TOWARDS AUTOMATED EXTRACTION OF TEMPO PARAMETERS FROM EXPRESSIVE MUSIC RECORDINGS

Meinard Müller, Verena Konz, Andi Scharfstein
Saarland University and MPI Informatik
Saarbrücken, Germany
{meinard,vkonz,ascharfs}@mpi-inf.mpg.de

Sebastian Ewert, Michael Clausen
Bonn University, Computer Science
Bonn, Germany
{ewerts,clausen}@iai.uni-bonn.de

ABSTRACT

A performance of a piece of music heavily depends on the musician's or conductor's individual vision and personal interpretation of the given musical score. As basis for the analysis of artistic idiosyncrasies, one requires accurate annotations that reveal the exact timing and intensity of the various note events occurring in the performances. In the case of audio recordings, this annotation is often done manually, which is prohibitive in view of large music collections. In this paper, we present a fully automatic approach for extracting temporal information from a music recording using score-audio synchronization techniques. This information is given in the form of a tempo curve that reveals the relative tempo difference between an actual performance and some reference representation of the underlying musical piece. As shown by our experiments on harmony-based Western music, our approach allows for capturing the overall tempo flow and for certain classes of music even finer expressive tempo nuances.

1. INTRODUCTION

Musicians give a piece of music their personal touch by continuously varying tempo, dynamics, and articulation. Instead of playing mechanically they speed up at some places and slow down at others in order to shape a piece of music. Similarly, they continuously change the sound intensity and stress certain notes. The automated analysis of different interpretations, also referred to as *performance analysis*, has become an active research field [1–4]. Here, one goal is to find commonalities between different interpretations, which allow for the derivation of general performance rules. A kind of orthogonal goal is to capture what is characteristic for the style of a particular musician. Before one can analyze a specific performance, one requires the information about when and how the notes of the underlying piece of music are actually played. Therefore, as the first step of performance analysis, one has to annotate the performance by means of suitable attributes that make explicit the exact timing and intensity of the various note events. The extraction of such performance attributes constitutes a challenging problem, in particular for the case of audio recordings.

Many researchers manually annotate the audio material by marking salient data points in the audio stream. Using novel music analysis interfaces such as the Sonic Visualiser [5], experienced annotators can locate note onsets very accurately even in complex audio material [2, 3]. However, being very labor-intensive, such a manual process is prohibitive in view of large audio collections. Another way to generate accurate annotations is to use a computer-monitored *player piano*. Equipped with optical sensors and electromechanical devices, such pianos allow for recording the key movements along with the acoustic audio data, from which one directly obtains the desired note onset information [3, 4]. The advantage of this approach is that it produces precise annotations, where the symbolic note onsets perfectly align with the physical onset times. The obvious disadvantage is that special-purpose hardware is needed during the recording of the piece. In particular, conventional audio material taken from CD recordings cannot be annotated in this way. Therefore, the most preferable method is to automatically extract the necessary performance aspects directly from a given audio recording. Here, automated approaches such as *beat tracking* [6, 8] and *onset detection* [9] are used to estimate the precise timings of note events within the recording. Even though great research efforts have been directed towards such tasks, the results are still unsatisfactory, in particular for music with weak onsets and strongly varying beat patterns. In practice, semi-automatic approaches are often used, where one first roughly computes beat timings using beat tracking software, which are then adjusted manually to yield precise beat onsets.

In this paper, we present a novel approach towards extracting temporal performance attributes from music recordings in a fully automated fashion. We exploit the fact that for many pieces there exists a kind of "neutral" representation in the form of a musical score (or MIDI file) that explicitly provides the musical onset and pitch information of all occurring note events. Using music synchronization techniques, we temporally align these note events with their corresponding physical occurrences in the music recording. As our main contribution, we describe various algorithms for deriving tempo curves from these align-

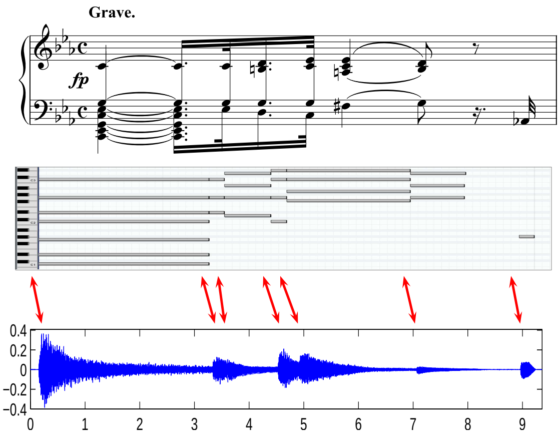

Figure 1. First measure of Beethoven's Pathétique Sonata Op. 13. The MIDI-audio alignment is indicated by the arrows.

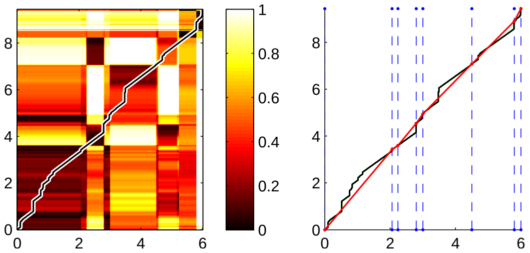

Figure 2. **Left:** Cost matrix and cost-minimizing alignment path for the Beethoven example shown in Fig. 1. The reference representation (MIDI) corresponds to the horizontal and the performance (audio) to the vertical axis. **Right:** Original (black) and onset-rectified alignment path (red). The MIDI note onset positions are indicated by the blue vertical lines.

ments which reveal the relative tempo differences between the actual performance and the neutral reference representation. We have evaluated the quality of the automatically extracted tempo curves on harmony-based Western music of various genres. Besides a manual inspection of a representative selection of real music performances, we have also conducted a quantitative evaluation on synthetic audio material generated from randomly warped MIDI files. Our experiments indicate that our automated methods yield accurate estimations of the overall tempo flow and, for certain classes of music such as piano music, of even finer expressive tempo nuances.

The remainder of this paper is organized as follows. After reviewing some basics on music synchronization (Sect. 2), we introduce various algorithms for extracting tempo curves from expressive music recordings (Sect. 3). Our experiments are described in Sect. 4, and prospects on future work are sketched in Sect. 5. Further related work is discussed in the respective sections.

2. MUSIC SYNCHRONIZATION

The largest part of Western music is based on the equal-tempered scale and can be represented in the form of musical scores, which contain high-level note information such as onset time, pitch, and duration. In the following, we assume that a score is given in the form of a "neutral" MIDI file, where the notes are played with a constant tempo in a purely mechanical way. We refer to this MIDI file as *reference representation* of the underlying piece of music. On the other hand, we assume that the performance to be analyzed is given in the form of an audio recording. In a first step, we use conventional *music synchronization* techniques to temporally align the note events with their corresponding physical occurrences in the audio recording [10, 11]. The synchronization result can be regarded as an automated annotation of the audio recording with the note events given by the MIDI file, see Fig. 1.

Most synchronization algorithms rely on some variant of dynamic time warping (DTW) and can be summarized as follows. First, the MIDI file and the audio recording to be aligned are converted into feature sequences, say $X := (x_1, x_2, \ldots, x_N)$ and $Y := (y_1, y_2, \ldots, y_M)$, respectively. Then, an $N \times M$ cost matrix C is built up by evaluating a local cost measure c for each pair of features, i. e., $C((n,m)) = c(x_n, y_m)$ for $n \in [1:N] := \{1, 2, \ldots, N\}$ and $m \in [1:M]$. Each tuple $p = (n,m)$ is called a *cell* of the matrix. A (global) *alignment path* is a sequence $(p_1, \ldots, p_L)$ of length L with $p_\ell \in [1:N] \times [1:M]$ for $\ell \in [1:L]$ satisfying $p_1 = (1,1)$, $p_L = (N,M)$ and $p_{\ell+1} - p_\ell \in \Sigma$ for $\ell \in [1:L-1]$. Here, $\Sigma = \{(1,0),(0,1),(1,1)\}$ denotes the set of admissible step sizes. The *cost* of a path $(p_1, \ldots, p_L)$ is defined as $\sum_{\ell=1}^{L} C(p_\ell)$. A cost-minimizing alignment path, which constitutes the final synchronization result, can be computed via dynamic programming from C, see Fig. 2. For a detailed account on DTW and music synchronization we refer to [11].

Based on this general strategy, we employ a synchronization algorithm based on high-resolution audio features as described in [12]. This approach, which combines the high temporal accuracy of onset features with the robustness of chroma features, generally yields robust music alignments of high temporal accuracy. In the following, we use a feature resolution of 50 Hz with each feature vector corresponding to 20 milliseconds of MIDI or audio. For details, we refer to [12].

3. COMPUTATION OF TEMPO CURVES

The feeling of pulse and rhythm is one of the central components of music and closely relates to what one generally refers to as tempo. In order to define some notion of tempo, one requires a proper reference to measure against. For example, Western music is often structured in terms of measures and beats, which allows for organizing and sectioning musical events over time. Based on a fixed time signature, one can then define the tempo as the number of beats per minute (BPM). Obviously, this definition requires a regular and steady musical beat or pulse over a certain period in time. Also, the very process of measurement is not as well-defined as one may think. Which musical entities (e. g., note onsets) characterize a pulse? How precisely can these entities be measured before getting drowned in noise? How many pulses or beats are needed to obtain a meaningful tempo estimation? With these questions, we want to indicate that the notion of tempo is far from being well-defined. Different representations of timing and tempo are presented in [7].

In this paper, we assume that we have a reference representation of a piece of music in the form of a MIDI file generated from a score using a fixed global tempo (measured in BPM). Assuming that the time signature of the piece is known, one can recover measure and beat positions from MIDI time positions. Given a specific performance in the form of an audio recording, we first compute a MIDI-audio alignment path as described in Sect. 2. From this path we derive a *tempo curve* that describes for each time position within the MIDI reference (given in seconds or measures) the tempo of the performance (given as a multiplicative factor of the reference tempo or in BPM). Fig. 4 and Fig. 5 show some tempo curves for various performances.

Intuitively, the value of the tempo curve at a certain reference position corresponds to the slope of the alignment path at that position. However, due to discretization and alignment errors, one needs numerically robust procedures to extract the tempo information by using average values over suitable time windows. In the following, we describe three different approaches for computing tempo curves using a fixed window size (Sect. 3.1), an adaptive window size (Sect. 3.2), and a combined approach (Sect. 3.3).

3.1 Fixed Window Size

Recall from Sect. 2 that the alignment path $p = (p_1, \ldots, p_L)$ between the MIDI reference and the performance is computed on the basis of the feature sequences $X = (x_1, \ldots, x_N)$ and $Y = (y_1, \ldots, y_M)$. Note that one can recover beat and measure positions from the indices $n \in [1 : N]$ of the reference feature sequence, since the MIDI representation has constant tempo and the feature rate is assumed to be constant.

To compute the tempo of the performance at a specific reference position $n \in [1 : N]$, we basically proceed as follows. First, we choose a neighborhood of n given by indices n_1 and n_2 with $n_1 \leq n \leq n_2$. Using the alignment path, we compute the indices m_1 and m_2 aligned with n_1 and n_2, respectively. Then, the tempo at n is defined as quotient $\frac{n_2 - n_1 + 1}{m_2 - m_1 + 1}$. The main parameter to be chosen in this procedure is the size of the neighborhood. Furthermore, there are some technical details to be dealt with. Firstly, the boundary cases at the beginning and end of the reference need special care. To avoid boundary problems, we extend the alignment path p to the left and right by setting $p_\ell := (\ell, \ell)$ for $\ell < 1$ and $p_\ell := (N+\ell-L, M+\ell-L)$ for $\ell > L$. Secondly, the indices m_1 and m_2 are in general not uniquely determined. Generally, an alignment path p may assign more than one index $m \in [1 : M]$ to a given index $n \in [1 : N]$. To enforce uniqueness, we chose the minimal index over all possible indices. More precisely, we define a function $\varphi_p : \mathbb{Z} \to [1 : M]$ by setting

$$\varphi_p(n) := \min\{m \in [1 : M] \mid \exists \ell \in \mathbb{Z} : p_\ell = (n, m)\}.$$

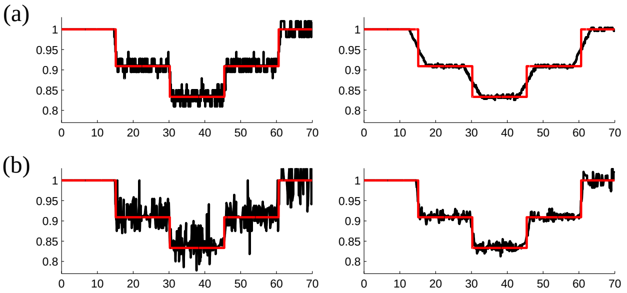

Figure 3. Ground truth tempo curve (step function) and various computed tempo curves. **(a)** τ_w^{FW} using a fixed window size with small w (left) and large w (right). **(b)** τ_v^{AW} using an adaptive window size with small v (left) and large v (right).

We now give the technical details of the sketched procedure for the case that the neighborhoods are of a fixed window (FW) size $w \in \mathbb{N}$. The resulting tempo curve is denoted by $\tau_w^{\mathrm{FW}} : [1 : N] \to \mathbb{R}_{\geq 0}$. For a given alignment path p and an index $n \in [1 : N]$, we define

$$n_1 := n - \left\lfloor \tfrac{w-1}{2} \right\rfloor \quad \text{and} \quad n_2 := n + \left\lceil \tfrac{w-1}{2} \right\rceil. \qquad (1)$$

Then $w = n_2 - n_1 + 1$ and the tempo at reference position n is defined by

$$\tau_w^{\mathrm{FW}}(n) = \frac{w}{\varphi_p(n_2) - \varphi_p(n_1) + 1}. \qquad (2)$$

The tempo curve τ_w^{FW} crucially depends on the window size w. Using a small window allows for capturing sudden tempo changes. However, in this case the tempo curve becomes sensible to inaccuracies in the alignment path and synchronization errors. In contrast, using a larger window smooths out possible inaccuracies, while limiting the ability to accurately pick up local phenomena. This effect is also illustrated by Fig. 3 (a), where the performance is synthesized from a temporally warped MIDI reference. We continue this discussion in Sect. 4.

3.2 Adaptive Window Size

Using a window of fixed size does not account for specific musical properties of the piece of music. We now introduce an approach using an adaptive window size, which is based on the assumption that note onsets are the main source for inducing tempo information. Intuitively, in passages where notes are played in quick succession one may obtain an accurate tempo estimation even when using only a small time window. In contrast, in passages where only few notes are played one needs a much larger window to obtain a meaningful tempo estimation.

We now formalize this idea. We assume that the note onsets of the MIDI reference are given in terms of feature indices. Furthermore, for notes with the same onset position we only list one of these indices. Let $O = \{o_1, \ldots, o_K\} \subseteq [1 : N]$ be the set of onset positions with $1 \leq o_1 < o_2 < \ldots < o_K \leq N$. The distance between two neighboring onset positions is referred to as inter onset interval (IOI). Now, when computing the tempo curve at position $n \in [1 : N]$, the neighborhood of n is specified not in terms of a fixed number w of feature indices but in terms of a fixed number $v \in \mathbb{N}$ of IOIs. This defines an onset-dependent adaptive window (AW). More precisely, let $\tau_v^{\mathrm{AW}} : [1 : N] \rightarrow \mathbb{R}_{\geq 0}$ denote the tempo function to be computed. To avoid boundary problems, we extended the set O to the left and right by setting $o_k := o_1 + k - 1$ for $k < 1$ and $o_k := o_K + k - K$ for $k > K$. First, we compute τ_v^{AW} for all indices n that correspond to onset positions. To this end, let $n = o_k$. Then we define

$$k_1 := k - \left\lfloor \tfrac{v-1}{2} \right\rfloor \quad \text{and} \quad k_2 := k + \left\lceil \tfrac{v-1}{2} \right\rceil.$$

Setting $n_1 := o_{k_1}$ and $n_2 := o_{k_2}$, the tempo at reference position $n = o_k$ is defined as

$$\tau_v^{\mathrm{AW}}(n) := \frac{n_2 - n_1 + 1}{\varphi_p(n_2) - \varphi_p(n_1) + 1}. \qquad (3)$$

Note that, opposed to (2), the window size $n_2 - n_1 + 1$ is no longer fixed but depends on the sizes of the neighboring IOIs around the position $n = o_k$. Finally, $\tau_v^{\mathrm{AW}}(n)$ is defined by a simple linear interpolation for the remaining indices $n \in [1 : N] \setminus O$. Similar to the case of a fixed window size, the tempo curve τ_v^{AW} crucially depends on the number v of IOIs, see Fig. 3 (b). The properties of the various tempo curves are discussed in detail in Sect. 4.

3.3 Combined Strategy

So far, we have introduced two different approaches using on the one hand a fixed window size and on the other hand an onset-dependent adaptive window size for computing average slopes of the alignment path. Combining ideas from both approaches, we now present a third strategy, where we first rectify the alignment path using onset information and then apply the FW-approach on the rectified path for computing the tempo curve. As in Sect. 3.2, let $O = \{o_1, \ldots, o_K\} \subseteq [1 : N]$ be the set of onsets. By possibly extending this set, we may assume that $o_1 = 1$ and $o_K = N$. Now, within each IOI given by two neighboring onsets $n_1 := o_k$ and $n_2 := o_{k+1}$, $k \in [1 : K-1]$, we modify the alignment path p as follows. Let $\ell_1, \ell_2 \in [1 : L]$ be the indices with $p_{\ell_1} = (n_1, \varphi_p(n_1))$ and $p_{\ell_2} = (n_2, \varphi_p(n_2))$, respectively. While keeping the cells p_{ℓ_1} and p_{ℓ_2}, we replace the cells $p_{\ell_1} + 1, \ldots, p_{\ell_2} - 1$ by cells obtained from a suitably sampled linear function having the slope $\frac{n_2 - n_1 + 1}{\varphi_p(n_2) - \varphi_p(n_1) + 1}$. Here, in the sampling, we ensure that the step size condition given by Σ is fulfilled, see Sect. 2. The resulting rectification is illustrated by Fig. 2 (right). Using the rectified alignment path, we then compute the tempo curve using a fixed window size $w \in \mathbb{N}$ as described in Sect. 3.1. The resulting tempo curve is denoted by τ_w^{FWR}. This third approach, as our experiments show, generally yields more robust and accurate tempo estimations than the other two approaches.

4. EXPERIMENTS

In this section, we first discuss some representative examples and then report on a systematic evaluation based on temporally warped music. In the following, we specify the window size w in terms of seconds instead of samples. For example, by writing $w \propto 3$ seconds, we mean that $w \in \mathbb{N}$ is a window size with respect to the feature rate corresponding to 3 seconds of the underlying audio.

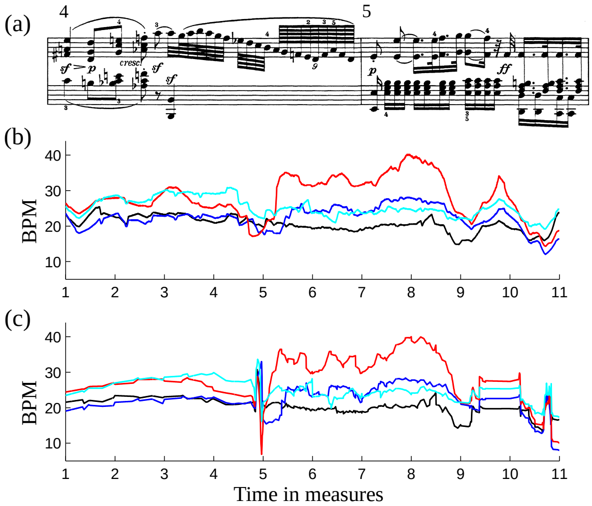

Figure 4. Tempo curves of four different interpretations played by different pianists of the first ten measures (slow introductory theme marked *Grave*) of Beethoven's Pathétique Sonata Op. 13. **(a)** Score of measures 4 and 5. **(b)** Tempo curves τ_w^{FWR} for $w \propto 3$ seconds. **(c)** Tempo curves τ_v^{AW} for $v = 10$ IOIs.

In our first example, we consider Beethoven's Pathétique Sonata Op. 13. The first ten measures correspond to the slow introductory theme marked *Grave*. For these measure, Fig. 4 (b) shows the tempo curves τ_w^{FWR} for four different performances using the combined strategy with a window size $w \propto 3$ seconds. From these curves, one can read off global and local tempo characteristics. For example, the curves reveal the various tempi chosen by the pianists, ranging from roughly 20 to 30 BPM. One of the pianists (red curve) significantly speeds up after measure 5, whereas the other pianists use a more balanced tempo throughout the introduction. It is striking that all four pianists significantly slow down in measure 8, then accelerate in measure 9, before slowing down again in measure 10. Musically, the last slow-down corresponds to the fermata at the end of measure 10, which concludes the *Grave*. Similarly, the curves indicate a ritardando in all four performances towards the end of measure 4. In this passages, there is a run of 64th notes with a closing nonuplet, see Fig. 4 (a). Using a fixed window size, the ritardando effect is smoothed out to a large extent, see Fig. 4 (b). However, having many consecutive note onsets within a short passage, the ritardando becomes much more visible when using tempo curves with an onset-dependent adaptive window size. This is illustrated by Fig. 4 (c), which shows the four tempo curves τ_v^{AW} with $v = 10$ IOIs.

As a second example, we consider the Schubert Lied *Der Lindenbaum* (D. 911 No. 5). The first seven measures (piano introduction) are shown in Fig. 5 (a). Using the combined strategy with a window size $w \propto 3$ seconds, we computed tempo curves for 13 different interpretations, see Fig. 5 (b). As shown by the curves, all interpretations exhibit an accelerando in the first few measures followed

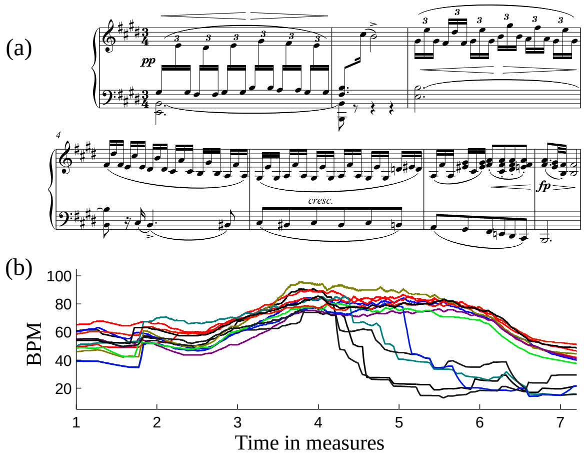

Figure 5. Tempo curves of 13 different performances of the beginning of the Schubert song *Der Lindenbaum*. **(a)** Score of measures 1 to 7. **(b)** Tempo curves τ_w^{FWR} for $w \propto 3$ seconds.

by a ritardando towards the end of the introduction. Interestingly, some of the pianists start with the ritardando in measure 4 already, whereas most of the other pianists play a less pronounced ritardando in measure 6. These examples indicate that our automatically extracted tempo curves are accurate enough for revealing interesting performance characteristics.

In view of a more quantitative evaluation, we computed tempo curves using different approaches and parameters on a corpus of harmony-based Western music of various genres. To allow for a reproduction of our experiments, we used pieces from the RWC music database [13]. In the following, we consider 15 representative pieces, which are listed in Table 1. These pieces include five classical piano pieces, five classical pieces of various instrumentations (full orchestra, strings, flute, voice) as well as five jazz pieces and pop songs. To automatically determine the accuracy of our tempo extraction procedures, we temporally modified MIDI files for each of the 15 pieces. To this end, we generated continuous piecewise linear tempo curves τ^{GT}, referred to as *ground-truth tempo curves*. These curves have a constant slope on segments of roughly 10 seconds of duration, where the slopes are randomly generated either using a value $v \in [1 : 2]$ (corresponding to an accelerando) or using a value $v \in [1/2 : 1]$ (corresponding to a ritardando). These values cover a range of tempo changes of $\pm 100\%$ of the reference tempo. Intuitively, the ground-truth tempo curves simulate on each segment a gradual transition between two tempi to mimic ritardandi and accelerandi. For an example, we refer to Fig. 6. We then temporally warped each of the original MIDI files with respect to a ground-truth tempo curve τ^{GT} and generated from the modified MIDI file an audio version using a high-quality synthesizer. Finally, we computed tempo curves using the original MIDI files as reference and the warped audio versions as performances.

To determine the accuracy of a computed tempo curve τ, we compared it with the corresponding ground-truth tempo curve τ^{GT}. Here, the idea is to measure deviations by *scale* rather than by *absolute value*. Therefore, as distance function, we use the average multiplicative difference and standard deviation (both measured in percent) of τ and τ^{GT}. More precisely, we define

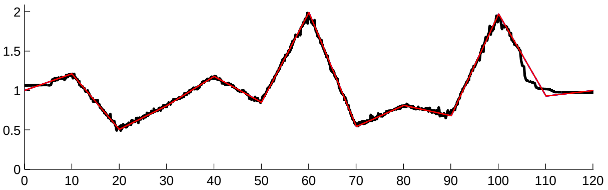

Figure 6. Piecewise linear ground-truth tempo curve (red) and computed tempo curves (black).

RWC ID (Comp./Int., Instr.)	**FW**		**AW**		**FWR**	
	μ	σ	μ	σ	μ	σ
C025 (Bach, piano)	3.29	7.30	2.60	5.05	1.59	2.86
C028 (Beethoven, piano)	3.24	6.98	6.36	21.14	2.66	6.72
C031 (Chopin, piano)	3.32	7.72	2.77	4.76	1.75	3.42
C032 (Chopin, piano)	2.54	4.17	3.05	4.67	1.56	2.34
C029 (Schumann, piano)	4.52	8.86	4.18	5.97	2.44	5.13
C003 (Beethoven, orchestra)	4.20	5.39	10.58	22.97	3.56	4.79
C015 (Borodin, strings)	2.44	2.85	4.68	9.85	2.25	2.71
C022 (Brahms, orchestra)	1.70	1.95	2.41	2.96	1.31	1.66
C044 (Rimski-K., flute/piano)	1.62	2.59	2.47	4.27	1.61	2.58
C048 (Schubert, voice/piano)	2.61	3.27	3.95	7.76	2.07	2.98
J001 (Nakamura, piano)	1.44	1.87	1.44	2.43	1.03	1.59
J038 (HH Band, big band)	2.24	2.96	3.20	5.41	1.91	2.74
J041 (Umitsuki, sax/bass/perc.)	1.88	2.40	3.75	4.69	1.72	2.34
P031 (Nagayama, electronic)	2.01	2.42	8.35	14.89	1.94	2.39
P093 (Burke, voice/guitar)	2.50	3.26	6.21	14.74	2.34	3.13
Average over all	2.64	4.27	4.40	8.77	1.98	3.16

Table 1. Tempo curve evaluation using the approaches FW and FWR (with $w \propto 4$ seconds) and AW (with $v = 10$ IOIs). The table shows for each of the 15 pieces the mean error μ and standard deviation σ (given in percent) of the computed tempo curves and the ground truth tempo curve. For generating the ground-truth tempo curves, MIDI segments of 10 seconds were used.

$$\mu(\tau, \tau^{\mathrm{GT}}) = 100 \cdot \frac{1}{N} \cdot \sum_{n=1}^{N} \left(2^{|\log_2(\tau(n)/\tau^{\mathrm{GT}}(n))|} - 1\right).$$

Similarly, we define the standard deviation $\sigma(\tau, \tau^{\mathrm{GT}})$. For example, one obtains $\mu(\tau, \tau^{\mathrm{GT}}) = 100\%$ in the case $\tau = 2 \cdot \tau^{\mathrm{GT}}$ (double tempo) and in the case $\tau = \frac{1}{2} \cdot \tau^{\mathrm{GT}}$ (half tempo). Similarly, a computed tempo of 110 BPM or 90.9 BPM would imply a mean error of $\mu = 10\%$ assuming a ground-truth tempo of 100 BPM.

In a first experiment, we computed the curves τ_w^{FW} and τ_w^{FWR} with $w \propto 4$ seconds as well as τ_v^{AW} with $v = 10$ IOIs for each of the 15 pieces. Table 1 shows the mean error μ and standard deviation σ between the computed tempo curves and the ground truth tempo curves. For example, for the Schubert song *Der Lindenbaum* with identifier **C048**, the mean error between the computed tempo curve τ_w^{FW} and the ground-truth tempo τ^{GT} amounts to 2.61%. This error decreases to 2.07% when using the FWR-approach based on the rectified alignment path. Looking at the average mean error over all pieces, one can notice that the error amounts to 2.64% for the FW-approach, 4.40% for the AW-approach, and 1.98% for the FWR-approach. For example, assuming a tempo of 100 BPM, the last number implies a mean difference of less than 2 BPM between the computed tempo and the actual tempo.

In general, the FWR-approach yields the best tempo es-

w [sec]	FW		FWR		v [IOI]	AW	
	μ	σ	μ	σ		μ	σ
1	10.62	49.88	5.58	12.47	2	14.50	31.00
2	5.37	14.21	3.58	6.16	4	9.54	23.44
3	4.39	6.90	3.42	5.34	6	7.34	17.34
4	4.62	6.52	3.99	5.74	8	6.18	12.99
5	5.48	7.08	5.06	6.63	10	5.65	10.66
6	6.79	8.02	6.52	7.74	12	5.46	9.48
7	8.40	9.19	8.22	9.00	16	5.54	8.20
8	10.15	10.51	10.03	10.38	20	5.98	8.09

Table 2. Tempo curve evaluation using the approaches FW, AW, and FWR with various window sizes w (given in seconds) and v (given in IOIs). The table shows the average values over all 15 pieces, see Table 1. For generating the ground-truth tempo curves, MIDI segments of 5 seconds were used.

timation, whereas the AW-approach often produces poorer results. Even though the onset information is of crucial importance for estimating local tempo nuances, the AW-approach relies on accurate alignment paths that correctly align the note onsets. Synchronization approaches as described in [12] can produce highly accurate alignments in the case of music with pronounced note attacks. For example, this is the case for piano music. In contrast, such information is often missing in string or general orchestral music. This is the reason why the purely onset-based AW-strategy yields a relatively poor tempo estimation with a mean error of 10.58% for Beethoven's Fifth Symphony (identifier **C003**). On the other hand, using a fixed window size without relying on onset information, local alignment errors cancel each other out, which results in better tempo estimations. E. g., the error drops to 3.56% for Beethoven's Fifth Symphony when using the FWR-approach.

Finally, we investigated the dependency of the accuracy of the tempo estimation on the window size. We generated strongly fluctuating ground-truth tempo curves using MIDI segments of only 5 seconds length (instead of 10 seconds as in the last experiment). For the corresponding synthesized audio files, we computed tempo curves for various window sizes. The mean errors averaged over all 15 pieces are shown in Table 2. The numbers show that the mean error is minimized when using medium-sized windows. E. g., in the FWR-approach, the smallest error of 3.42% is attained for a window size of $w \propto 3$ seconds. Actually, the window size constitutes a trade-off between robustness and temporal resolution. On the one hand, using a larger window, possible alignment errors cancel each other out, thus resulting in a gain of robustness. On the other hand, sudden tempo changes and fine agogic nuances can be recovered more accurately when using a smaller window.

5. CONCLUSIONS

In this paper, we have introduced automated methods for extracting tempo curves from expressive music recordings by comparing the performances with neutral reference representations. In particular when using a combined strategy that incorporates note onset information, we obtain accurate and robust estimations of the overall tempo progression. Here, the window size constitutes a delicate trade-off between susceptibility to alignment errors and sensibility towards timing nuances of the performance. In practice, it becomes a difficult problem to determine whether a given change in the tempo curve is due to an alignment error or whether it is the result of an actual tempo change in the performance. Here, one idea for future work is to use tempo curves as a means for revealing problematic passages in the music representations where synchronization errors may have occurred with high probability. Furthermore, it is of crucial importance to further improve the temporal accuracy of synchronization strategies. This constitutes a challenging research problem in particular for music with less pronounced onset information, smooth note transitions, and rhythmic fluctuation.

Acknowledgements: The first three authors are supported by the Cluster of Excellence on *Multimodal Computing and Interaction* at Saarland University. The last author is funded by the German Research Foundation (DFG CL 64/6-1).

6. REFERENCES

[1] J. Langner and W. Goebl, “Visualizing expressive performance in tempo-loudness space,” *Computer Music Journal*, vol. 27(4), pp. 69–83, 2003.

[2] C. S. Sapp, “Comparative analysis of multiple musical performances,” in *ISMIR Proceedings*, pp. 497–500, 2007.

[3] G. Widmer, “Machine discoveries: A few simple, robust local expression principles,” *Journal of New Music Research*, vol. 31(1), pp. 37–50, 2002.

[4] G. Widmer, S. Dixon, W. Goebl, E. Pampalk, and A. Tobudic, “In search of the Horowitz factor,” *AI Magazine*, vol. 24(3), pp. 111–130, 2003.

[5] Sonic Visualiser. Retrieved 19.03.2009, `http://www.sonicvisualiser.org/`.

[6] S. Dixon, “Automatic extraction of tempo and beat from expressive performances,” *Journal of New Music Research*, vol. 30, pp. 39–58, 2001.

[7] H. Honing, “From Time to Time: The Representation of Timing and Tempo,” *Computer Music Journal*, vol. 25(3), pp. 50–61, 2001.

[8] E. D. Scheirer, “Tempo and beat analysis of acoustical musical signals,” *Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998.

[9] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, “A Tutorial on Onset Detection in Music Signals,” *IEEE Trans. on Speech and Audio Proc.*, vol. 13, no. 5, pp. 1035–1047, 2005.

[10] N. Hu, R. Dannenberg, and G. Tzanetakis, “Polyphonic audio matching and alignment for music retrieval,” in *Proc. IEEE WASPAA, New Paltz, NY*, October 2003.

[11] M. Müller, *Information Retrieval for Music and Motion*. Springer, 2007.

[12] S. Ewert, M. Müller, and P. Grosche, “High resolution audio synchronization using chroma onset features,” in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, (Taipei, Taiwan), 2009.

[13] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, “RWC music database: Popular, classical and jazz music databases,” in *ISMIR*, 2002.