A Beat Tracking System for Audio Signals

Simon Dixon
Austrian Research Institute for Artificial Intelligence,
Schottengasse 3,
A-1010 Vienna, Austria.
simon@ai.univie.ac.at

October 29, 1999

Abstract

We present a system which processes audio signals sampled from recordings of musical performances, and estimates the tempo at each point throughout the piece. The system employs a bottom-up approach to beat tracking from acoustic signals, assuming no a priori high-level knowledge of the music such as the time signature or approximate tempo, but rather deriving this information from the timing patterns of detected note onsets. Results from the beat tracking of several popular songs are presented and discussed.

1 Introduction

Although most people can tap their foot in time with a piece of music, equivalent performance on a computer has proved remarkably difficult to emulate. In this paper, we present a system which processes audio signals sampled from recordings of musical performances, and estimates the tempo at each point throughout the piece. The system has been tested with various types of popular music, and produces tempo estimates as accurately as the performers keep to the tempo.

We do not attempt to model or describe the cognitive mechanisms involved in human rhythm perception, but we do note certain features of perception which motivate an ambitious unsupervised approach to the beat tracking problem. Firstly, human rhythm perception sets its own parameters; the tempo and the metrical structure are not specified explicitly at the beginning of a piece, and if they change suddenly during the piece, the perceptual system is able to adjust itself within seconds to the new listening framework. Secondly, it copes well with "noise" in the input, that is, deviations from precise timing are allowed, as are variations in tempo, without disturbing the overall perception of the music. Thirdly, it is able to cope with syncopation, that is, sections of music where more salient events occur between the beats and less salient events (or perhaps no event at all) occur on the beat.

In contrast with these capabilities, computer music software does not cope well in these situations. Commercial sequencing and transcription programs usually require

the beat to be declared explicitly before the music is processed, so that all data can then be indexed relative to this given beat. Even many research systems are limited by the fact that once they get out of synchronization with the music, it is very difficult for them to recover and resume correct interpretation of the rhythmic structure [2]. The robustness of human perception is one feature which is extremely difficult to reproduce in a computer system.

In this paper, we present a bottom-up approach to beat tracking from acoustic signals. We assume no a priori high-level knowledge of the music such as the time signature or approximate tempo, but attempt to derive this information from the timing patterns of detected note onsets. We distinguish the tasks of *beat induction*, which involves estimating the tempo and location of the main rhythmic pulse of a piece of music, and *beat tracking*, which is the subsequent estimation of tempo fluctuations in the light of previous tempo estimations.

We conclude this section with a brief outline of the paper: the following section contains a review of related work; section 3 describes the lowest level details of the system, detecting the onsets of musical events from the raw audio data; the theoretical model of musical time is then briefly discussed in section 4, as are the assumptions made about the musical data to be analyses; section 5 presents the beat induction algorithm which creates classes of similar inter-onset intervals as a foundation for estimating the inter-beat interval; then, in section 6, we present the results produced by the system; the final section concludes the paper with a discussion of the results, design issues and future research directions.

2 Related Work

A substantial amount of research has been performed in the area of rhythm recognition by computer, including a demonstration of various beat tracking methods using a computer attached to a shoe which tapped in time with the calculated beat of the music [5]. Many of these methods cannot be compared quantitatively, as they process different forms of input (MIDI vs audio), or make different assumptions about the complexity or style of the music being analyses, or rely on user interaction.

Much of the work in machine perception of rhythm has used MIDI files as input [12, 3, 10], which contain control information for a synthesizer instead of audio data. MIDI files consist of chronologically ordered sequences of events, such as the onsets and offsets of notes (usually corresponding to pressing and releasing keys on a piano-style keyboard), and timing information representing the time delays between successive pairs of events. Type 1 MIDI files also allow for the encoding of structural information such as the time signature and tempo, but most research in this area presumes that this information is not available to the rhythm recognition program. The other types of information present in MIDI files are not relevant to this work and shall not be discussed here.

Using MIDI files, the input is usually interpreted as a series of inter-onset intervals, ignoring the offset times, pitch, amplitude and chosen synthesizer voice. That is, each note is treated purely as an uninterpreted event. It is assumed that the other param-

eters do not provide *essential* rhythmic information, which in many circumstances is true. However, there is no doubt that these factors provide useful rhythmic cues, as more salient events tend to occur on stronger beats. Another factor that is not usually considered in this work is the possibility of separating parts using principles of auditory streaming [1], which relies heavily on frequency and timbral information.

Although the use of MIDI input simplifies the task of rhythm recognition by sidestepping the problem of onset detection, it is still valuable to examine these approaches, as they correspond to the subsequent stages of analysis after onset detection has been performed.

Notable work using MIDI file input is the emulation of human rhythm perception by Rosenthal [12] which produces multiple hypotheses of possible hierarchical structures in the timing, assigning a score to each hypothesis, corresponding to the likelihood that a human listener would choose that interpretation of the rhythm. This technique gives the system the ability to adjust to changes in tempo and meter, as well as avoiding many of the implausible rhythmic interpretations produced by commercial systems.

A similar approach is advocated by Tanguiane [16], who uses Kolmogorov complexity as the measure of the likelihood of a particular interpretation, with the least complex interpretations being favoured. He presents an information-theoretic account of human perception, and argues that many of the "rules" of music composition and perception can be explained in information-theoretic terms.

Desain [3] compares two different approaches to modeling rhythm perception, the symbolic approach of Longuet-Higgins [11] and the connectionist approach of Desain and Honing [4]. Although this work only models one aspect of rhythm perception, the issue of quantization, and the results of the comparison do not provide a definitive preference for one style over the other, it does highlight the need to model expectancy, either explicitly or implicitly. Expectancy, as described in the work cited above, is a type of predictive modeling which is particularly relevant to real-time processing as it provides a contextual framework in which subsequent rhythmic patterns can be interpreted with less ambiguity.

An alternative approach uses a nonlinear oscillator to model the expectation created by detecting a regular pulse in the music [10]. A feedback loop controls the frequency of the oscillator so that it can track variations in the rhythm. This system performs quite robustly, but due to its intricate mathematics it does not correspond to any intuitive notion of perception, and in this sense is very similar to connectionist approaches.

One early project on rhythm using audio input was the percussion transcription system of Schloss [13]. Onsets were detected as peaks in the slope of the amplitude envelope, where the envelope was defined to equal the maximum amplitude in each period of the sound, and the period defined as the inverse of the lowest frequency expected to be present in the signal. The audio signal was high-pass filtered to obtain more accurate onset times. The limitations of the system were that it required parameters to be set interactively, and it was evaluated only by resynthesis of the signal.

A more complete approach to beat tracking of acoustic signals was developed by

Goto and Muraoka [7, 8, 9]. They developed two systems for following the beat of popular music in real time. The earlier system (BTS) used frequency histograms to find significant peaks in the low frequency regions, corresponding to the frequencies of the bass and snare drums, and then tracked these low frequency signals by matching patterns of onset times to a set of pre-stored drum beat patterns. This method was successful in tracking the beat of most of the popular songs on which it was tested. A later system allowed music without drums to be tracked by recognizing chord changes, assuming that significant harmonic changes occur at strong rhythmic positions.

Commercial transcription and sequencing programs do not address the issues covered by these research systems. It is generally assumed that the tempo and time signature are explicitly specified before the music is played, and the system then aligns each note with the nearest position on a metrical grid. Recent systems allow parameterization of this grid in terms of a resolution limit (the shortest allowed note length) and also various restrictions on the complexity of rhythm, such as the use of tuplets, that can be produced by the system. Nevertheless, these systems still produce implausible rhythmic interpretations, and cannot be used in an unsupervised manner for anything but simple rhythms.

3 Processing of Audio Data

In this and the following sections, we describe the successive stages of processing performed by the beat tracking system. The input to the system is a digitally sampled acoustic signal, such as is found on audio compact discs. In this paper, the stereo compact disc data was converted to a single channel format by averaging the left and right channels, resulting in a single channel 16 bit linear pulse code modulated (PCM) format, with a sampling rate of 44.1kHz. All of the software was written in C++ and runs under Solaris and Linux. The complete processing of a song takes about 20 seconds of CPU time on a current PC, so the system could be used for real-time applications, but it is not currently built to be used in real-time.

The aim of the initial signal processing stage is to detect *events* in the audio data, from which rhythmic information can be derived. For the purposes of this work, events correspond to note onsets, that is, the beginnings of musical notes, including percussive events. By ignoring note durations and offset times, we discard valuable information, but our results justify the assumption that there is sufficient information in note onset times to perform beat tracking.

In previous work [6] we used multi-resolution Fourier analysis to detect events. In this work, a simpler time domain method is employed, based on [13], which gives equally good results. This method involves passing the signal through a simple high-pass filter, then calculating the absolute sum of small overlapping windows of the signal, and finding peaks in the slope of these window sums using a 4 point linear regression. Onset times are detected reliably and accurately with this method, which is essential for the determination of tempo.

4 Modeling Musical Time

The formal model of musical time underlying this work, which will not be discussed at length in this paper, defines the tempo of a performance as a piecewise constant non-negative function of time (i.e. a step function), which has units of beats per second, and is constrained to lie within some arbitrary bounds consistent with human perception and standard musical notation. This model is not a cognitive or perceptual model, but it is intended at least to be plausible from the cognitive perspective, as well as from an information theoretic viewpoint.

The tempo function is restricted further in that it may only change value at a note onset. This is justified on the basis that no information about tempo can be provided between musical events. (It is possible that rhythmic information could be inferred from data within a musical event, such as speed of vibrato, but this is considered to be a secondary effect, not one that provides conclusive rhythmic information.) As already noted, the durations of notes play a part in rhythm perception, but are not used in this work.

It remains to define precisely how quickly the tempo can change – arbitrary leaps at each function value weaken the perception of tempo, and do not provide sufficient information for beat tracking to be meaningful. A solo piano piece played *molto rubato* is a case in point: although there may be a beat notated in the score, it is unlikely that a listener unfamiliar with the score would have sufficient information from listening to a performance to reconstruct the score unambiguously.

In this work, it is assumed that the musical data has a recognizable and stable tempo (as perceived by human listeners), as is true of most popular music and dance music. It is planned to extend the software to perform automatic segmentation into stable sections, but currently we do not allow for changes in meter or sudden large changes in tempo; instead we require that such pieces be segmented into smaller units which are processed separately.

The data was chosen from a range of modern popular musical styles (e.g. pop, salsa, folk and jazz), all containing multiple instruments. We expect that beat tracking the music of a solo performer would be more difficult, as solo performers do not need to synchronize their playing with any other performers. In an ensemble situation, it is necessary for the performers to give each other timing cues, which often come through the performed music itself.

5 Beat Induction

The beat induction section of the system aims to develop a local model of the tempo, and to use that to determine the local structure. As each local value is determined, it can be compared with previous values and adjusted to satisfy a continuity constraint, reflecting the assumption that the local tempo will not change significantly between areas. This is more likely to be true where overlapping time windows are used, as in this work.

Once the onsets have been detected, we analyze the elapsed time between the

onsets of near pairs of notes. These times are often called *inter-onset intervals (or IOI's)* in the literature, but usually only refer to the times between successive onsets. In our work, we extend the term to include times between onsets of events that have other event onsets occurring between them. It does not make sense to examine all pairs of onset times, since even a small tempo variation will result in a significant change in an inter-onset interval containing many beats. (We could also argue that the limitations of human temporal memory imply that tempo information can only be provided by local features of the music.) Therefore we set an upper bound on the length of inter-onset intervals that we examine. In the algorithm below, the upper bound is labelled IOI_Limit, which was set to 2.5 seconds in this work.

Results from psychoacoustic research suggest that there are limits on the accuracy of production and perception of timing information in music which also may be used to set parameters for beat tracking analysis. It is known that deviations of up to 40ms from the timing indicated in the score are not uncommon in musical performances, and often go unnoticed by listeners [15]. This allows us to group inter-onset intervals into classes which are considered sufficiently similar to be perceived as the same interval. These classes are characterized by the average size of their members, and new members are added if their sizes are close enough to this average. Closeness is defined in absolute terms by the constant *Resolution* in the algorithm below. If an interval does not fit into any existing class, a new class is created for it.

Note that the process of adding an interval to a class automatically adjusts the average of the members, so that the class boundaries are not rigid, but may drift over a period of time. It is important that these classes are not constructed over too long a time window, or else tempo variations may corrupt the accuracy of results. This is a disadvantage of using averaging, which is intended to be outweighed by the smoothing of random errors. In this work, time windows of 5-10 seconds were used.

An alternative to the current approach of limiting the time window in which intervals are examined is to timestamp each of the intervals and delete them from the classes once they reach an "expiry age". This technique has yet to be tested.

The grouping algorithm as used in previous work [6] is shown below:

Algorithm: Generate_Classes

For each pair of onset times t_1, t_2 (with $t_1 < t_2$)
    If $t_2 - t_1 < IOI\_Limit$ (maximum distance between intervals)
        $I := t_2 - t_1$
        Find class C_n such that $|Average(C_n) - I|$ is minimum
        If $|Average(C_n) - I| < Resolution$ then
            $C_n := C_n \cup \{I\}$
        Else
            Create new class $C_m := \{I\}$
        End If
    End If
End For

For each class generated, we calculate a score based on the number of intervals in the class and the agreement of the lengths of the intervals. This gives a ranking of classes, most of which are often integer multiples or sub-multiples of the beat. Each score is adjusted to reflect the scores of other intervals which are related in this way, and a final best estimate of the inter-beat interval is determined.

This technique gives a reasonably reliable estimate of the inter-beat interval, and when combined with some continuity constraints, successfully calculated the beat on all data tested (see the results section). But it does not calculate the location of the beat. That is, by analogy with wave theory, it calculates the frequency but not the phase of the beat. We use the term phase here, but note that we measure it in fractions of beats rather than radians, so that integer values of phase correspond precisely with beat times.

We present two methods of phase calculation, and then discuss their relative merits. The first method divides the beat into a number of equal sized units, and counts the number of onsets that occur within (or near) each of these units. The onset times are normalized by the beat and then adjusted to a value between 0 and 1 by discarding the integer portion of the normalized onset time, which gives a representation of the onset position within the beat in which the onset occurs. The unit with the maximum number of onsets is chosen to be the beat position, under the assumption that the greatest number of events occur on the beat.

The second approach to phase calculation assumes only that at least one event e lies on the beat, and calculates the "goodness" or "badness" for each other onset time that results from choosing the onset of e as defining the beat position. To do this, we must first choose values for each position within a beat, representing whether events are expected or not expected to occur at that position, in order to define the goodness and badness measures. The goodness measure rewards beat positions for each event that occurs at that position, as well as for events occurring at half-beat and other fractional beat positions. The badness measure penalizes positions for each event which is not explainable as an onset time which is a simple fraction of a beat.

Neither of these techniques produce a sufficiently reliable estimate of phase. The main difficulty with phase calculations is that they are extremely sensitive to errors in the inter-beat interval, because they are measured in fractions of a beat, and the tempo error is multiplied by the number of beats from the beginning of the window to the event in question. Also, it is not possible to average phase values, as the actual positions of events are unknown, and it is only meaningful to average the phases of events in the same relative position within the beat. In current work, we are developing a multiple-hypothesis extension to the second approach, which has proved to be successful in tracking the beat throughout complete songs.

6 Results

One of the most difficult tasks in this work is to evaluate the results, as there is no definitive meaning of *beat* for performed music. One could define the beat in relation to the score, if scores were available for the music being tested. In the case of popular

music, complete scores are not generally available, but even for classical music and synthesizer performances where the score is available, there is no formal model of beat covering all possible musical scores. That is, given an accurate performance of arbitrary score, it is not always clear what a beat tracking system should produce. The reason for the problem is that there is no one-to-one mapping between scores and performances; many different scores can produce the same performance, and vice versa. Nevertheless, for a large amount of music, there is at least a socially agreed definition of beat (consider dancing), and in this work we only consider music with such an "agreed" beat.

To test the results of the beat tracking system, the inter-beat intervals were calculated manually from the positions of salient events in the audio signal. That is, the sound files were segmented at beat boundaries, and the length of each segment was divided by the number of beats to give an average inter-beat interval for the segment. We also calculated error margins for the inter-beat intervals by estimating the error in determining the beat locations for segmentation. The error in locating an event was estimated to be 10ms. This low error bound was made possible by only performing segmentation where percussive events occurred on a beat. There was no error in determining the number of beats in a segment; this was simply a matter of counting. Having calculated the error in the inter-beat interval to be between 0.1% and 0.2%, this error was ignored, as it was negligible compared to the variations in tempo. By using smaller segments we could achieve smaller tempo variations at the expense of greater error in the inter-beat interval and much more human effort, but gaining maximal information from our results.

The following table shows the results for initial beat induction in 6 songs, where the system is given a 10 second fragment of the song with no contextual information (previous or subsequent beat computation). The errors refer to the difference between the system's value and the value derived manually for that section. The row labelled *Variation* contains the range of variation in manually computed inter-beat intervals between different segments of each song. Since the manually derived values are the average values for each segment, the maximum deviation is likely to be larger than the average. Also, because the exact values are not calculated for each 10 second segment, one cannot expect precise agreement between the measured and calculated values. Nevertheless, it is clear that within the range of measured deviation, the initial beat induction performed on any 10 second fragment of these songs is correct in well over 90% of cases.

Errors	Song 1	Song 2	Song 3	Song 4	Song 5	Song 6
$< 1\%$	60.4%	32.3%	67.7%	30.5%	20.5%	80.4%
1% to 2%	32.1%	28.2%	28.3%	27.3%	27.9%	19.6%
2% to 3%	6.7%	12.1%	4.0%	18.8%	27.4%	0.0%
3% to 5%	0.0%	14.5%	0.0%	22.7%	16.7%	0.0%
$> 5\%$	0.7%	12.9%	0.0%	0.8%	7.4%	0.0%
Variation	2.2%	3.0%	1.9%	5.2%	6.5%	0.9%

Table 1: Beat induction results for 6 popular songs

When beat tracking is performed throughout a whole song, the contextual information is sufficient to correct all of the errors. For all of the songs tested, there is no more than one value with greater than 5% error in the first 30 values calculated, so the system is able to lock in to the correct tempo and reject the incorrect values almost immediately.

7 Discussion and Future Work

We have described a beat tracking system which analyses acoustic data, detects the salient note onsets, and then discovers patterns in the intervals between the onsets, from which the most likely inter-beat interval is induced. Errors in the inter-beat interval estimates are corrected by comparison with previous values, under the assumption of a slowly changing tempo. The system is successful in tracking the beat in a number of popular songs.

There are many ways in which the system can be improved. The use of other data apart from onset times would give the beat tracking system more information, which would allow more intelligent processing of the data. Amplitude, pitch and duration all give important rhythmic cues, which are currently not used by the system.

The design of the software is a modular design with a low degree of coupling between modules, as recommended by software engineering principles. So the data is processed in a bottom-up fashion, from raw audio to onset data to inter-beat interval estimates, without any feedback from the higher levels to the lower levels of abstraction. This simplifies the construction and maintenance of the software, but denies the powerful processing achievable using multiple feedback paths, as exist in the human brain. A strong argument for combining bottom-up and top-down processing for this type of work is found in [14].

The use of manual beat tracking for evaluation of the system limits the amount of testing that can be performed, but is necessary if we are to analyze performed music. It would also be useful to perform a study of beat tracking in synthetically generated music, where the variations in tempo and onset times can be controlled precisely.

The intended application for this work is as part of an automatic music transcription system. In previous work [6], we discussed how subsequent processing can generate structural information such as the time signature of the music, and also began to address the issue of quantization. In further work, these issues will be revisited, and the system will also be extended to perform score extraction of classical music performances. Other current work is focussed on the precise calculation of beat location, that is, beat phase.

8 Acknowledgements

This research is part of the project Y99-INF, sponsored by the Austrian Federal Ministry of Science and Transport in the form of a START Research Prize. The author also wishes to thank Emilios Cambouropoulos for many helpful discussions on beat tracking.

References

[1] A.S. Bregman. *Auditory Scene Analysis: The Perceptual Organisation of Sound.* Bradford, MIT Press, 1990.

[2] R.B. Dannenberg. Recent work in real-time music understanding by computer. *Proceedings of the International Symposium on Music, Language, Speech and Brain*, 1991.

[3] P. Desain. A connectionist and a traditional AI quantizer, symbolic versus sub-symbolic models of rhythm perception. *Contemporary Music Review*, 9:239–254, 1993.

[4] P. Desain and H. Honing. Quantization of musical time: A connectionist approach. *Computer Music Journal*, 13(3), 1989.

[5] P. Desain and H. Honing. Foot-tapping: a brief introduction to beat induction. In *Proceedings of the International Computer Music Conference*, pages 78–79. Computer Music Association, San Francisco CA, 1994.

[6] S.E. Dixon. Beat induction and rhythm recognition. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 311–320, 1997.

[7] M. Goto and Y. Muraoka. A real-time beat tracking system for audio signals. In *Proceedings of the International Computer Music Conference.* Computer Music Association, San Francisco CA, 1995.

[8] M. Goto and Y. Muraoka. Real-time rhythm tracking for drumless audio signals – chord change detection for musical decisions. In *Proceedings of the IJCAI'97 Workshop on Computational Auditory Scene Analysis.* International Joint Conference on Artificial Intelligence, 1997.

[9] M. Goto and Y. Muraoka. An audio-based real-time beat tracking system and its applications. In *Proceedings of the International Computer Music Conference.* Computer Music Association, San Francisco CA, 1998.

[10] E.W. Large. Beat tracking with a nonlinear oscillator. In *Proceedings of the IJCAI'95 Workshop on Artificial Intelligence and Music.* International Joint Conference on Artificial Intelligence, 1995.

[11] H.C. Longuet-Higgins. *Mental Processes.* MIT Press, 1987.

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

[13] W.A. Schloss. *On the Automatic Transcription of Percussive Music — From Acoustic Signal to High Level Analysis.* PhD thesis, CCRMA, Stanford University, 1985.

[14] M. Slaney. A critique of pure audition. In *Proceedings of the IJCAI'95 Computational Auditory Scene Analysis Workshop.* International Joint Conference on Artificial Intelligence, 1995.

[15] J. Sundberg. *The Science of Musical Sounds.* Academic Press, 1991.

[16] A.S. Tanguiane. *Artificial Perception and Music Recognition.* Springer-Verlag, 1993.