis window. If the result is above a threshold value we predict surprise. The method is preliminarily evaluated with a few classical music examples.

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

1 INTENSITY CALCULATION AND SURPRISE MEASURES

We compare three representations for the intensity: 1) the instantaneous energy, i.e. the root mean square of the amplitude, 2) the specific loudness in 24 bark bands (Zwicker and Fastl [1999] p.225) and 3) loudness as the sum of the specific loudness across the bark bands. We use the implementation within the IRCAM descriptor (Peeters [2004]).

After experimenting with different frame and hop sizes we chose a hop size of 0.6 s and a frame size of 1.2 s (rectangular window), calculating the intensity or (specific) loudness respectively.

The intensity is fed into a surprise prediction function. The output of such a model yields a curve of "surprisingness". Each surprise curve is normalized with respect to its maximal value. Applying a threshold leads to the binary decision surprise point/ no surprise. We use a threshold of 0.95. We compare four different models. The first two methods (Δ energy and Δ loudness) are defined by taking the differences of consecutive samples from energy or loudness. The third method is an FIR filter across the sampled loudness within an analysis frame of 7 s. The forth method is based on polynomial regression of the sampled loudness. Across a time frame of 7 seconds a regression polynomial is calculated and used for extrapolating the subsequent loudness value. In addition, the root mean square approximation error σ is calculated across this window. The value of "surprisingness" is calculated by dividing the deviation of the extrapolated and the actual next loudness value by σ.

2 DATA SET AND EVALUATION METHOD

We use a small set of 8 excerpts of classical music by Haydn (*Symphony No. 94 mit dem Paukenschlag*) Beethoven (*Symphonies 5 & 6*), Strauss (*Tod und Verklärung*), and Rossini (*Guillaume Tell*). The surprise points have been manually annotated by a subject

Method	F-Measure
Δ Energy	0.71
Δ Loudness	0.75
Filter	0.25
Polynomial Regression	0.83

Table 1: Comparison of surprise measures.

who had 13 years of violin instruction. The subject listened to the excerpts several times to determine the point in time when he was maximally surprised, expressed by high attention that this moment called and/or a gooseflesh shortly after, due to the dynamics of the piece when listening to it for the first time.

We adopt the usual evaluation method in music information retrieval for onset detection to surprise by transferring it to a larger time scale. The predicted surprise points are compared to the annotated surprise points. Within a tolerance range of $\pm 0.6s$, coinciding annotated and predicted surprise points are considered as correct hits. Multiple hits within the tolerance window are considered as simple hits. Then precision, recall, and f-measure are calculated.

3 EVALUATION AND EXAMPLES

Table 1 shows that the polynomial regression method works best. Due to its psychoacoustical justification, the Δ loudness method works better than the Δ energy method. As a demonstrate we show the analysis of two sound examples in Figures 1 and 2.

4 CONCLUSION AND DISCUSSION

The use of the polynomial regression model allows us to distinguish a continuous *crescendo* from an abrupt *subito forte*. Due to the rather short analysis window of 7 s (which is slightly longer than what is considered the short term memory) only surprise effects are considered that reflect a direct reaction to the sound. Surprise that is due to longer range structure or due to knowledge of stylistic particularities cannot be considered by the model suggested here. The optimal threshold of the polynomial regression model could be learned on pieces of a particular style and evaluated on a hold-out set of pieces of the same style. Other loudness models can be compared with the ones used here. It would help to use more sophisticated prediction models, e.g. such that consider periodic regularities in the loudness. Possible candidates would be autocorrelation or wavelets. Another effect to be taken into account is that the repetition of a surprising passage is less surprising. Therefore

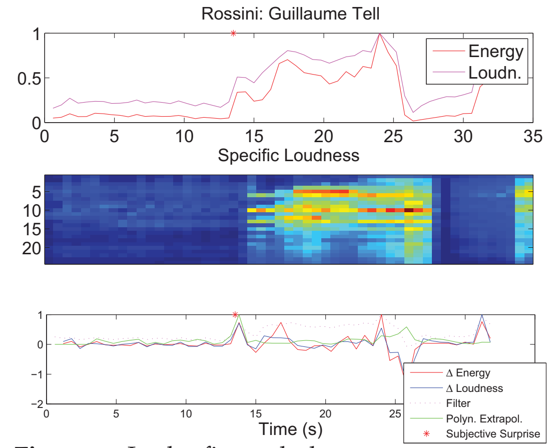


Figure 1: In the first subplot, root mean square energy and loudness is displayed. The specific loudness reveals the perceptually relevant energy distribution across the Bark bands. For surprise prediction, the loudness proofs more useful than the channel-wise specific loudness. The lowest subplot shows the various surprise measures. We observe a pronounced peak of the polynomial extrapolation method at the subjective surprise point with low values otherwise. On the contrary, the Δ methods yield high surprise values also at other points in time.

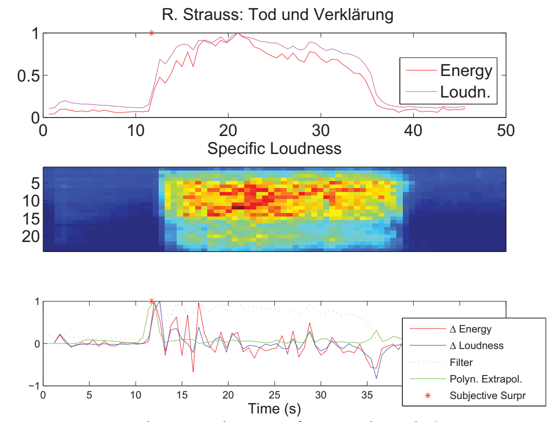


Figure 2: The analysis of a Richard Strauss excerpt (cf. Figure 1). Oscillating behavior of the Δ methods can be seen.

patterns of temporal development leading to a surprise point should be stored and time aligned to future surprise candidates. This could be performed by Dynamic Time Warping. For considering multiple context dependencies, Bayesian networks provide a useful methodology.

Acknowledgments

This research was supported by the European projects EmCAP (Emergent Cognition through Active Perception, FP6-IST, contract 013123) and CLOSED (Closing the Loop of Sound Evaluation and Design, FP6-NEST-PATH, project no. 29085). The first author received support from a Juan de la Cierva scholarship from the Spanish Ministry of Education and Science.

References

Peeters, G. (2004). A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical report, IRCAM, Analysis/Syntesis Team.

Zwicker, E. and Fastl, H. (1999). *Psychoacoustics – Facts and Models*. Springer, 2nd edition.

COMPUTATIONAL INVESTIGATIONS INTO BETWEEN-HAND SYNCHRONIZATION IN PIANO PLAYING: MAGALOFF'S COMPLETE CHOPIN

Werner Goebl, Sebastian Flossmann, and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz, Austria

ABSTRACT

The paper reports on first steps towards automated computational analysis of a unique and unprecedented corpus of symbolic performance data. In particular, we focus on between-hand asynchronies – an expressive device that plays an important role particularly in Romantic music, but has not been analyzed quantitatively in any substantial way. The historic data were derived from performances by the renowned pianist Nikita Magaloff, who played the complete work of Chopin live on stage, on a computer-controlled grand piano. The mere size of this corpus (over 320,000 performed notes or almost 10 hours of continuous performance) challenges existing analysis approaches. The computational steps include score extraction, score-performance matching, definition and measurement of the analyzed features, and a computational visualization tool. We then present preliminary data to demonstrate the potential of our approach for future computational modeling and its application in computational musicology.

1 INTRODUCTION

This paper presents research towards automated computational analysis of large corpora of music performance data. In particular, we give a first report on a computational approach to analyzing a unique corpus of historic performance data: basically the complete work of Chopin, performed by the renowned pianist Nikita Magaloff. Corpora of that size – hundreds of thousands of played notes – require a level of automation of analysis that has not been accomplished so far. We describe the required processing steps, from converting scanned scores into symbolic notation, to score-performance matching, definition and automatic measurement of between-hand asynchronies, and a computational visualization tool for exploring and understanding the extracted information.

SMC 2009, July 23–25, Porto, Portugal

As the two hands of a pianist have the possibility to produce different musical parts independently, the between-hand asynchronies yield a spectrum of artistic expression ranging from the fairly constrained "melody lead" effect [1] to bass anticipations [5], or the "earlier type of tempo rubato" [3]. Towards the end of the paper, we present preliminary results on the between-hand asynchronies in Magaloff's complete Chopin to demonstrate the scope of insights that such large corpora can offer. Finally, we discuss the future pathways of this research endeavor and its potential for computational modeling and musicological investigation.

2 THE CHOPIN CORPUS

The analyzed Chopin corpus comprises live concert performances by the Georgian-Russian pianist Nikita Magaloff (1912–1992), who played almost the entire solo repertoire of Chopin in a series of 6 recitals between January and May 1989 at the *Mozart-Saal* of the *Wiener Konzerthaus* [1] in Vienna on a Bösendorfer computer-controlled grand piano. In this unprecedented project, Magaloff, by that time already 77 years old, performed all works of Chopin for solo piano that appeared in print during Chopin's life time, keeping a strict ascending order by opus number, starting with the *Rondo*, Op. 1 up to the three *Waltzes* Op. 64, including the three Sonatas, 41 Mazurkas, 25 Préludes, 24 Études, 18 Nocturnes, 8 Waltzes, 6 Polonaises, 4 Scherzos, 4 Ballades, 3 Impromptus, 3 Rondos, and other works (Variations brillantes, Bolero, Tarantelle, Allegro de Concert, Fantaisie, Berceuse, Barcarole, and Polonaise-Fantaisie). [2]

Magaloff played these recitals on a Bösendorfer SE computer-controlled grand piano that recorded his performances onto computer hard disk. The SE format stores the performance information in a symbolic format with high precision, providing detailed information on the onset and off-

[1] This concert hall provides about 700 seats, `http://www.konzerthaus.at`.

[2] The works not played were either piano works with orchestra accompaniment (Op. 2, 11, 13, 14, 21, and 22), works with other instruments (Op. 3, 8 and 65), or works with higher (*op. posth.*, starting from Op. 66, the *Fantaisie-Impromptu*) or no opus numbers.

set timing of each played tone, the dynamics, and the pedalling. The entire corpus comprises more than 150 individual pieces, over 323,000 performed notes or almost 10 hours of continuous performance.

3 COMPUTATIONAL ANALYSIS OF PERFORMANCE DATA

3.1 Score Extraction

In order to analyze symbolic performance data automatically, the performances have to be connected to the corresponding musical scores (score-performance matching). As symbolic scores were not available for the complete work of Chopin, the first step was to extract this information from the printed music scores. We used a music recognition software ("SharpEye") to convert the 946 pages of scanned music into a musicXML representation. Extensive manual verification of the conversion process was necessary to eliminate a large number of conversion errors, as well as post-correction of conversion incapabilities of the used software (e.g., ottava lines, parts crossing staffs, etc.).

3.2 Score–Performance Matching

The symbolic scores were then matched to Magaloff's performances employing a matching algorithm based on an edit distance metric [4]. Also the matching results had to be inspected and corrected manually with an interactive graphical user interface that displays the note-by-note match between the score information and the performance. All uncorrectly played notes or performed variants were identified and labeled. (This, by the way, will also make it possible to perform large-scale, in-depth analyses of the kinds of errors accomplished pianists make.)

3.3 Defining and Measuring Asynchronies

Our aim was to analyze the between-hand asynchronies of notes that are notated as nominally simultaneous in the score (that is, all tones belonging to the same 'score event'). To that end, we first needed to compute these asynchronies automatically from the corpus.

The staff information of the musical notation (upper versus lower staff) was used to calculate the between-hand asynchronies. As the performance data does not contain information about what hand played what parts of the music, we assumed that overall the right hand played the upper staff tones and the left hand the lower. [3]

We define a between-hand asynchrony as follows: *For all notes belonging to the same score event, subtract the onset timing of each upper-staff note from the onset timing of each lower-staff note* ("lower minus upper"). Thus, positive asynchrony values indicate that the upper staff (right hand) is early, while negative numbers denote bass anticipations (left hand early). Multiple asynchronies within one score event were averaged for subsequent data analyses.

All notated arpeggios, ornaments, trills, or grace notes were excluded from our preliminary data analysis (about 10% of the entire data), as these cases feature special and usually larger asynchronies than 'regular' score events. These special cases deserve a separate detailed analysis that would exceed the scope of the present paper.

3.4 Computational Visualization

For a first intuitive analysis and understanding of this huge amount of measurement data, adequate visualization methods are needed. Thus, we developed a dedicated computational visualization tool. A screenshot is presented in Figure 1. It comprises three panels on top of each other, sharing the same time axis. The upper panel shows the individual tempo curves of the two hands; the middle panel shows the average asynchronies for each score event that contained simultaneous notes in each staff; and the lower panel features a piano-roll representation of the performances with the relevant notes connected by vertical lines. The color of these lines is either red (indicating a right-hand lead) or green (indicating a left-hand lead). The grey area in the middle panel marks a range of +/–30 ms within which asynchronies are not likely to be perceived as such [2]. Furthermore, the tool indicates occurrences of bass anticipations ("B.A.," lower panel) and out-of-sync regions (horizontal bars, middle panel; for description see below).

4 PRELIMINARY RESULTS

In the following, we give a first glimpse of the scope of results that such large-scale analyses may yield.

4.1 Overall Asynchronies

The distribution of all asynchronies between the two hands is shown in Figure 2, including the mean and the mode value. The positive mode value reflects an overall tendency for the right hand to be early, which is most likely attributable to the well-known 'melody lead' effect [1]. Moreover, the mean value is slightly below the mode value reflecting a slightly skewed histogram towards the left side. Particularly in the region of –100 to –300 ms there is a slight increase of values, most likely due to frequent bass anticipations (red line, see below).

The asynchrony distributions of the individual pieces vary considerably and may depend on the specifics of the pieces. The pieces played most synchronously by Magaloff are those

[3] Certainly, there are numerous passages where this simple assumption is wrong or not likely to be true, but given the sheer size of the data set, the potential bias may be tolerable.

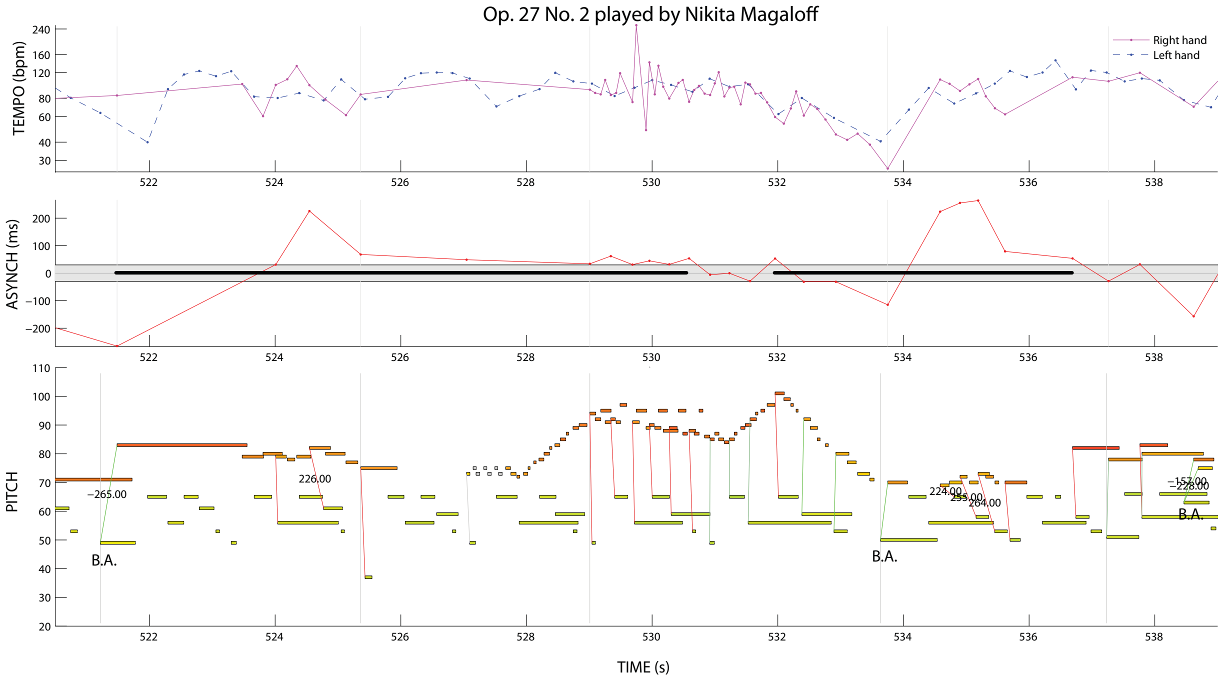


Figure 1. Screenshot of a computational visualization tool showing bars 50–54 of Chopin's Nocturne Op. 27 No. 2 performed by Magaloff. The tempo curves of the two hands are plotted in the upper panel, the asynchronies in middle (positive values indicate an early right hand; negative an early left hand; the grey area sketches the +/–30-ms region around zero), and a piano roll representation is shown in the lower panel. All tones of computed asynchronies are connected by (almost) vertical lines that are plotted in red when the melody (right hand) was ahead, green when it lagged.

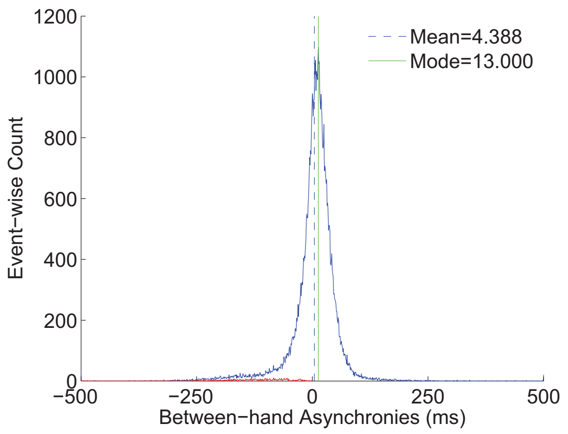


Figure 2. Histogram of the signed between-hand asynchronies per event over the entire Chopin corpus (displaying a total of 163,208 asynchronies). The distribution of bass anticipations (see Section 4.2) is drawn by a red line on the left part of the histogram.

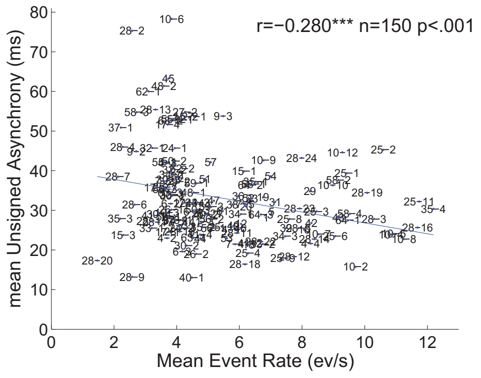
that feature predominantly chordal textures (Op. 40-1, 28-9, 28-20, 10-2, see Figure 3); the least synchronous pieces are those that have strong melodic textures that leave room for artistic interpretation.

There is a significant tempo effect within the investigated pieces. Figure 3 shows the mean absolute (unsigned) asynchronies per piece (a) and the standard error of the asynchronies (b) against the average event rate (in events per second). An event rate value was computed for each score event by counting the performed events (chords) within a time window of 3 seconds around it. The average event rate is the piece-wise mean of those values. We found that the faster the piece, the lower is the absolute asynchrony and also the lower is the variability of the asynchronies, which suggests that Magaloff takes more room to employ "expressive" asynchronies in slower pieces than faster pieces.

4.2 Bass Anticipations

A bass anticipation is labeled as such when the lowest tone of a lower-staff score event is more than 50 ms ahead of the mean onsets of the upper-staff tones of that event. The

a)



b)

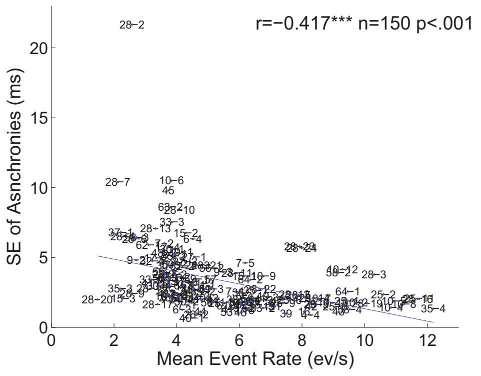


Figure 3. Absolute (unsigned) asynchronies (a) and standard error (SE) of the mean asynchronies (b) against the mean event rate per piece. The numbers refer to the opus numbering of the pieces.

overall distribution of the bass leads is shown in Figure 2 (red line) on the left side of the histogram, and the invididual pieces are shown in Figure 4. The proportion of bass anticipations is lowest on average for the Etudes, the Preludes, and the Rondos (well below an average of 1% of the events), and highest in the Mazurkas and the Nocturnes (almost 2%). No bass anticipations were found particularly in the Preludes (16 out of 25 did not contain bass anticipations) and the Etudes (7 of 24).

An exception is the Prelude Op. 28 No. 2, which exhibits both the highest mean asynchronies and the largest proportion of bass anticipations among all pieces (clearly visible in Figures 3 and 4). This very slow and short piece features a constant 1/8-note accompaniment with a single-note melody above it. The sad character and the slow tempo may be the reason for the high temporal independence of the melody in

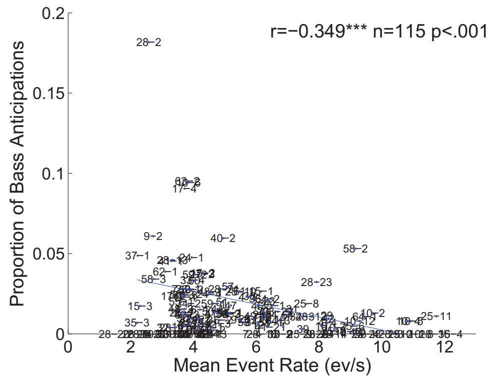


Figure 4. Proportion of bass anticipations against mean event rate per piece. Zero proportions (34 pieces) were excluded from the calculation of the regression line.

Magaloff's performance.

There is also an effect of event rate, suggesting that bass leads become less frequent as the tempo of the pieces increases (Figure 4). Again, slower pieces leave more room for expressive freedom than do faster pieces. To further analyze the occurrences of bass anticipations, we categorized all score events bar-wise into first beats, on-beats (all beat events except the first beat), and off-beats. It turns out that metrical position has a significant effect: the highest number of bass anticipations fall on the first beat (1.80%); other on-beat events receive 1.48% bass anticipations, and 0.66% are found on off-beat events. This suggests that bass anticipations are used by Magaloff to emphasize predominantly strong beats.

4.3 The Earlier Type of Tempo Rubato

An expressive means that has a long performance tradition is the "*tempo rubato* in the earlier meaning" [3]. It refers to expressive temporal deviations of the melody line, while the accompaniment, offering the temporal reference frame, remains strictly in time. Chopin in particular often recommended his pupils to keep the accompaniment undisturbed like a conductor, and give the right hand the "freedom of expression with fluctuations of speed" [3, p. 193]. In contrast, the later meaning of tempo rubato was used more and more to refer to the parallel slowing down and speeding up of the entire music (today more generally referred to as *expressive timing*). In expressive performance, both forms of rubato are present simultaneously.

We aim at identifying sequences of earlier tempo rubato automatically from the entire corpus. To extract overall information about sequences where Magaloff apparently employed an earlier tempo rubato, we count the *out-of-sync re-*

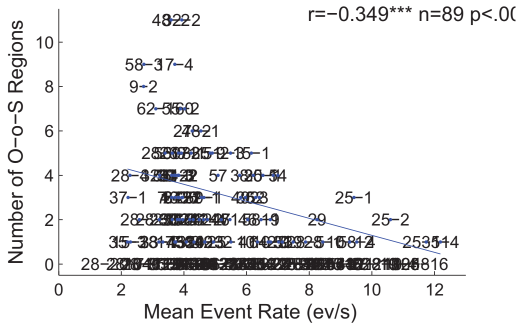


Figure 5. The number of out-of-sync regions (earlier tempo rubato) per piece is plotted against the event rate.

gions of each piece. An out-of-sync region is defined as a sequence of consecutive asynchronies that are larger than the typical perceptual threshold (30 ms) and that contain more elements (events) than the average event rate of that piece (2–13 tones). Usually faster pieces contain many shorter runs that are out-of-sync, but due to the fast tempo, these regions extend only to some fraction of a second. To take this into account, the search for out-of-sync regions is coupled to the piece-wise average event rate.

On average, a piece contains 1.8 such regions. The pieces with the lowest numbers are generally the Mazurkas, Preludes, and Etudes (below 1), the pieces with the highest counts are by far the Nocturnes (on average well over 5), suggesting that particularly this genre within Chopin's music leaves most room for letting the melody move freely above the accompaniment. Figure 5 shows the number of out-of-sync regions per piece against the average event rate of the piece. It demonstrates that faster pieces contain fewer such regions, suggesting that this form of tempo rubato is bound to slower and medium tempi (such as the Nocturnes, the slowest piece category in the Chopin corpus). This overall finding is not surprising; the earlier tempo rubato is expected to be more often found in melodic contexts rather than in virtuoso pieces, as the historic origins of the earlier tempo rubato go back to vocal music.

To illustrate, the example of the visualization tool presented in Figure 1 is briefly discussed. It shows an excerpt (bars 50–54) of the Nocturne Op. 27 No. 2. This example contains two runs of tempo rubato as determined by the algorithm (indicated by horizontal bars in the middle panel). The first starts on the downbeat of bar 50, where Magaloff delayed the melody note by 265 ms, only to be early over the next few notes of the descending triplet passage. The beginning of the 48-tuplet figure (which is interpreted as 1/16-note triplets as well) also leads the accompaniment. Towards its end, the second run of tempo rubato as determined by our algorithm begins, just when Magaloff starts to lag behind the accompaniment. This lag coincides with a downward motion and a notated decrescendo. The following embellishment of the b-flat (notated as 1/32 notes and 1/32 triplets) is again clearly ahead of the accompaniment. The first note of the next phrase is also ahead, potentially to underline the notated anticipation of the upcoming harmony change towards e-flat minor.

5 SUMMARY AND FUTURE WORK

This paper presented a computational approach to making large performance corpora accessible for detailed analysis. It defined and automatically measured between-hand synchronization in over 150 pieces by Frédéric Chopin. Working with data sets of that size, i.e., complete sets of performances of a single composer or several hundred thousand played notes requires, among other things, effective score-performance matching algorithms and interactive graphical user interfaces for post-hoc data inspection and correction.

Preliminary data analysis of the between-hand synchronization attempted to demonstrate the rich use of asynchronies in Magaloff's complete Chopin corpus, a historic document of a unique performance project that offers unequaled opportunities for performance analysis. We sketched overall trends of asynchronicity with respect to pieces, tempo, and metrical constraints, as well as specific cases of bass anticipations and occurrences of tempo rubato in its earlier meaning.

This research endeavor is preliminary as it stands. It is meant to lead to further modeling efforts to be able to predict asynchronies from Romantic scores following Nikita Magaloff's instrinsic style. Piece category, overall musical texture, as well as local and global aspects from the scores may be promising features for training machine learning algorithms in order to derive predictive computational models of between-hand asynchronization. Such models may enhance existing performance rendering systems by adding an important expressive feature.

To be able to automatically assess performance corpora of this scale offers completely new pathways for computational musicology. Historic documents such as the present corpus are in managable reach for detailed analysis. Other large corpora, such as piano rolls of historic reproducing pianos or the performance database of the Yamaha eCompetition [4], may be other sources for future large-scale performance investigation.

Finally, detailed knowledge derived from performances by established musicians may help us develop real-time visualization tools that give intelligent feedback to practicing piano students to enhance their awareness of what they are doing, and potentially help them to improve their play instantly.

[4] `http://www.piano-e-competition.com`

6 ACKNOWLEDGEMENTS

This research was supported by the Austrian Research Fund (FWF, grant P19349-N15). We are indebted to Mme. Irène Magaloff for her generous permission to use her late husband's performance data for our research.

7 REFERENCES

[1] GOEBL, W. Melody lead in piano performance: Expressive device or artifact? *Journal of the Acoustical Society of America 110*, 1 (2001), 563–572.

[2] GOEBL, W., AND PARNCUTT, R. The influence of relative intensity on the perception of onset asynchronies. In *Proceedings of the 7th International Conference on Music Perception and Cognition, Sydney (ICMPC7), Aug. 17–21, 2002* (Adelaide, 2002), C. Stevens, D. Burnham, G. McPherson, E. Schubert, and J. Renwick, Eds., Causal Productions, pp. 613–616.

[3] HUDSON, R. *Stolen Time: The History of Tempo Rubato*. Clarendon Press, Oxford, 1994.

[4] MONGEAU, M., AND SANKOFF, D. Comparison of musical sequences. *Computers and the Humanities 24* (1990), 161–175.

[5] VERNON, L. N. Synchronization of chords in artistic piano music. In *Objective Analysis of Musical Performance*, C. E. Seashore, Ed., vol. IV of *Studies in the Psychology of Music*. University Press, Iowa City, 1936, pp. 306–345.

SOUND OBJECT CLASSIFICATION FOR SYMBOLIC AUDIO MOSAICING: A PROOF-OF-CONCEPT

Jordi Janer, Martin Haro, Gerard Roma
Universitat Pompeu Fabra
{firstname.lastname}@upf.edu

Takuya Fujishima
Yamaha Corporation
fujishim@beat.yamaha.co.jp

Naoaki Kojima
Media Artist
animatedmotion@gmail.com

ABSTRACT

Sample-based music composition often involves the task of manually searching appropriate samples from existing audio. Audio mosaicing can be regarded as a way to automatize this process by specifying the desired audio attributes, so that sound snippets that match these attributes are concatenated in a synthesis engine. These attributes are typically derived from a *target* audio sequence, which might limit the musical control of the user.

In our approach, we replace the target audio sequence by a symbolic sequence constructed with pre-defined sound object categories. These sound objects are extracted by means of automatic classification techniques. Three steps are involved in the sound object extraction process: supervised training, automatic classification and user-assisted selection. Two sound object categories are considered: *percussive* and *noisy*. We present an analysis/synthesis framework, where the user explores first a song collection using symbolic concepts to create a set of sound objects. Then, the selected sound objects are used in a performance environment based on a loop-sequencer paradigm.

1 INTRODUCTION

Corpus based sound synthesis encompasses a wide range of approaches that leverage the developments of audio description technologies for speech and music synthesis applications. Initially developed for text to speech applications, Concatenative Sound Synthesis [1] has been adapted to the musical domain, notably to singing voice and musical instrument synthesis. Musical mosaicing [2] introduced a more general approach from the point of view of unit selection by mapping the search of suitable sounds from the database to a constraint satisfaction problem. However, while traditionally closer to popular practices in the reuse of musical materials, applications of musical mosaicing have been limited by the dependence on an acoustic target that is analyzed to define the constraints. In this paper we present an initial step towards symbolic audio mosaicing based on machine learning of abstract sound categories. The system describes acoustic events in polyphonic music using human-understandable concepts (e.g. percussive, having prominence of singing voice, harmonic sound with constant pitch, etc.). Then it provides the user with a symbolic audio mosaicing interface to compose music by concatenating these "concepts". Sound descriptors from the symbolic sequence are used as target in an audio mosaicing system. One of the main bottlenecks we face when trying to develop such system is the need to automatically characterize music segments using perceptually meaningful sound object categories. Nowadays we have a lot of signal-level descriptors, but these descriptors are rarely linked with perceptually meaningful sound events in polyphonic music (with few exceptions like chroma features or some rhythmic descriptors). We make use of Music Information Retrieval (MIR) techniques like sound segmentation [3], instrument classification [4], and audio feature extraction [5] to classify polyphonic sound segments into pre-defined sound object categories.

Depending on the area of research we look from, there are several interpretations of the concept of sound object. If we look from a perceptually-oriented point of view we find concepts related to auditory streams and perceptual grouping of sound events [6]. Looking from a signal processing point of view we can rely on concepts like onsets or ADSR envelope. Traditional music theory deals the concepts of notes and chords. In *Musique Concrète*, Schaeffer proposed the idea of musical objects [7]. One simple working definition can be found in [8]: "Sound Object: a basic unit of musical structure, generalizing the traditional concept of note to include complex and mutating sound events on a time scale ranging from a fraction of a second to several seconds". Within this proof-of-concept paper we start by considering only two easily identifiable types of sound objects found in polyphonic music: *percussive* and *noisy* sounds. The former category includes sounds objects with a sharp attack followed by a decay (e.g. drums, pizzicato), while the latter includes *unpitched* sound objects with flat amplitude and spectral envelopes (e.g. stable white noise, highly distorted sounds).

The main characteristic of this project is the concept of semi-automatic sound object retrieval. The research tasks focus on detecting segments of a song with high probability of being member of a pre-defined sound category. At the same time, with the analyzed sounds, the user should be

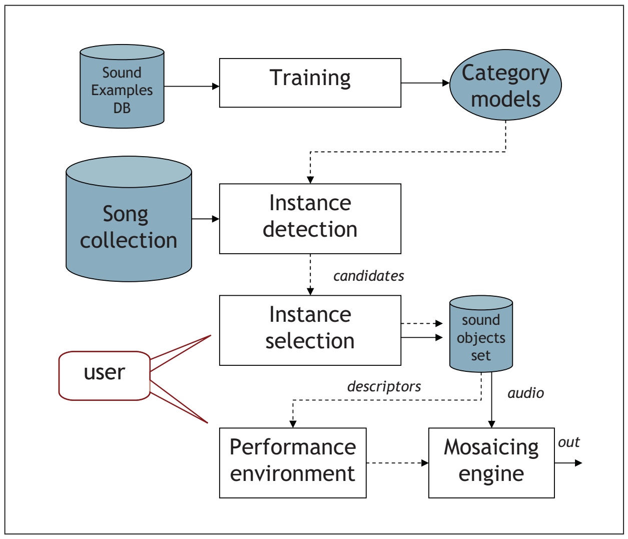


Figure 1. General overview of the implemented system. Note the user intervention in both exploration and composition processes.

able to create new audio material in the context of electronic music performances.

An overview of the process is depicted in figure 1. The steps involved are: *Instance Detection*, *Instance Selection* and *Performance Environment*. The first consists of two sub-processes: the model training, given a manual annotated ground-truth database; and the extraction of new sound object candidates from a song collection. In the instance selection process, the user selects among the extracted candidates in a GUI. Finally, the performance interface allows the user to specify different steps in a sequencer using sound categories. In the next sections we describe each step in detail.

2 AUTOMATIC INSTANCE DETECTION

In the proposed analysis framework, the user selects from a list of automatically detected sound objects. In order to detect these objects the system extracts acoustic descriptors from all songs in the collection, and creates a list of candidates. The instance detection process applies machine learning techniques to classify the list of sound candidates into pre-defined categories, outputting a class-label and a likelihood value for every proposed segment. Additional features, such as amplitude envelope and spectral content, are computed to further describe the sound objects. These features are used in the following stage, the sound object selection (see section 3).

2.1 Segment-based Sound Object Detection

Initially, one can foresee two different approaches to detect sound objects: frame-based and segment-based. In a frame-based approach sound objects can be identified by using a continuous descriptor computed out of frames of short duration. In a segment-based approach segments of longer duration are described and evaluated by an automatic classifier.

The problem of using a frame-based approach is that it ignores time-evolving aspects of the sound, such as the energy envelope. Therefore, our approach is segment-based, relying on supervised machine learning techniques. A manually annotated ground truth database is used to train models of sound object categories. From the audio file, we use an in-house sound analysis library to extract low-level acoustic features. In the next step we generate a list of segment candidates, which can overlap. For each candidate, a confidence measure is computed using automatic classification algorithms. We keep the non-overlapping segments with a confidence measure of more than 50%.

2.1.1 Training Database

In order to build a model for each sound category we need a ground truth database with labeled examples. We decide to construct two labeled databases (one per sound category) namely PercussionDB and NoiseDB.

PercussionDB: Concerning the *percussive* category, in order to obtain a more generalized model, we use a ground truth database built by processing the *ENST drums* database [9]. This is the largest publicly available drum database. It contains recordings from three different drummers and drum sets playing single hits, drum phrases and complete songs covering various styles. The authors provide two type of drum recording tracks namely dry (without sound effects) and wet tracks, along with the corresponding music accompaniments. In our case, since we want to detect percussive events in real world music, we use the wet tracks. In order to obtain "realistic" songs we mix the drums and their accompaniments tracks directly (without further changes of sound levels). Afterwards we segment 30 seconds of each song (and their labels) for a total of 64 songs. Since we want to detect all percussive sounds as belonging to one class we merge all the provided labels into one "parent" category named as "*percussive*". Finally we keep the sound events that are found by an onset detector [3], labelling intra-onset segments with a maximum length of 150ms. If the onset has at least one percussive label in its vicinity ($\pm$40ms) it is labeled as "*percussive*" otherwise is labeled as "*non-percussive*". At the end of this process we obtain 3.690 examples of percussive events (and 3.690 of non-percussive ones).

NoiseDB: Concerning the *noisy* category, we collected several songs containing potential "noisy" sounds, as well as isolated noise sound from sample collections. We manually

Amplitude-related object descriptors
Mean
Variance
Minimum
Maximum
Skewness
Kurtosis
Time-related object descriptors
Temporal Skewness
Temporal Kurtosis
Temporal Centroid
Maximum Normalized Position (MxNP)
Minimum Normalized Position
Slope: arc tangent of the slope of the linear regression of the data.
Normalized Attack (Decay): Slope form beginning (end) to the MxNP.
Attack (Decay): Slope from beginning (end) to Max position (in frames).

Table 1. Object-level descriptors

annotate the noisy objects obtaining a total of 208 "*noisy*" sound objects.

2.1.2 Audio Descriptors

To characterize each sound object we first compute more than 90 frame-level audio descriptors. Then we compute several object-level descriptors to extract amplitude-related and time-related features from the time series of audio frames. For each frame-level descriptor, we compute the whole set of object-level features depicted in table 1. Thus, we obtain more than 1.400 descriptors for each sound object. A more detailed explanation of this process can be found in [10].

2.1.3 Classification Experiments

Once we have properly labeled databases (i.e. PercussionDB and NoiseDB) we use them to train several supervised classification algorithms.

For classification experiments we first perform a Correlation-based Feature Selection [11] (CFS) on each database to retain the most informative features (those with low intracorrelation and, at the same time, high class- correlation). Then we train both Support Vector Machines (SVM) and Logistic Regression [12] (LR) algorithms on each binary class problem (i.e percussive vs. non-percussive and noisy vs. non-noisy) using 10-fold cross validation in WEKA [1].

Since we obtain quite similar results from SVM and LR, we adopted the later for simplicity. See table 2 for an overview on classification results. From the evaluation of the classification experiments, we can conclude that the model for the *percussive* category achieves good results (F-measure of 0.71) using a data set of more than 7.300 instances. We consider that this model is sufficiently general for classifying new *percussive* sound objects. Regarding the *noisy* model we also obtain good classification results (F-measure of 0.88). Although it would be interesting to have more examples to train the model we consider that this model is quite representative for *noisy* sound objects.

[1] http://www.cs.waikato.ac.nz/ml/weka/

Class	# instances	# descriptors	F-measure
P vs. N-P	7.380	74	0.71
N vs. N-N	416	38	0.88

Table 2. Classification results for percussive vs. non-percussive (P vs. N-P) and noisy vs. non-noisy (N vs. N-N) classes

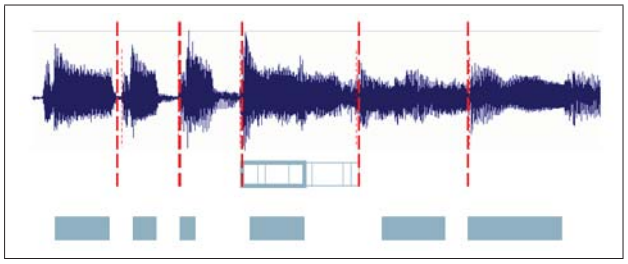


Figure 2. Audio onset segmentation (dashed lines), variable window length segmentation (empty squares), and sound object candidates (solid squares).

At the end of this process we have two LR models, one in charge of detecting *percussive* sound events and the other in charge of detecting *noisy* events. Since the LR algorithm outputs a class label and its corresponding probability measure, we store this probability to be used as a measure of confidence in the prediction.

2.1.4 Candidate Detection

In order to detect the *percussive* or *noisy* sound objects (and their confidence values) we need to evaluate all possible sound segments against our classification models. For every new song we compute its onsets and define sound segment boundaries as inter-onset intervals (see figure 2).

After the segmentation step we compute all possible candidate segments between two onsets, using a frame resolution of 512 samples. Additionally, we use category-specific rules in order to reduce the number of candidates and improve the performance of the classification. For the *percussive* sound category, the possible candidate segments must start at the onset position. Conversely, for the *noisy* category, candidates are not allowed to start at onset position in order to avoid impulsive segments, and a minimum length is required so that they can be perceived as stable sounds.

At the end of the process, we obtain a list of sound object instance candidates and their confidence value. We also compute some additional object-level descriptors to be used in the assisted instance selection process described next. For *percussive* objects we compute their attack, decay and magnitude values and for the *noisy* category we compute spectral energy values for high, mid and low frequency bands.

3 ASSISTED INSTANCE SELECTION

3.1 Overview

The main feature of the selection prototype is to discover sound objects in a song collection. Given a song collection, we compute a list of candidate objects for each song. When a song is loaded in the main panel, the user can play the candidates, which are highlighted in different colors for *percussive* and *noisy* categories. Also, the user may restrict the found sound objects by setting a threshold, so that only those sounds with a confidence value above that threshold will be highlighted. Additional filtering can be done according to specific characteristics of each sound category. For example, for *percussive* sound objects the user might set values for attack, decay and magnitude and, according to the selected parameters, a triangular shape is drawn (see figure 3). When applying this filter, only those candidates whose attack, decay and magnitude values are within the ranges defined by the corresponding parameters and the tolerance slider are highlighted. *Noisy* sound objects may be filtered according to the distribution of energy in the spectrum. A dual slider allows specifying the relative proportion of high, mid and low frequency energy. A second slider specifying the desired total absolute level of energy. A rough representation of the target spectrum shape defined by these parameters is also displayed.

Sound object candidates may be added to a list using a context menu. The list represents the selection of sound objects that the user can export to the performance environment (section 4).

Additionally, a list of similar objects in the whole song collection is computed in advance for each candidate, using Euclidean distance. A second list shows the most similar objects for the currently selected candidate.

4 PERFORMANCE ENVIRONMENT

The performance tool consists in a mosaicing system that concatenates sound objects previously selected using the exploration prototype. It is composed of two separate modules: the graphical user interface and the audio engine. The interface displays information about the current status of the synthesis process, and sends sound object descriptors to the

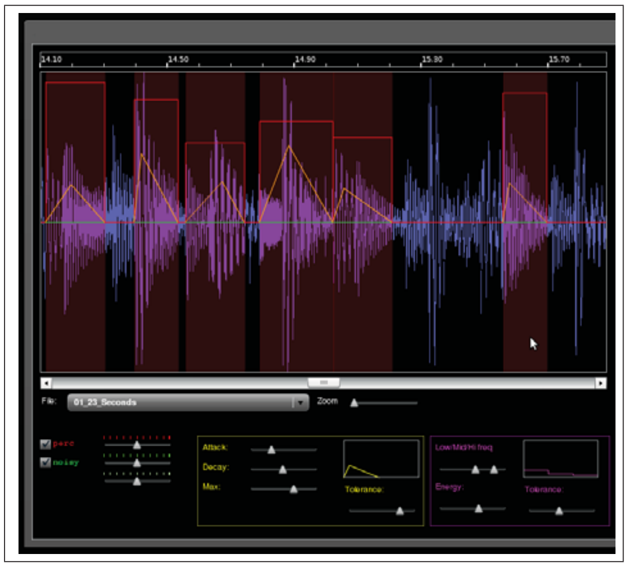


Figure 3. Interface of the exploration prototype. Three sound objects are identified. Amplitude envelope for the percussive objects(red) are represented with triangles.

audio engine through Open Sound Control [13]. The audio engine receives the list of descriptors for each step of the sequence and uses them to select the appropriate sound object for that step.

4.1 Interface

The interface consists of three hierarchical levels: *Composer*, *Sequencer* and *Browser*. Each level can be also related to a different temporal scale: song / performance (*Composer*), loop (*Sequencer*) and sound object (*Browser*). Each *sequencer* has a variable number steps (e.g. sixteen steps in figure 4), and for each step the user can browse and select sound objects of a given category to compose a loop.

In the *Browser*, the user can listen to the sound objects of a given category in a specific sound objects set (previously stored with the selection tool). Typically, the number of objects for one category will be under 100 instances. Sound objects can be scattered according to some signal descriptor (e.g. mapping energy to distance to the center). The user can pre-listen to one sound object at any time before selecting it.

The *Sequencer* is a classical step sequencer. It has a circular shape, showing in real-time the current playing position. The number of steps is specific to a given loop and can be set by the user. Each *Sequencer* can also be rotated in order to modify its relative phase. Each step of the *Sequencer* can be filled with a sound object of one of the defined categories or left empty. The desired target object is selected through the *Browser*. A preset management system allows the user to store, load and unload sequences.

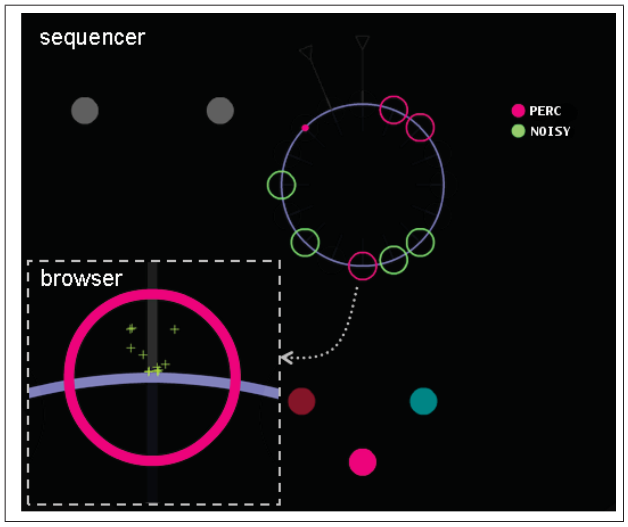


Figure 4. Synthesis interface, showing a *Sequencer* example, and a detailed view of the sound objects *Browser*. The *Composer* level (not shown in the figure) is the top level and can contain multiple *Sequencers*.

The top level is the *Composer*, which might contain one or several *Sequencers* synchronized to a global user-defined tempo. The user can create a composition by manipulating the *Sequencers* (changing location, muting, soloing) on the fly. When having several *Sequencers* in one *Composer* panel, the descriptors sent to the audio engine will depend on the position of the *Sequencers*. For each step, the sent descriptors will be a linear combination of the descriptors of the individual sound objects.

4.2 Audio mosacing engine

The audio mosaicing engine is responsible for the actual sound generation. It selects and concatenates samples from the internal sound bank obtained in the selection process. Each step in a loop is described by the vector of audio descriptors sent by the the interface module. The audio engine seeks the sound objects that best match the incoming descriptors.

One of the main motivations of using audio mosaicing is the flexibility in modifying the synthesized sounds by changing the content of the audio engine's internal sound bank. In this case, the synthesized output will mimic the structure and timbre characteristics of the sound objects used in the *Composer*, but using different sounds. Also, more complex interactions based on the mosaicing paradigm are possible using multiple *Sequencers* as described.

5 CONCLUSIONS AND FUTURE WORK

We have presented and implemented a proof-of-concept prototype for symbolic audio mosaicing. The main idea behind this system is to combine MIR and corpus-based synthesis techniques to obtain a new analysis/synthesis framework for music creation. The proposed application replaces, within the mosaicing paradigm, the target audio sequence by a symbolic sequence constructed with pre-defined sound categories.

We have implemented a fully working prototype considering two sound categories (i.e. *percussive* and *noisy* sounds) automatically detected by machine learning techniques. We believe that this user-assisted application is an engaging interface for audio mosaicing. This application gives the users an alternative to pre-defined sample banks by exploring their own music collections using high level categories.

The next logical step is to add some more categories and perform a user study in order to evaluate the concept. Ultimately, it should be possible for users to define their own sound categories. Concerning the segmentation step, we plan to extend the research by using other descriptors than onset detection. This approach might improve the generation of candidates for some categories.

6 ACKNOWLEDGMENTS

This work is partially funded by Yamaha Corp., Japan and the EU IST project Salero FP6-027122. The authors would like to thank Lucas Kuzma, Perfecto Herrera, Bee Suan Ong and Sebastian Streich.

7 REFERENCES

[1] Schwarz, D. (2004). 'Data-Driven Concatenative Sound Synthesis'. PhD Thesis. Université Paris 6 - Pierre et Marie Curie.

[2] Zils, A. and Pachet, F. (2001). 'Musical Mosaicing'. In Proc. Of the COST-G6 Workshop on Digital Audio Effects (DAFx-01), Limerick.

[3] Brossier, P. (2007) 'Automatic Annotation of Musical Audio for Interactive Applications,'Centre for Digital Music, Queen Mary University of London 2007

[4] Herrera, P. Klapuri, A. Davy, M. (2006) 'Automatic classification of pitched musical instrument sounds' in Signal processing methods for music transcription, Springer, 2006

[5] Peeters, G. (2003). 'A large set of audio features for sound description (similarity and classification) in the Cuidado project'. IRCAM.

[6] Bregman, A. S. (1990) 'Auditory scene analysis: the perceptual organization of sound', Cambridge, Mass. : MIT Press, 773.

[7] Schaeffer, P. (1966), 'Traité des objets Musicaux'. Paris. : Seuil.

[8] Roads, C. (2001), 'Microsound'. Cambridge, Mass. : MIT Press, 409.

[9] Gillet, O. and Richard, G. (2006) 'ENST-Drums: an extensive audio-visual database for drum' ISMIR, 156-159.

[10] Haro, M. (2008) 'Detecting and Describing Percussive Events in Polyphonic Music', Master Thesis, UPF, Barcelona.

[11] Hall, M. (2000) 'Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning', Proc. 17th International Conf. on Machine Learning, 359–366.

[12] le Cessie, S. and van Houwelingen, J (1992) 'Ridge Estimators in Logistic Regression', Applied Statistics, 41(1):191-201.

[13] Kuzma, L. (2008) 'An Interface for Sequencing with Concatenative Sound Synthesis', Master Thesis, UPF, Barcelona.

TOWARDS A PROSODY MODEL OF ATTIC TRAGIC POETRY: FROM LOGOS TO MOUSIKÉ

Anastasia Georgaki
Department of Music
University of Athens, Greece
georgaki@music.uoa.gr

Stelios Psaroudakēs
Department of Music
University of Athens, Greece
spsaroud@music.uoa.gr

Martin Carlé
Institute of Music and Media Studies
Humboldt-University, Berlin
martin.carle@culture.hu-berlin.de

Panagiotis Tzevelekos
Department of Informatics and Telecommunications, University of Athens, Greece
taktzev@di.uoa.gr

ABSTRACT

Recently there has been increasing interest of scientists for the performance of singing or reciting voices of the past in utilising analysis-synthesis methods. In the domain of Ancient Greek musicology indeed, where we find the roots of the occidental music, the main research has been done mostly by scholars of classical Greek literature. However, there is still a vast territory for research in audio performances to be carried out with the help of new digital technologies.

In this paper, we will present an attempt to decode a recited text of Ancient Greek tragedy and render it into sound. At the first paragraph of this article we underline the origin of music arising from the melodicity of speech in Ancient Greek tragedy. In the second paragraph, we describe the methodology we have used in order to analyse the voice of S. Psaroudakēs, himself professor of Ancient Greek music, by an open source prosodic feature extraction tool based on *Praat*. We give a description of the prosodic analysis, implementation details and discuss its feature extension capabilities as well. Last, we refer to the difference between the Ancient and Modern Greek phonological system, the application of this research in music and further development.

1. INTRODUCTION

Our sources on the pronunciation of ancient Hellenic speech (at least, classical Attic) mention two main “prosodies”:

1) syllabic duration (long, short), and
2) musical accents (acute, grave, circumflex).
3) To this phonetic picture modern scholarship adds another prosody: stress accent.

Based on this information, an attempt has been made by Professor S. Psaroudakēs, to reconstruct the sound (intonation) of poetic recitation in an extract from the parodos of Aeschylus’ *Agamemnon* (ll.40-46) [1].

The present paper describes and analyses the recorded vocal contours. It concentrates especially on the pitch contour of an expert voice in order to explore the melodicity of Ancient Greek poetry according to an analysis of the theoretical sources and possible alternatives or modifications of their application in performance. Particularly with regard to the insight of the pre-semiotic linguist Wilhelm von Humboldt which he gained as he was *metrically* translating "Aeschylos Agamemnon" we like to quote: "[...]; one always thinks to find everything in the mental domain. Though it's not the place to elaborate on this here; it always appeared to me that it's predominantly the way in which letters link to syllables and syllables combine to words in language, and how in speech again those words link up with one another according to timing and tone which defines the intellectual, yes that it actually designates no less than the moral and political fate of a nation." [20, p. 136]

2. THE PROSODY OF THE TRAGIC POETRY IN ATTIC THEATER

As our main concern is to analyse the interpretation of a text by the performance of an expert in the field, like professor Psaroudakēs, this makes up just the beginning of a new research that can lead us to elaborate on the relationship between music and logos [5]. What was the pitch range of the voice while reciting the tragic text? Which were the timbers according to the meaning of the words, the dynamics, the rhythm?

According to scholars [17, 10] the solo singing voice was particularly associated with Greek tragedy. Early tragic actors’ roles may have consisted almost entirely of singing. The actor of fifth-century tragedy had to sing in a variety of metres in rapid succession and to negotiate the delicate transitions between them: the shift between recitative and lyrics was regarded as particularly emotive [19]. Anapaestic and lyric verses repeatedly alternated with iambic trimeters, and these were spoken.

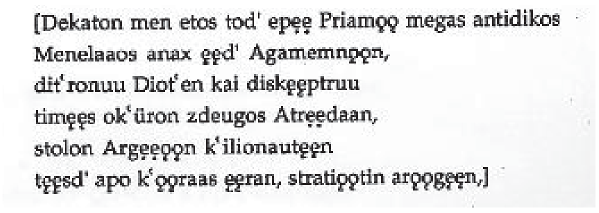

[Dekaton men etos tod' epęę Priamǫǫ megas antidikos
Menelaaos anax ęęd' Agamemnǫǫn,
ditʿronuu Diotʿen kai diskęęptruu
timęęs okʿüron zdeugos Atręędaan,
stolon Argeeǫǫn kʿilionautęęn
tęęsd' apo kʿǫǫraas ęęran, stratiǫǫtin arǫǫgęęn,]

Figure 1. the recorded text: parodos of Aeschylus' *Agamemnon* (ll.40-46).

Besides some important external evidence, tragic poetry offers internal clues to the way in which the voice was being used; in iambics people constantly use such verbs as *legein* and *phrazein* in reference to their own speech and that of their interlocutors, whereas the semantic range referring to lyric utterance, - which includes *melpein* and *aidein*, is quite different.[1] [10]

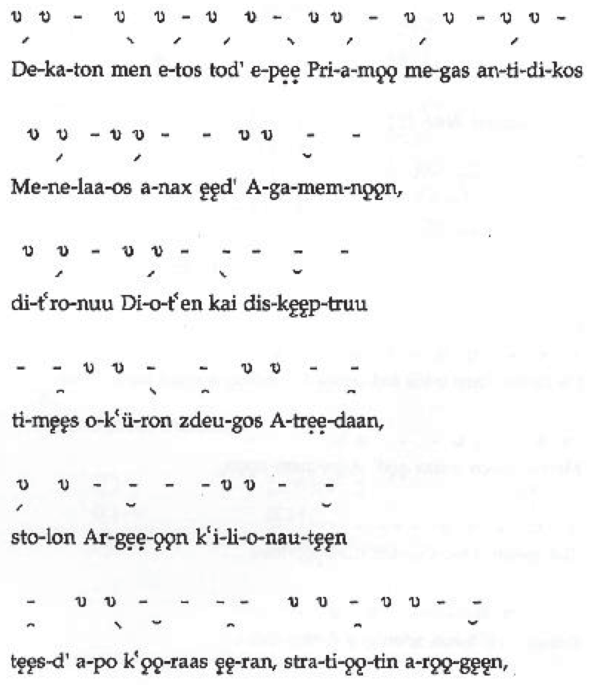

De-ka-ton men e-tos tod' e-pęę Pri-a-mǫǫ me-gas an-ti-di-kos
Me-ne-laa-os a-nax ęęd' A-ga-mem-nǫǫn,
di-tʿro-nuu Di-o-tʿen kai dis-kęęp-truu
ti-męęs o-kʿü-ron zdeu-gos A-tręę-daan,
sto-lon Ar-gęę-ǫǫn kʿi-li-o-nau-tęęn
tęęs-d' a-po kʿǫǫ-raas ęę-ran, stra-ti-ǫǫ-tin a-rǫǫ-gęęn,

Figure 2. The tonal movement of the voice upon the analysis of Psaroudakēs [1].

Tragedy thus offered the dramatist a palette of vocal techniques with which to paint his sound pictures, and certain patterns can be discerned in the way that he handled them. Our main concern is to develop this palette of vocal techniques in order to trace new audio ways of performing the tragic text and thus make the connection of Tragedy to the etymology of *tragoudi* which in modern Greek means song.[2]

2. METHODOLOGY FOR THE PROSODIC MODELLING

Intonation modelling for speech synthesis is now one of the big issues facing not only speech synthesis systems but also music synthesis systems.

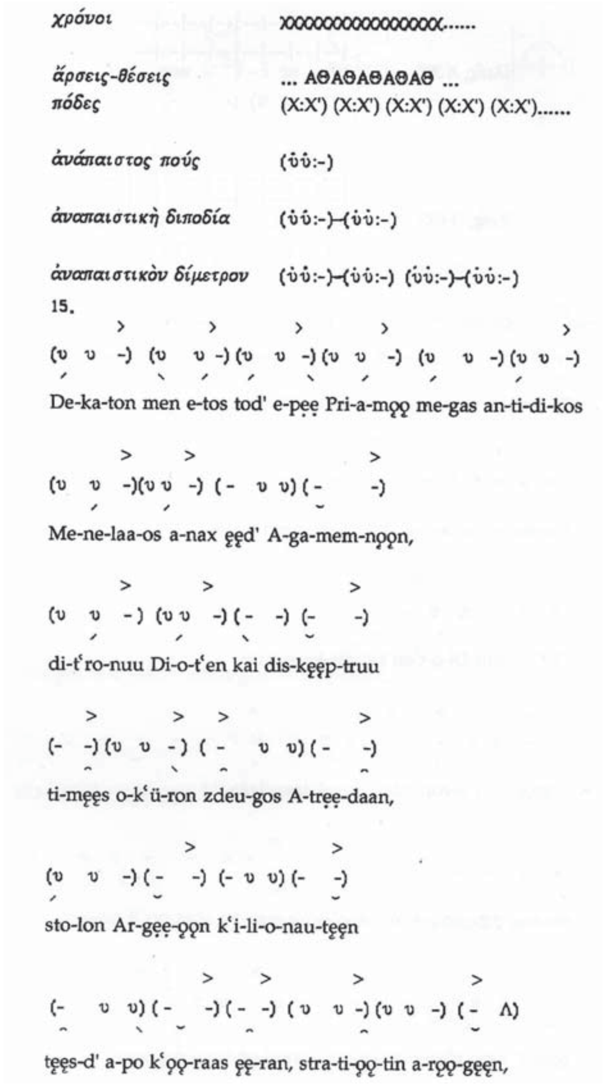

χρόνοι
ἄρσεις-θέσεις
πόδες
ἀνάπαιστος πούς
ἀναπαιστικὴ διποδία
ἀναπαιστικὸν δίμετρον
15.
De-ka-ton men e-tos tod' e-pęę Pri-a-mǫǫ me-gas an-ti-di-kos
Me-ne-laa-os a-nax ęęd' A-ga-mem-nǫǫn,
di-tʿro-nuu Di-o-tʿen kai dis-kęęp-truu
ti-męęs o-kʿü-ron zdeu-gos A-tręę-daan,
sto-lon Ar-gęę-ǫǫn kʿi-li-o-nau-tęęn
tęęs-d' a-po kʿǫǫ-raas ęę-ran, stra-ti-ǫǫ-tin a-rǫǫ-gęęn,

Figure 3. The rhythmic interpretation of the text upon the analysis of Psaroudakēs [1]

[1]assets.cambridge.org/97805216/51400/excerpt/9780521651400_excerpt.pdf

[2] Theatrical singers are attested from tragedies of Thespis in the sixth century BC to the Byzantine theatres in which Theodora performed in the sixth century AD, when the word 'tragedy' gave rise to what is still the word for 'song' in the Greek language (*tragoudi*).

One of the basic questions that derive from ancient Greek text is the intonation of speech according to the text and the emotions. Many evolutionary theories suggest that musical behaviour evolved in conjunction with, or as an adaptation of, vocal communication [7].

In music, pitch and temporal relations define musical tunes, which retain their identities across transformations in pitch level and tempo. In speech, pitch variation provides an important source of semantic and emotional information. In addition, temporal properties help listeners to determine boundaries between words and phrases. Typically, descending pitch contours and syllables or notes of long duration mark ends of phrases in speech [11] and in music [8].

The problem of intonation modelling for speech synthesis is summed up by the following quote regarding segmental effects on pitch: We have some basic intuitive ideas about what natural pitch should sound like, but we just don't understand enough to know how the pitch associated with a specific segment, in a specific syllable with a specific accent, in a specific word in a specific phrase with a specific phrase type, in a specific context, spoken by a specific speaker, should behave.

We have followed the next steps in order to extract the prosodic contours of the recited voice of Psaroudakēs according to the last research results of pronunciation [3] utilising the open source environment of *Praat* in order to investigate and compare the special interpretation and the diagrams that professor Psaroudakēs has designed by hand concerning the rhythm, the accent and the intonation. [1].

1.) Record the text by the special interpretation of S. Psaroudakēs according to the Allen's pronunciation system.
2.) Implement the prosodic feature extraction tools in order to get several features that describe the performative model of the prosody.

3. IMPLEMENTATION

For the implementation of the prosodic feature extraction tool we have chosen *Praat's* programmable scripting language [9].

An important reason to use *Praat* as our platform is that it provides an existing suite of high quality speech analysis routines, such as pitch tracking.

Several different types of features are computed based on the stylised pitch contour.

- Range features: These features reflect the pitch range of a single word or a window preceding or following a word boundary. These include the minimum, maximum, mean, and last values of a specific region (i.e., within a word or window) relative to each word boundary. These features are also normalised by the baseline values, the top line values, and the pitch range using linear difference and log difference.
- Range features: These features reflect the pitch range of a single word or a window preceding or following a word boundary. These include the minimum, maximum, mean, and last values of a specific region (i.e., within a word or window) relative to each word boundary. These features are also normalised by the baseline values, the top line values, and the pitch range using linear difference and log difference.
- Movement features: These features measure the movement of the contour for the voiced regions of the word or window preceding and the word or window following a boundary. The minimum, maximum, mean, the first, and the last stylised values are computed and compared to that of the following word or window, using log difference and log ratio.
- Slope features: Pitch slope is generated from the stylised pitch values. The last slope value of the word preceding a boundary and the first slope value of the word following a boundary are computed. We also include the slope difference and dynamic patterns (i.e., falling, rising, and unvoiced) across a boundary as slope features, since a continuous trajectory is more likely to correlate with non-boundaries; whereas, a broken trajectory tends to indicate a boundary of some type.
- Energy features: The energy features are computed based on the intensity contour produced by *Praat*.

Similar to the features, a variety of energy related range features, movement features, and slope features are computed, using various normalisation methods.

- Other features: We add the gender type to our feature set. Currently the gender information is provided in a metadata file, rather than obtaining it via automatic gender detection.

	Duration Features	F_0 Features	Energy Features
Word	✓	✓	✓
Phone	✓	×	×
Vowel	✓	×	×
Rhyme	✓	×	×
VUV	×	✓	×
Raw Pitch	×	✓	×
Stylized Pitch	×	✓	×
Pitch Slope	×	✓	×
Raw Energy	×	×	✓
Stylized Energy	×	×	✓
Energy Slope	×	×	✓

Table 1. The use of raw files for extracting various features

Given a corpus with audio and time aligned words and phones as input, our tool first extracts a set of basic elements (e.g., raw pitch, stylised pitch, VUV) representing duration, and energy information, as is shown in Figure 5 (c). Then a set of duration statistics (e.g., the means and variances of pause duration, phone duration, and last rhyme duration), related statistics (e.g., the mean and variance of logarithmic values), and energy related statistics are calculated. Given the duration, , and energy information, as well as the statistics, it is straightforward to extract the prosodic features at each word boundary, according to the definition of features in and our tool documentation [12].

We describe below how to obtain and represent these basic elements in *Praat*. Table 1 summarises their use in the computation of the prosodic features.

Word and Phone Alignments: A forced alignment system 2 is used to determine the starting and ending times of words and phones. In our tool these alignments are represented in *TextGrid IntervalTiers*, as in Figure 4

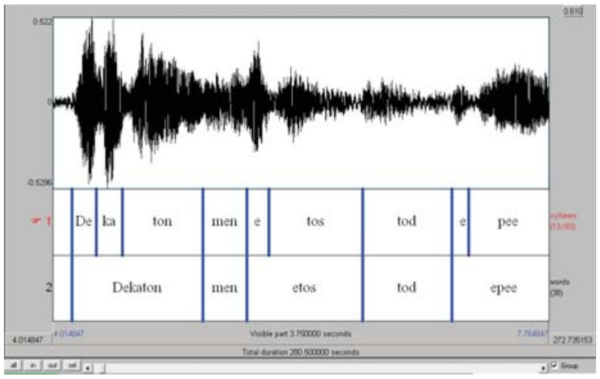

De
ka
ton
men
e
tos
tod
e
pee
Dekaton
men
etos
tod
epee
Visible part 3.750000 seconds
Total duration 280.500000 seconds

Figure 4. Sound and Textgrid Tiers of syllabus and words

Vowel and Rhyme: The starting and ending times of vowels and rhymes are determined from the phone alignments. As to rhyme, we only consider the last rhyme, which is defined to be the sequence of phones starting from the last vowel and covering all the remaining phones in a word. Vowels and rhymes are also represented in *TextGrid IntervalTiers.*

We rely on *Praat's* autocorrelation based pitch tracking algorithm to extract raw pitch values, using gender dependent pitch range.

The raw pitch contour (Figure 5) is smoothed and the voiced/unvoiced regions are determined and stored in a *TextGrid IntervalTier. Praat's* pitch stylisation function is used to stylise raw values over each voiced region (Figure 6). Both raw values and stylised values are represented in *PitchTiers*. The pitch slope values are generated based on the stylised pitch contour, and are stored in a *TextGrid IntervalTier.*

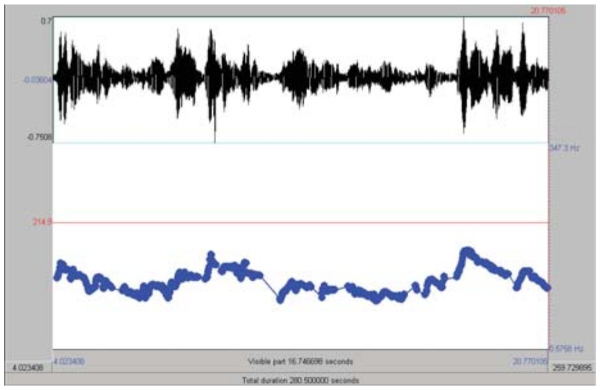

Visible part 16.746698 seconds
Total duration 280.500000 seconds

Figure 5. Sound and raw pitch countour (Pitch Tier)

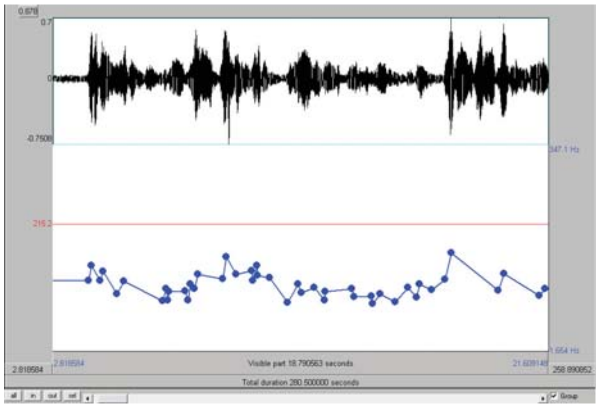

Visible part 18.790563 seconds
Total duration 280.500000 seconds

Figure 6. Sound and stylised pitch contour (Pitch Tier).

Energy: Intensity values are computed for each frame and stored in an IntensityTier (Figure 7). Since there is no intensity stylisation function in *Praat*, we choose to represent intensity values in a *PitchTier*, and apply the pitch stylisation function to stylise the intensity contour. Note that stylisation is performed on the entire intensity contour, in contrast to the pitch case, for which this applies only in voiced regions. The raw and stylised intensity values are stored in *PitchTiers*, and the slope values are stored in a TextGrid *IntervalTier.*

As we discussed above, the major advantage of building a prosodic model based on *Praat* is the capability of taking advantage of *Praat's* existing built-in speech analysis algorithms and other *Praat* scripts that have been written as extensions. In addition, because *Praat* is a public domain tool, there is the promise of future extensions to *Praat* functionality.

Although the features we have implemented have been used for a variety of event detection tasks, they are not necessarily equally effective for all tasks. Hence, it is important to have the flexibility to easily add new features to the system.

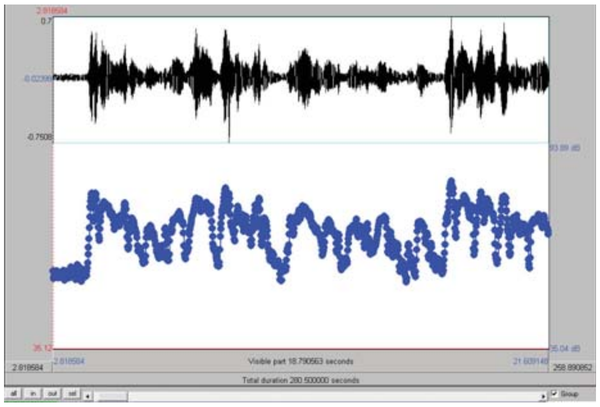


Figure 7. Sound and raw intensity contour (Intensity Tier).

4. FURTHER DEVELOPMENT

This research is only the starting point of understanding the audio aspects of Ancient Greek music.

In the future we hope to focus, on the extraction of rules that describe this kind of prosody and implement it in the model of Greek speech synthesis *Demosthenes* [18] in order to instruct the system with new information about ancient Greek poetry. This will be a further starting point to compare the phonological systems of Ancient[3] and Modern Greek concerning singing and the relation of the language.

On the other hand our research will be appreciated by directors and actors of Ancient Greek tragedy who seek for approaching the realistic way of interpreting the text [4, 8].

5. CONCLUSION

In this paper, we have attempted to extract especially the pitch contours as well as other features that characterise the interpretation of the tragic text prosody upon the special analysis of professor S. Psaroudakēs.

This research initiates a new domain: computational archeomusicology where computer-based analytical methods are used for the study of long-term musical behaviour and evolution of ancient Greek music, its systematisation and systems. The procedure of the prosodic feature extraction has been achieved through the environment extraction tools we built based on *Praat*.

[3] In modern Greek, especially in singing, we find great differences in the accentual system. Ancient Greek is generally referred to as a language with melodic accent [3] using register tone (as opposed to e.g. Mandarin Chinese which uses contour tone), varying the pitch of the voice and thereby changing the lexical or the grammatical meaning of a word [16, p. 253]. Modern Greek with its dynamic accent focuses on using stress, i.e., using more air and muscular energy, thereby producing a mixture of increased loudness, pitch and quantity [16, pp.249–250]. This difference manifested itself in the Modern Greek writing system, in that all accentual signs that had been introduced by Aristophanes of Byzantium around the third century BCE, except for the acute, the οξεία, were dropped from the official orthography in the reforms of the 1970s and 80s.

6. REFERENCES

[1] Psaroudakes S. "Meters in ancient Greek tragedy «Μέτρα στην αρχαία ελληνική τραγωδία»", *Μούσα 3:77*, Athens, 1996, pp. 77-92.

[2] Allen, W. S. *Accent and rhythm, Prosodic features in Latin and Greek: a study in theory and reconstruction.* Cambridge University Press, Cambridge, 1993.

[3] Allen, W. S. *Vox Graeca. The pronunciation of Classical Greek.* Cambridge University Press, Cambridge, 1987.

[4] Georgaki, A. "Allegories et mutations vocales au sein de la Yannis Christou (1926-1971) et Michael Adamis(1929)", in: Giordano Ferrari, H. (ed.) *la voix dans la dramaturgie contemporaine Grecque*, Gallimard, Paris, 2008.

[5] Lohmann, J. *Mousiké et Logos.* Ed.T.E.R, Paris, 1989.

[6] Loraux, N. "La voix endeuillée. Un essai sur la tragédie", MIT Press, Cambridge, MA, 1999.

[7] Dissanayake, "Antecedents of the temporal arts in early mother-infant interaction", in: Wallin N., Merker B. & Brown S. (eds.), *The Origins of Music*. Cambridge, MA: MIT Press, 1992 pp. 389-410.

[8] Narmour, E. *The analysis and cognition of basicmelodic structures*, University of Chicago Press, Chicago, 1990.

[9] Boersma P. & Weeninck, D. "Praat, a system for doing phonetics by computer", T*echnical Report 132, Inst. of Phonetic Sc.*, University of Amsterdam, 1996.

[10] Easterling P & Hall E. (Eds.) "The art of the actor", *Greek and Roman Actors: Aspects of an Ancient Professionm*, in: www.cambridge.org Press, pp. 370-380.

[11] P. Price, Ostendorf, M., Shattuck-Hufnagel, S. & Fong, C. "The Use of Prosody in Syntactic Disambiguation", *J. of the Acoust. Society of America 90*, 6, 1991, pp. 2956-2970.

[12] Huang Z., Chen L., & Harper M. "Purdue Prosodic Feature Extraction Toolkit on Praat", *Zeitschrift für Phonetik 16*, Diesterweg, 2006, pp. 293-326.

[13] Devine, A. M. & Stephens, L. D. *The prosody of Greek speech,* Oxford University Press, New York, 1994.

[14] Xydas, G. Spiliotopoulos, D. & Kouroupetroglou, G. "Prosody Prediction from Linguistically Enriched Documents Based on a Machine Learning Algorithm", *Proceedings of the 6th International Conference of Greek Linguistics (6th ICGL)*, Rethymno, Greece, September 18-21 2003.

[15] Devine, A. M. & Stephens, L. D. *The prosody of Greek speech,* Oxford University Press, New York, 1994.

[16] Ladefoged, P. *A course in phonetics. Forth Worth:* Harcourt Brace, 1993.

[17] Pöhlmann, E. & West, M. L. Documents of Ancient Greek Music. Clarendon, Oxford Press, Oxford, 2001.

[18] Xydas, G. & Kouroupetroglou, G. "The DEMOSTHENES Speech Composer", *Proceedings of the 4th ISCA Tutorial and Workshop on Speech Synthesis*, SSW4, Perthshire, Scotland, September 2001, pp. 167-172

[19] Aristotle, Problems, Book 19, 6.

[20] Aeschylos Agamemnon, metrical translation by Wilhelm von Humboldt, bei Gerhard Fleischer dem Jüngeren, Leipzig 1816, introduction, p. 136

FIRST-ORDER LOGIC CLASSIFICATION MODELS OF MUSICAL GENRES BASED ON HARMONY

Amélie Anglade
Centre for Digital Music
Queen Mary
University of London
amelie.anglade@elec.qmul.ac.uk

Rafael Ramirez
Music Technology Group
Universitat Pompeu Fabra
rramirez@iua.upf.edu

Simon Dixon
Centre for Digital Music
Queen Mary
University of London
simon.dixon@elec.qmul.ac.uk

ABSTRACT

We present an approach for the automatic extraction of transparent classification models of musical genres based on harmony. To allow for human-readable classification models we adopt a first-order logic representation of harmony and musical genres: pieces of music are represented as lists of chords and musical genres are seen as context-free definite clause grammars using subsequences of these chord lists. To induce the context-free definite clause grammars characterising the genres we use a first-order logic decision tree induction algorithm, Tilde. We test this technique on 856 Band in a Box files representing academic, jazz and popular music. We perform 2-class and 3-class classification tasks on this dataset and obtain good classification results: around 66% accuracy for the 3-class problem and between 72% and 86% accuracy for the 2-class problems. A preliminary analysis of the most common rules extracted from the decision tree models built during these experiments reveals a list of interesting and/or well-known jazz, academic and popular music harmony patterns.

1 INTRODUCTION

Users tend to be sceptical about automatic recommender systems that are not transparent. Providing some insight into the reasoning to the user has proven to improve both the user's trust and his involvement in the system [3, 11]. Thus, for a better user acceptance, automatic music classification systems (which can be used as part of a music recommender system) should provide an explanation to the user on how a piece of music is classified by the system.

Recent studies [1, 9] have shown that a logic-based representation of the musical events together with a logical inference such as Inductive Logic Programming (ILP) [5] allow for a human-readable characterisation of music. In this article we extend these works by building human readable and transparent music classification models – namely first-order logic decision tree models (an extension of the classical decision trees using ILP) – performing both classification and characterisation. We focus on classification into musical genres using as descriptor the harmony of each song.

SMC 2009, July 23-25, Porto, Portugal

The paper is organised as follows: In Section 2 we review some existing studies using harmony for automatic classification. In Section 3 we introduce the harmonic content description employed in this study. In Section 4 we present the details of our learning task, including the data, the inductive logic decision tree algorithm and the results (classification performances and characterisation rules obtained) before concluding in Section 5.

2 PREVIOUS RELATED WORK

Although some harmonic (or chord) sequences are famous for being used by a composer or in a given genre, little attention has been paid in the automatic genre recognition literature to how harmony can help in this task. For example, in [12] the authors use a chroma feature representation describing the harmonic content of the music. A comparison of the histograms reveals some patterns which contain some genre specific information. Recognition rates around 70% are reported for a five class classification. However this study focuses on low-level harmony features.

In [10], a rule-based system is used to classify sequences of chords belonging to three categories: Enya, Beatles and Chinese folk songs. A vocabulary of 60 different chords was used, including triads and seventh chords. Classification accuracy ranged from 70% to 84% using two-way classification, and the best results were obtained when trying to distinguish Chinese folk music from the other two styles, which is a reasonable result as both western styles should be closer in terms of harmony.

Paiement et al. [7] also used chord progressions to build probabilistic models. In that work, a set of 52 jazz standards was used, encoded as sequences of 4-note chords. The authors compared the generalization capabilities of a probabilistic tree model against a Hidden Markov Model

(HMM), both capturing stochastic properties of harmony in jazz, and the results suggested that chord structures are a suitable source of information to represent musical genres.

More recently, Lee [4] has proposed genre-specific Hidden Markov Models that learn chord progression characteristics for each genre. Although the ultimate goal of this work is using the genre models to improve the chord recognition rate, he also presented some results on the genre classification task. For that task a reduced set of chords (major, minor, and diminished) was used.

Finally, Perez-Sancho et al. [8] have investigated if 2, 3 and 4-grams of chords can be used for automatic genre classification on both symbolic and audio data. They report better classification results when using a richer vocabulary (seventh chords) and longer n-grams.

3 HARMONIC CONTENT DESCRIPTION AND REPRESENTATION

We extend these previous studies in which chord sequences are of fixed length by using context-free definite-clause grammars to represent chord sequences of arbitrary length.

3.1 Harmonic content

The music pieces used in this study have been kindly provided by the Pattern Recognition and Artificial Intelligence Group of the University of Alicante. There, experts have collected, annotated and double-checked files encoded in the format of the PG Music software Band in a Box (aka BIAB) [1] and then converted into MMA [2] format. These files can be seen as simplified scores only containing the chords which are labelled in a jazz/pop/rock shorthand fashion (e.g. using G7 for G dominant seventh chord, D for D major, etc.). In these transcriptions from the University of Alicante, chords are limited to major or minor triads and 7th chords (dominant seventh, major seventh or minor seventh). But there is no unique way to transcribe chords and notice that different levels of detail in chord representation might lead to the induction of different classification rules. Furthermore only the chord changes are annotated in the provided files. Although meter positions of chords are important since we do not have access to this information we leave this for future work.

3.2 Using context-free definite-clause grammars as representation scheme

Context-free definite-clause grammars proved to be useful in the logic-based extraction of biological patterns in a particular class of amino acids sequences, the neuropeptide precursor proteins (NPPs) [6]. NPPs share common characteristics with musical pieces (represented as chord sequences): these sequences are highly variable in length, they tend to show almost no overall sequence similarity and the class (NPPs or non-NPPs in the case of amino acids sequences, musical genres in the case of songs) to which a given sequence belongs is not always clear (some NPPs have not yet been discovered and experts can disagree on the genre of a given song). Both because of these similarities in the data and because context-free definite-clause grammars can be induced using Inductive Logic Programming, we choose to adopt this representation scheme.

In the definite clause grammar (DCG) formalism a sequence over a finite alphabet of letters is represented as a list of letters. Here the chords (e.g. G7, Db, BM7, F#m7, etc.) are the letters of our alphabet. A DCG is described using predicates. For each predicate `p/2` (or `p/3`) of the form `p(X,Y)` (or `p(c,X,Y)`), `X` (the input) is a list representing the sequence to analyse and `Y` (the output) is the remaining part of the list `X` when its prefix matching the predicate `p` (or property `c` of the predicate `p`) is removed.

```
%We assume the tonality is C Major
perfect_cadence(A,B):-
gap(A,C), degree(5,C,D), degree(1,D,E), gap(E,B).

%definition of the gap predicate
gap(A,A).
gap([_|A],B) :- gap(A,B).

%definition of the rootNote predicate
rootNote('C',[c|T],T).
rootNote('C',[cm|T],T).
...

%definition of the degree predicate
degree(5,A,B) :- rootNote('G',A,B).
degree(1,A,B) :- rootNote('C',A,B).
```

Table 1. Simple definite clause grammar describing a perfect cadence in C major.

To illustrate this, an example of a simple chord sequence context-free definite-clause grammar encoding the concept of perfect cadence (in C major) is given in Table 1. In this example, the target concept is `perfect_cadence/2`. To describe it three background predicates, `rootNote/3`, `degree/3` and `gap/2`, are used. `rootNote(n,A,B)` means that the first chord of list `A` has for root note `n`. `B` is the remaining list when the first chord of `A` is removed. `degree(d,A,B)` means that the first chord of list `A` has for degree `d`. The last lines of the Table 1 state that root note G corresponds to the 5th degree (dominant) in C major and C corresponds to the 1st degree (tonic). In Prolog the underscore (_) can match anything, so the `gap/2` predicate (also

[1] http://www.pgmusic.com/products_bb.htm
[2] http://www.mellowood.ca/mma/

from [6]) matches any chord sequence of any length, allowing to skip uninteresting subsequences (not characterised by the grammar rules) and to handle large sequences for which otherwise we would need very large grammars. Finally, the first lines of Table 1 define a perfect cadence as a chord on the fifth degree directly followed by a chord on the first degree (using the `degree/3` predicate), sequence that can happen anywhere in the list of chords that define the song (due to the `gap/2` predicate).

```
rootNote('C',[c|T],T).      rootNote('C',[cm|T],T).     ...
rootNote('Cs',[cs|T],T).    rootNote('Cs',[csm|T],T).   ...
...                         ...                         ...

interval(perf_uni,A,B) :- rootNote('C',A,B), rootNote('C',B,C).
interval(perf_uni,A,B) :- rootNote('Cs',A,B), rootNote('Cs',B,C).
...
interval(min_sec,A,B) :- rootNote('C',A,B), rootNote('Db',B,C).
...
interval(dim_oct,A,B) :- rootNote('C',A,B), rootNote('Cb',B,C).
...

gap(A,A).
gap([_|A],B) :- gap(A,B).
```

Table 2. Background knowledge predicates used in the first-order logic decision tree induction algorithm to describe genres. For each chord in a chord sequence its root note is identified using the `rootNote/3` predicate. The intervals between the root notes (measured upwards) are "computed" using the `interval/3` predicate.

For the genre classification tasks our target predicate is `genre/3` and the patterns we extract are based on the intervals between root notes of the chords. Root interval progressions capture some degree information but do not depend on the tonality. Thus when using root intervals no preprocessing of the data or key extraction is necessary. The background predicates used to describe our grammar (given as background knowledge to our learning system) are given in Table 2. Notice that contrary to the example in Table 1 in which one rule was enough to describe a perfect cadence, we look for a set of rules to describe each genre, each rule describing one characteristic chord sequence of this genre.

4 LEARNING CLASSIFICATION RULES FOR MUSICAL GENRES

4.1 Training data

The data set contains three genres: popular, jazz, and academic music. Popular music data consists of pop, blues, and celtic (mainly Irish jigs and reels) music; jazz consists of a pre-bop class grouping swing, early, and Broadway tunes, bop standards, and bossanovas; and academic music consists of baroque, classical and romantic music. All the categories have been defined by music experts at the University of Alicante who have also collaborated in the task of assigning meta-data tags to the files and rejecting outliers. The total amount of pieces is 856 (Academic 235; Jazz 338; Popular 283), providing around 60 hours of music data.

4.2 Learning algorithm

We have applied Tilde's top-down decision tree induction algorithm [2]. Tilde can be considered as a first order logic extension of the C4.5 decision tree algorithm: instead of testing attribute values at the nodes of the tree, Tilde tests logical predicates. This provides the advantages of both propositional decision trees (i.e. efficiency and pruning techniques) and the use of first-order logic (i.e. increased expressiveness). First-order logic enables us to use a background knowledge (which is not possible with non relational data mining algorithms). It also provides a more elegant way to represent musical concepts/events/rules which can be transmitted as they are to the users. Thus the classification process can be made transparent to the user.

Tilde builds models, namely first-order logic decision trees which can also be represented as ordered sets of rules (or Prolog programs). In the case of classification, the target predicate of each model represents the classification problem.

4.3 Learning task

We use Tilde with `genre/3` as target predicate, where `genre(g,A,B)` means the song `A` (represented as its full list of chords) belongs to genre `g`. The last argument `B`, the output list (i.e. the empty list) is necessary to comply with the context free definite clause grammar representation. The predicates considered to build the model are `interval/3` and `gap/2`, defined in the background knowledge (cf. Table 2). In addition we constrain the system to use at least two consecutive `interval` predicates between two `gap` predicates. This guarantees that we are considering local root interval sequences of a least length 2 (i.e. chord sequences of length 3) in the songs. However notice that the context free grammar definite clause representation allows these local root interval sequences to be of any length larger than 2.

4.4 Classification results

Our objective was to classify musical pieces into the three main genres present in the dataset: academic, jazz and popular music. For that we both built a model that was directly dealing with the 3-class problem and induced three models dealing with each of the 2-class subproblems. For each classification task we performed a 5-fold cross-validation.

Furthermore we controlled the complexity of the decision trees built by varying the minimal number of examples a leaf should cover (MC). The results of these experiments are shown in Table 3.

academic/jazz/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.398)	0.663	**0.665**	0.655
Stderr	0.016	**0.016**	0.016
# nodes in the tree	91.0	**42.6**	26.6
# literals in the tree	248.6	**116.2**	71.0
academic/jazz	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.590)	0.839	**0.860**	0.844
Stderr	0.015	**0.015**	0.015
# nodes in the tree	22.2	**10.6**	5.2
# literals in the tree	62.6	**29.4**	14.8
academic/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.540)	0.743	**0.768**	0.723
Stderr	0.019	**0.019**	0.020
# nodes in the tree	42.8	**24.0**	12.0
# literals in the tree	113.6	**61.6**	31.6
jazz/popular	MC=2	**MC=5**	MC=10
Accuracy (baseline = 0.551)	0.843	**0.819**	0.804
Stderr	0.015	**0.016**	0.016
# nodes in the tree	44.0	**21.8**	12.2
# literals in the tree	125.2	**61.0**	31.8

Table 3. Classification results (on the test data) using a 5-fold cross-validation. MC (minimal number of examples a leaf should cover) is a parameter of the decision tree learning algorithm. The number of nodes and literals present in a tree gives an estimation of its complexity.

The models for all the classification tasks have a good accuracy which is not much affected by the value of the minimal coverage of a leaf (MC): the accuracy is always much higher than the baseline. Changing the minimal coverage of a leaf from 2 examples to 5 examples (and similarly when going from MC=5 to MC=10) leads to trees containing half the number of nodes and literals (so models that are two times simpler). As long as the classification accuracy is not affected using simpler models has several advantages. Firstly, the processing time to assign a class to an unseen example is smaller when using simpler models. Moreover simpler models contain less rules and each rule covers on average a higher number of examples. Such rules tend to be more meaningful and do not capture local or isolated phenomena, so are less subject to overfitting. Finally if we want to display the model to the user (for transparency reasons) simpler models are easier to understand. Here a good compromise is reached when using MC=5 for which the classification accuracy is generally higher than for any other MC value and the models are simpler but not overly simplified.

The confusion matrices for the four classification tasks when using MC=5 are shown in Table 4. We obtain respectively 86% (academic vs. jazz), 77% (academic vs. popular), 82% (jazz vs. popular) and 67% (3-class problem) accuracy. The best results are obtained when trying to distinguish jazz from another genre (academic or popular). The biggest difficulty that appears in both the 3-class task and the 2-class task is to distinguish academic music from popular music. Indeed the harmony of these two genres can be very similar, whereas jazz music is known for its characteristic chord sequences, very different from other genres harmonic progressions.

Real/Predicted	academic	jazz	popular	Total
academic	145	31	58	234
jazz	33	267	37	337
popular	68	56	151	275
Total	246	354	246	846
academic	197	37		234
jazz	43	294		337
Total	240	331		571
academic	165		69	234
popular	49		226	275
Total	214		295	509
jazz		272	65	337
popular		46	229	275
Total		318	294	612

Table 4. Confusion matrices (on the test data) for the four classification tasks using a 5-fold cross-validation and for minimal coverage of a leaf set to 5 (MC=5).

4.5 Overview of the extracted rules

As explained in Section 4.2 for each run Tilde returns a classification model that can be represented as a tree or as an ordered set of rules (or a Prolog program). Because of space limitation we only show some interesting and recurrent rules extracted from the various models built (a complete list of classification models and their rules is available upon request). However note that a rule in itself can not perform classification both because of having a lower accuracy than the full model and because the ordering of rules in the model is important to the classification (i.e. some rule might never be used on some example because one of the preceding rules in the model covers this example).

The following rule was found in the academic vs. jazz classification models:
genre(academic,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), gap(E,B).
"Some academic music pieces contain a chord root interval sequence of two consecutive ascending perfect fifth."

This rule can be interpreted as the common IV-I-V degree progression. Interestingly the same rule appears in the popular vs. jazz classification model to characterise popular music. So rather than characterising academic music or popular music this rule suggests that a sequence of two consecutive ascending fifth does not occur very frequently in jazz music (at least not as frequently as in academic or popular music).

The above rule is further specialised in the 3-class models to characterise popular music this time:
genre(popular,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), interval(min_sev,E,F), gap(F,B).
"Some popular music pieces contain a chord root interval sequence of two consecutive ascending fifth directly followed by an ascending minor seventh."
This sequence could be for instance the IV-I-V-IV sequence found in the verse of *"Let it be"* by the Beatles.

Other rules found both in the academic vs. jazz and the 3-class models are:
genre(academic,A,B) :- gap(A,C), interval(min_sev,C,D), interval(perf_fifth,D,E),interval(perf_fourth,E,F),gap(F,B).
"Some academic music pieces contain a chord root interval sequence in which an ascending minor seventh is followed by an ascending perfect fifth, followed by an ascending perfect fourth."
and:
genre(academic,A,B) :- gap(A,C), interval(perf_fifth,C,D), interval(perf_fifth,D,E), gap(E,F), interval(perf_fifth,F,G), interval(perf_fourth,G,H), gap(H,B).
"Some academic music pieces contain a chord root interval sequence of two ascending perfect fifth later (but not necessarily directly) followed by an ascending perfect fifth and an ascending perfect fourth."
They can be respectively interpreted as V-IV-I-IV and IV-I-V later followed by a back and forth pattern such as I-V-I or IV-I-IV.

Some very jazzy patterns were also found, such as:
genre(jazz,A,B) :- gap(A,C), interval(perf_fourth,C,D), interval(aug_fourth,D,E), gap(E,B).
"Some jazz music pieces contain a chord root interval sequence containing an ascending perfect fourth followed by an ascending augmented fourth."
and:
genre(jazz,A,B) :- gap(A,C), interval(maj_sev,C,D), interval(perf_fourth,D,E), gap(E,B).
"Some jazz music pieces contain a root interval sequence containing an ascending major seventh directly followed by an ascending perfect fourth."

5 CONCLUSIONS AND FUTURE WORK

In this paper we presented a first-order logic approach to automatically extract genre classification models using harmony. This models are not black boxes: thanks to the expressiveness of first order logic the decision tree models we obtained can be presented to the users as sets of human readable rules. Good classification results (comparable to previous work results in the field) were obtained with these first-order decision trees algorithms. With almost no accuracy loss we managed to lower the complexity of our models from 50 rules to 25 rules on average, getting simpler, faster to use and more meaningful decision trees. By using a context-free definite-clause grammar representation which can encode chord sequences of any length we extended previous classification and characterisation studies that were limited to chords sequences of fixed length. For instance in [1] musical style was characterised using chord sequences of length 4 . In [8], the n-gram representation is used to study chord sequences of length 2, 3 or 4 only. Our system not only allows for any chord sequence length but also enables the coexistence of harmony progressions of various lengths in the same model.

Future work includes adding the chord categories (e.g. minor triad, dominant seventh, etc.) in our grammar to try to increase the classification accuracy of our models. We also plan to test if using degrees (when key estimation is possible) instead of root intervals can improve our models. Finally we intend to test our framework on audio data using chord transcription algorithms.

6 ACKNOWLEDGMENTS

This work is supported by the EPSRC project OMRAS2 (EP/ E017614/1) and partially supported by the Spanish TIN project ProSeMus (TIN2006-14932-C02-01). The first author's internship at the Music Technology Group was supported by the EPSRC Platform grant EP/E045235/1. We would like to thank the Pattern Recognition and Artificial Intelligence Group of the University of Alicante for providing the data.

7 REFERENCES

[1] A. Anglade and S. Dixon. Characterisation of harmony with inductive logic programming. In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 63–68, Philadelphia, Pennsylvania, USA, 2008.

[2] H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 53–63, Madison, Wisconsin, USA, 1998. Morgan Kaufmann.

[3] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings*

of the 2000 ACM conference on Computer supported cooperative work, pages 241–250, Philadelphia, Pennsylvania, USA, December 2000.

[4] K. Lee. A system for automatic chord transcription using genre-specific hidden markov models. In *Proceedings of the International Workshop on Adaptive Multimedia Retrieval*, pages 134–146, Paris, France, 2007.

[5] S. H. Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, 1991.

[6] S. H. Muggleton, C. H. Bryant, A. Srinivasan, A. Whittaker, S. Topp, and C. Rawlings. Are grammatical representations useful for learning from biological sequence data? - a case study. *Journal of Computational Biology*, 8(5):493–522, 2001.

[7] J.-F. Paiement, D. Eck, and S. Bengio. A probabilistic model for chord progressions. In *Proceedings of the 6th International Conference on Music Information Retrieval*, pages 312–319, London, UK, September 2005.

[8] C. Perez-Sancho, D. Rizo, S. Kersten, and R. Ramirez. Genre classification of music by tonal harmony. In *International Workshop on Machine Learning and Music, International Conference on Machine Learning*, Helsinki, Finland, July 2008.

[9] R. Ramirez, A. Hazan, E. Gómez, E. Maestre, and X. Serra. Discovering expressive transformation rules from saxophone performances. *Journal of New Music Research*, 34(4):319–330, 2005.

[10] M.-K. Shan, F.-F. Kuo, and M.-F. Chen. Music style mining and classification by melody. In *Proceedings of 2002 IEEE International Conference on Multimedia and Expo*, volume 1, pages 97–100, 2002.

[11] R. Sinha and K. Swearingen. The role of transparency in recommender systems. In *2002 Conference on Human Factors in Computing Systems*, pages 830–831, Minneapolis, Minnesota, USA, April 2002.

[12] G. Tzanetakis, A. Ermolinskiy, and P. Cook. Pitch histograms in audio and symbolic music information retrieval. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 31–38, Paris, France, 2002.

AUDITORY REPRESENTATIONS AS LANDMARKS IN THE SOUND DESIGN SPACE

Carlo Drioli
Pietro Polotti
University of Verona
{carlo.drioli,pietro.polotti}
@univr.it

Davide Rocchesso,
Stefano Delle Monache
IUAV University of Venice
roc@iuav.it
stefano.dellemonache@gmail.com

Kamil Adiloğlu,
Robert, Anniés,
Klaus Obermayer
Berlin Institute of Technology
{kamil, robokopp, oby}
@cs.tu-berlin.de

ABSTRACT

A graphical tool for the timbre space exploration and interactive design of complex sounds by physical modeling synthesis is presented. It is built around an auditory representation of sounds based on spike functions and provides the designer with both a graphical and an auditory insight. The auditory representation of a number of reference sounds, located as landmarks in a 2D sound design space, provides the designer with an effective aid to direct his search along the paths that lie in the proximity of the most inspiring landmarks.

1 INTRODUCTION

Sound synthesis techniques are nowadays available which offer a high degree of naturalness and expressiveness. Sometimes, however, this comes at a price of a consistent complexity in both the control of real time performance, and the parametric tuning required in the sound design process. One such case is physical modeling, in which the process that produces the sound is represented by means of more or less simplified equations of the underlying physical laws. As a result, the parameters involved in the modeling should in principle be easy to understand because they have a real counterpart, and should be easy to control because our sensorial experience should mediate the action-reaction patterns to which they relate[1]. Nonetheless, the most accurate and expressive models available today are often described in terms of detailed physical relations and parameters that are not always accessible or understandable to the non specialists. Moreover, the nonlinear nature of the phenomena under observation may sometimes lead to numerical schemes that do not always reflect the behavior of the real systems in the whole parameters space. These considerations motivate the search for new tools to aid the synthesis parametric tuning, and are of particular interest in our opinion in the case of physical modeling audio synthesis.

SMC 2009, July 23-25, Porto, Portugal

In this paper we propose a graphical tool for the timbre space exploration and interactive design of complex sounds by physical modeling synthesis that is intended to assist the sound designer. It is aimed at exploiting the auditory representation of sounds based on spike functions and provides the designer with both a graphical and an auditory insight that may be used in place of, or combined with, the set of low-level physical parameters of the models. The graphical tool adopts a sonic landscape paradigm, in which the auditory representation of a number of reference sounds can be located as landmarks in a 2D sound design space, and provides an effective aid to direct the search along the paths that lie in the proximity of the most inspiring sonic landmarks.

The use of terminology and metaphors referring to the environment and landscapes has a rather old tradition in the field of sounds perception, especially when referred to the perception of ecological and everyday sounds. In the late 70's, the Canadian composer R. Murray Schafer introduced the term "soundscape", defined as the auditory equivalent to landscape[2], and Barry Truax published his Handbook for Acoustic Ecology[3]. The term soundscape perception is also used in a scientific context to characterize how inhabitants perceive, experience and appraise their sonic environment.

Our use of the terms "sonic landscape" and "sonic landmark" however refers to a particular organization of sounds in a 2D sonic space. The sound synthesis framework we rely on is based on a class of physical models for everyday sounds, which includes low level events (impacts and friction) and high level ones (bouncing, breaking, rolling, crumpling). This Physically-based Sound Design Tools (SDT from now on) is being developed and supported within a number of EU funded research projects on audio synthesis and sound design (Sounding Objetcs (SOB), Closing the Loop of Sound Evaluation and Design (CLOSED), Natural interactive walking (NIW))[4, 5]. An example of graphical interface for the parametric tuning of friction sounds, included in the SDT and implemented in Max/Msp, is shown in Figure 1. The aim of the tool proposed here is to add to this class of tuning interfaces a perceptual representation output layer (spike functions) and a tool to perform parametric interpolations using reference auditory representations in the perceptual space. The representation of sounds based on the spike functions provides the designer with both a graphical and an auditory insight. The latter point is of extreme interest since it is based on a sort of ”auditory representation of sound”. The peculiarity of the spike representation is, in fact, to be lossy. This aspect turns to be an advantage, whereas the reconstructed sounds maintain the temporal articulation of the original ones but they are decolored from their original timbre characteristics. Any reconstruction of a sound has a “watery character”, so that all the sounds become timbrically similar, even if distinguishable according to their temporal structure. In this way, the sound designer has at disposal a kind of “auditory-radiography” of the sounds that can be compared as sonic archetypes, helping the designer in her/his sound space exploration.

The paper is organized as follows: first, a description of the sound synthesis engine and of the spike-based auditory representation is given in Section 2; Section 3 describes the interpolation sound space and the graphical tool proposed to assist the sound designer; in Section 4, same sound design examples obtained with the tool are illustrated; Section 5 contains the conclusions and future issues.

2 DESIGN AND IMPLEMENTATION OF THE SPIKE-BASED PARAMETER INTERPOLATION TOOL

The interactive interpolation tool presented is organized as a client-server distributed application, in which the client side hosts the user graphical front-end based on the auditory spike sound representation, and the server side hosts the SDT audio synthesis engine. In the following we provide some details on the sound synthesis algorithm used to generate the reference and the new sounds, on the spike analysis framework used to graphically represent the sounds, and the properties of the interpolation in the 2D space.

2.1 The sound synthesis engine

The sound synthesis chosen to illustrate our design tool is a physical modeling implementation of friction sounds synthesis, included in the aforementioned SDT package.

The scope of the SDT is to provide a platform of sound synthesis tools that interaction designers can easily exploit in their sketching activities and that can be run on common real time software such as Max/MSP and Pd. The aim is also to provide the patches with a set of side tools to easily manage projects and that can be of help when working with

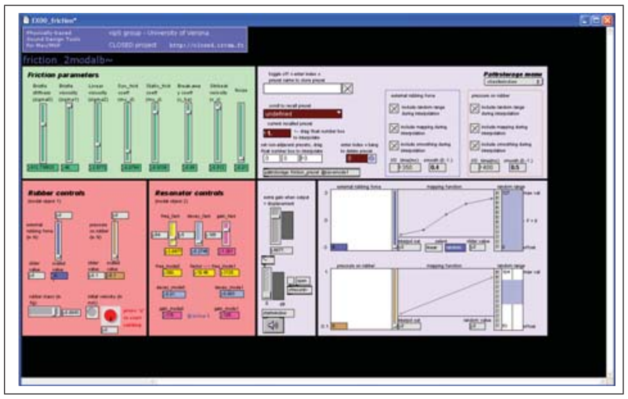


Figure 1. The friction MAX/Pd patch, contained in the SDT synthesis tools package. The user can tune up to 27 synthesis parameters representing the driving forces, the characteristics of the modal resonators, and the characteristics of the nonlinear interaction.

acquisition boards and sensors. Most of the models contained in the SDT, aimed at reproducing sounds from solid objects interaction, are structured as two resonating objects interacting by means of a contact model. The friction model specifically referred to here, relies on a description of the average behavior of a multitude of micro-contacts made by hypothetical bristles extending from each of two sliding surfaces. A modal decomposition is adopted for both interacting objects, leading to a first parametric subset including modes frequency, decay factors and gain. The remaining low-level parameters are related to the interaction mechanisms and to the interaction force specification. To gain an insight of the phenomenological role of the low-level physical parameters of the friction model, and of what a sound designer can be asked to deal with, a description is given in Table 1, and are visible in the SDT tuning interface of Figure 1 . Further details on the friction model can be found in [4], Chap. 8.

Sym.	Physical Description	Phenomenological Description
σ_0	bristle stiffness	affects the evolution of mode lock-in
σ_1	bristle dissipation	affects the sound bandwidth
σ_2	viscous friction	affects the speed of timbre evolution and pitch
σ_3	noise coefficient	affects the perceived surface roughness
μ_d	dynamic friction coeff.	high values reduce the sound bandwidth
μ_s	static friction coeff.	affects the smoothness of sound attack
v_s	Stribeck velocity	affects the smoothness of sound attack
f_N	normal force	high values give rougher and louder sounds

Table 1. A phenomenological guide to the friction model parameters.

2.2 Spike representation

In a classical signal representation approach, overlapping discrete blocks are used. However, in this method, in particular for non-stationary signals, a small shift of a block can cause a large change in the representation, depending on where an acoustic event falls within the block. A sparse, shiftable representation method based on atom-like filter functions can solve the problem. Hence, following [6], a sound signal $x(t)$ can be approximated as a linear combination of K filter functions, the so-called spikes. In this study, we employed the Gammatone functions $\gamma_{f_k,t_k}(t)$ with amplitudes a_k and residual $\epsilon_{K+1}(t)$:

$$x(t) \quad = \quad \sum_{k=1}^{K} a_k \gamma_{f_k,t_k}(t) + \epsilon_{K+1}(t). \tag{1}$$

A Gammatone function is defined by its center frequency f_k and its filter order. In a Gammatone filter bank, the filter center frequencies are distributed across the frequency axis in proportion to their bandwidth. The shape of the magnitude characteristics of the fourth order Gammatone filter approximates the magnitude characteristics of the human auditory filter in a proper way [7]. Hence, these filter functions have a biological background.

Each spike $s_k = (t_k, f_k, a_k)$ is composed of the temporal offset t_k, the center frequency f_k of the corresponding Gammatone filter and the amplitude a_k.

For a pre-determined number of spikes K, the optimal coefficients a_k, f_k, t_k, in terms of a minimal residual signal $\epsilon(t)$ have to be found. We employ matching pursuit [8, 6] to iteratively determine the spikes s_k and to minimize the residual. The algorithm is initialized by setting the residual

$$\epsilon_1(t) := x(t). \tag{2}$$

Then for $1 \leq k \leq K$, the optimal time offset t_k and center frequency f_k are determined iteratively so that the Gammatone filter $\gamma_{f_k,t_k}(t)$ maximally correlates with the signal $\epsilon_k(t)$:

$$(f_k, t_k) = \mathrm{argmax}_{f,t^*} < \epsilon_k(t), \gamma_{m_{f,t^*}}(t) > . \tag{3}$$

In order to perform the scalar product shown in Equation 3 between the filters and the signal, the filters should have certain window lengths in time domain. In his auditory toolbox, Slaney [9] used fixed window lengths for each filter in the filterbank. However, the energy levels decrease much slower for the low frequency filters than for the high frequency filters. Therefore, we adapted the window lengths considering the center frequencies. We calculated the positions within each filter, where the total energy in the time envelope falls to its thousandth. We used these position values in time as the window lengths to calculate the scalar products. Hence, the amplitude of the k^{th} spike is defined to be the scalar product between the corresponding filter, with its window length, and the residual signal, as follows:

$$a_k :=< \epsilon_k(t), \gamma_{f_k,t_k}(t) > . \tag{4}$$

Then we update the residual by

$$\epsilon_{k+1}(t) = \epsilon_k(t) - a_k \gamma_{f_k,t_k}(t) \tag{5}$$

Finally, for $k = K$ we yield Equation 1.

By varying K within Equation 1, the SNR values of the spike code corresponding to the sparsity of the representation can be changed. Increasing K increases the SNR of the spike code. Small K (high sparsity) decreases the SNR.

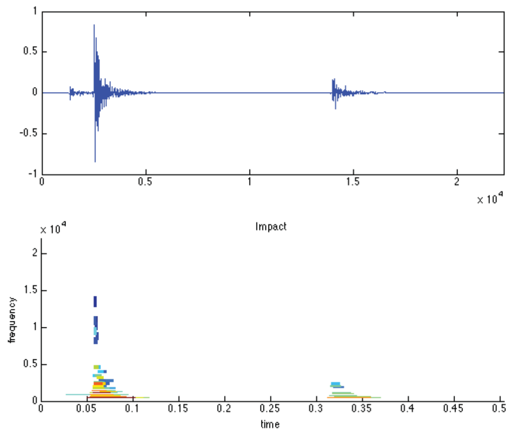


Figure 2. For an impact sound, we display the sound wave and the spike code. This picture clearly indicates how the spike code captures the skeleton of the sound.

Figure 2 shows the wave form and the spike code representation of an *impact* sound. In this example, a Gammatone filter bank of $M = 256$ filters is used for generating the spike code consisting of $K = 32$ spikes. Note that salient areas in the wave of this sound is coded with more spikes than other areas. This indicates that the spike representation captures the signal characteristics properly.

3 THE SONIC SPACE, SONIC LANDMARKS AND GUI

The sonic landscape is organized as a 2D space in which reference sounds (sonic landmarks) are located. The organization of the reference sounds may rely on perceptual criteria, derived on a statistical basis (e.g., clustering), or may be arbitrarily decided by the user.

The sonic landmarks in the sonic space form a 2D scatter points set. The sound designer may chose to generate a new

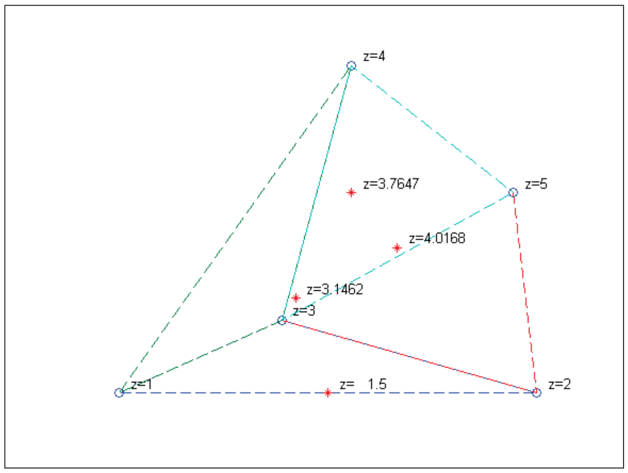


Figure 3. Scatter points interpolation: Delaunay triangulation (dashed lines) on a set of data points (circles), and four interpolated points (stars).

sound in the vicinity of a set of sonic landmarks inspired by their sonic properties. Depending on the coordinates of the new position indicated by the user, the new set of synthesis parameters is generated through an interpolation scheme accounting for the neighboring sonic landmarks. Consider a set of landmark points

$$\{(x_1, y_1, \mathbf{z}_1), (x_2, y_2, \mathbf{z}_2), \ldots, (x_n, y_n, \mathbf{z}_n)\}$$

where (x_i, y_i) is the position in the 2D space, and $\mathbf{z}_i$ is the value of the synthesis parameters vector. A convenient interpolation scheme for this class of problems is based on the following steps (for sake of simplicity, we consider a scalar parameter, z):

- Compute a grid of triangles connecting the scatter points together. One possibility, which is used here, is the Delaunay triangulation.
- For a new interpolated point, find the three vertices (x_{i1}, y_{i1}, z_{i1}), (x_{i2}, y_{i2}, z_{i2}) and (x_{i3}, y_{i3}, z_{i3}) defining the triangle in which the point is confined and compute the coefficients A, B, C, D for the equation of the plane defined by the triangle $Ax + By + Cz + D = 0$.
- Finally, compute the interpolated value through linear interpolation: $f(x, y) = -\frac{A}{C}x - \frac{B}{C}y - \frac{D}{C}$

Figure 3 gives an example of such interpolation for a small set of data points (circles) and a scalar z parameter. The interpolated data are depicted as stars.

A graphical user front-end (GUI) was implemented in Matlab and is shown in Figures 4 and 5. Its main frame represents the 2D sonic space with the sonic landmarks located in pre-defined spots (the figures show a configurations in which the landmarks were disposed as vertices of a rectangle). The location of the reference sounds can be arbitrarily decided and is specified in a configuration file loaded ad the beginning of each session. Once the landmarks sounds have been loaded and the auditory representation computed, the user can perform a number of actions, including:

- Creating a new sound by choosing with the mouse a position in the proximity of the sonic landmarks whit desirable characteristics
- Listening to landmarks and new sounds by clicking on the spike plot
- Deleting newly generated sounds which are not of interest for the user
- Saving the parametric setting of a new sound as presets in the XML-based format used by the SDT GUI
- Checking network connectivity with the Max/Msp or Pd server running the SDT sound synthesis engine

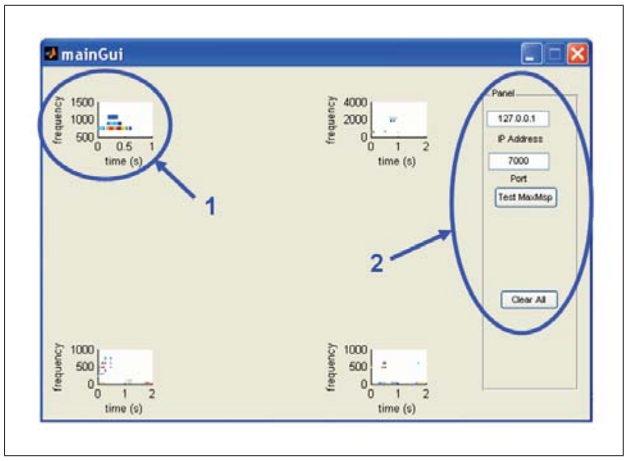


Figure 4. Matlab GUI. Spike representation of four sounds (sonic landmarks) representative of the entire sound space generated by means of the SDT friction model. By clicking in the graphical area around a spike representation it is possible to listen to the corresponding sound (see the area highlighted by circle 1). Circle 2 shows the control panel for the communication with the SDT model in Max/MSP

The spike-based GUI is connected through an OSC network layer to a Max/Msp or Pd server running the SDT sound synthesis engine. When the user confirms the position of a new sound to be generated, the GUI starts the communication with the SDT server, proceeding through the following steps: first, the new interpolated parameters are sent to the server, which updates the controller values; when done, the SDT synthesis engine is started and an auidio file is generated; finally, the GUI loads the audio file, generates the new auditory representation, and plot the result in the 2D sonic space.

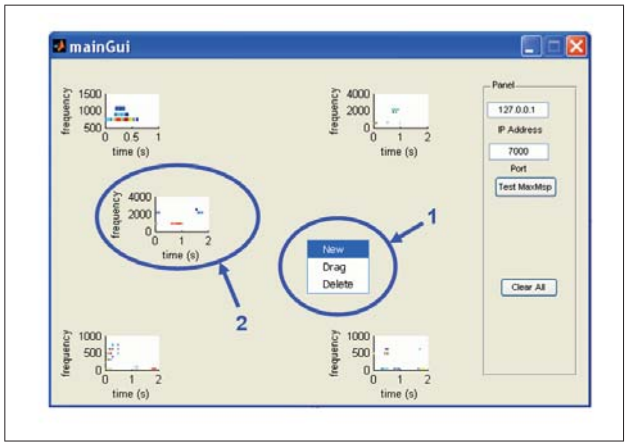


Figure 5. Creation of new sounds. By clicking in any part of the Matlab GUI it is possible to start the computation of a new sound, whose SDT synthesis parameters are calculated according to the graphical position with respect to the four sonic landmarks (see circle 1). The interpolation of the parameters of the four sonic landmarks for the creation of a new sound is linear. The new parameter set is stored as an SDT preset in xml format and acquired by the SDT model in Max/MSP. Circle 2 highlights the Spike representation of a new sound ready generated to be played in Max/MSP.

4 INTERPOLATION EXAMPLES

This section presents the result of a sound design experiment in which six reference sounds (the sonic landmarks) where organized in the 2D space as shown in Figure 6 (the spike representation of the reference sounds, and the Delaunay triangulation generated with this configuration, are shown). Six new sounds were generated by choosing six interpolation positions (depicted as red stars). The synthesis sounds resulting from the interpolated parameters, converted into the spike-based representation, are represented in Figure 7.

Perceptually, the results are in good agreement with the user's expectations, and the dynamical and spectral characteristics are recognized as actually deriving from the characteristics of the neighboring landmarks. The audio files corresponding to the sound landmarks and to the interpolated sounds are available for download at the link provided in the footnote [1] .

[1] http://mordente.sci.univr.it/∼carlodrioli/SoMuCo09/Experiments.htm

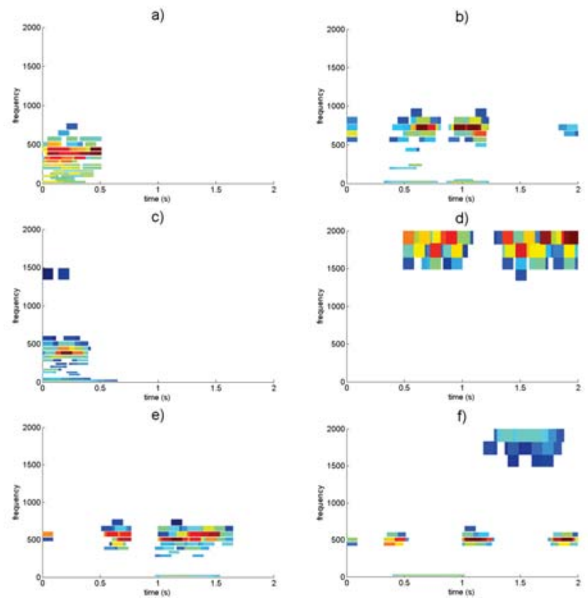


Figure 7. Interpolation example: new sounds (a), b) and c)) are generated by interpolation of points in the same triangle; the new sounds d), e) and f) are located in three different triangles

5 CONCLUSIONS

These results show that the visualization platform helps the sound designer to easily navigate through the soundscape and to find the desired sound fastly. In a following step, this visualization platform will be extended to perform a similar navigation within the soundscape towards a pre-defined (eg. recorded) sound with certain characteristics automatically. A probabilistic method will be defined to model the distribution of the input parameters of the sound model. Given the pre-defined sound, this method will automatically find the optimal input parameters for the sound model, which will produce the closest possible sound to the given sound, in terms of a spike distance. Hence, a distance measure will be defined to measure the distance between the spike code of the pre-defined sound and the spike code of the current sound produced by the sound model.

6 ACKNOWLEDGEMENTS

This work is part of the research carried out within two EU funded project: CLOSED (Closing the Loop of Sound Evaluation and Design) and NIW (Natural Interactive Walking).

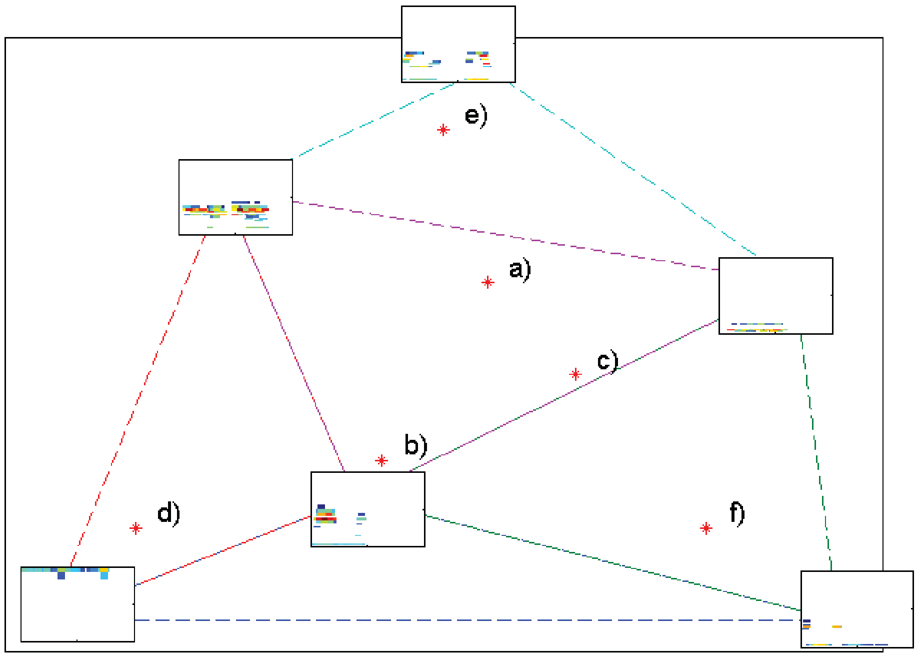


Figure 6. Interpolation example: the new sounds a), b) and c) are generated by interpolation of points in the same triangle; the new sounds d), e) and f) are located in three different triangles

7 REFERENCES

[1] Cumhur Erkut, Vesa Välimäki, Matti Karjalainen, and Henri Penttinen, *Physics-based Sound Synthesis*, vol. Sound to Sense, Sense to Sound. A State of the Art in Sound and Music Computing, chapter 8, pp. 303–343, Logos Verlag, Berlin, 2008.

[2] Murray Schafer, *The Soundscape: Our Sonic Environment and the Tuning of the World*, Rochester, Vermont: Destiny Books, 1994.

[3] Berry Truax, *Handbook for Acoustic Ecology*, ARC Publications, 1978.

[4] Davide Rocchesso and Federico Fontana Eds., *The Sounding Object*, Mondo Estremo, 2003.

[5] Stefano Delle Monache et Al., “Sound design tool: Algorithms for ecologically-founded sound synthesis,” *submitted to Sound and Music Computing Conference 09*, 2009.

[6] E. Smith and M. S. Lewicki, “Efficient coding of time-relative structure using spikes,” *Neural Computation*, vol. 17, pp. 19–45, 2006.

[7] R. D. Patterson and J. Holdsworth, “A functional model of neural activity patterns and auditory images,” *Advances in Speech, Hearing and Language Processing*, vol. 3-B, pp. 547–563, 1992.

[8] S. G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[9] M. Slaney, “A matlab toolbox for auditory modeling work,” *Interval Research Corporation*, 1998.

THREE METHODS FOR PIANIST HAND ASSIGNMENT

Aristotelis Hadjakos
TU Darmstadt
telis@tk.informatik.tu-darmstadt.de

François Lefebvre-Albaret
IRIT Toulouse
lefebvre@irit.fr

ABSTRACT

Hand assignment is the task to determine which hand of the pianist has played a note. We propose three methods for hand assignment: The first method uses Computer Vision and analyzes video images that are provided by a camera mounted over the keyboard. The second and third methods use Kalman filtering to track the hands using MIDI data only or a combination of MIDI and inertial sensing data. These methods have applications in musical practice, new piano pedagogy applications, and notation.

1 INTRODUCTION

This paper presents three methods for real-time hand assignment that are able to determine which hand of the player has played a note. The first method uses Computer Vision to detect the hands in video images provided by a camera mounted over the keyboard. The second method uses MIDI data only while the third method combines MIDI and movement data from inertial sensors worn on the player's wrist. Our methods can be used to improve existing notation applications. Furthermore, we see applications for musical practice and upcoming piano pedagogy applications. In the following, we provide descriptions of applications that would benefit by using our methods:

1) Hand-instrument mapping: Current electronic keyboards, e.g., the Korg X5 [16], typically allow to separate the claviature into two areas, one for the left hand and one for the right, so that the player can play a different instrumental sound with the left hand than with the right. However, to do so, each hand is confined to a fixed area, which is contrary to normal piano practice. Hand assignment methods could eliminate the need for this static boundary and enable a more natural playing experience.

2) New piano pedagogy applications: Sonification of playing movements has a potential to help piano students to become more aware of their playing movements, which can help to improve their technique. One way to perform the sonification is to modify the timbre of a played note according to the movement that leads to it. In order to do so, the computer has to know which hand has played a note so that the movement signal of the corresponding arm is evaluated. We are currently developing such an application based on data from inertial sensors, which are attached to the user's arm.

SMC 2009, July 23-25, Porto, Portugal

3) Notation: To notate a MIDI-recording of a piano performance, it is necessary to perform hand assignment to assign the notes to the correct note system. Current notation software typically assigns notes to hands based on their position relative to a split point. Hand assignment methods could minimize the amount of post-editing by the user.

When comparing the three methods within oneanother each method exposes individual advantages and disadvantages. The camera-based method provides the best accuracy rate, followed by the sensor-based method and the MIDI-based method. However, when using the MIDI-based method no additional hardware is needed. This makes the MIDI-based solution ideal for improving existing notation applications. In order to use the sensor-based method, the user has to attach two clock-like sensors to his wrists, which can be done in an instant. However, to use the camera-based method, the user has to mount the camera over the keyboard. Therefore, the sensor-based method has an advantage over the camera-based method if mobility is important for the user, e.g., if the player frequently has to bring her equipment to rehearsals or concerts. Adverse lighting conditions on a concert stage can be problematic for camera-based hand assignment. The sensor- and MIDI-based methods on the other hand are not influenced by stage conditions. Finally, the sensor-based and the MIDI-based methods are computationally cheaper than the camera-based method and can therefore run on an ordinary microcontroller. Finally, the sensor-based and the MIDI-based methods are computationally cheaper than the camera-based method, making it possible to run them on an ordinary microcontroller. This makes it possible to build a self-contained pedagogical sonification application, which could This makes it possible to build self-contained pedagogical sonification applications that do not rely on an additional laptop or desktop computer.

The remaining paper is structured as follows. Section 2 discusses related work. The three hand assignment methods are presented in the sections 3 to 5. We provide an evaluation in section 6. Conclusion are presented in section 7.

2 RELATED WORK

In this section we report methods for hand assignment and hand tracking. We will discuss voice-separation techniques based on MIDI data and camera-based approaches for hand tracking. Methods for hand assignment using data from inertial sensors are not discussed in literature.

2.1 Methods based on MIDI

Kilian and Hoos proposed a method that finds a separation of a piece into different voices for notation [9]. Chords are allowed to occur in one voice. The method allows the user to select the number of present voices. Therefore, it can be used to find a left hand and right hand part of a MIDI performance (see [9] for notated examples). To separate voices, Kilan's and Hoos' method splits the piece into a sequence of slices with overlapping notes and finds the voice separation by minimizing an elaborate cost function using a stochastic local search algorithm. This approach, while reasonable for notation, cannot be used for real-time hand assignment of a live performance because the slice of overlapping notes cannot be immediately determined when a note is received. Furthermore, the stochastic local search algorithm operates on the entire piece. Other voice separation methods [1, 7, 11] do not allow chords inside a voice and can therefore not be used for hand assignment.

2.2 Methods based on Computer Vision

To detect the two hands in the video, most of the studies make use of a skin color model. To be more robust to illumination changes, other color spaces than RGB, like YUC or HSV, are often used for hand tracking. The hand color distribution can then be modeled as a histogram, a mixture of gaussian, or any other parametric model [14]. Recent studies propose to combine the color information with a displacement information between two consecutive frames [4]. Hands are then identified by their motion and color. Edge detection is often used to refine the estimation of hand shape. After the hand pixels are detected, several algorithms can be applied for hand tracking (an overview is provided in [12]). Algorithms like CamShift, CONDENSATION, etc. give very robust and accurate results as long as there is no hand occlusion. However, they often fail at labeling the right and left hand correctly after a big occlusion. However, overlappings and occlusions frequently occur in piano playing.

Gorodnichy and Yogeswaran developed a system for hand assignment that relies on visual tracking [3]. The system finds the position of the keyboard in the video and identifies the Middle C key. Background subtraction is used to find the hands in the image. Through the identification of cervices in the hand image, fingers are detected. The system annotates MIDI recordings with hand and finger labels. In contrast to our camera-based method, crossing over of the hands is not considered.

3 HAND ASSIGNMENT WITH COMPUTER VISION

In this section, we describe our Computer Vision based tracking algorithm. The pixels belonging to the hands are detected by their color. The hand tracking is performed with a particle filter. The disambiguation of the two hands is achieved by taking into account the principal direction of the hand shape and motion continuity. Finally, the bounding box of the hand is refined to provide accurate positions for the following hand assignment.

3.1 Hand tracking

The detection of skin pixels is made using a Bayesian approach. The skin model is learned form the HSV space of a skin picture that can be changed according to the pianist's skin color. The back-projection of the skin color provides a map of skin color probabilities. This method gives acceptable data for hand-tracking but is not sufficiently accurate to find the hand shape because of the spectral reflexion on the keyboard (see Figure 1). The hand-tracking is made using an annealed particle filter inspired by [2], where each hand is tracked by one cloud of particles. During the tracking process, the cloud pixels of each hand are alternatively subtracted from the skin detection map so that each cloud converges to a different hand. Hand positions are located at the centers of gravity of the particle clouds.

3.2 Hand disambiguation

The problem of hand disambiguation is hard to solve, especially when the two hands overlap frequently. To overcome this problem, we adapted a method originally designed for French Sign Language video processing [10] for pianist hand detection. Hand shape orientation is used as disambiguation criterion. The principal axis of each hand shape is computed after a bounding box has been determined for each hand (see Figure 1) .

We model the joint distribution of the pairs (α_r, α_l) as a gaussian probability density function $f(\alpha_r, \alpha_l, \theta_\alpha)$. Given a pair $(\alpha_1(t), \alpha_2(t))$ of hand shape orientation at the time t, the confidence measure that the hand 1 is the right hand can be computed with the following log-likelihood ratio.

$$llr(t) = \frac{f(\alpha_1(t), \alpha_2(t), \theta_\alpha)}{f(\alpha_2(t), \alpha_1(t), \theta_\alpha)} \quad (1)$$

Continuity of the movement is used as an additional criterion since the hand shape orientation is not sufficient to provide a robust distinction between the hands. While the log-likelihood ratio typically detects the hands correctly, it

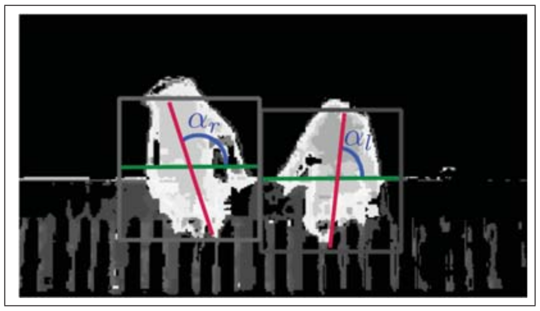


Figure 1. Hand shape orientations

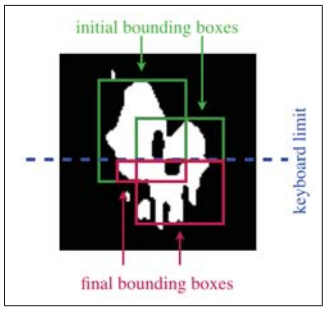


Figure 2. Bounding box refinement

fails if the hands adopt an unusual hand posture. In these occasions the continuity criterion ensures that the hands are not confused.

The two criteria are embedded in an optimization function that has to be maximized between the beginning of the video and the current frame The optimization is achieved with the Viterbi algorithm. Let $dist(t, t-1)$ be the sum of displacements of the right and the left hand between the frame $t-1$ and t then the global optimization function can be written as:

$$argmin \sum_t dist(t, t-1) - c \cdot llr(t) \qquad (2)$$

The value of the weight c was empirically determined.

3.3 Bounding box refinement

Once the two hands have been localized by the particle filter, it is important to localize the part of the hand that is located over the keyboard more precisely by making a bounding box refinement.

Once the two hands have been localized by the particle filter, the intersection of the hand with the keyboard is determined since a large part of the hand may be localized behind the keyboard. To this end, we use bounding box refinement. First, the background is subtracted from the hand image. Outliers are removed using thresholds and morphological operators. The actual hand shape is then considered as being the intersection of a large bounding box surrounding the hand centroid and the keyboard area. As visible in Figure 2, this approach finds the bounding boxes of the hands reliably, even if the two hands overlap.

3.4 Hand assignment

Hand assignment is performed by comparing the horizontal position of the played key in the video with the boundaries of the hands. The decision procedure takes into account whether the played key is inside the span of one hand, both hands, or outside both hands. If the key is inside the span of both hands, it is assigned to the hand where the key more inside the hand span. If the key is outside the span of both hands, it is assigned to hands based on distance. If one hand is outside the keyboard area, no notes will be assigned to it. To calculate the horizontal position of the played key, the procedure uses information about the keyboard position in the video, which is provided once by the user in a visual configuration dialogue.

4 HAND ASSIGNMENT WITH MIDI

The methods for MIDI-based hand assignment (described in this section), and hand assignment based on sensor and MIDI data (described in the next section) are closely related. Both methods are composed of a series of two steps. In the first step a received note-on event is assigned to the left or right hand. In the second step the note-on event is used to modify the estimated position of the corresponding hand.

4.1 Hand assignment

Hand assignment of a note is done with two mechanisms: the identification of unique notes and the examination of the distances of the played note to the estimated hand positions. The method does not allow crossing over of the hands so that the left hand has to be located left of the right hand. It is possible to find simultaneously pressed keys that are located too far from each other to be played by one hand, a condition that will be called a *unique note*. As the hands are not allowed to cross over unique notes can be directly assigned to the left or right hand. Unique notes are identified as notes with an interval of more than an eleventh to the highest or lowest currently pressed key as most players cannot grasp such intervals. If a note is not an unique note, it is assigned to a hand based on the distance of the note to the estimated hand positions..

The position of the hands are estimated with a Kalman filter for each hand. The received note-on events are handed over to the Kalman filter of the assigned hand.

4.2 Position estimation

For each hand, a Kalman filter is used to estimate the position of the hand. The state p of the filter is the position of the center of the hand. The position p is expressed in MIDI units. For example, let the center of the hand lie between the keys corresponding to MIDI pitch values of 60 and 61. Then the position p would be 60.5. The uncertainty of the position is expressed by the variance σ_p^2. The uncertainty of the position decreases when a measurement of hand position is obtained and increases otherwise.

A received note-on event is interpreted as an approximate measurement of hand position. The variance σ_m^2 expresses the uncertainty involved in the measurement. When a note-on message is received, the variance expressing the uncertainty in the position prior to incorporating the measurement $\sigma_p^2(t_2^-)$ is computed. Let t_1 be point in time when the last note was assigned to the Kalman filter and t_2 be the point in time when the new note was received. The uncertainty of the position before incorporating the new measurement $\sigma_p^2(t_2^-)$ is then updated based on the time difference between the two notes $t_2 - t_1$, a constant term σ_s^2, and the previous uncertainty after incorporating the measurement $\sigma_p^2(t_1^+)$.

$$\sigma_p^2(t_2^-) = \sigma_p^2(t_1^+) + (t_2 - t_1) \cdot \sigma_s^2 \qquad (3)$$

The uncertainty of the position after incorporating the measurement $\sigma_p^2(t_2^+)$ is updated based on the uncertainty of the position before incorporating the measurement $\sigma_p^2(t_2^-)$ and the constant term σ_m^2 that expresses measurement uncertainty.

$$\sigma_p^2(t_2^+) = \sigma_p^2(t_2^-) - \frac{\sigma_p^2(t_2^-)}{\sigma_p^2(t_2^-) + \sigma_m^2}\sigma_p^2(t_2^-) \qquad (4)$$

Let n be the pitch of the note received at t_2. Then the new position $p(t_2)$ is estimated based on the old position $p(t_1)$, the uncertainty of the position before incorporating the measurement $\sigma_p^2(t_2^-)$, and the pitch of the received note n.

$$p(t_2) = p(t_1) + \frac{\sigma_p^2(t_2^-)}{\sigma_p^2(t_2^-) + \sigma_m^2}(n - p(t_1)) \qquad (5)$$

Values for σ_s^2 and σ_m^2 were empirically determined.

4.3 Discussion

This section illustrates the method with an example. Say, a user repeatedly plays two notes that are one octave apart with one hand. The first note is played after the hand has been inactive for some time. Therefore, the uncertainty of the hand position is high according to equation 3. Because of the high uncertainty, the new measurement has great influence on the estimated hand position according to equation 5 and the new estimated position will be near the pressed key. The uncertainty of the position reduces because of the new measurement according to equation 4. Because the position uncertainty has been reduced, the next note, which is played one octave apart, receives less weight so that the new position is between the first and second note, slightly towards the second. After several touches, the position uncertainty levels off at a low value controlled by equations 3 and 4 and execution speed. Therefore, new measurements do not drastically change the estimated position. The estimated hand position lies between the two alternating notes and only slightly oscillates when new measurements are made. If the user changes the position of the hand, the estimated position will adapt as older measurements loose influence over time according to equation 3.

5 HAND TRACKING WITH INERTIAL MEASUREMENT AND MIDI

The method described in the previous section can be improved by using measurement of arm movement. This section details on the method based on inertial measurement and MIDI.

To re-position the hand, a player can use various movements of the arm and the body. Despite the many possibilities to move the hand to a given position, players usually reach a position with consistent body and arm posture. Therefore, the angle between the player's forearm and the keyboard can be interpreted as an indication for the position of the hand. The rate of change of this angle can be obtained from an inertial sensor attached to the wrist of the player. However, this measurement provides only information of position change. To obtain absolute hand position, the inertial measurement is combined with the MIDI through Kalman filtering [8].

Similar to the MIDI-based method, unique notes are assigned to the corresponding hand; non-unique notes are assigned to the hands based on the distances of the played note to the positions of the hands.

5.1 Arm movement measurement

To determine the rate of change of the angle between the forearm and the keyboard, which will be called the rate of sideways movement for simplicity, it is necessary to obtain the orientation of the sensor toward gravity. It would be possible to calculate pitch and roll angles directly from the accelerometer signal. However, the playing movements create additional sources of acceleration, which would adversely affect the accuracy. To improve the accuracy of the calculated pitch and roll angles, Kalman filtering is used to fuse accelerometer and gyroscope signals. Given the pitch and roll angle, the rate of sideways movement is calculated from the gyroscope signals.

5.2 Posture measurement

It is necessary to be able to convert a given angle between forearm and keyboard to a hand position (in MIDI pitch units) and vice versa. Because of different movement habits, the relation between playing position and angle has to be measured for each player individually. To this end, the player executes several touches with the same finger in a distance of, for example an octave, over the entire playing range of the keyboard. The change of the angle between two played notes is measured by summation of the rate of sideways movement. The measurement has to be performed for both hands. To convert from hand position to angle between forearm and keyboard and vice versa, linear interpolation between the measured values is used.

5.3 Signal fusion

For each arm, a Kalman filter is used to fuse MIDI and inertial measurement data. The state of the filter is the angle θ between the forearm and the keyboard.

When a new inertial measurement sample is received, the angle is updated. The new angle θ_{i+1} is computed based on the previous angle θ_i, the rate of sideways movement s, and the sample time dt.

$$\theta_{i+1} = \theta_i + s_i \cdot dt \tag{6}$$

Crossing over of the hands is not supported and is avoided by setting s to zero if it would lead to a crossing over condition.

The uncertainty of the angle θ is expressed by the variance σ_θ^2. The uncertainty of the angle θ increases based on σ_s^2, which is the variance of the rate of sideways movement, and the sample time dt.

$$\sigma_{\theta,i+1}^2 = \sigma_{\theta,i}^2 + \sigma_s^2 \cdot dt \tag{7}$$

When a note is assigned to the Kalman filter, the corresponding angle has to be calculated (see section 5.2). The measurement has an effect on the estimated angle and reduces the uncertainty of the angle. Let ϕ be the angle that corresponds to the pressed key that is assigned to the Kalman filter. The new estimate of the angle θ_{i+1} is calculated based on the previous angle θ_i, the previous uncertainty of the angle $\sigma_{\theta,i}^2$, and the angle ϕ.

$$\theta_{i+1} = \theta_i + \frac{\sigma_{\theta,i}^2}{\sigma_{\theta,i}^2 + \sigma_m^2}(\phi - \theta_i) \tag{8}$$

The uncertainty of the position is calculated based on the previous uncertainty and the measurement accuracy which is represented by the constant σ_m^2.

$$\sigma_{\theta,i+1}^2 = \sigma_{\theta,i}^2 - \frac{\sigma_{\theta,i}^2}{\sigma_{\theta,i}^2 + \sigma_m^2}\sigma_{\theta,i}^2 \tag{9}$$

Values for σ_s^2 and σ_m^2 were empirically determined.

6 EVALUATION

To evaluate hand assignment accuracy, performances of different piano pieces were analyzed with our methods. To this end, video, MIDI, and inertial measurement signals were recorded with one pianist playing different pieces. The recordings were conducted with the inertial sensors that we previously developed [5], which provide six degrees of freedom measurements of acceleration and angular rates at an update frequency of 100 Hz.

Simple approaches to automatically evaluate the hand assignment results, for example by using score-following to match the obtained separation with a given correct separation, are problematic because of playing errors and differences because of ornamentation. Therefore, the results were manually examined. To this end it was necessary to present the result in a human-readably way. We used the text-based GUIDO format [6, 13] to create graphical musical scores for the left and right hand part. The human reader can then identify correct and wrong assignments.

The recorded pieces were the Sinfonias 1–5 by J. S.Bach (BWV 787–791) and the "Six Dances in Bulgarian Rhythm" (No. 148–153) from Bartok's Mikrokosmos vol. 6. Bartok's dances contain many instances where the hands overlap. Furthermore the dances contain frequent changes of hand position on the keyboard, which are performed very quick re-positioning movements. Also the hands are crossing over several times. Therefore, the dances are especially challenging for hand assignment.

The accuracies of the obtained hand assignments are shown in Table 1. For comparison with a baseline, we include the results of hand assignment with the split point method (split point is the Middle C). For all examined pieces, the our methods achieves better results than the split point method. The sensor-based method typically achieves better results than MIDI-based method. The camera-based method typically achieves better results than the MIDI-based and sensor-based methods.

The Bulgarian dance No. 152 shows a limitation of our methods. The hands often completely overlap in this piece, i.e., one hand is positioned above the other hand while both hands play notes in the same range. This is contrary to the assumptions of our methods, as they implicitly split the keyboard at a (time-variable) split-point. Therefore, our methods are not able to perform hand-assignment correctly if, e.g., the right hand plays a note that lies between two notes that are played with the left hand.

The camera-based hand disambiguation allows hands to be identified during crossing over. The disambiguation is successful when the hands cross over distinctly. To improve the disambiguation, the probability model (see equation 1) could be extended to additionally include the positions of the hands as the distribution of hand orientation changes according to its position on the keyboard.

Table 1. Hand assignment accuracy

Piece	Split	MIDI	Inert.	CV
Sinfonia 1	86.6%	97.6%	98.6%	97.8%
Sinfonia 2	86.4%	94.3%	97.0%	98.6%
Sinfonia 3	94.2%	97.3%	98.3%	99.7%
Sinfonia 4	90.9%	97.5%	98.6%	98.9%
Sinfonia 5	97.2%	99.4%	99.3%	99.6%
No. 148	81.8%	88.7%	91.2%	94.2%
No. 149	79.4%	82.5%	89.6%	88.1%
No. 150	81.9%	86.4%	83.6%	90.8%
No. 151	69.8%	83.9%	87.8%	93.6%
No. 152	65.2%	66.6%	68.4%	70.6%
No. 153	78.2%	85.6%	91.2%	92.5%

The and sensor- and MIDI-based methods could be improved by using a hand model that filters out impossible hand-note configurations, which could be used instead of the unique-note mechanism (see section 4.1).

7 CONCLUSION

The main contributions of this paper are three methods for hand assignment: The first method is based on video images from a camera mounted over the keyboard, the second method is based on MIDI, and the third method combines inertial measurement and MIDI. The methods are real-time capable and can therefore be used for hand assignment in interactive scenarios like hand-instrument mapping and for new piano pedagogy applications. The methods were evaluated by running them on performances of pieces by Bach and Bartok. Applications of our methods are instrument-hand mapping, new piano pedagogy applications, and notation applications.

8 REFERENCES

[1] Chew, E. and Wu, X. "Separating voices in polyphonic music: A contig mapping approach", *Computer Music Modeling and Retrieval: Second International Symposium*, 2004.

[2] Gianni F., Collet C., Dalle P. "Robust tracking for processing of videos of communications gestures." *GW 2007*, 2007.

[3] Gorodnichy, D. O. and Yogeswaran, A. "Detection and tracking of pianist hands and fingers", *CRV '06: Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*, 2006.

[4] Habili, N., Lim, C. C., and Moini, A. "Segmentation of the face and hands in sign language video sequences using color and motion cues", *IEEE Trans. Circuits Syst. Video Techn.*, 14(8), 2004.

[5] "SYSSOMO: A Pedagogical Tool for Analyzing Movement Variants Between Different Pianists", *Enactive08 Proceedings*, 2008.

[6] Hoos, H. H., Hamel, K. A., Renz, K., and Kilian, J. "The GUIDO Music Notation Format - A Novel Approach for Adequately Representing Score-level Music", *ICMC'98 Proceedings*, 1998.

[7] Jordanous, A. "Voice Separation in Polyphonic Music: A Data-Driven Approach", *ICMC 2008*, 2008.

[8] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems", *J. Basic Eng.*, 1960.

[9] Kilian, J. and Hoos, H.H. "Voice separation—a local optimisation approach", *ISMIR 2002*, 2002.

[10] Lefebvre-Albaret F., Dalle P. "Body posture estimation in a sign language video", *GW 2009*, 2009.

[11] Madsen, S.T. and Widmer, G. "Separating voices in MIDI", *Proceedings of the 9th International Conference in Music Perception and Cognition (ICMPC2006)*, 2006.

[12] Mahmoudi, F. and Parviz, M. "Visual Hand Tracking Algorithms", *Geometric Modeling and Imaging–New Trends*, 2006.

[13] Renz, K., "An Improved Algorithm for Spacing a Line of Music", *Proceedings of the ICMC 2002*, 2002.

[14] Vezhnevets, V., Sazonov, V., Andreeva, A. "A Survey on Pixel-Based Skin Color Detection Techniques". *Proc. Graphicon-2003*, 2003.

[15] Zieren, J., Unger, N., Akyol, S., "Hands Tracking from Frontal View for Vision-Based Gesture Recognition" LNCS 2449, Springer, 2002.

[16] Korg X5 Music Synthesizer AI2 Synthesis System Bedienungshandbuch, 1994.

INSTRUMENT AUGMENTATION USING ANCILLARY GESTURES FOR SUBTLE SONIC EFFECTS

Otso Lähdeoja
CICM
Université Paris 8
MSH Paris-Nord, France
otso.Lahdeoja@gmail.com

Marcelo M. Wanderley **Joseph Malloch**
Input Devices and Music Interaction Laboratory,
McGill University,
Montreal, QC, Canada
{marcelo.wanderley, joseph.malloch}@mcgill.ca

ABSTRACT

In this paper we present an approach to instrument augmentation using the musician's ancillary gestures to enhance the liveliness of real-time digitally processed sound. In augmented instrument praxis, the simultaneous control of the initial instrument and its' electric/electronic extension is a challenge due to the musician's physical and psychological constraints. Our work seeks to address this problem by designing non-direct gesture-sound relationships between ancillary gestures and subtle sonic effects, which do not require a full conscious control of the instrumentalist. An application for the electric guitar is presented on the basis of an analysis of the ancillary movements occurring in performance, with specific gesture data acquisition and mapping strategies, as well as examples of musical utilizations. While the research work focuses on the electric guitar, the system is not instrument-specific, and can be applied to any instrument using digital sound processing.

1.INTRODUCTION

Instrument augmentation with real-time digital audio signal processing offers numerous possibilities for musical performance. However, integrating new features into an already complex instrumental playing environment is constrained by the musician's physical and psychological capacities of accomplishing multiple and simultaneous tasks [3][10]. There is always a tradeoff between the extended sonic possibilities and the ability of the performer to dynamically control them. In musical applications this often translates as an inability to simultaneously control the numerous parameters of the sound processing algorithms; a specific "effect" is applied to the sound and only its most prevalent perceptive features are controlled in time. The remaining "secondary" parameters typically stay static or are modulated by fixed-frequency LFO's. This leads to a lack of "liveliness" in the real-time processed sound, and certainly makes for an underuse of the processing algorithms' sonic possibilities. In our view, significant areas of sonic effect, subtlety and nuance are left unused. One established strategy of enhancing multi-parameter variation of DSP in time is the adaptative digital audio effects (A-DAFX): a content-based transformation where the sound features provide data to control processing parameters [12]. In the present work, we investigate another strategy, namely the possibility of connecting the performer's sound-ancillary gestures to the evolution of the processing parameters [14]. Our hypothesis is that instrumentalist's movements which are not directly involved in the creation of the sound convey performance-related information which may be used to enhance the liveliness of digitally processed sound.

SMC 2009, July 23-25, Porto, Portugal

1.2.ANCILLARY GESTURES AND MUSICAL APPLICATIONS

Musician's performance movements which are not directly related to the production or sustain of the sound have received academic attention. In his study of Glenn Gould's piano performance videos, F. Delalande identified three levels of gesture which form a continuum going from purely functional to purely symbolic [4]. With his work on clarinet players' movements, M. Wanderley established the term ancillary gestures, signifying movements occurring in the performance which are not directly related to the production of sound [13]. This discretization of musician's movements was continued with the following typology [11]:
• Sound-producing gestures,
• Ancillary gestures (support sound-producing gestures)
• Sound-accompanying gestures (musically "engaged" body movements not involved in the sound production)
• Communicative gestures (communication between performers and towards the public)

In this article, we adopt the motion-related terminology mentioned above. Our project concentrates on the use of both ancillary and sound-accompanying gestures.

In instrument augmentation there is a general tendency to work on direct causal relationships between gesture and sound, either using the acoustic instrument's sound-producing gestures or by adding new interfaces with direct mappings. The use of ancillary or sound-accompanying movements is rather rare in this area. Previous applications include a flanger-effect control with clarinetist’s ancillary movements [14], a weight balance tracking floor module [8], and the *Multimodal Music Stand* which tracks the instrument's tilt and the performer's head movements [1]. Our current project seeks to address the specific problem of digital signal processing "liveliness" by using the performer's sound-accompanying movements for dynamic variation of DSP parameters which affect the sound's subtle perceptive features (cf. 3.2). Our system introduces a second level of gesture-sound relationship into the augmented instrument-playing environment, where sound-accompanying movements provide complementary control data for the signal processing.

3. GESTURE-SOUND RELATIONSHIP: AN APPLICATION FOR THE ELECTRIC GUITAR

The initial research work was carried out on the electric guitar and its related set of movements.

3.1.Sound-accompanying movements

The electric guitar is traditionally related to popular music styles such as jazz and rock where there is a social acceptance and even expectation for an engaged performance body [5]. Embodied expression and rhythm are part of the guitar playing tradition. The sound-accompanying movements being thus relatively emphasized, the electric guitar provides for a fertile testing ground for our project. In order to analyze the different movements of electric guitar performance in standing position, we first set out to review video excerpts from a number of players. Subsequently we filmed a series of performances in laboratory conditions, allowing for a finer analysis of the players' movements and their relationships to the instrument and to each other.

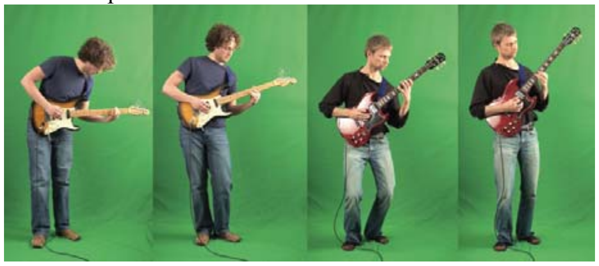


Figure 1. Still images taken from our filmed guitar performances. Note the weight shifts and the head movements.

The analysis of our video excerpts revealed a set of sound-producing gestures on the hands, prolonged by ancillary movements of the arms and shoulders, and to a lesser degree, adjustments of the pelvis and torso to maintain the instrument level. Movements which were less involved in sound production included more ample torso movements, weight shifts, knee bends and head movements. A distinctive feature was foot tapping, which occurred often. While being highly individual in style and conduct, these basic movements were abundant and rather consistent from one performer to another. Concentrating solely on the sound-accompanying movements, we distinguished two fairly independent motion ensembles: firstly weight shifts resulting from the leg (knee) and torso movements and secondly seemingly autonomous movements of the head. Consequently, we decided to focus on the extremities of the body, the most remote areas from the sound-producing central area: head movements and body weight shifts.

3.2."Subtle" perceptive sound features

In hybrid acoustic-electric/electronic instruments such as the electric guitar, two levels of gesture-sound relationship coexist. The first is that of an acoustic instrument, where a direct causality connects gesture to sound through energy transduction from kinetic energy to acoustic waves [2]. This is not the case for the second level: the electric/electronic part of the hybrid instrument where the sound processing is not physically related to the initial gesture, the relationship between body and sound having to be defined via mapping strategies. The two levels of the "electroacoustic instrument" have specific sonic functions: the acoustic level determines the fundamental articulation of the musical discourse (pitch, duration, rhythm, basic timbre), whereas the electric/electronic level determines timbral transformations, spatialization, and sound structure modifications. The range of transformations applied on the initial gesture-related signal varies from subtle to radical. In this project, we seek to work on the more perceptively subtle features of the hybrid instrument's electric/electronic level's sonic possibilities, such as those appealing to the perception as modifications of sound color, presence, space and timbre. We use the term "subtle perceptive sound features" to term these somewhat moderate effects. In a technical perspective, this translates into inducing minor variations on the sound's spectrum, spatialization, reverberation, delay and granularity or distortion, among other numerous possibilities.

4.A NON-DIRECT GESTURE-SOUND RELATIONSHIP

In the present project, we investigate the possibility of avoiding saturating the augmented instrumental environment by introducing non-direct relationships between movement and sound into it. By voluntarily designing "loose" gesture-sound relationships we aim to give the musician the sensation of sound transformations which accompany the performance, without demanding conscious attention [7]. This enables for better concentration on the essential features of the instrument, while providing performance-related data for the signal processing. The established relation between movement and sound can be qualified as "loosely causal"; body motion induces changes in the sound's subtle perceptive features, but not in a direct, predictable manner. In order to achieve this "blurred" gesture-sound relationship, we explored the following design principles: complex, multi-layer mappings [6], the non-reproducibility of one gesture-sound couple and maintaining the sonic effects subtle (ensuring that the musician does not feel "out of control"). In our view, these characteristics are necessary to provide a system that the musician may accept as a part of his/her instrument. The augmentation should find an equilibrium in the playing environment, not too obvious for the performer to be tempted to "take control" yet not too distant from the gesture so as not to appear as intrusive and disturbing. The introduction of "loose" connections establishes a hierarchy in the gesture-sound relationships of the augmented playing environment: the central constituents of the musical discourse are controlled by focused gestures and direct mappings, while secondary sonic features are induced by peripheral, non-conscious movements.

5.SYSTEM DESCRIPTION

5.1.Ancillary gesture data acquisition

The system extracts data from a selection of guitarist's ancillary gestures and uses it to produce sound variations via a two-layer mapping strategy. This system functions in parallel with the traditional instrument. The sound-accompanying movements are sampled in three points: on top of the head with a two-axis accelerometer and under both feet with a board sensing the weight distribution with four equidistant FSRs. The system outputs seven channels of data, which are grouped to represent three distinct variables: general "amount" of head movement, left/right and back/front weight distribution. In addition, the system seeks to capture particularly dramatic body imbalances such as standing on one foot, on the toe(s), or on the heel(s), by polling the four FSR sensor network for relative peak values. A detection of the feet's tapping movements is implemented, and it will be upgraded with a tempo tracking algorithm in a near future version of the system.

5.2.A kinetic metaphor as a mapping strategy

Our system seeks to establish a non-direct relationship between movement and sound through a two-layer mapping: the gesture data is first routed as an input to a mass-spring physical model which in turn modulates the sound processing parameters. The mapping strategy is inspired by a kinetic metaphor where the performance gestures are seen as energy inducing motion in an independent system affecting the sound processing parameters. The physical model has its own behavior and inertia which are determined according to the specificities of the target parameters. The mapping was implemented using the msd/pmpd physical model in the max/msp environment [9]. Each of the main input variables (head movement, 2-dimension weight distribution) feeds a distinct model, resulting in three mass-damper systems affecting the sound simultaneously. A minute tuning is required to achieve a "blurred yet connected" perception of the gesture-sound relationship. This may be achieved through careful selection of the physical models' inherent attributes (i.e. behavior), and by mapping relationships between the models' output and chosen DSP parameters.

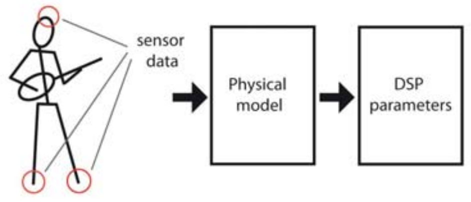


Figure 2. System schema

5.3.Applications

A series of test applications was implemented in max/msp, based on the electric guitar's use of real-time "effects". Testing our system in a musical praxis proved highly interesting, providing subjective insights to the "feel" of a non-direct gesture-sound relationship and to its' tuning via mapping strategies. Through practice, the head gestures and the body weight shifts acquired distinct connections to different sound parameters: the head movements linked well to minute, relatively high frequency variations of the sound's spectrum and space, while the weight shifts were used for slower and more important transformations of the soundscape. Following are some examples of the applications:

• *Spectral panning*: Body weight shifts were mapped to two physical models driving a four-section spectral stereo panning effect. Weight shifts would induce increased stereo field motion of the spectral divisions, while absence of motion would bring the soundscape to a standstill at the center. Head movements were mapped to the gain of an unobtrusive stereo delay, increasing the kinetic effect of the panning.

• *Autofilter*: A combination of an adaptative audio effect (A-DAFX) and non-direct gestural control. The autofilter would respond to the amplitude of the notes played, while the ancillary gestures would modulate the behavior of the filter. The weight shifts were mapped to the filter base cutoff frequency and the head movements to the filter's "slope" (Q). This produced a lively and musically rewarding autofilter application for the instrumentalist.

• *Drive and granulation*: This application sought to induce subtle variations in the saturation or "drive" of the sound; an important timbre element of the electric guitar. Weight balance was connected to the saturation level, and head movements to the Q of a low pass filter. A variation of the effect was implemented with the msp munger~ object, working on the sound's granularity instead of saturation.

• *Stereo delay*: Weight shifts were used to control the stereo spread of two delay channels, similarly to the spectral panning application. The head movements were mapped to an amplitude modulation (tremolo) affecting the delay lines.

Gesture type	DSP parameters
Weight shifts (left/right & front/rear)	- Spectral panning width - Autofilter base cutoff freq. - Overdrive/granulation level - Stereo delay spread
Head movements	- Delay gain - Autofilter Q - Lowpass filter Q - Tremolo amplitude

Table 1. The ancillary movements and their corresponding DSP parameters used in the test sessions

The possibilities are numerous, and the system may be adapted to any instrument using digital signal processing. A case-specific study of the instrument's characteristic ancillary gestures is nevertheless necessary[1].

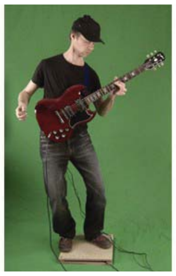


Figure 3. A still image from the filmed test sessions with the sensor board and head accelerometer

6.CONCLUSION AND FUTURE WORK

In this paper we presented a system connecting guitarist's ancillary and sound-accompanying gestures to subtle variations of the digitally processed sound. The resulting gesture-sound relationship is designed to be "loose", i.e. not requiring conscious control from the instrumentalist while providing gesture-driven sonic variations. The adopted strategy uses multi-layer mapping and physical models to establish a non-direct control of the DSP parameters. The initial results were musically inspiring, and they point to a vast domain of research on ancillary gesture-sound relationships, their perception (both from the musician's and the public's perspective), and their use in performance. These results suggest that a well-integrated ancillary gesture based control system for enhancing the liveliness of digitally processed sound is a promising perspective for instrument augmentation.

[1]Video examples of the augmentations can be viewed at: http://lahdeoja.org/ftplahdeoja/ancillary_gesture/

7.ACKNOWLEDGEMENTS

The authors would like to thank the COST-SID program and the Kordelin Foundation for funding the research work, as well as Bertrand Scherrer for help in the guitarist ancillary gesture study.

8.REFERENCES

[1]Bell, B. et al. "The Multimodal Music Stand" *Proceedings of the 2007 Conference on New Interfaces for Musical Expression*, New York, USA, 2007

[2]Cadoz C. "Musique, geste, technologie" *Les nouveaux gestes de la musique* éd. Parenthèses, Marseille, 1999

[3]Cook, P. "Principles for Designing Computer Music Controllers" *Proceedings of the 2001 Workshop on New Interfaces for Musical Expression,* Seattle, U.S.A.

[4]Delalande, F. "La gestique de Gould", *Glenn Gould pluriel*, Momentum, Québec, Canada 2007 p. 135

[5]Fast, S. *In the houses of the holy, Led Zeppelin and the power of rock*, Oxford University Press, 2001, pp. 123

[6]Hunt, A. et al. "The Importance of Parameter Mapping in Electronic Instrument Design" *Journal of New Music Research*, Vol. 32, No. 4, 2003 pp. 429–440

[7]Marshall, M. ; Malloch, J. ; Wanderley, M. M. "Non-conscious Gesture Control of Sound Spatialization" *Proceedings of the enactive 07 conference, Grenoble*, France, 2007

[8]McElligott, L. Dixon, E. Dillon, M. "'*PegLegs* in Music' - Processing the Effort Generated by Levels of Expressive Gesturing in Music" *Proceedings of the 2002 Conference on New Instruments for Musical Expression, Dublin, Ireland*

[9]Montgermont M. MSD externals for max/msp http://nim.on.free.fr/index.php?id=software

[10]Pressing, J. "Psychological constraints on improvisational expertise and communication" *In the course of performance - studies in the world of musical improvisation*, University of Chicago Press, 1998

[11]Refsum Jensenius, A. et al. "Concepts and methods in research on music-related gestures", *ConGAS book*, in press

[12]Verfaille, V. ; Wanderley, M. M. ; Depalle, P. "Mapping Strategies for Gestural and Adaptive Control of Digital Audio Effects" *Journal of New Music Research* 2006, Vol. 35, No. 1, pp. 71 – 93

[13]Wanderley M. M. ; Vines, B. "Origins and functions of clarinettist's Ancillary Gestures", *Music and Gesture* Ashgate, Hampshire, England, 2006 p.167

[14]Wanderley, M. M. ; Depalle, P. "Gesturally-controlled digital audio effects" *Proceedings of the Conference on Digital Audio Effects (DAFX-01), Ireland, 2001*

A CHROMA-BASED SALIENCE FUNCTION FOR MELODY AND BASS LINE ESTIMATION FROM MUSIC AUDIO SIGNALS

Justin Salamon
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
justin.salamon@upf.edu

Emilia Gómez
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
emilia.gomez@upf.edu

ABSTRACT

In this paper we present a salience function for melody and bass line estimation based on chroma features. The salience function is constructed by adapting the Harmonic Pitch Class Profile (HPCP) and used to extract a mid-level representation of melodies and bass lines which uses pitch classes rather than absolute frequencies. We show that our salience function has comparable performance to alternative state of the art approaches, suggesting it could be successfully used as a first stage in a complete melody and bass line estimation system.

1 INTRODUCTION

With the prevalence of digital media, we have seen substantial growth in the distribution and consumption of digital audio. With musical collections reaching vast numbers of songs, we now require novel ways of describing, indexing, searching and interacting with music.

In an attempt to address this issue, we focus on two important musical facets, the melody and bass line. The melody is often recognised as the *'essence'* of a musical piece [11], whilst the bass line is closely related to a piece's tonality [8]. Melody and bass line estimation has many potential applications, an example being the creation of large databases for music search engines based on Query by Humming (QBH) or by Example (QBE) [2].

In addition to retrieval, melody and bass line estimation could facilitate tasks such as cover song identification and comparative musicological analysis of common melodic and harmonic patterns. An extracted melodic line could also be used as a reduced representation (thumbnail) of a song in music applications, or on limited devices such as mobile phones. What is more, a melody and bass line extraction system could be used as a core component in other music computation tasks such as score following, computer participation in live human performances and music transcription systems. Finally, the determination of the melody and bass line of a song could be used as an intermediate step towards the determination of semantic labels from musical audio, thus helping to bridge the *semantic gap* [14].

SMC 2009, July 23-25, Porto, Portugal
Copyrights remain with the authors

Much effort has been devoted to the extraction of a score representation from polyphonic music [13], a difficult task even for pieces containing a single polyphonic instrument such as piano or guitar. In [8], Goto argues that musical transcription (i.e. producing a musical score or piano roll like representation) is not necessarily the ideal representation of music for every task, since interpreting it requires musical training and expertise, and what is more, it does not capture non-symbolic properties such as the expressive performance of music (e.g. vibrato and ornamentation). Instead, he proposes to represent the melody and bass line as time dependent sequences of fundamental frequency values, which has become the standard representation in melody estimation systems [11].

In this paper we propose an alternative mid-level representation which is extracted using a salience function based on chroma features. Salience functions provide an estimation of the predominance of different fundamental frequencies (or in our case, pitch classes) in the audio signal at every time frame, and are commonly used as a first step in melody extraction systems [11]. Our salience function makes use of chroma features, which are computed from the audio signal and represent the relative intensity of the twelve semitones of an equal-tempered chromatic scale. As such, all frequency values are mapped onto a single octave. Different approaches to chroma feature extraction have been proposed (reviewed in [5]) and they have been successfully used for different tasks such as chord recognition [4], key estimation [6] and similarity [15].

Melody and bass line extraction from polyphonic music using chroma features has several potential advantages – due to the specific chroma features from which we derive our salience function, the approach is robust against tuning, timbre and dynamics. It is efficient to compute and produces a final representation which is concise yet maintains its applicability in music similarity computations (in which an octave agnostic representation if often sought after, such as [10]). In the following sections we present the proposed approach, followed by a description of the evaluation methodology, data sets used for evaluation and the obtained results. The paper concludes with a review of the proposed approach and consideration of future work.

2 PROPOSED METHOD

2.1 Chroma Feature Computation

The salience function presented in this paper is based on the *Harmonic Pitch Class Profile* (HPCP) proposed in [5]. The HPCP is defined as:

$$HPCP(n) = \sum_{i=1}^{nPeaks} w(n, f_i) \cdot a_i^2 \qquad n = 1 \ldots size \quad (1)$$

where a_i and f_i are the linear magnitude and frequency of peak i, $nPeaks$ is the number of spectral peaks under consideration, n is the HPCP bin, $size$ is the size of the HPCP vector (the number of HPCP bins) and $w(n, f_i)$ is the weight of frequency f_i for bin n. Three further pre/post-processing steps are added to the computation. As a preprocessing step, the tuning frequency is estimated by analyzing frequency deviations of peaks with respect to an equal-tempered scale. As another preprocessing step, spectral whitening is applied to make the description robust to timbre. Finally, a post-processing step is applied in which the HPCP is normalised by its maximum value, making it robust to dynamics. Further details are given in [5].

In the following sections we detail how the HPCP computation is configured for the purpose of melody and bass line estimation. This configuration allows us to consider the HPCP as a salience function, indicating salient pitch classes at every time frame to be considered as candidates for the pitch class of the melody or bass line.

2.2 Frequency Range

Following the rational in [8], we assume that the bass line is more predominant in the low frequency range, whilst the melody is more predominant in the mid to high frequency range. Thus, we limit the frequency band considered for the HPCP computation, adopting the ranges proposed in [8]: 32.7Hz (1200 cent) to 261.6Hz (4800 cent) for bass line, and 261.6Hz (4800 cent) to 5KHz (9907.6 cent) for melody. The effect of limiting the frequency range is shown in Figure 1. The top pane shows a chromagram (HPCP over time) for the entire frequency range, whilst the middle and bottom panes consider the melody and bass ranges respectively. In the latter two panes the correct melody and bass line (taken from a MIDI annotation) are plotted on top of the chromagram as white boxes with diagonal lines.

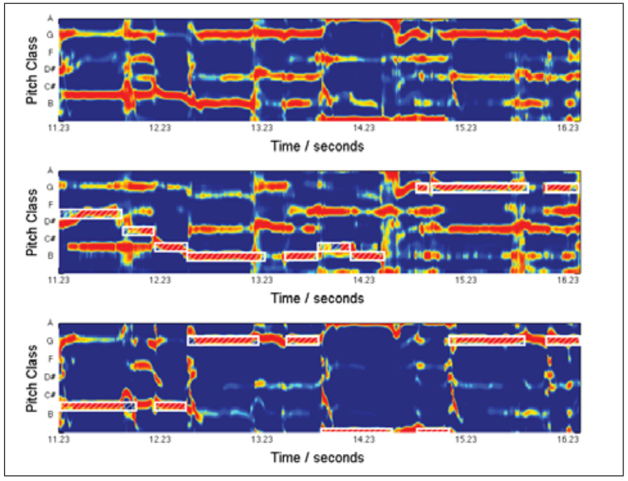


Figure 1. Original (top), melody (middle) and bass line (bottom) chromagrams

2.3 HPCP Resolution and Window Size

Whilst a 12 or 36 bin resolution may suffice for tasks such as key or chord estimation, if we want to properly capture subtleties such as vibrato and glissando, as well as the fine tuning of the singer or instrument, a higher resolution is needed. In Figure 2 we provide an example of the HPCP for the same 5 second segment of *train05.wav* from the MIREX 2005 collection, taken at a resolution of 12, 36, and 120 bins. We see that as we increase the resolution, elements such as glissando (seconds 1-2) and vibrato (seconds 2-3) become better defined. For the rest of the paper we use a resolution of 120 bins.

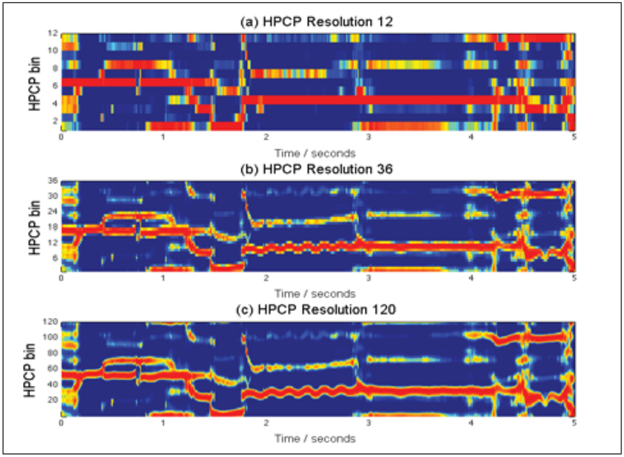


Figure 2. HPCP computed with increasing resolution

Another relevant parameter is the window size used for the analysis. A smaller window will give better time resolution hence capturing time-dependent subtleties of the melody, whilst a bigger window size gives better frequency resolution and is more robust to "noise" in the analysis (single frames in which the melody is temporarily not the most salient). We empirically set the window size to 186ms (due to the improved frequency resolution given by long windows, their use is common in melody extraction [11]).

2.4 Melody and Bass Line Selection

Given our salience function, the melody (or bass line depending on the frequency range we are considering) is selected as the highest peak of the function at every given time frame. The result is a sequence of pitch classes (using a resolution of 120 HPCP bins, i.e. 10 cents per pitch class) over time. It is important to note that no further post processing is performed. In [11] a review of systems participating in the MIREX 2005 melody extraction task is given, in which a common extraction architecture was identified. From this architecture, we identify two important steps that would have to be added to our approach to give a complete system: firstly, a postprocessing step for selecting the melody line out of the potential candidates (peaks of the salience function). Different approaches exist for this step, such as streaming rules [3], heuristics for identifying melody characteristics [1], Hidden Markov Models [12] and tracking agents [8]. Then, voicing detection should be applied to determine when the melody is present.

3 EVALUATION METHODOLOGY

3.1 Ground Truth Preparation

For evaluating melody and bass line estimation, we use three music collections, as detailed below.

3.1.1 MIREX 2004 and 2005 Collections

These collections were created by the MIREX competition organisers for the specific purpose of melody estimation evaluation [11]. They are comprised of recording-transcription pairs, where the transcription takes the form of timestamp-F0 tuples, using 0Hz to indicate unvoiced frames. 20 pairs were created for the 2004 evaluation, and another 25 for the 2005 evaluation of which 13 are publicly available [1]. Tables 1 and 2 (taken from [11]) provide a summary of the collection used in each competition.

3.1.2 RWC

In an attempt to address the lack of standard evaluation material, Goto et al. prepared the *Real World Computing* (RWC) Music Database [7]. It contains several databases of different genres, and in our evaluation we use the Popular Music

[1] `http://labrosa.ee.columbia.edu/projects/melody/`

Category	**Style**	**Melody Instrument**
Daisy	Pop	Synthesised voice
Jazz	Jazz	Saxophone
MIDI	Folk, Pop	MIDI instruments
Opera	Classical Opera	Male voice, Female voice
Pop	Pop	Male Voice

Table 1. Summary of data used in the 2004 melody extraction evaluation

Melody Instrument	**Style**
Human voice	R&B, Rock, Dance/Pop, Jazz
Saxophone	Jazz
Guitar	Rock guitar solo
Synthesised Piano	Classical

Table 2. Summary of data used in the 2005 melody extraction evaluation

Database. The database consists of 100 songs performed in the style of modern Japanese (80%) and American (20%) popular music typical of songs on the hit charts in the 1980s and 1990s.

At the time of performing the evaluation the annotations were in the form of MIDI files which were manually created and not synchronised with the audio [2]. To synchronise the annotations, we synthesised the MIDI files and used a local alignment algorithm for HPCPs as explained in [15] to align them against the audio files. All in all we were able to synchronise 73 files for evaluating melody estimation, of which 7 did not have a proper bass line leaving 66 for evaluating bass line estimation (both collections are subsets of the collections used for evaluating melody and bass line transcription in [13] [3]).

3.2 Metrics

Our evaluation metric is based on the one first defined for the MIREX 2005 evaluations. For a given frame n, the estimate is considered correct if it is within $\pm\frac{1}{4}$ tone (± 50 cents) of the reference. In this way algorithms are not penalised for small variations in the reference frequency. This also makes sense when using the RWC for evaluation, as the use of MIDI annotations means the reference frequency is discretised to the nearest semitone. The concordance error for frame n is thus given by:

$$err_n = \begin{cases} 100 & \text{if } |f^{est}_{cent}[n] - f^{ref}_{cent}[n]| > 50 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

[2] A new set of annotations has since been released with audio synchronised MIDI annotations.

[3] With the exception of *RM-P034.wav* which is included in our evaluation but not in [13].

The overall transcription concordance (the score) for a segment of N frames is given by the average concordance over all frames:

$$score = 100 - \frac{1}{N}\sum_{n=1}^{N} err_n \qquad (3)$$

As we are using chroma features (HPCP) to describe melody and bass lines, the reference is mapped onto one octave before the comparison (this mapping is also used in the MIREX competitions to evaluate the performance of algorithms ignoring octave errors which are common in melody estimation):

$$fchroma_{cent} = 100 + mod(f_{cent}, 1200) \qquad (4)$$

Finally it should be noted that as voicing detection is not currently part of our system, performance is evaluated for voiced frames only.

4 RESULTS

In this section we present our melody and bass line estimation results, evaluated on the three aforementioned music collections. For comparison we have also implemented three salience functions for multiple-F0 estimation proposed by Klapuri in [9] which are based on the summation of harmonic amplitudes (henceforth referred to as the Direct, Iterative and Joint methods). The Direct method estimates the salience $s(\tau)$ of a given candidate period τ as follows:

$$s(\tau) = \sum_{m=1}^{M} g(\tau, m)|Y(f_{\tau,m})| \qquad (5)$$

where $Y(f)$ is the STFT of the whitened time-domain signal, $f_{\tau,m} = m \cdot f_s/\tau$ is the frequency of the m^{th} harmonic partial of a F0 candidate f_s/τ, M is the total number of harmonics considered and the function $g(\tau, m)$ defines the weight of partial m of period τ in the summation. The Iterative method is a modification of the Direct method which performs iterative estimation and cancellation of the spectrum of the highest peak before selecting the next peak in the salience function. Finally the Joint method is a further modification of the Direct method which attempts to model the Iterative method of estimation and cancellation but where the order in which the peaks are selected does not affect the results. Further details are given in [9]. The three methods were implemented from the ground up in Matlab, using the parameters specified in the original paper, a window size of 2048 samples (46ms) and candidate periods in the range of 110Hz-1KHz (the hop size was determined by the one used to create the annotations, i.e. 5.8ms for the MIREX 2004 collection and 10ms for the MIREX 2005 and RWC collections).

4.1 Estimation Results

The results for melody estimation are presented in Table 3.

Collection	HPCP	Direct	Iterative	Joint
MIREX04	71.23%	75.04%	74.76%	74.87%
MIREX05	61.12%	66.64%	66.76%	66.59%
RWC Pop	56.47%	52.66%	52.65%	52.41%

Table 3. Salience function performance

We note that the performance of all algorithms decreases as the collection used becomes more complex and resemblant of real world music collections. A possible explanation for the significantly decreased performance of all approaches for the RWC collection could be that as it was not designed specifically for melody estimation, it contains more songs in which there are several lines competing for salience in the melody range, resulting in more errors when we only consider the maximum of the salience function at each frame. We also observe that for the MIREX collections the HPCP based approach is outperformed by the other algorithms, however for the RWC collection it performs slightly better than the multiple-F0 algorithms.

A two-way analysis of variance (ANOVA) comparing our HPCP based approach with the Direct method is given in table 4.

Source	SS	df	Mean Squares	F-ratio	p-value
Collection	11,971.664	2	5,985.832	41.423	0.000
Algorithm	75.996	1	75.996	0.526	0.469
Collection* Algorithm	705.932	2	352.966	2.443	0.089
Error	29,768.390	206	144.507		

Table 4. ANOVA comparing the HPCP based approach to the Direct method over all collections

The ANOVA reveals that the collection used for evaluation indeed has a significant influence on the results (p-value $< 10^{-3}$). Interestingly, when considering performance over all collections, there is no significant difference between the two approaches (p-value 0.469), indicating that overall our approach has comparable performance to that of the other salience functions and hence potential as a first step in a complete melody estimation system [4].

We next turn to the bass line estimation results. Given that the multiple-F0 salience functions proposed in [9] are not specifically tuned for bass line estimation, only the HPCP based approach was evaluated. We evaluated using the RWC

[4] When comparing the results for each collection separately, only the difference in performance for the RWC collection was found to be statistically significant (p-value 0.016).

collection only as the MIREX collections do not contain bass line annotations, and achieved a score of 73%.

We note that the performance for bass line is significantly higher. We can attribute this to the fact that the bass line is usually the most predominant line in the low frequency range and does not have to compete with other instruments for salience as is the case for the melody.

In Figure 3 we present examples in which the melody and bass line are successfully estimated. The ground truth is represented by o's, and the estimated line by x's. The scores for the estimations presented in Figure 3 are 85%, 80%, 78% and 95% for *daisy1.wav* (MIREX04), *train05.wav* (MIREX05), *RM-P014.wav* (RWC, melody) and *RM-P069.wav* (RWC, bass) respectively.

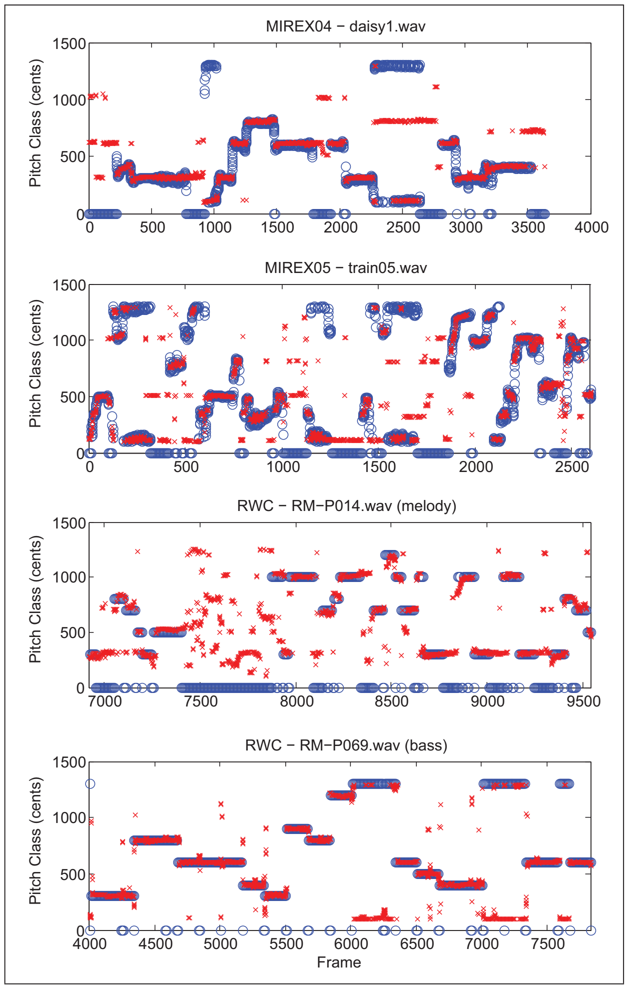


Figure 3. Extracted melody or bass line (x's) against its reference (o's) for each of the collections

In order to evaluate what are the best possible results our approach could potentially achieve, we have calculated estimation performance considering an increasing number of peaks of the salience function and taking the error of the closest peak to the reference frequency (mapped onto one octave) at every frame. This tells us what performance could be achieved if we had a peak selection process which always selected the correct peak as long as it was one of the top n peaks of the salience function. The results are presented in Figure 4.

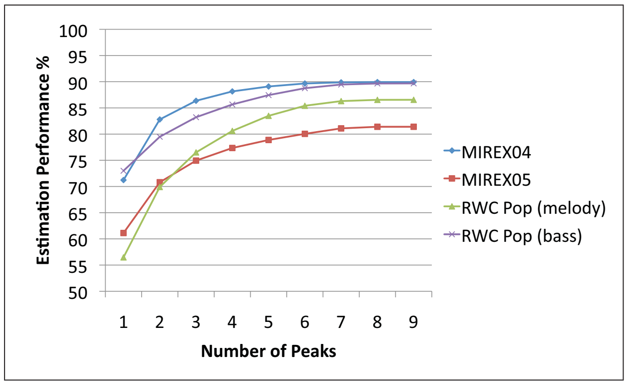


Figure 4. Potential performance vs peak number

The results reveal that our approach has a "glass ceiling" – an inherent limitation which means that there are certain frames in which the melody (or bass line) is not present in any of the peaks of the salience function. The glass ceiling could potentially be "pushed up" by further tuning the pre-processing in the HPCP computation, though we have not explored this in our work.

Nonetheless, we see that performance could be significantly improved if we implemented a good peak selection algorithm even considering just the top two peaks of the salience function. By considering more peaks performance could be improved still, however the task of melody peak tracking is non trivial and we cannot assert how easy it would to get close to these theoretical performance values.

5 CONCLUSION

In this paper we introduced a method for melody and bass line estimation using chroma features. We adapt the Harmonic Pitch Class Profile and use it as a salience function, which would be used as the first stage in a complete melody and bass line estimation system. We showed that as a salience function our approach has comparable performance to that of other state of the art methods, evaluated on real world music collections. Future work will involve the implementation of the further steps required for a complete melody and bass line estimation system, and an evaluation of the extracted representation in the context of similarity based applications.

6 ACKNOWLEDGEMENTS

We would like to thank Anssi Klapuri and Matti Ryynänen for sharing information about the test collections used for the evaluation and for their support; and Joan Serrà for his support and assistance with the HPCP alignment procedure.

7 REFERENCES

[1] P. Cancela. "Tracking Melody in Polyphonic Audio", In *Proc. MIREX*, 2008.

[2] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek, and G. Tzanetakis. "A Comparative Evaluation of Search Techniques for Query-by-Humming Using the MUSART Testbed", *Journal of the American Society for Information Science and Technology*, February 2007.

[3] K. Dressler. "Extraction of the melody pitch contour from polyphonic audio", *Proc. 6th International Conference on Music Information Retrieval*, Sept. 2005.

[4] T. Fujishima. "Realtime Chord Recognition of Musical Sound: a System using Common Lisp Music", *Computer Music Conference (ICMC)*, pages 464–467, 1999.

[5] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2006.

[6] E. Gómez. "Tonal Description of Polyphonic Audio for Music Content Processing", *INFORMS Journal on Computing, Special Cluster on Computation in Music*, 18(3), 2006.

[7] M. Goto, H. Hashiguchi, T. Nishinura, and R. Oka. "Rwc music database: Popular, classical, and jazz music databases", *Proc. Third International Conference on Music Information Retrieval ISMIR-02*, Paris, 2002. IRCAM.

[8] M. Goto. "A real-time music-scene-description system: predominant-f0 estimation for detecting melody and bass lines in real-world audio signals", *Speech Communication*, 43:311–329, 2004.

[9] A. Klapuri. "Multiple fundamental frequency estimation by summing harmonic amplitudes", *Proc. 7th International Conference on Music Information Retrieval*, Victoria, Canada, October 2006.

[10] M. Marolt. "A mid-level representation for melody-based retrieval in audio collections", *Multimedia, IEEE Transactions on*, 10(8):1617–1625, Dec. 2008.

[11] G. E. Poliner, D. P. W. Ellis, F. Ehmann, E. Gómez, S. Steich, and O. Beesuan. "Melody transcription from music audio: Approaches and evaluation", *IEEE Transactions on Audio, Speech and Language Processing*, 15(4):1247–1256, 2007.

[12] M. Ryynänen and A. Klapuri. "Transcription of the singing melody in polyphonic music", *Proc. 7th International Conference on Music Information Retrieval*, Victoria, Canada, Oct. 2006.

[13] M. Ryynänen and A. Klapuri. "Automatic transcription of melody, bass line, and chords in polyphonic music", *Computer Music Journal*, 32(3):72–86, 2008.

[14] X. Serra, R. Bresin, and A. Camurri. "Sound and Music Computing: Challenges and Strategies", *Journal of New Music Research*, 36(3):185–190, 2007.

[15] J. Serrà, E. Gómez, P. Herrera, and X. Serra. "Chroma binary similarity and local alignment applied to cover song identification", *IEEE Transactions on Audio, Speech and Language Processing*, 16:1138–1151, August 2008.

INTERACTIVE INFRASONIC ENVIRONMENT: A New Type of Sound Installation for Controlling Infrasound

Reinhard Gupfinger, Hideaki Ogawa, Christa Sommerer, Laurent Mignonneau
Graduate School of Interface Culture
University of Art and Design Linz
Linz, Austria
{ reinhard.gupfinger, hideaki.ogawa, christa.sommerer, laurent.mignonneau } @ufg.ac.at

ABSTRACT

This paper proposes a new type of interactive sound instrument for use with audiences in sound installations and musical performances. The Interactive Infrasonic Environment allows users to perceive and experiment with the vibration and acoustic energy produced by infrasound.

This article begins with a brief overview of infrasound and examines its generation, human perception, areas of application and some odd myths. Infrasound is sound with a frequency lower than 20 hertz (20 cycles per second) – outside the normal limits of human hearing. Nevertheless the human body can perceive such low frequencies via cross-modal senses.

This paper describes three key aspects of infrasonic sound technologies: the artificial generation of infrasound, the human perception of infrasound, and the interactive environment for sound installations and musical performances.

Additionally we illustrate these ideas with related works from the field of sound art and interactive art.

Keywords

sound installation, interactive environment, infrasound, video tracking system.

1. INTRODUCTION

Our idea for building an interactive system that uses infrasound came from the myth of "Demutspfeife" in which a single tone from an organ brings humility to its listeners. The legend says that these big organ pipes are often used in churches to subdue people. While researching the field of infrasound further myths about the affects of infrasound on humans were found. We focused on the possibility for bringing infrasound to audiences in sound installations and musical performances wherein the users could experiment with infrasound on their own bodies. With this project we attempt to increase acoustic awareness by sensitizing people to very low sound frequencies. This sense is still underdeveloped in our culture.

A main problem was just how to generate infrasound. There are only a few possibilities and most of them would be unsuitable for an audience in a sound installation. Due to the fact that sound installations tend to be difficult to perceive and understand for the audience we wanted to create a simple interactive environment that offers a great deal of creative freedom and options for the users.

We could not find many artistic projects relating to the field of infrasound but there have been a number of studies from several areas of study such as medicine, weaponry and noise reduction. There are too many facets to this research and far too much speculation on infrasound to fit within the scope of this paper.

2. INFRASOUND

Infrasound is sound that is lower than 20 cycles per second. This is sound that is just below the lower limit of the human sense of hearing.

Sound range	Frequency	Wave length
Infrasound	1 Hz < f < 20 Hz	1 Hz = 1125 ft = 342,9 m
Hearable sound	20 Hz < f < 16 kHz	20 Hz = 56 ft = 17 m

SMC 2009, July 23-25, Porto, Portugal

2.1. The Generation of Infrasound

Natural sources:	Artificial sources:
- Wind and atmospheric turbulence - Earthquakes and volcanic eruption - Waterfalls and breaking waves - Animals (e.g. whales, elephants, rhinoceros, giraffes, okapi and alligators)	- Air conditioning systems - Wind energy turbines - Gas turbine power stations - Industrial facilities (e.g. compressors, compactors) - Buildings (e.g. skyscrapers, tunnels, bridges) - Vehicles(cars and trucks, trains, ships, planes) - Explosions -Speaker systems

Table 1. Sources of infrasound

2.2. The Human Perception of Infrasound

Hearing does not abruptly stop below 20 Hz. As careful measurements have shown, with high enough sound pressure the ear can register infrasound down to about 1 Hz. [1] Infrasound perceived as a mixture of auditory and tactile sensation at a high threshold level.

Sense	Perception
- Ear	- Feeling of pressure
- Skin	- Pulsation and vibration
- Viscera	- Resonance vibration
- Sinuses, nares, chest, bowel	- Barometric variation
- Eye	- Vibration

Table 2. The human perception with cross-modal senses

Infrasound especially affects the cavities of the human body though its affect on air pressure. Different pitches and intensities of infrasound can be perceived as changes in pressure and vibration. The effects of very low frequency noises such as infrasound on human beings have been documented in many articles; these include: temporary hearing threshold shifts, changes in blood pressure, changes in heart rate, changes in respiratory rate, balance disturbances, cognition disturbances. [2]

During medical research at the Hellersen Hospital in Lüdenscheid (Germany) the psychosomatic effects of infrasound were tested on people with chronic pain. Six inpatients suffering with chronic pain were exposed to infrasound at 9 Hz for 20 min per day. After one week they concluded that infrasound activates the autonomic nervous system and has positive effects on stress and also has a palliative effect on pain. [3] If the sound pressure of infrasound is higher than 120 dB, negative effects of infrasound appear: headache, breathing problems, changes in heart rate and general stress. Constant pressure (more than 10 minutes) with infrasound at more than 170 dB causes the death of the test animals. [4] Additionally there is an additional impressive effect, which is produced by infrasound. A NASA technical report mentions a resonant frequency for the eyes of 18 Hz (NASA Technical Report 19770013810). In this case the eyeball would begin vibrating which would cause a notable “smearing” of vision. [5] Vic Tany outlines the idea that a standing wave of 19Hz could, under certain conditions, generate sensory phenomena suggestive of a ghost in his paper “The Ghost in the Machine” from 1998.

2.3. The Areas of Application for Infrasound

Infrasound is currently being utilized in various fields. A relatively new discipline is the medical use of infrasound therapy. It is useful in treating chronic pain and arteriosclerosis wherein vibrating medical devices are attached to the body.

Interest in atmospheric infrasound peaked during the Cold War, as it is one of several ways to detect, locate, and classify nuclear explosions at very great distances. At present the Comprehensive Test Ban Treaty requires a more sophisticated global sensor network to monitor compliance. [6]

A global network of infrasonic detectors has been installed to observe the atmosphere. The intention is the early detection of meteorites, tornados, earthquakes and volcanic eruptions. In the 1970s, the National Oceanic and Atmospheric Administration began a study of atmospheric infrasound to determine whether it could be used to improve warning capabilities for severe weather events such as tornadoes. [7] They found that many thunderstorms radiated infrasound, which could be detected by observatories more than thousand miles away.

2.4. Infrasound Myths

The most bizarre myth about infrasound is the *brown note*. As the name implies, it is assumed that this is an infrasound frequency, which causes humans to lose control of their bowels. There is no scientific evidence that such an infrasound note exists.

Another legend is about infrasound weapons. Some infrasound review articles mention the fact that several countries have investigated this possibility. Such an infrasound weapon would be a huge installation that could generate high-pressure low frequencies, which would cause anxiety, internal injuries and death to humans.

Finally there is the myth of "Demutspfeife" that was mentioned earlier. It is a single organ pipe and part of a church organ, which brings humility to its listeners. Our Interactive Infrasonic Environment project strongly references this idea, but we do not intend to bring any negative effects of infrasound. We propose a strictly positive application of infrasound and low sound frequencies.

3. INTERACTIVE INFRASONIC ENVIRONMENT

As there are so many fascinating aspects and also a few strange myths concerning infrasound, we started to develop an infrasonic installation in early 2007.

The goal of the project was to make infrasound approachable for everyone. We attempted to build an installation where the audience can experiment with the perception of infrasound regardless whether it creates a positive or negative effect in their bodies. For this reason we soon realized that an important part of the installation would be a multiple-user interface through which all users can interact simultaneously and in real time.

The Interactive Infrasonic Environment is the first interactive instrument that allows users to generate infrasound while moving around the space. It is an installation that overlaps auditory and tactile stimuli to increase the level of acoustic awareness.

3.1. The Organ Pipe

The installation hardware is based on a 19 ft wooden organ pipe placed in the center of the environment. The pitch of the pipe can be tuned with an adjuster at the end of the organ pipe. The wavelength of the pipe is modified to the characters and sizes of the room in which the installation is located. Adjustments are needed to get satisfying resonates from the specific architecture of the room. The organ pipe can generate sound frequencies down to 15 Hz, going beyond the limits of human hearing.

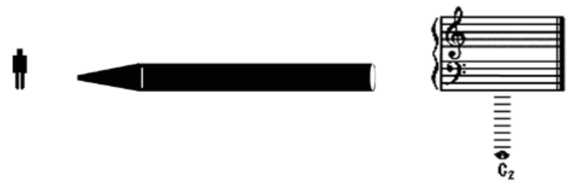


Figure 1. Dimensions and tone pitch of the organ pipe

3.2. Sound Generation

The sound of the pipe is produced via the vibration of air in the same way as a flute. The airflow is driven over an open aperture and against a sharp lip called a labium. The airflow begins fluttering and creates high and low pressure waves within the pipe's air column. The low sound wave generated has a frequency between 15 Hz and 17 Hz. As described earlier the tuning of the organ pipe depends on the size of the room. The vibration energies of infrasound are transmitted to the human body and the architectural space enhances the natural resonance of both. The source of a sound in an interactive computer based instrument is not some abstract or concrete concept, or even the algorithm(s) that have been written; it is the gesture of the performer, the excitation moment - it is fundamentally about that nature of excitation. [8]

3.3. Video Tracking System

Interaction with an instrument that uses video tracking is a particular case in point, for the nature of engagement is abstract, and as such is based not so much on the physical relationship of the self to the physical space that houses the instrument or interactive installation. [9] The project uses a video camera situated high on a wall, which continuously observes the surroundings of the organ pipe, tracking the position and movement of the users. The software is programmed in Max/MSP and uses the Cyclops object to receive and analyze video input. The program rasterizes the video input and analyses the grayscales of predetermined zones. The users interact by moving around and changing their positions. These actions directly control the wind machine, which is fluently controlling the airflow and thus the volume and pitch of the organ pipe.

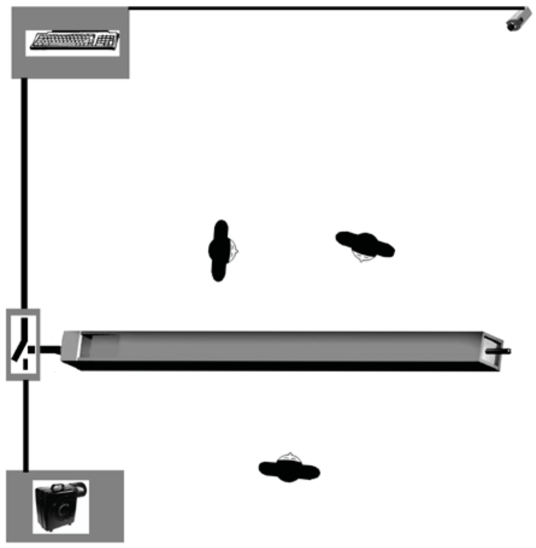


Figure 2. Schematic view of the Interactive Infrasonic Environment

3.4. Experiments with Test Subjects

During the first testing phase we explored the sensations in the Interactive Infrasonic Environment with the help of ten participants. We performed separate individual experiments by using an exposure chamber. A single experiment consisted of three 5 min exposure periods and after each period a 15 min post exposure period, including a short interview of the test subject. The test was performed with a continuous tone at a frequency of 15 Hz and three different sound intensities: low, middle and high. The goal of these experiments was to establish the connection between the Interactive Infrasonic Environment and psychoacoustic human perception. The experiments showed that the low sound frequencies produced feelings of pressure, pulsation and vibration on cross-modal senses for all test persons. It was surprising that the threshold where the test persons started to feel uncomfortable varied from person to person. For two subjects the feeling of discomfort started at the second level of intensity and for five subjects it started at the highest level. For the other subjects (3) there were no uncomfortable feelings experienced during the entire test period. Physical contact with the vibrating organ pipe was enjoyable for all test participants.

3.5. Present and Future Work

To date the Interactive Infrasonic Environment installation has been shown at several exhibitions. The feedback from the audience and the curators confirmed our intention to continue with and further expand this infrasonic project.

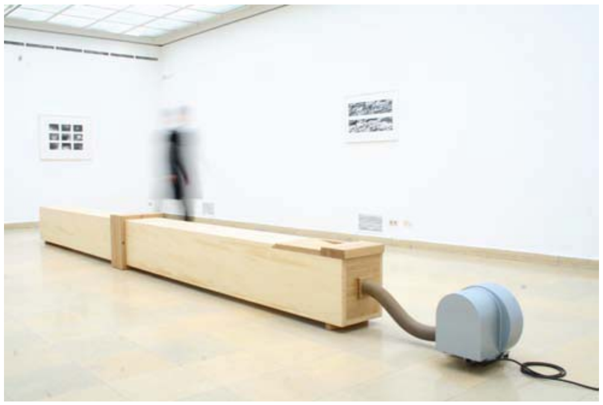


Figure 3. Interactive Infrasonic Environment, Sound Characters exhibition, Innsbruck, 2009

We are currently working on and researching a musical performance using the Interactive Infrasonic Environment. We use infrasound to conduct a choir and likewise the members of the choir can control the infrasound through the installation. The first live performance was staged in May 2009 in Linz (Austria), the European Cultural Capital at the time. The members of a women's ensemble improvised to the accompaniment of the Interactive Infrasonic Environment. With certain sequences of movements, members of the choir could steer the tones produced by the organ pipe. The tones generated by the organ pipe in turn provided the impetus for tonal variations in the choir's singing.

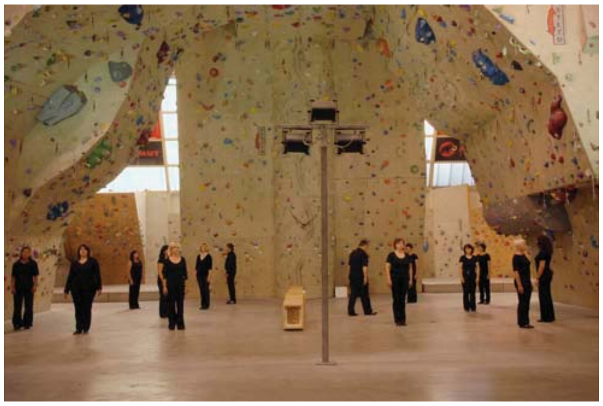


Figure 4. Improvisation Concert for Choir and Organ Pipe, Linz, 2009

4. RELATED WORKS

There have been a several studies and projects concerning infrasound in the field of media art. We want to highlight a few of the projects that inspired our vision of an interactive infrasonic installation. The last example refers to our video tracking system, which is a basis to a certain extent on the milestone of the interactive sound installation.

4.1. IIT

The Infrasonic Transmission Tube was designed and constructed by Laton, a research lab and record label based in Vienna. The prototype sound system was able to generate infrasound frequencies from 1 Hz to 20 Hz. They used the Infrasonic Transmission Tube for their realization of their self-titled "Infrasonic Music". The project was shown at the Ars Electronica Festival in 1996.

4.2. Live Room

Mark Bain developed the Live Room in 1998. It was a project that used small acoustic-intensifying machines, which were attached directly to the structure of a room. The installation incorporated the architecture by running impulsive energy throughout, creating sound and vibration in direct relation to the building and the dimensions of the space. With this work, I was interested in TRANSDUCING ARCHITECTURE, driving the space with external influences of a vibro-kinetic nature. [10] Bain was able to effectively tune a space by delivering the resonant frequency to its different parts. The Live Room also generated infrasonic sound, which brought strangeness to this project related to the production and injection of these unique low frequencies.

4.3. Very Nervous System

In 1982 the Canadian Artist David Rokeby started to develop his Very Nervous System, an interactive sound environment with a real time motion tracking system. Video cameras observed the users action and a computer analyzed the data and responds acoustically to the input. It was the intention to design a space in which the movements of one's body create sound. David Rokeby was interested in creating a complex relationship between the user's body and the system. "Because the computer is purely logical, the language of interaction should strive to be intuitive. Because the computer removes you from your body, the body should be strongly engaged. Because the computer's activity takes place on the tiny playing fields of integrated circuits, the encounter with the computer should take place in human-scaled physical space. Because the computer is objective and disinterested, the experience should be intimate." [11] In the early days of interactive art, the interaction with the Very Nervous System was very novel because the interface was invisible. The system was used in performances, exhibitions and additionally in music therapy applications.

5. CONCLUSION

In summary, this paper provides a brief overview of infrasound, its generation, perception, areas of application and myths. We noted that there is a need to sensitize people to allow them to better register infrasound and that our project intends to increase acoustic awareness; this is still an underdeveloped sense in our culture.

This paper presented a new type of interactive sound instrument that allows users to experiment with the vibration and acoustic energies produced by infrasound.

The challenge for this project was to construct a sound generator, which produces perceptible infrasound. As a final remark we would like to say that there is still much work to do in order to fully implement all of our ideas, especially those involving live musical performances.

6. REFERENCES

[1] Altmann, J. "Acoustic Weapons—A Prospective Assessment: Sources, Propagation, and Effects of Strong Sound" *Cornell University Peace Studies Program*, Dortmund, 2008.

[2] Takahashi Y. "An Infrasound Experiment for Industrial Hygiene" *Industrial Health 32*, p. 480, 1997

[3] Deutschmann-Hütt, H. "Psychosomatische Wirkung von Infraschall am Beispiel chronischen Schmerzes" *Sportkrankenhaus Hellersen*, Lüdenscheid 2005.

[4] El-Nounou, M. "Messung und Bewertung von niederfrequenten Luftdruckschwankungen und Infraschall in Personenkraftwagen bei unterschiedlichen Fahrbedingungen" *Ludwig-Maximilans-Universität,* München, 2006.

[5] Tany, V. “The Ghost in the Machine” *Journal of the Society for Psychical Research Vol. 62*, 1998

[6] National Research Council, “Research Required to Support Comprehensive Nuclear Test Ban Treaty Monitoring” *National Academy Press*, Washington, DC 1997.

[7] Georges, T. M. “*Instruments and Techniques for Thunderstorm Observation and Analysis*”, E. Kessler, ed., U. Oklahoma P., Norman, Okla. 1988.

[8] Traube, C., Depalle, P., Wanderley, M. “Indirect Acquisition of Instrument Gesture Based on Signal, Physical and Perceptual Information”, *NIME* 2003

[9] Paine, G. “Gesture and Musical Interaction: Interactive Engagement Through Dynamic Morphologie” *NIME* 2004

[10] Bain, M. “The Live Room: Transducing Resonant Architectures” *Cambridge University Press,* New York. 2003

[11] Rokeby,D. http://homepage.mac.com/davidrokeby/vns.html

VOCALISTENER: A SINGING-TO-SINGING SYNTHESIS SYSTEM BASED ON ITERATIVE PARAMETER ESTIMATION

Tomoyasu Nakano **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{t.nakano, m.goto} [at] aist.go.jp

ABSTRACT

This paper presents a singing synthesis system, *VocaListener*, that automatically estimates parameters for singing synthesis from a user's singing voice with the help of song lyrics. Although there is a method to estimate singing synthesis parameters of pitch (F_0) and dynamics (power) from a singing voice, it does not adapt to different singing synthesis conditions (*e.g.*, different singing synthesis systems and their singer databases) or singing skill/style modifications. To deal with different conditions, VocaListener repeatedly updates singing synthesis parameters so that the synthesized singing can more closely mimic the user's singing. Moreover, VocaListener has functions to help modify the user's singing by correcting off-pitch phrases or changing vibrato. In an experimental evaluation under two different singing synthesis conditions, our system achieved synthesized singing that closely mimicked the user's singing.

1 INTRODUCTION

Many end users have started to use commercial singing synthesis systems to produce music and the number of listeners who enjoy synthesized singing is increasing. In fact, over one hundred thousand copies of popular software packages based on Vocaloid2 [1] have been sold and various compact discs that include synthesized vocal tracks have appeared on popular music charts in Japan. Singing synthesis systems are used not only for creating original vocal tracks, but also for enjoying collaborative creations and communications via content-sharing services on the Web [2, 3]. In light of the growing importance of singing synthesis, the aim of this study is to develop a system that helps a user synthesize natural and expressive singing voices more easily and efficiently. Moreover, by synthesizing high-quality human-like singing voices, we aim at discovering the mechanism of human singing voice production and perception.

Much work has been done on singing synthesis. The most popular approach for singing synthesis is *lyrics-to-singing (text-to-singing) synthesis* where a user provides note-level score information of the melody with its lyrics to synthesize a singing voice [1, 4, 5]. To improve naturalness and provide original expressions, some systems [1] enable a user to adjust singing synthesis parameters such as pitch (F_0) and dynamics (power). The manual parameter adjustment, however, is not easy and requires considerable time and effort. Another approach is *speech-to-singing synthesis* where a speaking voice reading the lyrics of a song is converted into a singing voice by controlling acoustic features [6]. This approach is interesting because a user can synthesize singing voices having the user's voice timbre, but various voice timbres cannot be used.

SMC 2009, July 23-25, Porto, Portugal

In this paper, we propose a new system named *VocaListener* that can estimate singing synthesis parameters (pitch and dynamics) by mimicking a user's singing voice. Since a natural voice is provided by the user, the synthesized singing voice mimicking it can be human-like and natural without time-consuming manual adjustment. We named this approach *singing-to-singing synthesis*. Janer *et al.* [7] tried a similar approach and succeeded to some extent. Their method analyzes acoustic feature values of the input user's singing and directly converts those values into the synthesis parameters. But their method is not robust with respect to different singing synthesis conditions. For example, even if we specify the same parameters, the synthesized results always differ when we change to another singing synthesis system or a different system's singer database because of the results' nonlinearity. The ability to mimic a user's singing is therefore limited.

To overcome such limitations on robustness, VocaListener iteratively estimates singing synthesis parameters so that after a certain number of iterations the synthesized singing can become more similar to the user's singing in terms of pitch and dynamics. In short, VocaListener can synthesize a singing voice while listening to its own generated voice through an original feedback-loop mechanism. Figure 1 shows examples of synthesized voices under two different conditions (different singer databases). With the previous approach [7], there were differences in pitch (F_0) and dynamics (power). On the other hand, such differences are minimal with VocaListener.

Moreover, VocaListener supports a highly-accurate lyrics-to-singing synchronization function. Given the user's singing and the corresponding lyrics without any score information, VocaListener synchronizes them automatically to determine each musical note that corresponds to a phoneme of the lyrics. We therefore developed an originally-adapted/trained acoustic model for singing syn-

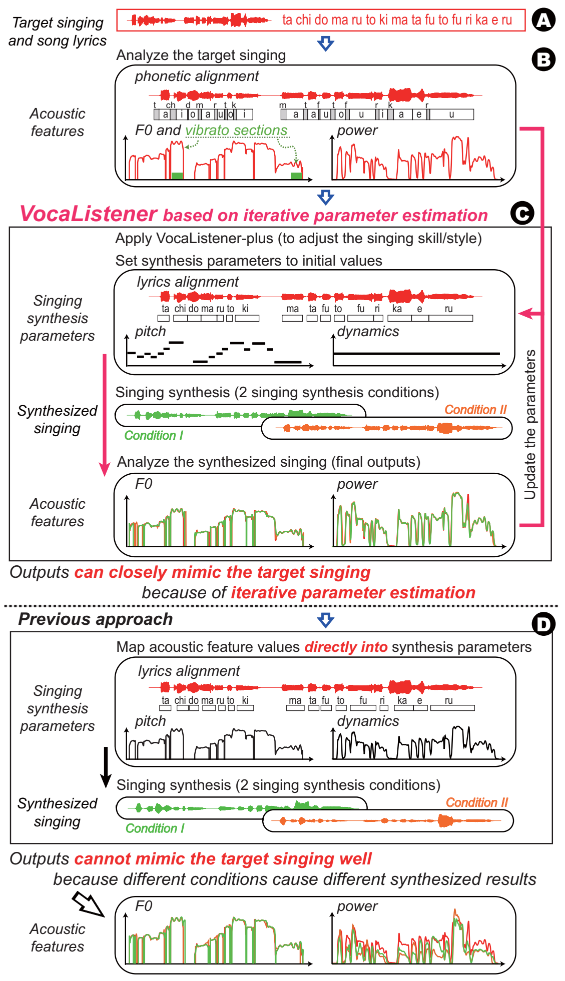

Target singing and song lyrics
ta chi do ma ru to ki ma ta fu to fu ri ka e ru
A
Analyze the target singing
B
phonetic alignment
Acoustic features
F0 and vibrato sections
power
VocaListener based on iterative parameter estimation
C
Apply VocaListener-plus (to adjust the singing skill/style)
Set synthesis parameters to initial values
lyrics alignment
Singing synthesis parameters
pitch
dynamics
Singing synthesis (2 singing synthesis conditions)
Synthesized singing
Condition I
Condition II
Analyze the synthesized singing (final outputs)
F0
power
Update the parameters
Outputs can closely mimic the target singing because of iterative parameter estimation
Previous approach
D
Map acoustic feature values directly into synthesis parameters
Outputs cannot mimic the target singing well because different conditions cause different synthesized results

Figure 1. Overview of *VocaListener* and problems of a previous approach by Janer *et al.* [7].

chronization. Although synchronization errors with this model are rare, we also provide an interface that lets a user easily correct such errors just by pointing them out. In addition, VocaListener also supports a function to improve synthesized singing as if the user's singing skill were improved.

2 PARAMETER ESTIMATION SYSTEM FOR SINGING SYNTHESIS: VOCALISTENER

VocaListener consists of three components, the *VocaListener-front-end* for singing analysis and synthesis, the *VocaListener-core* to estimate the parameters for singing synthesis, and the *VocaListener-plus* to adjust the singing skill/style of the synthesized singing.

Figure 1 shows an overview of the VocaListener system. The user's singing voice (i.e., *target singing*) and the lyrics[1] are taken as the system input (Ⓐ). Using this input, the system automatically synchronizes the lyrics with the target singing to generate note-level score information, estimates the fundamental frequency (F_0) and the power of the target singing, and detects vibrato sections that are used just for the *VocaListener-plus* (Ⓑ). Errors in the lyrics synchronization can be manually corrected through simple interaction. The system then iteratively estimates the parameters through the VocaListener-core, and synthesizes the singing voice (Ⓒ). The user can also adjust the singing skill/style (*e.g.*, vibrato extent and F_0 contour) through the VocaListener-plus.

[1] In our current implementation, Japanese lyrics spelled in a mixture of Japanese phonetic characters and Chinese characters are mainly supported. English lyrics can also be easily supported because the underlying ideas of VocaListener are universal and language-independent.

2.1 VocaListener-front-end: analysis and synthesis

The VocaListener-front-end consists of singing analysis and singing synthesis. Throughout this paper, singing samples are monaural recordings of solo vocal digitized at 16 bit / 44.1 kHz.

2.1.1 singing analysis

The system estimates the fundamental frequency (F_0), the power, and the onset time and duration of each musical note. Since the analysis frame is shifted by 441 samples (10 ms), the discrete time step (1 *frame-time*) is 10 ms. This paper uses time t for the time measured in frame-time units.

In VocaListener, these features are estimated as follows:

Fundamental frequency: $F_0(t)$ is estimated using SWIPE [8]. Hereafter, unless otherwise stated, $F_0(t)$ are log-scale frequency values (real numbers) in relation to the MIDI note number (a semitone is 1, and middle C corresponds to 60).

Power: $Pow(t)$ is estimated by applying a Hanning window whose length is 2048 samples (about 46 ms).

Onset time and duration: To estimate the onset time and duration of each musical note, the system synchronizes the phoneme-level pronunciation of the lyrics with the target singing. This synchronization is called *phonetic alignment* and is estimated through Viterbi alignment with a phoneme-level hidden Markov model (monophone HMM). The pronunciation is estimated by using a Japanese language morphological analyzer [9].

2.1.2 singing synthesis

In our current implementation, the system estimates parameters for commercial singing synthesis software based on Yamaha's Vocaloid or Vocaloid2 technology [1]. For example, we use software named Hatsune Miku (referred to as CV01) and Kagamine Rin (referred to as CV02) [10] for synthesizing Japanese female singing. Since all parameters are estimated every 10 ms, they are linearly interpolated at every 1 ms to improve the synthesized quality, and are fed via a VSTi plug-in (Vocaloid Playback VST Instrument).

2.2 VocaListener-plus: adjusting singing skill/style

To extend the flexibility, the VocaListener-plus provides functions, *pitch change* and *style modification*, which can modify the value of the estimated acoustic features of the target singing. The user can select whether to use these functions based on personal preference. Figure 2 shows an example of using these functions.

2.2.1 *Pitch change*

We propose *pitch transposition* and *off-pitch correction* to overcome the limitations of the user's singing skill and pitch range. The pitch transposition function changes the target $F_0(t)$ just by adding an offset value for transposition during the whole section or a partial section.

The off-pitch correction function automatically corrects off-pitch phrases by adjusting the target $F_0(t)$ according to an offset of F_d ($0 \leq F_d < 1$) estimated for each voiced section. The off-pitch amount F_d is estimated by fitting a semitone-width grid to $F_0(t)$. The grid is defined as a comb-filter-like function where Gaussian distributions are aligned at one semitone intervals. Just for this fitting, $F_0(t)$ is temporarily smoothed by using an FIR lowpass filter with a 3-Hz cutoff frequency[2] to suppress F_0 fluctuations (overshoot, vibrato, preparation, and fine fluctuation) of the singing voice [11, 12]. Last, the most fitted offset F_d is used to adjust $F_0(t)$ to its nearest correct pitch.

2.2.2 *Style modification*

In this paper, *vibrato adjustment* and *singing smoothing* are proposed to emphasize or suppress the F_0 fluctuations. Since the F_0 fluctuations are important factors to characterize human singing [11, 12], a user can change the impression of singing. The $F_0(t)$ and $Pow(t)$ of the target singing are adjusted by interpolating or extrapolating between the original values ($F_0(t)$ and $Pow(t)$) and their smoothed values obtained by using an FIR lowpass filter. A user can separately adjust vibrato sections and other sections. The vibrato sections are detected by using the vibrato detection method [13].

2.3 VocaListener-core: estimating the parameters

Figure 3 shows the estimation process for *VocaListener-core*. After acoustic features of the target singing (modified by VocaListener-plus, if necessary) are estimated, these features are converted into synthesis parameters that are then fed to the singing synthesis software. The synthesized singing is then analyzed and compared with the target singing. Until the synthesized singing is sufficiently close to the target singing, the system repeats the parameter update and its synthesis.

2.3.1 *Parameters for singing synthesis*

The system estimates parameters for *pitch*, *dynamics*, and *lyrics alignment* (Table 1). The pitch parameters consist of *MIDI note number (Note#)*[3], *pitch bend (PIT)*, and *pitch bend sensitivity (PBS)*, and the dynamics parameter is *dynamics (DYN)*. For the pitch (F_0), the fractional portion (PIT) is separated from the integer portion (Note#). PIT represents a relative decimal deviation from the corresponding integer note number (Note#), and PBS specifies the range (magnitude) of its deviation. The results of the lyrics alignment are represented by the note onset (onset time) and its duration.

These MIDI-based parameters can be considered typical and common, not specific to the Vocaloid software. A set of these parameters, PIT, PBS, and DYN, are iteratively estimated after being initialized to 0, 1, and 64, respectively.

[2] We avoid unnatural smoothing by ignoring silent sections and leaps of F_0 transitions wider than a 1.8-semitone threshold.

[3] For synthesis, each mora of Japanese pronunciation is mapped into a musical note, where the mora representation can be classified into three types: "V", "CV", and "N". "V" denotes vowel (*a, i, ...*), "C" denotes consonant (*t, ch, ...*), and "N" denotes syllabic nasal (*n*).

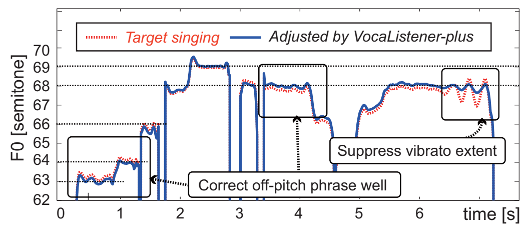
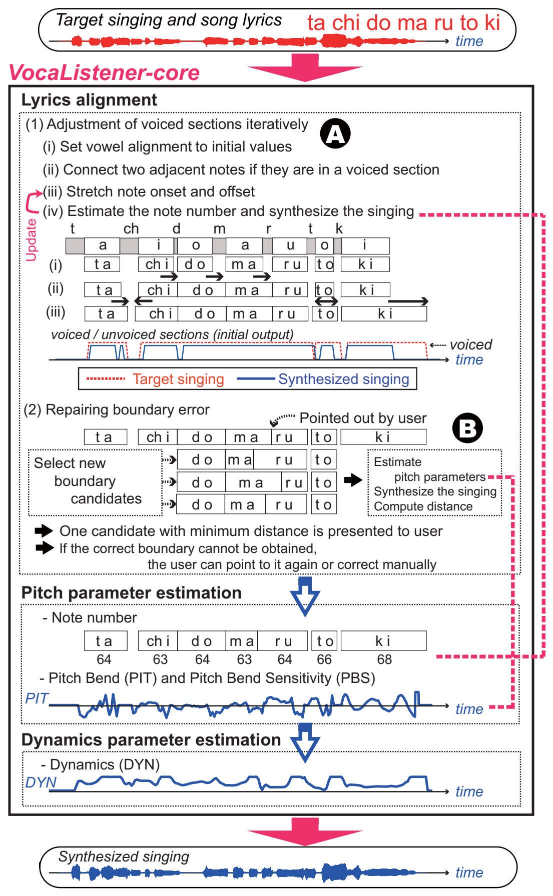


Figure 2. Example of $F_0(t)$ adjusted by VocaListener-plus.



Target singing and song lyrics
ta chi do ma ru to ki
time
VocaListener-core
Lyrics alignment
(1) Adjustment of voiced sections iteratively
A
(i) Set vowel alignment to initial values
(ii) Connect two adjacent notes if they are in a voiced section
(iii) Stretch note onset and offset
(iv) Estimate the note number and synthesize the singing
Update
voiced / unvoiced sections (initial output)
voiced
Target singing
Synthesized singing
(2) Repairing boundary error
Pointed out by user
B
Select new boundary candidates
Estimate pitch parameters
Synthesize the singing
Compute distance
One candidate with minimum distance is presented to user
If the correct boundary cannot be obtained, the user can point to it again or correct manually
Pitch parameter estimation
- Note number
- Pitch Bend (PIT) and Pitch Bend Sensitivity (PBS)
PIT
Dynamics parameter estimation
- Dynamics (DYN)
DYN
Synthesized singing

Figure 3. Overview of the parameter estimation procedure, *VocaListener-core*.

Table 1. Relation between singing synthesis parameters and acoustic features.

Acoustic features	Synthesis parameters	
F_0	Pitch	Note#, PIT, and PBS
Power	Dynamics	DYN
Phonetic alignment	Lyrics alignment	Note onset Note duration

2.3.2 *Lyrics alignment estimation with error repairing*

Even if the same note onset and its duration (lyrics alignment) are given to different singing synthesis systems (such as Vocaloid and Vocaloid2) or different singer databases (such as CV01 and CV02), the note onset and note duration often differ in the synthesized singing because of their nonlinearity (caused by their internal waveform concatenation mechanism). We therefore have to adjust (update) the lyrics alignment iteratively so that each voiced section of the synthesized singing can be the same as the original voiced section of the target singing. As shown in Figure 3Ⓐ, the last two steps (iii) and (iv) in the following four steps are repeated:

Step (i) Given the phonetic alignment of the automatic synchronization, the note onset and duration are initialized by using its vowel.

Step (ii) If two adjacent notes are not connected but their sections are judged to be a single voiced section, the duration of the former note is extended to the onset of the latter note so that they can be connected. This eliminates a small gap and improves the naturalness of the synthesized singing.

Step (iii) By comparing voiced sections of the target and synthesized singing, the note onset and duration are adjusted so that they become closer to those of the target.

Step (iv) Given the new alignment, the note number (Note#) is estimated again and the singing is synthesized.

Although the automatic synchronization of song lyrics with the target singing is accurate in general, there are sometimes a few boundary errors that degrade the synthesized quality. We therefore propose an interface that lets a user correct each error just by pointing it out without manually adjusting (specifying) the boundary. As shown in Figure 3Ⓑ, other boundary candidates are shown on a screen so that the user can simply choose the correct one by listening to each one. Even if it is difficult for a user to specify the correct boundary from scratch, it is easy to choose the correct candidate interactively. To generate candidates, the system computes timbre fluctuation values of the target singing by using ΔMFCCs, and several candidates with high fluctuation values are selected. The system then synthesizes each candidate and compares it with the target singing by using MFCCs. The candidates are sorted and presented to the user in the order of similarity to the target singing. If none of the candidates are correct, the user can correct manually at the frame level.

2.3.3 *Pitch parameter estimation*

Given the results of lyrics alignment, the pitch parameters are iteratively estimated so that the synthesized F_0 can become closer to the target F_0. After the note number of each

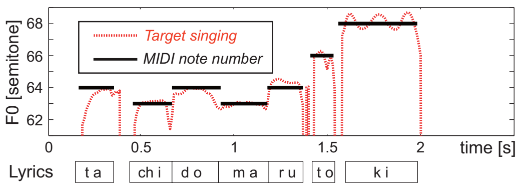


Figure 4. F_0 of the target singing and estimated note numbers.

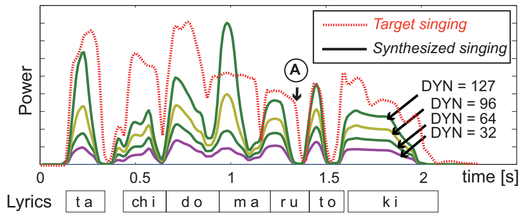


Figure 5. Power of the target singing and power of the singing synthesized with four different dynamics.

note is estimated, PIT and PBS are repeatedly updated by minimizing a distance between the target F_0 and the synthesized F_0.

The note number $Note\#$ for each note is estimated by

$$\text{Note\#} = \underset{n}{\text{argmax}} \left(\sum_t \exp\left\{ -\frac{(n - F_0(t))^2}{2\sigma^2} \right\} \right), \quad (1)$$

where n denotes a note number candidate, is set to 0.33, and t is 0 at the note onset and continues for its duration. Figure 4 shows an example of F_0 and its estimated note numbers.

The PIT and PBS are then estimated by repeating the following steps, where i is the number of updates (iterations), $F0_{\text{org}}(t)$ denotes F_0 of the target singing, and PIT and PBS are represented by $PIT^{(i)}(t)$ and $PBS^{(i)}(t)$:

Step 1) Obtain synthesized singing from the current parameters.

Step 2) Estimate $F0_{\text{syn}}^{(i)}(t)$ that denotes F_0 of the synthesized singing.

Step 3) Update $Pb^{(i)}(t)$ by

$$Pb^{(i+1)}(t) = Pb^{(i)}(t) + \left(F0_{\text{org}}(t) - F0_{\text{syn}}^{(i)}(t) \right), \quad (2)$$

where $Pb^{(i)}(t)$ is a log-scale frequency computed from $PIT^{(i)}(t)$ and $PBS^{(i)}(t)$.

Step 4) Obtain the updated $PIT^{(i+1)}(t)$ and $PBS^{(i+1)}(t)$ from $Pb^{(n+1)}(t)$ after minimizing $PBS^{(i+1)}(t)$. Since a smaller PBS gives better resolution of the synthesized F_0, PBS should be minimized at every iteration as long as PIT can represent the correct relative deviation.

2.3.4 *Dynamics parameter estimation*

Given the results of lyrics alignment and the pitch parameters, the dynamics parameter is iteratively estimated so that the synthesized power can be closer to the target power. Figure 5 shows the power of the target singing before normalization and the power of the singing synthesized with four different dynamics. Since the power of the target singing depends on recording conditions, it is important to mimic the relative power after normalization that is determined so

Table 2. Dataset for experiments A and B and synthesis conditions. All of the song samples were sung by female singers.

Exp. No.	Song No.	Excerpted section	Length [s]	Synthesis conditions
A	No.07	intro–verse–chorus	103	CV01
A	No.16	intro–verse–chorus	100	CV02
B	No.07	verse A	6.0	CV01, CV02
B	No.16	verse A	7.0	CV01, CV02
B	No.54	verse A	8.9	CV01, CV02
B	No.55	verse A	6.5	CV01, CV02

that the normalized target power can be covered by the synthesized power with DYN = 127 (maximum value). However, because there are cases where the target power exceeds the limit of synthesis capability (*e.g.*, Fig.5Ⓐ), the synthesized power cannot perfectly mimic the target. As a compromise, the normalization factor α is determined by minimizing an error defined as a square error between $\alpha Pow_{\mathrm{org}}(t)$ and $Pow_{\mathrm{syn}}^{\mathrm{DYN}=64}(t)$, where $Pow_{\mathrm{syn}}^{\mathrm{DYN}=64}(t)$ denotes the synthesized power with DYN = 64.

The DYN is then estimated by repeating the following steps, where $Pow_{\mathrm{org}}(t)$ denotes the power of the target singing:

Step 1) Obtain synthesized singing from the current parameters.

Step 2) Estimate $Pow_{\mathrm{syn}}^{(i)}(t)$ that denotes the power of the synthesized singing.

Step 3) Update $Db^{(i)}(t)$ by

$$Db^{(i+1)}(t) = Db^{(i)}(t) + \left(\alpha Pow_{\mathrm{org}}(t) - Pow_{\mathrm{syn}}^{(i)}(t)\right), \quad (3)$$

where $Db^{(i)}(t)$ is the actual power given by the current DYN.

Step 4) Obtain the updated DYN from $Db^{(i+1)}(t)$ by using the relationship between the DYN and the actual power values. Before these iteration steps, this relationship should be investigated once by synthesizing the current singing with five DYN values ($= 0, 32, 64, 96, 127$). The relationship for each of the other DYN values is linearly interpolated.

3 EXPERIMENTAL EVALUATIONS

The VocaListener was tested in two experiments. Experiment A evaluated the number of times manual corrections had to be made, and experiment B evaluated the performance of the iterative estimation under different conditions.

In these experiments, two singer databases, CV01 and CV02, were used with the default software settings except for the note-level properties of "No Vibrato" and "0% Bend Depth". Unaccompanied song samples (solo vocal) were taken from the RWC Music Database (Popular Music [14]), and were used as the target singing as shown in Table 2.

For the automatic synchronization of the song lyrics in experiment A, a speaker-independent HMM provided by CSRC [15] for speech recognition was used as the basic acoustic model for MFCCs, ΔMFCCs, and Δpower. The HMM was adapted with singing voice samples by applying MLLR-MAP [16]. As in cross validation where one song sample is evaluated as the test data and the other samples are used as the training data, we excluded the same singer from the HMM adaptation data.

3.1 Experiment A: interactive error repairing for lyrics alignment

To evaluate the lyrics alignment, experiment A used two female songs that were over 100 s in length. Table 3 shows the number of boundary errors that had to be repaired (pointed out) and the number of repairs needed to correct those errors[4]. For example, among 128 musical notes for song No.16, there were only three boundary errors that should be manually pointed out on our interface, and two of these were pointed out twice. In other words, one error was corrected by choosing the first candidate, and the other two errors were corrected by choosing the second candidate. In our experience with many songs, errors tend to occur around /w/ or /r/ (semivowel, liquid) and /m/ or /n/ (nasal sound).

3.2 Experiment B: iterative estimation experiment

Experiment B used four song excerpts sung by four female singers. As shown in Table 2, each song was tested with two conditions — i.e., two singer databases, CV01 and CV02. Since the experiment focused on the performance of the iterative estimation for the pitch and dynamics, we used the hand-labeled lyrics alignment here. The results were evaluated by the *mean error value* defined by

$$\mathrm{err}_{\mathrm{f0}}^{(i)} = \frac{1}{T_f} \sum_t \left| F0_{\mathrm{org}}(t) - F0_{\mathrm{syn}}^{(i)}(t) \right|, \quad (4)$$

$$\mathrm{err}_{\mathrm{pow}}^{(i)} = \frac{1}{T_p} \sum_t \left| 20\log\left(\alpha Pow_{\mathrm{org}}(t)\right) - 20\log\left(Pow_{\mathrm{syn}}^{(i)}(t)\right) \right|, \quad (5)$$

where T_f denotes the number of voiced frames, and T_p denotes the number of nonzero power frames.

Table 4 shows the mean error values after each iteration for song No.07, where the "$\times n$" column denotes the number of iterations before synthesis and the "$\times 0$" column denotes initial synthesis without any iteration. Starting from large errors of initial synthesis ("$\times 0$"), the mean error values were monotonically decreased after each iteration and the synthesized singing after the fourth iteration ("$\times 4$") was most similar to the target singing. The results for the other songs also showed similar improvement as shown in Table 5. The "Previous approach" column in Tables 4 and 5 denotes the results of mapping acoustic feature values directly into synthesis parameters (almost equivalent to [7]). The mean error values after the fourth iteration were much smaller than the previous approach. In fact, when we listened to those synthesized results, the synthesized results after the fourth iteration ("$\times 4$") were clearly better than the synthesized results without any iteration ("$\times 0$" and "Previous approach").

3.3 Discussion

The results of experiment A show that our automatic synchronization (lyrics alignment) worked well. Even if there were a few boundary errors (eight errors among 166 notes in No.07 and three errors among 128 notes in No.16), they

[4] This table does not show another type of error where the global phrase boundary was wrong. There were two such errors in No.16 and they could also be corrected through simple interaction (just by moving roughly).

Table 3. Number of boundary errors and number of repairs for correcting (pointing out) errors in experiment A.

Song No.	Synthesis conditions	Number of notes	Number of boundary errors after each repair			
			$\times 0$	$\times 1$	$\times 2$	$\times 3$
No.07	CV01	166	8	5	2	0
No.16	CV02	128	3	2	0	—

Table 4. Mean error values after each iteration for song No.07 in experiment B.

Parameters	Synthesis conditions	Mean error values ($\mathrm{err}_{\mathrm{f0}}^{(i)}$ [semitone] and $\mathrm{err}_{\mathrm{pow}}^{(i)}$ [dB])					
		Previous approach	VocaListener				
			$\times 0$	$\times 1$	$\times 2$	$\times 3$	$\times 4$
Pitch	CV01	0.217	0.386	0.091	0.058	0.042	0.034
Pitch	CV02	0.198	0.352	0.074	0.041	0.029	0.024
Dynamics	CV01	13.65	11.22	4.128	3.617	3.472	3.414
Dynamics	CV02	14.17	15.26	6.944	6.382	6.245	6.171

Table 5. Minimum and maximum error values for all four songs in experiment B.

Parameters	Mean error values (min−max)		
	Previous approach	VocaListener	
		$\times 0$	$\times 4$
Pitch	0.168−0.369	0.352−1.029	0.019−0.107
Dynamics	9.545−15.45	10.46−19.04	1.676−6.560

could be easily corrected by choosing from the top three candidates. We thus confirmed that our interface for correcting boundary errors was easy-to-use and efficient. Moreover, we recently developed an original acoustic model that was trained from scratch with singing voices including a wide range of vocal timbres and singing styles. Although we did not use this high-performance model in the above experiments, our preliminary evaluation results suggest that more accurate synchronization can be achieved.

The results of experiment B show that iterative updates were an effective way to mimic the target singing under various conditions. In addition, we tried to estimate the parameters for CV01/CV02 using song samples synthesized with CV01 as the target singing, and confirmed that the estimated parameters for CV01 were almost same with the original parameters and the synthesized singing with CV01/CV02 sufficiently mimicked the target singing. VocaListener can thus be used not only for mimicking singing by human, but also for re-estimating the parameters under different synthesis conditions without time-consuming manual adjustment.

4 CONCLUSION

We have described a singing-to-singing synthesis system, VocaListener, that automatically estimates parameters for singing synthesis by mimicking a user's singing. The experimental results indicate that the system effectively mimics target singing with error values decreasing with the number of iterative updates. Although Japanese lyrics are currently supported in our implementation, our approach can be utilized for any other language.

In our experience of synthesizing various songs with VocaListener using seven different singer databases on two different singing synthesis systems (Vocaloid and Vocaloid2), we found the synthesized quality was high and stable[5] . One benefit of VocaListener is that a user does not need to perform time-consuming manual adjustment even if the singer database changes. Before VocaListener, this problem was widely recognized and many users had to repeatedly adjust parameters. With VocaListener, once a user synthesizes a song based on the target singing (even synthesized singing the user has adjusted in the past), its vocal timbre can be easily changed just by switching a singer database on our interface. Since this ability is very useful for end users, we name this meta-framework a *Meta-Singing Synthesis System*. We hope that a future singing synthesis framework will support this promising idea, thus expediting wider use of singing synthesis systems to produce music.

[5] A demonstration video including examples of synthesized singing is available at http://staff.aist.go.jp/t.nakano/VocaListener/.

5 ACKNOWLEDGEMENTS

We thank Jun Ogata (AIST), Takeshi Saitou (CREST/AIST), and Hiromasa Fujihara (AIST) for their valuable discussions. This research was supported in part by CrestMuse, CREST, JST.

6 REFERENCES

[1] Kenmochi, H. *et al.* "VOCALOID – Commercial Singing Synthesizer based on Sample Concatenation," *Proc. INTERSPEECH 2007*, pp.4011–4010, 2007.

[2] Hamasaki, M. *et al.* "Network Analysis of Massively Collaborative Creation of Multimedia Contents: Case Study of Hatsune Miku Videos on Nico Nico Douga," *Proc. uxTV' 08*, pp.165–168, 2008.

[3] Cabinet Office, Government of Japan. "Virtual Idol," *Highlighting JAPAN through images*, Vol.2, No.11, pp.24–25, 2009. http://www.gov-online.go.jp/pdf/hlj_img/vol_0020et/24-25.pdf

[4] Bonada, J. *et al.* "Synthesis of the Singing Voice by Performance Sampling and Spectral Models," *IEEE Signal Processing Magazine*, Vol.24, Iss.2, pp.67–79, 2007.

[5] Saino K. *et al.* "HMM-based singing voice synthesis system, " *Proc. ICSLP06*, pp.1141–1144, 2006.

[6] Saitou, T. *et al.* "Speech-To-Singing Synthesis: Converting Speaking Voices to Singing Voices by Controlling Acoustic Features Unique to Singing Voices," *Proc. WASPAA2007*, pp.215–218, 2007.

[7] Janer, J. *et al.*: "Performance-Driven Control for Sample-Based Singing Voice Synthesis," *Proc. DAFx-06*, pp.42–44, 2006.

[8] Camacho, A. "SWIPE: A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music," Ph.D. Thesis, University of Florida, 116 p., 2007.

[9] Kudo, T. "MeCab: Yet Another Part-of-Speech and Morphological Analyzer". http://mecab.sourceforge.net/

[10] Crypton Future Media. "What is the HATSUNE MIKU movement?," http://www.crypton.co.jp/download/pdf/info_miku_e.pdf

[11] Saitou, T. *et al.* "Development of an F0 control Model Based on F0 Dynamic Characteristics for Singing-Voice Synthesis," *Speech Communication*, Vol.46, pp.405-417, 2005.

[12] Mori, H. *et al.* "F_0 Dynamics in Singing: Evidence from the Data of a Baritone Singer," *IEICE Trans. Inf. & Syst.*, Vol.E87-D, No.5, pp.1086–1092, 2004.

[13] Nakano, T. *et al.* "An Automatic Singing Skill Evaluation Method for Unknown Melodies Using Pitch Interval Accuracy and Vibrato Features," *Proc. ICSLP 2006*, pp.1706–1709, 2006.

[14] Goto, M. *et al.* "RWC Music Database: Popular, Classical, and Jazz Music Databases," *Proc. ISMIR 2002*, pp.287–288, 2002.

[15] Lee, A. *et al.* "Continuous Speech Recognition Consortium – An Open Repository for CSR Tools and Models –," *Proc. LREC2002*, pp.1438–1441, 2002.

[16] Digalakis, V.V. *et al.*: "Speaker Adaptation Using Combined Transformation and Bayesian Methods," *IEEE Transactions on Speech and Audio Processing*, Vol.4, No.4, pp.294–300, 1996.

AUTOMATIC JAZZ HARMONY EVOLUTION

Kjell Bäckman
IT University
kjell.backman@hv.se

ABSTRACT

Jazz harmony has during jazz history mainly been functionally based on principles of tonality derived from the classical and romantic periods of the 18th and 19th centuries. In the Evolutionary Jazz Harmony project we introduced a functionless harmony system that impacted the musical feeling in jazz compositions to imitate the harmonic feeling in an avant-garde way. The main features of that new harmony system were chords not built on any specific base note and not necessarily connected to the major/minor concept. In this project we introduce an automatic evaluation of the produced harmony sequences that both looks at each individual chord and the chord progression. A population of chord progressions is evaluated and the highest ranked ones will most likely be used for breeding of the offspring.

This project is one of the sub-projects of the EJI (Evolutionary Jazz Improvisation) project, where we explore various aspects of jazz music; improvised solo, harmony, tune creation, algorithmic creation of piano, bass and drum accompaniment, communication between instruments etc. The results have been evaluated by a live jazz group consisting of professional jazz musicians.

1. INTRODUCTION

Jazz harmony has since the birth of jazz been functionally based, which means that each chord has been related to a base note and classified as minor or major, and optionally also enriched with colouring, such as:

Cm, Eb7, G13b9, A7#11

This situation has prevailed throughout jazz history, with some exceptions however. The earliest experiments with other kinds of harmony were made in the 1950's by advanced and forward-thinking musicians like Ornette Coleman, Cecil Taylor, Don Cherry and others. Experiments have also been made during the 60's and 70's by e.g. Herbie Hancock, Miles Davis and fusion musicians like the Brecker Brothers. Not to mention all the experiments in the classical music domain during the 20th century from Schoenberg and onwards.

However, from the last quarter of the 20th century a stagnation of the harmonic development in jazz ensued, and very little harmonically essential has occurred. The Automatic Jazz Harmony Evolution project is an attempt to break the ice and open new dimensions in harmonic thinking. Persichetti [11] has made a harmony study that has been a valuable resource in this project. Pachet [10] has designed a system for rhythm and harmony evolution, however without the automatic evaluation feature of this project.

The Automatic Jazz Harmony Evolution project uses a non-functional harmony philosophy (no specific base note and not necessarily connected to the major/minor concept), where the "chords" are built up by means of the computer using evolutionary principles.

The produced chord progressions are used by the automatic jazz composer function described in another paper to produce tunes, and by the generative jazz improvisation program to produce jazz solos based on this kind of harmony. Some of these papers are still work in progress but others have been published. There are some publications written by the author that provide valuable background information to this project [1,2,3].

Dahlstedt [4,5], Dean [7], and Thywissen [14] have made valuable contributions in the same area and have been sources of inspiration for this project.

2. BACKGROUND

The harmony organization in jazz has already from the beginning and during its first two decades of the 20th century been systematically organized around a tonal centre by fifth progressions, see Levine [8]. Blues and ragtime harmony mainly used simple major/minor triads at a distance of fifths. Swing music enriched the chords with sixths and ninths but the chord progressions were mainly the same. Bebop further enhanced the chords with colouring such as b9, #9, #11, 13, b13 etc. and exchanged some chord progressions by inserting an extra subdominant parallel, e.g. *G7 – C* was replaced by *Dm7 – G7 – C*. However the focus was still on major/minor and fifth progressions. The main harmonic contribution of cool jazz and hardbop during the 50's meant further advanced chord colouring. A few forward-thinking musicians began at the end of the 50's to split up the harmonic foundation

prevailing until then, and this development continued during the subsequent decades under the stylistic classification of “modal jazz”, “avant-garde”, “free form” etc. To some extent current jazz musicians have adopted this break-up tendency.
However, mostly in the “modern” jazz styles some remainders of the functional harmony principles and the fifth circle basis can be traced. When traditional musicians create compositions with new harmony, there is still a risk of getting stuck in conventions dictated by routine behaviour and idiomatic properties of the instrument. An evolutionary algorithm has no such restrictions but creates harmonies controlled by the algorithms having been programmed. The aim is to free oneself from traditional thinking and create other kinds of harmony.

3. AUTOMATIC EVOLUTION

A typical evolutionary algorithm process starts with a basic set of parameters, from which it creates an initial random population of pictures, melodies, chord progressions or whatever. The evaluation function then examines the population individuals and gives each one a score. Individuals with the highest score have the highest probability of becoming parents of the next generation. The breeding is done by combining the genome of two or more parents, optionally by applying a mutation somewhere in the genome. The mutation might imply a shift between two genome values, or a slight modification of a genome value.
The principle of using evolutionary algorithms to develop new artistic production, enhance artistic thinking and stimulate creativity, first started on a broader scale in the digital graphics area, by forerunner Karl Sims [13]. The evolutionary algorithms principle is well accommodated to that area because when using interactive evaluation of a created generation, as described by Dawkins [6], you can swiftly scan a great number of pictures and select the best according to your personal preference. With audio material, however, the evaluation procedure is much slower since you will have to listen through each music individual produced in a generation, one at a time. The first experiments in the music area were made by Collin Johnson and Palle Dahlstedt.
The evaluation, selection and breeding is repeated generation by generation until you arrive at a genome good enough to be used for the reproduction of artworks (pictures / melodies etc.).
This process is much the same as the genetic process of creating a new species generation in nature, only that it must be sped up considerably to have a chance of being completed in the proper time. The number of generations used for one evolution session must be limited, the calculation of parameter values must be optimized and efficient to allow for a rapid development towards a good genome, and the fitness function must be user friendly to minimize tedious manual intervention. Therefore, to take full advantage of the strength of the evolution process in terms of a large population and a great number of generations, we have in this project made an automatic process, which has required a careful analysis of abstract items such as tension, climax, phrasing, musicality etc. Such a function has also been developed by the author for jazz improvisation solos [3].
The genome in this project consists of parameter values specifying the internal structure of each chord and the progress from one chord to the next. For each new generation one parent chord progression is combined with another by selecting various portions of each of the parents’ genomes. For each child different sections of the parents’ genomes are selected, optionally also by performing a mutation which might consist of a slight modification of some genome parameter values.

4. METHOD

There are a number of parameters controlling the overall behaviour of the genetic evolution process:
- Number of notes per chord (4-5)- Number of notes to change from chord to chord (1-5). A higher value gives abrupt chord changes, while a lower value gives a more homogenous chord sequence.
- Maximum number of half-note steps to be allowed when a voice moves from chord to chord; 1, 2 or 4. Also in this case, greater tolerance gives more abrupt chord changes.
These parameter values can be manually set prior to starting an evaluation session. We have experimented with different settings, where the following seems to produce the best result: 4 notes per chord, 2 notes changed per chord, maximum 2 half-note steps.
A genome consists of the absolute MIDI pitches for the initial chord. The pitches are randomly created within a specific pitch range around middle C. For each chord change the genome holds the number of half-note steps per note (fig. 1).

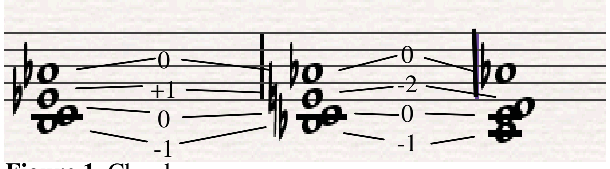


Figure 1. Chord genome

In this case the genome will be:
59 60 63 68 -1 0 +1 0 -1 0 -2 0 …

From the beginning, an initial population of 100 individuals (chord sequences) is created.
Each individual is then evaluated. The evaluation of a chord sequence is based on the principles that small chromatic steps from chord to chord have the strongest emotionally pushing character, and that upward intervals tend to increase the intensity at most. Therefore, such characteristics will be favoured (table 1).

Type of interval	Contribution
Upward minor second	3
Upward major second	1
Downward minor second	2

Table 1. Voice step scoring.

The internal structure of each chord is also evaluated, where intervals like small seconds and quarters are premiered, since they tend to avoid tonal centres, while intervals like thirds and fifths are avoided for the same reason. However, within one chord, only one small second is allowed to avoid cluster chords. The same applies to quarters. Table 2 shows the contribution figures from the internal chord analysis.

Type of interval	Contribution
Minor second	3
Major second	2
Fourth	1

Table 2. Internal chord interval scoring.

The contribution total for the entire chord sequence is saved for each individual of the population. The individuals with the highest score will most likely be subject to parentship for the next generation. A probability figure corresponding to the score of the individual impacts the random selection of parents.
On breeding, the crossover is made by combining different sections of two parents' genomes just like the process of combining DNA for species. The two parents are randomly selected with consideration to their probability figure based on their evaluation score. This means that the figures (e.g. -1, 0, 1, 0) are taken from one of the parents from the beginning of the genome, up to the randomly selected break point, from where the remaining figures are taken from the other parent.
At the end of the breeding a mutation is made by amending a few values one step up or down, so -1 might be -2 or 0, etc.

When a child has been created in this way, it is evaluated as described above. If the child's score exceeds that of the worst parent, it will replace that parent, which is discarded. If the created child is worse than the worst individual of the population, the child will be discarded. Thus, the elitism principle is used, which means that a created child, if kept, will always improve the quality of the entire population.
In this experiment 1000 iterations are used in each run. The solution acquired should not be considered a global optimum. In relation to the evaluation function we can not even be sure that we have arrived at a local optimum, since further iterations might have given still a better score. Maybe a larger number of iterations could result in a still better solution, but by experimentation we have found 1000 iterations enough.
At each run we arrive at a new "near-local-optimum", and selection of the best of all solutions is a question of personal taste.
The program code is written in C++, including the MIDI compiler function, which makes it possible to use any media player to listen to the produced MIDI files, and also import them into a note editing program, such as Sibelius. The resulting chord progression is also stored in an ASCII file in a format possible to copy to the project folder for jazz improvisation solos [1,3].

5. EXPERIMENT EXAMPLE

In this test run, one of the initial individuals had the genome shown in table 3.

58	-1	1	0	-1	-1	-1	0	0	-1	1	1
59	-1	1	0	0	-1	0	-1	1	1	0	0
62	1	0	1	0	-1	0	0	0	1	0	0
69	0	0	0	0	2	0	0	0	-1	0	0

1	-1	0	0	1	0	0	0	0	1	0	0
1	0	1	1	0	0	1	-1	1	0	0	0
-1	0	0	0	0	1	0	0	0	0	1	1
0	1	-1	1	0	0	0	0	0	0	0	0

Table 3. Experiment genome example.

The genome example corresponds to the first bars of the score shown in figure 2.

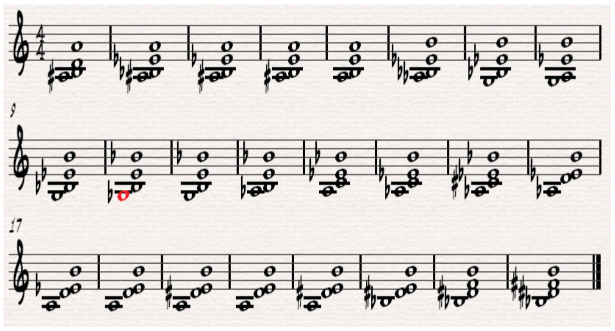


Figure 2. Experiment genome example.

Here is a link to the sound file:
http://oden.ei.hv.se/kjell/porto/Chords1.wav
Figure 3 shows the score after mutation (the mutations are indicated by X in the score).

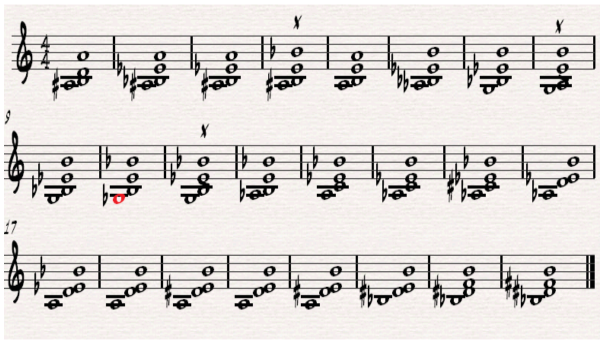


Figure 3. Score after mutation.

Here is a link to the mutated sound example:
http://oden.ei.hv.se/kjell/porto/Chords2.wav
Figure 4 shows the score after 1000 iterations.

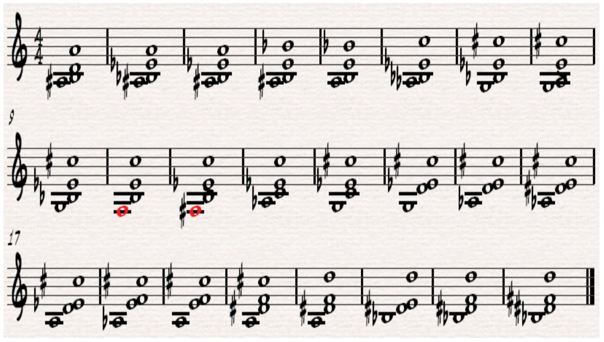


Figure 4. Score after 1000 iterations.

The sound file link for the score in figure 4 is:
http://oden.ei.hv.se/kjell/porto/Chords3.wav

The system also creates a scale per chord, which is used as basis for creating the tune and as basis for improvisations. Figure 5 shows the scales for the first chords.

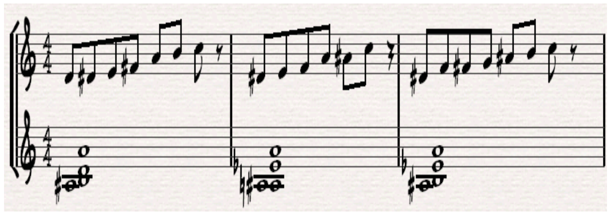


Figure 5. Scales for the first chords.

One of the EJI sub-projects uses an algorithm for creation of a tune based on the chord progression and scales. That sub-project has not yet been completed, but a prototype has been developed, and the full algorithm will be documented in the future. Figure 6 shows the score for a tune generated by the prototype:

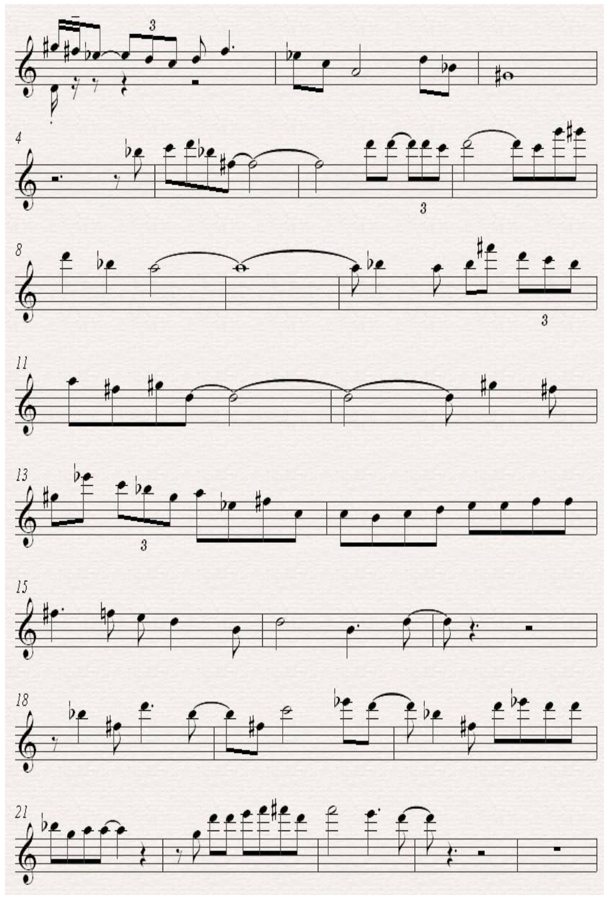


Figure 6. The generated tune.

The following link gives the complete tune with chords, melody and improvised solos:
http://oden.ei.hv.se/kjell/porto/Tune.mid
The drum, bass and piano accompaniment in this sound example are algorithmically created. The procedure for this

is documented in a paper not yet published. Further documentation will follow on this link:
http://oden.ei.hv.se/kjell/eji/eji.htm
A similar tune has been rehearsed and recorded by our live jazz group. The following link gives a recorded jam session:
http://oden.ei.hv.se/kjell/trio/Random2.mp3

6. RESULTS

Chord progressions created this way provide the feeling of a continuous progress towards new heights without arriving at rest points, which is the case with traditional functional harmony, where some chords have a striving character to dissolve into tonics. Compare the chord sequence of a tune like ‘Autumn Leaves’:

Am7	*D7*	*Gmaj7*	*Cmaj7*
F#m7b5	*H7b9*	*Em7*	*Em7*

There is an intermediate rest point at the chord Cmaj7 and then a final rest point at Em7. These rest points provide a relaxation at various positions of the tune, which gives a periodic character. Such relaxation points are not found in tunes with the new kind of harmony. Whether this depends on what people have been used to for a long time, or real built-in features of the functional harmony, is another topic not discussed here. Our conclusion is that this new kind of harmony has an on-going forward-striving feature not prevalent in standard jazz harmony.
Compared to manual evaluation, where you have to listen to each generated individual, one at a time, the automatic evaluation has a number of advantages. We can have a much larger population, the evaluation criteria are kept strictly constant i.e. we do not change focus on the objectives of our evolution process, and the evolution process is rapid. Of course there are also drawbacks with automatic evaluation. It is difficult, if at all possible, to make the computer evaluate abstract concepts such as musicality, tension, expectation, climax, relaxation etc. Anyway, with the automatic evaluation we obtain results that might not otherwise have been discovered.
When jamming with a jazz group on tunes with this new type of harmony, it has the effect on the soloist of continuously proceeding towards a climax never completely reached. The soloist is compelled to go on and on and on. The listener will be involved in this forward-striving feeling of wanting more all the time, and this is an interesting feature that some people might find valuable.
When I experimented with these ideas in a live jazz group, it turned out that the musicians had apparent difficulties in keeping chords and scales in their minds during their solos, since they had to learn completely new chords and scales. The harmony was of a kind that they could not apply their current knowledge and personal routine and not trust old learnt patterns of behaviour. Clever and experienced musicians appeared to be relative beginners, at least during the first rehearsals. Difficulties became obvious especially when playing tunes with an odd periodicity where a chord could last for 3 bars and the next chord for 2 ½ bar, etc. So the time required for rehearsal tended to grow remarkably. For example the bassist, who normally bases his walking bass paths on a base note accentuated at the first beat of each bar and scale walking at the remaining beats, got into problems when there was no specific base note. Learning to play this new kind of music is a laborious task that requires a new way of thinking and a lot of practice and patience.
Furthermore, to find the most adequate way of playing, a lot of time in discussion and reflection has been used in the acoustic live jazz group. For instance, a great deal of cooperative work has been spent by accommodating the bassist’s notes and the piano chord layout to each other.

7. CONCLUSIONS AND FUTURE WORK

Do evolutionary algorithms provide any valuable artistic material? At least some sounding examples are of interest and provide unpredictable and novel artistic output. A jazz tune composer often uses standard chord progressions learnt during a long time of practicing and concerting. He relies on routines built up through repeated usage of similar chord colouring.
The new harmonic system presented in this paper provides a tool for creating a new kind of harmonic base by means of evolutionary algorithms and automatic evaluation, enabling us to take full advantage of the powerful evolution process by virtue of hugh populations and a large number of generations. The resulting harmonic schemes can be used as a foundation for new jazz tunes and for exploring the world of jazz improvisation.
It may appear paradoxical to use tonal rules to build atonal music as implemented in the evaluation process (avoiding thirds and fifths), but since we during several hundred years have grown accustomed to tonal music, we have chosen to originate from that culture when designing the evaluation rule system. This may however be changed in future development, and we will welcome any feedback from the reader about the design of the evaluation process.
The main purpose of using computer based support to produce jazz music is that it opens your mind to new ways of thinking and frees you from old habits of reflection. Hopefully it can enrich your harmonic and improvisation style with new kinds of musical material.
However, introducing a new way of thinking revolutionizes the musical habits of experienced musicians. Such a new system requires considerable time for reflection and rehearsal, which has been proved by experience and discussions in the live jazz group. However, Psyche et al

[12] verify that learning a new musical grammar could be done by repeated exposure.

The project described in this paper is a subproject to the entire EJI (Evolutionary Jazz Improvisation) project, where we work with algorithmic production of jazz harmony, jazz tunes and jazz improvisation. The results of this subproject will be used for future EJI work which will be documented on the EJI web page http://oden.ei.hv.se/kjell.

Our plans are for instance to experiment with the application of PSO (partical swarm optimization), ACO (ant colony optimization), simulated annealing, multiobjective optimization, neural networks, artificial intelligence and other types of heuristics in jazz music creation. This work has been initiated with promising results.

8. REFERENCES

[1] Bäckman, K. (2008) A Generative Representation for the Evolution of Jazz Solos. In *Proceedings of EvoWorkshop Conference 2008, Napoli*.

[2] Bäckman K. (2008) Evolutionary Jazz Harmony, In *Proceedings of BIOMA Conference 2008, Ljubljana*

[3] Bäckman K. (2008) Automatic Fitness in Evolutionary Jazz Improvisation, In *Proceedings of ICMC Conference 2008, Belfast*.

[4] Dahlstedt, P. (2004) Sounds Unheard of - Evolutionary algorithms as creative tools for the contemporary composer, PhD thesis, Chalmers University of Technology, Gothenburg.

[5] Dahlstedt, P. (2001) Creating and Exploring Huge Parameter Spaces: Interactive Evolution as a Tool for Sound Generation In Proceedings of ICMC Conference 2001, La Habana, Cuba.

[6] Dawkins, R. (1976) The Selfish Gene. Oxford University Press Inc., New York , USA.

[7] Dean, T. (2003) Hyperimprovisation: Computer-Interactive Sound Improvisation. A-R Editions Inc.,Middleton, Wisconsin.

[8] Levine, M. (1995) The Jazz Theory Book. SHER MUSIC CO. Petaluma, CA, USA.

[9] Michalewicz, Z. and Fogel, D. (2000) How to Solve It: Modern Heuristics. Springer-Verlag, Berlin Heidelberg, Germany.

[10] Pachet, F. (2000) Rhythm as Emerging Structure. In I. Zannos, editor, Proceedings of ICMC, pages 316-319, Berlin, Germany. ICMA

[11] Persichetti, V. (1961) Twentieth Century Harmony. Vail-Ballon Press, Inc., USA.

[12] Psyche Loui, et al (2006) Acquiring New Musical Grammars: a Statistical Learning Approach, Proceedings of ICMPC06, Bologna, Italy.

[13] Sims, K. (1991) Artificial Evolution for Computer Graphics, ACM SIGGRAPH '91 Conference Proceedings, Las Vegas, Nevada, July 1991.

[14] Thywissen, K. (1996) GeNotator: An environment for investigating the application of generic algorithms in computer assisted composition. In Proceedings of International Computer Music Conference 1996 (ICMC96), pp. 274-277, Hong Kong.

CONTENT ANALYSIS OF NOTE TRANSITIONS IN MUSIC PERFORMANCE

Loureiro, Mauricio
Escola de Música
UFMG
mauricioloureiro@ufmg.br

Yehia, Hani
Escola de Engenharia
UFMG
hani@ufmg.br

de Paula, Hugo
Dep. de Informática
PUC - Minas
hugodepaula@gmx.net

Campolina, Thiago
Escola de Engenharia
UFMG
thicampolina@yahoo.com.br

Mota, Davi
Escola de Música
UFMG
davialvesmota@yahoo.com.br

ABSTRACT

Different aspects of music performance have been quantified by a set of descriptor parameters, which try to incorporate the resources used by the performer to communicate his/her intention of expressiveness and intelligibility. The quality of note transitions are quite important in the construction of an interpretation. They are manipulated by the performer by controlling note durations and the quality of attacks and note groupings. These characteristics can be modeled by parameters that may describe what happens between the notes of a musical sentence, which attempt to represent how we perceive note articulations and groupings of legato or detached notes. On the other hand, the quality of transitions between legato notes may be related to the musician's abilities, to the reverberation characteristics of the performance room and to the acoustic characteristics of the instrument. This text illustrates methods of extraction and definition of descriptor parameters related to the quality of transitions between notes, which are capable to reveal relevant aspects about the accomplishment of these transitions. The procedures here described integrate a model of analysis of the expressiveness of performance in monophonic musical instruments. The samples used consist of recordings of interpretations of excerpts of the classic repertoire for solo clarinet.

1. INTRODUCTION

The comprehension of the processes involved in the production and perception of an expressive performance has been approached by models that quantify the player's expressive intentions upon acoustic information. Studies on musical expressiveness have demonstrated that musicians use small variations of duration, articulation, intensity, pitch and timbre to communicate to the listener aspects of the music that they interpret [2, 3]. By comparing performances of different musicians as well as different interpretations by the same musician, these deviations can be perceived with a surprising clarity, even by non specialized listeners. The quantification of the interpreter's expressive intentions upon such deviations involves the identification and measurement of a set of descriptor parameters defined and calculated from information extracted from the audio signal of the recorded musical performance. In order to incorporate different aspects of the resources used by the performer to communicate his/her intention of expressiveness and intelligibility, these parameters may be established for different segmentation levels, such as musical notes, groups of notes, specific areas in the extent of a same note or in the transition between consecutive notes.

SMC 2009, July 23-25, Porto, Portugal

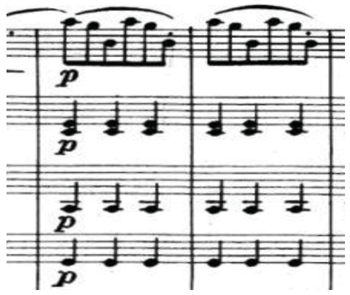
1.1. Transitions between Consecutive Notes

Characteristics related to the quality of the transitions between consecutive notes are decisive in the construction of an interpretation. They are manipulated by the performer by controlling note durations and the quality of attacks and note groupings. These characteristics can be modeled by parameters that try to describe the ephemeral sonorities that happen between the notes of a musical sentence, as an attempt to represent how we perceive note articulations and groupings of legato or detached notes. The quality of transitions between notes may be related not only to the performer's intention, but also to his/her musical skills, to reverberation characteristics of the performance room and to acoustic characteristics of the instrument.

During the short period of time of a transition between two consecutive notes of a musical phrase, inharmonic sounds or strange frequencies to both of the involved notes can occur. On the clarinet, occurrences of lower frequencies in low levels of intensity are very common in passages of fast and long range leaps or between different registers. This kind of sonority was explored by Carl Maria von Weber in the Minuet of his *Grand Quintetto* in B flat Major op. 34 for clarinet and string quartet. The clarinet plays a three-note arpeggio, in which a leap from B 4[1] (the instrument's longest vibration mode), to A 5 (vibration mode confined in a small portion of the tube)

[1] As written for B flat clarinet.

produces an F 4, that seems to be a sub-harmonic of A 5. While this is not well perceived as a pitch, this three-note arpeggio enhances the color of a dominant seventh chord (G-B-D-F), in which the A 5 on the clarinet works as a passing tone. Weber repeats the same arpeggio transposed up a minor third, a passage acoustically similar and with the same characteristic sonority. Did Weber hear this "sub-tone" as a the seventh of this dominant chord (Figure 1)?



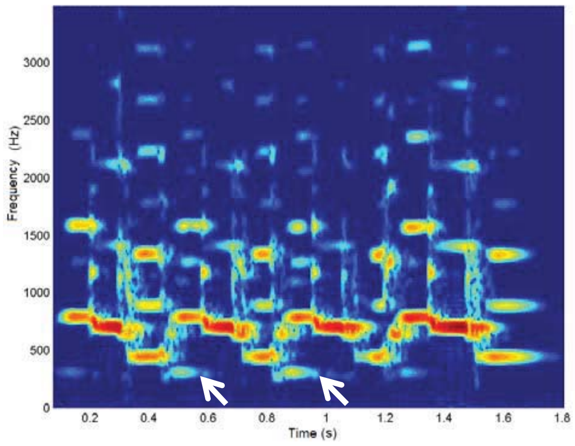


Figure 1. Sonogram of three-note arpeggio of the Minuet of the *Grand Quintetto* in B flat Major op. 34 for clarinet and string quartet by Carl Maria von Weber, showing a sub-harmonic pitched F during the leap from B 4[1] to A 5 (indicated by arrows).

Such sonorities can be found quite often in the clarinet repertoire. Bohuslav Martinu in his *Sonatina for Clarinet and Piano* and Alban Berg in the last of the *Four Pieces for Clarinet and Piano* Op. 5, explored the sound of fast passages between the lower and mid registers of the instrument. In *Clarinet Threads* for clarinet and tape, Denis Smalley explores the spectral morphologies of these sonorities by asking the player to reduce air and lips pressure. These sub-tones, which constitute the thematic material of the work, can only be produced in very low intensity levels. The balance of dynamic level is guaranteed by electronic amplification of these sounds.

[1] As written for B flat clarinet.

This text illustrates methods of extraction and definition of descriptor parameters related to the quality of transitions between consecutive notes, that may describe not only the degree of the musician's intentional manipulation of note articulations, but also aspects related to the musician's ability in accomplishing legato transitions and to acoustic characteristics of the instrument. The procedures here described integrate a model of analysis of the expressiveness of performance in monophonic musical instruments. The used samples consist of recordings of interpretations of excerpts of two major pieces from the traditional classic repertoire for solo clarinet.

2. METHODS

2.1. Materials

The compositions used in this study were the *Quintet for Clarinet and String Quartet* in A Major Kv 588 and the *Concert for Clarinet and Orchestra* in A Major Kv 622, both by W. A. Mozart.

Different solo performances of excerpts from these pieces were recorded without the accompaniment, in the same day, under the same conditions of acoustic environment and equipment. Four different professional clarinetists, 2 of them members of the Philharmonic Orchestra of …(my state), performed each excerpt 3 times, in different ways, according to the instructions:

- Performance **P1** – an expressive performance, as in a concert situation, in which the piece would be played entirely.
- Performance **P2** - without any expression intention, but trying to adopt the same tempo of **P1**.
- Performance **P3** - an expressive performance, as in the first execution, however with different expressive intentions.

2.2. Segmentation

The first step for an appropriate estimate of descriptor parameters is to segment the signal into events to be analyzed, such as musical notes, note groups, or specific regions within a single note. Segmentation is not a trivial problem, even on monophonic musical signals, especially if we consider the subjectivity in the discrimination of these events. Note onsets and offsets were detected on the RMS envelope averaged for 23 ms using an adaptive threshold, as suggested by De Poli [1], calculated as the average energy in a certain neighborhood (1 s for a step of 6 ms) of each point of the RMS. Onset and offsets are detected by searching the minimum RMS between two consecutive values crossed by this dynamic threshold. The estimate of the fundamental frequency changes, with a pitch threshold below a half tone, helped the segmentation in cases where the detection of onsets and offsets was not possible by means of energy level only, such as legato notes.

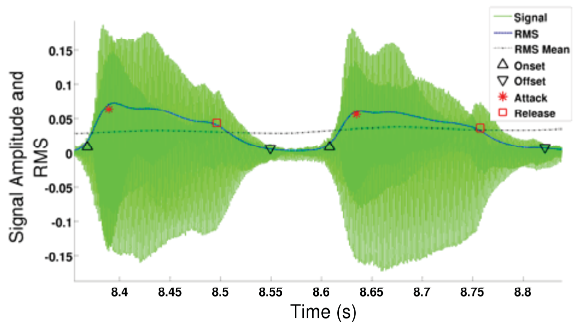


Figure 2. Detection of note onsets and offsets on the RMS envelope using a dynamic threshold calculated as the average energy surrounding each point of the RMS (1 s for a step of 6 ms). Onset and offsets are detected by searching the minimum RMS between two consecutive values crossed by this threshold.

The end of attack was defined as the first amplitude maximum after the note onset, and the beginning of release, as the first amplitude maximum before the note offset. These points were detected by searching for maximum variations of the first derivative of the RMS signal. It doesn't exist in the literature a measurement method that can describe the attack unequivocally [4]. This definition of attack is adequate to describe the attack in most situations, but further consideration was necessary in cases where maximum amplitude was reached much later in the sustained segment of the note.

The presence of transients during note transitions makes possible the use of spectral flux to detect the end of attack, since this point can be related to the reestablishment of harmonics amplitude correlation after note transitions:

$$SF = \frac{1}{T}\sum_{p=1}^{T}\left|r_{p,p-1}\right| \tag{1}$$

where $r_{p,p-1}$ is the correlation coefficient among spectral amplitudes calculated at instants t_p and t_{p-1} of the signal of duration T.

The spectral flux confirmed most of the end of attack estimated by energy local maxima and was also able to detect these points where the energy method was not.

2.3. Descriptors for Note Transitions

In order to analyze the quality of the transition between consecutive notes two descriptors were defined: *articulation index* and *legato index*. Some aspects of the quality of note transitions are related to the performer's intentional manipulations of note articulations, which are well controlled by the musician by manipulating note durations and attack quality. Hence, articulation is considered to be closely related to the performer's intentions of expressiveness and intelligibility.

The *articulation index*, defined as the ratio between note duration DR (time interval between note onset and offset) and the time interval between both note onsets (known as intra-onset-interval - IOI), was used to describe the intentional manipulation of note duration, by the performer:

$$AI(i) = 1 - \frac{DR(i)}{IOI(i)} \tag{2}$$

This index is appropriate to describe transitions between detached notes, usually produced in the clarinet with abrupt interruptions of the air flow by slightly beating the tongue on the reed. With this action, the player controls the quality of the attacks as well as the duration of each note. It should be pointed that the performer's ability to control note duration depends closely on the ambience reverberation conditions.

On the other hand, this index is not adequate for describing the legato transitions, since it assumes values close to 0 for most legato notes.

2.4. Legato Index

Legato notes on the clarinet are produced by means of a single blow without interrupting the air flow during the transition. To investigate the quality of transitions between notes played with the intention to be legato, we used a descriptor suggested by Maestre [4], defined as a comparison to an ideal legato between 2 notes without any decrease of energy, represented by the straight line traced from the beginning of release of a note to the end of attack of the subsequent note. The index is calculated by means of the area A_1 between this line and the energy curve and the total area $A_1 + A_2$ below this straight line:

$$LI = 1 - \left[\frac{A_1}{A_1 + A_2}\right] = 1 - \left[\frac{\sum_{t=release(i)}^{attack(i+1)}(r(t) - RMS(t))}{\sum_{t=release(i)}^{attack(i+1)} r(t)}\right] \tag{3}$$

r being the line traced from the beginning of release of a note to the end of attack of the subsequent note.

This index appears to be related to the musician's abilities, to the ambience reverberation conditions and to the acoustic characteristics of the instrument.

3. DISCUSSION

3.1. Articulation Index

The capability of these two descriptors to represent the note transitions of a musical phrase played in a monophonic instrument is illustrated below with the first sentence of the main theme of the first movement of the Quintet for Clarinet and String Quartet in A Major Kv 588 by W. A. Mozart, bars 118 through 124 (Figure 3).

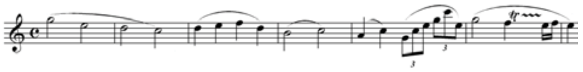


Figure 3. First sentence of the main theme of the first movement of the Quintet for Clarinet and String Quartet in A Major Kv 588 by W. A. Mozart (bars 118 through 124).

Figure 4 shows the evolution of the articulation index of the note transitions along performances **P1** and **P2** of the first five bars of the phrase of Figure 3, by one of the clarinetists. It can be observed that all transitions up to 13 (between the first 2 eighth triplets G-C of the 5th bar) present index values below 10 % for both performances. This is because all these notes were played legato, with the exception of some soft articulations. Transitions 14 through 18 of performance **P1** presented index values varying from 15 % to 26 %. In this performance, all these notes were articulated**,** while in performance **P2** the triplets arpeggio was played legato up to the high C (adopted articulation is shown in the upper part of Figure 4).

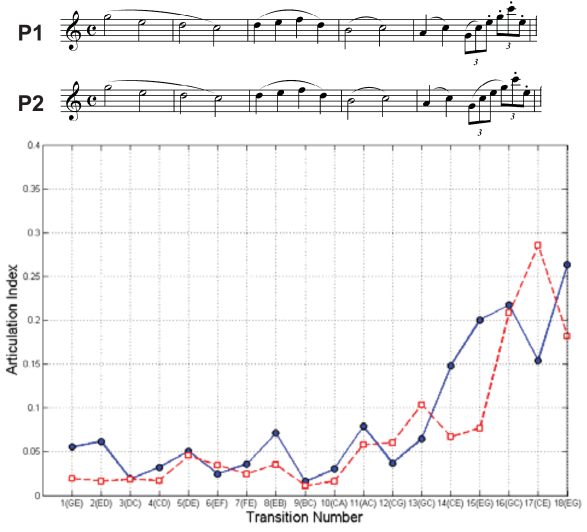


Figure 4. Articulation index of performances **P1** (continuous line) and **P2** (traced line) of the first 5 bars of the main theme of the first movement of Mozart Clarinet Quintet Kv 588, by the same musician (adopted articulation shown on the top).

As expected, this index was not adequate for describing transitions between legato notes, as the smooth articulations performers use to define phrase boundaries, such as transition 8 (D of the end of third bar to B of the beginning of the fourth bar). Despite the perceptual evidence of the articulation produced in performance **P1** in this transition, the index scored only 7 %, which appears to show that this index is not adequate for describing quasi legato articulations.

3.2. Legato Index

Figure 5 shows the legato index of performances **P2** (musician instructed to play with no expression) and **P3** of the first 4 bars of the same excerpt and by the same clarinetist of Figure 4. Lower values of this index confirm the phrase articulations at transitions 4 and 8, evident in both performances. Performance **P3** presented over all lower values than performance **P2.** This might be explained by consistent energy variation along the whole sequence of legato notes in performance **P3** (played with expression), which was not only clearly audible, but also confirmed by the RMS curve. Although it is well known that a good legato on the clarinet is achieved with a very uniform blow with minimal fluctuation of the air pressure, consistent intensity variation, as heard in performance **P3**, is not rare in "molto expressive" performances, in which the varying air pressure compromises the legato quality.

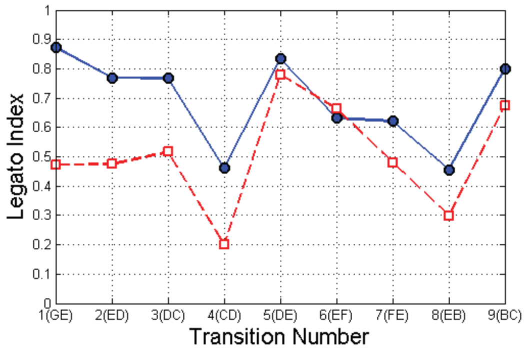


Figure 5. Legato index of performances **P2** (continuous line) and **P3** (traced line) of the first 4 bars of the main theme of the first movement of Mozart Clarinet Quintet Kv 588, by the same musician.

A sudden increase of the legato quality could be observed at transitions 5 and 9, which come right after both articulations phrase at transitions 4 and 8. Such increase, observed in several alike passages, might be explained by the fact that right after an interruption of the air flow to produce the articulation, it seems to be easier to build a more uniform air pressure for the subsequent note transition. Figure 6 shows another example of this at transition 5 of both performances **P2** and **P3** of the same excerpt by another clarinetist. Note that this player decided not to articulate transition 8, despite the natural articulation of this phrase, marked by harmony change to the dominant.

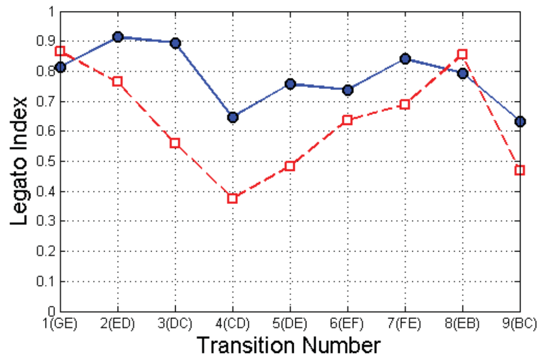


Figure 6. Legato index of performances **P2** (continuous line) and **P3** (traced line) of the first 4 bars of the main theme of the first movement of Mozart Clarinet Quintet Kv 588, by a second clarinetist.

Figure 7 illustrates another example of such behaviors of this index in fast note transitions (6/8th measure at 96 BPM): performances **P1** and **P2** of the first sentence (6th to 13th notes) of the third movement of Mozart Clarinet Concerto Kv 622, played by the second clarinetist. Better legato quality could be also observed in the non expressive performance, as well as at transition 7, which comes right after the phrase articulation of transition 6.

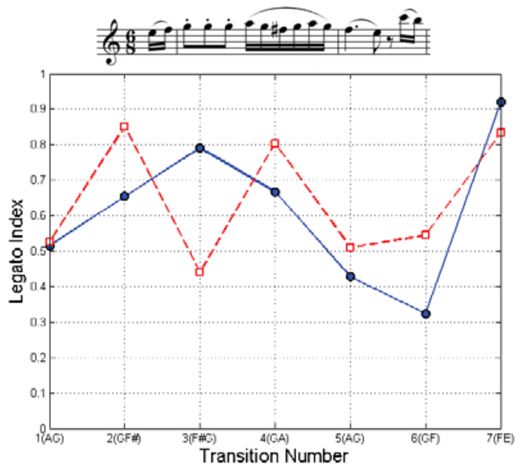


Figure 7. Legato index of fast note transitions: Performances **P1** (continuous line) and **P2** (traced line) of notes 6th through 13th of the third movement of Mozart clarinet Concerto, played by the second clarinetist (6 / 8th measure at 96 BPM).

4. CONCLUSION

This paper presented quantitative ways (namely, articulation index and legato index) to measure and compare transitions between notes, as performed by four clarinetists in three different ways. The articulation index allowed a description of the degree of the musician's intentional manipulation in the accomplishment of articulation between notes.

Measures of the legato index made possible to infer about the quality of transitions between notes executed with the intention to be legato. These results suggest a dependence of the value of this index to aspects related to the musician's ability to play legato notes as well as to acoustic characteristics of the instrument, as it corroborates many aspects of practical experiences in playing and perceiving legato notes.

Higher variance of the legato index was evident for all expressive performance, when compared with non expressive performances. However no single pattern could be derived from these variations. Possible correlation between this index with the degree of some aspect of musical expressiveness, may suggest the adequacy of this descriptor for music expression investigations.

5. ACKNOWLEDGEMENTS

The accomplishment of this work was possible thanks to the participant clarinetists. This work was supported in part by FAPEMIG (Research Funding Agency of Minas Gerais State, Brazil) and CNPq (Brazilian National Council for Scientific and Technological Development).

6. REFERENCES

[1] De Poli, G. and L. Mion, "From audio to content," In *Umpublished book*, G. De Poli, Ed. Padova: Dipartimento di Ingegneria Dell'Informazione - Università degli Studi di Padova, 2006.

[2] Gabrielsson, A., "Music Performance Research at the Millenium," *Psychology of Music*, vol. 31, pp. 221-272, 2003.

[3] Juslin, P. N., "Cue utilization in communication of emotion in music performance: relating performance to perception," *Journal of Experimental Psychology: Human perception and performance*, vol. 26, pp. 1797-1813, 2000.

[4] Maestre, E. and E. Gómez, "Automatic Characterization of Dynamics and Articulation of Expressive Monophonic Recordings," in *Audio Engineering Society 118th Convention Paper*, 2005.

SCORE PLAYBACK DEVICES IN PWGL

Mikael Laurson
CMT, Sibelius Academy
laurson@siba.fi

Mika Kuuskankare
CMT, Sibelius Academy
mkuuskan@siba.fi

ABSTRACT

This paper presents a novel system that allows the user to customize playback facilities in our computer-assisted environment, PWGL. The scheme is based on a class hierarchy. The behavior of an abstract root playback class containing a set of methods can be customized through inheritance. This procedure is demonstrated by a subclass that is capable of playing MIDI data. This playback device allows to realize automatically multi-instrument and micro-tonal scores by using pitchbend setups and channel mappings. Also continuous control information can be given in a score by adding dynamics markings and/or special Score-BPF expressions containing break-point functions. We give several complete code examples that demonstrate how the user could further change the playback behavior. We start with a simple playback device that allows to override channel information. Next we discuss how to implement the popular keyswitch mechanism in our system. This playback device is capable of mapping high-level score information with commercial orchestral database supporting keyswitch instruments. Our final example shows how to override the default MIDI output and delegate the play events to an external synthesizer using OSC.

1 INTRODUCTION

Computer-assisted composition (CAC) systems ([1], [2], [3]) have not focused on advanced playback facilities. Users can typically audition scores and other musical raw-material through basic MIDI playback routines that support simple note-on, note-off, pitch and velocity data. A notable exception of this rule are special cases to handle micro-tonal playback which is not well supported by the MIDI standard.

Commercial notation software packages (Sibelius, Finale, Igor), by contrast, do support tools that allow the user to audition orchestral scores. With the advent of recent high-quality orchestral sample databases (EastWest, Vienna Instruments, Garritan) and/or instrument synthesizers (Wallander Instruments, Synful) orchestral simulations are getting more and more convincing. Thus notation software systems combined with orchestral playback facilities have quickly become everyday tools for composers and arrangers not only in the film industry but also for musicians belonging to the contemporary music genre.

SMC 2009, July 23-25, Porto, Portugal

In this paper we investigate possibilities that would allow the user to combine various playback options, such as micro-interval playing and orchestral simulation in a CAC environment. Also the output would not be bound to a given orchestral database (as is the case in several notation systems where the samples are bundled with the application). The user should be able to customize playing routines for different libraries and synthesizers, and the control output does not necessarily have to be MIDI-oriented.

PWGL has a long history in controlling physics-based instruments [4] using our notation package ENP [5]. This research has resulted in several tools that allow to enrich basic score information, such as performance rules, scripting, tempo functions, and an enhanced set of expressions that can be inserted in the score either algorithmically or by hand. Other special extensions, such as the macro-note scheme [6], allow the user to further modify and enrich basic score information. Sound examples can be found at: www.siba.fi/pwgl/pwglsynth.html.

In the following we will concentrate on a novel extension of the PWGL system that allows the user to define the behavior of the playback engine. Each time the user starts to play a score the current playback device is evoked. PWGL contains a library of predefined playback devices. This library is written in Common Lisp and CLOS and it can be extended by subclassing one of the existing playback device classes. Multiple inheritance can also be used to combine features from several superclasses in the system. All playback devices support a standard protocol having four main steps: (1) the system first calls an initial preparation method, (2) then it collects the actual playback information, (3) next a setup method is called, and finally (4) the collected data is sent to the current output device.

The rest of the paper is organized as follows. First we give some background information concerning the general playback device scheme in PWGL. Then we discuss in more detail the current status, and enumerate which devices are already present in the system. After this we describe case studies that will show how the user can override the default behavior in order to customize her/his needs for auditioning of musical scores.

2 GENERIC PLAYBACK DEVICE CLASS AND ITS METHODS

Our playback system is based on a hierarchy of CLOS classes. At the root we have a generic class called 'pwgl-playback-device'. This class definition contains a set of primary methods to support the four main steps used by the playback scheme. Typically, the user should not override the primary methods (although this is also possible), but should instead redefine the secondary methods. The secondary methods have identical names as the primary ones except for an extension '*' at the end of the name. Next we enumerate the most commonly used methods in the playback scheme.

The method 'prepare-playback' is used to prepare score playback before event calculation. This method can be used to open a sample player application, load sound samples, prepare an instrument setup, etc.

After this initial phase the system starts to collect play data. By default it checks whether the current score has a selection or not (note that ENP also supports discontinuous selections). If a selection is found then only those notes that belong to the selection are considered, otherwise all notes are collected.

After this the system calls for each collected note the methods 'add-playback-cc-events' and 'add-playback-note-event'. The first one is used to collect continuous control information, and the second one is used for note events. (Before the 'add-playback-note-event' also the special method 'add-playback-note-pre-event' is called; we will come back to this method later in this paper in Section 5.) These methods are similar as they should at the end collect association lists (a list containing keyword/data pairs) that are meaningful for the current playback output. For instance if the play information is sent to MIDI then a note event list should contain information dealing with bus (port), status, key and velocity. Internally the 'add-playback-note-event' method calls the 'calc-playback-event' method that builds the association list based on the data that is returned from the 'calc-playback-chan/midi' and 'calc-playback-vel' methods.

Next, the system calls the 'setup-playback' method that is called after event calculation and before playback. This method can be used, for instance, to send volume and pitch-bend information just before playback.

Finally, the realtime playback starts and for each event data list the system calls 'send-playback-event'. Normally the method simply utilizes the pre-calculated association lists and sends the appropriate information to the current playback output.

3 THE DEFAULT PLAY DEVICE

As such 'pwgl-playback-device' is an abstract class and should be subclassed in order to be functional. In this section we outline a typical subclass that is specialized for MIDI output. This class is called 'midi-playback-device', and it is in fact the default play device in PWGL. 'midi-playback-device' has several options for output. First, the final output can either be a realtime stream of MIDI events or a MIDI file. Second, PWGL supports also a special mode for Mac OS X users where play data can be sent directly to a QuickTime synthesizer.

3.1 Micro-tonal playing

'midi-playback-device' supports up to eight MIDI buses or ports (in PWGL a note can have a channel number ranging from 1 to 128, where by default channels from 1 to 16 are sent to port 1, channels 17-32 to port 2, and so on).

During the setup phase 'midi-playback-device' calls the 'setup-playback' method that has two main tasks. First, the user has an option to send a default MIDI volume value to each channel. Second, 'setup-playback' sends pitch-bend data to detune channels in order to support micro-tonal playing. This is done by first analyzing the pitch information of the current score. If the score requires quarter-tone tuning then even-numbered channels are detuned by 50 cents; in case of eighth-tone tuning cannel 2 is detuned by 50 cents, channel 3 by 25 cents and channel 4 by 75 cents, and so on. Based on the micro-tonal content the mapping of MIDI channels is done as follows: if the score is in equal temperament (i.e. there are no micro-tones), then note channel numbers will not change, i.e. channel 1 is played on channel 1, and so on; if the score requires quarter-tone resolution then notes with channel number 1 are delegated either to channel 1 or 2 depending on the pitch content, notes with channel 2 are delegated to channel 3 and 4, and so on; in case of eighth-tone resolution notes with channel number 1 are delegated to channel 1, 2, 3 and 4 depending on the pitch content, and so on.

This scheme with detuning and channel delegation is transparent to the user and it allows to play micro-tonal and multi-instrument scores automatically. As the system supports 128 channels we have a theoretical upper limit of 1/256-tone resolution for micro-tonal tuning. If the user needs multi-instrument scores, this limit is of course lowered as each instrument needs a dedicated set of channels in a micro-tonal context.

During the event data collection phase 'midi-playback-device' calls the method 'calc-playback-event' that performs MIDI port, channel and pitch mapping as described above and returns an association list of raw MIDI data that is optimized for realtime playing. Finally, the 'send-playback-event' method utilizes this information and calls in realtime

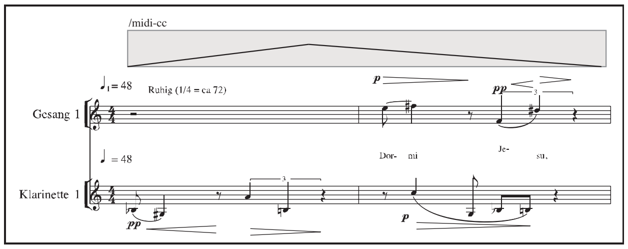


Figure 1. Continuous control options in ENP: (1) crescendo and diminuendo expressions, (2) Score-BPF expression containing a break-point function.

the low-level MIDI event routines in order to output the final MIDI data.

3.2 Continuous control

'midi-playback-device' supports also MIDI continuous control information. The PWGL preference pane has an option that allows to convert automatically ENP expressions to a stream of continuous control MIDI events. Figure 1 shows a two-part score that has several interleaved crescendo and diminuendo markings. When a dynamics expression is encountered in the score during the event calculations phase the 'add-playback-cc-events' method will be evoked. There are two basic options how the dynamics expressions are interpreted: (1) the system creates automatically a ramp that is sampled to get discrete MIDI continuous control events; or (2) the expression contains internally a break-point function that has been edited by the user (ENP allows to open the expression and edit the internal breakpoint function with the mouse). Continuous control information playback is also supported by the special Score-BPF expression (see in Figure 1 the Score-BPF expression above the first staff with the label 'midi-cc'). A Score-BPF can have up to three break-point functions each with individual continuous control designations.

4 CHANNEL PLAYBACK DEVICE

In the following sections we discuss various case studies where we change the behavior of the default PWGL play device. For each case we give a class definition followed by a redefinition of one of the methods discussed in Section 2. Finally we call 'add-playback-device' in order to add the new device to the PWGL playback device library.

We start with a simple case where the idea is to disregard the standard channel information. Instead we check the 'instrument' slot of the note whether it is found in a list of woodwind and brass instruments. Working with symbolic instrument names in conjunction with our notation tools is often more convenient than working with abstract channel numbers (see Figure 2). We use the Lisp function 'position'

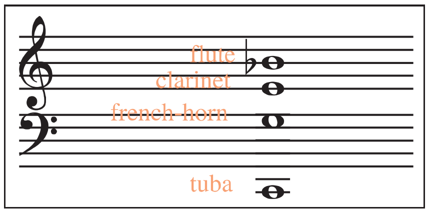


Figure 2. A chord where each note has a different instrument assignment.

to calculate the channel number (thus here 'flute' will be mapped with channel 1, 'oboe' with 2, etc.).

```
(defclass channel-player (midi-playback-device) ())

(defmethod calc-playback-chan/midi*
    ((device channel-player ) (note note))
  (let ((chan (position (instrument note)
          '(:flute :oboe :clarinet :bassoon :trumpet
            :french-horn :trombone :tuba))))
    (values (if chan (1+ chan) (chan note)) (midi note))))

(add-playback-device 'channel-player "channel-player")
```

5 ENP EXPRESSIONS AND KEYSWITCH EVENTS

Next we discuss a generic playback class, called 'keyswitch-player', that is useful when working in conjunction with orchestral databases. 'keyswitch-player' is a subclass of 'midi-playback-device' and thus inherits all features from that class. The idea is to add support to the keyswitch mechanism supported by several commercial sample databases. A keyswitch event is an additional note-on event where the key value is outside the normal range of an instrument. The keyswitch event is normally sent just before the actual note-on event. This allows to instruct the sample database application which specific articulation should be used for the next normal note event. Thus, typically, a keyswitch event having the low C0 as key value for a flute part could mean 'legato', while D0 could mean 'staccato', and so on.

The keyswitch mechanism is straightforward to implement in our scheme due to the 'add-playback-note-pre-event' method that is called just before the actual note event calculation (Section 2). Furthermore, it fits nicely in our system as ENP supports a wide range of expressions (see Figure 3). 'keyswitch-player' adds two new generic methods to the system called 'ksw-expression-no' and 'ksw-offset-no' that should be refined by a subclass of 'keyswitch-player'. These methods are used to calculate the final key value needed for the keyswitch event.

Next we discuss a concrete problem how we can map an instrumental score with expressions denoting various playing techniques and articulations with a keyswitch instrument found in the commercial EastWest sample library. For this end we define a subclass of the 'keyswitch-player' class called 'ew-player '. We need next to associate a keyswitch

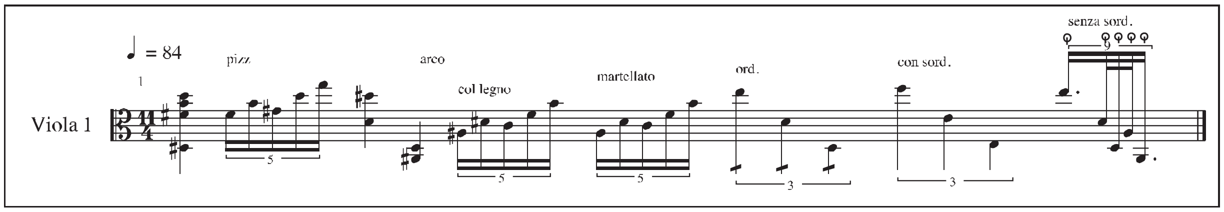


Figure 3. An ENP score for solo viola. The expressions are automatically mapped with correct keyswitch events.

event with an instrument (here viola) and various expressions (see Figure 3). This is done by defining several versions of the 'ksw-exp-no ' method that are specialized for each available expression name ('pizz', 'col-legno, 'martellato', etc.). Each method returns a unique number that–when added to the offset number returned by the 'ksw-offset-no' method–is used by the keyswitch event.

```
(defclass ew-player (keyswitch-player) ())

(defmethod ksw-offset-no
  ((device ew-player) (ins viola)) 24)

(defmethod ksw-exp-no
  (device ew-player) (ins viola) exp) 0)
(defmethod ksw-exp-no
  ((device ew-player) (ins viola) (exp pizz)) 1)
(defmethod ksw-exp-no
  ((device ew-player) (ins viola) (exp col-legno)) 2)
(defmethod ksw-exp-no
  ((device ew-player) (ins viola) (exp martellato)) 3)
(defmethod ksw-exp-no
  ((device ew-player) (ins viola) (exp tremolo8)) 4)
(defmethod ksw-exp-no
  ((device ew-player) (ins viola) (exp bartok-pizzicato)) 5)
(defmethod ksw-exp-no
   ((device ew-player) (ins viola) (exp con-sordino)) 6)

(add-playback-device 'ew-KSW-player "ew-KSW-player")
```

As the ENP instrument database is also based on a CLOS class hierarchy (thus 'viola' is a subclass of 'bowed-strings') the code above can easily be modified so that the keyswitch mechanism is valid for all bowed string instruments, for instance:

```
(defmethod ksw-exp-no
  ((device ew-player) (ins bowed-strings) (exp pizz)) 1)
```

6 OSC PLAYBACK DEVICE

Our final example demonstrates how we can define a playback device that is not using MIDI as output. This is done by defining a subclass of 'midi-playback-device' called 'OSC-playback-device'. Here we want to send OSC [7] messages to an external synthesizer (in this specific case to SuperCollider, [8]). For this reason we redefine the 'send-midi-event*' method. Now, instead of calling the standard MIDI routines, we use the association list information ('midi-info') collected by the event collector and convert the channel, the midi and the velocity data to OSC messages.

```
(defclass OSC-playback-device (midi-playback-device) ())

(defmethod send-midi-event
  ((device OSC-playback-device) midi-info &optional vel?)
  (when-let (osc-stream (read-key :osc-stream))
    (let ((chan (read-key midi-info :chan))
          (midi (read-key midi-info :midi))
          (vel (read-key midi-info :vel)))
      (if (zerop vel)
          (cl-osc:write-osc-message
           osc-stream nil "/noteOff" midi chan)
        (cl-osc:write-osc-message
         osc-stream nil "/noteOn" midi vel chan)))))

(add-playback-device 'OSC-playback-device "osc")
```

7 CONCLUSIONS

The paper presents a playback scheme that allows to convert high-level score information to event lists. The events can be sent to MIDI or to external synthesizers using, for example, the OSC protocol. The system is programmable and through inheritance device subclasses can change the standard behavior of the default playback device of PWGL. This scheme offers many interesting applications such as allowing ENP scores to control orchestral databases. The new playback protocol, when combined with the other tools found in PWGL (i.e. performance rules, scripting, tempo functions, macro-notes), forms a unique system for score-based playback control.

8 ACKNOWLEDGMENTS

The work of Mikael Laurson and Mika Kuuskankare has been supported by the Academy of Finland (SA 114116 and SA 105557).

9 REFERENCES

[1] Laurson, M., *PATCHWORK: A Visual Programming Language and some Musical Applications*. Studia musica no.6, doctoral dissertation, Sibelius Academy, Helsinki, 1996.

[2] Assayag, G., C. Rueda, M. Laurson, C. Agon, and O. Delerue, "Computer Assisted Composition at IRCAM: From PatchWork to OpenMusic", *Computer Music Journal*, vol. 23, pp. 59–72, Fall 1999.

[3] Laurson, M., M. Kuuskankare, and V. Norilo, “An Overview of PWGL, a Visual Programming Environment for Music”, *Computer Music Journal*, vol. 33, no. 1, 2009.

[4] Laurson, M., V. Norilo, and M. Kuuskankare, “PWGLSynth: A Visual Synthesis Language for Virtual Instrument Design and Control”, *Computer Music Journal*, vol. 29, pp. 29–41, Fall 2005.

[5] Kuuskankare, M. and M. Laurson, “Expressive Notation Package”, *Computer Music Journal*, vol. 30, no. 4, pp. 67–79, 2006.

[6] Laurson, M. and M. Kuuskankare, “Towards Idiomatic and Flexible Score-based Gestural Control with a Scripting Language”, *Proceedings of NIME’08 Conference*, (Genova, Italy), pp. 34–37, 2008.

[7] Wright, M., “Open sound control: an enabling technology for musical networking”, *Organised Sound*, vol. 10, pp. 193–200, 2005.

[8] McCartney, J., “Continued Evolution of the SuperCollider Real Time Environment”, *Proceedings of ICMC’98 Conference*, pp. 133–136, 1998.

Author Index