An Efficient Multiscale Approach to Audio Synchronization

Meinard Müller **Henning Mattes** **Frank Kurth**
Department of Computer Science, University of Bonn
Römerstraße 164, 53117 Bonn, Germany
meinard@cs.uni-bonn.de, henning1000@gmx.de, frank@cs.uni-bonn.de

Abstract

We present an efficient and robust multiscale DTW (MsDTW) approach to music synchronization for time-aligning CD recordings of different interpretations of the same piece. The general strategy is to recursively project an alignment path computed at a coarse resolution level to the next higher level and then to refine the projected path. As main contributions, we address several crucial issues including the design and specification of robust and scalable audio features, suitable local cost measures, MsDTW levels, constraint regions, as well as sampling rate adaptation and structural enhancement strategies. Extensive experiments on Western classical music show that our MsDTW-based algorithm yields the same alignment result as the classical DTW-based strategy while significantly reducing the running time and memory requirements. Even for pieces of a duration of 10 to 15 minutes, the alignment (based on previously extracted feature sequences) can be computed in less than a second.

Keywords: audio synchronization, alignment, multiscale, chroma feature

1. Introduction

For one and the same piece of music, there often exists a large number of CD recordings representing different interpretations by various musicians. In particular for Western classical music, these interpretations may exhibit considerable deviations in tempo, note realizations, dynamics, and instrumentation. For example, a music database may contain for Beethoven's Fifth Symphony some interpretations by Karajan and Bernstein, some historical recordings by Furthwängler and Toscanini, Liszt's piano transcription of Beethoven's Fifth played by Sherbakov and Glenn Gould, or some synthesized version of a corresponding MIDI file. Then, the goal of *audio synchronization* is to automatically generate *alignments* between corresponding note events between different interpretations. These alignments can be used to jump freely between different audio recordings, thus affording efficient and convenient music browsing.

In the last few years, several alignment strategies have been proposed, see, e. g., [1, 2, 3, 4, 5, 6] and the references therein. Most of these approaches rely on some variant of dynamic time warping (DTW). However, due to the quadratic time and space complexity, DTW-based strategies become infeasible for long pieces. To reduce the computational cost, Salvador et al. [7] propose for general time series a multiscale DTW (MsDTW) approach that recursively projects an alignment path computed at a coarse resolution level (using coarse features, e.g., obtained by averaging and downsampling) to the next higher level and then refines the projected path. One hazard with this approach is that an incorrect alignment on a low resolution level propagates to higher levels resulting in erroneous alignment results. This hazard is fostered by the fact that coarsening the features can lead to heavily deteriorated cost matrices, as is also illustrated by Fig. 4 (a)-(c). Dixon et al. [1] describe a linear time DTW approach based on forward path estimation. Further related work will be discussed in the respective sections.

In this paper, we describe an MsDTW approach to efficient as well as robust audio synchronization. In Sect. 2, the general ideas of MsDTW are summarized. In Sect. 3, we then propose solutions to several crucial issues of MsDTW including the design of robust and scalable audio features, specification of suitable local cost measures, adaptation strategies for the feature sampling rate, determination of MsDTW resolution levels and constraint regions, as well as enhancement strategies of DTW cost matrices. We then report on our extensive experiments based on a wide range of classical music demonstrating the practicability of our algorithm. Furthermore, the synchronization results have been integrated in our audio player [8] and sonified for evaluation, see Sect. 4. For some audio demos we refer to www-mmdb.iai.uni-bonn.de/projects/MsDTWsync.

2. General Multiscale Approach

In this section, we summarize the main ideas of classical DTW and MsDTW. Let $X := (x_1, x_2, \ldots, x_N)$ and $Y := (y_1, y_2, \ldots, y_M)$ be two feature sequences with $x_n, y_m \in \mathcal{F}$, $n \in [1 : N]$, $m \in [1 : M]$, where $\mathcal{F}$ denotes a suitable feature space. Furthermore, let $c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ denote a *local cost measure* on $\mathcal{F}$. The resulting $(N \times M)$-cost matrix C is defined by $C(n, m) := c(x_n, y_m)$.

2.1. Classical DTW

DTW is a well-known technique to align X and Y with respect to the cost measure c. Recall that a *warping path* is a sequence $p = (p_1, \ldots, p_L)$ with $p_\ell = (n_\ell, m_\ell) \in [1 : N] \times [1 : M]$ for $\ell \in [1 : L]$ satisfying $1 = n_1 \leq n_2 \leq \ldots \leq n_L = N$ and $1 = m_1 \leq m_2 \leq \ldots \leq m_L = M$ (boundary and monotonicity condition), as well as $p_{\ell+1} - p_\ell \in \{(1,0), (0,1), (1,1)\}$ (step size condition). The total cost of p is defined as $\sum_{\ell=1}^{L} c(x_{n_\ell}, y_{m_\ell})$. Then, an optimal warping path between X and Y is given by a warping path p^* having minimal total cost among all possible warping paths. To determine such an optimal path, one recursively computes an $(N \times M)$-matrix D, where the matrix entry $D(n, m)$ is the total cost of an optimal path between $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_m)$. In the following, we denote a matrix entry $D(n, m)$ as *cell*. Introducing an additional weight vector $(w_d, w_h, w_v) \in \mathbb{R}^3$ yields the recursive definition

$$D(n,m) := \min \begin{cases} D(n-1, m-1) + w_d \cdot c(x_n, y_m), \\ D(n-1, m) + w_h \cdot c(x_n, y_m), \\ D(n, m-1) + w_v \cdot c(x_n, y_m), \end{cases}$$

for $n, m > 1$. Furthermore, $D(n, 1) := \sum_{k=1}^{n} w_h \cdot c(x_k, y_1)$ for $n > 1$, $D(1, m) = \sum_{k=1}^{m} w_v \cdot c(x_1, y_k)$ for $m > 1$, and $D(1, 1) := c(x_1, y_1)$. Note that in the unweighted case $(w_d, w_h, w_v) = (1, 1, 1)$ one has a preference of the diagonal alignment direction, since one diagonal step (cost of one cell) corresponds to the combination of one horizontal and one vertical step (cost of two cells). To counterbalance this preference, we chose $(w_d, w_h, w_v) = (2, 1.5, 1.5)$ (still slightly favoring the diagonal direction). Now, p^* can be derived from D in some linear fashion, see [9] for details. The overall computational cost is proportional to the number of cells $D(n, m)$ to be computed during this process leading to a time and space complexity of $O(NM)$.

2.2. Multiscale DTW (MsDTW)

To speed up DTW computations, one common strategy is to impose a global *constraint region* $R \subseteq [1 : N] \times [1 : M]$ on the possible warping paths (e. g., Itakura parallelogram, Sakoe-Chiba band), thus limiting the number of cells needed to be computed, see [9]. However, the usage of global constraint regions is problematic, since the optimal warping path may leave the specified region. In other words, let p_R^* denote the optimal warping path with respect to R, then p_R^* may differ from p^*, see Fig. 1. A second strategy is to reduce the feature sampling rate (also referred to as dimensionality reduction or data abstraction), thus reducing the lengths N and M of the sequences to be synchronized. However, the resulting optimal warping path becomes increasingly inaccurate or even completely useless as the resolution decreases, see Fig. 4 for an illustration. MsDTW employs these two strategies in some iterative fashion to generate data-dependent constraint regions. We summarize the main ideas and refer to [7] for details. For a similar approach, applied to melody alignment we refer to [10].

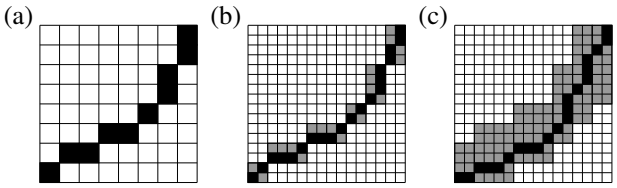

Figure 1. (a) Optimal warping path p_2^* on Level 2. (b) Optimal warping path p_R^* with respect to the constraint region R obtained by projecting path p_2^* to Level 1. (Here, p_R^* does not coincide with the (unconstrained) optimal warping path p^*.) (c) Optimal warping path $p_{R^\delta}^*$ using an increased constraint region $R^\delta \supset R$ with $\delta = 2$. Here, $p_{R^\delta}^* = p^*$.

Let $X_1 := X$ and $Y_1 := Y$ be the sequences to be synchronized having lengths $N_1 := N$ and $M_1 = M$, respectively. It is the objective to compute an optimal warping path p^* between X_1 and Y_1. The highest resolution level will also referred to as Level 1. By reducing the feature sampling rate by a factor $f_2 \in \mathbb{N}$, one obtains sequences X_2 of length $N_2 := N_1/f_2$ and Y_2 of length $M_2 := M_1/f_2$. (Here, we assume that f_2 divides N_1 and M_1, which can be achieved, e. g., by suitably padding X_1 and Y_1.) Next, one computes an optimal warping path p_2^* between X_2 and Y_2 on the resulting resolution level (Level 2). This path is projected onto Level 1 and there defines a constraint region R. Note that R consists of $L_2 \times f_2^2$ cells, where L_2 denotes the length of p_2^*. Finally, an optimal alignment path p_R^* relative to R is computed. We say that this procedure is *successful*, if $p^* = p_R^*$. The overall number of cells to be computed in this procedure is $N_2 M_2 + L_2 \cdot f_2^2$, which is generally much smaller than the total number $N_1 M_1$ of cells on Level 1. In an obvious fashion, this procedure can be recursively applied by introducing further levels of decreasing resolution. For a complexity analysis, we refer to [7].

The constraint path p_R^* may not coincide with the optimal path p^*. To alleviate this problem, one can increase the constraint region R—at the expense of efficiency—by adding δ cells to the left, right, top, and bottom of every cell in R for some parameter $\delta \in \mathbb{N}$. The resulting region R^δ will be referred to as δ-neighborhood of R, see Fig. 1.

3. MsDTW Audio Synchronization

The multiscale approach to DTW constitutes a general framework to speed up computations. In view of robustness, efficiency, and practicability of the resulting synchronization procedure, however, one has to specify several important parameters. In this section, we describe and discuss the design choices for our audio synchronization algorithm and report on our experiments. To avoid incorrect alignments even in extreme situations we sketch a strategy for structural enhancement of the cost matrix, which works even at very low resolution levels, see Sect. 3.5.

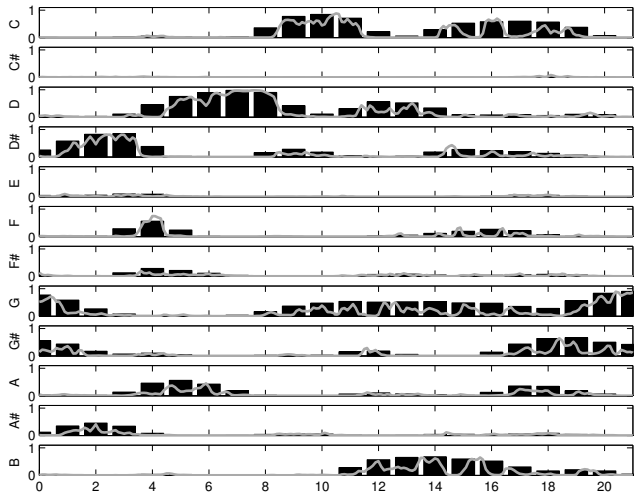

Figure 2. The first 21 seconds of Bernstein's interpretation of Beethoven's Fifth Symphony. The light curves represent the local chroma energy distributions (10 features per second). The dark bars represent the CENS features (1 feature per second).

3.1. Audio Features

Previous studies have shown that chroma-based audio features are well suited to characterize harmony-based music, see [11, 2, 12]. Representing the spectral energy of each of the 12 traditional pitch classes of the equal-tempered scale, chroma features do not only account for the close octave relationship in both melody and harmony as it is prominent in Western music, but also introduce a high degree of robustness to variations in timbre and articulation [11]. Since different interpretations typically exhibit significant variations in such parameters, chroma-based features are well-suited for audio synchronization leading to robust and accurate alignments, see [2].

There are various ways to compute chroma features, e. g., by suitably pooling spectral coefficients obtained from some short-time Fourier transform [11] or by suitably summing up pitch subbands obtained as output after applying some pitch-based filter bank [12]. For details, we refer to the literature. In our implementation, we convert an audio signal into a sequence $X = (x_1, x_2, \ldots, x_N)$ of normalized 12-dimensional feature vectors $x_n \in [0,1]^{12}$, $1 \leq n \leq N$, which express the local energy distribution in the 12 chroma classes. Here, we use a feature sampling rate of 10 Hz, where each feature vector x_n covers 200 ms of audio with an overlap of 100 ms. This rate, which will constitute the finest resolution level, turns out to be sufficient in view of our intended applications.

For our multiscale approach, we need a computationally inexpensive way to adjust the feature resolution. Instead of simply modifying the analysis window in the chroma computation, we introduce a second, much larger statistics window (covering w consecutive chroma vectors) and consider *short-time statistics* of the chroma energy distribution over this window. This again results in a sequence of 12-dimensional vectors, which is then downsampled by a factor of q and renormalized with respect to the Euclidean norm (hence being invariant towards variations in dynamics). For example, $w = 41$ and $q = 10$ yields a feature sampling rate of 1 Hz, where each feature vector represents information of the audio signal within a window of 4200 ms. The resulting feature sequence will be referred to as CENS(w, q) sequence (**C**hroma **E**nergy **N**ormalized **S**tatistics). Similar features have been applied in the audio matching scenario and are described in detail in [12]. Note that by modifying the parameters w and q, one can adjust the feature granularity and sampling rate without repeating the cost-intensive chroma computations. Fig. 2 shows the resulting chroma and CENS$(41, 10)$ features sequences extracted from a Bernstein interpretation of Beethoven's Fifth.

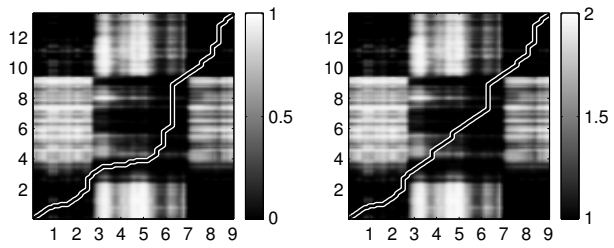

Figure 3. Optimal warping path based on the cost measure c_α with $\alpha = 0$ (left) and $\alpha = 1$ (right).

3.2. Local Cost Measure

The normalized chroma and CENS features are elements in $\mathcal{F} := [0,1]^{12}$. To compare two features $x, y \in \mathcal{F}$, we use the cost measure $c_\alpha : \mathcal{F} \times \mathcal{F} \to [0,1] + \alpha$ defined by $c_\alpha(x, y) := 1 - \langle x, y \rangle + \alpha$ for some offset $\alpha \in \mathbb{R}_{\geq 0}$. (Note that $\langle x, y \rangle$ is the cosine of the angle between x and y, since x and y are normalized.) The offset α is introduced for the following reason. Audio recordings often contain long segments of little variance such as pauses or sustained chords. This leads to rectangular regions in the cost matrix, also referred to as *plateaus*. If $\alpha = 0$, all cells within a plateau reveal some cost close to zero. Being close to zero, there may be large, more or less random *relative* differences among the costs of these cells (e. g., one cell has cost 0.01 and another one 0.001). As a result, one obtains an uncontrollable run of the optimal warping path within a plateau. By increasing α, the relative differences decrease, whereas the absolute differences are retained unchanged (e. g., for $\alpha = 1$, one cell has cost 1.01 and another one 1.001). As a consequence, the effect of the weight vector (w_d, w_h, w_v) as introduced in Sect. 2.1 becomes more dominant. For the parameters $(w_d, w_h, w_v) = (2, 1.5, 1.5)$ and $\alpha = 1$, which have turned out to be suitable in our experiments, the diagonal direction receives a slight but stable preference in plateau regions, see Fig. 3. In the following, we set $c := c_1$.

Table 1. Specification of the features used in our MsDTW audio synchronization system.

Level	Feature	Resolution	Factor
1	Chroma	10 Hz	-
2	CENS$(41, 10)$	1 Hz	10
3	CENS$(121, 30)$	$1/3$ Hz	3
4	CENS$(271, 90)$	$1/9$ Hz	3

Table 2. Total running time (for 363 synchronization pairs) against the number of levels used in the MsDTW algorithm based on the features of Table 1.

Run time \ Levels	1	$1-2$	$1-3$	$1-4$
t_{Cells} [s]	1434.0	78.5	67.6	67.2
t_{CENS} [s]	0.0	16.5	24.7	29.8
t_{MsDTW} (absolute) [s]	1434.0	95.0	92.3	97.0
t_{MsDTW} (relative) [%]	100	6.62	6.44	6.76

3.3. Resolution Levels and δ-Neighborhood

As described in Sect. 3.1, the chroma features at a time resolution of 10 Hz constitute the basic resolution (Level 1) of our audio synchronization. The CENS features, which can be efficiently derived from the chroma features, are used at the lower resolution levels. To determine a suitable number of levels as well as the resolutions at each level used in the MsDTW, we conducted comprehensive experiments. We report on some experiments that are based on the features indicated by Table 1. For our tests, we used 363 pairs of CD recordings, where each pair corresponds to two different interpretations of the same piece. The recordings have a duration of 3 to 20 minutes and cover a wide range of classical music, see Table 5 for examples. In a preprocessing step, we computed and stored the chroma features of all recordings. For each test series, we performed an audio synchronization for all 363 pairs. The algorithms have been implemented in C/C++ and tests were run on an Intel Pentium M, 1.7 GHz, 1 GByte RAM, under Windows XP.

In one test series, we used classical DTW on Level 1 and MsDTW based on the first two, three, and four levels. In all these cases we used $\delta = 30$, see the discussion below. The resulting running times are shown in Table 2. The DTW-based strategy required 1434.0 seconds to synchronize all of the 363 pairs. In contrast, the two-level MsDTW-based strategy required $t_{\text{CENS}} = 16.5$ seconds to compute the CENS$(41, 10)$-features used for Level 2, and $t_{\text{Cells}} = 78.5$ seconds to compute all required cells on the two levels, amounting to a total running time of $t_{\text{MsDTW}} = 95.0$ seconds—only 6.62% of the running time of classical DTW. Similarly, it took 92.3 and 97.0 seconds when using the three-level and four-level MsDTW-based strategy, respectively. In particular, we obtained the lowest total running time for three levels. Using a fourth level indeed further decreased the total number of cells to be evaluated, but the computational overhead due to the additional CENS features needed at Level 4 deteriorated the overall result. Also introducing additional intermediate levels had only a marginal effect on the overall running time. We therefore use three multiscale levels as default setting in our audio synchronization system.

Table 3. Running time, absolute and relative error against the size of the δ-neighborhood and adaptive neighborhood based on a three-level MsDTW.

Run time \ δ	0	10	20	30
t_{Cells} [s]	29.3	41.6	54.6	67.6
t_{CENS} [s]	25.0	24.7	24.6	24.7
t_{MsDTW} (absolute) [s]	54.3	66.3	79.2	92.3
t_{MsDTW} (relative) [%]	3.79	4.62	5.52	6.44
Error (absolute)	92	27	6	0
Error (relative) [%]	25.34	7.44	1.65	0

For our evaluation, we use the DTW-alignment (unconstrained warping path) as ground truth and check whether the MsDTW-alignment (constrained warping path) entirely coincides with the DTW-alignment (then MsDTW is called successful, see Sect. 2.2) or not (then MsDTW produces an error). In another test series, we evaluated different δ-neighborhoods for a three-level MsDTW. Table 3 shows the performance for $\delta = 0, 10, 20, 30$. For $\delta = 0$, in 92 of the 363 cases the MsDTW-based strategy was not successful (corresponding to an error rate of 25.34%). Increasing δ leads to an increase of the running time and a decrease of the error rate. For $\delta = 30$, all 363 pairs have been successfully aligned by the MsDTW strategy. The running time to evaluate the cells has increased from $t_{\text{Cells}} = 29.3$ seconds ($\delta = 0$) to $t_{\text{Cells}} = 67.6$ seconds ($\delta = 30$)— a moderate increase with regard to the total running time. For our audio synchronization system, we therefore use $\delta = 30$ as default.

3.4. Experimental Results

In this section, we discuss some representative experimental results based on the three-level MsDTW with $\delta = 30$. For these parameters, as was reported above, the audio synchronization has been successful for all of the 363 pairs of audio recordings. Table 5 shows a selection of these recordings including complex orchestral pieces having a duration of 3 to 20 minutes. Some of the interpretations significantly differ in tempo, instrumentation, and articulation. For example, Sacchi's interpretation of Schubert's Unfinished is much faster (817 seconds) than Solti's interpretation (951 seconds). Or, there is an orchestral as well as a piano version (piano transcription) of Beethoven's Fifth and Wagner's Prelude, respectively. Furthermore, the Mae interpretation of Vivaldi's spring includes many additional ornamentations, which can not be found in the Zukerman interpretation. In view of such significant variations, the CENS$(41, 10)$ features as used on Level 2 constitute a good

Table 4. Performance of the implementation of our MsDTW-based audio synchronization algorithm for a representative selection of recordings using three-levels and $\delta = 30$. For each level, the total number of cells (DTW) as well as the number of cells to be evaluated by MsDTW are indicated. The last three columns show a comparison of the DTW and MsDTW running times.

Synchronization				Number of cells to be evaluated by DTW and MsDTW in each level							Total run time[s]		
Recording 1		Recording 2		Level 1			Level 2			Level 3	Levels 1-3		
Id 1	Length [s]	Id 2	Length [s]	DTW	MsDTW	[%]	DTW	MsDTW	[%]	DTW	DTW	MsDTW	[%]
Beet9Bern	1144.9	Beet9Kar	1054.8	120808050	2117929	1.75	1209030	17657	1.46	134464	31.18	1.08	3.46
RavBolAbb	862.5	RavBolOza	901.0	77737897	1694610	2.18	778426	14121	1.81	86688	20.04	0.80	3.99
Schub8Sac	817.3	Schub8Sol	950.8	77736075	1704150	2.19	777918	14322	1.84	86541	20.01	0.80	3.99
Dvo9Maaz	704.2	Dvo9Franc	710.7	50068752	1356308	2.71	501255	11295	2.25	55695	12.85	0.60	4.67
WagPreArm	595.0	WagPreGould	576.9	34337270	1124029	3.27	343892	9387	2.73	38407	8.85	0.48	5.42
Beet5Bern	519.0	BeLi5Sher	444.1	23068056	923444	4.00	231400	7721	3.34	25926	5.84	0.37	6.34
Beet5Bern	519.0	Beet5Kar	443.9	23052480	923028	4.00	230880	7716	3.34	25752	5.87	0.38	6.47
SchosJazzCha	223.6	SchosJazzYab	193.6	4335306	394259	9.09	43456	3291	7.57	4875	1.11	0.15	13.51
VivSpringMae	192.5	VivSpringZuk	218.9	4217940	389034	9.22	42267	3261	7.72	4745	1.07	0.13	12.15
ElgEnigDel	91.8	ElgEnigSino	93.1	856508	167699	19.58	8648	1404	16.23	992	0.22	0.06	27.27

Table 5. Some audio recordings (with identifier) contained in our test database comprising 33 hours of audio.

Composer / piece / interpreter	Identifier
Beethoven / Symph. 5, Op. 67, 1st mov. / Bernstein	Beet5Bern
Beethoven / Symph. 5, Op. 67, 1st mov. / Karajan	Beet5Kar
Beeth. (Liszt) / Symph. 5, Op. 67, 1st mov. (piano) / Sherbakov	BeLi5Sher
Beethoven / Symph. 9, Op. 125, 4th mov. / Bernstein	Beet9Bern
Beethoven / Symph. 9, Op. 125, 4th mov. / Karajan	Beet9Kar
Dvorak / Symph. 9, Op. 95, 1st mov. / Francis	Dvo9Franc
Dvorak / Symph. 9, Op. 95, 1st mov. / Maazel	Dvo9Maaz
Elgar / Op. 36, Andante (Enigma) / Del Mar	ElgEnigDel
Elgar / Op. 36, Andante (Enigma) / Sinopoli	ElgEnigSino
Ravel / Bolero / Abbado	RavBolAbb
Ravel / Bolero / Ozawa	RavBolOza
Schubert / Symph. 8, D759, 1st mov. (Unfinished) / Sacchi	Schub8Sac
Schubert / Symph. 8, D759, 1st mov. (Unfinished)/ Solti	Schub8Sol
Shostakovich / Jazz Suite No. 2, 6th mov. (Waltz) / Chailly	SchosJazzCha
Shostakovich / Jazz Suite No. 2, 6th mov. (Waltz) / Yablonsky	SchosJazzYab
Vivaldi / RV 269 (Spring), 1st mov. / Mae	VivSpringMae
Vivaldi / RV 269 (Spring), 1st mov. / Zukerman	VivSpringZuk
Wagner / Meistersinger Prelude / Armstrong	WagPreArm
Wagner / Meistersinger Prelude (piano) / Gould	WagPreGould

compromise between reasonable feature resolution, computational efficiency and robustness of the alignment result. (Actually, in situations where one has significant differences in note realizations—e. g., one interpreter plays a sustained chord whereas another interpreter plays an ornamentation—the alignment on a finer resolution level may not even be semantically meaningful.)

The increase in performance of our MsDTW-based audio synchronization in comparison to a classical DTW-based approach is illustrated by Table 4. For example, to synchronize 'Beet9Bern' and 'Beet9Kar' only 1.75% of the cells on Level 1 have to be evaluated by MsDTW, decreasing the memory requirements roughly by a factor of 57. Here, note that the overall memory requirement is proportional to the maximal number of cells needed to be evaluated at some level. The number of additional cells needed to be computed at Levels 2 and 3 is comparatively small. The overall running time (including the CENS feature computation) to synchronize the two recordings (each version having a duration of almost 20 minutes) was 1.08 seconds—nearly thirty times faster than classical DTW. Obviously, the relative savings increase with the lengths of the pieces.

3.5. Enhancing Cost Matrices

We have also conducted experiments with manually distorted and highly repetitive audio material, which constitutes an extreme scenario for MsDTW. As one example, we used the first 22 seconds of an Cabrera interpretation of Bach's Toccata BWV 565, where the theme is repeated three times at three different octaves. We generated an audio file, referred to as 'Bach12', by concatenating four copies of this segment, resulting in 12 repetitions of the theme. We then generated a time-warped version of 'Bach12', referred to as 'Bach12Warp', by locally increasing and decreasing the tempo up to 50 percent. The cost matrix using $\mathrm{CENS}(41, 10)$ and the resulting optimal warping between 'Bach12' and 'Bach12Warp' is shown in Fig. 4 (a). Now, reducing the feature sampling rate leads to a heavily deteriorated cost matrix, where the repetitions cannot be resolved any longer. This, in turn, results in absurd optimal warping paths on the lower resolution levels, see Fig. 4 (b) and (c).

To alleviate this problem, we improve the structural properties of the cost matrix by incorporating contextual information into the local cost measure, see [13]. Intuitively, the idea is to enhance the diagonal path structure of the 2D cost matrix by applying a local 1D low-pass filter along the diagonals (having gradient $(1, 1)$). Note that this process only works if corresponding segments of the two audio recordings reveal the same tempo progression. To account for tempo differences in the two interpretations, the idea is to simultaneously filter along different directions (in our implementation we used eight different gradients in a neighborhood of $(1, 1)$, which cover tempo variations of roughly -30 to $+40$ percent) and then to take the minimum over the filter outputs. The technical details of this approach are described in [13]. The effect of this enhancement strategy, which allows to reduce the feature sampling rate without completely destroying the structural properties of the cost

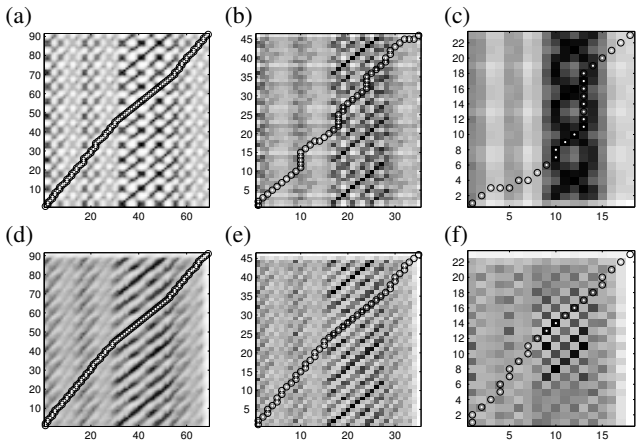

Figure 4. Optimal warping path between 'Bach12' and 'Bach12Warp' using CENS$(41, 10)$ (a), CENS$(81, 20)$ (b), and CENS$(161, 40)$ (c). The paths are semantically incorrect for the two lower resolution levels. Locally filtering the cost matrix based on CENS$(41, 10)$ (using eight different gradients) and downsampling by a factor 1 (d), 2 (e), and 4 (f) leads to correct optimal warping paths even on the lower resolution levels.

matrix, is illustrated by Fig. 4 (d)-(f). Even at the lowest resolution level (obtained by filtering with an averaging filter of length 8 and downsampling by a factor of 4), the optimal warping path leads to the 'correct' alignment. In practice, one has to assess the trade-off between increased computational complexity caused by the additional filtering step and the boost of robustness and confidence due to the structural enhancement.

4. Conclusions and Future Work

In this paper, we presented an efficient MsDTW-based approach to audio synchronization yielding stable alignments even in the presence of significant variations. One main application of audio synchronization is to allow for efficient music browsing. To this end, we integrated the alignment results into the SyncPlayer framwork [8] (an audio player with additional retrieval and browsing functionality). During playback of a CD recording, SyncPlayer allows the user to directly jump to the corresponding position of any other interpretation of the same piece. In view of this application, an alignment at a resolution of 10 Hz is more than sufficient. To evaluate the absolute alignment quality achieved by our system, we conducted the following experiment. Based on the alignment result of two recordings, we time-warped the second recording to run synchronously to the first recording. For the warping, we used an overlap-add technique based on waveform similarity (WSOLA) as described in [14]. We then produced a stereo audio file containing the mono version of the original first recordings in one channel and a mono version of the time-warped second recording in the other channel. Listening to this stereo audio file exhibits even small temporal deviations of less than 100 ms between corresponding note events. Some representative audio examples can be found at `www-mmdb.iai.uni-bonn.de/projects/MsDTWsync`. Based on the synchronization result, it is possible to continuously blend from one interpretation to another one. It would be an interesting task to employ synchronization techniques for mixing and morphing different interpretations. As another task, based on our chroma-based alignment, one can apply more refined techniques and features to further enhance the alignment. Finally, we plan to develop strategies to automatically detect critical audio segments (e. g., segments leading to plateaus), where one can then locally switch between various synchronization strategies (e. g., locally activating and deactivating the enhancement strategy).

References

[1] Simon Dixon and Gerhard Widmer. Match: A music alignment tool chest. In *Proc. ISMIR, London, GB*, 2005.

[2] Ning Hu, Roger Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proc. IEEE WASPAA, New Paltz, NY*, October 2003.

[3] Meinard Müller, Frank Kurth, and Tido Röder. Towards an efficient algorithm for automatic score-to-audio synchronization. In *Proc. ISMIR, Barcelona, Spain*, 2004.

[4] Christopher Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proc. ISMIR, Barcelona, Spain*, 2004.

[5] Ferréol Soulez, Xavier Rodet, and Diemo Schwarz. Improving polyphonic and poly-instrumental music to score alignment. In *Proc. ISMIR, Baltimore, USA*, 2003.

[6] Robert J. Turetsky and Daniel P.W. Ellis. Force-Aligning MIDI Syntheses for Polyphonic Music Transcription Generation. In *Proc. ISMIR, Baltimore, USA*, 2003.

[7] S. Salvador and P. Chan. FastDTW: Toward accurate dynamic time warping in linear time and space. In *Proc. KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[8] Frank Kurth, Meinard Müller, David Damm, Christian Fremerey, Andreas Ribbrock, and Michael Clausen. Syncplayer - an advanced system for content-based audio access. In *Proc. ISMIR, London, GB*, 2005.

[9] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals Of Speech Recognition*. Prentice Hall, 1993.

[10] Norman Adams, Daniela Marquez, and Gregory H. Wakefield. Iterative deepening for melody alignment and retrieval. In *Proc. ISMIR, London, GB*, 2005.

[11] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Trans. on Multimedia*, 7(1):96–104, 2005.

[12] Meinard Müller, Frank Kurth, and Michael Clausen. Audio Matching via Chroma-based Statistical Features. In *Proc. ISMIR, London, GB*, 2005.

[13] Meinard Müller and Frank Kurth. Enhancing similarity matrices for music audio analysis. In *Proc. ICASSP*, 2006.

[14] W. Verhelst and M. Roelands. An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *Proc. ICASSP*, 1993.