An Interactive Beat Tracking and Visualisation System

Simon Dixon
Austrian Research Institute for Artificial Intelligence, Vienna
email: simon@oefai.at

Abstract

This paper describes BeatRoot, a system which performs automatic beat tracking on audio or MIDI data and creates a graphical and audio representation of the data and results, as part of an interactive interface for correcting errors or selecting alternative metrical levels for beat tracking. The graphical interface displays the input data and the computed beat times, and allows the user to add, delete and adjust the beat times and then automatically re-track the remaining data based on the user input. The system also provides audio feedback consisting of the original input data accompanied by a percussion instrument sounding at the computed beat times. At the heart of the system is a beat tracking algorithm which estimates tempo based on the frequency of occurrence of the various time durations between pairs of note onset times, and then uses a multiple hypothesis search to find the sequence of note onsets that best matches one of the possible tempos. The primary application of this system is in the analysis of tempo and timing in musical performance, although the beat tracking algorithm itself has been shown to perform at least as well as other state-of-the-art systems.

1 Introduction

Significant progress has been made in the last decade in developing computer systems that automatically find the beat in a musical performance (Rowe 1992; Rosenthal 1992; Large and Kolen 1994; Large 1996; Goto and Muraoka 1995; Goto and Muraoka 1999; Scheirer 1998; Dixon 2000; Dixon and Cambouropoulos 2000; Dixon 2001a; Cemgil, Kappen, Desain, and Honing 2000; Cemgil, Kappen, Desain, and Honing 2001). However, no such system comes close to approximating a musician's ability on the same task, and therefore automatic beat detection, or beat tracking, as it is often called, has not been employed in many applications beyond beat tracking itself. We demonstrate a system which performs automatic beat tracking on audio or MIDI data and creates a graphical and audio representation of the results as part of an interactive interface for correcting errors or selecting alternative metrical levels for beat tracking. The graphical interface displays the input data and the computed beat times, and allows the user to add, delete and adjust the beat times and then automatically re-track the remaining data based on the user input. The system also provides audio feedback consisting of the original input data accompanied by a percussion instrument sounding at the computed (or adjusted) beat times.

At the heart of the system is a beat tracking algorithm which estimates tempo based on the frequency of occurrence of the various durations between note onset times, and then uses a multiple hypothesis search to find the sequence of note onsets that best matches one of the possible tempos. No prior knowledge or specific characteristics of the input data are assumed; all required information is derived from the data. In this way, the beat tracking system is able to perform well for many different musical styles, tempos and meters, including expressively performed classical and jazz music.

The primary application of this system is in the analysis of timing in musical performance. However there are other research and application areas for which components of the system could be used. A beat tracking system is an important component of any system that performs content analysis of audio, for example indexing and content-based retrieval of audio data in multimedia databases and libraries, score extraction and automatic transcription of music. Another application of beat tracking is in the synchronisation of devices such as lights, electronic musical instruments, recording equipment, computer animation and video with musical data. Such synchronisation might be necessary for multimedia or human-machine interactive performances or studio post-production work. There is an increasingly high demand for systems which can process data in a 'musically intelligent' way, and the interpretation of beat is one of the most fundamental aspects of musical intelligence.

The graphical front end is also useful for visualising performance and beat tracking data. The low coupling in the system allows alternative beat tracking algorithms to be attached with minimal effort, thus making the system ideal as a testing and evaluation framework for different beat tracking algorithms.

In the remaining three sections of this paper, we describe the beat tracking algorithm, then the user interface, and conclude with a discussion of the current major application of the system, performance timing analysis.

2 The Beat Tracking Algorithm

This section gives a brief description of how beat tracking is performed by the system. See Dixon (2001a) for a full description of the beat tracking algorithm. The input data is processed off-line to detect the salient rhythmic events and the relative timing of these events is analysed to generate hypotheses of the tempo at various metrical levels. Based on these tempo hypotheses, a multiple hypothesis search finds the sequence of beat times which has the best fit to the rhythmic events. Although the system does not operate in real time, it is currently being extended to provide real-time capabilities. The system is already efficient enough: audio data at CD quality is processed in less than 1/5 of the time length of the data (a 5 minute song is processed in less that one minute), and MIDI data is 2 to 10 times faster, depending on the note density.

The input data may be either digital audio or a symbolic representation such as MIDI. Audio data is pre-processed by a time-domain onset detection algorithm which calculates onset times from peaks in the slope of the amplitude envelope. The onsets are weighted by the magnitude of the amplitude envelope at each peak. MIDI data requires less processing, as the onset times can be extracted directly from the data. However, some pre-processing is performed: the note onsets are grouped as chords (rhythmic events), and these are weighted by the pitch, velocity and duration of the constituent notes. These weightings are a rough estimate of the perceptual salience of the rhythmic events, which has been shown to be a significant factor affecting the performance of a beat tracking system (Dixon and Cambouropoulos 2000).

The first main stage of processing uses a clustering algorithm on inter-onset intervals, which are the time durations between onsets of not necessarily consecutive events. The clusters represent the significant metrical units, and by examining the relationships between clusters, a set of tempo hypotheses is generated. These are fed into the second stage of the system, the tracking stage, which attempts to align the performed events with a flexible grid of events representing each tempo hypothesis. This stage uses a multiple agent architecture, where each agent represents a hypothesis about the tempo and alignment (phase) of the beat with respect to the performed events, and the agents evaluate their performance on the basis of the goodness of fit of their hypotheses to the data. The agent which is evaluated as the best returns its beat tracking solution as the output of the beat tracking system.

The beat tracking system has been tested on a large corpus of data. In almost all cases, the set of tempo hypotheses output by the first stage of the system contains the correct (performed) tempo, although the highest ranking hypotheses do not always correspond to the notated metrical level. The second stage of processing, calculation of beat times and choice of the best agent, is less robust, but when errors are made concerning the beat alignment (for example when the system tracks half a beat out of phase relative to the true beat), the system can often recover quickly to resume correct beat tracking, without recourse to any high level musical knowledge. In a recent comparative study (Dixon 2001b), the system was shown to perform marginally better than the state-of-the-art system of Cemgil et al. (2001), using their test data.

3 The Audio-Graphical User Interface

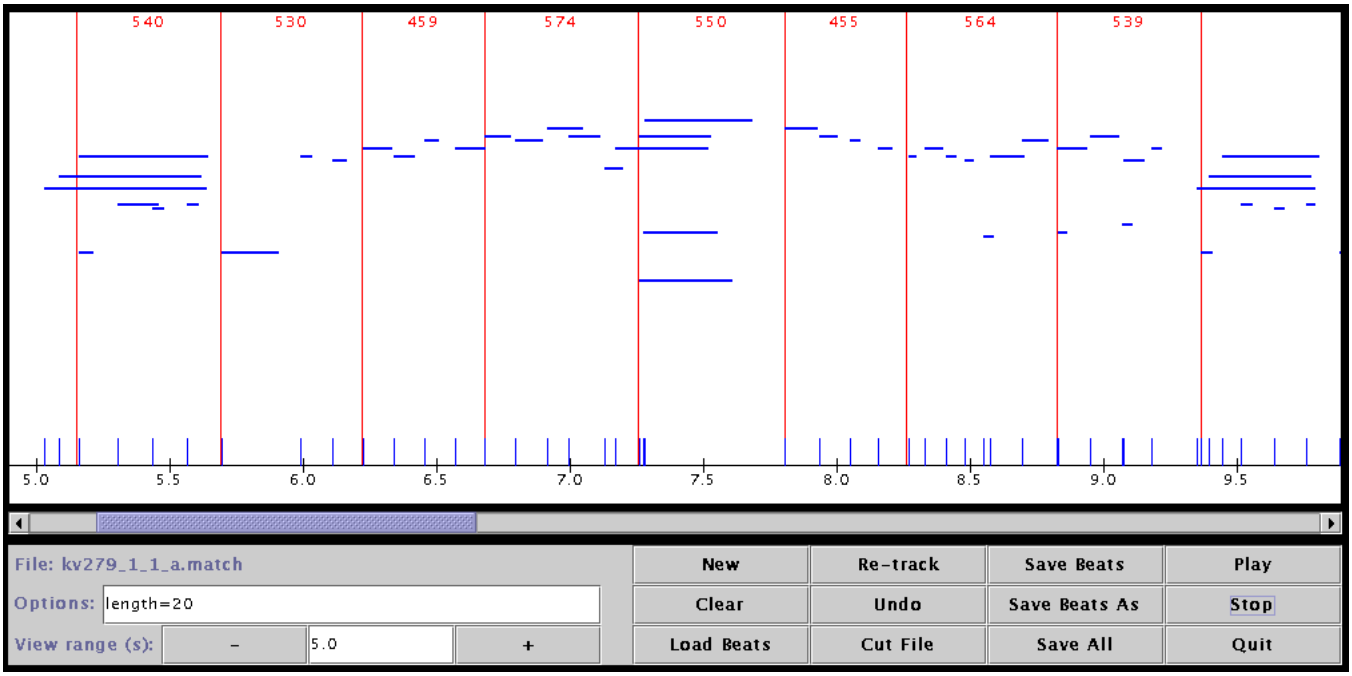
The most unique characteristic of this system is the interface, which provides visualisation, sonification, data editing, file management and control of the beat tracking system. The graphical display shows MIDI data in piano roll notation (Figure 1) and audio data as a smoothed amplitude envelope with the detected onsets marked on the display (Figure 2). The display of audio data may also contain an optional spectrogram (not shown). The estimated beat times are displayed as vertical lines, and the time intervals between beats are displayed at the top of the display. Each type of data is colour-coded so that the display is easy to read. A scroll bar controls the visible time window for the data, and a zoom function allows the time resolution to be adjusted. A control panel allows easy access to all of the functions of the system, such as loading and saving of input data and results, audio playback, and beat tracking itself.

The beat times can be edited directly by dragging the vertical lines representing beats (using mouse button 1). The other mouse buttons are used to insert and delete beats at the position of the cursor. By clicking on the time axis at the bottom of the display, it is possible to select a section of the data for playback, deletion, or re-tracking of the beat. In this way, the user can alternate between automatic and manual beat tracking, in order to obtain an accurate tempo track of the performance.

The audio playback function generates an audio or MIDI track (corresponding to the type of input data) containing a user-selectable percussion instrument sounding at each estimated beat time. This is combined with the input data and played back via the sound card (MIDI output is sent via a software synthesiser). The combined data can also be saved to file for playback at another time.

The options box provides access to all internal parameters of the beat tracking system. The parameters all have default values, but these can be overridden, for example to adjust the sensitivity of the system to tempo changes.

The beat tracking system is written in C++ (about 10000 lines of code) and the interface is written in Java (about 1000 lines) and runs on the Linux operating system. The software is available for download from `http://www.oefai.at/~simon/beatRoot` for non-commercial use.

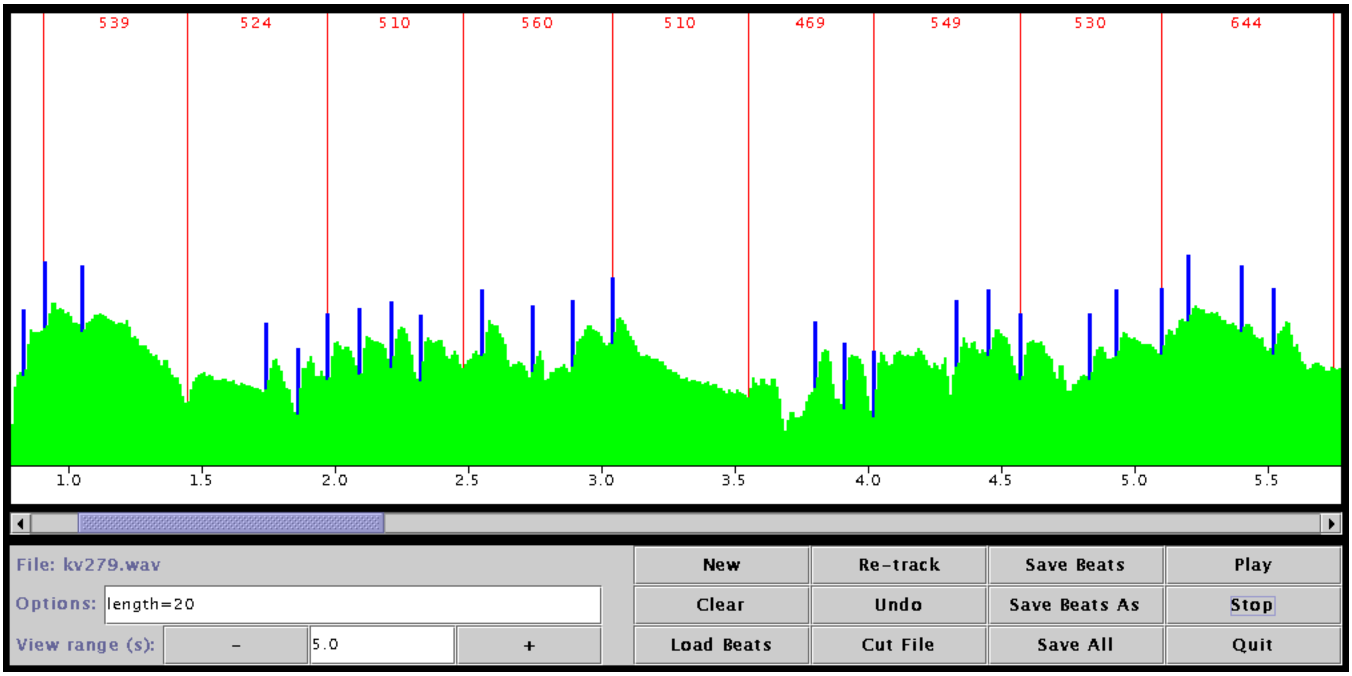


File: kv279_1_1_a.match

Options: length=20

View range (s): 5.0

Figure 1: Screen shot of beat tracker using MIDI input data. The MIDI data is shown in piano roll notation (blue), with the beat times superimposed as vertical lines (red).



File: kv279.wav

Options: length=20

View range (s): 5.0

Figure 2: Screen shot of beat tracker using audio input data. The amplitude envelope is shown at the bottom of the display (green), with the onset times as detected by the system marked by short vertical lines extending from the amplitude envelope (blue) and the beat times marked by long vertical lines (red).

4 Performance Timing Analysis

In studies of musical performance, the interpretation of musical works by performers is investigated, for example, their choice of tempo and expressive timing. These parameters are important in conveying structural and emotional information to the listener (Clarke 1999). By finding the times of musical beats, we can automatically calculate the tempo and variations in tempo within a performance. This accelerates the analysis process, thus allowing more wide-ranging studies to be performed.

As a tool for automating the analysis of musical performance, one would expect an automatic beat tracking system to be very useful, since, for example, it could generate tempo curves directly from performance data. But to date, such systems have not proved to be practical, because they do not avoid the tedious manual editing of timing data. One necessary feature required of such a system is that it provide an easy means of editing its output, for instance, by means of an interactive graphical user interface.

Our system provides a graphical visualisation of the input data, the generated beat times and the intervals between beats (instantaneous tempo), and allows editing (insertion, deletion and adjustment) of the beat times using the graphical interface, and restarting of automatic beat tracking after a correction has been performed. This provides the user with a complete tool for the generation of tempo curves from expressive performance data. The system is currently being used in a study of tempo in CD performances of Mozart piano sonatas (Goebl and Dixon 2001), and an initial study has been performed, analysing the resolution of the system and the effects of the various forms of feedback on the user's perception of the beat (Dixon, Goebl, and Cambouropoulos 2001).

In current work, the audio onset detection algorithm is being improved. In further work, we intend to extend the system in two major directions. The first is to allow the system to use score information to guide its beat tracking; this would greatly reduce the error rate of beat tracking when musical scores are available, and greatly accelerate studies of multiple interpretations of the same work. The second planned extension is to create a score extraction system, by adding features such as quantisation, meter recognition, note spelling, part separation and score display and editing, so that a symbolic score can be generated interactively from performance data.

Acknowledgments

This research is part of the START programme Y99-INF, financed by the Austrian Federal Ministry for Education, Science, and Culture (BMBWK). The BMBWK also provides financial support to the Austrian Research Institute for Artificial Intelligence. Thanks to Emilios Cambouropoulos, Gerhard Widmer and Werner Goebl for their input to this work.

References

Cemgil, A., B. Kappen, P. Desain, and H. Honing (2000). On tempo tracking: Tempogram representation and Kalman filtering. In *Proceedings of the 2000 International Computer Music Conference*, pp. 352–355. International Computer Music Association.

Cemgil, A., B. Kappen, P. Desain, and H. Honing (2001). On tempo tracking: Tempogram representation and Kalman filtering. *Journal of New Music Research*. To appear.

Clarke, E. (1999). Rhythm and timing in music. In D. Deutsch (Ed.), *The Psychology of Music*, pp. 473–500. Academic Press.

Dixon, S. (2000). A lightweight multi-agent musical beat tracking system. In *PRICAI 2000: Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pp. 778–788. Springer.

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

Dixon, S. (2001b). An empirical comparison of tempo trackers. In *Proceedings of the 8th Brazilian Symposium on Computer Music*. To appear.

Dixon, S. and E. Cambouropoulos (2000). Beat tracking with musical knowledge. In *ECAI 2000: Proceedings of the 14th European Conference on Artificial Intelligence*, pp. 626–630. IOS Press.

Dixon, S., W. Goebl, and E. Cambouropoulos (2001). Beat extraction from expressive musical performances. In *2001 Meeting of the Society for Music Perception and Cognition (SMPC2001), Kingston, Ontario*. To appear.

Goebl, W. and S. Dixon (2001). Analysis of tempo classes in performances of mozart sonatas. In *Proceedings of the VII International Symposium on Systematic and Comparative Musicology, Jyväskylä, Finland*. To appear.

Goto, M. and Y. Muraoka (1995). A real-time beat tracking system for audio signals. In *Proceedings of the International Computer Music Conference*, pp. 171–174. Computer Music Association, San Francisco CA.

Goto, M. and Y. Muraoka (1999). Real-time beat tracking for drumless audio signals. *Speech Communication 27*(3–4), 331–335.

Large, E. (1996). Modelling beat perception with a nonlinear oscillator. In *Proceedings of the 18th Annual Conference of the Cognitive Science Society*.

Large, E. and J. Kolen (1994). Resonance and the perception of musical meter. *Connection Science 6*, 177–208.

Rosenthal, D. (1992). Emulation of human rhythm perception. *Computer Music Journal 16*(1), 64–76.

Rowe, R. (1992). Machine listening and composing with cypher. *Computer Music Journal 16*(1), 43–63.

Scheirer, E. (1998). Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustical Society of America 103*(1), 588–601.