MATCH: A MUSIC ALIGNMENT TOOL CHEST

Simon Dixon
Austrian Research Institute for Artificial Intelligence
Freyung 6/6
Vienna 1010, Austria
simon@ofai.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. In tests with Classical and Romantic piano music, the average alignment error was 41ms (median 20ms), with only 2 out of 683 test cases failing to align. 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. 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 or music lover might want to compare how several different pianists play a particular phrase, which would involve a manual 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. Another use would be in an audio recording system to provide intelligent editing operations such as aligning splice points in corresponding files. The toolkit also provides functions for displaying 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 describe the modifications necessary for an efficient implementation for audio alignment. We also present the cost function used to evaluate the similarity of frames of audio data, and a brief description of the user interface and implementation details of MATCH. Section 3 reports 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 EFFICIENT 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 (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 local path constraints are placed on W, namely that the path is bounded by the ends of both sequences, and it is monotonic and continuous. 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). 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 path from $(1, 1)$ to (i, j), and $D(1, 1) = d(1, 1)$. 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 A Linear Time 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 compute 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), which estimates the alignment of a live performance with a recording in real time. The DTW path is 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 possible to use a narrower band with less 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 intuition behind the forward path algorithm can be explained with reference to Figure 1, where a band width of $w = 4$ is used for illustrative purposes. (In practice, a band width of $w = 500$ is used.) At any time the *active area* of the matrix is the top row and the right column of the calculated area. The minimum cost path to each of these cells is evaluated and the cell with the lowest minimum cost path (normalised by length) is used as an indication of the direction in which the optimal path appears to be heading. (The true optimal path cannot be known until the complete matrix is calculated.) If this cell is in the top right corner, the algorithm is considered to be on target. If it is to the left of the target (for example, after expansions 7 and 8 in Figure 1), then the calculated part of the matrix is expanded upwards until the algorithm is on target again (expansions 9 to 11). Likewise if the cell is below the target, expansion is performed to the right.

The algorithm is initialised by computing a square matrix of size w; then the calculated area is iteratively 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 in the active area with lowest minimum path cost. If this cell is in the top row, a new row is calculated, and if it is in the right column, a new column is calculated. To avoid pathological solutions, limits are placed on the number of successive row (respectively column) computations. 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 Hamming 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

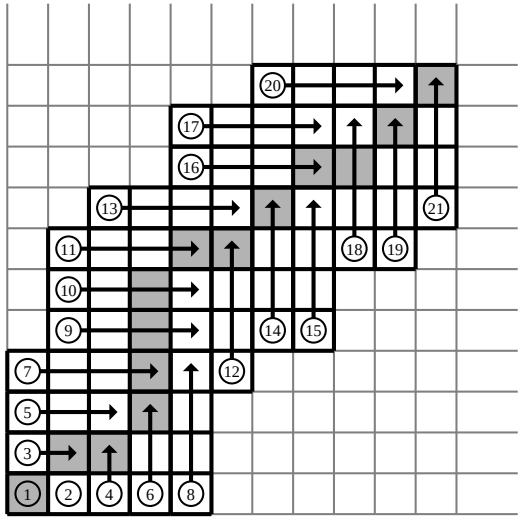

Figure 1: *An example of the on-line time warping algorithm with band width $w = 4$, showing the order of evaluation for a particular sequence of row and column increments. The axes represent time in the two files. All calculated cells are framed in bold, and the optimal path is coloured grey.*

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 its 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 bidirectional mapping between the time variables in the two files, with the resolution of the frame hop size.

2.4 Implementation Details

MATCH has a familiar graphical user interface which is similar to most media players (Figure 2). When files are loaded, the first file is used as the reference file, and subsequent files are each 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. One unfamiliar function (the '*' button) marks positions of interest in a piece, which are mapped to the corresponding locations in the other versions, so that the user can compare performances of a particular section or test the operation of the alignment algorithm. MATCH has functions for displaying the cost matrix, the forward and backward paths, and any other metadata associated with the files. The audio from one file can be played as matching is performed, with the matrix scrolling in real time and displaying a causal estimate of the alignment.

MATCH is implemented in Java 1.5, and on a 3GHz Linux PC, alignment of two audio files takes approximately 4% of the sum of durations of the files, using a time resolution of 20 ms. A lower frame rate could be used without significant loss of precision. MATCH is available for download at:

`http://www.ofai.at/~simon.dixon/match`

3 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.

3.1 Bösendorfer Data

The Bösendorfer SE290 is a grand piano with sensors which measure the precise timing and dynamics of every note with a time resolution of 1.25ms. This test set consists 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 $\frac{22\times21}{2} \times 2 = 462$ 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

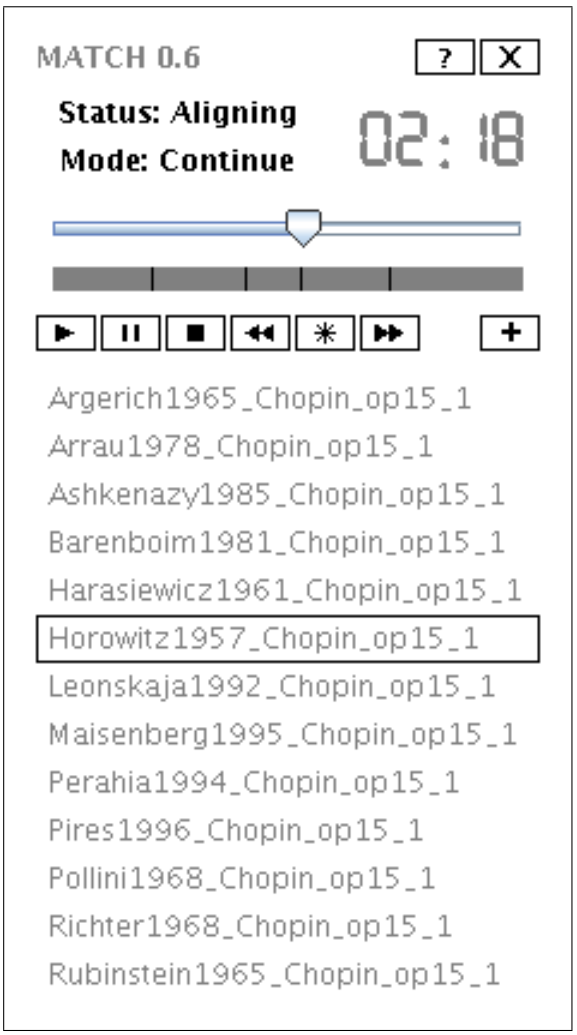

Figure 2: Screenshot of MATCH showing the user interface.

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. Two factors make this difficult: differences in the performed notes, which might be due to different score versions, ornaments, or mistakes; and asynchronies in chords (sets of simultaneous notes according to the musical notation), which are typically around 30 ms, but sometimes up to 150 ms, and not necessarily in any fixed temporal order. In these cases there is no unique "correct" alignment of the notes involved.

Therefore, 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 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 the average of the point-wise errors over all score events.

Error $\leq$		Cumulative error counts	
Frames	Seconds	Count	Percent
0	0.00	38655	46.1%
1	0.02	72934	87.1%
2	0.04	79126	94.5%
3	0.06	80540	96.2%
5	0.10	81343	97.1%
10	0.20	82325	98.3%
25	0.50	83292	99.4%
50	1.00	83658	99.9%

Table 1: Distribution of alignment errors, shown as cumulative counts and percentages of score events with an error up to the given value. The average error was 23ms.

Table 1 shows the distribution of point-wise errors less than or equal to 0,1,2,3,5,10,25 and 50 frames, where a hop size of 20 ms was used. The median and average errors are below the human temporal order threshold (the ability to distinguish the order of two sounds occurring closely in time), which 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.

3.2 BeatRoot Data

The second set of test data involved music with a large range of recording conditions, pianos, pieces and interpretations, where the beat had been annotated using the interactive beat tracking system BeatRoot (Dixon, 2001). This data set is larger, complexer and more varied, containing Classical and Romantic Period piano music recorded over the second half of the twentieth century by great pianists such as those listed in Figure 2. However, the data is only annotated at beat times (not note onsets) and is less precise, having an estimated accuracy of 30ms.

The results are summarised in Table 2, showing the maximum, mean and median error for each piece. In 2 of the 221 test cases, the alignment failed, and these results were not included in the statistics. In most cases, the maximum error occurred at the end of a piece, where there is no further data to orient the alignment. The mean error tends to be biased by the maximum errors, so we also show the median error, which is less biased, but it gives no indication of the spread of the errors. The overall average error of 64ms is worse than for the previous test set, where the controlled recording conditions made similarity judgements much easier.

3.3 Further Tests

The above tests consisted only of piano music, which could be easier to align than other instruments, due to the sharpness of onsets and the fixed timbre of piano tones. Since we do not have any annotated non-piano music, in-

Composer	Piece (work, section)	Versions	Test Pairs	Events (total)	Error (seconds)		
					Maximum	Mean	Median
Beethoven	Op.15, No.2, bar 1–8	4	6	366	0.70	0.085	0.040
Chopin	Op.15, No.1	13*	77	16863	7.48	0.061	0.020
Mozart	KV279, 1st movt	5	10	5510	5.26	0.036	0.020
Mozart	KV279, 2nd movt	4	6	1836	2.26	0.058	0.020
Mozart	KV279, 3rd movt	5	10	4474	1.38	0.025	0.020
Mozart	KV280, 1st movt	5	10	5990	8.12	0.037	0.020
Mozart	KV280, 2nd movt	5	10	5012	8.90	0.102	0.020
Mozart	KV280, 3rd movt	5	10	2783	4.08	0.044	0.020
Schubert	D899, No.3	12	66	22506	5.74	0.071	0.020
Schumann	Op.15, No.7	6*	14	3570	2.28	0.087	0.020

Table 2: Alignment results for commercial CDs of the given works. Two lines are marked with '*', indicating that one pair failed to align and was not included in the statistics.

formal tests on other music were performed by marking positions in one file, and listening to the aligned files to check that the marks were transferred to the corresponding positions in each recording. This method has several disadvantages: it can only detect errors of at least several hundred milliseconds, it relies on human judgement, it is not automated, and it only checks specific points on the alignment path, not the complete path.

We tested some solo classical guitar pieces by Albeniz (Asturias, Cordoba, Sevilla), Granados (Spanish Dances 4 and 5), Tarrega (Capricho Arabe) and Villa Lobos (Prelude 1). The alignments of Asturias and Spanish Dance No. 4 were partially unsuccessful, due to many differences in the arrangements. The other works were successfully aligned. Tests with orchestral music, including Tchaikovsky's Piano Concerto No. 1 and 10 different interpretations of the first movement of Schumann's Piano Concerto, revealed no problems in alignment. Some errors were apparent in the alignment of other works, particularly at the beginnings and ends of the pieces. Two popular Beatles songs (*I Wanna Hold Your Hand* and *She Loves You*) in English and German versions were also aligned successfully. These tests suggest that the similarity measure is not restricted to piano tones, but is applicable to a variety of instruments.

4 DISCUSSION AND CONCLUSION

This paper presented an audio alignment toolkit which uses a modified DTW algorithm. The average alignment error for solo piano music was 41ms, with only 2 out of 683 test cases failing to align. Informal tests with guitar, orchestral and popular music confirmed the generality of the system.

A low-level audio representation was used in preference to a high-level representation, which would enable a more efficient DTW computation, but is less reliable in its extraction of features. 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) and 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 chromagram, 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 more information than is necessary. 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; Turetsky and Ellis, 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 this work is somewhat exceptional; in general, we will not have access to multiple labelled performances.

4.1 Future Work

We conclude with some ideas for extending and improving this work. 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. However, smoothing tends to worsen the numerical results, as the only improvements are between the evaluated points, and some outlying points adversely influence correctly aligned note onsets. Our current smoothing algorithm uses interpolation to remove outlying points, replacing adjacent horizontal and vertical path segments with diagonal segments.

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, by computing a default slope (relative tempo) from the durations of the audio files, and biasing the forward algorithm to favour this slope. In future work, 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 recordings.

ACKNOWLEDGEMENTS

This work was supported by: the Vienna Science and Technology Fund, project CI010 *Interfaces to Music*; the Austrian Ministry BMBWK, START project Y99-INF; and the European Union, project EU-FP6-IST-507142 SIMAC. The Austrian Research Institute for Artificial Intelligence acknowledges the support of the ministries BMBWK and BMVIT.

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

R. Dannenberg and N. Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the International Computer Music Conference*, pages 27–34, 2003.

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

S. Dixon. Live tracking of musical performances using on-line time warping. In *Proceedings of the 8th International Conference on Digital Audio Effects*, 2005.

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

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.

D. Mazzoni and R. 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, 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, 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. 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.

R. Turetsky and D. Ellis. Ground-truth transcriptions of real music from force-aligned MIDI syntheses. In *4th International Conference on Music Information Retrieval*, pages 135–141, 2003.

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