Preprint for Journal of New Music Research, 36, 2007/8

Evaluation of the Audio Beat Tracking System BeatRoot

Simon Dixon
Centre for Digital Music
Department of Electronic Engineering
Queen Mary, University of London
Mile End Road, London E1 4NS, UK
Email: simon.dixon@elec.qmul.ac.uk
Phone: +44 20 7882 7681
Fax: +44 20 7882 7997

Abstract

BeatRoot is an interactive beat tracking and metrical annotation system which has been used for several years in studies of performance timing. This paper describes improvements to the original system and a large-scale evaluation and analysis of the system's performance. The beat tracking algorithm remains largely unchanged: BeatRoot uses a multiple agent architecture which simultaneously considers several different hypotheses concerning the rate and placement of musical beats, resulting in accurate tracking of the beat, quick recovery from errors, and graceful degradation in cases where the beat is only weakly implied by the data. The original time domain onset detection algorithm, which caused problems for beat tracking music without prominent drums, has been replaced with a more robust onset detector based on spectral flux. Other new features include annotation of multiple metrical levels and phrase boundaries, as well as improvements in the user interface. The software has been rewritten entirely in Java, and it runs on all major operating systems. BeatRoot is evaluated on a new test set of 1360 musical excerpts from a wide range of Western musical styles, and the results are compared with other evaluations such as the MIREX 2006 Audio Beat Tracking Evaluation.

Beat tracking is the task of identifying and synchronising with the basic rhythmic pulse of a piece of music. It often takes place in natural settings, for example when people tap their feet, clap their hands or dance in time with music. At first sight this task does not appear to be particularly difficult, as it is performed by people with little or no musical training, but despite extensive research (see Gouyon & Dixon, 2005, for a review), computational models of beat tracking still fall short of human beat tracking ability (McKinney, Moelants, Davies, & Klapuri, 2007).

In previous work (Dixon, 2001a) we presented an algorithm for beat tracking of music in audio or symbolic formats using a two-stage process: *tempo induction*, finding the rate of

the beats, and *beat tracking*, synchronising a quasi-regular pulse sequence with the music. A simple time-domain onset detection algorithm found the most salient onsets, and clustering was used to find the most significant metrical units based on inter-onset intervals. The clusters were then compared to find reinforcing groups, and a ranked list of tempo hypotheses was computed. Based on these hypotheses, a multiple agent architecture was employed to match sequences of beats to the music, where each agent represented a specific tempo and alignment of beats with the music. The agents were evaluated on the basis of the regularity, continuity and salience of the onsets corresponding to hypothesised beats, and the highest ranked beat sequence was returned as the solution.

This algorithm was built into an application called BeatRoot (Dixon, 2001c), which displayed the musical data and beats in a graphical interface allowing interactive correction and re-tracking of the beats, as well as providing audio feedback, playing back the music with a percussive sound marking each beat position. Several evaluations of BeatRoot have been reported, including results on 13 complete Mozart piano sonatas (Dixon, 2001a), 108 recordings of each of 2 Beatles songs (Dixon, 2001b), several thousand short excerpts used in the audio tempo induction competition at ISMIR 2004 (Gouyon et al., 2006), and the 140 excerpts used in the MIREX 2006 audio beat tracking evaluation (Dixon, 2006b; McKinney et al., 2007).

Data created and corrected with BeatRoot has been used in large scale studies of interpretation in piano performance, where machine learning and data mining methods were employed to find patterns in the performance data corresponding to general principles of musical interpretation or specific traits of famous performers (Widmer, 2002; Widmer, Dixon, Goebl, Pampalk, & Tobudic, 2003). BeatRoot has also been used for visualisation of expression — the "Performance Worm" (Dixon, Goebl, & Widmer, 2002), automatic classification of dance styles (Dixon, Gouyon, & Widmer, 2004), and performer recognition and style characterisation (Saunders, Hardoon, Shawe-Taylor, & Widmer, 2004; Stamatatos & Widmer, 2005). Other possible applications are in music transcription, music information retrieval (e.g. query by rhythm), audio editing, and the synchronisation of music with video or other media. Some of the perceptual issues surrounding beat tracking and the BeatRoot interface are addressed by Dixon, Goebl, and Cambouropoulos (2006).

The BeatRoot system has been rewritten since its original release in 2001. In the next section we describe the audio beat tracking algorithm, which features a new onset detection algorithm using spectral flux. The following section presents the improved user interface, including facilities for annotation of multiple metrical levels and phrase boundaries. We then describe a new extensive evaluation of the system, using 1360 musical excerpts covering a range of musical genres, as well as the evaluation results from the 2006 MIREX evaluation. The final section contains the conclusions and ideas for further work. Since several recent reviews of beat tracking systems exist (e.g. Gouyon & Dixon, 2005, see also the other papers in this volume), we will not duplicate that work in this paper.

BeatRoot Architecture

Figure 1 depicts the architecture of the BeatRoot system, omitting the user interface components. BeatRoot has two major components, which perform tempo induction and beat tracking respectively. The digital audio input is preprocessed by an onset detection stage, and the list of onset (event) times is fed into the tempo induction module, where the

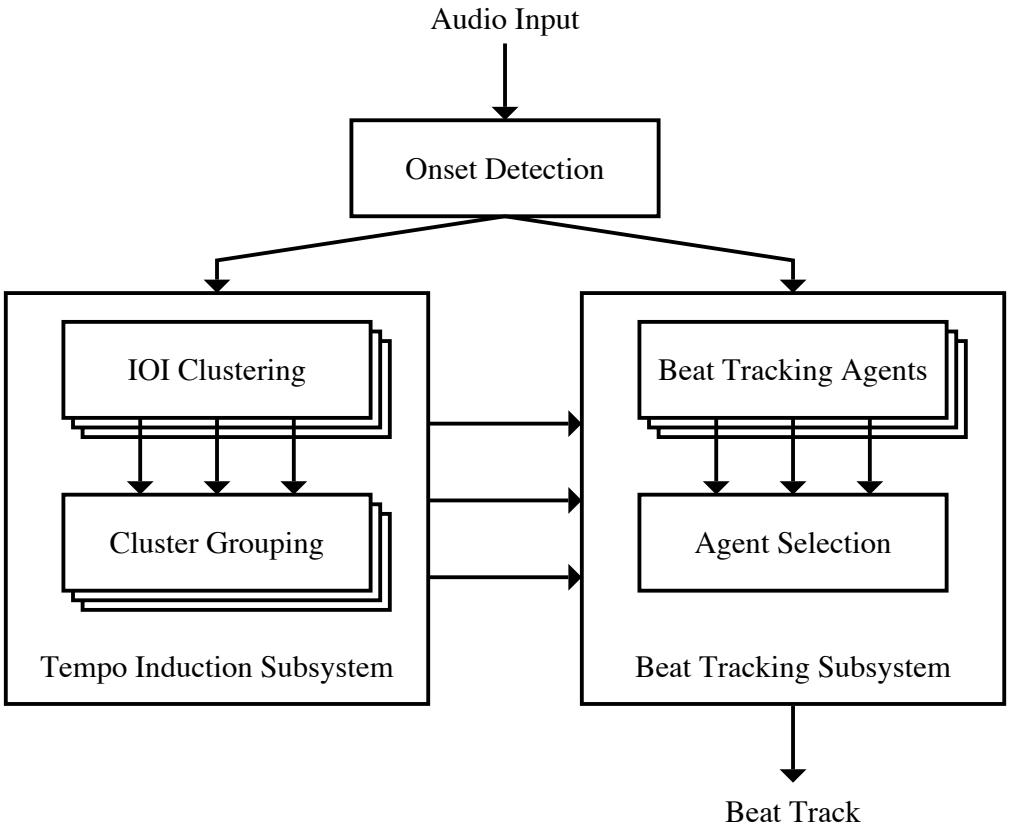

Figure 1. System architecture of BeatRoot

time intervals between events are then analysed to generate tempo hypotheses corresponding to various metrical levels. These hypotheses, together with the event times, are the input to the beat tracking subsystem, which uses an agent architecture to test different hypotheses about the rate and timing of beats, finally outputting the beat times found by the highest-ranked agent. In the following subsections each stage of processing is described in detail.

Onset Detection

BeatRoot takes an audio file as input, which is processed to find the onsets of musical notes, since the onsets are the primary carriers of rhythmic information. Secondary sources of rhythmic information, such as melodic patterns, harmonic changes and instrumentation, although potentially useful, are not used by the system. The estimation of event salience has been shown to be useful for beat tracking (Dixon & Cambouropoulos, 2000), with duration being the most important factor, and pitch and intensity also being relevant, but as there are no reliable algorithms for polyphonic pitch determination or offset detection, these parameters are only useful when working from symbolic (e.g. MIDI) data.

An onset detection function is a function whose peaks are intended to coincide with

the times of note onsets. Onset detection functions usually have a low sampling rate (e.g. 100 Hz) compared to audio signals; thus they achieve a high level of data reduction whilst preserving the necessary information about onsets. Most onset detection functions are based on the idea of detecting changes in one or more properties of the audio signal. In previous work (Dixon, 2001a), a simple time-domain onset detection algorithm was used, which finds local peaks in the slope of a smoothed amplitude envelope using the "surfboard" method of Schloss (1985). This method is particularly well suited to music with drums, but less reliable at finding onsets of other instruments, particularly in a polyphonic setting. Existing and new onset detection methods were evaluated in an empirical study of frequency domain onset detection methods (Dixon, 2006a), and performance in terms of precision and recall, as well as speed and complexity of the algorithm, were analysed. The three best detection functions were found to be spectral flux, weighted phase deviation and the complex domain detection function, and there was little difference in performance between these algorithms. Spectral flux was chosen based on its simplicity for programming, speed of execution and accuracy of correct onsets. Other comparisons of onset detection algorithms are found in (Bello et al., 2005; Collins, 2005; Downie, 2005).

The spectral flux onset detection function sums the change in magnitude in each frequency bin where the change is positive, that is, the energy is increasing. First, a time-frequency representation of the signal based on a short time Fourier transform using a Hamming window $w(m)$ is calculated at a frame rate of 100 Hz. If $X(n,k)$ represents the kth frequency bin of the nth frame, then:

$$X(n,k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(hn+m)\ w(m)\ e^{-\frac{2j\pi mk}{N}}$$

where the window size $N = 2048$ (46 ms at a sampling rate of $r = 44100$ Hz) and hop size $h = 441$ (10 ms, or 78.5% overlap). The spectral flux function SF is then given by:

$$SF(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - |X(n-1,k)|)$$

where $H(x) = \frac{x+|x|}{2}$ is the half-wave rectifier function. Empirical tests favoured the use of the L_1-norm over the L_2-norm used by Duxbury, Sandler, and Davies (2002) and Bello et al. (2005). Test results also favoured the linear magnitude over the logarithmic (relative or normalised) function proposed by Klapuri (1999).

The onsets are selected from the detection function by a peak-picking algorithm which finds local maxima in the detection function, subject to various constraints. The thresholds and constraints used in peak-picking have a large impact on the results, specifically on the ratio of false positives to false negatives. For example, a higher threshold generally reduces the number of false positives and increases the number of false negatives. The best values for thresholds are dependent on the application and the relative undesirability of false positives and false negatives. In the case of beat tracking, we speculate that false negatives are less harmful than false positives, but we have not tested this hypothesis. In any case, as it is difficult to generate threshold values automatically, the parameters were

determined empirically using the test sets in (Dixon, 2006a). The comparisons of onset detection algorithms were also performed by testing on the same data.

Peak picking is performed as follows: the spectral flux onset detection function $f(n)$ is normalised to have a mean of 0 and standard deviation of 1. Then a peak at time $t = \frac{nh}{r}$ is selected as an onset if it fulfils the following three conditions:

$$f(n) \geq f(k) \text{ for all } k \text{ such that } n - w \leq k \leq n + w$$
$$f(n) \geq \frac{\sum_{k=n-mw}^{n+w} f(k)}{mw + w + 1} + \delta$$
$$f(n) \geq g_\alpha(n-1)$$

where $w = 3$ is the size of the window used to find a local maximum, $m = 3$ is a multiplier so that the mean is calculated over a larger range before the peak, δ is the threshold above the local mean which an onset must reach, and $g_\alpha(n)$ is a threshold function with parameter α given by:

$$g_\alpha(n) = \max(f(n),\ \alpha g_\alpha(n-1) + (1-\alpha)f(n))$$

Experiments were performed with various values of the two parameters δ and α, and it was found that best results were obtained using both parameters, but the improvement in results due to the use of the function $g_\alpha(n)$ was marginal, assuming a suitable value for δ is chosen. The chosen values for the parameters were $\delta = 0.35$ and $\alpha = 0.84$. Using these values, the spectral flux onset detection algorithm correctly detected 97% of notes for solo piano music, and 88% of onsets for multi-timbral music (Dixon, 2006a). In a study of 172 different possible features for beat tracking, it was found that the complex domain onset detection function performed marginally better than spectral flux, which in turn performed better than the other 170 features (Gouyon, Dixon, & Widmer, 2007).

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, which shows the onsets along a time line (above), and the various IOIs (below), labelled with their corresponding cluster names (C1, C2, etc.). Clustering is performed with a greedy algorithm which assigns an IOI to a cluster if its difference from the cluster mean is less than a constant (25 ms). Likewise, a pair of clusters is merged if their means fall below this threshold.

The next stage is to combine the information about the clusters, by recognising approximate integer relationships between clusters. For example, in Figure 2, 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. See (Dixon, 2001a) for more details.

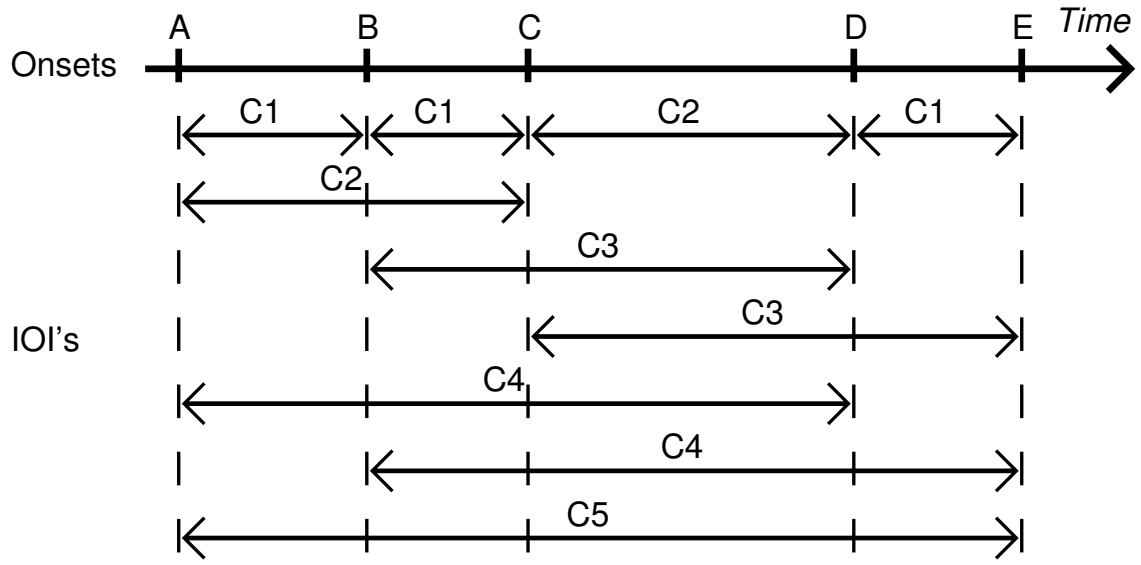

Figure 2. Clustering of inter-onset intervals: each interval between any pair of onsets is assigned to a cluster (C1, C2, C3, C4 or C5)

Beat Tracking

The most complex part of BeatRoot is the beat tracking subsystem, which 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. The music is processed sequentially from beginning to end, and at any particular point in time, the agents represent the various hypotheses about the rate and the timing of the beats up to that time, and make predictions of the next beats based on their current state.

Each agent is initialised with a tempo (rate) hypothesis from the tempo induction subsystem and an onset time, taken from the first few onsets, 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 tolerance windows to allow for deviations from perfectly metrical time (see Figure 3). Onsets which correspond with the inner window of predicted beat times 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 4 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 solid circles represent predicted beat times which correspond to onsets, and the hollow circles represent predicted beat times which do not correspond to onsets. The circles of Agent1 are more closely spaced, representing a faster tempo than that of the other agents.

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

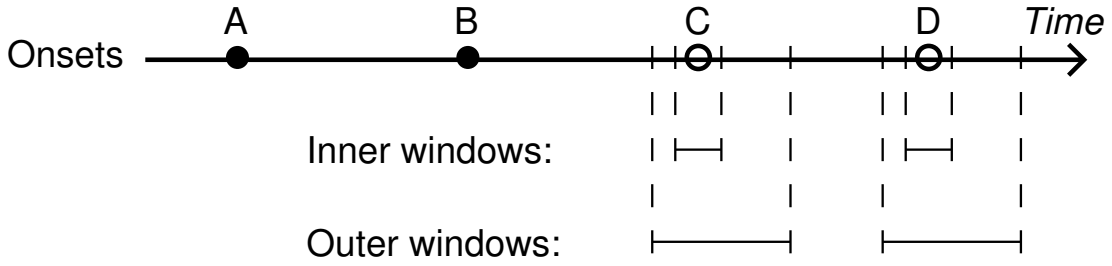

Figure 3. Tolerance windows of a beat tracking agent predicting beats around C and D after choosing beats at onsets A and B

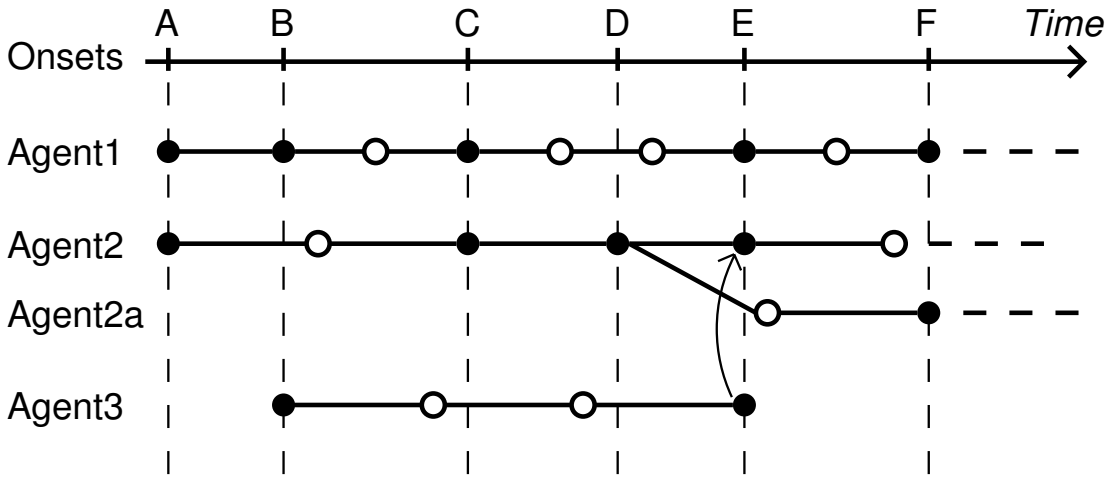

Figure 4. Beat tracking by multiple agents (see text for explanation)

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 when 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 is terminated. The choice of agent to terminate is based on the evaluation function described in the following paragraph. 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 onsets, 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 and User Interface

The algorithms described in the previous section have been incorporated into an application for annotation of audio data with metrical metadata. Unlike the original version which only ran on one operating system (Dixon, 2001c), the new version of BeatRoot is written entirely in Java, so that it can be used on any major operating system. BeatRoot can be downloaded under the GNU Public License from:

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

BeatRoot is equipped with a graphical user interface which shows the audio data as an amplitude envelope and a spectrogram, with the beats as vertical lines superimposed on the audio display (Figure 5). The user can edit the times of beats by dragging the lines left or right, and can add or delete beats as necessary. The editing features can be used for correction of errors made by the system or selection of an alternate metrical level. After editing, beat tracking can be performed on a selection of the data, using the corrections manually entered into the system. An annotation mode allows the user to specify the metrical level of each pulse, so that it is possible to annotate the complete metrical hierarchy.

The inter-beat intervals in milliseconds are visible at the top of the display, and the metrical level of each beat is indicated by the number of short horizontal lines through the beat marker (only one metrical level is annotated in the figure). The audio feedback provided by the system consists of audio playback accompanied by a percussion sound indicating the beat times. The various metrical levels can be indicated by using different percussion sounds, so that the correctness of annotations can be quickly verified by listening.

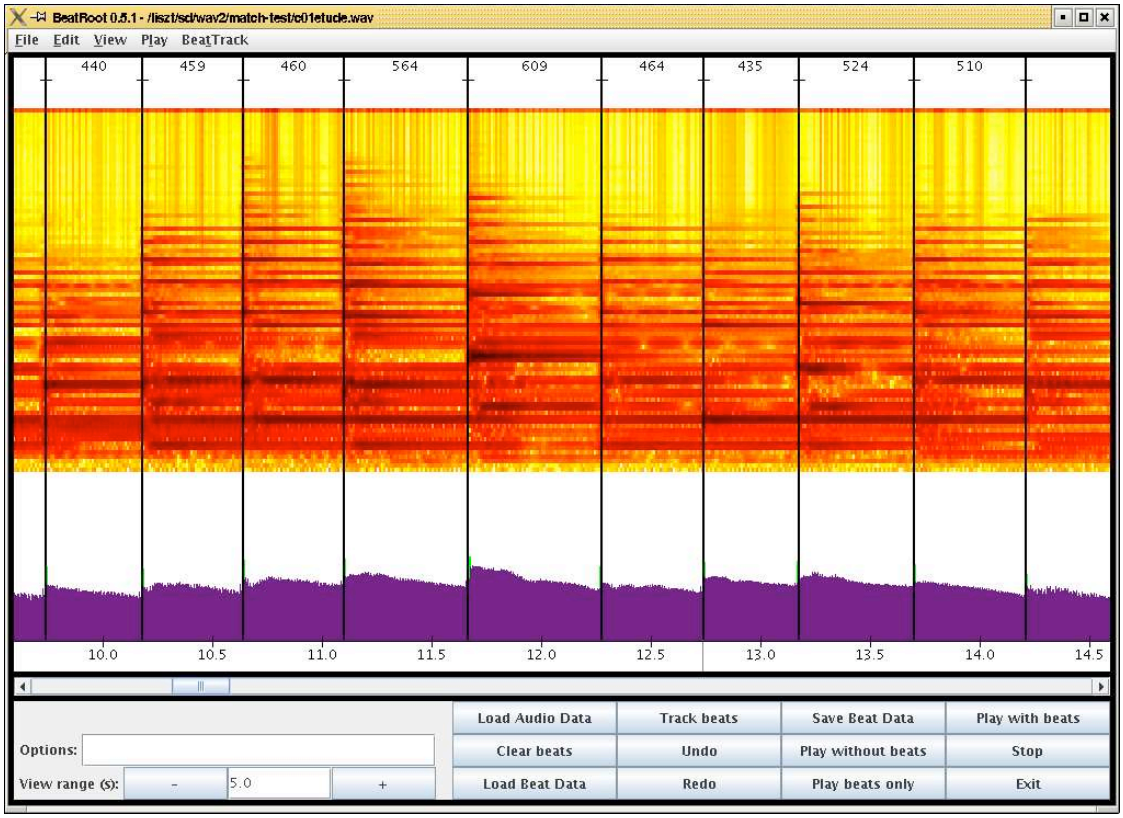

Figure 5. Screen shot of BeatRoot showing a 5-second excerpt from a Chopin piano Etude (Op.10, No.3), with the inter-beat intervals in ms (top), calculated beat times (long vertical lines), spectrogram (centre), amplitude envelope (below) marked with detected onsets (short vertical lines) and the control panel (bottom)

For use in performance studies, it is important to understand the limitations and biases introduced by the interface. The effects of each part of BeatRoot's interface were compared in an investigation of beat perception (Dixon et al., 2006, Experiment 2), where participants used the interface to manually annotate musical data with beat times. Various experimental conditions were tested, consisting of different types of feedback from the system (parts of the display or audio feedback were disabled), and the resulting sequences of beats were compared. Using audio feedback alone, participants preferred a smoother sequence of beats than that which is obtained by aligning beat times with note onsets (the usual assumption in beat tracking systems). Likewise, the presentation of inter-beat intervals encouraged smoothing of the tempo. Conversely, visual presentation without audio feedback resulted in closer alignment of beats with onsets, and a less smooth tempo curve. Large inter-participant differences were observed in these results, which was related to the ability of participants to interpret the visual feedback in terms of onsets. This leads to the conclusion that some training is necessary in order to use the interface accurately. Further, since there is a difference between the times of perceived beats and the times of

onsets corresponding to these beats, it is important to distinguish between two types of beat tracking: the standard beat tracking task involves the annotation of perceived beat times, but performance analysis is often concerned with annotating the timing of performed notes which are nominally (according to the score) on the beat. A clear understanding of this difference is necessary before annotation work commences.

Evaluation and Results

Evaluation of Beat Tracking Systems

Evaluation of beat tracking systems has been discussed at some length in the literature (Goto & Muraoka, 1997; Dixon, 2001a, 2001b; Gouyon & Dixon, 2005; Gouyon et al., 2006; Davies & Plumbley, 2007). Several main issues arise. First, as discussed above, the task of beat tracking is not uniquely defined, but depends on the application, and this has an impact on the way systems should be evaluated. Ambiguity exists both in the choice of metrical level and the precise placement of beats. Arbitrarily assigning a particular metrical level for "correct" beat tracking is too simplistic in the general case, as is restricting the tempo to an unambiguous range such as 61–120 beats per minute, which is not even sensible for non-binary meters. Performance analysis applications have the added problem of determining the beat times when different notes which are nominally on the beat are not performed precisely simultaneously, due either to chord asynchronies (single musician) or timing differences between performers (participatory discrepancies).

The second issue is that some systems are designed for a limited set of musical styles, which leads to the question of whether such systems can be compared with other systems at all, since the test data will give one system a natural advantage over another if it is closer to the first system's intended domain of usage. This issue does not only apply to musical style, but also features of the music such as the time signature, the variability of the tempo and the instrumentation.

Third, the availability of test data is a major constraint, and manual annotation in order to create ground truth data is labour-intensive, so it is difficult to create test sets large enough to cover a wide range of musical styles and give statistically significant results. In recent years, the availability of test data has increased, and with it the expectation that systems are tested systematically on common data sets. Ideally such data sets should be publicly available, but if the data consists of commercial music, copyright laws do not allow free distribution of the data. This has led to the MIREX series of evaluations, in which algorithms are submitted to a central laboratory and tested on that laboratory's data, so that the data never leaves its owner's premises. The fact that system developers do not have access to the test data has the advantage that it is less likely that their systems will be overfitted to the data, which would lead to overly optimistic performance figures, but it also has the disadvantage that developers do not have the opportunity to analyse and learn from the cases for which their system fails.

In the rest of this section, we describe several evaluations of the beat tracking performance of BeatRoot. First we briefly review published results, then discuss the MIREX 2006 Audio Beat Tracking Evaluation results, and finally we describe a new evaluation of BeatRoot using the largest data set to date.

Previous Tests

BeatRoot was originally tested qualitatively on a small number of audio files covering a range of different musical styles, including classical, jazz, and popular works with a variety of tempi and meters. The tempo induction was correct in each case (the metrical level was not fixed in advance; any musically correct metrical level was accepted), and the errors were that the system sometimes tracked the off-beat.

The first set of quantitative results were obtained on a set of performances of 13 complete Mozart piano sonatas, which had been recorded on a Bösendorfer computer-monitored piano by a professional pianist. For this data set, the onset detection stage was not required; instead, any almost-simultaneous notes were grouped into chords, and a salience calculation was performed based on duration, pitch and intensity of the notes. For the examples where the metrical level agreed with the nominal metrical level suggested by the denominator of the time signature, BeatRoot found an average of over 90% of the beats (Dixon & Cambouropoulos, 2000).

BeatRoot was compared with the system of (Cemgil, Kappen, Desain, & Honing, 2000; Cemgil & Kappen, 2001) on a large collection of solo piano performances of two Beatles songs (Dixon, 2001b). Under the same conditions (initial tempo and phase given), BeatRoot performed slightly better of the two systems. The numerical results were very high for both systems, indicating that the data was quite simple, as an analysis of the musical scores confirmed.

The above evaluations can be summarised in two points: the tempo induction stage of BeatRoot was correct in most cases, as long as there is no insistence on a specific metrical level, that is, if musically related metrical levels such as double or half the subjectively chosen primary rate are accepted. The estimation of beat times was also quite robust; when the system lost synchronisation with the beat, it usually recovered quickly to resume correct beat tracking, only rarely missing the beat for extended periods, for example by tracking the off-beat.

MIREX 2006 Audio Beat Tracking Evaluation

MIREX 2006 was the third annual series of evaluations of music information retrieval tasks, and the first time that the task of beat tracking was evaluated. The goal of this particular task was to evaluate algorithms "in terms of their accuracy in predicting beat locations annotated by a group of listeners" (Brossier, Davies, & McKinney, 2006). In order to create the annotations, the listeners tapped in time with the music, which in general is not equivalent to annotating the times of beats from the performer's perspective, since changes in tempo are reflected in the tapping data after a time lag of 1–2 beats (Dixon et al., 2006). Other issues such as precision of tapping and systematic bias in the taps should also be taken into consideration when using this data. The musical excerpts were chosen as having a constant tempo, so the problem of time lags is not relevant for this data, and the phrasing of the aim of the contest as a prediction of human beat tracking (tapping) means that the annotations are by definition correct, even though there are significant inter-participant differences (e.g. in choice of metrical level) in the data (McKinney et al., 2007). The use of 40 listeners for each musical excerpt (39 listeners for a few excerpts) ensures that individual errors in tapping do not greatly influence results.

The audio files consisted of 160 30-second excerpts, selected to provide: a stable tempo within each excerpt, a good distribution of tempi across excerpts, a variety of instrumentation and beat strengths (with and without percussion), a variety of musical styles, including many non-western styles, and the presence of non-binary meters (about 20% of the excerpts have a ternary meter and a few examples have an odd or changing meter) (Brossier et al., 2006; McKinney & Moelants, 2006). One disadvantage of this evaluation was the use of constant-tempo data, which does not test the ability of beat-tracking algorithms to track tempo changes.

An evaluation method was chosen which implicitly evaluates the choice of metrical level via the explicit evaluation of the timing of beats. The first step was to create a binary vector (*impulse train*) from the algorithm output and from each of the 40 annotated ground truth beat vectors. The vectors were sampled at 100 Hz, and the first 5 seconds of the vectors were deleted, leaving a 2500 point binary vector with a unit impulse (1) at each beat time and 0 elsewhere. If the annotation vectors are denoted by $a_s[n]$, where s is the annotator number (1–40), and the algorithm vector is denoted by $y[n]$, then the performance P of the beat-tracking algorithm for a single excerpt is given by the cross-correlation of $a_s[n]$ and $y[n]$ within a small "error" window W around zero, averaged across the number of annotators S:

$$P = \frac{1}{S} \sum_{s=1}^{S} \frac{1}{F} \sum_{m=-W}^{+W} \sum_{n=1}^{N} y[n] \cdot a_s[n-m]$$

where $N = 2500$ is the length of the binary vectors $y[n]$ and $a_s[n]$, $S = 40$ is the number of annotators, and F is a normalisation factor equal to the maximum number of impulses in either vector:

$$F = \max_s (\sum_n y[n], \sum_n a_s[n]).$$

The window size W is defined as 20% of the median inter-beat interval (IBI) of the annotated taps:

$$W = \text{round}(0.2 * \text{median}(\text{IBI}_s[n]))$$

The results of the MIREX 2006 Audio Beat Tracking Evaluation are shown in Table 1. Of the 5 systems submitted for the evaluation, BeatRoot had the best performance in terms of P-score (although the difference in performance between the top 4 algorithms is not statistically significant), and median performance in terms of speed. We note that BeatRoot was not designed for the task of predicting human tapping, nor for selecting the same metrical level as human annotators, so it would not have been surprising if the numerical results were lower. The fact that the P-score is higher than that of the other systems suggests that they also have been developed for a different task than predicting human beat tracking.

The choice of metrical level has a large impact on the P-score. For example, perfect beat tracking at double or half the annotator's level would result in a P-score of only 50%, although one might argue that it is a musically acceptable solution. The counter-argument is that if an alternative metrical level also represents a reasonable tapping rate, then a good proportion of the annotators would tap at that level, raising the score of the algorithms choosing that level, and bringing down the score of those who chose the first metrical level. Thus the rather harsh scoring on individual annotations is (in theory) averaged out by

Contestant	P-Score (average)	Run-time
Dixon	0.575	639
Davies	0.571	1394
Klapuri	0.564	1218
Ellis	0.552	498
Brossier	0.453	139

Table 1: Results of the MIREX 2006 Audio Beat Tracking Evaluation. (Note that an error was discovered in the original results released in 2006; these figures reflect the corrected results.)

variability between annotators. The agent evaluation function of BeatRoot is known to favour lower (faster) metrical levels than human annotators (see Dixon, 2001a, and the next subsection), so an improvement in the P-score would be achieved by changing the preferences to favour slower rates. Further, the off-line beat tracking as performed by BeatRoot could be modified to simulate on-line tapping behaviour in terms of smoothness of IBIs and lags in response to tempo changes (see Dixon et al., 2006, for a discussion).

Since the results were released as shown in Table 1, that is, summarised as a single P-score, we are unable to perform further analysis. The statistical significance of differences in P-score are reported by McKinney et al. (2007): there was no significant difference between the top 4 algorithms. The results are clearly very close, and we do not know whether the systems' choice of metrical levels was a deciding factor. BeatRoot is not programmed to select the metrical level corresponding to the perceived beat, nor to a typical tapping rate; it tends to prefer faster rates, because they turn out to be easier to track, in the sense that the agents achieve higher scores. More detailed results and analysis would be interesting, but because the data is not available, it is not possible to investigate in this direction.

To compare the beat tracking systems with human tapping ability, McKinney et al. (2007) evaluated each human annotator by comparing their tapping with that of the other annotators. The human annotators achieved average P-scores between about 0.34 and 0.73, with one annotator performing worse than all of the automatic systems, and many performing worse than the top 4 systems. Most of the annotators achieved scores between 0.5 and 0.7, whereas the top 4 systems were very tightly grouped between 0.552 and 0.575, just below the average human score. An interesting task for future years would be to test beat tracking performance for a given metrical level, by providing the systems with the first two beats or the initial tempo, or by separating results according to the chosen metrical level, as we do for the following evaluation.

Large Scale Evaluation of BeatRoot

A new evaluation of BeatRoot was performed using a total of 1360 audio files excerpted from commercial CDs. The files are between 11 s and 1:56 s long, with a minimum of 7 beats and a maximum of 262 beats per piece, totalling 90643 manually annotated beats. The beat annotations were, as far as we are aware, contributed by a single annotator per piece, and some but not all of the annotations were checked carefully and errors were corrected. We have not checked all files for accuracy, but random examination of some of the data has revealed gross inaccuracies in some of the annotations, with beat placement

errors of the order of 100ms being observed. To give an indication of the types of music covered by this evaluation, the data was grouped in the following 10 categories: *Acoustic*, 84 pieces; *Afro-American*, 93 pieces; *Balkan/Greek*, 144 pieces; *Choral*, 21 pieces; *Classical*, 204 pieces; *Classical solo*, 79 pieces; *Electronic*, 165 pieces; *Jazz/Blues*, 194 pieces; *Rock/Pop*, 334 pieces; and *Samba*, 42 pieces. See (Gouyon, 2005) for more details on the data. Audio data is not publicly available for copyright reasons.

In our previous work, evaluation was performed by counting the percentage of correctly tracked beats, where a beat is considered correctly tracked when it is within some fixed error window of an annotated beat. The numbers of correct beats b, false positives p and false negatives n were combined to give a score T between 0 and 1:

$$T = \frac{b}{b+p+n}$$

This measure is harsher than the MIREX P-score, which (for a given window size) punishes the false positives *or* the false negatives (whichever is worse), but not both. Similarly, it is somewhat harsher than the F-measure often used in MIR tasks, which is a combination of precision ($\frac{b}{b+p}$) and recall ($\frac{b}{b+n}$) equal to:

$$F = \frac{b}{b+\frac{p+n}{2}}$$

The P-score as defined above by the MIREX evaluation is equivalent to:

$$P = \frac{b}{b+\max(p,n)}$$

where the correct beats are defined as those which have a distance from an annotated beat of less than 20% of the median inter-beat interval of the annotator's beats. This is a middle ground between the T and F measures, since for any p, n, we have $T \leq P \leq F$, with the inequalities being strict if both p and n are non-zero.

To enable a direct comparison with the MIREX results, the current results are reported as P-scores with the same parameters as used in MIREX, except that the times are not quantised in this work. A further difference is that we show more detailed results, separating results by the metrical level chosen by the beat tracking system relative to the annotator. The results are shown in Tables 2 and 3. Table 2 shows the number of pieces tracked at each metrical level, relative to the metrical level of the annotation. Three conditions were tested, differing in the number of initial beats given to the system. The first condition (no beats given) is the standard case of "blind" beat tracking, where the system is required to find the metrical level and alignment of beats given the audio data only. In the second case, the system is given the time of the first beat, which determines the initial phase of all agents, but not the initial tempo. The third case (2 beats given) determines not only the initial phase but also the initial tempo of the beat tracking agents, and should ensure that the correct metrical level is chosen. The fact that the wrong metrical level was still chosen in several cases could be due to the first beat being non-representative of the tempo (e.g. a slow introduction, or inaccurate tapping by the annotator), or the onset detection performing poorly (e.g. due to the instruments used in the recording).

	Tempo ratio of BeatRoot to annotation							
Condition	1:1	2:1	3:1	4:1	3:2	2:3	1:2	Other
0 beats given	613	535	41	8	61	5	7	90
1 beat given	629	531	40	9	59	4	6	82
2 beats given	1245	10	3	1	8	14	4	75

Table 2: Number of excerpts tracked at each metrical level, relative to the metrical level of the annotation, for 0, 1, or 2 beats given. When the metrical level was not given, BeatRoot tracked about 46% of the pieces at the same rate as the human annotator, and 39% at double this rate.

Given just the audio data as input, BeatRoot chose the same metrical level as the annotator for 613 of the 1360 excerpts (45%). 535 pieces (39%) were tracked at double the rate of the annotation, which agrees with the bias for faster rates noted in previous work (Dixon, 2001a). The remaining pieces were spread over 1.5 times, 3 times, and other ratios of the annotated level, with about 7% of pieces not corresponding to any recognised metrical level (probably due to changing tempo during the excerpt). When 2 initial beats were given to the system, 92% of the excerpts were then tracked at the given metrical level, as expected.

The results for timing of beats are shown in Table 3, expressed as the P-score for each corresponding entry in Table 2, with the overall average in the right column. In other words, Table 2 gives the number of excerpts tracked at each (relative) metrical level for each condition, and Table 3 gives the P-score calculated over these excerpts. For example, given no beat information, BeatRoot tracked 8 pieces at the ratio of 4:1, that is, four times faster than the annotator, and for these 8 pieces, the T-score was 0.27. The reason that it is important to separate the P-scores by metrical levels is that the maximum score (last row of Table 3) is different for each level, as described previously. We note that some of the scores are above the expected maximum — this reveals that BeatRoot did not consistently track at the stated metrical level, but switched to a metrical level where it found more of the annotated beats. The metrical level shown in the results is calculated from the average IBI across the whole excerpt, relative to the average IBI of the annotation, so it is possible that some of these values do not accurately reflect the beat tracking behaviour. By comparing the two tables of results, it can be seen that these anomalies occur only for a small number of excerpts.

The effect of initialising BeatRoot with the first 1 or 2 beats can be seen by comparing the rows of the results. Knowledge of the first beat rarely alters BeatRoot's choice of metrical level, and surprisingly does not greatly improve the P-scores. This suggests that BeatRoot does not often track complete excerpts on the off-beat, but rather, as a side-effect of its ability to track tempo changes, it switches between on-beat (in-phase) and off-beat (out-of-phase) tracking. We did not test the effect of varying the parameters which determine the reactiveness/inertia balance in tracking tempo changes; the default values were used for these experiments.

The right column of Table 3 gives a single score summarising the performance of BeatRoot on this data. We used the same measure as the MIREX evaluation, so that a simple comparison of results could be made. BeatRoot achieved a slightly better perfor-

Condition	Tempo ratio of BeatRoot to annotation								All
	1:1	2:1	3:1	4:1	3:2	2:3	1:2	Other	
0 beats given	.81	.48	.34	.27	.38	.26	.44	.33	.60
1 beat given	.83	.48	.34	.27	.38	.25	.46	.33	.62
2 beats given	.80	.45	.33	.26	.41	.33	.32	.33	.77
Theoretical maximum	1.0	.50	.33	.25	.25	.25	.50	—	—

Table 3: P-scores for beat tracking, with each column showing the scores for the cases where the beat is tracked at each metrical level relative to the annotation (see Table 2), for the three conditions (0, 1, or 2 beats given).

mance on this data (0.60) than on the MIREX competition data (0.575). We speculate that the reason is that this data set contains a higher proportion of music which is easy to track, e.g. Western pop music; whereas the MIREX data was chosen to represent a wider range of styles and tempi, including some quite obscure examples. We also do not know what the maximum possible score would be on the MIREX data set. It is impossible to score near 100%, because the annotators tapped at different metrical levels in many cases. Human annotators achieved P-scores in the range 0.34 to 0.73, with the majority of them scoring between 0.5 and 0.7 (McKinney et al., 2007).

Conclusions and Further Work

We have described a system for automatic tracking and annotation of the beat for a wide range of musical styles, subject to a few limitations. BeatRoot assumes that the music has a reasonably regular beat, with no large discontinuities; it does not answer the question of whether or not a piece of music has a beat. Also, although the tempo induction and beat tracking algorithms are not directly dependent on the instrumentation, they are reliant on the onset detection algorithm for processing the audio data. Thus the system does not work well on music for which few onsets are found. The results of several evaluations indicate that BeatRoot functions well across a range of styles, and that beat tracking accuracy is not dependent on musical style directly, but rather on rhythmic complexity (Dixon, 2001b).

There are a number of parameters which can be varied in order to tune the behaviour of the system, but as the system was designed to work autonomously, we generated the results in this paper without any fine-tuning of the parameters, that is, using the system's default values. In complex music there are competing rhythmic forces, and higher level knowledge of the musical structure makes the correct interpretation clear to a human listener. The beat tracking agents do not make use of such high level knowledge, and therefore their decisions are influenced by more arbitrary factors such as the numerical values of parameters. Despite the beat tracking system's lack of higher level musical knowledge, such as notions of off-beats or expected rhythmic patterns, it still exhibits an apparent musical intelligence, which emerges from patterns and structure which it finds in the data, rather than from high-level knowledge or reasoning. This makes the system simple, robust and general.

In order to disambiguate more difficult rhythmic patterns, some musical knowledge is necessary. This can take the form of salience values for onsets (Dixon & Cambouropoulos,

2000), high-level knowledge of stylistic expectations, or knowledge of the score of the piece being tracked (either in a symbolic format or as audio with metadata). The improvement in performance due to higher level information comes at the expense of generality. For performance analysis, the use of audio alignment (Dixon & Widmer, 2005) can be combined with or even replace beat tracking as an efficient means of annotation with metrical metadata (Dixon, 2007).

There are many avenues open for further work to improve or extend BeatRoot. Although BeatRoot is not intended as a real-time system, the approach is sufficiently fast, and the algorithms could be modified to perform tempo induction on small windows of data, and continually feed the tempo estimates to the beat tracking agents. Some modifications would need to be performed to ensure smoothness and consistency when the winning agent changes.

The beat tracking agents ignore any onsets which lie between hypothesised beats, although these provide potentially useful rhythmic information. A possible source of improvement would be to program the agents to use this information, so that each hypothesis corresponds to a complete rhythmic parse of the onset data. Alternatively, agents tracking different metrical levels could communicate with each other to ensure consistency in their interpretation of the data, which could also lead to a complete rhythmic parse.

Finally, although the system is not a model of human perception, further comparison between the correctness and accuracy of the system and of human subjects would be interesting, and would shed light on the more difficult evaluation issues, perhaps leading to a clearer understanding of beat tracking. It is not known what the limits of human beat tracking performance are, but the MIREX results suggest that current computational models are approaching human beat tracking ability.

Acknowledgements

The author acknowledges the support of the UK Engineering and Physical Sciences Research Council (EPSRC) for the OMRAS2 project (EP/E017614/1). Most of this work was performed while the author was at the Austrian Research Institute for Artificial Intelligence (OFAI). Thanks to the proposers of the MIREX Audio Beat Tracking Evaluation, and the team that conducted the evaluation. Thanks also to Fabien Gouyon for providing evaluation data, and to all who performed annotations or contributed to BeatRoot over the last 7 years.

References

Bello, J., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. (2005). A tutorial on onset detection in musical signals. *IEEE Transactions on Speech and Audio Processing*, *13*(5), 1035–1047.

Brossier, P., Davies, M., & McKinney, M. (2006). *Audio beat tracking – MIREX 2006.* http://www.music-ir.org/mirex2006/index.php/Audio_Beat_Tracking.

Cemgil, A., & Kappen, B. (2001). Tempo tracking and rhythm quantisation by sequential monte carlo. In *Advances in neural information processing systems.*

Cemgil, A., Kappen, B., Desain, P., & Honing, H. (2000). On tempo tracking: Tempogram representation and Kalman filtering. In *Proceedings of the international computer music conference* (pp. 352–355). San Francisco CA: International Computer Music Association.

Collins, N. (2005). A comparison of sound onset detection algorithms with emphasis on psychoacoustically motivated detection functions. In *118th convention of the audio engineering society.* Barcelona, Spain.

Davies, M., & Plumbley, M. (2007). On the use of entropy for beat tracking evaluation. In *Proceedings of the 2007 international conference on acoustics, speech and signal processing* (Vol. IV, pp. 1305–1308).

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

Dixon, S. (2001b). An empirical comparison of tempo trackers. In *Proceedings of the 8th brazilian symposium on computer music* (pp. 832–840). Brazilian Computing Society.

Dixon, S. (2001c). An interactive beat tracking and visualisation system. In *Proceedings of the international computer music conference* (pp. 215–218).

Dixon, S. (2006a). Onset detection revisited. In *Proceedings of the 9th international conference on digital audio effects* (pp. 133–137).

Dixon, S. (2006b). *MIREX 2006 audio beat tracking evaluation: BeatRoot.* http://www.music-ir.org/evaluation/MIREX/2006_abstracts/BT_dixon.pdf.

Dixon, S. (2007). Tools for analysis of musical expression. In *19th international congress on acoustics.*

Dixon, S., & Cambouropoulos, E. (2000). Beat tracking with musical knowledge. In *ECAI 2000: Proceedings of the 14th European conference on artificial intelligence* (pp. 626–630). Amsterdam: IOS Press.

Dixon, S., Goebl, W., & Cambouropoulos, E. (2006). Perceptual smoothness of tempo in expressively performed music. *Music Perception*, *23*(3), 195–214.

Dixon, S., Goebl, W., & Widmer, G. (2002). Real time tracking and visualisation of musical expression. In *Music and artificial intelligence: Second international conference (ICMAI2002)* (pp. 58–68). Edinburgh, Scotland: Springer.

Dixon, S., Gouyon, F., & Widmer, G. (2004). Towards characterisation of music via rhythmic patterns. In *5th international conference on music information retrieval* (pp. 509–516).

Dixon, S., & Widmer, G. (2005). MATCH: A music alignment tool chest. In *6th international conference on music information retrieval* (pp. 492–497).

Downie, J. (2005). *2005 MIREX contest results - audio onset detection.* www.music-ir.org/evaluation/mirex-results/audio-onset.

Duxbury, C., Sandler, M., & Davies, M. (2002). A hybrid approach to musical note onset detection. In *Proceedings of the 5th international conference on digital audio effects* (pp. 33–38).

Goto, M., & Muraoka, Y. (1997). Issues in evaluating beat tracking systems. In *Issues in AI and music – evaluation and assessment: Proceedings of the IJCAI'97 workshop on AI and music* (pp. 9–16). International Joint Conference on Artificial Intelligence.

Gouyon, F. (2005). *A computational approach to rhythm description.* Unpublished doctoral dissertation, Pompeu Fabra University, Barcelona, Audio Visual Institute.

Gouyon, F., & Dixon, S. (2005). A review of automatic rhythm description systems. *Computer Music Journal*, *29*(1), 34–54.

Gouyon, F., Dixon, S., & Widmer, G. (2007). Evaluating low-level features for beat classification and tracking. In *Proceedings of the 2007 international conference on acoustics, speech and signal processing* (Vol. IV, pp. 1309–1312).

Gouyon, F., Klapuri, A., Dixon, S., Alonso, M., Tzanetakis, G., & Uhle, C. (2006). An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech and Language Processing*, *14*(5), 1832–1844.

Klapuri, A. (1999). Sound onset detection by applying psychoacoustic knowledge. In *Proceedings of the IEEE international conference on acoustics, speech and signal processing.* Phoenix, Arizona.

McKinney, M., & Moelants, D. (2006). Ambiguity in tempo perception: What draws listeners to different metrical levels? *Music Perception*, *24*(2), 155–166.

McKinney, M., Moelants, D., Davies, M., & Klapuri, A. (2007). Evalutation of audio beat tracking and music tempo extraction algorithms. *Journal of New Music Research*, to appear.

Saunders, C., Hardoon, D., Shawe-Taylor, J., & Widmer, G. (2004). Using string kernels to identify famous performers from their playing style. In *Proceedings of the 15th European conference on machine learning.*

Schloss, W. (1985). *On the automatic transcription of percussive music: From acoustic signal to high level analysis.* Unpublished doctoral dissertation, Stanford University, CCRMA.

Stamatatos, E., & Widmer, G. (2005). Automatic identification of music performers with learning ensembles. *Artificial Intelligence*, *165*(1), 37–56.

Widmer, G. (2002). Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research*, *31*(1), 37–50.

Widmer, G., Dixon, S., Goebl, W., Pampalk, E., & Tobudic, A. (2003). In search of the Horowitz factor. *AI Magazine*, *24*(3), 111–130.