# Refinement Strategies for Music Synchronization

Sebastian Ewert[1] and Meinard Müller[2]

Universität Bonn, Institut für Informatik III
Römerstr. 164, 53117 Bonn, Germany
ewerts@cs.uni-bonn.de
Max-Planck-Institut für Informatik
Campus E1-4, 66123 Saarbrücken, Germany
meinard@mpi-inf.mpg.de

**Abstract.** For a single musical work, there often exists a large number of relevant digital documents including various audio recordings, MIDI files, or digitized sheet music. The general goal of music synchronization is to automatically align the multiple information sources related to a given musical work. In computing such alignments, one typically has to face a delicate tradeoff between robustness, accuracy, and efficiency. In this paper, we introduce various refinement strategies for music synchronization. First, we introduce novel audio features that combine the temporal accuracy of onset features with the robustness of chroma features. Then, we show how these features can be used within an efficient and robust multiscale synchronization framework. In addition we introduce an interpolation method for further increasing the temporal resolution. Finally, we report on our experiments based on polyphonic Western music demonstrating the respective improvements of the proposed refinement strategies.

## 1 Introduction

Modern information society is experiencing an explosion of digital content, comprising text, audio, image, and video. For example, in the music domain, there is an increasing number of relevant digital documents even for a single musical work. These documents may comprise various audio recordings, MIDI files, digitized sheet music, or symbolic score representations. The field of music information retrieval (MIR) aims at developing techniques and tools for organizing, understanding, and searching multimodal information in a robust, efficient and intelligent manner. In this context, various alignment and synchronization procedures have been proposed with the common goal to automatically link several types of music representations, thus coordinating the multiple information sources related to a given musical work [1, 3–6, 9, 12, 13, 15–21].

In general terms, *music synchronization* denotes a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. Depending upon the respective data formats, one distinguishes between various synchronization tasks [1, 13]. For example, *audio-audio* synchronization [5, 17, 20] refers to the task of time aligning two different audio recordings of a piece of music. These alignments can be used to jump freely between different interpretations, thus affording efficient and convenient audio browsing. The goal of *score-audio* and *MIDI-audio* synchronization [1, 3, 16, 18, 19] is to coordinate note and MIDI events with audio data. The result can be regarded as an automated annotation of the audio recording with available score and MIDI data. A recently studied problem is referred to as *scan-audio* synchronization [12], where the objective is to link regions (given as pixel coordinates) within the scanned images of given sheet music to semantically corresponding physical time positions within an audio recording. Such linking structures can be used to highlight the current position in the scanned score during playback of the recording. Similarly, the goal of *lyrics-audio* synchronization [6, 15, 21] is to align given lyrics to an audio recording of the underlying song. For an overview of related alignment and synchronization problems, we also refer to [4, 13].

Automated music synchronization constitutes a challenging research field since one has to account for a multitude of aspects such as the data format, the genre, the instrumentation, or differences in parameters such as tempo, articulation and dynamics that result from expressiveness in performances. In the design of synchronization algorithms, one has to deal with a delicate tradeoff between robustness, temporal resolution, alignment quality, and computational complexity. For example, music synchronization strategies based on chroma features [3] have turned out to yield robust alignment results even in the presence of significant artistic variations. Such chroma-based approaches typically yield a reasonable synchronization quality, which suffices for music browsing and retrieval applications. However, the alignment accuracy may not suffice to capture fine nuances in tempo and articulation as needed in applications such as performance analysis [22] or audio editing [3]. Other synchronization strategies yield a higher accuracy for certain classes of music by incorporating onset information [16, 19], but suffer from a high computational complexity and a lack of robustness. Dixon et al. [5] describe an online approach to audio synchronization. Even though the proposed algorithm is very efficient, the risk of missing the optimal alignment path is relatively high. Müller et al. [17] present a more robust, but very efficient offline approach, which is based on a multiscale strategy.

In this paper, we introduce several strategies on various conceptual levels to increase the time resolution and quality of the synchronization result without sacrificing robustness and efficiency. First, we introduce a new class of audio features that inherit the robustness from chroma-based features and the temporal accuracy from onset-based features (Sect. 2). Then, in Sect. 3, we show how these features can be used within an efficient and robust multiscale synchronization framework. Finally, for further improving the alignment quality, we introduce an interpolation technique that refines the given alignment path in some time consistent way (Sect. 4). We have conducted various experiments based on polyphonic Western music. In Sect. 5, we summarize and discuss the results indicating the respective improvements of the proposed refinement strategies. We conclude in Sect. 6 with a discussion of open problems and prospects on future work. Further references will be given in the respective sections.

## 2 Robust and Accurate Audio Features

In this section, we introduce a new class of so-called DLNCO (decaying locally adaptive normalized chroma-based onset) features that indicate note onsets along with their chroma affiliation. These features posses a high temporal accuracy, yet being robust to variations in timbre and dynamics. In Sects. 2.1 and 2.2, we summarize the necessary background on chroma and onset features, respectively. The novel DLNCO features are then described in Sect. 2.3.

### 2.1 Chroma Features

In order to synchronize different music representations, one needs to find suitable feature representations being robust towards those variations that are to be left unconsidered in the comparison. In this context, chroma-based features have turned out to be a powerful tool for synchronizing harmony-based music, see [2, 9, 13]. Here, the chroma refer to the 12 traditional pitch classes of the equal-tempered scale encoded by the attributes $\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots,$B. Note that in the equal-tempered scale, different pitch spellings such $\mathrm{C}^\sharp$ and $\mathrm{D}^\flat$ refer to the same chroma. Representing the short-time energy of the signal in each of the 12 pitch classes, 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 [2]. Furthermore, normalizing the features makes them invariant to dynamic variations. There are various ways to compute chroma features, e. g., by suitably pooling spectral coefficients obtained from a short-time Fourier transform [2] or by suitably summing up pitch subbands obtained as output after applying a pitch-based filter bank [13, 14]. For details, we refer to the literature.

In the following, the first six measures of the Etude No. 2, Op. 100, by Friedrich Burgmüller will serve us as our running example, see Fig. 1a. For short, we will use the identifier **Burg2** to denote this piece, see Table 1. Figs. 1b and 1c show a chroma representation and a normalized chroma representation, respectively, of an audio recording of **Burg2**. Because of their invariance, chroma-based features are well-suited for music synchronization leading to robust alignments even in the presence of significant variations between different versions of a musical work, see [9, 17].

### 2.2 Onset Features

We now describe a class of highly expressive audio features that indicate note onsets along with their respective pitch affiliation. For details, we refer to [13, 16]. Note that for many instruments such as the piano or the guitar, there is sudden energy increase when playing a note (attack phase). This energy increase may

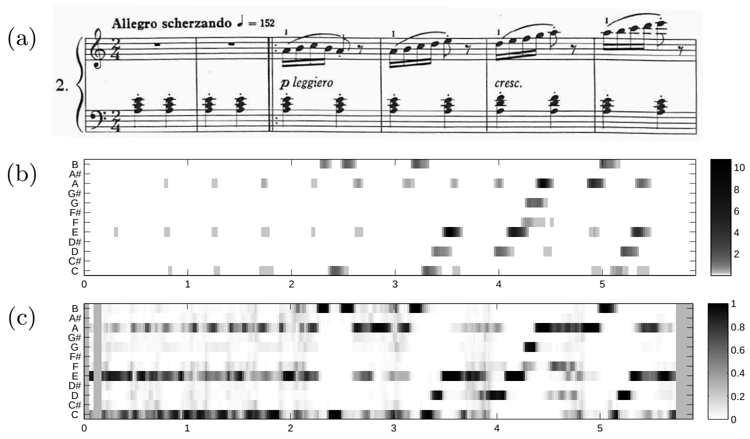

**Fig. 1. (a)** First six measures of Burgmüller, Op. 100, Etude No. 2 (**Burg2**, see Table 1). **(b)** Chroma representation of a corresponding audio recording. Here, the feature resolution is 50 Hz (20 ms per feature vector). **(c)** Normalized chroma representation.

not be significant relative to the entire signal's energy, since the generated sound may be masked by the remaining components of the signal. However, the energy increase relative to the spectral bands corresponding to the fundamental pitch and harmonics of the respective note may still be substantial. This observation motivates the following feature extraction procedure.

First the audio signal is decomposed into 88 subbands corresponding to the musical notes A0 to C8 (MIDI pitches $p = 21$ to $p = 108$) of the equal-tempered scale. This can be done by a high-quality multirate filter bank that properly separates adjacent notes, see [13, 16]. Then, 88 local energy curves are computed by convolving each of the squared subbands with a suitably window function. Finally, for each energy curve the first-order difference is calculated (discrete derivative) and half-wave rectified (positive part of the function remains). The significant peaks of the resulting curves indicate positions of significant energy increase in the respective pitch subband. An onset feature is specified by the pitch of its subband and by the time position and height of the corresponding peak.

Fig. 2 shows the resulting onset representation obtained for our running example **Burg2**. Note that the set of onset features is sparse while providing information of very high temporal accuracy. (In our implementation, we have a pitch dependent resolution of $2-10$ ms.) On the downside, the extraction of onset features is a delicate problem involving fragile operations such as differentiation and peak picking. Furthermore, the feature extraction only makes sense for music

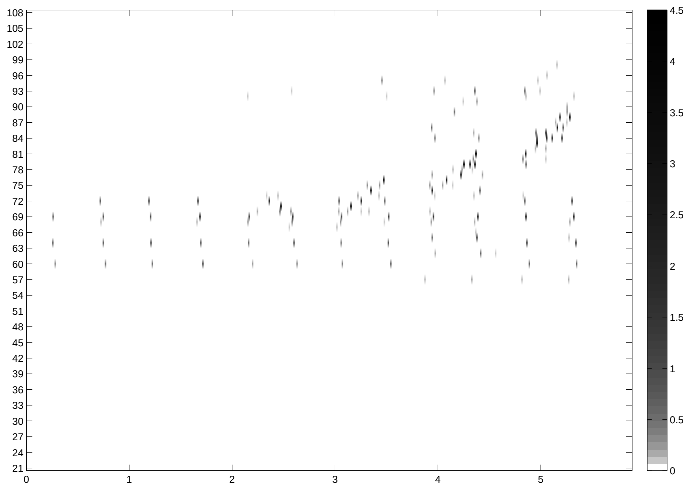

**Fig. 2.** Onset representation of **Burg2**. Each rectangle represents an onset feature specified by pitch (here, indicated by the MIDI note numbers given by the vertical axis), by time position (given in seconds by the horizontal axis), and by a color-coded value that correspond to the height of the peak. Here, for the sake of visibility, a suitable logarithm of the value is shown.

with clear onsets (e. g., piano music) and may yield no or faulty results for other music (e. g., soft violin music).

## 2.3 DLNCO Features

We now introduce a new class of features that combine the robustness of chroma features and the accuracy of onset features. The basic idea is to add up those onset features that belong to pitches of the same pitch class. To make this work, we first evenly split up the time axis into segments or frames of fixed length (In our experiments, we use a length of 20 ms). Then, for each pitch, we add up all onset features that lie within a segment. Note that due to the sparseness of the onset features, most segments do not contain an onset feature. Since the values of the onset features across different pitches may differ significantly, we take a suitable logarithm of the values, which accounts for the logarithmic sensation of sound intensity. For example, in our experiments, we use $\log(5000 \cdot v + 1)$ for an onset value $v$. Finally, for each segment, we add up the logarithmic values over all pitches that correspond to the same chroma. For example, adding up the logarithmic onset values that belong to the pitches A0,A1,. . .,A7 yields a value for the chroma A. The resulting 12-dimensional features will be referred to as *CO (chroma onset) features*, see Fig. 3a.

The CO features are still very sensitive to local dynamic variations. As a consequence, onsets in passages played in piano may be marginal in comparison

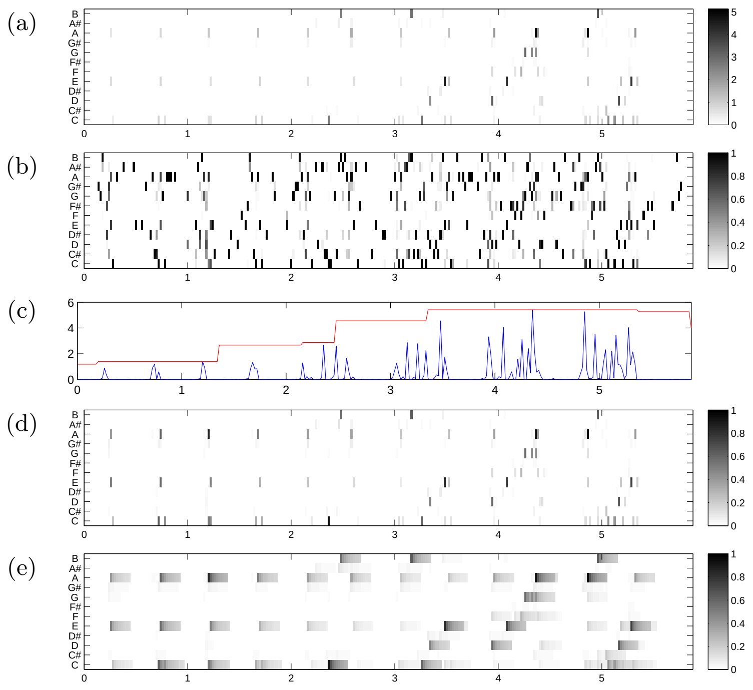

**Fig. 3.** **(a)** Chroma onset (CO) features obtained from the onset representation of Fig. 2. **(b)** Normalized CO features. **(c)** Sequence of norms of the CO features (blue) and sequence of local maxima over a time window of $\pm 1$ second (red). **(d)** Locally adaptive normalized CO (LNCO) features. **(e)** Decaying LNCO (DLNCO) features.

to onsets in passages played in forte. To compensate for this, one could simply normalize all non-zero CO feature vectors. However, this would also enhance small noisy onset features that are caused by mechanical noise, resonance, or beat effects thus leading to a useless representation, see Fig. 3b. To circumvent this problem, we employ a locally adaptive normalization strategy. First, we compute the norm for each 12-dimensional CO feature vector resulting in a sequence of norms, see Fig. 3c (blue curve). Then, for each time frame, we assign the local maxima of the sequence of norms over a time window that ranges one second to the left and one second to the right, see Fig. 3c (red curve). Furthermore, we assign a positive threshold value to all those frames where the local maximum falls below that threshold. The resulting sequence of local maxima is used to normalize the CO features in a locally adaptive fashion. To this end, we simply divide the sequence of CO features by the sequence of local maxima in a pointwise fashion, see Fig. 3d. The resulting features are referred to as *LNCO (locally adaptive normalized CO) features*. Intuitively, LNCO features account for the fact that onsets of low energy are less relevant in musical passages of high energy than in passages of low energy.

In summary, the octave identification makes LNCO features robust to variations in timbre. Furthermore, because of the locally adaptive normalization, LNCO features are invariant to variations in dynamics and exhibit significant onset values even in passages of low energy. Finally, the LNCO feature representation is sparse in the sense that most feature vectors are zero, while the non-zero vectors encode highly accurate temporal onset information.

In view of synchronization applications, we further process the LNCO feature representation by introducing an additional temporal decay. To this end, each LNCO feature vector is copied $n$ times (in our experiments we chose $n = 10$) and the copies are multiplied by decreasing positive weights starting with 1. Then, the $n$ copies are arranged to form short sequences of $n$ consecutive feature vectors of decreasing norm starting at the time position of the original vector. The overlay of all these decaying sequences results in a feature representation, which we refer to as *DLNCO (decaying LNCO) feature* representation, see Figs. 3e and 6a. The benefit of these additional temporal decays will become clear in the synchronization context, see Sect. 3.1. Note that in the DLNCO feature representation, one does not loose the temporal accuracy of the LNCO features—the onset positions still appear as sharp left edges in the decays. However, spurious double peaks, which appear in a close temporal neighborhood within a chroma band, are discarded. By introducing the decay, as we will see later, one looses sparseness while gaining robustness.

As a final remark of this section, we emphasize that the opposite variant of first computing chroma features and then computing onsets from the resulting chromagrams is not as successful as our strategy. As a first reason, note that the temporal resolution of the pitch energy curves is much higher ($2-10$ ms depending on the respective pitch) then for the chroma features (where information across various pitches is combined at a common lower temporal resolution) thus yielding a higher accuracy. As a second reason, note that by first changing to a chroma representation one may already loose valuable onset information. For example, suppose there is a clear onset in the C3 pitch band and some smearing in the C4 pitch band. Then, the smearing may overlay the onset on the chroma level, which may result in missing the onset information. However, by first computing onsets for all pitches separately and then merging this information on the chroma level, the onset of the C3 pitch band will become clearly visible on the chroma level.

## 3 Synchronization Algorithm

In this section, we show how our novel DLNCO features can be used to significantly improve the accuracy of previous chroma-based strategies without sacrificing robustness and efficiency. First, in Sect. 3.1, we introduce a combination of cost matrices that suitably captures harmonic as well as onset information. Then, in Sect. 3.2, we discuss how the new cost matrix can be plugged in an efficient multiscale music synchronization framework by using an additional alignment layer.

(a)

(b)

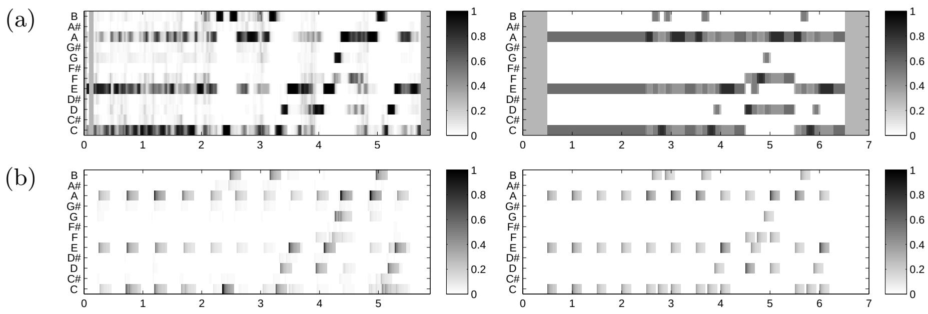

**Fig. 4. (a)** Sequences of normalized chroma features for an audio version (left) and MIDI version (right) of **Burg2**. **(b)** Corresponding sequences of DLNCO features.

## 3.1 Local Cost Measures and Cost Matrices

As discussed in the introduction, the goal of music synchronization is to time align two given versions of the same underlying piece of music. In the following, we consider the case of MIDI-audio synchronization. Other cases such as audio-audio synchronization may be handled in the same fashion. Most synchronization algorithms [3, 5, 9, 16, 17, 19, 20] rely on some variant of dynamic time warping (DTW) and can be summarized as follows. First, the two music data streams to be aligned are converted into feature sequences, say $V := (v_1, v_2, \ldots, v_N)$ and $W := (w_1, w_2, \ldots, w_M)$, respectively. Note that $N$ and $M$ do not have to be equal, since the two versions typically have a different length. 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., $C(n, m) = c(v_n, w_m)$ for $1 \leq n \leq N, 1 \leq m \leq M$. Finally, an optimum-cost alignment path is determined from this matrix via dynamic programming, which encodes the synchronization result. Our synchronization approach follows these lines using the standard DTW algorithm, see [13] for a detailed account on DTW in the music context. For an illustration, we refer to Fig. 5, which shows various cost matrices along with optimal alignment paths.

Note that the final synchronization result heavily depends on the type of features used to transform the music data streams and the local cost measure used to compare the features. We now introduce three different cost matrices, where the third one is a simple combination of the first and second one.

The first matrix is a conventional cost matrix based on normalized chroma features. Note that these features can be extracted from audio representations, as described in Sect. 2.1, as well as from MIDI representations, as suggested in [9]. Fig. 4a shows normalized chroma representations for an audio recording and a MIDI version of **Burg2**, respectively. To compare two normalized chroma vectors $v$ and $w$, we use the cost measure $c_{\mathbf{chroma}}(v, w) := 2 - \langle x, y \rangle$. Note that $\langle v, w \rangle$ is the cosine of the angle between $v$ and $w$ since the features are normalized. The offset 2 is introduced to favor diagonal directions in the DTW

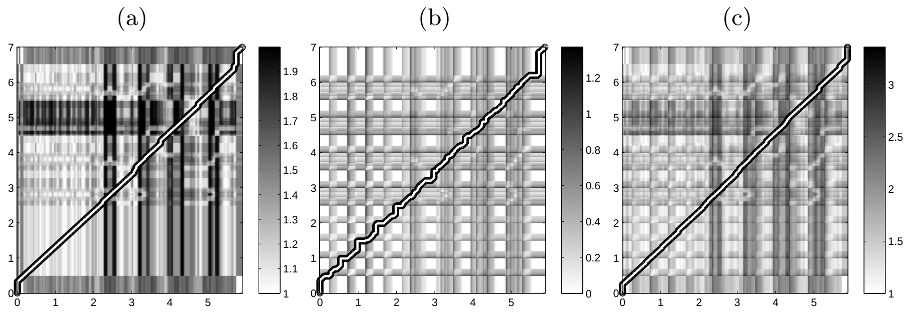

**Fig. 5.** **(a)** Cost matrix $C_{\mathbf{chroma}}$ using normalized chroma features and the local cost measure $c_{\mathbf{chroma}}$. The two underlying feature sequences are shown Fig. 4a. A cost-minimizing alignment path is indicated by the white line. **(b)** Cost matrix $C_{\mathbf{DLNCO}}$ with cost-minimizing alignment path using DLNCO features and $c_{\mathbf{DLNCO}}$. The two underlying feature sequences are shown Fig. 4b. **(c)** Cost matrix $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ and resulting cost-minimizing alignment path.

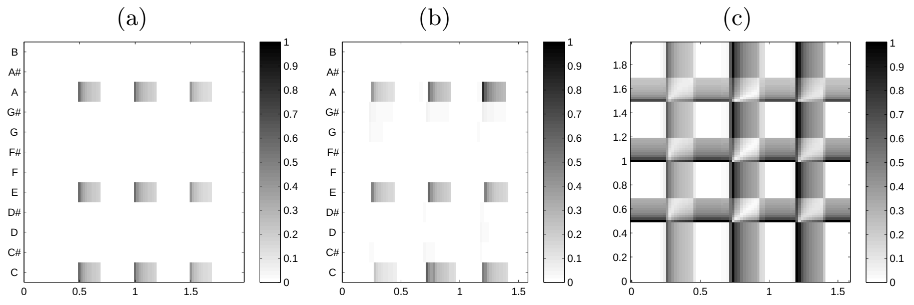

**Fig. 6.** Illustration of the effect of the decay operation on the cost matrix level. A match of two onsets leads to a small corridor within the cost matrix that exhibits low costs and is tapered to the left (where the exact onsets occur). **(a)** Beginning of the DLNCO representation of Fig. 4b (left). **(b)** Beginning of the DLNCO representation of Fig. 4b (right). **(c)** Resulting section of $C_{\mathbf{DLNCO}}$, see Fig. 5b.

algorithm in regions of uniformly low cost, see [17] for a detailed explanation. The resulting cost matrix is denoted by $C_{\mathbf{chroma}}$, see Fig. 5a.

The second cost matrix is based on DLNCO features as introduced in Sect. 2.3. Again, one can directly convert the MIDI version into a DLNCO representation by converting the MIDI note onsets into pitch onsets. Fig. 4b shows DLNCO representations for an audio recording and a MIDI version of **Burg2**, respectively. To compare two DLNCO feature vectors, $v$ and $w$ we now use the Euclidean distance $c_{\mathbf{DLNCO}}(v, w) := \|v - w\|$. The resulting cost matrix is denoted by $C_{\mathbf{DLNCO}}$, see Fig. 5b. At this point, we need to make some explanations. First, recall that each onset has been transformed into a short vector sequence of decaying norm. Using the Euclidean distance to compare two such decaying sequences leads to a diagonal corridor of low cost in $C_{\mathbf{DLNCO}}$ in the case that the directions (i. e., the relative chroma distributions) of the onset vectors are similar. This corridor is tapered to the lower left and starts at the precise time positions of the two onsets to be compared, see Fig. 6c. Second, note that $C_{\mathbf{DLNCO}}$ reveals a grid like structure of an overall high cost, where each beginning of a corridor forms a small needle's eye of low cost. Third, sections in the feature sequences with no onsets lead to regions in $C_{\mathbf{DLNCO}}$ having zero cost. In other words, only significant events in the DLNCO feature sequences take effect on the cost matrix level. In summary, the structure of $C_{\mathbf{DLNCO}}$ regulates the course of a cost-minimizing alignment path in event-based regions to run through the needle's eyes of low cost. This leads to very accurate alignments at time positions with matching chroma onsets.

The two cost matrices $C_{\mathbf{chroma}}$ and $C_{\mathbf{DLNCO}}$ encode complementary information of the two music representations to be synchronized. The matrix $C_{\mathbf{chroma}}$ accounts for the rough harmonic flow of the two representations, whereas $C_{\mathbf{DLNCO}}$ exhibits matching chroma onsets. Forming the sum $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ yields a cost matrix that accounts for both types of information. Note that in regions with no onsets, $C_{\mathbf{DLNCO}}$ is zero and the combined matrix $C$ is dominated by $C_{\mathbf{chroma}}$. Contrary, in regions with significant onsets, $C$ is dominated by $C_{\mathbf{DLNCO}}$, thus enforcing the cost-minimizing alignment path to run trough the needle's eyes of low cost. Note that in a neighborhood of these eyes, the cost matrix $C_{\mathbf{chroma}}$ also reveals low costs due to the similar chroma distribution of the onsets. In summary, the component $C_{\mathbf{chroma}}$ regulates the overall course of the cost-minimizing alignment path and accounts for a robust synchronization, whereas the component $C_{\mathbf{DLNCO}}$ locally adjusts the alignment path and accounts for highly temporal accuracy.

### 3.2 Multiscale Implementation

Note that the time and memory complexity of DTW-based music synchronization linearly depends on the product $N \cdot M$ of the lengths $N$ and $M$ of the feature sequences to be aligned. For example, having a feature resolution of 20 ms and music data streams of 10 minutes of duration, results in $N = M = 30000$ making computations infeasible. To overcome this problem, we adapt an efficient multiscale DTW (MsDTW) approach as described in [17]. The idea is to calculate an alignment path in an iterative fashion by using multiple resolution levels going from coarse to fine. Here, the results of the coarser level are used to constrain the calculation on the finer levels, see Fig. 7.

In a first step, we use the chroma-based MsDTW as described in [17]. In particular, we employ an efficient MsDTW implementation in C/C++ (used as a MATLAB DLL), which is based on three levels corresponding to a feature resolution of 1/3 Hz, 1 Hz, and 10 Hz, respectively. For example, our implementation needs less than a second (not including the feature extraction, which is linear in the length of the pieces) on a standard PC for synchronizing two music data streams each having a duration of 15 minutes of duration. The MsDTW synchronization is robust leading to reliable, but coarse alignments, which often reveal deviations of several hundreds of milliseconds.

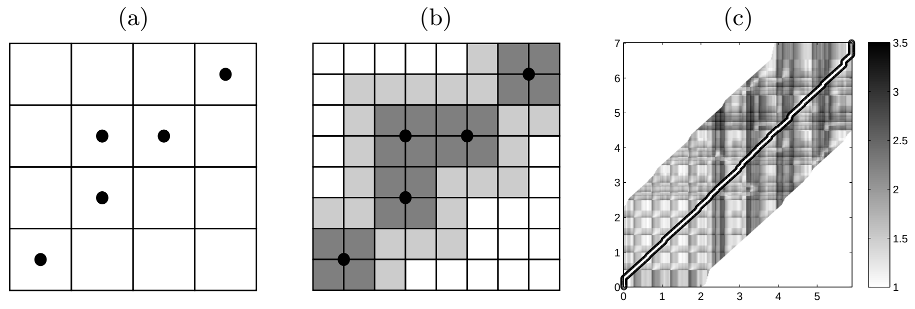

**Fig. 7.** Illustration of multiscale DTW. **(a)** Optimal alignment path (black dots) computed on a coarse resolution level. **(b)** Projection of the alignment path onto a finer resolution level with constraint region (dark gray) and extended constraint region (light gray). **(c)** Constraint region for **Burg2**, cf. Fig. 5c. The entries of the cost matrix are only computed within the constraint region. The resulting MsDTW alignment path indicated by the white line coincides with the DTW alignment path shown in Fig. 5c.

To refine the synchronization result, we employ an additional alignment level corresponding to a feature resolution of 50 Hz (i. e., each feature corresponds to 20 ms). On this level, we use the cost matrix $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ as described in Sect. 3.1. First, the resulting alignment path of the previous MsDTW method (corresponding to a 10 Hz feature resolution) is projected onto the 50 Hz resolution level. The projected path is used to define a tube-like constraint region, see Fig. 7b. As before, the cost matrix $C$ is only evaluated within this region, which leads to large savings if the region is small. However, note that the final alignment path is also restricted to this region, which may lead to incorrect alignment paths if the region is too small [17]. As our experiments showed, an extension of two seconds in all four directions (left, right, up, down) of the projected alignment path yields a good compromise between efficiency and robustness. Fig. 7c shows the resulting extended constraint region for our running example **Burg2**. The relative savings with respect to memory requirements and running time of our overall multiscale procedure increases significantly with the length of the feature sequences to be aligned. For example, our procedure needs only around $3 \cdot 10^6$ of the total number of $15000^2 = 2.25 \cdot 10^8$ matrix entries for synchronizing two versions of a five minute piece, thus decreasing the memory requirements by a factor of 75. For a ten minute piece, this factor already amounts to 150. The relative savings for the running times are similar.

## 4 Resolution Refinement through Interpolation

A synchronization result is encoded by an alignment path, which assigns the elements of one feature sequence to the elements of the other feature sequence. Note that each feature refers to an entire analysis window, which corresponds to a certain time range rather than a single point in time. Therefore, an alignment path should be regarded as an assignment of certain time ranges. Furthermore,

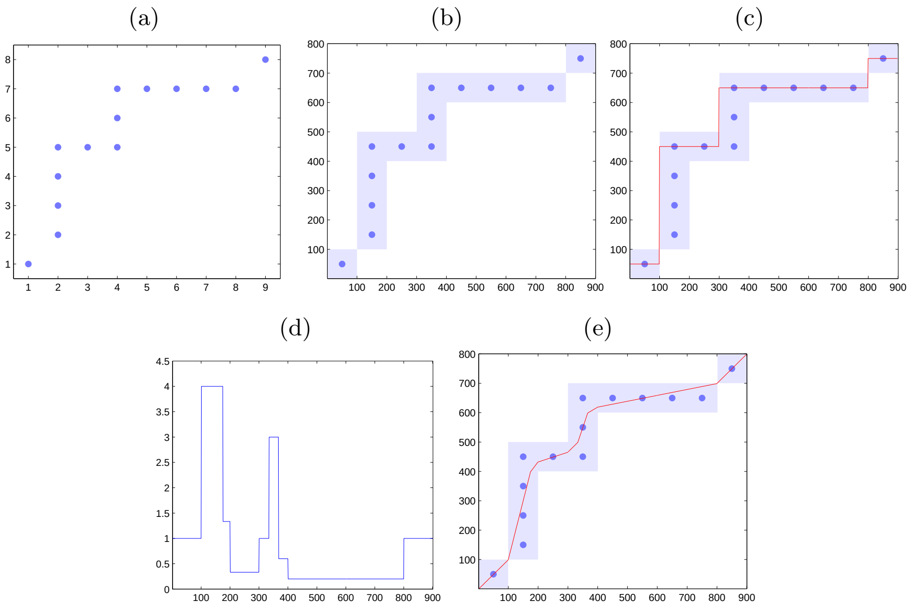

**Fig. 8.** **(a)** Alignment path assigning elements of one feature sequence to elements of the other feature sequence. The elements are indexed by natural numbers. **(b)** Assignment of time ranges corresponding to the alignment path, where each feature corresponds to a time range of 100 ms. **(c)** Staircase interpolation path (red line). **(d)** Density function encoding the local distortions. **(e)** Smoothed and strictly monotonic interpolation path obtained by integration of the density function.

an alignment path may not be strictly monotonic in its components, i. e., a single element of one feature sequence may be assigned to several consecutive elements of the other feature sequence. This further increases the time ranges in the assignment. As illustration, consider Fig. 8, where each feature corresponds to a time range of 100 ms. For example, the fifth element of the first sequence (vertical axis) is assigned to the second, third, and forth element of the second sequence (horizontal axis), see Fig. 8a. This corresponds to an assignment of the range between 400 and 500 ms with the range between 100 and 400 ms, see Fig. 8b. One major problem of such an assignment is that the temporal resolution may not suffice for certain applications. For example, one may want to use the alignment result in order to temporally warp CD audio recordings, which are typically sampled at a rate of 44,1 kHz.

To increase the temporal resolution, one usually reverts to interpolation techniques. Many of the previous approaches are based on simple staircase paths as indicated by the red line of Fig. 8c. However, such paths are not strictly monotonic and reveal abrupt directional changes leading to strong local temporal distortions. To avoid such distortions, one has to smooth the alignment path in such a way that both of its components are strictly monotonic increasing.

To this end, Kovar et al. [10] fit a spline into the alignment path and enforce the strictness condition by suitably adjusting the control points of the splines. In the following, we introduce a novel strictly monotonic interpolation function that closely reflects the course of the original alignment path. Recall that the original alignment path encodes an assignment of time ranges. The basic idea is that each assignment defines a local distortion factor, which is the proportion of the ranges' sizes. For example, the assignment of the range between 400 and 500 ms with the range between 100 and 400 ms, as discussed above, defines a local distortion factor of 1/3. Elaborating on this idea, one obtains a density function that encodes the local distortion factors. As an illustration, we refer to Fig. 8d, which shows the resulting density function for the alignment path of Fig. 8a. Then, the final interpolation path is obtained by integrating over the density function, see Fig. 8e. Note that the resulting interpolation path is a smoothed and strictly monotonic version of the original alignment path. The continuous interpolation path can be used for arbitrary sampling rates. Furthermore, as we will see in Sect. 5, it also improves the final synchronization quality.

## 5 Experiments

In this section, we report on some of our synchronization experiments, which have been conducted on a corpus of harmony-based Western music. To allow for a reproduction of our experiments, we used pieces from the RWC music database [7, 8]. In the following, we consider 16 representative pieces, which are listed in Table 1. These pieces are divided into three groups, where the first group consists of six classical piano pieces, the second group of five classical pieces of various instrumentations (full orchestra, strings, flute, voice), and the third group of five jazz pieces and pop songs. Note that for pure piano music, one typically has concise note attacks resulting in characteristic onset features. Contrary, such information is often missing in string or general orchestral music. To account for such differences, we report on the synchronization accuracy for each of the three groups separately.

To demonstrate the respective effect of the different refinement strategies on the final synchronization quality, we evaluated eight different synchronization procedures. The first procedure (MsDTW) is the MsDTW approach as described in [17], which works with a feature resolution of 10 Hz. The next three procedures are all refinements of the first procedure working with an additional alignment layer using a feature resolution of 50 Hz. In particular, we use in the second procedure (Chroma 20ms) normalized chroma features, in the third procedure (DLNCO) only the DLNCO features, and in the forth procedure (Chroma+DLNCO) a combination of these features, see Sect. 3.1. Besides the simple staircase interpolation, we also refined each of these four procedure via smooth interpolation as discussed in Sect. 4. Table 2, which will be discussed later in detail, indicates the accuracy of the alignment results for each of the eight synchronization procedures.

| ID | Comp./Interp. | Piece | RWC ID | Instrument |
|---|---|---|---|---|
| **Burg2** | Burgmüller | Etude No. 2, Op. 100 | – | piano |
| **BachFuge** | Bach | Fuge, C-Major, BWV 846 | C025 | piano |
| **BeetApp** | Beethoven | Op. 57, 1st Mov. (Appasionata) | C028 | piano |
| **ChopTris** | Chopin | Etude Op. 10, No. 3 (Tristesse) | C031 | piano |
| **ChopBees** | Chopin | Etude Op. 25, No. 2 (The Bees) | C032 | piano |
| **SchuRev** | Schumann | Reverie (Träumerei) | C029 | piano |
| **BeetFifth** | Beethoven | Op. 67, 1st Mov. (Fifth) | C003 | orchestra |
| **BorString** | Borodin | String Quartett No. 2, 3rd Mov. | C015 | strings |
| **BrahDance** | Brahms | Hungarian Dance No. 5 | C022 | orchestra |
| **RimskiBee** | Rimski-Korsakov | Flight of the Bumblebee | C044 | flute/piano |
| **SchubLind** | Schubert | Op. 89, No. 5 (Der Lindenbaum) | C044 | voice/piano |
| **Jive** | Nakamura | Jive | J001 | piano |
| **Entertain** | HH Band | The Entertainer | J038 | big band |
| **Friction** | Umitsuki Quartet | Friction | J041 | sax,bass,perc. |
| **Moving** | Nagayama | Moving Round and Round | P031 | electronic |
| **Dreams** | Burke | Sweet Dreams | P093 | voice/guitar |

**Table 1.** Pieces of music with identifier (ID) contained in our test database. For better reproduction of our experiments, we used pieces from the RWC music database [7, 8].

To automatically determine the accuracy of our synchronization procedures, we used pairs of MIDI and audio versions for each of the 16 pieces listed in Table 1. Here, the audio versions were generated from the MIDI files using a high-quality synthesizer. Thus, for each synchronization pair, the note onset times in the MIDI file are perfectly aligned with the physical onset times in the respective audio recording. (Only for our running example **Burg2**, we manually aligned some real audio recording with a corresponding MIDI version.) In the first step of our evaluation process, we randomly distorted the MIDI files. To this end, we split up the MIDI files into $N$ segments of equal length (in our experiment we used $N = 20$) and then stretched or compressed each segment by a random factor within an allowed distortion range (in our experiments we used a range of $\pm 30\%$). We refer to the resulting MIDI file as the *distorted MIDI file* in contrast to the original *annotation MIDI file*. In the second evaluation step, we synchronized the distorted MIDI file and the associated audio recording. The resulting alignment path was used to adjust the note onset times in the distorted MIDI file to obtain a third MIDI file referred to as *realigned MIDI file*. The accuracy of the synchronization result can now be determined by comparing the note onset times of the realigned MIDI file with the corresponding note onsets of the annotation MIDI file. Note that in the case of a perfect synchronization, the realigned MIDI file exactly coincides with the annotation MIDI file.

For each of the 16 pieces (Table 1) and for each of the eight different synchronization procedures, we computed the corresponding realigned MIDI file. We then calculated the mean value, the standard deviation, as well as the maximal value over all note onset differences comparing the respective realigned MIDI file with the corresponding annotation MIDI file. Thus, for each piece, we obtained 24 statistical values, which are shown in Table 2. (Actually, we also repeated all experiments with five different randomly distorted MIDI files and averaged all

| | | staircase | | | smooth | | |
|---|---|---|---|---|---|---|---|
| ID | Procedure | mean | std | max | mean | std | max |
| **Burg2** | MsDTW | 73 | 57 | 271 | 71 | 65 | 307 |
| | Chroma 20ms | 49 | 43 | 222 | 50 | 48 | 228 |
| | DLNCO | 31 | 20 | 94 | 21 | 17 | 73 |
| | **Chroma+DLNCO** | **28** | **16** | **77** | **18** | **14** | **61** |
| **BachFuge** | MsDTW | 97 | 55 | 319 | 55 | 41 | 223 |
| | Chroma 20ms | 34 | 34 | 564 | 27 | 33 | 554 |
| | DLNCO | 20 | 30 | 318 | 18 | 27 | 296 |
| | **Chroma+DLNCO** | **18** | **15** | **96** | **14** | **12** | **81** |
| **BeetApp** | MsDTW | 116 | 102 | 1197 | 77 | 94 | 1104 |
| | Chroma 20ms | 62 | 58 | 744 | 54 | 58 | 757 |
| | DLNCO | 136 | 318 | 2323 | 131 | 318 | 2335 |
| | **Chroma+DLNCO** | **37** | **41** | **466** | **29** | **40** | **478** |
| **ChopTris** | MsDTW | 115 | 76 | 1041 | 72 | 62 | 768 |
| | Chroma 20ms | 66 | 69 | 955 | 57 | 64 | 754 |
| | DLNCO | 30 | 68 | 1318 | 22 | 68 | 1305 |
| | **Chroma+DLNCO** | **31** | **34** | **539** | **22** | **33** | **524** |
| **ChopBees** | MsDTW | 108 | 79 | 865 | 59 | 71 | 817 |
| | Chroma 20ms | 41 | 49 | 664 | 30 | 47 | 625 |
| | DLNCO | 20 | 14 | 104 | 12 | 9 | 95 |
| | **Chroma+DLNCO** | **22** | **24** | **366** | **13** | **21** | **355** |
| **SchuRev** | MsDTW | 93 | 95 | 887 | 66 | 77 | 655 |
| | Chroma 20ms | 51 | 80 | 778 | 46 | 72 | 567 |
| | DLNCO | 98 | 261 | 1789 | 94 | 264 | 1841 |
| | **Chroma+DLNCO** | **22** | **38** | **330** | **15** | **36** | **315** |
| Average over piano examples | MsDTW | 100 | 77 | 763 | 67 | 68 | 646 |
| | Chroma 20ms | 51 | 56 | 655 | 44 | 54 | 581 |
| | DLNCO | 56 | 119 | 991 | 50 | 117 | 991 |
| | **Chroma+DLNCO** | **26** | **28** | **312** | **19** | **26** | **302** |
| **BeetFifth** | MsDTW | 194 | 124 | 1048 | 142 | 116 | 952 |
| | Chroma 20ms | 128 | 98 | 973 | 116 | 96 | 959 |
| | DLNCO | 254 | 338 | 2581 | 241 | 338 | 2568 |
| | **Chroma+DLNCO** | **128** | **99** | **1144** | **116** | **98** | **1130** |
| **BorString** | MsDTW | 157 | 110 | 738 | 118 | 106 | 734 |
| | Chroma 20ms | 88 | 68 | 584 | 79 | 68 | 576 |
| | DLNCO | 275 | 355 | 2252 | 268 | 356 | 2233 |
| | **Chroma+DLNCO** | **91** | **57** | **682** | **82** | **56** | **675** |
| **BrahDance** | MsDTW | 104 | 62 | 385 | 64 | 54 | 470 |
| | Chroma 20ms | 58 | 54 | 419 | 50 | 54 | 427 |
| | DLNCO | 31 | 52 | 567 | 26 | 52 | 556 |
| | **Chroma+DLNCO** | **24** | **22** | **185** | **17** | **20** | **169** |
| **RimskiBee** | MsDTW | 99 | 48 | 389 | 50 | 32 | 196 |
| | Chroma 20ms | 51 | 17 | 167 | 41 | 17 | 155 |
| | DLNCO | 31 | 23 | 183 | 22 | 19 | 160 |
| | **Chroma+DLNCO** | **37** | **17** | **108** | **27** | **15** | **91** |
| **SchubLind** | MsDTW | 124 | 73 | 743 | 78 | 59 | 549 |
| | Chroma 20ms | 66 | 57 | 718 | 55 | 50 | 509 |
| | DLNCO | 79 | 175 | 1227 | 70 | 173 | 1206 |
| | **Chroma+DLNCO** | **41** | **36** | **406** | **31** | **34** | **387** |
| Average over various intstrumentation examples | MsDTW | 136 | 83 | 661 | 90 | 73 | 580 |
| | Chroma 20ms | 78 | 59 | 572 | 68 | 57 | 525 |
| | DLNCO | 134 | 189 | 1362 | 125 | 188 | 1345 |
| | **Chroma+DLNCO** | **64** | **46** | **505** | **55** | **45** | **490** |
| **Jive** | MsDTW | 97 | 105 | 949 | 58 | 93 | 850 |
| | Chroma 20ms | 44 | 61 | 686 | 34 | 59 | 668 |
| | DLNCO | 23 | 38 | 638 | 17 | 37 | 632 |
| | **Chroma+DLNCO** | **22** | **18** | **154** | **14** | **15** | **158** |
| **Entertain** | MsDTW | 100 | 67 | 579 | 66 | 58 | 492 |
| | Chroma 20ms | 52 | 44 | 407 | 45 | 46 | 414 |
| | DLNCO | 93 | 204 | 1899 | 85 | 204 | 1887 |
| | **Chroma+DLNCO** | **40** | **65** | **899** | **31** | **64** | **889** |
| **Friction** | MsDTW | 94 | 81 | 789 | 58 | 75 | 822 |
| | Chroma 20ms | 47 | 67 | 810 | 39 | 67 | 815 |
| | DLNCO | 44 | 120 | 2105 | 37 | 117 | 2106 |
| | **Chroma+DLNCO** | **30** | **55** | **810** | **23** | **55** | **819** |
| **Moving** | MsDTW | 114 | 76 | 497 | 76 | 64 | 473 |
| | Chroma 20ms | 77 | 51 | 336 | 68 | 50 | 343 |
| | DLNCO | 127 | 216 | 1443 | 124 | 217 | 1432 |
| | **Chroma+DLNCO** | **53** | **45** | **284** | **46** | **43** | **275** |
| **Dreams** | MsDTW | 136 | 105 | 659 | 115 | 106 | 674 |
| | Chroma 20ms | 97 | 94 | 702 | 91 | 95 | 673 |
| | DLNCO | 73 | 103 | 692 | 71 | 103 | 702 |
| | **Chroma+DLNCO** | **43** | **57** | **429** | **40** | **58** | **434** |
| Average over jazz/pop examples | MsDTW | 108 | 87 | 695 | 75 | 79 | 662 |
| | Chroma 20ms | 63 | 63 | 588 | 55 | 63 | 583 |
| | DLNCO | 72 | 136 | 1355 | 67 | 136 | 1352 |
| | **Chroma+DLNCO** | **38** | **48** | **515** | **31** | **47** | **515** |
| Average over all examples | MsDTW | 114 | 82 | 710 | 77 | 73 | 630 |
| | Chroma 20ms | 63 | 59 | 608 | 55 | 58 | 564 |
| | DLNCO | 85 | 146 | 1221 | 79 | 145 | 1214 |
| | **Chroma+DLNCO** | **42** | **40** | **436** | **34** | **38** | **428** |

**Table 2.** Alignment accuracy for eight different synchronization procedures (MsDTW, Chroma 20 ms, DLNCO, Chroma+DLNCO with staircase and smooth interpolation, respectively). The table shows for each of the eight procedures and for each of 16 pieces (Table 1) the mean value, the standard deviation, and the maximal value over all note onset difference of the respective realigned MIDI file and the corresponding annotation MIDI file. All values are given in milliseconds.

statistical values over these five repetitions). For example the value 73 in the first row of Table 2 means that for the piece **Burg2** the difference between the note onsets of the realigned MIDI file and the annotation MIDI file was in average 73 ms when using the MsDTW synchronisation approach in combination with a staircase interpolation. In other words, the average synchronization error of this approach is 73 ms for **Burg2**.

We start the discussion of Table 2 by looking at the values for the first group consisting of six piano pieces. Looking at the averages of the statistical values over the six piece, one can observe that the MsDTW procedures is clearly inferior to the other procedures. This is by no surprise, since the feature resolution of MsDTW is 100 ms compared to the resolution of 20 ms used in the other approaches. Nevertheless the standard deviation and maximal deviation of MsDTW is small relative to the mean value indicating the robustness of this approach. Using 20 ms chroma features, the average mean values decreases from 100 ms (MsDTW) to 51 ms (Chroma 20 ms). Using the combined features, this value further decreases to 26 ms (Chroma+DLNCO). Furthermore, using the smooth interpolation instead of the simple staircase interpolation further improves the accuracy, for example, from 100 ms to 67 ms (MsDTW) or from 26 ms to 19 ms (Chroma+DLNCO). Another interesting observation is that the pure DLNCO approach is sometimes much better (e. g. for **ChopBees**) but also sometimes much worse (e. g. for **BeetApp**) than the Chroma 20ms approach. This shows that the DLNCO features have the potential for delivering very accurate results but also suffer from a lack of robustness. It is the combination of the DLNCO features and chroma features which ensures robustness as well as accuracy of the overall synchronization procedure.

Next, we look at the group of the five classical pieces of various instrumentations. Note that for the pieces of this group, opposed to the piano pieces, one often has no clear note attacks leading to a much poorer quality of the onset features. As a consequence, the synchronization errors are in average higher than for the piano pieces. For example, the average mean error over the second group is 136 ms (MsDTW) and 134 ms (DLNCO) opposed to 100 ms (MsDTW) and 56 ms (DLNCO) for the first group. However, even in the case of missing onset information, the synchronization task is still accomplished in a robust way by means of the harmony-based chroma features. The idea of using the combined approach (Chroma+DLNCO) is that the resulting synchronization procedure is at least as robust and exact as the pure chroma-based approach (Chroma 20 ms). Table 2 demonstrates that this idea is realized by the implementation of our combined synchronization procedure. Similar results are obtained for the third group of jazz/pop examples, where the best results were also delivered by the combined approach (Chroma+DLNCO).

At this point, one may object that one typically obtains better absolute synchronization results for synthetic audio material (which was used to completely automate our evaluation) than for non-synthetic, real audio recordings. We therefore included also the real audio recording **Burg2**, which actually led to similar results as the synthesized examples. Furthermore, our experi-

| | | Distortion range | | | | |
|---|---|---|---|---|---|---|
| ID | Procedure | ±10% | ±20% | ±30% | ±40% | ±50% |
| **Burg2** | MsDTW | 48 | 53 | 65 | 85 | 94 |
| | **Chroma+DLNCO** | **15** | **16** | **19** | **17** | **22** |
| **BachFuge** | MsDTW | 44 | 49 | 52 | 62 | 67 |
| | **Chroma+DLNCO** | **11** | **12** | **13** | **15** | **15** |
| **BeetApp** | MsDTW | 53 | 68 | 75 | 96 | 170 |
| | **Chroma+DLNCO** | **22** | **25** | **29** | **36** | **98** |
| **ChopTris** | MsDTW | 57 | 64 | 72 | 75 | 82 |
| | **Chroma+DLNCO** | **18** | **19** | **21** | **22** | **29** |
| **ChopBees** | MsDTW | 51 | 54 | 57 | 60 | 67 |
| | **Chroma+DLNCO** | **11** | **12** | **13** | **14** | **18** |
| **SchuRev** | MsDTW | 50 | 58 | 64 | 77 | 85 |
| | **Chroma+DLNCO** | **11** | **14** | **12** | **13** | **22** |
| Average over piano examples | MsDTW | 51 | 58 | 64 | 76 | 94 |
| | **Chroma+DLNCO** | **15** | **16** | **18** | **20** | **34** |
| **BeetFifth** | MsDTW | 119 | 126 | 141 | 143 | 184 |
| | **Chroma+DLNCO** | **101** | **106** | **113** | **113** | **145** |
| **BorString** | MsDTW | 86 | 97 | 109 | 118 | 153 |
| | **Chroma+DLNCO** | **75** | **78** | **82** | **84** | **101** |
| **BrahDance** | MsDTW | 52 | 58 | 66 | 70 | 81 |
| | **Chroma+DLNCO** | **13** | **15** | **18** | **19** | **25** |
| **RimskiBee** | MsDTW | 49 | 47 | 52 | 53 | 56 |
| | **Chroma+DLNCO** | **25** | **26** | **26** | **28** | **28** |
| **SchubLind** | MsDTW | 69 | 73 | 78 | 99 | 91 |
| | **Chroma+DLNCO** | **28** | **28** | **31** | **35** | **35** |
| Average over various intstrumentation examples | MsDTW | 75 | 80 | 89 | 97 | 113 |
| | **Chroma+DLNCO** | **48** | **51** | **54** | **56** | **67** |
| **Jive** | MsDTW | 44 | 62 | 50 | 63 | 77 |
| | **Chroma+DLNCO** | **12** | **13** | **14** | **14** | **15** |
| **Entertain** | MsDTW | 47 | 53 | 62 | 78 | 94 |
| | **Chroma+DLNCO** | **21** | **25** | **30** | **36** | **44** |
| **Friction** | MsDTW | 44 | 48 | 54 | 70 | 82 |
| | **Chroma+DLNCO** | **14** | **17** | **22** | **28** | **37** |
| **Moving** | MsDTW | 61 | 63 | 75 | 127 | 871 |
| | **Chroma+DLNCO** | **33** | **39** | **47** | **59** | **732** |
| **Dreams** | MsDTW | 71 | 84 | 114 | 142 | 178 |
| | **Chroma+DLNCO** | **24** | **28** | **39** | **52** | **85** |
| Average over jazz/pop examples | MsDTW | 53 | 62 | 71 | 96 | 260 |
| | **Chroma+DLNCO** | **21** | **24** | **30** | **38** | **183** |
| Average over all examples | MsDTW | 59 | 66 | 74 | 89 | 152 |
| | **Chroma+DLNCO** | **27** | **30** | **33** | **37** | **91** |

**Table 3.** Dependency of the final synchronization accuracy on the size of the allowed distortion range. For each of the 16 pieces and each range, the mean values of the synchronization errors are given for the MsDTW and Chroma+DLNCO procedure both post-processed with smooth interpolation. All values are given in milliseconds.

ments on the synthetic data are still meaningful in the relative sense by revealing relative performance differences between the various synchronization procedures. Finally, we also generated MIDI-audio alignments using real performances of the corresponding pieces (which are also contained in the RWC music database). These alignments were used to modify the original MIDI files to run synchronously to the audio recordings. Generating a stereo file with a synthesized version of the modified MIDI file in one channel and the audio recording in the other channel, we have acoustically examined the alignment results. The acoustic impression supports the evaluation results obtained from the synthetic data. The stereo files have been made available on the website `http://www-mmdb.iai.uni-bonn.de/projects/syncDLNCO/`.

For the experiments of Table 2, we used a distortion range of ±30%, which is motivated by the observation that the relative tempo difference between two real performances of the same piece mostly lies within this range. In a second experiment, we investigated the dependency of the final synchronization accuracy on the size of the allowed distortion range. To this end, we calculated the mean values of the synchronization error for each of the 16 pieces using different distortion ranges from ±10% to ±50%. Table 3 shows the resulting vales for two of the eight synchronization procedures described above, namely MsDTW and Chroma+DLNCO both post-processed with smooth interpolation. As one may expect, the mean error values increase with the allowed distortion range. For example, the average mean error over all 16 pieces increases from 59 ms to 152 ms for the MsDTW and from 27 ms to 91 ms for the combined procedure (Chroma+DLNCO). However, the general behavior of the various synchronization procedures does not change significantly with the ranges and the overall synchronization accuracy is still high even in the presence of large distortions. As an interesting observation, for one of the pieces (**Moving**) the mean error exploded from 59 ms to 732 ms (Chroma+DLNCO) when increasing the range from ±40% to ±50%. Here, a manual inspection showed that, for the latter range, a systematic synchronization error happened. Here, for an entire musical segment of the piece, the audio version was aligned to a similar subsequent repetition of the segment in the distorted MIDI version. However, note that such strong distortion (±50% corresponds to the range of having half tempo to double tempo) rarely occurs in practice and only causes problems for repetitive music.

## 6 Conclusions

In this paper, we have discussed various refinement strategies for music synchronization. Based on a novel class of onset-based audio features in combination with previous chroma features, we presented a new synchronization procedure that can significantly improve the synchronization accuracy while preserving the robustness and efficiency of previously described procedures. For the future, we plan to further extend our synchronization framework by including various features types that also capture local rhythmic information [11] and that detect even smooth note transitions as often present in orchestral or string music [23]. As a further extension of our work, we will consider the problem of partial music synchronization, where the two versions to be aligned may reveal significant structural differences.

## References

1. V. Arifi, M. Clausen, F. Kurth, and M. Müller. Synchronization of music data in score-, MIDI- and PCM-format. *Computing in Musicology*, 13, 2004.
2. M. A. Bartsch and G. H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Trans. on Multimedia*, 7(1):96–104, Feb. 2005.
3. R. Dannenberg and N. Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proc. ICMC, San Francisco, USA*, pages 27–34, 2003.
4. R. Dannenberg and C. Raphael. Music score alignment and computer accompaniment. *Special Issue, Commun. ACM*, 49(8):39–43, 2006.
5. S. Dixon and G. Widmer. Match: A music alignment tool chest. In *Proc. ISMIR, London, GB*, 2005.
6. H. Fujihara, M. Goto, J. Ogata, K. Komatani, T. Ogata, and H. G. Okuno. Automatic synchronization between lyrics and music cd recordings based on viterbi alignment of segregated vocal signals. In *ISM*, pages 257–264, 2006.

7. M. Goto. Development of the rwc music database.
8. M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. Rwc music database: Popular, classical and jazz music databases. In *ISMIR*, 2002.
9. N. Hu, R. Dannenberg, and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proc. IEEE WASPAA, New Paltz, NY*, October 2003.
10. L. Kovar and M. Gleicher. Flexible automatic motion blending with registration curves. In *Proc. 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 214–224. Eurographics Association, 2003.
11. F. Kurth, T. Gehrmann, and M. Müller. The cyclic beat spectrum: Tempo-related audio features for time-scale invariant audio identification. In *Proc. ISMIR, Victoria, Canada*, pages 35–40, 2006.
12. F. Kurth, M. Müller, C. Fremerey, Y. Chang, and M. Clausen. Automated synchronization of scanned sheet music with audio recordings. In *Proc. ISMIR, Vienna, AT*, 2007.
13. M. Müller. *Information Retrieval for Music and Motion*. Springer, 2007.
14. M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proc. ISMIR, London, GB*, 2005.
15. M. Müller, F. Kurth, D. Damm, C. Fremerey, and M. Clausen. Lyrics-based audio retrieval and multimodal navigation in music collections. In *Proc. 11th European Conference on Digital Libraries (ECDL)*, 2007.
16. M. Müller, F. Kurth, and T. Röder. Towards an efficient algorithm for automatic score-to-audio synchronization. In *Proc. ISMIR, Barcelona, Spain*, 2004.
17. M. Müller, H. Mattes, and F. Kurth. An efficient multiscale approach to audio synchronization. In *Proc. ISMIR, Victoria, Canada*, pages 192–197, 2006.
18. C. Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proc. ISMIR, Barcelona, Spain*, 2004.
19. F. Soulez, X. Rodet, and D. Schwarz. Improving polyphonic and poly-instrumental music to score alignment. In *Proc. ISMIR, Baltimore, USA*, 2003.
20. R. J. Turetsky and D. P. Ellis. Force-Aligning MIDI Syntheses for Polyphonic Music Transcription Generation. In *Proc. ISMIR, Baltimore, USA*, 2003.
21. Y. Wang, M.-Y. Kan, T. L. Nwe, A. Shenoy, and J. Yin. LyricAlly: automatic synchronization of acoustic musical signals and textual lyrics. In *MULTIMEDIA '04: Proc. 12th annual ACM international conference on Multimedia*, pages 212–219, New York, NY, USA, 2004. ACM Press.
22. G. Widmer. Using ai and machine learning to study expressive music performance: project survey and first report. *AI Commun.*, 14(3):149–162, 2001.
23. W. You and R. Dannenberg. Polyphonic music note onset detection using semi-supervised learning. In *Proc. ISMIR, Vienna, Austria*, 2007.