

The logo for the Centre for Digital Music features a stylized banner with a green-to-blue gradient and a grid pattern. The text "centre for digital music" is written in white, lowercase letters, with "digital" in a bold font.

centre for **digital** music

Context-dependent beat tracking of musical audio

MATTHEW E. P. DAVIES AND MARK D. PLUMBLEY

Technical Report C4DM-TR-06-02
5 April 2006

Context-dependent beat tracking of musical audio

Matthew E. P. Davies and Mark. D. Plumbley

Centre for Digital Music
Queen Mary, University of London
Mile End Road – London E1 4NS – United Kingdom
matthew.davies@elec.qmul.ac.uk

Abstract: We present a simple and efficient method for beat tracking of musical audio. With the aim of replicating the human ability of tapping in time to music, we formulate our approach using a two state model. The first state performs tempo induction and tracks tempo changes, while the second maintains contextual continuity within a single tempo hypothesis. Beat times are recovered by passing the output of an onset detection function through adaptively weighted comb filterbank matrices to separately identify the beat period and alignment. We evaluate our beat tracker both in terms of the accuracy of estimated beat locations and computational complexity. In a direct comparison with existing algorithms we demonstrate equivalent performance at significantly reduced computational cost.

Keywords: Beat tracking, rhythm analysis, musical meter, onset detection.

1 Introduction

The act of tapping one’s foot in time to music is an intuitive and often unconscious human response [1]. The computational equivalent to this behaviour, commonly referred to as *beat tracking*, has the aim of recovering a sequence of beat onset times from a musical input consistent with human foot taps. This task is related both to note onset detection [2], that of identifying the start points of musical events in an audio signal, and tempo induction [3], that of finding the underlying rate of a piece of music, and forms part of research into automatic rhythm description [4].

The need for robust beat tracking extends beyond the extraction of a sequence of beat onset times. In particular, within music information retrieval (MIR) based research, automatically extracted beats can provide a musically meaningful temporal segmentation for further analysis, such as chord estimation for harmonic description [5], long term structural segmentation of audio [6, 7], higher level metrical analysis [8] and rhythmic genre classification [9, 10].

If we consider a black-box beat tracker which receives a musical input and produces a sequence of beat times, we can suggest several desirable properties for such a system: (i) both audio and symbolic musical signals can be processed; (ii) no a-priori knowledge of the input is required (e.g. regarding genre, timbre or polyphony); (iii) perceptually accurate beat locations can be identified, efficiently and in real-time if necessary; and (iv) changes in tempo can be followed, such as the result of step or ramp changes and expressive timing variation. Despite a large body of research into beat tracking (e.g. [8, 11, 12, 13]), a system that meets all of these requirements does not currently exist.

1.1 Prior Art

Early approaches to beat tracking and rhythm analysis (e.g. [11, 14, 15]) processed symbolic data rather than audio signals, perhaps due to limited computational resources or a lack of sophistication in note onset detection. Recent advances in onset detection and audio feature extraction (reviewed in [2]) have enabled the use of audio signals for beat tracking. We now present a brief review of five well known approaches to beat tracking of audio signals. For more details see [4].

Dixon’s approach, *Beatroot* [1], processes a sequence of note onset times either extracted from an audio signal or from a symbolic representation, within a multi-agent system. Likely tempo hypotheses are derived from clustering inter-onset-intervals. These are used to form multiple beat agents (with varying tempi and phase) which compete based on how well each can predict beat locations. Dixon’s algorithm is designed to track beats in expressively performed music.

Goto’s approach [8] is also agent based. In addition to tracking the beats (at the 1/4 note level) analysis is extended to the 1/2 and whole note levels. Onset analysis is performed across seven parallel sub-bands, where spectral models are used to extract snare and bass drum events. Chord changes and pre-defined rhythmic pattern templates are used to infer the beats and higher level metrical structure. Goto’s system operates accurately and in real-time, provided the input signal has a steady tempo and a 4/4 time-signature (i.e. 4 beats per bar).

Hainsworth [16] extracts note onsets from sub-band analysis using two distinct models, one for finding transient events, the other designed to detect harmonic change, which are combined to give a single sequence of onsets. Particle filters are used within a statistical framework to track the beats, which are modelled as a quasi-periodic sequence driven by time-varying tempo and phase processes. The main limitation of this approach is its computational complexity.

In contrast to the previous approaches, which rely on the automatic extraction of note onsets, Scheirer [12] proposes the use of a psycho-acoustically motivated amplitude envelope signal to emphasise (but not extract) note onset locations, as the front-end to his beat tracker. Six octave-spaced sub-band envelope signals are passed through a parallel bank of tuned comb filter resonators representing tempo hypotheses over the range of 60–180 bpm. These phase-lock to the envelope signals across the sub-bands, to simultaneously infer the tempo and predict the phase of the beats within a real-time system.

Klapuri et al [13] expand upon Scheirer’s amplitude envelope and comb filter model. They adopt a more robust *registrar accent signal* across four parallel analysis bands as the input to their system and use comb filterbanks within a probabilistic framework to simultaneously track three metrical levels. These correspond to the *tatum*, the lowest or fastest metrical level, the *tactus*, the rate at which humans are most likely to tap, and the *measure*, which indicates the grouping of beats into bars. Analysis can be performed causally and non-causally, and is not restricted to any particular genre, tempo or time-signature. The algorithm has been tested over a large annotated database with results

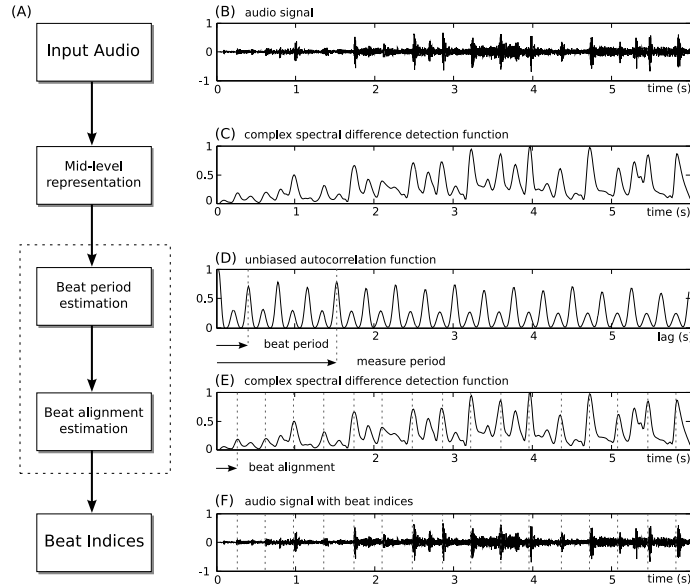


Figure 1: (A): Flow chart of typical beat tracking process. (B)-(F): Proposed model. (B) input audio signal, (C) onset detection function, (D) autocorrelation function with beat period, (E) detection function with beat alignment, (F) audio signal with extracted beats.

demonstrating superior performance to the Scheirer [12] and Dixon [1] approaches, while Hainsworth [16] reports comparable accuracy (a comparison against the Goto system [8] is not included).

A comprehensive comparison of audio beat trackers has not yet been undertaken. However, in a recent study of audio tempo induction algorithms [3], many of which inferred the tempo by first extracting the beats, the approach of Klapuri et al [13] was shown to be most accurate. We therefore identify this method as representative of the current state of the art for beat tracking of audio signals.

1.2 Proposed Model

A key component in audio based beat trackers appears to be the use of a multi-band decomposition prior to any rhythmic analysis. Scheirer [12] proposed that the use of sub-bands was essential; that a single analysis band was not sufficient to preserve the rhythmic content when using an amplitude envelope based input representation. We demonstrate that by using a more sophisticated input representation, derived across a wide analysis band, we can provide a single input to our beat tracker, without the need for sub-band analysis. We use the complex spectral difference onset detection function [17], described in section 2.1.

To extract the beats from this input representation, we could attempt to recover the beat period and phase simultaneously, as in [12, 13] or separate this task into two individual processes [1, 8]. The latter is computationally more efficient, as it decomposes a two-dimensional search into two, one-dimensional searches. We estimate the beat period by passing the autocorrelation function of the onset detection function through a shift invariant comb filterbank matrix, as described in section 2.2.1. We then use the beat period to recover the beat alignment by passing the onset detection function through a comb filter matrix described in section 2.2.2.

Klapuri et al [13] show that the robustness of their meter analysis model is due to the probabilistic modelling of the temporal evolution and interaction between each of the three metrical levels analysed. To increase the reliability of the beat output within our beat tracker, we adopt a simpler, heuristic approach by embedding context-dependent information directly into the beat period and alignment estimation processes. This is described in sections 2.3 and 2.4. An overview of our approach is shown in figure 1.

We evaluate our proposed beat tracking system in section 3, both in terms of the accuracy of generated beats and computational complexity. We demonstrate equivalent performance to the current state of the art, but at a significant reduction in computational cost, presenting discussions and conclusions in sections 4 and 5.

2 Approach

We recover beat locations from a continuous mid-level representation derived from a musical audio input, described in section 2.1. Our approach to beat tracking is comprised of two states. The first of these, which we refer to as the *General State*, performs beat period and beat alignment induction without any prior knowledge of the input (see section 2.2). We then describe the incorporation of context-dependent information regarding the tempo, time-signature and the locations of past beats, into the extraction of these parameters as part of the *Context-dependent State* (section 2.3). The higher level operation of the beat tracking system, which controls the interaction between the General and Context-dependent States is given in section 2.4.

2.1 Mid-level input representation

A common first stage in audio beat tracking algorithms is the transformation of the input audio signal into a more meaningful, compact representation from which to infer the beats. This can take the form of a sequence of automatically extracted note onsets [16], or a continuous function that emphasises them by exhibiting local maxima at likely onset locations [12]. These continuous signals, often referred to as *onset detection functions* [2], can more generally be considered *mid-level representations* that act as an intermediary signal between the input audio and the output beats. The process of extracting onsets from an onset detection function is not 100% effective, and can introduce undesirable errors both in terms of missed detections and false positives. We therefore use a continuous detection function as the input to our beat tracker.

To allow for the widest range of input signals, we would like the onset detection function to be sensitive both to percussive events (e.g. a drum hit) and softer tonal onsets (e.g. from a bowed violin). Referring to a recent comparative study of onset detection algorithms [2], we select the *complex spectral difference* onset detection function as that which best matches our aims. The ability of this detection function to emphasise note onsets relies on the extent to which the spectral properties of the signal at the onset of musical events are unpredictable. By working in the complex domain, note onsets are emphasised either as a result of significant change in energy in the magnitude spectrum, and/or a deviation from the expected phase values in the phase spectrum, caused by a change in pitch. The detection function $\Gamma(m)$, at frame m , is calculated by measuring the sum of the Euclidean distance between the observed spectrum at frame $S_k(m)$ and a prediction of the spectrum $\hat{S}_k(m)$, for all frequency bins k

$$\Gamma(m) = \sum_{k=1}^K |S_k(m) - \hat{S}_k(m)|^2. \quad (1)$$

To allow the beat tracker to operate independently of the sampling frequency of the input, the detection function is derived with a fixed time resolution. We set this resolution equal to that used in recent work in onset detection [2] at $t_{DF} = 11.6\text{ms}$ per detection function (DF) sample. A derivation of the detection function is given in Appendix 6, with further details in [17]. An example audio signal and detection function are shown in figures 1 (B) and (C).

To permit the the tempo and phase of the beats to vary across the length of the input, we partition the detection function $\Gamma(m)$ into overlapping analysis frames, $\Gamma_i(m)$

$$\Gamma_i(m) = \begin{cases} \Gamma(m) & m = 1 + (i - 1)B_f \dots, B_f + (i - 1)B_h \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where frames are of length $B_f = 512$ DF samples with hop size $B_h = B_f/4$ giving an overlap of 75%. Given the fixed time-resolution of the onset detection function, this equates to a 6 second analysis frame with a 1.5 second step increment.

2.2 General State

Within our proposed beat tracking model we use context-dependent information regarding the tempo and past beat locations to improve the reliability of the extracted beat locations. However, in order to incorporate this information into the beat tracking model, we must first extract it from the input signal. For this purpose we propose a *General*

State for beat tracking. Given an input onset detection function frame, the role of the General State is to infer the time between successive beats, the *beat period*, and the offset between the start of the frame and the locations of the beats, the *beat alignment*, without any prior knowledge of the input.

2.2.1 Beat Period Induction

Other approaches to beat tracking have simultaneously extracted the beat period and beat alignment from a mid-level representation using comb filter resonators [12, 13]. We adopt a more efficient approach by first identifying the beat period and then subsequently finding the beat alignment. To facilitate this separation, we discard all phase related information using an autocorrelation function of the current detection function frame. To achieve an autocorrelation function with clear, well defined peaks, we first process the detection function to emphasise the strongest and discard the least significant peaks. We calculate an adaptive, moving mean threshold $\bar{\Gamma}_i(m)$,

$$\bar{\Gamma}_i(m) = \text{mean}\{\Gamma_i(q)\} \quad m - \frac{Q}{2} \leq q \leq m + \frac{Q}{2} \quad (3)$$

where $Q = 16$ DF samples. We then subtract $\bar{\Gamma}_i(m)$ from the original detection function and half wave rectify the output, setting any negative valued elements to zero, to give a modified detection function frame $\tilde{\Gamma}_i(m)$

$$\tilde{\Gamma}_i(m) = \text{HWR}(\Gamma_i(m) - \bar{\Gamma}_i(m)) \quad (4)$$

where $\text{HWR}(x) = (x + |x|)/2$.

We calculate an autocorrelation function, $A(l)$, normalised to correct the bias occurring as a linear function of increasing lag:

$$A(l) = \sum_{m=1}^{B_f} \tilde{\Gamma}_i(m) \tilde{\Gamma}_i(m-l) / |l - B_f| \quad l = 1, \dots, B_f. \quad (5)$$

We can map a lag l (in DF samples) from the autocorrelation domain into a measure of tempo, in beats per minute, using the following relationship

$$\text{bpm} = \frac{60}{l \times t_{DF}} \quad (6)$$

where $t_{DF} = 11.6\text{ms}$ is the resolution of the detection function, specified in section 2.1 above.

Brown [18] uses autocorrelation to infer the meter of musical scores and symbolic performances observing periodicity at integer multiples of the beat level with a strong peak at the measure or bar level. We adopt a similar approach to finding the beat period, using comb filter type structures. To reflect periodicity at several metrical levels we derive a comb template $\lambda_\tau(l)$ as the sum of weighted delta functions at integer multiples of an underlying periodicity τ ,

$$\lambda_\tau(l) = \sum_{p=1}^4 \sum_{v=1-p}^{p-1} \frac{\delta(l - \tau p + v)}{2p - 1} \quad l = 1, \dots, B_f. \quad (7)$$

We define the longest observable periodicity $\tau_{\max} = B_h$ DF samples, which is equivalent to 40 bpm from (6). Equating τ_{\max} to the analysis frame hop size means we will derive at least one beat from every analysis frame. Given each autocorrelation function frame is of length $B_f = 512$ DF samples and $\tau_{\max} = B_h$, this permits the use four comb elements, $p = 1, \dots, 4$, within each comb template $\lambda_\tau(l)$. To account for the poor resolution of the autocorrelation function at short lags, each comb element has a width proportional to its relationship to τ set by $v = 1 - p, \dots, p - 1$, and a height normalised by its width $2p - 1$. We refer to this comb template as *metrically unbiased* because each comb element has equal influence over the extracted beat period. An example comb template and autocorrelation function are given in figure 2.

We combine a number of comb templates to span the range of beat period hypotheses ($1 \leq \tau \leq B_h$) into a *shift-invariant* comb filterbank $F_G(l, \tau)$. To reflect an approximate prior distribution of beat period hypotheses we use a weighting curve, $w_G(\tau)$, to indicate those beat periods which are most likely. Other systems have used a log-Gaussian distribution [13] or a resonance function [19] for this purpose. We require a function which strongly attenuates short lags while slowly decaying over longer lags and formulate $w_G(\tau)$ from the Rayleigh distribution function

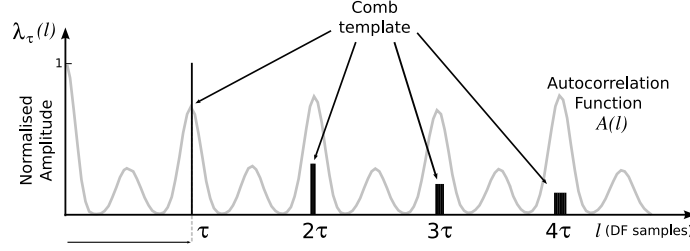


Figure 2: Autocorrelation function with metrically unbiased comb template.

$$w_{\mathbf{G}}(\tau) = \frac{\tau}{\beta^2} e^{\frac{-\tau^2}{2\beta^2}} \quad \tau = 1, \dots, B_h, \quad (8)$$

where the β parameter sets the strongest point of the weighting. An acceptable range for β was found to be between 40 and 50 DF samples, from which we select $\beta = 43$, equivalent to 120 bpm from (6). Informal tests indicated the resonance curve [19] and log-Gaussian distribution [13] to be less effective than the Rayleigh weighting within our beat tracking framework.

The shift-invariant comb filterbank $F_{\mathbf{G}}(l, \tau)$ can be conveniently represented in matrix form, where each column of the matrix represents a comb template $\lambda_{\tau}(l)$ with a fundamental periodicity τ equal to the column number. We apply the weighting curve multiplying each $\lambda_{\tau}(l)$ with the appropriate value from $w_{\mathbf{G}}(\tau)$, to give

$$F_{\mathbf{G}}(l, \tau) = w_{\mathbf{G}}(\tau) \lambda_{\tau}(l). \quad (9)$$

We take the product of the autocorrelation function $A(l)$ with $F_{\mathbf{G}}(l, \tau)$ to give an output $y_{\mathbf{G}}(\tau)$ as a function of τ

$$y_{\mathbf{G}}(\tau) = \sum_{l=1}^{B_f} A(l) F_{\mathbf{G}}(l, \tau) \quad (10)$$

from which we identify the beat period $\tau_{\mathbf{G}}$ as the index of the maximum value

$$\tau_{\mathbf{G}} = \arg \max_{\tau} (y_{\mathbf{G}}(\tau)). \quad (11)$$

The comb filterbank $F_{\mathbf{G}}(l, \tau)$ is shown in figure 3(A), with an example output signal $y_{\mathbf{G}}(\tau)$ in figure 3(B). We can observe multiple peaks in $y_{\mathbf{G}}(\tau)$, which correspond to integer and fractional multiples of $\tau_{\mathbf{G}}$. The General State relies on the lag consistent with the tempo of the input being the strongest peak within $y_{\mathbf{G}}(\tau)$ for all analysis frames.

2.2.2 Beat Alignment Induction

We define beats in terms of two parameters: the beat period $\tau_{\mathbf{G}}$, and the beat alignment $\alpha_{\mathbf{G}}$. We now address finding the locations of the beats within each frame given a known beat period. In our beat tracking system, we can perform either causal or non-causal analysis. To distinguish between causal and non-causal analysis we use x^* to indicate parameter x is extracted causally.

For non-causal analysis, each frame contains B_f DF samples of future data. We define the beat alignment $\alpha_{\mathbf{G}}$ to represent the offset from the start of the current frame to the location of the first beat within this frame, as shown in figure 1(E). For causal analysis, each frame contains the past B_f DF samples of data. Without access to any future data we predict the locations of future beats based solely on analysis of past data. We define the causal beat alignment $\alpha_{\mathbf{G}}^*$ as the offset from the end of current frame back to the last beat occurring in this frame. This represents the best location from which future beats can be predicted.

To extract the beat alignment we adopt a similar matrix based approach to finding the beat period. For beat period extraction each column of the matrix $F_{\mathbf{G}}(l, \tau)$ represents a comb template with a different periodicity. Once this periodicity is known, we can formulate a comb filter matrix $H_{\mathbf{G}}(m, \alpha)$ containing all possible shifts of the beat period to infer the beat alignment from the current detection function frame $\Gamma_i(m)$.

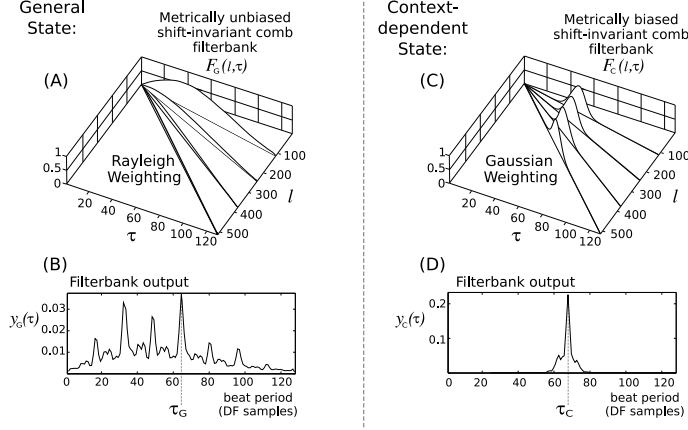


Figure 3: Shift-invariant comb filterbanks for the General State (A) and the Context-dependent State (C), with respective output signals: (B) and (D). The index of the maximum value in the output is selected as the beat period for the current analysis frame.

We construct an alignment comb template $\psi_\alpha(m)$ as a sequence of impulses at beat period intervals with offset α . Since we cannot guarantee that the tempo will be constant across the entire frame, we multiply each comb template with a linearly decreasing weighting $\nu(m)$ to give most emphasis to the first few comb elements

$$\psi_\alpha(m) = \sum_{k=1}^{\lfloor \frac{B_f}{\tau_G} \rfloor} \nu(m) \delta(m - k\tau_G + \alpha) \quad m = 1, \dots, B_f \quad (12)$$

where k ranges over the number of elements in each comb template and $\nu(m)$ is defined as

$$\nu(m) = \frac{B_f - m}{B_f} \quad m = 1, \dots, B_f. \quad (13)$$

To form the matrix $H_G(m, \alpha)$ each column represents a comb template $\psi_\alpha(m)$ with an incremental offset α ,

$$H_G(m, \alpha) = \psi_\alpha(m) \quad 1 \leq \alpha \leq \tau_G \quad (14)$$

For non-causal analysis we find the product of the current detection function frame $\Gamma_i(m)$ with $H_G(m, \alpha)$, to give an output function of α ,

$$z_G(\alpha) = \sum_{m=1}^{B_f} \Gamma_i(m) H_G(m, \alpha) \quad (15)$$

with α_G extracted as the index of the maximum value of $z_G(\alpha)$

$$\alpha_G = \arg \max_{\alpha} (z_G(\alpha)). \quad (16)$$

We then place beats $\gamma_{i,b}$ for the i^{th} frame at beat period intervals from α_G such that

$$\gamma_{i,b} = (t_i + \alpha_G) + (b - 1)\tau_G \quad b = 1, \dots, B \quad (17)$$

where t_i indicates the time at the start of the i^{th} detection function frame, $\Gamma_i(m)$. To allow for variations in the beat structure, any beats beyond one step increment from t_i will be assigned from analysis of the subsequent detection function frame $\Gamma_{i+1}(m)$, we therefore constrain $\gamma_{i,B} < t_i + B_h$.

For the causal version we modify (15), replacing the current detection function frame $\Gamma_i(m)$ with a time-reversed version $\Gamma_i^*(m) = \Gamma_i(-m)$ leaving $H_G(m, \alpha)$ unchanged. This now gives strongest emphasis to the comb elements

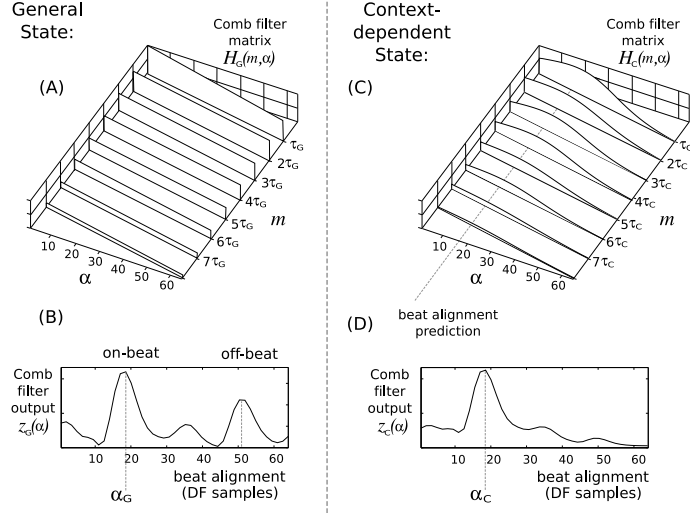


Figure 4: Comb filter matrices for the General State (A) and Context-dependent State (B), with respective example output signals (C) and (D). The index of the maximum value in the output is selected as the phase of the beats for the current analysis frame

at the end of each frame, which are the most recent DF samples to have occurred. We find $z_G^*(\alpha)$, the causal version of $z_G(\alpha)$ as

$$z_G^*(\alpha) = \sum_{m=1}^{B_f} \Gamma_i^*(m) H_G(m, \alpha) \quad (18)$$

extracting α_G^* as the index of the maximum value

$$\alpha_G^* = \arg \max_{\alpha} (z_G^*(\alpha)). \quad (19)$$

We modify (17) to predict causal beats $\gamma_{i,b}^*$ up to one step increment (B_h DF samples) *beyond* the end of the current frame,

$$\gamma_{i,b}^* = (t_i^* + \alpha_G^*) + b\tau_G \quad b = 1, \dots, B \quad (20)$$

where $\gamma_{i,B}^* < t_i^* + B_h$ and t_i^* represents the time at the end of the frame i .

The comb filter matrix $H_G(m, \alpha)$ is shown in figure 4(A) with a corresponding output signal $z_G(\alpha)$ in figure 4(B). We can observe two peaks in the output signal, which are labelled as the *on-beat*, the result of tapping in phase, and the *off-beat*, tapping in anti-phase, i.e. at an offset of $\frac{\tau_G}{2}$ DF samples from α_G ($\frac{\tau_G}{2} \approx 30$ here). We assume that the on-beat will be the strongest peak in $z_G(\alpha)$ for all analysis frames. To maintain a consistent beat output it is important not to erroneously switch between the on and the off-beat.

2.3 Context-dependent State

The General State for beat tracking operates in a memoryless fashion, extracting the beat period and beat alignment through a process of repeated induction. The principal limitation of this method is that no effort is made to enforce continuity. For a given analysis frame τ_G and α_G can vary independently of any past values. This can cause two common beat tracking errors [20]: (i) the switching of metrical levels, i.e. tapping at one rate and then changing to half or double this rate without a tempo change having occurred; and (ii) switching between the on and the off-beat, or in other words changing from tapping in phase to tapping beats in anti-phase.

To prevent these undesirable effects we propose a *Context-dependent State* for beat tracking. It operates in a similar way to the General State in that it separately extracts the beat period and the beat alignment. We now incorporate prior

knowledge of the tempo of the input and feedback past locations of beats, based on the simple assumption that, if beats are arriving at regular beat period intervals then our focus is no longer on inducing beat locations but in maintaining continuity [21].

2.3.1 Beat Period

There are two principle distinctions between the process of beat period extraction in the Context-dependent State and the General State both of which relate to the construction of the context-dependent comb filterbank $F_{\mathbf{C}}(l, \tau)$. All other aspects involved in the extraction of the beat period from the autocorrelation function $A(l)$ remain constant. First we classify the time-signature of the input, and use this to incorporate metrical bias into each comb template, setting the number of elements equal to the numerator of the time-signature. Secondly, we replace the Rayleigh weighting curve, which gave an approximate prior distribution of beat periods, with a tighter Gaussian weighting localised around a known beat period. This prevents the extraction of beat periods at related metrical levels.

To extract the time-signature T of the input we use a similar approach to the method of Brown [18]. This method extracts a strong peak in the autocorrelation function at the measure level periodicity and relates this to the beat period. We follow Gouyon and Herrera [22] and classify the input as having a *binary* meter $T = 4$, or a *ternary* meter $T = 3$. We do not attempt to infer odd time-signatures (e.g. 5/4, 7/8, 9/8) leaving this as a topic for further work.

Given a value for the beat period either from analysis in the General State, $\tau_{\mathbf{G}}$, or from a notated value, we classify T by comparing the energy at odd and even multiples of the beat period within the autocorrelation function based on the following condition

$$A(2\tau_{\mathbf{G}}) + A(4\tau_{\mathbf{G}}) > A(3\tau_{\mathbf{G}}) + A(6\tau_{\mathbf{G}}). \quad (21)$$

If condition (21) holds, i.e. we find stronger periodicity at even multiples than odd multiples, we infer a binary meter setting $T = 4$, otherwise we presume a ternary meter and set $T = 3$. This allows us to reformulate the comb filterbank matrix $F_{\mathbf{C}}(l, \tau)$. We set the number of elements in each comb template $\lambda_{\tau}(l)$ equal to T . To make the comb template metrically biased (in contrast to 7) we remove the $(2p - 1)$ normalisation from (7) to give most emphasis to the measure level periodicity,

$$\lambda_{\tau}(l) = \sum_{p=1}^T \sum_{v=1-p}^{p-1} \delta(l - \tau p + v) \quad l = 1, \dots, B_f. \quad (22)$$

Prior knowledge of the beat period allows us to limit the range of likely beat period hypotheses replacing the Rayleigh weighting with a context-dependent Gaussian function centred on $\tau_{\mathbf{G}}$

$$w_{\mathbf{C}}(\tau) = e^{-\frac{(\tau - \tau_{\mathbf{G}})^2}{2\sigma_w^2}} \quad \tau = 1, \dots, B_h. \quad (23)$$

The width of the weighting, defined by σ_w , constrains the range of likely beat periods. If σ_w is too tight, no local variation in the beat period will be observable. If it is too wide, the weighting will not be able to prevent the beat period switching between metrical levels. To give an approximate measure of local variation in beat periods, we analysed the distribution of inter-annotation-intervals, Δ_j , (normalised to have zero mean) across the annotated test database we use to evaluate our beat tracker (in section 3). We set $\sigma_w = 4$ DF samples equating it to the standard deviation of the inter-annotation-interval distribution. Informal tests revealed σ_w could vary by a factor of 2 without altering the overall performance of the beat tracker.

The shift-invariant comb filterbank $F_{\mathbf{C}}(l, \tau)$ for the Context-dependent State can now be formed by setting each comb template $\lambda_{\tau}(l)$ as the τ^{th} column of the matrix and weighting each template using the Gaussian weighting $w_{\mathbf{C}}(\tau)$

$$F_{\mathbf{C}}(l, \tau) = w_{\mathbf{C}}(\tau)\lambda_{\tau}(l). \quad (24)$$

As in the General State we take the product of the autocorrelation function $A(l)$ with $F_{\mathbf{C}}(l, \tau)$

$$y_{\mathbf{C}}(\tau) = \sum_{l=1}^{B_f} A(l)F_{\mathbf{C}}(l, \tau) \quad (25)$$

extracting τ_C as the index of the maximum value of $y_C(\tau)$

$$\tau_C = \arg \max_{\tau} (y_C(\tau)). \quad (26)$$

We may now compare the comb filterbanks for the General and Context-dependent States by examining figures 3 (A) and (C). The effect of the Gaussian weighting is evident by comparing the respective example comb output signals from each state in figures 3 (B) and (D). For the General State, there is a strong ambiguity between two beat period hypotheses, with either one a possible interpretation of the input. However in the Context-dependent State, only a small range around the stronger of the two remains unattenuated by the Gaussian weighting function. The restriction in likely beat periods prevents the possibility of switching between metrical levels that could occur using the General State alone.

2.3.2 Beat Alignment

The extraction of the beat alignment in the Context-dependent State does not differ greatly from the General State. Again we rely on a previously identified beat period τ_C and use a matrix based approach to find the alignment α_C from the current detection function frame $\Gamma_i(m)$. The role of the Context-dependent State is to maintain a consistent beat output. For beat period extraction, we limit the range of likely beat periods to be close to an expected value. When this is the case, we can reasonably assume that beats will occur approximately at beat period intervals. For a given frame $\Gamma_i(m)$ we can use this to predict a likely value for α_C given the beat output from the previous frame $\Gamma_{i-1}(m)$. As in the General State we can perform both non-causal and causal analysis.

In the non-causal case, the beat alignment represents the offset from the start of the current detection function frame $\Gamma_i(m)$ to the location of the first beat within that frame. We can predict α_C as approximately τ_C DF samples beyond the last beat $\gamma_{i-1,B}$ from the previous detection function frame $\Gamma_{i-1}(m)$,

$$\alpha_C \approx \gamma_{i-1,B} + \tau_C. \quad (27)$$

Equating (27) would cause the beats to occur precisely at beat period intervals. To allow for the location of the beats to vary we create an alignment weighting function $\rho_C(\alpha)$ centred on the most likely α_C :

$$\rho_C(\alpha) = e^{-\frac{(\alpha - (\gamma_{i-1,B} + \tau_C))^2}{2\sigma_\rho^2}} \quad \alpha = 1, \dots, \tau_C. \quad (28)$$

As with the Gaussian weighting for beat period extraction (23), σ_ρ defines the width of the weighting. To prevent the possibility of extracting the alignment corresponding to the off-beat, which should occur approximately $\frac{\tau_C}{2}$ samples away from the on-beat (see figure 4(B)), we use a beat period dependent width setting $\sigma_\rho = \frac{\tau_C}{4}$ DF samples.

We formulate a context-dependent comb filter matrix $H_C(m, \alpha)$ where each column of the matrix represents an alignment comb, $\psi_\alpha(m)$ at an offset α ,

$$\psi_\alpha(m) = \sum_{k=1}^{\lfloor \frac{B_f}{\tau_C} \rfloor} \nu(m) \delta(m - k\tau_C + \alpha) \quad m = 1, \dots, B_f \quad (29)$$

with $\nu(m)$ from (13) as the linear weighting function, to give

$$H_C(m, \alpha) = \rho_C(\alpha) \psi_\alpha(m). \quad (30)$$

We then take the product of $\Gamma_i(m)$ with $H_C(m, \alpha)$ to give an output function of α

$$z_C(\alpha) = \sum_{m=1}^{B_f} \Gamma_i(m) H_C(m, \alpha) \quad (31)$$

with α_C found as the index of the maximum value of $z_C(\alpha)$

$$\alpha_C = \arg \max_{\alpha} (z_C(\alpha)). \quad (32)$$

Beats $\gamma_{i,b}$ can then be placed at intervals of $\tau_{\mathbf{C}}$ up to one step increment into the current analysis frame using

$$\gamma_{i,b} = (t_i + \alpha_{\mathbf{C}}) + (b - 1)\tau_{\mathbf{C}} \quad b = 1, \dots, B \quad (33)$$

with $\gamma_{i,B} < t_i + B_h$ and t_i as the start of the current frame.

For causal analysis, where each frame represents the past B_f DF samples of data, the beat alignment $\alpha_{\mathbf{C}}^*$ indicates the offset back from the end of the current frame. If the tempo of the input is constant then $\alpha_{\mathbf{C}}^*$ will equal the location of the last beat $\gamma_{i-1,B}^*$ within the previous analysis frame $\Gamma_{i-1}^*(m)$. Allowing for timing variations we reformulate (28) as

$$\rho_{\mathbf{C}}^*(\alpha) = e^{-\frac{-(\alpha - \gamma_{i-1,B}^*)^2}{2(\sigma_{\rho}^*)^2}} \quad \alpha = 1, \dots, \tau_{\mathbf{C}}. \quad (34)$$

When predicting the locations of future beats, we cannot directly observe expressive timing changes as in the non-causal approach. We therefore restrict the likely beat alignment values more so than in (28) by setting $\sigma_{\rho}^* = \frac{\tau_{\mathbf{C}}}{8}$ DF samples.

We create the causal comb matrix $H_{\mathbf{C}}^*(m, \alpha)$ from (30) now using $\rho_{\mathbf{C}}^*(\alpha)$ to weight the alignment comb templates $\psi_{\alpha}(m)$

$$H_{\mathbf{C}}^*(m, \alpha) = \rho_{\mathbf{C}}^*(\alpha)\psi_{\alpha}(m). \quad (35)$$

We find the product of the time-reversed detection function frame $\Gamma_i^*(m)$ with $H_{\mathbf{C}}^*(m, \alpha)$,

$$z_{\mathbf{C}}^*(\alpha) = \sum_{m=1}^{B_f} \Gamma_i^*(m)H_{\mathbf{C}}^*(m, \alpha) \quad (36)$$

extracting $\alpha_{\mathbf{C}}^*$ as the index of the maximum value of $z_{\mathbf{C}}^*(\alpha)$

$$\alpha_{\mathbf{C}}^* = \arg \max_{\alpha} (z_{\mathbf{C}}^*(\alpha)). \quad (37)$$

We can now predict the locations of future beats,

$$\gamma_{i,b}^* = (t_i^* - \alpha_{\mathbf{C}}^*) + b\tau_{\mathbf{C}} \quad b = 1, \dots, B \quad (38)$$

where t_i^* represents the end of the current analysis frame and beats are constrained $\gamma_{i,B}^* < t_i^* + B_h$.

Figure 4 shows both the General and Context-dependent States comb filter matrices (A) and (C), with corresponding output signals (B) and (D). The alignment weighting in the Context-dependent State attenuates the energy in the output around the location of the off-beat leaving just a single peak to indicate the beat alignment. This prevents the possibility of the beats switching from the on-beat to the off-beat, maintaining a consistent beat output.

2.4 Two State Model

Although the General State and Context-dependent States can work in isolation to provide the locations of the beats, neither one is ideally suited to do so. Without any means to enforce continuity, the General State is too susceptible to switching metrical levels and changing from the on-beat to the off-beat. The Context-dependent State, while explicitly designed to prevent these errors by limiting the range of likely beat periods and alignments, is unable to react to any tempo changes that might occur. To exploit the particular properties of each state we combine them into a *Two State Model*. We use the General State to infer an initial beat period and detect tempo changes, and the Context-dependent State to maintain continuity within a given beat period hypothesis. The operation of the Two State Model is shown in figure 5.

We define a state variable q to indicate which state is currently active, and therefore which is used to derive the beat locations. We label the General State S_1 and the Context-dependent State S_2 . Without any prior knowledge of the input we initially set $q = S_1$, with beat locations assigned using the General State.

The activation of the first Context-dependent State S_2 is the result of the observation of consistent beat periods across three consecutive frames, which satisfy

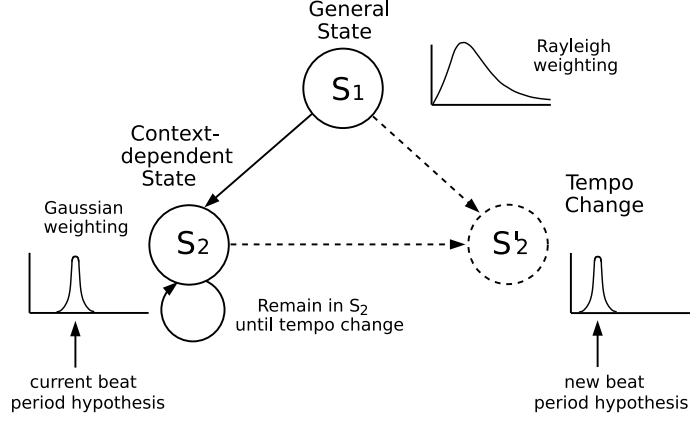


Figure 5: Two state switching model: Analysis begins in General State S_1 . The observation of a consistent beat period generates a Context-dependent state, S_2 , which is replaced by S_2' if a tempo change is observed.

$$|2\tau_G(i) - \tau_G(i-1) - \tau_G(i-2)| < \eta_1 \quad (39)$$

where $\tau_G(i)$ is the beat period from the i^{th} frame and η_1 is the threshold. We define a consistent beat period to be one that can be reliably observed within the range of the Gaussian weighting (23) for beat period extraction. We therefore set $\eta_1 = \sigma_w$ from (23) at 4 DF samples. If condition (39) is met, $q = S_2$, with beats assigned from the Context-dependent State.

When S_2 is active, the General State continues to operate, acting in a supervisor role, to verify whether the Context-dependent State is appropriate to the current properties of the signal. At each frame, we extract τ_G and compare it to τ_C . If the values are close, then we maintain the operation of the Context-dependent State. For the Context-dependent State to be considered a poor match to the input, we require two conditions to be met: (i) the absolute difference between τ_G and τ_C is greater than η_2 , the threshold for tempo changes,

$$|\tau_G(i) - \tau_C(i)| > \eta_2 \quad (40)$$

where η_2 is set to be outside of the range of the Gaussian weighting in (23), e.g. $\eta_2 = 2\eta_1$; and (ii) the new beat period hypothesis τ_C is consistently observed, and therefore meets (39).

If S_2 is found to be no longer a valid match to the rhythmic properties of the signal, then it is discarded, to be replaced by a new Context-dependent State S_2' indicative of the new beat period hypothesis observed by the General State. Before setting $q = S_2'$ and using the new Context-dependent State to find the beat locations, we let the General State resume control of the beat tracker for a single analysis frame, setting $q = S_1$, to provide a new reference point α_G for the locations of the beats.

For subsequent frames we now activate the new Context-dependent State setting $q = S_2'$ and use it to maintain continuity in the beat output, until a new consistent beat period is observed. Operating in this manner, the Two State Model can explicitly model tempo discontinuities while smoothing out odd beat errors during each consistent beat period hypothesis.

3 Evaluation

We pursue an objective approach to evaluate our beat tracking model. We use an existing annotated test database [16] containing 222 musical audio files, with the following genre breakdown: Rock/pop (68), Dance (40), Jazz (40), Classical (30), Folk (22), and Choral (22). Each file is between 30 and 60 seconds in length, mono and sampled 44.1kHz with 16 bit resolution. The files were annotated by a trained musician, by recording taps in time to the audio files. The annotations were then synthesised over each example and where necessary, manually corrected to be perceptually *in time* according to the judgement of the annotator. For more details see [16].

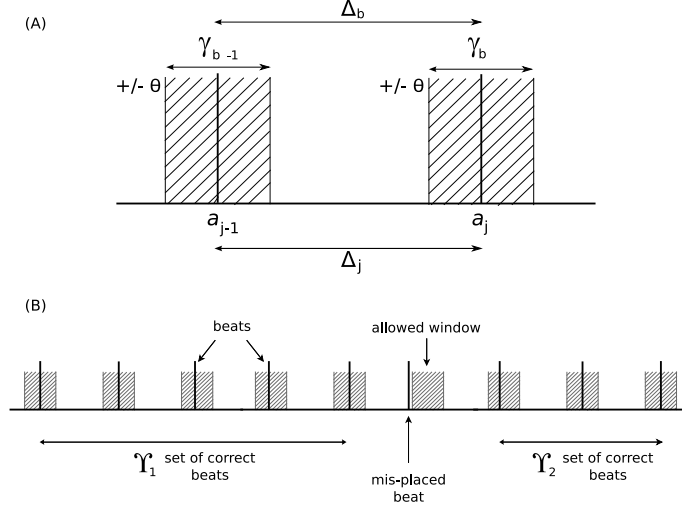


Figure 6: (A) Continuity based beat tracking evaluation metric. Acceptance of beat, γ_b , depends on its proximity to annotation, a_j , and the localisation of the previous beat. θ defines the acceptance window around each annotation. (B) Overview of evaluation metric over several beats.

We compared the performance of our proposed model to two published beat tracking systems: (i) Simon Dixon’s algorithm *Beatroot* [1] and ii) the more recent approach of Klapuri et al [13], which we will refer to as KEA. To permit an informal comparison between computational and human approaches to beat tracking we also evaluate an example human tapping performance (that of the first author). For each file in the database, beats were recorded as real-time keyboard taps using a published annotation tool [23]. In contrast to the ground truth annotated data, no manual correction of the tapped beats was permitted.

3.1 Beat Accuracy

To quantify the performance of our beat tracker we use an objective measure to describe beat accuracy. We follow the recent approaches of [24, 16, 13] by placing an emphasis on the generation of a continuously correct beat output. The accuracy of given beat γ_b is defined by its proximity to annotation a_j , and, that of the preceding beat γ_{b-1} to annotation a_{j-1} . The dependence on the previous beat enforces the continuity constraint. We also require the current inter-beat-interval $\Delta_b = \gamma_b - \gamma_{b-1}$ is consistent with the current inter-annotation-interval $\Delta_j = a_j - a_{j-1}$, creating three conditions,

1. $a_j - \theta\Delta_j < \gamma_b < a_j + \theta\Delta_j$
2. $a_{j-1} - \theta\Delta_{j-1} < \gamma_{b-1} < a_{j-1} + \theta\Delta_{j-1}$
3. $(1 - \theta)\Delta_j < \Delta_b < (1 + \theta)\Delta_j$

where in each condition θ defines the width of the acceptance window around each annotation. Klapuri et al [13], evaluate their beat tracking using similar continuity based criteria, and set $\theta = 0.175$, which equates to acceptance window of $\pm 17.5\%$ around each annotation. The conditions for beat accuracy are shown figure 6(A).

For a sequence of beats $\{\gamma_b\}$ we can find the number of correct beats in each continuously correct segment Υ_k (see figure 6(B)) that meet the criteria described above,

$$\hat{\Upsilon}_k = \max(\Upsilon_k) \quad k = 1, \dots, K \quad (41)$$

where K is the number of segments found. We extract $\hat{\Upsilon}_k$ as the largest number of correct beats and therefore the longest continuously correct segment. To give a figure of overall accuracy for a given file, we find the ratio of $\hat{\Upsilon}_k$, to the number of annotated beats \mathfrak{A} in that file

$$\text{accuracy}_{\text{cont}} = \frac{\hat{\Upsilon}_k}{\mathfrak{A}} \times 100\%. \quad (42)$$

The principle weakness of this approach, illustrated in figure 6(B), is that a single mis-placed beat can cause the accuracy to drop significantly. In the worst case, where the erroneous beat is in the middle of the file, the accuracy can fall from 100% to 50%. To allow for continuity breaks, we also find the ratio of the total number of correct beats to the number of annotations

$$\text{accuracy}_{\text{total}} = \frac{\sum_{k=1}^K \Upsilon_k}{\mathfrak{A}} \times 100\%. \quad (43)$$

A further consideration is the choice of metrical level at which the tapping occurs, as humans often tap in time to music at different rates [25]. If we insist on the correct metrical level, then beats tapped at twice or half the rate of the annotations would be assigned an accuracy of 0% using (42) and (43), which may be considered to be too harsh. To allow for metrical ambiguity we therefore also calculate a measure of continuously correct tracking, and total number of correct beats for cases where tapping occurs at twice or half the annotated rate. To summarise, we present four measures of beat accuracy:

- Correct Metrical Level, continuity required (CML cont);
- Correct Metrical Level, continuity not required (CML total);
- Allowed Metrical Levels, continuity required (AML cont);
- Allowed Metrical Levels, continuity not required (AML total).

Results are shown in Table 1 first for causal analysis: our approach, labelled DP (C); the causal model of Klapuri et al [13], labelled KEA (C); and the Human tapping performance; and then for non-causal analysis: DP (NC); the non-causal version of KEA (NC); and the Dixon’s (non-causal) Beatroot method [1]. We find equivalent performance for our non-causal approach compared to that of KEA, both of which are more accurate than Dixon across each of the four measures of beat accuracy. We also observe the expected trend of poorer performance for the causal implementations of DP and KEA in comparison to the non-causal versions. Somewhat surprisingly the Human tapping performance is weaker than both DP and KEA when continuity is required in the beat output.

Examining each column in Table 1, we find weakest performance when continuity is required at the correct metrical level, and strongest for the total number of correct beats at allowed metrical levels. This is only a partial ordering however, as demonstrated by the results for the Human performance, where the accuracy for CML (total) is greater than that of AML (cont.). From this observation we find the Human tapper to be most adversely affected by the continuity constraint. A second interesting feature of the Human performance relates to relationship between results at CML and AML. In contrast to the computational approaches, which display a significant improvement when allowing for metrical ambiguity, the Human tapper only shows a moderate improvement, suggesting a more accurate inference of the annotated metrical level.

We can further examine the beat tracking accuracy as a function of the allowance window θ . Results shown in figure 7 compare the non-causal approaches to the Human data under each of the four evaluation measures. As $\theta > 20\%$ the Human performance exceeds the algorithmic approaches under all conditions. In figure 7 it is clear exactly how close the performance of our proposed model DP is to KEA. The accuracy of the two approaches does not differ by more than 3% for all θ . There are genre-related differences shown in Table 2, where our non-causal approach is better able to infer the beats for Jazz examples, with KEA more accurate for Folk and Classical music. For Choral music, we find all algorithms having very low performance, with accuracy $< 10\%$. Only the Human tapper demonstrates any ability to find the beats.

3.2 Beat Localisation

An alternate way to visualise the performance of a beat tracker is to measure the *normalised beat error* – to show how close the beats are to the annotations. We find the normalised beat error, $\epsilon_{n,j}$ for file n , as difference between each annotated beat a_j and the nearest generated beat γ_b , from all beats γ , divided by the inter-annotation-interval Δ_j

Beat Tracker	CML cont. (%)	CML total (%)	AML cont. (%)	AML total (%)
DP (C)	46.7	53.9	58.4	69.3
KEA (C)	52.3	59.7	65.2	76.1
Human (C)	44.5	78.9	48.5	85.4
DP (NC)	54.8	61.2	68.1	78.9
KEA (NC)	55.7	62.4	70.0	80.0
Dixon (NC)	25.1	30.9	46.2	59.6

Table 1: Comparison of beat tracking performance: Causal analysis (C) – DP, KEA and Human data; Non-causal analysis (NC) – DP, KEA and Dixon.

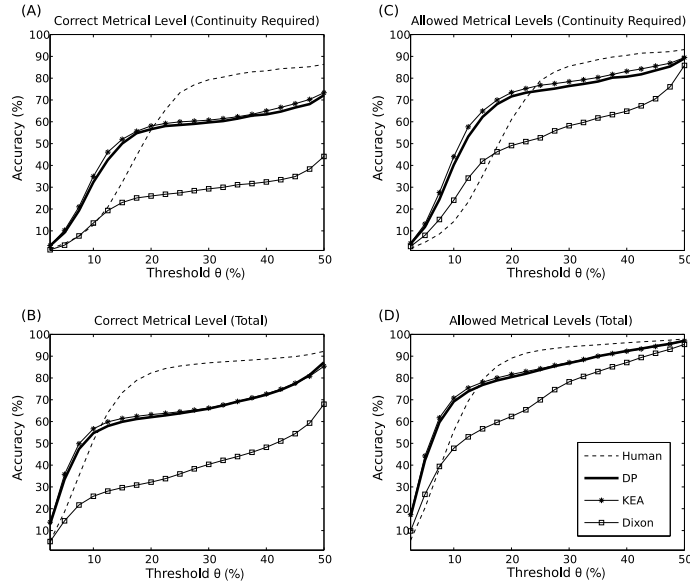


Figure 7: Comparison of Davies (DP), Klapuri (KEA), Dixon and Human beat tracking performance as a function of the acceptance window θ . Correct metrical level with continuity required (A). Allowed metrical levels with continuity required (B). Correct metrical level without continuity (C) and Allowed metrical levels without continuity requirement (D).

$$\epsilon_{n,j} = \min_{\gamma_b} \frac{(\gamma - a_j)}{\Delta_j}. \quad (44)$$

At most any one beat can be 50% ahead or behind the annotated value. This represents tapping in anti-phase to the annotated beats. By finding the error for all annotations in the database, approximately 20,000 beats, we can create histograms to show the distribution of beat error. These are shown in figure 8 for each of the non-causal approaches and the Human tapping data.

Each of the three algorithmic approaches (figures 8(B–D)) exhibit a strong peak close to an error of 0, indicating beats close to the annotations, and then two further peaks each around $\pm 50\%$ of a beat away from the annotations, corresponding to tapping on the *off-beat*. We see an almost identical shape to the distributions of our approach and KEA. For Dixon’s algorithm the distribution is flatter indicating a greater proportion of erroneous beats, which are neither close to 0, nor at $\pm 50\%$. The histogram of Human data has a different shape. It is close to Gaussian, with very few beats in anti-phase at $\pm 50\%$ beats. This suggests greater consistency in finding the correct metrical level. However the main peak of the distribution is wider than in the computational approaches indicating less precise beat localisation. We should note that the histogram of Human performance has a non-zero mean. This demonstrates that the Human subject, on average, tapped ‘ahead’ of the beat over the test database.

Beat Tracker	Dance (%)	Rock (%)	Jazz (%)	Folk (%)	Classical (%)	Choral (%)
DP (C)	79.1	68.1	36.1	24.9	17.5	2.8
KEA (C)	73.8	72.4	37.2	47.6	37.0	4.0
Human (C)	51.9	49.9	37.9	57.5	32.5	29.5
DP (NC)	76.8	77.9	51.6	34.5	26.9	7.2
KEA (NC)	76.3	77.9	39.1	52.2	40.2	4.0
Dixon (NC)	31.0	34.0	23.5	12.9	10.2	5.4

Table 2: Breakdown of performance across musical genre: Causal analysis (C) – DP, KEA and Human data; Non-causal analysis (NC) – DP, KEA and Dixon. Results are measured at the correct metrical level with continuity required.

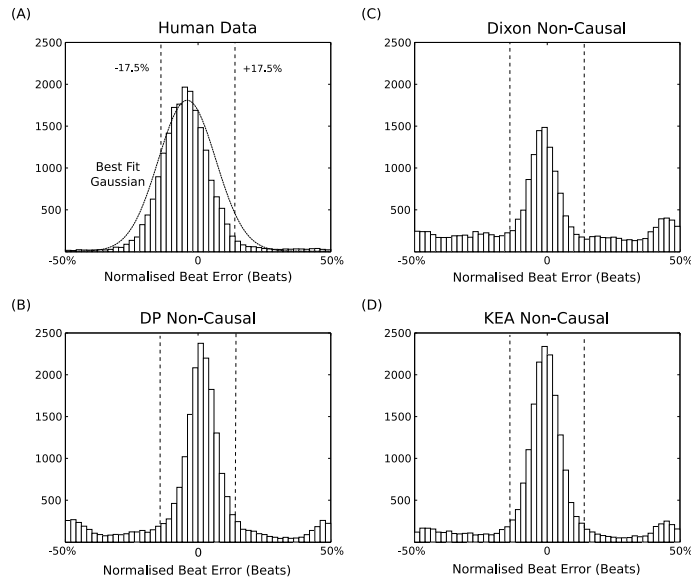


Figure 8: Histogram of normalised beat error for (A) Human, (B) DP, (C) KEA and (D) Dixon beat tracking algorithms. Vertical dotted lines indicate $\pm 17.5\%$, the acceptance window θ .

3.3 Computation Time

Having shown similar performance between our proposed model and the Klapuri et al [13] approach, we now examine the computational complexity of these two algorithms. We have implemented our approach to mirror that of the KEA system, as generously supplied to us by Anssi Klapuri. Both systems are implemented in Matlab, but use pre-compiled functions to implement the generation of mid-level representation from the input audio signal. We measure the computation time ratio as the compute time (in seconds) for a file n , normalised by its length (in seconds), for all files N in the test database

$$\text{computation time ratio} = \frac{1}{N} \sum_{n=1}^N \frac{\text{compute time (n)}}{\text{length (n)}}. \quad (45)$$

Tests were run on a 3GHz Linux machine running Matlab version 7.0. Results are given in figure 9 showing the time taken to generate the mid-level representation input and the remaining time to extract the beats from this representation. For our model the computational complexity is significantly lower than for the KEA system. We also note that for our approach the time taken in beat tracking is much less than the time required to generate the onset detection function mid-level representation. The converse is true for the KEA system, which spends a much more significant period of time recovering the beats.

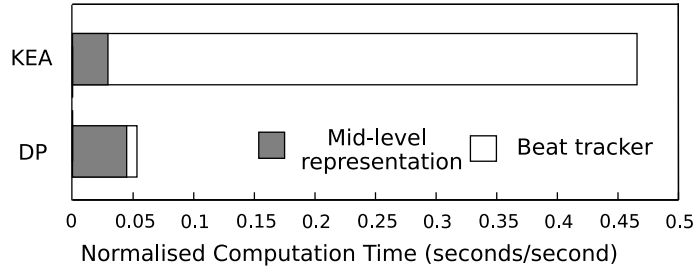


Figure 9: Proportion of normalised computation time taken in beat tracking compared to generation of mid-level representation for KEA and DP algorithms

4 Discussion

We have shown our non-causal beat tracking algorithm to be of equivalent accuracy to the KEA system [13] when evaluated objectively using a continuity based metric. Further, we have illustrated that this level of accuracy can be obtained at a considerable reduction in computational cost. By comparing the architecture of the two approaches, we can identify three factors in which the KEA system is more complex than our beat tracker: i) Klapuri et al [13] use a multi-band approach to feature extraction, performing beat analysis in each sub-band; ii) for each analysis frame the beat period and alignment are extracted simultaneously through parallel resonator filterbanks; and iii) KEA incorporate higher level knowledge using probabilistic analysis of multiple metrical levels. By making a single pass across one mid-level representation, separating the extraction of beat period and phase into two processes (therefore reducing a two-dimensional search into two, one-dimensional searches), and embedding high level knowledge directly into the extraction of the beats, we are able to present a more efficient approach to beat tracking, which is equally robust.

A notable weakness of our model in comparison to the KEA system is the extent to which accuracy is reduced for causal analysis. Tracking beats causally leads to a 10% reduction in accuracy compared to non-causal results, which for KEA is only a 3% reduction. It appears that our static prediction of future beats at precise beat period intervals (20) is not as effective as the higher level knowledge used within the KEA system. The simultaneous analysis of multiple metrical levels increases the reliability of beat predictions, which are directed towards the start of the next measure. In our approach, we currently use no knowledge of the phase of higher metrical levels, extracting only the measure level periodicity. This is most telling in cases which exhibit expressive timing, where inaccurate beat predictions break the continuity requirement, causing a reduction in overall accuracy.

We see from the results in Table 1 that our causal approach is most competitive when continuity is not required and the metrical level is not specified (the AML total column). The human performance is even more adversely affected by the requirement of a continuously correct output. We must question the importance of strict continuity when evaluating a beat tracking system. A single mis-placed beat, in the context of a sequence of otherwise accurate beats, is a far less disturbing error than the accuracy calculated with the continuity requirement would suggest.

Given the original aim of replicating the human ability of tapping in time to music we have illustrated that, at tight thresholds, algorithmic approaches can be shown to be superior to an example human tapping performance. The algorithmic beat trackers perform most accurately in cases where the beat structure is typically very rigid, for example in the Rock and Dance genres, where the human tapping fails to give a consistent correct beat output when the allowance threshold is $\theta < 20\%$ of a beat. Conversely when the allowance threshold θ is increased we see the accuracy of the Human performance exceeds all of the algorithmic approaches.

Three areas in which the human beat tracker can outperform the computational approaches are: i) in selecting the correct metrical level for tapping, as demonstrated by the small difference between results for CML and AML in Table 1; (ii) in selecting the correct phase, tapping on the on-beat rather than the off-beat, (see figure 8); and (iii) tapping in time to music without any clear rhythmic structure, shown by the Choral examples in the test database (see Table 2). These remain open challenges for automatic beat tracking, which may be solved by providing algorithms with higher level knowledge, including harmonic data, instrumentation and genre. We plan to investigate these factors as part of our future work into rhythmic analysis of audio signals, in addition to the development of a real-time implementation of our approach toward an automatic musical accompaniment system.

5 Conclusions

We have presented an efficient audio based beat tracking algorithm able to produce equivalent results to the current state of the art when evaluated using continuity based criteria. The particular advantages of our model lie in its simplicity and computational efficiency, where our algorithm performs competitively at a significant reduction of the computational cost of the best performing published algorithm. In our future work we intend to further investigate methods for evaluating beat tracking algorithms and extend our proposed model to operate in real-time for use within an interactive musical system.

6 Appendix: Onset detection function

We use the complex spectral difference onset detection function as described in [17]. It is derived from analysis of overlapping short term audio frames. To maintain a fixed time-resolution for the detection function that is independent of the sampling frequency of the input, we make the hop size h (in equation (46)) dependent on the sampling frequency: $h = f_s \times t_{DF}$, where we set t_{DF} to represent the desired resolution (in seconds) of each detection function sample. We then use frames of length $N = 2h$ creating a 50% overlap of the analysis at each iteration across the length of the audio input signal. For audio sampled at $f_s = 44.1kHz$ this creates a frame size of $N = 1024$ with a hop size of $h = 512$ audio samples.

We take the windowed Short Term Fourier Transform (STFT) of an input audio signal $s(n)$ using an N length Hanning window $w(m)$ and find the k^{th} bin of the short term spectral frame $S_k(m)$ as

$$S_k(m) = \sum_{n=-\infty}^{\infty} s(n)w(mh - n)e^{-j2\pi nk/N} \quad (46)$$

with a magnitude

$$R_k(m) = |S_k(m)| \quad (47)$$

where $S_k(m) \in \mathbb{C}$ and $R_k(m) \in \mathbb{R}$.

By applying the assumption that during steady state regions of the signal, i.e. not at a note onset, the magnitude spectrum should remain approximately constant, we can make a prediction of the magnitude spectrum \hat{R}_k at frame m , given the magnitude of the previous frame:

$$\hat{R}_k(m) = R_k(m - 1). \quad (48)$$

In addition we apply an assumption about the properties of the phase spectrum, that during steady state regions of the signal, the phase velocity at the k^{th} bin should ideally be constant.

$$\tilde{\phi}_k(m) - \tilde{\phi}_k(m - 1) \approx \tilde{\phi}_k(m - 1) - \tilde{\phi}_k(m - 2). \quad (49)$$

We adopt a short-hand notation for the left hand side of equation (49)

$$\Delta\tilde{\phi}_k(m) = \tilde{\phi}_k(m) - \tilde{\phi}_k(m - 1). \quad (50)$$

By substituting (50) into (49) and rearranging terms, we can make prediction about the phase of the k^{th} bin for frame $\tilde{\phi}_k(m)$ given the observations of the previous two frames:

$$\hat{\phi}_k = \text{princarg}[\tilde{\phi}_k(m - 1) + \Delta\tilde{\phi}_k(m - 1)] \quad (51)$$

where `princarg` unwraps the phase value, mapping it into the range $[-\pi, \pi]$.

The predictions of the magnitude spectrum $\hat{R}_k(m)$ and the phase spectrum $\hat{\phi}_k(m)$ can be represented in polar form to give a spectral prediction $\hat{S}_k(m)$ in the complex domain

$$\hat{S}_k(m) = \hat{R}_k(m)e^{j\hat{\phi}_k(m)} \quad (52)$$

which we compare to the observed complex spectrum:

$$S_k(m) = R_k(m)e^{j\phi_k(m)}. \quad (53)$$

To derive the complex spectral difference detection function $\Gamma(m)$ at frame m , we find the sum of the Euclidean distance between the predicted and observed spectra for all k bins

$$\Gamma(m) = \sum_{k=1}^K |S_k(m) - \hat{S}_k(m)|^2. \quad (54)$$

Acknowledgements

Special thanks to Anssi Klapuri for making his source code for the KEA algorithm available, Stephen Hainsworth for providing the ground truth beat annotations and Nicolas Ch  try for technical assistance.

Matthew Davies is supported by a college studentship from Queen Mary University of London. This research has been partially funded by the EU-FP6-IST-507142 project SIMAC, and by EPSRC grants GR/S75802/01 and GR/S82213/01.

References

- [1] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, pp. 39–58, 2001, Source code at <http://www.ofai.at/~simon.dixon/beatroot/>.
- [2] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, part 2, pp. 1035–1047, 2005.
- [3] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech and Language Processing*, 2006, in press.
- [4] F. Gouyon and S. Dixon, "A review of automatic rhythm description systems," *Computer Music Journal*, vol. 29, no. 1, pp. 34–54, 2005.
- [5] J. P. Bello and J. Pickens, "A robust mid-level representation for harmonic content in music signals," in *Proceedings of 6th International Symposium on Music Information Retrieval*, London, United Kingdom, 2005, pp. 304 – 311.
- [6] M. A. Bartsch and G. H. Wakefield, "Audio thumbnailing of popular music using chroma-based representations," *IEEE Transactions on Multimedia*, vol. 7, no. 1, pp. 96–104, 2005.
- [7] M. Levy, M. Sandler, and M. Casey, "Extraction of high level musical structure from audio data and its application to thumbnail generation," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2006, to appear.
- [8] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Journal of New Music Research*, vol. 30, pp. 159–171, 2001.
- [9] S. Dixon, F. Gouyon, and G. Widmer, "Towards characterisation of music via rhythmic patterns," in *Proceedings of 5th International Conference on Music Information Retrieval*, Barcelona, Spain, 2004, pp. 509–517.
- [10] C. Uhle and C. Dittmar, "Drum pattern based genre classification of popular music," in *Proceedings of 25th International AES Conference on Semantic Audio*, London, UK, 2004, pp. 189–195.
- [11] P. E. Allen and R. B. Dannenberg, "Tracking musical beats in real time," in *Proceedings of International Computer Music Conference (ICMC)*, Hong Kong, 1990, pp. 140–143.

- [12] E. D. Scheirer, “Tempo and beat analysis of acoustic musical signals,” *Journal of Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998, Source code at <http://sound.media.mit.edu/~eds/beat/tapping.tar.gz>.
- [13] A. P. Klapuri, A. Eronen, and J. Astola, “Analysis of the meter of acoustic musical signals,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 342–355, 2006.
- [14] D. F. Rosenthal, *Machine Rhythm: Computer Emulation of Human Rhythm Perception*, Ph.D. thesis, Massachusetts Institute of Technology, 1992.
- [15] E. W. Large, “Beat tracking with a nonlinear oscillator,” in *Working Notes of the IJCAI-95 Workshop on Issues in AI and Music*, 1995, pp. 24–31.
- [16] S. Hainsworth, *Techniques for the Automated Analysis of Musical Audio*, Ph.D. thesis, Department of Engineering, Cambridge University, 2004.
- [17] J. P. Bello, C. Duxbury, M. E. Davies, and M. B. Sandler, “On the use of phase and energy for musical onset detection in the complex domain,” *IEEE Signal Processing Letters*, vol. 11, no. 6, pp. 553–556, 2004.
- [18] J. C. Brown, “Determination of the meter of musical scores by autocorrelation,” *Journal of the Acoustical Society of America*, vol. 94, no. 4, pp. 1953–1957, October 1993.
- [19] L. van Noorden and D. Moelants, “Resonance in the perception of musical pulse,” *Journal of New Music Research*, vol. 28, no. 1, pp. 43–66, 1999.
- [20] M. E. P. Davies and M. D. Plumbley, “Causal tempo tracking of audio,” in *Proceedings of 5th International Symposium on Music Information Retrieval*, Barcelona, Spain, 2004, pp. 164–169.
- [21] M. Minsky, “Music, mind and meaning,” *Computer Music Journal*, vol. 5, no. 3, pp. 28–44, 1981.
- [22] F. Gouyon and P. Herrera, “Determination of the meter of musical audio signals: Seeking recurrences in beat segment descriptors,” in *Proceedings of 114th Convention of the Audio Engineering Society*, 2003, Pre-print no. 5811.
- [23] F. Gouyon, N. Wack, and S. Dixon, “An open source tool for semi-automatic rhythmic annotation,” in *Proceedings of the 7th International Conference on Digital Audio Effects (DAFx)*, Naples, Italy, 2004, pp. 193–196.
- [24] 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.
- [25] D. Moelants and M. McKinney, “Tempo perception and musical content: what makes a piece fast, slow or temporally ambiguous?,” in *Proceedings of the 8th International Conference on Music Perception and Cognition*, Evanston, IL, USA, 2004, pp. 558–562.