19th INTERNATIONAL CONGRESS ON ACOUSTICS
MADRID, 2-7 SEPTEMBER 2007

TOOLS FOR ANALYSIS OF MUSICAL EXPRESSION

PACS: 43.75.St

Dixon, Simon[1]
[1] Queen Mary, Univ. of London; Mile End Rd, London, E1 4NS, UK; simon.dixon@elec.qmul.ac.uk

ABSTRACT
Studies of expressive music performance require precise measurements of the parameters (such as timing, dynamics and articulation) of individual notes and chords. Particularly in the case of the great performers, the only data usually available to researchers are audio recordings and the score, and digital signal processing techniques are employed to estimate the higher level control parameters from the audio signal. In this paper, two systems for extraction of timing information from audio recordings are described. The first system is an interactive beat tracking and annotation system called BeatRoot, which estimates the times of the beats in the music by finding regularities in the timing of note onsets using a multi-agent architecture. The second system, MATCH, performs alignment (or synchronisation) of different versions of the same piece of music, computing an index of corresponding locations in the different recordings using an efficient time warping algorithm. MATCH can be used to automatically transfer content-based metadata from one recording to another, or to follow a live performance, for example in order to automatically turn pages for a musician. Both systems are equipped with an intuitive graphical user interface and are being used by musicologists in large-scale studies of performance.

INTRODUCTION

A musical composition, expressed as a score, is only an approximation of what is performed on stage. It is like a sketch, where the fine details are left to the performer to determine. Expert performers use various means to shape the music, in order to communicate aesthetics, emotion and aspects of the musical structure to the audience. These means include changes in rate of performance (tempo), dynamics (loudness), articulation (connectedness of notes), and instrument-specific techniques such as vibrato or use of the pedal on the piano.

In order to develop computational models of expressive performance, precise measurements of the parameters of individual notes and chords must be computed from the audio signals. Timing information is determined first, since each of the notes must be located in the audio signal before any of its other parameters can be determined. The other parameters are then extracted as required, and machine learning and data mining methods can be employed to find patterns in the performance data, which might correspond to general principles of musical interpretation or specific traits of famous performers [12].

In this paper, we are concerned with only the first step of this process, the measurement of performance timing, and we present two systems for this task. The next section describes the interactive beat tracking and annotation system BeatRoot, which estimates the times of the beats in the music by finding regularities in the timing of note onsets using a multi-agent architecture. The following section describes MATCH, which performs alignment (or synchronisation) of different versions of the same piece of music, computing an index of corresponding locations in the different recordings using an efficient time warping algorithm. The paper concludes with a discussion of the use of the systems in studies of music performance.

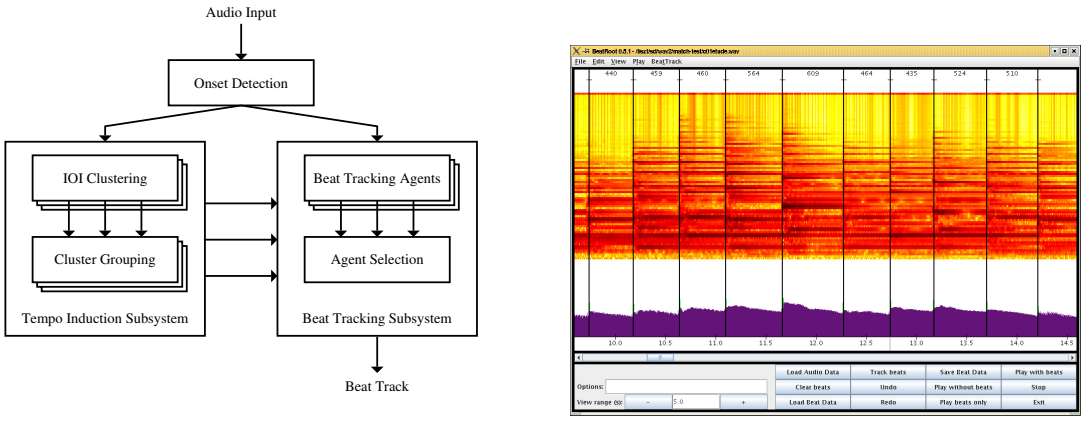

Figure 1: Left: system architecture of BeatRoot; Right: Screen shot of BeatRoot showing an excerpt from Chopin Etude Op.10, No.3, with the inter-beat intervals in ms (top), beat times (long vertical lines), spectrogram (centre), amplitude envelope (below) and control panel (bottom)

BEATROOT: TRACKING & ANNOTATION OF METRICAL STRUCTURE

Beat tracking is the task of identifying the basic rhythmic pulse of a piece of music, such as when people tap their feet, clap their hands or dance in time with music. Since this requires little or no musical training, it appears to be a relatively simple task, but despite extensive research (see [8] for a review) computer beat tracking systems still fall short of human abilities. BeatRoot [1, 3] is a beat tracking system which models the perception of beat by two interacting processes: the first finds the rate of the beats (*tempo induction*), and the second synchronises a pulse sequence with the music (*beat tracking*), as shown in Figure 1 (left). Initial processing of the audio signal is concerned with finding the onsets of musical notes, which are the primary carriers of rhythmic information. Earlier versions of BeatRoot used a time-domain onset detection algorithm, which found local peaks in the slope of a smoothed amplitude envelope, but this has been replaced with an onset detector which finds peaks in the spectral flux [4].

Tempo Induction

The tempo induction algorithm uses the calculated onset times to compute clusters of inter-onset intervals (IOIs). An IOI is defined to be the time interval between any pair of onsets, not necessarily successive. In most types of music, IOIs corresponding to the beat and simple integer multiples and fractions of the beat are most common. Due to fluctuations in timing and tempo, this correspondence is not precise, but by using a clustering algorithm, it is possible to find groups of similar IOIs which represent the various musical units (e.g. half notes, quarter notes).

This first stage of the tempo induction algorithm is represented in Figure 2 (left), which shows the events along a time line (above), and the various IOIs (below), labelled with their corresponding cluster names (C1, C2, etc.). The next stage is to combine the information about the clusters, by recognising approximate integer relationships between clusters. For example, in Figure 2 (left), cluster C2 is twice the duration of C1, and C4 is twice the duration of C2. This information, along with the number of IOIs in each cluster, is used to weight the clusters, and a ranked list of tempo hypotheses is produced and passed to the beat tracking subsystem.

Beat Tracking

The beat tracking subsystem uses a multiple agent architecture to find sequences of events which match the various tempo hypotheses, and rates each sequence to determine the most likely sequence of beat times. Each agent represents a hypothesis about the rate and the timing of the beats up to the current time, and makes predictions of the next beats based on its current state. Each agent is initialised with a tempo (rate) hypothesis from the tempo induction subsystem and an onset time which defines the agent's first beat time (phase). The agent then predicts further beats spaced according to the given tempo and first beat, using a pair of tolerance windows to

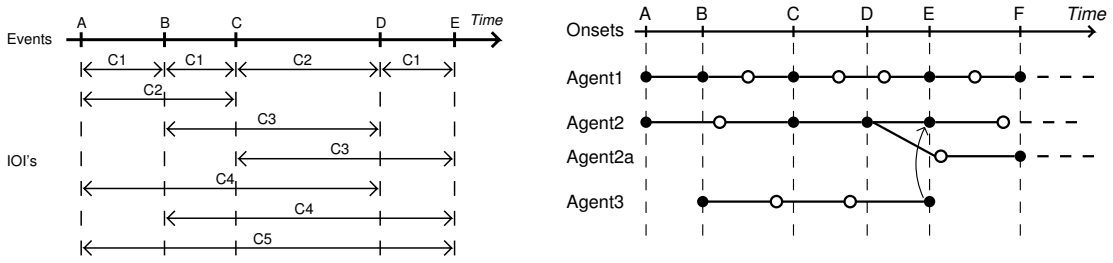

Figure 2: Left: clustering of inter-onset intervals – each interval between any pair of events is assigned to a cluster (C1, C2, C3, C4 or C5); Right: beat tracking by multiple agents (see text)

allow for deviations from perfectly metrical time. Onsets which occur within the inner window around a predicted beat time are taken as actual beat times, and are stored by the agent and used to update its rate and phase. Onsets falling in the outer window are taken to be possible beat times, but the possibility that the onset is not on the beat is also considered.

Figure 2 (right) illustrates the operation of beat tracking agents. A time line with 6 onsets (A to F) is shown, and below the time line are horizontal lines marked with solid and hollow circles, representing the behaviour of each agent. The circles represent predicted beat times which respectively correspond with onsets (solid circles), or do not correspond with onsets (hollow circles).

Agent1 is initialised with onset A as its first beat. It then predicts a beat according to its initial tempo hypothesis from the tempo induction stage, and onset B is within the inner window of this prediction, so it is taken to be on the beat. Agent1's next prediction lies between onsets, so a further prediction, spaced two beats from the last matching onset, is made. This matches onset C, so the agent marks C as a beat time and interpolates the missing beat between B and C. Then the agent continues, matching further predictions to onsets E and F, and interpolating missing beats as necessary. Agent2 illustrates the case where an onset matches only the outer prediction window, in this case at onset E. Because there are two possibilities, a new agent (Agent2a) is created to cater for the possibility that E is not a beat, while Agent2 assumes that E corresponds to a beat. A special case is shown by Agent2 and Agent3 at onset E, when it is found that two agents agree on the time and rate of the beat. Rather than allowing the agents to duplicate each others' work for the remainder of the piece, one of the agents, in this case Agent3, is terminated, as indicated by the arrow. A further special case (not illustrated) is that an agent can be terminated if it finds no events corresponding to its beat predictions (it has lost track of the beat).

Each agent is equipped with an evaluation function which rates how well the predicted and actual beat times correspond. The rating is based on how evenly the beat times are spaced, how many predicted beats correspond to actual events, and the salience of the matched events, which is calculated from the spectral flux at the time of the onset. At the end of processing, the agent with the highest score outputs its sequence of beats as the solution to the beat tracking problem.

Implementation, Results and Discussion

The system described above has been implemented with a graphical user interface which allows playback of the music with the beat times marked by clicks, and provides a graphical display of the signal and the beats with editing functions for correction of errors or selection of alternate metrical levels (see Figure 1, right). BeatRoot is written in Java and is available from:

`http://www.elec.qmul.ac.uk/people/simond/beatroot/`

BeatRoot has been tested on classical, jazz, and popular works with a variety of tempi and meters [1]. Two main types of error have been identified: the choice of a musically related metrical level such as double or half the subjectively chosen rate (tempo errors), and the loss of synchronisation with the beat (phase errors), from which it usually recovers quickly, despite the fact that it has no high level knowledge of music to guide it. BeatRoot performed best of the systems submitted for the MIREX 2006 Audio Beat Tracking Evaluation [3].

Although BeatRoot works well for music with complex rhythms (syncopation), it is not suitable for music with extreme tempo variations, such as Romantic piano music, where tempo changes of 50% from beat to beat are not uncommon. Other limitations of BeatRoot are that its onset detection is not suitable for music with weak onsets, and that it provides a low resolution analysis of timing (the "beat level"), with a low precision due to manual correction, which is also tedious and error-prone. One insight from this work is that much knowledge is being ignored, in the form of the score, other performances, and general principles of music performance, which could inform the beat tracking process. Rather than create a musically intelligent beat tracker, in the next section we describe an alternative approach which addresses many of these issues.

MATCH: AN AUDIO ALIGNMENT SYSTEM

To analyse tempo changes, a musicologist might painstakingly mark 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. Similarly, to compare how several different performers play a particular phrase of music, the relevant phrase in each recording must first be found. MATCH [7] is an automatic audio alignment system which addresses these needs, using an efficient time warping algorithm to create a mapping between the time axes of pairs of performances showing where corresponding notes begin in each performance, and allowing annotations of score positions (bookmarks) to be transferred automatically between the different renditions. MATCH has time and space costs that are linear in the lengths of the performances, allowing arbitrarily long pieces to be aligned faster than real time.

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. The quadratic time and space cost is often cited as a limiting factor for the use of DTW with long sequences. Global or local path constraints can be implemented to reduce complexity, but these can easily lead to the exclusion of the desired solution. The *on-line time warping* algorithm [2] uses adaptive constraints based on a forward path estimation which defines the centre of the band of the cost matrix that is considered for alignment, enabling an efficient and robust alignment.

The intuition behind the forward path algorithm can be explained with reference to Figure 3 (left), where a band width of $w = 4$ frames is used for illustrative purposes. (In practice, a band width of 500 frames, or 10 seconds, is used.) 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. 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. 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 3, left), then the calculated part of the matrix is expanded upwards by calculating new rows 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. 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 [2]. 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. The path returned by the alignment algorithm is used as a lookup table between the two audio files to find the location in one file corresponding to a selected location in the other file.

The Frame Comparison Cost Function

The alignment of audio files is based on a cost function which assesses the similarity of frames of audio data. MATCH uses 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 frequency axis is mapped to a scale which is

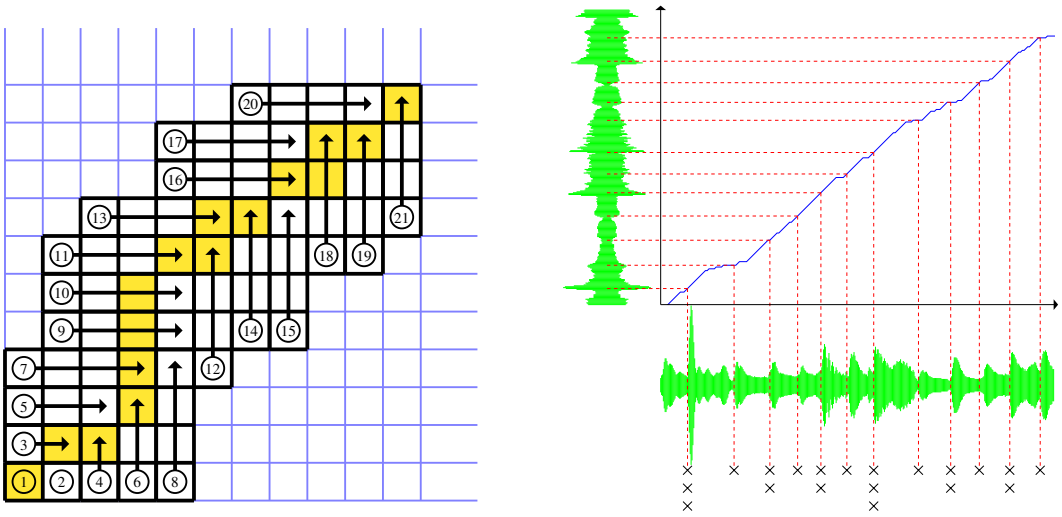

Figure 3: Left: an example of the on-line time warping algorithm with band width $w = 4$, showing the order of evaluation for a particular sequence of row and column expansions. The axes represent time in the two files. All calculated cells are framed in bold, and the optimal path is coloured yellow; Right: annotation via alignment – the Chassain performance of Villa Lobos' Prelude 1 (horizontal axis) is annotated with the metrical structure, which is mapped onto the Bream performance (vertical axis) via audio alignment.

linear at low frequencies and logarithmic at high frequencies. This achieves 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) are mapped linearly to the first 34 elements of the new scale. The bins from 370Hz – 12.5kHz are 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) are summed into the last element of the new scale. The resulting vector contains 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 Euclidean distance.

Implementation, Results and Discussion

MATCH has been tested on several sets of data [7]. A precise quantitative evaluation using Chopin works recorded on a Bösendorfer computer-monitored piano by 22 different pianists gave an average error of 23ms and median error 20ms (1 frame). A further quantitative evaluation based on semi-automatic annotation of CD recordings of Classical and Romantic piano music recorded over the second half of the twentieth century gave an average error of 64ms (median 20ms), the larger average error being mainly due to errors at the ends of pieces, where there is no further data to orient the alignment. Additionally, alignment failed entirely on 2 of the 221 pairs of recordings. Finally a qualitative evaluation based on unannotated CD recordings of other instruments (guitar music, piano concertos, Beatles songs) was performed, also with a high degree of success.

In contrast with BeatRoot, MATCH works well even with extreme tempo variations. Alignment is performed with a higher resolution ("chord level") and greater precision, and an interface for interactive error correction is included in the system. The main limitation of MATCH is that recordings must be structurally identical (same repeats) for the alignment to be successful. MATCH is implemented in Java, 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. It also has a real-time mode for tracking live performances [2]. MATCH has a familiar graphical user interface which is similar to most media players, and is available for download at:

`http://www.elec.qmul.ac.uk/people/simond/match/`

CONCLUSION

Two programs were described for analysing musical performance timing. BeatRoot has been used in a large scale study of interpretation in piano performance [11, 12] to create symbolic metadata from audio CDs for automatic analysis using machine learning. It has also been used for visualisation of expression (the Performance Worm [5]), automatic classification of dance styles [6], and performer recognition and style characterisation [9, 10].

Since BeatRoot is not able to cope with extreme tempo variations, a large amount of human effort was expended to correct the machine-made annotations. A better approach for analysing several performances of one piece is to make use of audio alignment. In this case, beat tracking and manual correction is only performed on one performance, and the other performances are aligned to this reference performance with MATCH, so that the beat position metadata can be generated automatically as in Figure 3 (right). Since alignment errors are much less frequent than beat tracking errors, this significantly reduces the human effort required. Further, the beat tracking step can be by-passed by synthesising a performance directly from a MIDI score, giving an almost fully automatic annotation of audio recordings with musical timing metadata.

References

[1] S. Dixon. Automatic extraction of tempo and beat from expressive performances. Journal of New Music Research **30 (2001), no. 1** 39–58

[2] S. Dixon. Live tracking of musical performances using on-line time warping. In *Proceedings of the 8th International Conference on Digital Audio Effects* (2005) 92–97

[3] S. Dixon. MIREX 2006 audio beat tracking evaluation: BeatRoot. http://www.music-ir.org/evaluation/MIREX/2006_abstracts/BT_dixon.pdf (2006)

[4] S. Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects* (2006) 133–137

[5] S. Dixon, W. Goebl, G. Widmer. Real time tracking and visualisation of musical expression. In *Music and Artificial Intelligence: Second International Conference, ICMAI2002*. Springer, Edinburgh, Scotland (2002) 58–68

[6] S. Dixon, F. Gouyon, G. Widmer. Towards characterisation of music via rhythmic patterns. In *5th International Conference on Music Information Retrieval* (2004) 509–516

[7] S. Dixon, G. Widmer. MATCH: A music alignment tool chest. In *6th International Conference on Music Information Retrieval* (2005) 492–497

[8] F. Gouyon, S. Dixon. A review of automatic rhythm description systems. Computer Music Journal **29 (2005), no. 1** 34–54

[9] C. Saunders, D. Hardoon, J. Shawe-Taylor, G. Widmer. Using string kernels to identify famous performers from their playing style. In *Proceedings of the 15th European Conference on Machine Learning* (2004)

[10] E. Stamatatos, G. Widmer. Automatic identification of music performers with learning ensembles. Artificial Intelligence **165 (2005), no. 1** 37–56

[11] G. Widmer. Machine discoveries: A few simple, robust local expression principles. Journal of New Music Research **31 (2002), no. 1** 37–50

[12] G. Widmer, S. Dixon, W. Goebl, E. Pampalk, A. Tobudic. In search of the Horowitz factor. AI Magazine **24 (2003), no. 3** 111–130