MATCH: A Music Alignment Tool Chest

Simon Dixon
Austrian Research Institute for Artificial Intelligence
Freyung 6/6
Vienna 1010, Austria
simon@oefai.at

Gerhard Widmer
Department of Computational Perception
Johannes Kepler University Linz
Altenberger Str 69, A-4040 Linz, Austria
gerhard.widmer@jku.at

ABSTRACT

We present MATCH, a toolkit for aligning audio recordings of different renditions of the same piece of music, based on an efficient implementation of a dynamic time warping algorithm. A forward path estimation algorithm constrains the alignment path so that dynamic time warping can be performed with time and space costs that are linear in the size of the audio files. Frames of audio are represented by a positive spectral difference vector, which emphasises note onsets in the alignment process. MATCH was tested on several hundred test cases and had a median error of 20 ms, with less than 1% of test cases failing to align at all. The software is useful for content-based indexing of audio files and for the study of performance interpretation; it can also be used in real-time for tracking live performances. Another possible application is in an intelligent audio recording and editing system for aligning splice points. The toolkit also provides functions for displaying the cost matrix, the forward and backward paths, and any metadata associated with the recordings, which can be shown in real time as the alignment is computed.

Keywords: audio alignment, content-based indexing, dynamic time warping, music performance analysis

1 INTRODUCTION

The use of random access media for audio data, making it possible to jump immediately to any point in the data, is advantageous only to the extent that the data is indexed. For example, content-based indexing of CDs is typically limited to the level of tracks (songs or movements), the information provided by the manufacturer. The indexing cannot be determined by the user, who might be interested in a more fine-grained or special purpose index. For example, a piano student might want to compare how several different pianists play a particular phrase, which would involve a manual linear search for the relevant phrase in each recording. Or alternatively, a musicologist studying the relationship between tempo and phrase structure painstakingly marks the times of beats in each rendition of a work, not having any way of transferring the metadata from one version to the next, since the beats occur at different times in each performance.

To address these and similar needs, we developed MATCH, a system for accurate automatic alignment of multiple renditions of the same piece of music. This tool can be used in musicology and music practice, to compare different interpretations of a work, or for annotation of music with content-based metadata (e.g. section, phrase, beat or note indexes), which could then be transferred automatically from one recording to the corresponding positions in another recording. It could also be used in an audio recording system to provide intelligent editing operations such as aligning splice points in corresponding files. The toolkit also provides for visualisation of the alignment as it is computed.

MATCH is based on an efficient dynamic time warping algorithm which has time and space costs that are linear in the lengths of the performances. This effectively allows arbitrarily long pieces to be processed faster than real time, that is, in less time than the duration of the audio files. The audio data is represented by positive spectral difference vectors. Frames of audio input are converted to a frequency domain representation using a short time Fourier transform, and then mapped to a non-linear frequency scale (linear at low frequencies and logarithmic at high frequencies). The time derivative of this spectrum is then half-wave rectified and the resulting vector is employed in the dynamic time warping algorithm's match cost function, using a Euclidean metric.

In the next section, we review the standard dynamic time warping algorithm and decribe the modifications necessary for an efficient implementation. We also present the cost function used to evaluate the similarity of frames of audio data. Section 3 contains a description of the user interface and implementation details of MATCH. We then report on the results of testing with three different data sets, which indicate that the current audio alignment algorithm works well for a range of music. The final section provides a discussion of the work, a comparison with other audio alignment methods, and an outline of planned future work.

2 DYNAMIC TIME WARPING

Dynamic time warping (DTW) is a technique for aligning time series which has been well known in the speech recognition community since the 1970's (Itakura, 1975). In this section we briefly review the DTW algorithm and describe how we apply it to the problem of audio alignment. A tutorial presentation of DTW can be found in (Rabiner and Juang, 1993).

DTW aligns two time series $U = u_1, ..., u_m$ and $V = v_1, ..., v_n$ by finding a minimum cost path $W = W_1, ..., W_l$, where each W_k is an ordered pair (i_k, j_k), such that $(i, j) \in W$ means that the points u_i and v_j are aligned. The alignment is assessed with respect to a local cost function $d_{U,V}(i, j)$, usually represented as an $m \times n$ matrix, which assigns a match cost for aligning each pair (u_i, v_j). The cost is 0 for a perfect match, and is otherwise positive. The path cost $D(W)$ is the sum of the local match costs along the path:

$$D(W) = \sum_{k=1}^{l} d_{U,V}(i_k, j_k)$$

Several constraints are placed on W, namely that the path is bounded by the ends of both sequences, and it is monotonic and continuous. Formally:

Bounds:	$W_1 = (1, 1)$
	$W_l = (m, n)$
Monotonicity:	$i_{k+1} \geq i_k$ for all $k \in [1, m-1]$
	$j_{k+1} \geq j_k$ for all $k \in [1, n-1]$
Continuity:	$i_{k+1} \leq i_k + 1$ for all $k \in [1, m-1]$
	$j_{k+1} \leq j_k + 1$ for all $k \in [1, n-1]$

Other local path constraints are also common, which alter the monotonicity and continuity constraints to allow increments of up to two or three steps in either direction and/or require a minimum of at least one step in each direction. Additionally, global path constraints are often used, such as the Sakoe-Chiba bound (Sakoe and Chiba, 1978), which constrains the path to lie within a fixed distance of the diagonal (typically 10% of the total length of the time series), or the Itakura parallelogram (Itakura, 1975), which bounds the path with a parallelogram whose long diagonal coincides with the diagonal of the cost matrix. By limiting the slope of the path, either globally or locally, these constraints prevent pathological solutions and reduce the search space.

The minimum cost path can be calculated in quadratic time by dynamic programming, using the recursion:

$$D(i, j) = d(i, j) + \min \left\{ \begin{array}{l} D(i, j-1) \\ D(i-1, j) \\ D(i-1, j-1) \end{array} \right\}$$

where $D(i, j)$ is the cost of the minimum cost path from $(1, 1)$ to (i, j), and $D(1, 1) = d(1, 1)$. (The subscripts on $d(.,.)$ have been omitted for convenience.) The path itself is obtained by tracing the recursion backwards from $D(m, n)$.

Some formulations of DTW introduce various biases in addition to the slope constraints, by multiplying $d(i, j)$ by a weight which is dependent on the direction of the movement. In fact, the above formulation is biased towards diagonal steps: the greater the number of diagonal steps, the shorter the total path length (Sankoff and Kruskal, 1983, p.177). We follow Sakoe and Chiba (1978) in using a weight of 2 for diagonal steps so that there is no bias for any particular direction.

2.1 An Efficient Implementation of DTW

The quadratic time and space cost is often cited as a limiting factor for the use of DTW with long sequences. However the widely used global path constraints can be trivially modified to create a linear time and space algorithm. For instance, if the width of the Sakoe-Chiba bound is set to a constant rather than a fraction of the total length, the number of calculations becomes linear in the length of the sequences. The danger with this approach is that it is not known how close to the diagonal the optimal solution is, so the desired solution is easily excluded by a band around the diagonal which is too narrow.

To avoid missing the optimal path, we use a forward path estimation algorithm to calculate the centre of the band of the cost matrix which is to be calculated. This is based on the on-line time warping algorithm presented in (Dixon, 2005). The DTW path is then constrained to lie within a fixed distance of the forward path, which ensures that the computation is bounded by linear time and space costs. If we had used standard global path constraints, a wider band would have been required, in order to cater for the estimated maximum possible deviation from the diagonal. With an "adaptive diagonal", it is safe to use a narrower band without risk of missing the optimal solution. This enables the system to perform with greater efficiency and accuracy than a system based on global path constraints.

The calculation of the forward path is illustrated in Figure 1, using a band width of $w = 4$ for illustrative purposes. (In practice, a band width of $w = 500$ is used.) After an initial square matrix (of size w) is computed, the calculated area is expanded by evaluating rows or columns of length w. The direction of expansion (i.e. whether a new row or a new column is calculated) is determined by the location of the cell with minimum path cost out of all the cells in the last row or column of the calculated area. If this cell is in the last row, a new row is calculated, and if it is in the last column, a new column is calculated. Limits are placed on the number of successive row (respectively column) computations. The band width w limits the number of cells in the row or column computation to the last w cells. A complete description of the forward path algorithm can be found in (Dixon, 2005). When the ends of both files are reached, the optimal path is traced backwards using the standard DTW algorithm, constrained by the fact that only the cells calculated previously during the forward path calculation can be used.

2.2 A Cost Function for Comparing Audio Frames

The alignment of audio files is based on a cost function which assesses the similarity of frames of audio data. We use a low level spectral representation of the audio data, generated from a windowed FFT of the signal. A Ham-

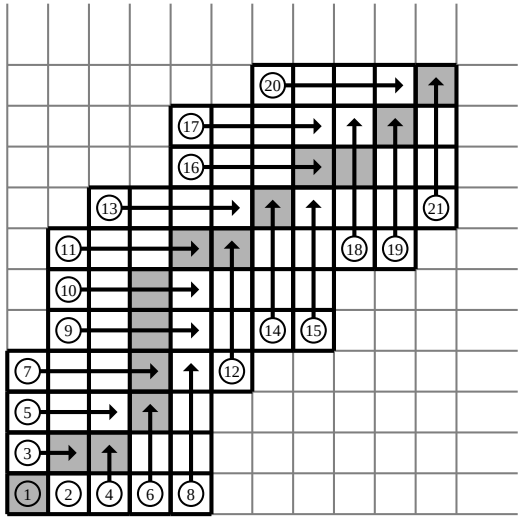

Figure 1: *An example of the on-line time warping algorithm with band width $w = 4$, showing the order of evaluation for a particular sequence of row and column increments. The axes represent time in the two files. All calculated cells are framed in bold, and the optimal path is coloured grey.*

ming window with a default size of 46 ms (2048 points) is used, with a default hop size of 20 ms. The spectral representation was chosen over a higher level symbolic representation of the music in order to avoid a pitch recognition step, which is notoriously unreliable in the case of polyphonic music. The frequency axis was mapped to a scale which is linear at low frequencies and logarithmic at high frequencies. This achieved a significant data reduction without loss of useful information, at the same time mimicking the linear-log frequency sensitivity of the human auditory system. The lowest 34 FFT bins (up to 370Hz, or F♯4) were mapped linearly to the first 34 elements of the new scale. The bins from 370Hz – 12.5kHz were mapped onto a logarithmic scale with semitone spacing by summing energy in each bin into the nearest semitone element. Finally, the remaining bins above 12.5kHz (G9) were summed into the last element of the new scale. The resulting vector contained a total of 84 points instead of the original 2048.

The most important factor for alignment is the timing of the onsets of tones. The subsequent evolution of the tone gives little information about the timing and is difficult to align using energy features, which change relatively slowly over time within a note. Therefore the final audio frame representation uses a half-wave rectified first order difference, so that only the increases in energy in each frequency bin are taken into account, and these positive spectral difference vectors are compared using the Euclidean distance:

$$d(i,j) = \sqrt{\sum_{b=1}^{84} (E'_u(b,i) - E'_v(b,j))^2}$$

where $E'_x(f,t)$ represents the increase in energy $E_x(f,t)$ of the signal $x(t)$ in frequency bin f at time frame t:

$$E'_x(f,t) = \max(E_x(f,t) - E_x(f,t-1), 0)$$

2.3 Interpretation of the DTW Path

The path returned by the DTW alignment algorithm is used as a lookup table between the two audio files to find the location in the second file corresponding to a selected location in the first file. Since the path is continuous and covers the full extent of both files, there is for each time index in one file at least one corresponding time point in the other. If there is more than one corresponding point, an average is taken. This defines a one to one mapping between the time variables in the two files, with the resolution of the frame hop size.

3 USER INTERFACE AND IMPLEMENTATION

The system operates as follows: the user starts up the program and loads the files to be aligned. The first file to be loaded is used as the reference file, and as more files are loaded, each is aligned to the reference file. Corresponding time points between arbitrary pairs of files can then be computed via the reference file, using composition of the respective time maps.

MATCH has a familiar graphical user interface which is similar to most media players, having buttons to play, stop, pause, skip forward, skip backward, load, show help information and quit, as well as a slider for setting and displaying the present playback position. A screen shot of the program is shown in Figure 2. One unfamiliar function (the "*" button) is to mark a position in a piece, which may be done while the piece is playing or when it is stopped. Any number of marks can be inserted into the piece; the skip buttons allow the user to move quickly between marks. Marks can be used to select interesting phrases in a piece of music, or to test the operation of the alignment algorithm. When the user switches to a different file, the marks are mapped to the corresponding locations in the new file, so that the user can compare how the section of interest was played in each version.

If the user changes files while playing, the system's response depends on the play mode which was selected, which is either *continue* or *repeat*. In the *repeat* mode, play continues from the first mark before the current position (mapped onto the new file). In this mode the user can listen to the same phrase in different versions simply by clicking on the file name. In the *continue* mode, the piece continues at the corresponding position in the new file, allowing the user to combine different interpretations. This mode is most useful for testing and demonstrations of the system.

MATCH also has functions for displaying the cost matrix, the forward and backward paths, and any other metadata associated with the files (see Figure 3). Since the complete matrix does not fit on the screen, scroll functions are provided. The audio from one file can be played as matching is performed, with the matrix scrolling in real time and displaying the alignment as it is being estimated.

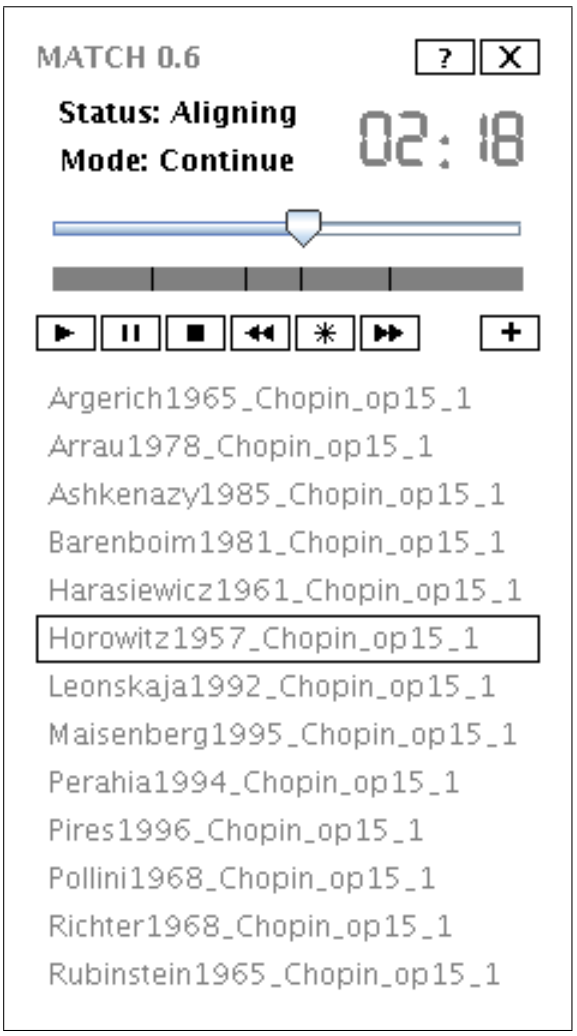

Figure 2: Screenshot of MATCH showing the user interface.

MATCH is implemented in Java, so it runs on most operating systems. On a 3GHz Linux PC, alignment of two audio files takes approximately 10% of the sum of the durations of the files, where a time resolution of 20 ms is used. For slower computers a lower time resolution could be used without significant degradation of precision. It is also possible to save and load alignment and mark information, so the user can restore a session without having to wait for any alignment computations. MATCH will be made available for download before the conference.

4 TESTING AND RESULTS

We report the results from 3 sets of test data: a precise quantitative evaluation using data recorded on a Bösendorfer computer-monitored piano; a quantitative evaluation based on semi-automatic annotation of various CD recordings; and a qualitative evaluation based on unannotated CD recordings.

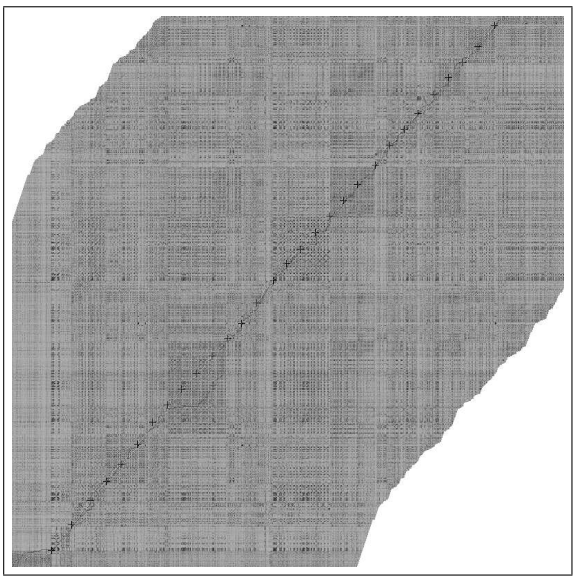

Figure 3: Screenshot of the initial 16 seconds of the cost matrix for two renditions of a Chopin Etude, showing the forward path used for determining the expansion direction (black), the optimal path (white), and the correct event onset times (metadata) as grey crosses.

4.1 Bösendorfer Data

The Bösendorfer SE290 is a grand piano with sensors on the keys and hammers which unintrusively measure the precise timing and dynamics of every note. The time resolution of 1.25 ms allows for extremely accurate evaluation of audio analysis algorithms. The only disadvantage is that the available data is restricted to solo piano music which was recorded on this instrument. In other words, this test set is the highest quality but also the most limited in scope.

We used a set of recordings of 22 pianists playing 2 excerpts of solo piano music by Chopin (Etude in E Major, Op.10, no.3, bars 1–21; and Ballade Op.38, bars 1–45) (Goebl, 2001). The Etude performances ranged from 70.1 to 94.4 seconds duration, and the Ballade ranged from 112.2 to 151.5 seconds, so the differences in execution speeds were by no means trivial. Alignment was performed on all pairs of performances of each piece (a total of $\frac{22\times21}{2} * 2 = 462$ test cases).

In order to estimate the correctness of the alignment, we compared it with the onset times of the corresponding notes in each interpretation. If we consider the alignment as a mapping from time in one interpretation to time in the other interpretation, a correct alignment should map the onset time of each note in the first interpretation to the onset time of the same note in the second interpretation. There are several reasons why this is not always possible.

First, it is clear that the mapping should be monotonic, but the correct mapping as we just defined it could not be guaranteed to be monotonic, except for monophonic music. The reason for this is that chords (sets of simultaneous notes according to the musical notation) are not played perfectly synchronously, but are spread over time, typically around 30 ms, but sometimes as much as 150

ms. Further, the order of notes in the chord is not fixed, although some regularities (such as the melody note leading the accompaniment notes) have been observed (Goebl, 2001).

The second issue is that there are occasional differences in the notes played between different interpretations, either due to one musician making a mistake, or due to the interpretative freedom given by a composer. A typical case is that of a trill (a rapid alternation of two notes), where the total musical time is specified, but not the number of alternations to be performed in this time. In this case there is no unique "correct" alignment of the notes involved. Other types of ornamentation provide further examples where the number or order of notes might differ between interpretations.

Third, a problem arises from our representation of alignments, that the paths are not functions but relations, and thus the mapping is not necessarily unique. A number of solutions to this problem are possible, for example averaging or smoothing the path to create a one-to-one mapping, but these turned out to be unnecessary, as the evaluation method we define below circumvents the problem.

For these reasons, we define a *score event* to be a set of simultaneous notes according to the score, and for each interpretation i we calculate the average onset time $t(i, e)$ of the performed notes in each score event e. The correct alignment is then defined in terms of the accuracy of the mapping of score events from one interpretation to the other, ignoring the time points between score events. For each score event e, the alignment path should pass through the point $(t(i_1, e), t(i_2, e))$, and the error is calculated as the Manhattan distance of this point from the nearest point on the alignment path. The total error of an alignment path is then the average of the pointwise errors over all score events.

Table 1 shows the distribution of pointwise errors less than or equal to 0,1,2,3,5,10,25 and 50 frames, where a frame size of 20 ms was used. The average and worst case errors are also shown. From the point of view of human perception, the average error is almost imperceptible, since the human temporal order threshold (the ability to distinguish the order of two sounds occurring closely in time) is approximately 40 ms, and can be much worse in the context of annotating musical recordings (Dixon et al., 2005). The success of the system with this data was aided by the fact that the audio recordings were all made under identical conditions (same piano, microphone, room and settings). In the following subsections we describe tests using data with a large variety of recording conditions.

Error $\leq$		Cumulative %age	
Frames	Seconds	Etude	Ballade
0	0.00	46.5%	36.6%
1	0.02	84.5%	77.1%
2	0.04	91.1%	88.9%
3	0.06	93.4%	92.5%
5	0.10	96.0%	95.1%
10	0.20	98.8%	97.4%
25	0.50	99.8%	99.1%
50	1.00	100.0%	99.8%
Average Error		23 ms	35 ms
Worst Error		2340 ms	3640 ms

Table 1: Alignment results shown as cumulative percentages of score events with an error up to the given value (see text).

4.2 BeatRoot Data

The second set of test data involved musical pieces where the beat had been annotated using the interactive beat tracking system BeatRoot (Dixon, 2001a,b). The data set available to us is considerably larger than the Bösendorfer data set, containing a range of Classical and Romantic Period piano music recorded over the second half of the twentieth century, including several complete piano sonatas. These works include more complex pieces than the Chopin excerpts mentioned above. The disadvantage with this data is that the measurements of beat times are much less precise, as they are generated by the beat tracking system using a simple onset detection algorithm and then (optionally) corrected manually. The error in measurement is probably of the order of 30 ms resolution (but this would be difficult to verify). Further, the annotations contain the beat times, not the times of note onsets, so measurements do not exist for every note and conversely some beats are interpolated in the cases where there is no note on the beat.

The data was taken from CD recordings of great pianists. For example, for the Chopin piece we used the following recordings: Argerich (1965), Arrau (1978), Ashkenazy (1985), Barenboim (1981), Harasiewicz (1961), Horowitz (1957), Leonskaja (1992), Maisenberg (1995), Perahia (1994), Pires (1996), Pollini (1968), Richter (1968) and Rubinstein (1965). This enabled the system to be tested with a large range of recording conditions, pianos, pieces and interpretations.

The results are summarised in Table 2. It is difficult to reduce any set of results to a single numerical value, and these results illustrate this point well. Three different measures of error are shown: maximum, mean and median. The maximum error indicates whether or not at least one of the alignments failed at some point, but gives no indication of overall performance. The mean error is more useful, but it is strongly biased by large errors. (In order to demonstrate the amount of this bias, the lower half of the table contains the same results with the worst aligning pair of performances excluded, or in the case of the Schumann excerpt, the two worst pairs. Some of the average errors were reduced by more than 80%.) The median error is a less biased measure of central tendency, but it gives no indication of the spread of the errors. Figures 4 and 5 show histograms of the errors respectively for the complete test set and the test set with the 5 worst alignments removed, where the rightmost bin represents all errors greater than one second.

Apart from a small number of cases (5 out of 175) where alignment fails, the results are very pleasing, although as expected, they are not as good as for the Bösendorfer data where the controlled recording condi-

Composer	Piece (work, section)	Versions	Events (total)	Error (sec)		
				Maximum	Mean	Median
Beethoven	Op.15, 2, b1–8	4	366	2.46	0.109	0.040
Chopin	Op.15, No.1	13	17082	7.28	0.054	0.020
Mozart	KV279, 1st movt	5	5510	15.26	0.189	0.020
Schubert	D899, No.3	12	22506	59.92	0.584	0.020
Schumann	Op.15, No.7	6	3825	20.34	0.427	0.020
Beethoven	Op.15, 2, b1–8	4*	305	1.10	0.083	0.040
Mozart	KV279, 1st movt	5*	4959	6.56	0.048	0.020
Schubert	D899, No.3	12*	22165	12.98	0.086	0.020
Schumann	Op.15, No.7	6**	3315	2.06	0.062	0.020

Table 2: Alignment results for CD recordings of the given works. '*' indicates the removal of the worst alignment from the results. See text for discussion.

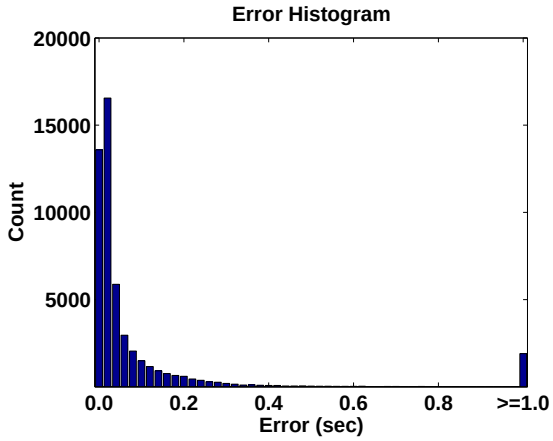

Figure 4: Histogram showing distribution of alignment errors for the complete BeatRoot data set. The rightmost bin represents the number of errors greater than or equal to one second.

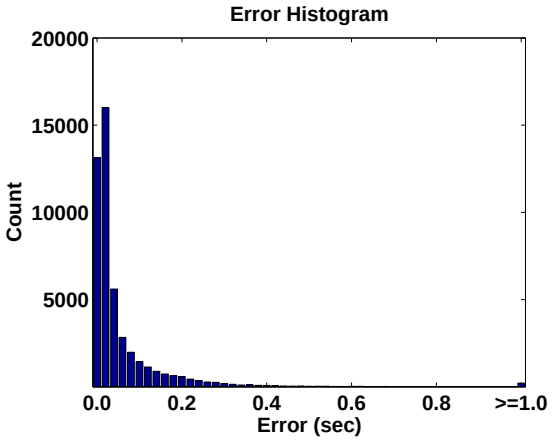

Figure 5: Histogram showing distribution of alignment errors on the BeatRoot data set, where the five worst alignments have been removed. The rightmost bin represents the number of errors greater than or equal to one second.

tions make similarity judgements much easier. The spread of errors indicates a low probability of large errors for this data, so that users of MATCH will rarely face serious misalignments. An analysis of the errors and some ideas for improvements to the system are discussed in the final section.

4.3 Further Evaluations

The above tests consisted only of piano music, which is generally easier to align than other instruments, due to the sharpness of onsets and the fixed timbre of piano tones. Although we do not have any annotated non-piano music, it was possible to perform informal tests with other pieces of music. In the absence of precise annotations, there are two ways of determining the correctness of an alignment using MATCH. The first is to mark various points in one interpretation and check that the corresponding points in the other interpretations are correctly aligned, and the second method is to skip between interpretations during playback, which also gives some indication of the alignment. Neither method is particularly precise, in that they rely on human judgement, but since the system is designed to be used in both of these ways, they are suitable for qualitative testing. A disadvantage of this form of testing is that it is not thorough, that is, only specific points on the alignment path are checked, and not the complete path.

Tests with 10 different interpretations of the first movement of Schumann's piano concerto revealed no problems in alignment. Several other works for piano and orchestra were also tested with good results. A number of solo classical guitar works were tested, including pieces by Albeniz (Asturias, Cordoba, Sevilla), Granados (Spanish Dance 4 and 5), Tarrega (Capricho Arabe) and Villa Lobos (Prelude 1). Some of the pieces were successfully aligned, although errors at the beginnings of a number of pieces were apparent. Further work is required to investigate the reasons for these errors.

It is much more difficult to find different versions of popular songs where the structure is identical in all versions. Two Beatles songs (*I Wanna Hold Your Hand* and *She Loves You*) were found in English and German versions; MATCH successfully aligned the first song entirely, but the second was partially misaligned, with the initial section being wrong but the bulk of the song correctly aligned. These tests suggest that the similarity measure is not restricted to piano tones, but could be generally applicable to different instruments.

5 DISCUSSION AND CONCLUSION

This paper presented an audio alignment toolkit which uses dynamic time warping based on a low-level representation of the audio data. A high-level representation would enable a more efficient DTW computation, but extraction of high level features is less reliable then using low-level features, and since efficiency is not a problem in the current approach, we plan to stay with the more robust approach.

The cost function was based on derivative spectral features, in order to emphasise tone onsets. Derivative features have been used in speech recognition (Sakoe and Chiba, 1978), are advocated generally by Keogh and Pazzani (2001), and they have been used in score following (Orio and Schwarz, 2001). A distance measure calculated directly from the short time spectrum was used for computing audio similarity in (Foote and Uchihashi, 2001). This used a much smaller window size (11 ms), since it was focussed on rhythmic analysis, where timing is critical and pitch not so important. In tests using spectral values instead of the spectral difference, we found that the results were clearly better using spectral difference.

Dannenberg and Hu (2003) propose the use of a chromogram, which reduces the frequency scale to twelve pitch classes, independent of octave. This might be suitable for retrieval by similarity, where absolute identity of matching musical pieces is not assumed, and a large number of comparisons must be performed in a short time, but it discards too much information for our purposes. Other features such as MFCCs are often used in speech and audio research, but they capture the spectral shape (reflecting the timbre of the instrument) rather than the pitch (reflecting the notes that were played).

DTW has been used for score-performance alignment (Orio and Schwarz, 2001; Soulez et al., 2003) and query by humming applications (Mazzoni and Dannenberg, 2001; Zhu and Shasha, 2003). The earliest score following systems used dynamic programming (Dannenberg, 1984), based on a high-level symbolic representation of the performance which was only usable with monophonic audio. Alternative approaches to music alignment use hidden Markov models (Cano et al., 1999; Orio and Déchelle, 2001) and hybrid graphical models (Raphael, 2004), which both require training data for each piece. The test data used in subsections 4.1 and 4.2 is somewhat exceptional; in general, we will not have access to multiple labelled performances.

In cases where the tracking was successful, it was also very accurate, with a median error of 20 ms, and average errors from 23 ms to 86 ms. For the purposes of the alignment tool, this is easily sufficient. For other purposes, such as audio editors, it is essential that the error be kept as low as possible.

5.1 Future Work

There are many directions in which this work can be extended and many improvements which can be made to the current system, some of which we note here. Experiments with normalisation have proved it to be a double-edged sword. Since we have no control of recording levels, some form of normalisation between files is essential. The frame to frame normalisation of energy is however more problematic, since it is more important that salient parts of the audio match, and as notes decay to silence, it is not desirable that they play an equally significant role as the tone onsets in determining the alignment. The use of positive spectral difference solves part of this problem, but further experimentation is required to determine the best audio representation.

The output from the DTW algorithm is not at all smooth at the local level, but we perceive most tempo changes as being smooth. Many irregularities in the path arise because the cost function is tuned to match note onsets, and therefore the frames where no new notes appear have very little to distinguish them. Some form of smoothing or interpolation could be performed in order to create a path which is musically plausible. First attempts at smoothing worsened the results, as outlying points influenced the matching points more than the converse. Further investigations into smoothing are being made, for example, considering interpolation to remove outlying points and replacement of adjacent horizontal and vertical path segments with diagonal segments. Discontinuities in the path could still occur, for example, if one performer paused and the other did not, but this is the exception rather than the rule.

Most of the large errors occur at the beginnings and ends of files; no example has been found where the alignment is correct at the beginning and then incorrect for the bulk of the file. Part of the reason for this is that the offset from the first (respectively last) frame to the first (last) note onset varies greatly between files, and the DTW algorithm is required to find a path from the first to the last frame. If we specifically detected the first and last note, or alternatively detected silence in the audio files, many of these errors could be avoided.

One issue that has not been addressed is the problem of structural differences between performances. For example, if one performer repeats the first section of a movement and another performer does not, there is no way for the DTW algorithm to recover, since the width of the search band is only 5 or 10 seconds. In order to find structural differences and perform partial matches, the complete similarity matrix would need to be calculated, which would then limit the size of pieces which could be matched, due to memory and time limitations.

This work stemmed from a real-time audio alignment tool for live performance analysis (Dixon, 2005). Since the current work does not require on-line processing, some improvements could be made to the off-line system in order to reduce the number of tracking errors. For example, a default slope (relative tempo) could be computed from the durations of the audio files, and the forward algorithm could be biased to tend towards this slope.

We intend to extend MATCH to include score-audio alignment, so that it can be used as a score-following system in real-time, and so that symbolic metadata can be automatically aligned with performances and audio recordings.

ACKNOWLEDGEMENTS

This work was supported by the following organisations: the Vienna Science and Technology Fund, as part of the project CI010 *Interfaces to Music*; the Austrian Federal Ministry of Education, Science and Culture (BMBWK), as part of the START project Y99-INF; and the European Union, as part of the project EU-FP6-IST-507142 SIMAC (*Semantic Interaction with Music Audio Contents*). The Austrian Research Institute for Artificial Intelligence acknowledges the financial support of the BMBWK and the Austrian Federal Ministry of Transport, Innovation and Technology.

REFERENCES

P. Cano, A. Loscos, and J. Bonada. Score-performance matching using HMMs. In *Proceedings of the International Computer Music Conference*, pages 441–444. International Computer Music Association, 1999.

R.B. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the International Computer Music Conference*, pages 193–198. International Computer Music Association, 1984.

R.B. Dannenberg and N. Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the International Computer Music Conference*. International Computer Music Association, 2003.

S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001a.

S. Dixon. An interactive beat tracking and visualisation system. In *Proceedings of the International Computer Music Conference*, pages 215–218, 2001b.

S. Dixon. Live tracking of musical performances using on-line time warping. 2005. Submitted.

S. Dixon, W. Goebl, and E. Cambouropoulos. Smoothed tempo perception of expressively performed music. *Music Perception*, 23, 2005. In press.

J. Foote and S. Uchihashi. The beat spectrum: A new approach to rhythm analysis. In *IEEE International Conference on Multimedia and Expo*, 2001.

W. Goebl. Melody lead in piano performance: Expressive device or artifact? *Journal of the Acoustical Society of America*, 110(1):563–572, 2001.

F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 23:52–72, 1975.

E.J. Keogh and M.J. Pazzani. Derivative dynamic time warping. In *SIAM International Conference on Data Mining*, 2001.

D. Mazzoni and R.B. Dannenberg. Melody matching directly from audio. In *2nd International Symposium on Music Information Retrieval*, pages 73–82, 2001.

N. Orio and F. Déchelle. Score following using spectral analysis and hidden Markov models. In *Proceedings of the International Computer Music Conference*, pages 151–154. International Computer Music Association, 2001.

N. Orio and D. Schwarz. Alignment of monophonic and polyphonic music to a score. In *Proceedings of the International Computer Music Conference*, pages 155–158. International Computer Music Association, 2001.

L. R. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice, Englewood Cliffs, NJ, 1993.

C. Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proceedings of the 5th International Conference on Musical Information Retrieval*, pages 387–394, 2004.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimisation for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26:43–49, 1978.

D. Sankoff and J.B. Kruskal. *Time warps, string edits, and macromolecules: The theory and practice of sequence comparison*. Addison-Wesley, New York/Menlo Park/Reading, 1983.

F. Soulez, X. Rodet, and D. Schwarz. Improving polyphonic and poly-instrumental music to score alignment. In *4th International Conference on Music Information Retrieval*, pages 143–148, 2003.

Y. Zhu and D. Shasha. Warping indexes with envelope transforms for query by humming. In *ACM SIGMOD Conference*, 2003.