# Extending voice-driven synthesis to audio mosaicing

Jordi Janer, Maarten de Boer
Music Technology Group, Universitat Pompeu Fabra, Barcelona

***Abstract*—This paper presents a system for controlling audio mosaicing with a voice signal, which can be interpreted as a further step in voice-driven sound synthesis. Compared to voice-driven instrumental synthesis, it increases the variety in the synthesized timbre. Also, it provides a more direct interface for audio mosaicing applications, where the performer voice controls rhythmic, tonal and timbre properties of the output sound. In a first step, voice signal is segmented into syllables, extracting a set of acoustic features for each segment. In the concatenative synthesis process, the voice acoustic features (target) are used to retrieve the most similar segment from the corpus of audio sources. We implemented a system working in pseudo-realtime, which analyzes voice input and sends control messages to the concatenative synthesis module. Additionally, this work raises questions to be further explored about mapping the input voice timbre space onto the audio sources timbre space.**

## I. Introduction

State of the art synthesizers are able to generate realistic sounds, using physical models or advanced sample-based techniques. In addition, feature-driven synthesis of audio material is a recently emerging field. A particularly recognizable instance of this field is Audio Mosaicing, the practice of automatically assembling micro-segments of songs, or other audio, to match a pre-determined source. At the same time, major challenges in current digital musical instruments are on the control side. Since the advent of MIDI, a wide variety of musical interfaces (musical controllers) have been proposed to control synthesizers. In this context, the use of the voice to control sound synthesis represents an interesting path for improving music interaction. In this paper, we aim to extend voice-driven synthesis from the control of instrumental sound synthesis to the control of audio mosaicing. Voice input controls the rhythmic, tonal and timbre properties of the output sound, which combined with a loop based mechanism becomes an appropriate system for live performing.

### A. Related work

Regarding the use of voice in music interaction, audio-driven synthesis [1] uses features from an input audio signal, usually from another instrument, to control a synthesis process. When using the voice as input signal, the front-end has been also referred to as singing-driven interface [2]. For the latter case, previous research addressed the characteristics of the singing voice in instrument imitation, highlighting the role of phonetics in terms of timbre and musical articulation. The present work seeks to exploit the timbre possibilities of the voice input to find similar audio micro-segments from other audio sources. This differs from other voice-related approaches in the area of Music Information Retrieval such as query-by-humming (QbH) that retrieves a song from voice melody [3], or query-by-beatboxing that retrieves a drum loop from voice timbre sequence [4]. Compared to the first, QbH systems apply a melody transcription of the voice input and the searches for is done in the symbolic domain (usually MIDI) without taking into account timbre information. Compared to the second, which searches for existing drum loops based on timbre and rhythmic similarity, our search unit is not a loop but a micro-segment and the generated sound is not limited to any pre-recorded sound loop. Furthermore, our system uses tonal information of the voice input when retrieving non-percussive sounds, e.g instrumental chords or single notes. Yet another approach [11] uses voice input timbre to train a 3-class classifier during the preparation phase and, in the performance phase, the output of the classifier is used to trigger three different percussive sounds.

Regarding audio mosaicing, it is a concatenative sound synthesis technique that has gained interest in the recent years [5], [6]. Audio or musical mosaicing aims at automatically, with or without user intervention to generate sequences of sound samples by specifying high-level properties of the sequence to generate. These properties are typically translated into constraints holding on acoustic features of the samples. The field also expands to include live, performance-oriented systems. One similar approach is found in *sCrAmBlEd?HaCkZ!* [7], a system that uses a speech input to generate a sequence of similar sounds. The principal differences compared to our system are that in our case the voice input drives also tonal properties (”chord”) of the synthesized output segment. We propose to work on a loop-based synthesis, where tempo and the loop length (number of segments) are modifiable. Also, we can build the vocal target loop by layering several voice input sequentially. e.g beatboxing a drum-line, adding later a bass-line, and adding other sounds on top. Summarizing, the objective of this work is to provide a live input control to an existing audio mosaicing system [10], and in particular addressing the necessary components when using the singing voice as input signal.

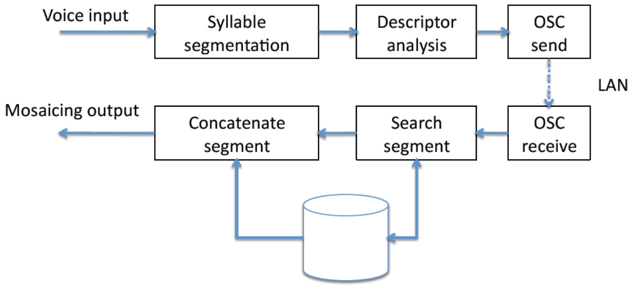


Fig. 1. Block diagram of the proposed system.

## II. System description

### A. Overview

The proposed system is composed of two modules: the interface and the feature-based synthesis engine. Regarding the musical output, the synthesized sound is loop-based, where the audio content resembles the characteristics of the vocal input loop(target), which can be altered on the fly by the user with additional control parameters. As shown in figure 1, the vocal input is segmented into syllables. For each voice segment, a vector of target features is generated, which is used to find the most similar audio segment in the selected audio sources. The corpus of audio sources are song excerpts with a duration of several seconds and which have been segmented using a general onset detector.

### B. Voice description

Voice input has a known tempo and the number of segments that conform the loop is also known. In a first step, the voice signal is segmented using an algorithm specifically designed for instrument imitation signals [2], which performs better than general purpose onset detection algorithms. The latter gives a lot of false positives since it is not adapted to voice signals. Our segmentation algorithm relies on heuristic rules that primarily looks at phonetic variations based on musical functions (attack, sustain, release, etc.).

The second step is to extract acoustic features. In voice-driven instrumental sound synthesis, instantaneous voice features were extracted in short-time frames, capturing the time evolution of pitch, energy, formants and degree of phonation (breathiness). In contrast, in the present approach, we work at a segment level, computing one vector of features per segment of a duration of one beat. Table I lists the acoustic features, including tonal descriptors (HPCP) [8], rhythm (audio centroid) and timbre (MFCC). Actually, the vector of acoustic features for one segment is the mean value of the instantaneous frame values computed using a hop-size of $512$ samples and a window size of $2048$ samples at a sampling rate of $44100$ Hz.

### C. Feature-driven synthesis

If we go back to the general description of audio mosaicing, we can look at the individual steps, and identify four processes: target selection, source selection, unit selection and concatenation. Essentially, the system operates by looping the selected target and concatenating segments from the selected audio sources that best match the target. In our case, the selected target is a vocal signal segmented into syllables. The audio sources consist of song excerpts, which are previously analyzed and segmented using a generic onset detector. Each audio source $i$ consists of a sequence of audio segments (with subindex $j$) from which a vector with the same set of acoustic features $y_i$ is extracted (see table I).

| *Feature* | *Characteristic* |
|---|---|
| audioCentroid | *impulsiveness* |
| energy | loudness |
| flatness | noisiness/harmonicity |
| hpcp | tonal description |
| mfcc | timbre |
| spectralCentroid | brightness |

TABLE I
List of the acoustic features extracted.

The user selects a reduced number of audio sources (usually a dozen of song excerpts or *loops*) that will constitute the corpus. The user is then able to interactively change the selected source material, as well as to interactively augment or diminish the presence of a particular source on the fly. This concept is further described in [10]. The unit selection process retrieves for each target segment a list of similar units in the audio sources using a distance measure. The distance measure compares two vectors (a voice target $x$ and an audio source $y_{ij}$) of acoustic features, where the presence of a given audio source is controlled by applying weights in the distance measure. In the equation 1, $d_{ij}$ is the euclidean distance between target $x$ and source $y_{ij}$, where $w_i$ is the weight of the $i$th audio source, $j$ is the segment index in an audio source $i$, and $M$ is the number of acoustic features.

$$d_{ij} = w_i \sqrt{\sum_{m=1}^{M} (x_m - y_{ij,m})^2} \qquad (1)$$

Finally, the concatenation process includes layering and randomization of source segments simultaneously in order to produce a richer sound.

## III. Prototype implementation

The implemented voice-driven interface analyzes the voice input signal and sends the target features $x$ to the concatenative synthesis engine. In a typical work-flow, the user sets the tempo and the number of steps in the loop buffer. Then, he records the vocal input loop, which is stored in an internal buffer. The user can layer several vocal takes (e.g. imitating drums, bass line), thus creating a richer target sound.

Next, the system analyzes the internal buffer, extracting the acoustic features $x$. Finally, it sends for each target segment its acoustic features in a synchronous way to the synthesis engine. The latter is in charge of retrieving the

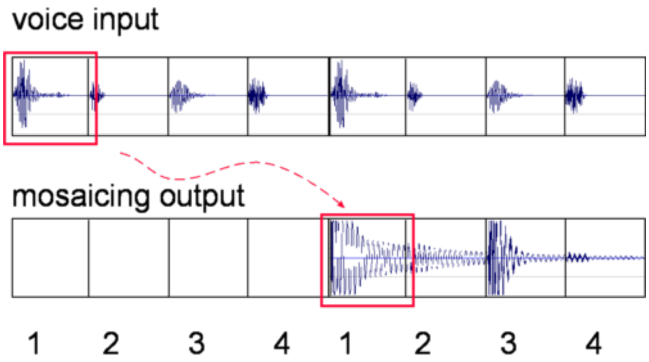
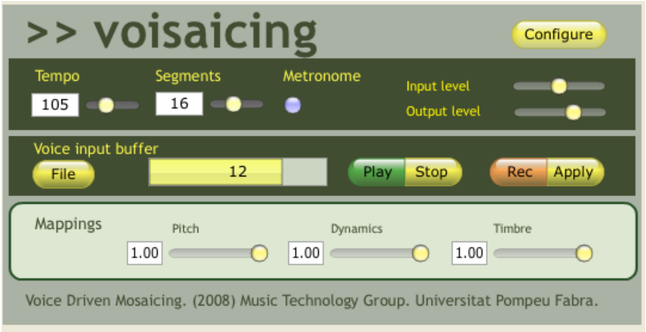


Fig. 2. Timeline of the pseudo-realtime work-flow with a loop length of 4 segments. The user sings a complete voice input loop before the target features are used in the mosaicing output.



Tempo 105
Segments 16
Voice input buffer 12
Pitch 1.00
Dynamics 1.00
Timbre 1.00
Voice Driven Mosaicing. (2008) Music Technology Group. Universitat Pompeu Fabra.

Fig. 3. Screen-shot of the interface module, implemented as a VST plugin.

nearest segment from the audio sources and concatenate the output stream.

Due to implementation considerations, the voice-driven interface and the concatenative synthesis engine are two separate processes that communicate each other through OpenSoundControl[1]. The interface module is a VST plug-in [2] working in pseudo-realtime, i.e. the voice input affects the output with a delay of one loop duration. Figure 2 shows the time evolution of the process, in this case with a loop length of 4 segments.

## IV. Discussion on timbre mapping

First tests with the implemented system shows that the user is able to control the timbre of the synthesis output. However, in order to improve the sense of control, we suggest to study the use of mapping functions from voice features to audio source features for the retrieval.

Intuitively, the sonic spaces of the audio sources and the vocal imitation are different, so that a mapping function might be needed. This mapping function should allow to retrieve any sound in the corpus with a vocal input, thus mapping voice timbre space onto a larger sonic space. In figure 4, we represent the sonic space of vocal sounds (left) and the sonic space of the audio sources corpus (right), which is larger. We propose to learn the mapping functions by imitating a few examples and using statistical methods to derive the mapping functions.

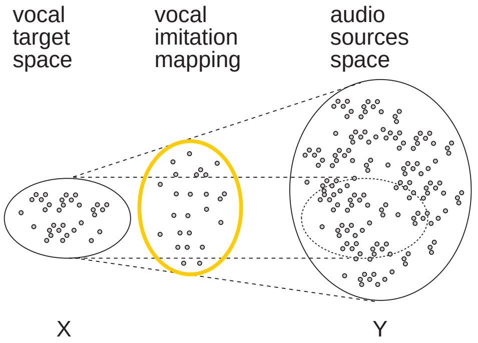


Fig. 4. Sonic spaces. Input voice target space(X), imitation mapping and output sonic space of the audio sources corpus (Y).

### A. *Comparing timbre spaces*

In a preliminary experiment, we collected a corpus of 10 loops (from different musical genres) and the corresponding vocal imitations. The original audio loop and its imitation were segmented and aligned. Each subset consists of 144 short segments. Our goal is to study the timbre space of the voice input compared to the space of audio sources. A priori, the variance in the voice imitation features subset is likely to be lower than in the imitated audio sources subset, since audio source loops will contain any musical sound (including voice), and not inversely. As an initial test, we compute the Principal Component Analysis of both subsets separately using 13 Mel-Frequency Cepstrum Coefficients (MFCC) as data, normalized in a range $[0..1]$ over the complete set (voice imitation and audio sources). MFCC vector data is the mean value of the instantaneous MFCC values within a segment. Then, we project both subsets on the corresponding first two principal components. The variance explained by the first two PCA component are of $63.11\%$ for the voice set, and $66.25\%$ for the audio set. In figure 5, one can observe the projection of the two subsets, audio sources and voice imitation, where each segment corresponds to a *diamond* in the plot. Data is generated using the built-in Matlab function in *princomp*.

One can observe from the plots that the projection of the voice subset is more localized than the audio sources subset. It might indicate that the timbre variance of the voice segments is lower than the variance of the audio sources' timbre. However, we have to stress that this experiment uses a small amount of data, where 10 audio loops were imitated by a single subject.

### B. *Mapping strategies*

In order to derive valid mapping functions, we should collect enough examples of audio loops and vocal imitation by several users. Then, to learn the mapping functions, we can used supervised training methods, where each vocal imitation segment is aligned with its corresponding imitated audio segment. Ideally, by building a sufficient large corpus of imitations for training, one can use statistical methods (e.g. gaussian mixtures) to model both source and target corpus and then find

[1] http://www.opensoundcontrol.org
[2] VST is trademark of Steinberg GmbH.

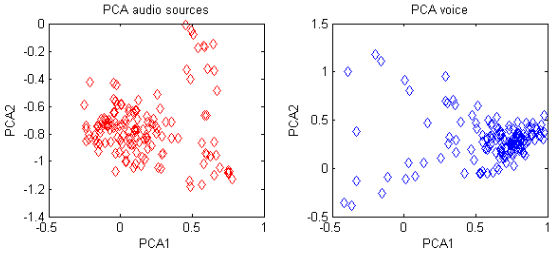


Fig. 5. Principal component analysis (PCA) of audio source (*left*) and voice imitation (*right*). Input data are vectors with 13 MFCC, which are previously normalized and projected on the corresponding first two PCA components.

the corresponding mapping functions using the models. This is similar to approaches found in voice conversion applications [12], where the timbre of the voice source has to be transformed to resemble the timbre of a target voice. Another strategy is to model non-linear mapping functions with neural networks. Alternatively, in cases where the training corpus is small, one possibility is to approximate the mapping functions with a linear regression of a reduced set of examples.

### C. *User-specific mappings*

An additional issue concerning the mapping function is to implement a user-adapted system. We have two options: either to build user-dependent mapping functions, or build a general user-independent functions. In practice, we cannot assume the every individual imitate a sound in the same manner with his voice. Therefore, it seems more convenient to allow the system to learn the mapping functions for every user.

## V. Conclusions

With the proposed system, we provide vocal control capabilities to one the synthesis techniques that has gained more interest in the recent years, audio mosaicing. Compared to voice-driven instrumental sound synthesis, this approach exploits in a higher degree the timbre possibilities of the human voice. At the same time, it offers a more direct way to interact with audio mosaicing, which is usually driven by graphical interfaces. Finally, this research has arisen questions about the mapping strategies between two different sonic spaces: input voice space and output audio source space. Audio examples demonstrating the achieved results can be found online[3].

## Acknowledgment

This research has been partially funded by the IST project SALERO, FP6- 027122. Authors would like to thank to Graham Coleman, Perfecto Herrera and Martin Haro for the discussions.

## References

[1] Jehan, T. and Schoner, B. (2001). ’An audio-driven, spectral analysis-based, perceptually meaningful timbre synthesizer’. In 110th Conv. Audio Eng. Soc., Amsterdam, Netherland.

[2] Janer, J. (2008), ’Singing-driven Interfaces for Sound Synthesziers’, PhD Thesis Universitat Pompeu Fabra.

[3] Lesaffre, M., et al.(2003). ’The MAMI query-by-voice experiment: Collecting and annotating vocal queries for music information retrieval’. In Proceedings of the ISMIR 2003, 4th Int. Conf. on Music Information Retrieval, Baltimore.

[4] Kapur, A., Benning, M., and Tzanetakis, G. (2004). ’Query-by-beat-boxing: Music retrieval for the dj’. In ISMIR-2004.

[5] Zils, A. and Pachet, F. (2001). ’Musical Mosaicing’. In Proc. of the COST-G6 Workshop on Digital Audio Effects (DAFx-01), Limerick.

[6] Schwarz, D. (2005). ’Current Research in Concatenative Sound Synthesis’. Proceedings of the International Computer Music Conference (ICMC).

[7] http://www.popmodernism.org/scrambledhackz/

[8] Gómez, E. (2006). ’Tonal Description of Music Audio Signals’. PhD thesis, Universitat Pompeu Fabra.

[9] Peeters, G. (2003). ’A large set of audio features for sound description (similarity and classification) in the cuidado project’. IRCAM.

[10] Fujishima, T. et al. (2008). ’Music-piece processing apparatus and method’, United States Patent 20080115658, Yamaha Corporation.

[11] Hazan, A. (2005). ’Billaboop real-time voice-driven drum generator’.Proceedings of the Digital Audio Effects Conference DAFX’05, Madrid.

[12] Dutoit, T. et al. (2007). ’Towards a voice conversion system based on frame selection’, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

[3] http://www.mtg.upf.edu/∼jjaner/presentations/smc08