Compensating for Asynchronies between Musical Voices in Score-Performance Alignment
Wang, S; Ewert, S; Dixon, S

© 2015, IEEE

For additional information about this publication click this link.
http://qmro.qmul.ac.uk/xmlui/handle/123456789/12170

Information about this research object was correct at the time of download; we occasionally make corrections to records, please therefore check the published record when citing. For more information contact scholarlycommunications@qmul.ac.uk
Compensating for Asynchronies Between Musical Voices in Score-Performance Alignment

Siying Wang  Sebastian Ewert  Simon Dixon
Queen Mary University of London, UK

ABSTRACT
The goal of score-performance synchronisation is to align a given musical score to an audio recording of a performance of the same piece. A major challenge in computing such alignments is to account for musical parameters including the local tempo or playing style. To increase the overall robustness, current methods assume that notes occurring simultaneously in the score are played concurrently in a performance. Musical voices such as the melody, however, are often played asynchronously to other voices, which can lead to significant local alignment errors. In this paper, we present a novel method that handles asynchronies between the melody and the accompaniment by treating the voices as separate timelines in a multi-dimensional variant of dynamic time warping (DTW). Constraining the alignment with information obtained via classical DTW, our method measurably improves the alignment accuracy for pieces with asynchronous voices and preserves the accuracy otherwise.

Index Terms—score-audio alignment, multi-dimensional dynamic time warping, asynchrony, melody lead.

1. INTRODUCTION

Methods for the automatic alignment of different versions of a piece of music have a long history in music signal processing. In particular, the score-performance alignment problem has seen significant efforts in recent years. Applications include real-time score following [1–5] and automatic page-turning [6], musical expression analysis [7, 8], navigation in large music collections [9], informed audio editing and source separation [10]. In general, given a symbolic score representation (MIDI, MusicXML) and an audio recording of a performance of a piece of music, score-performance synchronisation methods aim at linking each note event in the score to its corresponding position in the recording.

A main difficulty in computing such alignments stems from the diversity of possible interpretations of a piece by a musician, i.e. not only the acoustic conditions can change considerably between recordings but also musical parameters including the playing style, expressive timing, or embellishments. To increase the overall robustness, state-of-the-art methods typically make simplifying assumptions about the problem, and in particular, that notes occurring simultaneously in the score are also played concurrently during a performance [11–13]. However, introducing asynchronies between simultaneous notes is considered an important part of musical expression. For example, emphasising a musical voice such as the melody by playing it earlier compared to other voices is a form of expression typically referred to as melody lead [14]. While such asynchronies usually do not have a strong effect on the alignment on a coarse level, the alignment accuracy on a finer, local level can drop measurably as the asynchrony is not expected by current methods.

To cope with possible asynchronies between the melody and the accompaniment, the main idea in this paper is to separate the two voices in the score and to compute a joint three-dimensional alignment between the two score timelines and the audio timeline. While, in this basic form, the additional degree of freedom in the alignment can lead to measurable improvements in alignment accuracy on a fine level, it can also cause a loss of accuracy on a coarser, global level. Therefore, to exploit the overall robustness of existing alignment methods, we employ a state-of-the-art method to compute a coarser alignment in a first step, which is then used to constrain and guide the alignment in our proposed method. This way, our method not only combines the robustness of current methods with an improved alignment accuracy, but also drastically lowers the computational cost for computing a three-dimensional alignment (given the guiding alignment, from cubic to linear in the length of the recording or score).

The paper is organized as follows. Technical details of our method are described in Section 2. We report on some of our experiments in Section 3. Conclusions and discussions of future work are given in Section 4. Related work is discussed in the respective sections.

2. ALIGNMENT METHOD

A general procedure to synchronise a score and a performance can be summarized in three simple steps. First, the score and the audio are converted to a suitable, common feature representation. Second, by comparing each element in the score feature sequence with each element in the audio sequence using a distance measure, one obtains a distance or cost matrix. Third, based on such a matrix, a synchronisation method is applied to obtain a cost-minimizing alignment. In this context, various alignment methods have been proposed, including Dynamic Time Warping (DTW) [15], Hidden Markov Models (HMM) [16], Conditional Random Fields (CRF) [11], general graphical models [17], and Particle Filter / Monte-Carlo Sampling (MCS) based methods [3, 5]. While all these approaches typically yield robust alignments, none of them accounts for asynchronies between voices. An exception was presented in [18] but only for aligning MIDI files. Further, in [19], a greedy, post-processing method is introduced, which locally refines the alignment on a note level.

To model possible asynchronies between voices, we need to modify all three steps of the procedure above. First, the score can no longer be treated as a single data stream. Instead, the voices have to be isolated from the score and features have to be derived for each voice separately. Second, the comparison of features from all three sequences leads to a three-dimensional cost matrix (or cost tensor). Third, an extended alignment method is needed, which is able to deal with three sequences. Interpreting the alignment as a multi-dimensional data series synchronisation problem leads to two existing methods: the Asynchronous Hidden Markov Model (AHMM) [20] and the Multi-Dimensional Dynamic Time Warping...
Fig. 1. A three-dimensional cost tensor. (a) Three-dimensional alignment path of the melody (Mel), accompaniment (Acc) and audio; (b) Projections of the path (black) onto x-z (red), y-z (blue) and x-y (green) planes.

(MD-DTW) [21]. These methods have been applied to various problems, including audio-visual speech recognition, and in particular for bi-modal speech and gesture fusion [22]. Both approaches share similar algorithmic roots (dynamic programming). In the following, we introduce our method as an extension to MD-DTW.

2.1. Computing Features for Individual Voices

While a musical score can often be separated into various voices, we focus in the following on the melody and accompaniment parts. From a musical point of view, these voices are particularly important for us as asyncrony between them has been reported and analysed in musicological studies [14]. Also from a numerical point of view, we focus in the following on the melody and accompaniment parts.

We separate the melody and the accompaniment notes from the score using the skyline algorithm [23], which can be replaced by more complicated methods, such as the contig mapping [24], in future work. Once separated, the feature computation itself is essentially identical to previous methods. We compute the feature sequences \(X := (x_1, x_2, \ldots, x_K)\) and \(Y := (y_1, y_2, \ldots, y_K)\) for the two score voices as well as \(Z := (z_1, z_2, \ldots, z_K)\) for the audio, with \(x_n, y_m, z_l \in \mathcal{F}\) where \(\mathcal{F}\) is a space containing two types of features similar to the approach described in [12]. The first type is a 88-dimensional log-frequency feature, whose entries encode a short-time intensity in spectral bands with centre frequencies corresponding to the 88 keys on a grand piano, see [25, 26] for information on how to derive such features from audio and MIDI representations. Additionally, we include a second 88-dimensional feature type which indicates possible onset positions separately for each key, see [12] for details. As shown previously [12], such a combination of features can lead to a substantial increase in alignment accuracy.

2.2. Three-Dimensional Dynamic Time Warping

In previous alignment approaches, each element of one feature sequence is compared with that of another sequence, which results in a cost matrix. With three feature sequences, we now extend this idea to a three dimensional cost tensor, see also Fig 1(a). More precisely, given the three feature sequences\(X, Y\) and \(Z\), we define a \((K \times K \times L)\) cost tensor \(C\) by 
\[ c(x_n + y_m + z_l) \]
where \(c : F \times F \times F \rightarrow \mathbb{R}_{\geq 0}\) denotes a local cost measure on \(F\). For \(n \neq m\) we combine a melody and an accompaniment feature from different positions in the two score timelines into a single score feature, which is then compared to the audio feature. In this case, the difference \(n - m\) encodes the asynchrony between the two voices. In particular, the diagonal plane in the cost tensor \(C(n, n, \ell)\) for \(n \in [1 : N]\) and \(\ell \in [1 : L]\) is essentially identical to a cost matrix between the complete score and the audio as used in classical two-dimensional DTW. All entries in the cost tensor on planes parallel to the diagonal plane have the same asynchrony between the two score voices (i.e. \(n - m\) is constant), compare Fig 2(a).

An alignment between \(X, Y\) and \(Z\) is defined as a sequence \(p = (p_1, \ldots, p_Q)\) with \(p_q = (n_q, m_q, \ell_q) \in [1 : K] \times [1 : K] \times [1 : L]\) for \(q \in [1 : Q]\) satisfying \(p_1 = (1, 1, 1)\) and \(p_q = (K, K, L)\) as well as \(p_{q+1} - p_q \in \{(1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 1, 0), (1, 0, 1), (0, 1, 1), (1, 1, 1)\}\) (step size condition). An alignment through the cost tensor \(X, Y\) and \(Z\) is illustrated in Fig 1(a).

The cost of an alignment is defined as \(\sum_{q=1}^Q C(n_q, m_q, \ell_q)\) and an alignment having minimal cost among all possible alignments is called an optimal alignment. To determine such an optimal alignment, one can employ MD-DTW [21]. In summary, one recursively computes a \((K \times K \times L)\)-tensor \(D\) where the entry \(D(n, m, \ell)\) is the cost of an optimal alignment between \((x_1, \ldots, x_n), (y_1, \ldots, y_m)\) and \((z_1, \ldots, z_l)\). Using dynamic programming, this tensor can be recursively computed as follows:

\[
D(n, m, \ell) := \begin{cases} 
D(n - 1, m, \ell) + w_1 C(n, m, \ell), \\
D(n, m - 1, \ell) + w_2 C(n, m, \ell), \\
D(n, m, \ell - 1) + w_3 C(n, m, \ell), \\
D(n - 1, m - 1, \ell) + w_4 C(n, m, \ell), \\
D(n - 1, m, \ell - 1) + w_5 C(n, m, \ell), \\
D(n - 1, m - 1, \ell - 1) + w_6 C(n, m, \ell), \\
\end{cases}
\]

for \(n, m, \ell > 1\). Furthermore, \(D(n, 1, 1) := \sum_{k=1}^n w_1 C(k, 1, 1)\) for \(n > 1\), \(D(1, m, 1) := \sum_{k=1}^m w_2 C(1, k, 1)\) for \(m > 1\), \(D(1, 1, l) := \sum_{k=1}^l w_3 C(1, 1, k)\) for \(l > 1\), and \(D(1, 1, 1) := C(1, 1, 1)\). Calculations of entries on the \(x-y\), \(x-z\) and \(y-z\) planes, i.e., \(D(n, m, 1), D(n, 1, l)\) and \(D(1, m, \ell)\), are equivalent to the accumulated cost matrix calculation in classical two-dimensional DTW [21]. The weights \((w_1, w_2, w_3, w_4, w_5, w_6, w_7) \in \mathbb{R}_+^7\) can be set to adjust the preferences for the seven step sizes. Note that a bias for any direction is removed by setting these weights to \((w_1, w_2, w_3, w_4, w_5, w_6, w_7) = (1, 1, 1, 2, 2, 2, 3)\). An optimal alignment is obtained by tracing the minimizing argument backwards from \(D(K, K, L)\) to \(D(1, 1, 1)\). Its projections onto the \(x-z\) and \(y-z\) planes yield alignments between the melody and the audio as well as the accompaniment and the audio, respectively. The projection onto the \(x-y\) plane corresponds to an alignment between the two score voices and thus encodes the estimated local asynchrony between them, see Fig. 1(b).

2.3. Path Constraints

In principle, an asynchronous alignment could be computed using MD-DTW as described above. In practice, however, there are additional factors which render this approach practically infeasible. On the one hand, the computational complexity of MD-DTW is considerable. Assuming the sequences to be aligned are roughly of the same length \(L\), the memory and time complexity of an N-dimensional dynamic programming algorithm is \(O(L^N)\) and \(O(2N L^N)\), respectively [27]. Since our application requires a high temporal resolution for the features, the value of \(L\) is typically high and the alignment becomes practically infeasible even for pieces of average length. On the other hand, splitting the score into two independent voices results in the number of notes in each voice to...
be lower compared to the full score. This becomes a problem, if the remaining notes do no provide enough information to be discriminative in time. For example, if a chord is repeated consecutively in the accompaniment, an asynchronous alignment might easily confuse one instance of the chord for another, resulting in a substantial alignment error. We refer to this issue as the loss-of-structure problem in the following. Note that previous approaches will not suffer from this issue if the melody is discriminative enough.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.

Asynchrony Constraints

With the first constraint we account for the observation that asynchronies between musical voices are in practice not arbitrarily high [14]. Musicians typically employ asynchronies to highlight certain elements in a piece, and if used in an extreme way, the asynchrony might render the piece unrecognisable by the audience. To limit the amount of asynchronies between voices to a musically meaningful level, we obtain a ground truth alignment between the audio and the performance MIDI with the corresponding score MIDI on a note onset in the score the corresponding position in the audio. By comparing Section 2.2. More precisely, in order to compute the diagonal plane features are combined without any asynchrony between voices, surrounded by two parallel planes corresponding to regions with constant asynchrony. (b) The alignment is constrained to run in a neighbourhood of a reference alignment, illustrated only on the diagonal plane.
Table 1. Experimental results for three pieces played with strong asynchrony (upper) and three pieces without asynchrony (lower). This table shows the number of performances available and statistics over the alignment error in milliseconds for the respective pieces. Both results for the 2D-DTW [12] and our 3D-DTW alignment method are computed separately for the melody (Mel) and accompaniment (Acc). The error values of these two voices are averaged over the number of notes to get the overall (OA) alignment error.

<table>
<thead>
<tr>
<th>Piece</th>
<th>No. Rec</th>
<th>2D-DTW [12]</th>
<th>3D-DTW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mel</td>
<td>Acc</td>
</tr>
<tr>
<td>w/o Asyn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BWV 848</td>
<td>3</td>
<td>11</td>
<td>14</td>
</tr>
<tr>
<td>BWV 849</td>
<td>2</td>
<td>21</td>
<td>29</td>
</tr>
<tr>
<td>BWV 889</td>
<td>2</td>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>14</td>
<td>19</td>
</tr>
<tr>
<td>w/ Asyn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Op. 10/3</td>
<td>22</td>
<td>16</td>
<td>23</td>
</tr>
<tr>
<td>Op. 28/15</td>
<td>5</td>
<td>16</td>
<td>45</td>
</tr>
<tr>
<td>Op. 48/1</td>
<td>4</td>
<td>27</td>
<td>64</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>18</td>
<td>31</td>
</tr>
</tbody>
</table>

Fig. 3. Comparison of the 2D-DTW alignment results with our 3D-DTW alignment results. The boxplots illustrate the distribution of the alignment results in milliseconds for each piece separately.

The overall alignment error for the three pieces with strong asynchrony, drops from 27ms using 2D-DTW alignment to 22 ms using 3D-DTW alignment on average (decreases by 19%). This drop can also be seen from the boxplots in Fig 3, which show the distribution of the alignment error for all score-audio pairs for the three pieces. Note that the above results were obtained by separating the melody and accompaniment notes from the score using the skyline algorithm. Compared with results obtained using a manual separation, the overall alignment error remained the same on average.

4. CONCLUSION AND FUTURE WORK

In this paper, we introduced a score-audio alignment method that can compensate for an asynchrony between the melody and accompaniment. A 3D-DTW algorithm was employed in which the two score voices are treated as independent timelines. Further, the alignment was constrained by a guiding alignment obtained via a classical 2D-DTW, providing improved robustness and a reduced computational complexity. Our experiments demonstrated that our proposed method can indeed improve the alignment accuracy for pieces with strong asynchrony and preserves the accuracy otherwise, compared to a previously proposed alignment method using classical DTW.

As a by-product, the resulting alignment can be used to indicate the positions where asynchrony occurs. In initial experiments, our method achieved a precision of 0.44 and recall of 0.58 on average in detecting positions with strong asynchrony. In the future, we plan to further investigate how to improve the performance of our method in this respect in order to develop an assistive tool for musical expression analysis. Furthermore, we will also apply multi-dimensional DTW to different asynchronous data stream alignment problems, such as the asynchrony between different instruments in a musical ensemble.

5We use standard boxplots: the red bar indicates the median, the blue box gives the 25th and 75th percentiles (p25 and p75), the black bars correspond to the smallest data point greater than p25 − 1.5(p75 − p25) and the largest data point less than p75 + 1.5(p75 − p25). The red crosses are called outliers.
5. REFERENCES


