

# Beat and Tempo Induction for Music Performance

Joseph Malloch  
Music Technology Area  
McGill University  
Montreal, Canada  
joseph.malloch@mail.mcgill.ca

**Abstract**—This paper describes an adaptation and implementation of existing beat-induction algorithms for use as a real-time control parameter of an interactive music system. Although beat tracking has been implemented for various purposes, approaches have focused on traditional western music in which the tempo is generally constant. Leveraging existing beat-tracking algorithms for use as a parametric control in music performance appears to be a novel use in the field. The system was tested with both recorded and live percussion music, and used to control real-time adaptive audio effects. Various improvements are proposed.

## I. INTRODUCTION

Music from the Western and many other music traditions is commonly considered to be organized temporally around regularly repeating pulses, or *beats* [1]. In addition, these pulses are often further organized in varying hierarchical levels corresponding to the concept of *meter* (regularly repeating patterns of distinctively accented beats). The beat may also be sub-divided, into the *tatum* (variously, a “temporal atom” [2] or “time quantum” [3]) representing the shortest regularly occurring durations in the rhythm. Most approaches to beat tracking follow these divisions, calling for separate low-level beat induction and higher-level meter induction or grouping models [4] [5] [6] [7]. The temporal frequency of beats - the *tempo* (usually expressed in beats per minute rather than Hz) - is easily found once the beat period is known.

The desire to solve what Toiviainen terms the “foot-tapping problem” [8] - to automatically track beat and tempo - has varying roots. Goto and Muraoka consider five purposes for tracking beats: understanding human rhythm perception, building a computational model understanding rhythm, following live performances, beat-quantization for automatic transcription, and synchronization of computer graphics with music [9]. Models intended for live performance, such as those of Toiviainen [8], are typically used to control the speed and phase of playback of a pre-determined MIDI accompaniment, while interactive systems intended to closely “listen” to an improvising performer and respond more freely have been implemented [2] but typically use rhythmic pattern-recognition rather than tempo or beat following.

Beat-tracking implementations have typically been designed with the assumption that the meter and tempo of music to be tracked is generally constant, and that variations in human performance will be small [6] [9]. These restrictions, however, are not universally found in modern music. From the strict metric modulation of Elliott Carter and additive rhythms of

Olivier Messiaen, to non-Western musics in which a regular beat may not be perceptible, many examples exist that do not fit these assumptions. If beat tracking is already used to analyze and interact with jazz [8], rock and pop [6], and European folk music [1], to name a few, it seems more than appropriate to apply it to “experimental” music.

The following sections outline a system designed to use beat induction as a tool for controlling an interactive music system (IMS). The goal was to allow a performer to use long and short-duration static tempos, beat placement, and modulation of tempo and meter as controls to affect the response of the IMS. With appropriate output mapping, this approach could be considered to be *adaptive control of audio effects* (A-DAFx) [10]. An interesting aspect of this use (or mis-use) of beat-tracking algorithms is that success or failure of the system is evaluated qualitatively (did it sound good?) rather than quantitatively. In this way, even exceeding the limitations of a particular approach or algorithm, whether deliberately or by accident, can lead to aesthetically pleasing results.

## II. PREVIOUS APPROACHES

Previous approaches to solving the beat tracking problem are divided between those that act on event data such as MIDI files or a sequence of IOIs (inter-onset intervals) [4] [11] [8] [12] [1] or audio signal [7] [6] [9] [3]. They may also be distinguished by whether they consider only past information, what Scheirer terms a “process model” [7], or include information about future events in their calculations. Even considering only real-time implementations (which, needless to say, must use process models), there are still many to choose from. Large and Kolen [13] proposed a model using adaptive oscillators that synchronize automatically with the supposed period and phase of a stream of onsets. This approach was improved by Toiviainen [8] for a real-time MIDI score-following implementation, by weighting the effectiveness of incoming notes based on their IOI’s, with longer notes ascribed more rhythmic importance than shorter notes. Toiviainen and Snyder [12] developed this further by also considering pitch height and tonal significance in the weighting of input values, and by using many dynamically-created oscillators and continually picking the best-fit oscillator. Scheirer, however, considered the above methods, and others using IOI’s or MIDI data, to be *transcriptive* models for beat analysis, and unwieldy for beat induction from audio sources [7]. He advocated instead a *psychoacoustically informed* approach using amplitude en-

velopes of several spectral bands and banks of tuned resonant filters. Although similar to the autocorrelation methods for beat induction, analysis of the resonant filters used in Scheirer’s method can yield information on the phase of the signal as well as its tempo. Functionally, the use of many resonant filters closely parallels Rowe’s use of “multiple attractors” [4] and implementations that consider “multiple hypotheses” for the correct tempo. [14] [6] [9]

### III. LIMITATIONS OF BEAT TRACKING

Although much has been accomplished in the area of beat induction, previous and present systems still have many limitations. In order to reduce potential errors, systems are often deliberately limited in range sensitivity [6], may be designed to expect a roughly constant tempo input [9], or may require initialization with starting values [8]. Toiviainen also refers to a necessary compromise between responsiveness and stability of output [8] - parameters should be set to optimize the system’s output, but optimal values may vary widely between musical styles. Depending on the implementation approach, the system may also require “start-up time” before correctly matching the tempo [7].

The two most common errors generated by beat induction models are *harmonic relation mistakes* - in which the estimated output might be half or double the correct value - and  *$\pi$ -phase mistakes* - in which the system indicates that the beat occurs on the “upbeat” rather than the “downbeat” [9]. Avoiding these errors is not trivial, and many different solutions have been developed, from comparing tempo candidates using a list of likely factors [4], to probabilistic models [3], and multi-agent systems [6].

Many of these limitations are also shared by human listeners, making a quantitative evaluation of a model’s success difficult. Toiviainen and Eerola found that cultural background affects beat perception [15], and Goto and Muraoka acknowledge that multiple interpretations of beat and meter are sometimes possible [6]. Luckily, a quantitative analysis of success or failure is not required for evaluation of the present system: it is intended as a control for making music, and thus subjective evaluation should suffice.

### IV. IMPLEMENTATION

The system was implemented in Max/MSP [16], closely following the system described by Scheirer [7]. The implementation was designed to be easily extensible to allow efficient operation with varying computational resources in terms of speed and memory. Efforts were also made to build the system modularly, in such a way that the system could easily be distributed across a number of separate machines and communicate over a network. Although this aspect of the system was not tested, it may prove valuable in future testing.

The incoming audio to be analyzed is first processed to isolate onsets, then passed to a bank of tuned resonant filters. Examination of the outputs and contents of these filters yield possible values for the tempo and phase, which require further

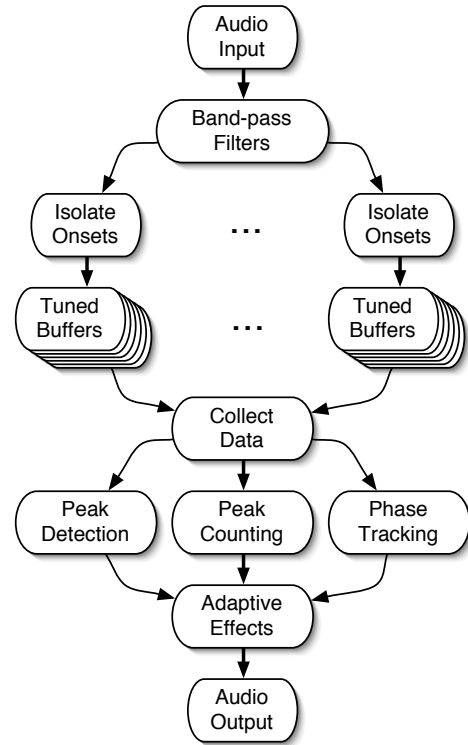


Fig. 1. Simplified diagram of the implemented system.

analysis before becoming useful in practical application. The following sections briefly describe the details of this process.

#### A. Signal Processing

After analog-to-digital conversion, the audio signal is split into spectral bands using 2nd-order IIR filters; the lowest filter is configured as a low-pass filter, the highest is configured as a high-pass filter, and the rest as band-pass filters. The output of each filter is processed to isolate audible attacks or onsets by half-wave rectifying the derivative of the signal’s amplitude envelope (see figure 2). According to Scheirer [7], using the amplitude envelopes of multiple frequency bands preserves information necessary for the perception of rhythm by human listeners (the *rhythmic percept*), whereas a summed amplitude envelope does not.

#### B. Tuned Resonant Filters

Each band of the processed signal is passed to a bank of IIR filters, each of which is tuned to resonate at a particular sub-harmonic frequency. The frequencies are spread logarithmically from 1Hz to 4Hz, corresponding to 60 to 240 beats per minute (BPM). Although Rowe [4] preferred to use a range of 40-200 BPM (corresponding to the range of a typical metronome), this is the range used by Scheirer [7]. Each buffer is contained in a Max/MSP sub-patch which generates its own unique ID number and is capable of reporting relevant information about the internal state of the buffer to the main patch. This information is sent once each iteration through

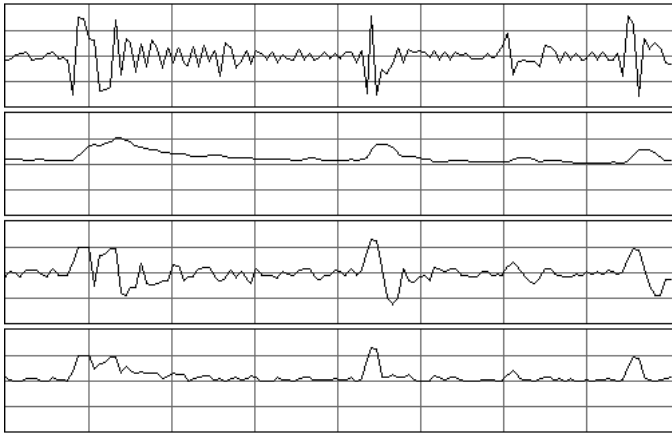


Fig. 2. Isolating onsets in the signal. From the top: 1) Input signal, 2) amplitude envelope, 3) derivative, 4) half-wave rectified

the buffer, meaning that each buffer updates at a different frequency.

Additionally, each bank of buffers has a unique ID, allowing the output of spectral bands to be routed and analyzed separately. The banks can be instantiated with a variable number of buffers, depending on the needs and processing resources of the user. The feedback coefficient is variable, allowing the “behavior” of the system to be tuned to consider past or present values more heavily. This allows the user to address the compromise between system responsiveness and stability cited earlier.

### C. Finding Tempo Candidates

The data reported by the individual buffers are collected into a single file and indexed by their unique IDs. The entries are sorted by peak magnitude and the top three results are chosen as tempo candidates, as these buffers will be those that are “phase-locked” with the input and thus resonating the most strongly. The consideration of multiple candidates for the tempo makes the system much more robust, in that it allows averaging (increasing the practical resolution of the system), comparison of harmonically related results (reducing the chance of harmonic errors), and the calculation of a value representing “confidence” in the output. It is possible that considering more candidates would result in a more stable output (Rowe [4] considered seven, for example), but in the interests of minimizing computational cost this was not explored. Following the implementation described by Scheirer [7], the outputs of buffers representing the same tempos are summed across spectral bands, although future implementations may consider the bands separately.

The frequency at which the file is sorted and the top buffers reported is also variable, but for testing was kept at the reporting frequency of the largest buffer (1Hz).

The value of the top peak is also used to control a compressor/expander patch that adjusts the gain at the buffer inputs. This maintains a normalized state in the resonating buffers (with the top peak equal to 1.0).

### D. Candidate Comparison

The top three results are compared in order to look for similarities and for harmonic relationships. Similar candidates, such as adjacent buffers, are used to generate an average value for the tempo, sometimes reinforcing the choice of a particular buffer, in other cases indicating a tempo that is not represented (fractional tempos, for example). Scheirer described errors made by his algorithm as being “typically due to [its] inability to understand beat relationships at various tempi.” [7] An attempt to avoid this problem is made by examining the internal state of the candidate buffers and counting amplitude peaks above a variable threshold; in order to avoid wrap-around issues, troughs are also counted and the lower value used. The represented tempo value for each of these buffers is multiplied by the number of peaks it contains to find a value which represents the real tempo of the signal. As a side-effect, this also enables the algorithm to find tempos that are much faster than the range of buffers chosen, since the signal will still cause harmonically-related buffers to resonate, but with more peaks-per-buffer.

The three tempo candidates are again compared in order to report an average tempo and to generate the “confidence” value. In order to allow for occasional errors or strange input effects in the buffers, only two of the buffers need to agree on the tempo (within a variable margin) in order to find a tempo with high confidence. The tempo reported is an average of the two closest tempos.

Although the variable threshold used to count peaks may be optionally automated to look for a particular range of peaks, this is usually not necessary with gain-normalized buffers.

### E. Phase Tracking

Three different approaches to phase tracking were implemented and tested:

- The first method requires the buffers to include the system time associated with a detected peak in the information reported at the end of each iteration. At the peak-sorting rate, these times are compared with the current system time, and the current tempo is used to predict the time of the next beat. A disadvantage of this method is that the next beat may have already occurred before the algorithm can predict it.
- The second method requires the buffers to report the offset into the buffer associated with the location of a detected peak. This information is immediately passed to the phase-tracking patch, which uses the current tempo to predict the time of the next beat.
- The third method uses the current tempo to tune a dedicated resonant filter. When the filter output passes a variable threshold, the phase is set to zero and a beat is assumed to have occurred. The threshold is set as a fraction of the previous iteration’s peak. Disadvantages of this method include the risk of multiple triggers per iteration, and the time-lag required for the buffer to begin resonating at a new frequency.

### F. Considerations for Real-time Operation

One of the major problems encountered by researchers implementing beat induction algorithms is the computational costs involved. Scheirer, for example, describes a system using 80 resonant filters for each spectral band, but later states that real-time operation was only possible using 50 filters per band and highly optimized coding [7]. Similarly, Goto and Muraoka required the use of a parallel computer to run their algorithms [17]. Although this problem may gradually disappear as faster computers become available, at the writing of this paper it is still very real.

In the interests of minimizing computational cost and enabling the use of the system in real-time, the signal is down-sampled to 1000Hz after initial processing but before passing to the banks of resonant filters. In addition to reducing the number of mathematical operations required, this permits large savings in memory, reducing the size of the buffers by a factor of 44.1. Another area in which costs are reduced is the band-pass filters used to split the signal into spectral bands. Scheirer describes the use of sixth-order elliptic filters [7]; in the present implementation simple second-order filters are used. Although this saves processing resources, higher-order filters would have allowed much cleaner cutoffs and less spectrum-sharing between filters.

Although both the number of spectral bands and the number of buffers per band were implemented so as to be easily variable, for testing purposes between one and three spectral bands were used, with eighty buffers per band. In order to cope with the loss of frequency resolution that would occur with a smaller number of buffers spread over the same range, a sub-patch was built containing seven *adaptive* buffers, which are automatically and dynamically tuned to center on the current consensus tempo. Three of the filters are tuned slightly higher than the center tempo, and three are tuned slightly below, covering a small but important range that allows frequencies to be detected that are not represented by a hard-coded resonant filter. As testing of the adaptive buffer system continues, it may become possible to reduce the number of buffers per spectral band considerably.

Some implementations simply use a smaller range of tempos [6], as this increases resolution without increasing computational costs; however as this system was intended for interaction with a live performer it was decided not to constrain the range further than originally implemented (60-240 BPM).

### V. INTEGRATION INTO THE IMS

The beat induction algorithm offers several interesting output values useful for mapping to control parameters of an interactive music system. In addition to the tempo itself, the change in tempo over time (the derivative of the tempo) and the confidence value - roughly representing to what extent the input contains strong, repeating pulses - are obvious choices. Mapping the phase as a control variable also leads to the possibility of mapping the degree of phase disruption - similar to the confidence value as an indication of "confusion" in the system - but at a much finer resolution.

	Simple	Complex
Clean	Impulse Train	Audio CD
Live	Digital Metronome	Live Input

Fig. 3. Input types used to test the system.

During testing of the system, the output side of the IMS mapping was restricted to controlling delay lines, with delay-times set by the induced tempo, and feedback affected by the confidence value. Although simple, this implementation allowed for the evaluation of beat induction as a musical control.

### VI. TESTING AND PERFORMANCE

The system was tested using four different types of input, each classified as to whether it is *clean* or *live*, and *simple* or *complex*. The input types can be seen in figure 3. The impulse train was generated on-system while the metronome sound was provided by an external, hardware metronome and recorded with a low-quality microphone. The audio CD consisted of recordings of a solo percussionist playing various frame drums, and the live input was provided by the same percussionist. The acoustical environment was somewhat controlled, being much noisier than a recording studio but considerably less noisy than some concert venues.

The system was tested running on two different hardware platforms: a dual 2GHz Apple G5 with 2GB of RAM, and a 1GHz Powerbook with 512MB of RAM. Audio was acquired using a Behringer ECM8000 microphone and a MOTU828 MkII firewire audio interface.

### VII. RESULTS

When finding the tempo, the system was found to be both highly accurate and responsive when tested with simple inputs - such as the impulse train or the external metronome - but less successful with complex inputs. Nevertheless, when the input was steady and rhythmic, the system did not have difficulty finding the correct tempo, only encountering difficulty when the input became rhythmically or metrically ambiguous. It was found that if the input contained constant sub-divisions of the beat - a style often played by the percussionist used for testing - the system would indicate the tatum rather than the beat. This was an unforeseen, but logical, consequence of the peak-counting algorithm described earlier, and could likely be remedied by comparing the heights of peaks in the peak-counting patch.

The phase-tracking portions of the algorithm were unfortunately less successful than the tempo induction. The output of the first method, in particular, was subjectively judged to be almost random when used to trigger a click sound at the time of a predicted beat. The second and third methods were more successful in predicting beats, but are still not suitable for use as a control for musical interaction.

For the purpose of testing the IMS, only tempo and confidence were used as controls, however the performer judged the system to be intuitive and interesting to interact with musically.

### VIII. DISCUSSION & FURTHER WORK

The implemented system was found to be promising as a control source for interactive music systems or for adaptive digital audio effects. This having been said, however, the use of delay lines in “interactive” percussion music is somewhat *cliché*, and the musical output of the IMS could have been much more interesting. A proposal was made to use the beat induction outputs to control the choice and playback of audio files, and design the system so that it takes a more active role in the interaction. This would also necessitate improving the phase-tracking implementation if the audio files were to be synchronized with a live performer.

Incremental steps must also be taken toward improving the accuracy and stability of the tempo output. If the changes involve increased computation, this may require further optimization of the processing and analysis code (perhaps writing Max/MSP externals) or a move to a distributed implementation in which several computers cooperate and communicate using OSC [18]. Testing the response of the system to other musical inputs will be important during this process, as hypothetically there are inputs which are completely rhythmically ambiguous, and thus cannot be tracked accurately.

The possibility of using onsets, rather than down-sampled amplitude envelopes, as input for the resonant filters should also be investigated. Scheirer [7] claimed that information essential to rhythmic percept is contained in these envelopes, but Toiviainen and Eerola found in their onset-based beat induction system that ignoring the accent structure (using equalized velocity) yielded more accurate results than when this information was considered [1].

Lastly, although it was the intent of this project to create an IMS rather than improve on the extant models for beat induction, a quantitative evaluation of the system should be performed. This would allow comparisons with other systems (such as Jehan’s object beat~ [19] for Max/MSP) and make it easier to mark the effect of future improvements. Goto and Muraoka proposed a standard method for measuring the accuracy of beat tracking results [9] that could be useful for comparisons with other systems.

### ACKNOWLEDGMENTS

The author would like to thank Ganesh Anandan for his time and patience and Kojiro Umezaki and Bruce Pennycook for helpful comments and suggestions during the implementation of this project.

### REFERENCES

- [1] P. Toiviainen and J. Snyder, “The role of accent periodicities in meter induction: A classification study,” in *Proceedings of the 8th International Conference on Music Perception and Cognition (ICMPC 2004)*, 2004.
- [2] M. Wright and D. Wessel, “An improvisation environment for generating rhythmic structures based on north indian “tal” patterns,” in *Proceedings of the International Computer Music Conference*, 1998, pp. 125–128.
- [3] A. P. Klapuri, “Musical meter estimation and music transcription,” in *Proceedings of the Cambridge Music Processing Colloquium*, 2003, pp. 40–45.
- [4] R. Rowe, *Machine Musicianship*. MIT Press, 2001.
- [5] P. Desain and H. Honing, “Computational models of beat induction: the rule-based approach,” *Journal of New Music Research*, vol. 28, no. 1, pp. 29–42, 1999.
- [6] M. Goto and Y. Muraoka, “Music understanding at the beat level - real-time beat tracking for audio signals,” in *Working Notes of the IJCAI-95 Workshop on Computational Auditory Scene Analysis*, 1995, pp. 68–75.
- [7] E. D. Scheirer, “Tempo and beat analysis of acoustic musical signals,” *Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998.
- [8] P. Toiviainen, “An interactive midi accompanist,” *Computer Music Journal*, vol. 22, no. 4, pp. 63–75, 1998.
- [9] M. Goto and Y. Muraoka, “Issues in evaluating beat tracking systems,” in *Working Notes of the IJCAI-97 Workshop on Issues in AI and Music - Evaluation and Assessment*, 1997, pp. 9–16.
- [10] V. Verfaillie and D. Arfib, “A-dafx: Adaptive digital audio effects,” in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-01)*, 2001, pp. 10–13.
- [11] T. Winkler, *Composing Interactive Music*. MIT Press, 1998.
- [12] P. Toiviainen and J. Snyder, “The time-course of pulse sensation: Dynamics of beat induction,” in *Proceedings of the 6th International Conference on Music Perception and Cognition (ICMPC 2000)*, 2000.
- [13] E. Large and J. Kolen, “Resonance and the perception of musical meter,” in *Musical Networks: Parallel Distributed Perception and Performance*, N. Griffith and P. Todd, Eds. MIT Press, 1999, pp. 279–312.
- [14] D. Rosenthal, M. Goto, and Y. Muraoka, “Rhythm tracking using multiple hypotheses,” in *Proceedings of the International Computer Music Conference*, 1994, pp. 85–88.
- [15] P. Toiviainen and T. Eerola, “Where is the beat? comparison of finnish and south-african listeners,” in *Proceedings of the 5th Triennial ESCOM Conference*, 2003, pp. 501–504.
- [16] (2004) Max/msp. [Online]. Available: <http://www.cycling74.com/products/maxmsp.html>
- [17] M. Goto and Y. Muraoka, “Beat tracking based on multiple-agent architecture - a real-time beat tracking system for audio signals,” in *Proceedings of the Second International Conference on Multiagent Systems*, 1996, pp. 103–110.
- [18] (2004) Open sound control home page. [Online]. Available: <http://www.cnmat.berkeley.edu/OpenSoundControl/>
- [19] (2005) Max/msp externals. [Online]. Available: <http://web.media.mit.edu/~tristan/maxmsp.html>