SIMPLE TEMPO MODELS FOR REAL-TIME MUSIC TRACKING

Andreas Arzt
Department of Computational Perception
Johannes Kepler University Linz

Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz
The Austrian Research Institute
for Artificial Intelligence (OFAI)

ABSTRACT

The paper describes a simple but effective method for incorporating automatically learned tempo models into real-time music tracking systems. In particular, instead of training our system with 'rehearsal data' by a particular performer, we provide it with many different interpretations of a given piece, possibly by many different performers. During the tracking process the system continuously recombines this information to come up with an accurate tempo hypothesis. We present this approach in the context of a real-time tracking system that is robust to almost arbitrary deviations from the score (e.g. omissions, forward and backward jumps, unexpected repetitions or re-starts) by the live performer.

1. INTRODUCTION

Real-time audio tracking systems, which listen to a musical performance through a microphone and automatically recognize at any time the current position in the musical score, even if the live performance varies in tempo and sound, promise to be useful in a wide range of applications. They can serve as a (musical) partner to the performer(s) by e.g. automatically accompanying them, interacting with them or supplementing their art by the creation of visualizations of their performance.

In this paper we propose a very simple and general method for incorporating learned tempo models into real-time music trackers. These tempo models need not reflect one specific way of how to perform a piece of music, but rather illustrate many different possible performance strategies (in terms of timing and tempo). We present this approach in the context of a real-time music tracking system that is extremely robust in the face of almost arbitrary structural changes (e.g. disruptions or re-starts) during a live performance.

This unique ability distinguishes our real-time tracking system from the two major advanced score followers that have been developed in recent years. These systems have two quite different domains in mind. While Christopher Raphael's *'Music Plus One'* [1] focuses on the automatic accompaniment of music containing a quite regular pulse, like western classical music, where close synchronization between the solo and the accompanying parts are required, Arshia Cont's system 'Antescofo' [2] addresses a slightly different domain, namely, contemporary music by composers like Boulez, Cage and Stockhausen, with musical characteristics quite different from 'classical' music. During the tracking process both systems are guided by sophisticated tempo models.

In contrast to the above-mentioned systems, which are based on probabilistic models, our music follower uses *Online Dynamic Time Warping (ODTW)* as its basic tracking algorithm (at multiple levels – see Section 3). Even without a predictive model of tempo, this algorithm is surprisingly robust. But for passages with extremely expressive timing, knowledge about plausible performance strategies is needed to improve the precision of real-time alignment. In this paper we will show two simple and very general ways of doing so, the second of which actually permits the system to adapt to different ways of playing without separate training each time.

In the following, we first re-capitulate the basic principles of our approach to on-line music following (Section 2), briefly point to a recent extension that makes the algorithm robust to almost arbitrary disruptions in a performance (Section 3; the details of this are described in a separate paper [3]), and then describe two simple, but effective ways of introducing expressive tempo information into the tracking process in Sections 4 and 5.

2. A HIGHLY ROBUST MUSIC TRACKER

Our approach to score following is via audio-to-audio alignment. That is, rather than trying to transcribe the incoming audio stream into discrete notes and align the transcription to the score, we first convert a MIDI version of the given score into a sound file by using a software synthesizer. The result is a 'machine-like', low-quality rendition of the piece, in which, due to the information stored in the MIDI file, we know the time of every event (e.g. note onsets).

2.1 Data Representation

The score audio stream and the live input stream to be aligned are represented as sequences of analysis frames, computed via a windowed FFT of the signal with a hamming window of size 46ms and a hop size of 20ms. The data is mapped into 84 frequency bins, spread linearly up

to 370Hz and logarithmically above, with semitone spacing. In order to emphasize note onsets, which are the most important indicators of musical timing, only the increase in energy in each bin relative to the previous frame is stored.

2.2 On-line Dynamic Time Warping (ODTW)

This algorithm is the core of our real-time audio tracking system. ODTW takes two time series describing the audio signals – one known completely beforehand (the score) and one coming in in real time (the live performance) –, computes an on-line alignment, and at any time returns the current position in the score. In the following we only give a short intuitive description of this algorithm, for further details we refer the reader to [4].

Dynamic Time Warping (DTW) is an off-line alignment method for two time series based on a local cost measure and an alignment cost matrix computed using dynamic programming, where each cell contains the costs of the optimal alignment up to this cell. After the matrix computation is completed the optimal alignment path is obtained by tracing the dynamic programming recursion backwards (*backward path*).

Originally proposed by Dixon in [4], the ODTW algorithm is based on the standard DTW algorithm, but has two important properties making it useable in real-time systems: the alignment is computed incrementally by always expanding the matrix into the direction (row or column) containing the minimal costs (*forward path)*, and it has linear time and space complexity, as only a fixed number of cells around the forward path is computed.

At any time during the alignment it is also possible to compute a *backward path* starting at the current position, producing an off-line alignment of the two time series which generally is much more accurate. This constantly updated, very accurate alignment of the last couple of seconds will be used heavily throughout this paper. See also Figure 1 for an illustration of the above-mentioned concepts.

Improvements to this algorithm, focusing both on adaptivity and robustness, were presented in [5] and are incorporated in our system, including the 'backward-forward strategy', which reconsiders past decisions (using the backward path) and tries to improve the precision of the current score position hypothesis.

In the following, we will give a short description of a dynamic and general solution to the problem of how to deal with structural changes effectively on-line, and then describe and evaluate our main new contribution: two ways to estimate the current tempo of a performance on-line, and how to use this information to improve the alignment.

3. 'ANY-TIME' REAL-TIME AUDIO TRACKING

In [3] we introduced a unique feature to this system, namely the ability to cope with arbitrary structural deviations from the score during a live performance. At the core is a process that continually updates and evaluates high-level hypotheses about possible current positions in the score, which are then verified or rejected by multiple instances of the basic alignment algorithm described above. To guide our

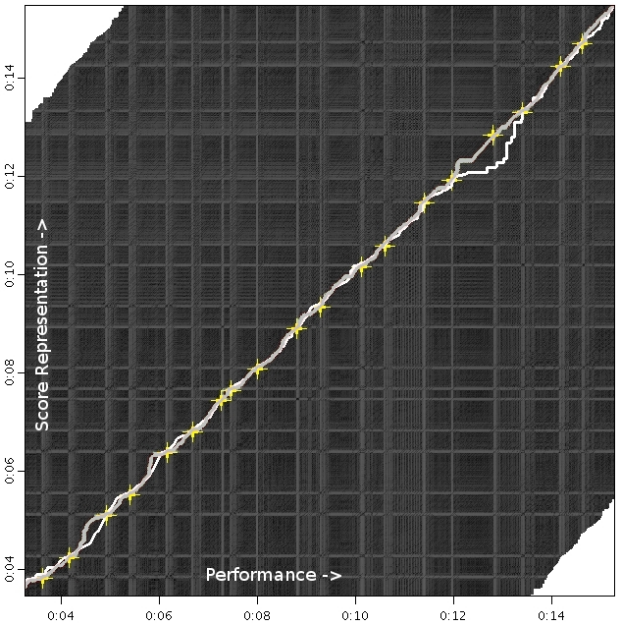

Figure 1. Illustration of the ODTW algorithm, showing the iteratively computed forward path (white), the much more accurate backward path (grey, also catching the one onset that the forward path misaligned), and the correct note onsets (yellow crosses, annotated beforehand). In the background the local alignment costs for all pairs of cells are displayed. Also note the white areas in the upper left and lower right corners, illustrating the constrained path computation around the forward path.

system in the face of possible repetitions and to avoid random jumps between identical parts in the score, we also introduced automatically computed information about the structure of the piece to be tracked. We chose to call our new approach *'Any-time Music Tracking'*, as the system is continuously ready to receive input and find out what the performers are doing, and where they are in the piece.

Figure 2 visually demonstrates the capabilities of our system. In this case 5 different performances of the Prelude in G minor Op. 23 No. 5 by Sergei Rachmaninoff are tracked that start not at the beginning, but 20 bars into the piece. While the basic system finds the correct position after a long timespan (basically by chance), our 'any-time' tracker almost instantly identifies the correct position.

While testing this real-time tracking system with complex piano music played with a lot of expressive freedom in terms of tempo changes, we realized the need for a tempo model to improve the alignment accuracy and the robustness of our system. In the following we propose two simple tempo models, one only based on the analysis of the most recent couple of seconds of the live performance (Section 4) and one having access to automatically extracted additional knowledge about possible future tempo developments (Section 5). The result will be a robust real-time tracker that is able to adapt to and even anticipate tempo changes of the performer, thus leading to a significant increase in alignment precision.

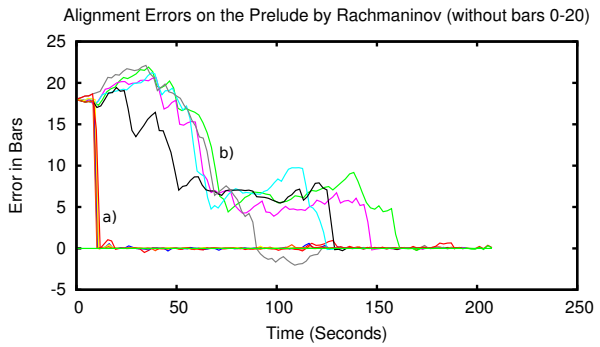

Figure 2. 'Starting in the middle': A visual comparison of the capabilities of the tracker in [5] and the 'any-time' real-time tracking system described in [3]. 5 performances of the g minor Prelude by Rachmaninoff, with bars 0-20 missing, are aligned to the score by both systems. For all performances, the 'any-time' real-time tracker (a) almost instantly identifies the correct position, while the old system (b) finds the correct position by mere chance.

4. A (VERY) SIMPLE TEMPO MODEL

4.1 Computation of the Current Tempo

The computation of the current tempo of the performance (relative to the score representation) is based on a constantly updated backward path starting in the current position of the forward calculation. As the backward path, in contrast to the forward path which has to make its decisions on-line, has perfect information about the performance – at least up to the current position in the performance –, it is much more accurate and reliable than the forward path (see also Figure 1).

Intuitively, the slope of such a backward path represents the relative tempo differences between the score representation and the actual performance. Given a perfect alignment, the slope between the last two onsets would give a very good estimation about the current tempo. But as the correctness of the alignment of these last onsets generally is quite uncertain, one has to discard the last few onsets and use a larger window over more note onsets to come up with a reliable tempo estimation.

In particular, our tempo computation algorithm uses a method described in [6]. It is based on a rectified version of the backward alignment path, where the path between note onsets is discarded and the onsets (known from the score representation) are instead linearly connected. In this way, possible instabilities of the alignment path between onsets (as, e.g., between the 2^{nd} and 3^{rd} onset in the lower left corner in Fig.1) are smoothed away.

After computing this path, the $n = 20$ most recent note onsets which lie at least 1 second in the past are selected, and the local tempo for each onset is computed by considering the slope of the rectified path in a window with size 3 seconds centered on the onset. This results in a vector v_t of length n of relative tempo deviations from the score representation. Finally, an estimate of the current relative tempo t is computed using Eq.1, which emphasizes more recent tempo developments while not discarding older tempo information completely, for robustness considerations.

$$t = \frac{\sum_{i=1}^{n}(t_i * i)}{\sum_{i=1}^{n} i} \quad (1)$$

Of course, due to the simplicity of the procedure and especially the fact that only information older than 1 second is used, this tempo estimation can recognize tempo changes only with some delay. However, the computation is very fast, which is important for real-time applications, and it proved very useful for the task we have in mind.

4.2 Feeding Tempo Information to the ODTW

Based on the observation that both the alignment precision and the robustness directly depend on the similarity between the tempo of the performance and the score representation, we now use the current tempo estimate to alter the score representation on the fly, stretching or compressing it to match the tempo of the performance as closely as possible. This is done by altering the sequence of feature vectors representing the score audio. The relative tempo is directly used as the probability to compress or extend the sequence by either adding new vectors or removing vectors.

More precisely, after every incoming frame from the live performance, and before the actual path computation, the current relative tempo t is computed as given above, where $t = 1$ means that the live performance and the score representation currently are in the exact same tempo and $t > 1$ means that the performance is faster than the score representation. The current position in the score p_s is given by the forward path and thus coincides with the index of the last processed frame of the score representation. If a newly computed random number r between 0 and 1 is larger than t (or $\frac{1}{t}$ if $t > 1$) an alteration step takes place. If $t > 1$, a feature vector is removed from the score representation by replacing p_s+1 and p_s+2 with a mean vector of $p_s + 1$ and $p_s + 2$. And if $t < 1$, a new feature vector, computed as the mean of p_s and p_s+1 is inserted next into the sequence between p_s and p_s+1. As our system is based on features emphasizing note onsets, score feature vectors representing onsets (which are known from the score) are not duplicated, as more (and wrong) onsets would be introduced to the score representation. In such cases the alteration process is postponed until the next frame. Furthermore, to avoid that the system could get stuck at one frame, alterations may take place at most 3 times in a row.

5. 'LEARNING' TEMPO DEVIATIONS FROM DIFFERENT PERFORMERS

As will be shown later in Section 6, the introduction of this very simple tempo model – simply using the current estimated tempo to stretch/compress the reference score audio – already leads to considerably improved tracking results. But especially at phrase boundaries with huge changes in tempo (e.g. a slow-down or a speed-up by a factor of 2 is not uncommon, see also Figure 3) the above-mentioned delay in the recognition of tempo changes still results in large alignment errors. Furthermore, such tempo changes

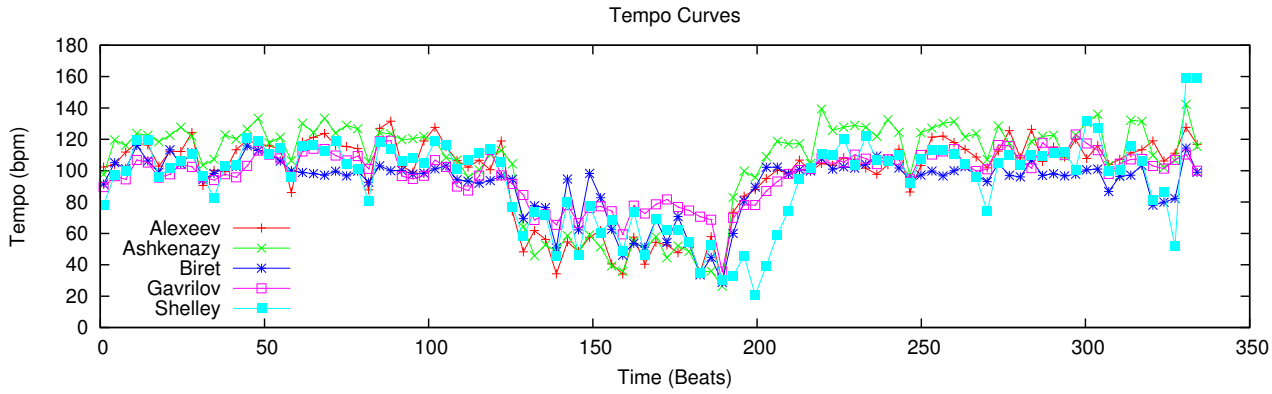

Figure 3. Tempo curves (at the level of quarter notes) automatically extracted from 5 different commercial recordings of the Prelude Op. 23 No. 5 by Rachmaninoff. Note especially the slow-down around beat 130 and the subsequent speed-up around beat 190 and the generally big differences in timing between the performances.

are very hard to catch instantly, even with more reactive tempo models. To cope with this problem we came up with an automatic and very general way to provide the system with information about possible ways in which a performer might shape the tempo of the piece.

First we extract tempo curves from various different performances (audio recordings) of the piece in question. Again, as for the real-time tempo estimation, this is done completely automatically using the method described in [6] (see Section 4.1), but as the whole performance is known beforehand and the tempo analysis can be done off-line there is now no need for further smoothing of the tempo computation. These tempo curves (see Figure 3) are directly imported into our real-time tracking system.

We use this additional information during the tracking process to compute a tempo estimate based not only on tracking information about the last couple of seconds, but also on similarities to other known performances. More precisely, as before, after every iteration of the path computation algorithm the vector v_t containing tempo information at note onsets is updated based on the backward path and the above-mentioned local tempo computation method. But now the tempo curve of the live performance over the last $w = 50$ onsets, again located at least 1 second in the past, is compared to the previously stored tempo curves at the same position. To do this all n tempo curves are first normalized to represent the same mean tempo over these w onsets as the live performance. The Euclidean distances between the curve of the live performance and the stored curves are computed. These distances are inverted and normalized to sum up to 1, thus now representing the similarity to the tempo curve of the live performance.

Based on the stored tempo curves our system can now estimate the tempo at the current position. As the current position should be somewhere between the last aligned onset o_j and the onset o_{j+1} to be aligned next, we compute the current tempo t according to Formula 2, where t_{i,o_j} and $t_{i,o_{j+1}}$ represent the (scaled) tempo information of curve i at onset o_j and o_{j+1} respectively, and s_i is the similarity value of tempo curve i.

$$t = \frac{\sum_{i=1}^{n}[(t_{i,o_j} + t_{i,o_{j+1}})s_i]}{2} \quad (2)$$

Intuitively, the tempo is estimated as the mean of the tempo estimates at these 2 onsets, which in turn are computed as a weighted sum of the (scaled) tempi in the stored performance curves, with each curve contributing according to its local similarity to the current performance. Please note that this approach somewhat differs from typical ways of training a score follower to follow a particular performance. We are not feeding the system with 'rehearsal data' by a particular musician, but with many different ways of how to perform the piece in question, as the analyzed performances may be by different performers and differ heavily in their interpretation style. The system then decides on-line at every iteration how to weigh the curves, effectively selecting a mixture of the curves which represents the current performance best.

6. EVALUATION

The precision of our system was thoroughly tested on various pieces of music (see Table 1), with very well known musicians like Vladimir Horowitz, Vladimir Ashkenazy and Daniel Barenboim amongst the performers. While we currently focus on classical piano music, to show the independence of specific instruments we also tested our system on an oboe sonata by Mozart and the 1^{st} movement of the 5^{th} symphony by Beethoven.

As for the evaluation reference alignments of the performances are needed, Table 1 also indicates how the ground truth data was prepared. For the performance excerpts of the Ballade Op. 38 No. 1 by Chopin (CB) we have access to very accurate data about every note onset ('matchfiles'), as these were recorded on a computer-monitored grand piano. For the performances of the 3 movements of Mozart's Sonata KV279 (MS) the evaluation is based on exact information about every beat time, which was manually compiled. The evaluation of the other pieces is based on off-line alignments produced by our system, which generally are much more precise than on-line alignments. We

ID	Composer	Piece Name	Instruments	# Perf.	Eval. Type
BF	Bach	Fugue BMV847	Piano	7	Offline Align.
BS	Beethoven	5^{th} Symphony, 1^{st} Movement	Orchestra	5	Offline Align.
CB	Chopin	Ballade Op. 38 No. 1 (excerpt)	Piano	22	Match
CW	Chopin	Waltz Op. 34 No. 1	Piano	8	Offline Align.
MO1	Mozart	Oboe Quartet KV370 Mov. 1	Oboe, Violin, Viola, Cello	5	Offline Align.
MO3	Mozart	Oboe Quartet KV370 Mov. 3	Oboe, Violin, Viola, Cello	5	Offline Align.
MS1	Mozart	Sonata KV279 Mov. 1	Piano	5	Beats
MS2	Mozart	Sonata KV279 Mov. 2	Piano	5	Beats
MS3	Mozart	Sonata KV279 Mov. 3	Piano	5	Beats
RP	Rachmaninoff	Prelude Op. 23 No. 5	Piano	5	Offline Align.
SI	Schubert	Impromptu D935 No. 2	Piano	12	Offline Align.

Table 1. The data set used for the evaluation of our real-time tracking system.

are well aware that this information is not guaranteed to be entirely accurate, but we manually checked the alignments for obvious errors and are quite confident that the results based on these alignments are reasonable, especially as evaluations of CB and MS based on these alignments led to very similar numbers compared to the evaluation on the correct reference alignments.

For all pieces we used audio files synthesized from publicly available 'flat' MIDI files with fixed tempo as score representation, only the MIDI representing the Beethoven Symphony contained sparse tempo annotations.

The evaluation took the form of a *cross-validation*. Every performance in our data set (Table 1) was aligned with 3 algorithms: the system introduced in [5] with only minor changes and optimizations; the system including the simple tempo model (Section 4); and the tempo model that has access to a set of possible performance strategies (Section 5). For the latter, all recordings pertaining to the given piece were used except, of course, for the performance currently being aligned. The result, for each performance and each algorithm, is a set of events with detection times in milliseconds.

The evaluation itself was performed as proposed in [7]. For each event i the difference (offset e_i) in milliseconds to the reference alignment is computed. An event i is reported as *missing* if it is aligned with $e_i > 250ms$. This percentage of notes thus misaligned (or, inversely, the percentage of correctly aligned notes) is the main performance measure for a real-time music tracking system. Further statistics, providing information about the alignment precision on those events that were correctly matched, and thus computed on e_i excluding missed events (ec_i), are the *average error*, defined as the mean over the absolute values of all ec_i, the *mean error*, defined as the regular mean without taking the absolute value, and the standard deviation of ec_i. Finally two measures are computed which sum up the overall performance of the system: the *piecewise precision rate* (PP) as the average of the percentage of correctly detected events for each group of performances (see Table 1) and the *overall precision rate* (OP) on the whole database.

Table 2 summarizes the results. Clearly, both tempo models lead to large improvements in tracking accuracy for pieces played with a lot of expressive freedom, especially for the Schubert Impromptu (SI), the Rachmaninov Prelude (RP) and the Chopin Waltz (CW), for which the number of missed notes is more than halved. Nonetheless these kinds of music still pose a great challenge to real-time tracking systems. As the results for the Beethoven Symphony (BS) show, our system can also cope quite well with orchestral music and does not depend on specific instruments. This is also supported by the results on the Oboe Quartet (MO).

As was to be expected, the results for pieces with less extreme tempo deviations were improved to a much smaller extent. Further investigation showed that as intended, the 'learned' tempo curves guided the alignment path more accurately and more reactively during huge tempo changes (i.e., at phrase boundaries).

Unfortunately it is not easy to make comparisons between different approaches in the literature, as the focus on a particular kind of music (e.g. contemporary vs. romantic piano music or monophonic vs. heavily polyphonic music) and the area of application (e.g. automatic accompaniment vs. visualization of music) have a huge influence on the design of the system. That makes it hard to compile a well-balanced ground truth database suitable for all systems.

With this in mind, and the fact that most of our results are currently only computed relative to off-line alignments as ground truth, we merely want to point out some observations. First there is an overlap between our data set and the one used for the evaluation of 'Antescofo' [2], which was already used professionally in a number of live performances. Using the same evaluation metrics, our system performed significantly better (1.9% vs. 9.33% missed notes) on the Fugue by Bach (BF). Of course the result for 'Antescofo' is based on only 1 single performance, which may not even be in our data set. Furthermore, we are quite sure that our system will perform significantly worse than 'Antescofo' on sparse monophonic data, as we do not explicitly detect note onsets and our forward path tends to 'randomly' wander around during long pauses between note onsets. Also, we allow our system to report notes early while 'Antescofo' is purely reactive, thus effectively giving our system twice as large a window to report onsets 'correctly'. While for the task of automatic accompani-

	No Tempo Model				Simple Tempo Model				'Learned' Tempo Model			
	Offset (ms)			%	Offset (ms)			%	Offset (ms)			%
ID	**Avg.**	**Mean**	**STD**	**Miss**	**Avg.**	**Mean**	**STD**	**Miss**	**Avg.**	**Mean**	**STD**	**Miss**
BF	52.1	-15.3	70.4	2.7%	41.7	0.1	61.3	2.2%	41.3	-0.3	59.3	1.9%
BS	84.1	4.3	106.5	15.9%	79.0	-11.6	100.8	15.0%	78.3	-6.4	100.3	13.9%
CB	63.1	16.6	83.7	10.9%	62.4	8.6	83.8	10.0%	63.1	3.9	85.2	9.9%
CW	86.3	-24.6	107.1	27.6%	78.7	-23.2	99.2	16.3%	75.4	-20.2	95.7	11.9%
MO1	94.8	-75.7	89.1	15.0%	70.1	-22.8	90.0	7.0%	72.1	-30.5	89.9	6.9%
MO3	99.9	-84.5	85.3	18.4%	64.3	-18.0	84.0	7.9%	65.7	-16.9	85.8	7.0%
MS1	47.4	13.8	64.5	3.6%	44.9	9.7	62.5	3.3%	42.7	10.1	59.5	3.2%
MS2	85.6	-21.3	104.8	19.8%	71.8	-4.7	93.7	13.8%	73.3	-6.4	94.5	11.3%
MS3	44.1	28.7	58.4	3.9%	40.2	6.7	59.5	3.3%	39.5	9.9	58.5	2.1%
RP	79.8	-18.7	102.0	31.8%	75.5	-10.5	96.8	17.1%	70.9	-10.6	93.2	14.8%
SI	107.3	-59.2	113.9	41.8%	77.9	-32.8	95.2	23.6%	78.7	-33.1	95.7	20.1%
OP	**83.2%**				**89.7%**				**91.1%**			
PP	**81.1%**				**87.9%**				**91.4%**			

Table 2. Real-time alignment results for all 3 evaluated systems (see text).

ment notes reported early are very bothersome, we think that for the task of real-time music visualization, which is our current focus, this is more tolerable.

Unfortunately, we could not find a comparable evaluation of 'Music Plus One' [1], which, like our system, focuses on classical music. However, a number of live demonstrations and available videos suggest that the system works very well in real-time accompaniment settings, not only reacting to tempo changes, but actually predicting them quite well.

That said, our real-time tracking system combines competitive alignment results with a unique feature not found in the above-mentioned systems: the ability to cope with arbitrary jumps of the performer(s) on-line by continuously tracking the performance at a coarser level and refining hypotheses about the current score position (see Section 3). This not only allows to, e.g., automatically cope with arbitrary rehearsal situations, where the musician(s) may keep repeating parts of the piece over and over, but effectively makes it impossible for the system to get lost. (Detailed experimental proof of that can be found in [3].)

7. CONCLUSION AND FUTURE WORK

We have presented a new approach to the incorporation of tempo information into a very robust real-time tracking system that is capable of dealing on-line with almost arbitrary structural deviations from the score. We demonstrated two ways to compute a tempo estimate, one only based on the alignment of the last couple of seconds of the performance, and one additionally based on a collection of previously extracted possible timing patterns, thus giving the system the means to anticipate tempo changes of the performer. The system was evaluated on a range of pieces from Western classical music. Both tempo models lead to significantly improved alignment results, especially for pieces played with a lot of expressive freedom.

An important direction for future work is the introduction of explicit event detection into our system, based on both an estimation of the timing and an analysis of the incoming audio frames. Furthermore we should think about ways to use the extracted tempo information to further improve the high level 'any-time' tracking process (not described in this paper – see [3]).

8. ACKNOWLEDGEMENTS

This research is supported by the City of Linz, the Federal State of Upper Austria, and the Austrian Federal Ministry for Transport, Innovation and Technology, and the Austrian Science Fund (FWF) under project number TRP 109-N23.

9. REFERENCES

[1] C. Raphael, "Current directions with music plus one," in *Proc. of the Sound and Music Computing Conference (SMC)*, (Porto, Portugal), 2009.

[2] A. Cont, "A coupled duration-focused architecture for realtime music to score alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, 2009.

[3] A. Arzt and G. Widmer, "Towards effective 'any-time' music tracking," in *Proc. of the Starting AI Researchers' Symposium (STAIRS 2010), European Conference on Artificial Intelligence (ECAI)*, (Lisbon, Portugal), 2010.

[4] S. Dixon, "An on-line time warping algorithm for tracking musical performances," in *Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, (Edinburgh, Scotland), 2005.

[5] A. Arzt, G. Widmer, and S. Dixon, "Automatic page turning for musicians via real-time machine listening," in *Proc. of the 18th European Conference on Artificial Intelligence (ECAI)*, (Patras, Greece), 2008.

[6] M. Mueller, V. Konz, A. Scharfstein, S. Ewert, and M. Clausen, "Towards automated extraction of tempo parameters from expressive music recordings," in *Proc. of the International Society for Music Information Retrieval Conference (ISMIR)*, (Kobe, Japan), 2009.

[7] A. Cont, D. Schwarz, N. Schnell, and C. Raphael, "Evaluation of real-time audio-to-score alignment," in *Proc. of the 8th International Conference on Music Information Retrieval (ISMIR)*, (Vienna, Austria), 2007.