Joint Beat & Tatum Tracking from Music Signals

Jarno Seppänen Antti Eronen Jarmo Hiipakka
Nokia Research Center
P.O.Box 407, FI-00045 NOKIA GROUP, Finland
{jarno.seppanen, antti.eronen, jarmo.hiipakka}@nokia.com

Abstract

This paper presents a method for extracting two key metrical properties, the beat and the tatum, from acoustic signals of popular music. The method is computationally very efficient while performing comparably to earlier methods. High efficiency is achieved through multirate accent analysis, discrete cosine transform periodicity analysis, and phase estimation by adaptive comb filtering. During analysis, the music signals are first represented in terms of accentuation on four frequency subbands, and then the accent signals are transformed into periodicity domain. Beat and tatum periods and phases are estimated in a probabilistic setting, incorporating primitive musicological knowledge of beat–tatum relations, the prior distributions, and the temporal continuities of beats and tatums. In an evaluation with 192 songs, the beat tracking accuracy of the proposed method was found comparable to the state of the art. Complexity evaluation showed that the computational cost is less than 1% of earlier methods. The authors have written a real-time implementation of the method for the S60 smartphone platform.

Keywords: Beat tracking, music meter estimation, rhythm analysis.

1. Introduction

Recent years have brought significant advances in the field of automatic music signal analysis, and music meter estimation is no exception. In general, the music meter contains a nested grouping of pulses called *metrical levels*, where pulses on higher levels are subsets of the lower level pulses; the most salient level is known as the *beat*, and the lowest level is termed the *tatum* [1, p. 21].

Metrical analysis of music signals has many applications ranging from browsing and visualization to classification and recommendation of music. The state of the art has advanced high in performance, but the computational requirements have also remained restrictively high. The proposed method significantly improves computational efficiency while maintaining satisfactory performance.

The technical approaches for meter estimation are various, including *e.g.* autocorrelation based methods [6], inter-onset interval histogramming [5], or banks of comb filter resonators [4], possibly followed by a probabilistic model [3]. See [2] for a review on rhythm analysis systems.

2. Algorithm Description

The algorithm overview is presented in Fig. 1: the input is audio signals of polyphonic music, and the output consists of the times of beats and tatums. The implementation of the beat and tatum tracker has been done in C++ programming language in the S60 smartphone platform. The algorithm design is causal and the implementation works in real time.

The operation of the system can be described in six stages (see Fig. 1):

1. Resampling stage,
2. Accent filter bank stage,
3. Buffering stage,
4. Periodicity estimation stage,
5. Period estimation stage, and
6. Phase estimation stage.

First, the signal is resampled to a fixed sample rate, to support arbitrary input sample rates. Second, the accent filter bank transforms the acoustic signal of music into a form that is suitable for beat and tatum analysis. In this stage, subband accent signals are generated, which constitute an estimate of the perceived accentuation on each subband. The accent filter bank stage significantly reduces the amount of data.

Then, the accent signals are accumulated into four-second frames. Periodicity estimation looks for repeating accents on each subband. The subband periodicities are then combined, and summary periodicity is computed.

Next, the most likely beat and tatum periods are estimated from each periodicity frame. This uses a probabilistic formulation of primitive musicological knowledge, including the relation, the prior distribution, and the temporal continuity of beats and tatums. Finally, the beat phase is found and beat and tatum times are positioned. The accent signal is filtered with a pair of comb filters, which adapt to different beat period estimates.

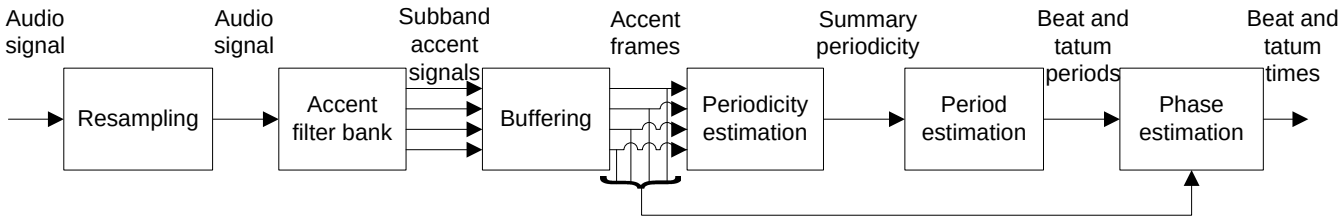

Figure 1. Beat and tatum analyzer.

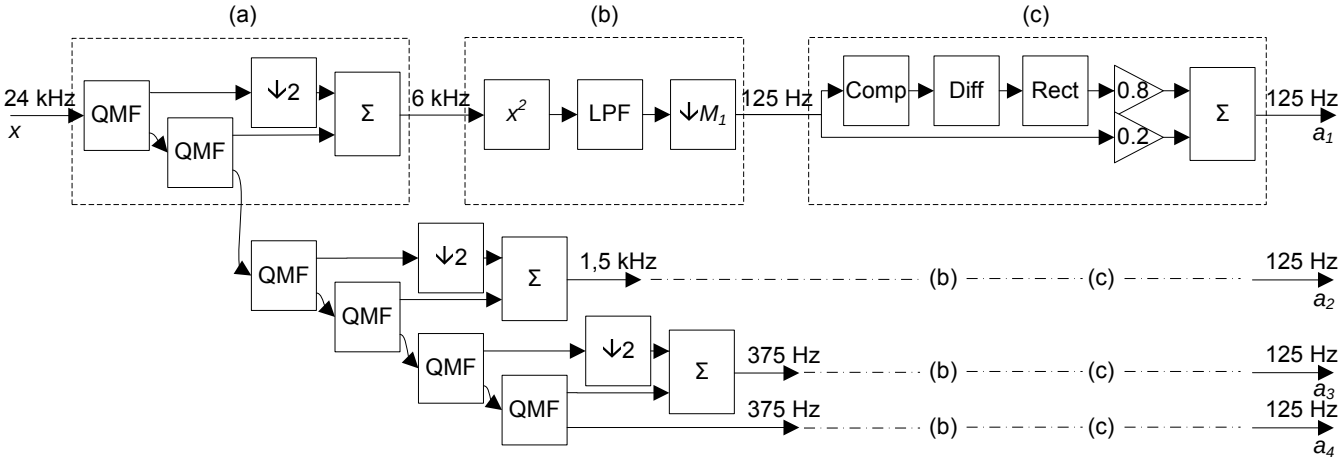

Figure 2. Accent filter bank overview. (a) The audio signal is first divided into subbands, then (b) power estimates on each subband are calculated, and (c) accent computation is performed on the subband power signals.

2.1. Resampling

Before any audio analysis takes place, the signal is converted to a 24 kHz sample rate. This is required because the filter bank uses fixed frequency regions. The resampling can be done with a relatively low-quality algorithm, linear interpolation, because high fidelity is not required for successful beat and tatum analysis.

2.2. Accent Filter Bank

Figure 2 presents an overview of the accent filter bank. The incoming audio signal $x[n]$ is (a) first divided into subband audio signals, and (b) a power estimate signal is calculated for each band separately. Last, (c) an accent signal is computed for each subband.

The filter bank divides the acoustic signal into seven frequency bands by means of six cascaded decimating quadrature mirror filters (QMF). The QMF subband signals are combined pairwise into three two-octave subband signals, as shown in Fig. 2(a). When combining two consecutive branches, the signal from the higher branch is decimated without filtering. However, the error caused by the aliasing produced in this operation is negligible for the proposed method. The sampling rate decreases by four between successive bands due to the two QMF analysis stages and the extra decimation step. As a result, the frequency bands are located at 0–190 Hz, 190–750 Hz, 750–3000 Hz, and 3–12 kHz, when the filter bank input is at 24 kHz.

There is a very efficient structure that can be used to implement the downsampling QMF analysis with just two allpass filters, an addition, and a subtraction. This structure is depicted in Fig. 5.2-5 in [7, p. 203]. The allpass filters for this application can be first-order filters, because only modest separation is required between bands.

The subband power computation is shown Fig. 2(b). The audio signal is squared, low-pass filtered (LPF), and decimated by subband specific factor M_i to get the subband power signal. The low-pass filter is a digital filter having 10 Hz cutoff frequency. The subband decimation ratios $M_i = \{48, 12, 3, 3\}$ have been chosen so that the power signal sample rate is 125 Hz on all subbands.

The subband accent signal computation in Fig. 2(c) is modelled according to Klapuri *et al.* [3, p. 344–345]. In the process, the power signal first is mapped with a nonlinear level compression function labeled *Comp* in Fig. 2(c),

$$f(x) = \begin{cases} 5.213 \ln(1 + 10\sqrt{x}), & x > 0.0001 \\ 5.213 \ln 1.1 & \text{otherwise.} \end{cases} \quad (1)$$

Following compression, the first-order difference signal is computed (*Diff*) and half-wave rectified (*Rect*). In accordance with Eq. (3) in [3], the rectified signal is summed to the power signal after constant weighting, see Fig. 2(c). The high computational efficiency of the proposed method lies mostly in the accent filter bank design. In addition to efficiency, the resulting accent signals are comparable to those of Klapuri *et al.*, see *e.g.* Fig. 3 in [3].

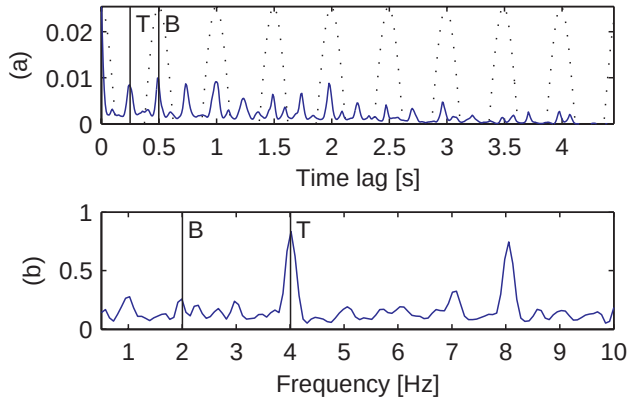

Figure 3. (a) Normalized autocorrelation and (b) summary periodicity, with beat (B) and tatum (T) periods shown.

2.3. Buffering

The buffering stage implements a ring buffer which accumulates the signal into fixed-length frames. The incoming signal is split into consecutive accent signal frames of a fixed length $N = 512$ (4.1 seconds). The value of N can be modified to choose a different performance–latency tradeoff.

2.4. Accent Periodicity Estimation

The accent signals are analyzed for intrinsic repetitions. Here, periodicity is defined as the combined strength of accents that repeat with a given period. For all subband accent signals, a joint summary periodicity vector is computed.

Autocorrelation $\rho[\ell] = \sum_{n=0}^{N-1} a[n]a[n-\ell]$, $0 \leq \ell \leq N-1$, is first computed from each N-length subband accent frame $a[n]$. The accent signal reaches peak values whenever there are high accents in the music and remains low otherwise. Computing autocorrelation from an impulsive accent signal is comparable to computing the inter-onset interval (IOI) histogram as described by Seppänen [5], with additional robustness due to not having to discretize the accent signal into onsets.

The accent frame power $\rho[0]$ is stored for later weighting of subband periodicities. Offset and scale variations are eliminated from autocorrelation frames by normalization,

$$\bar{\rho}[\ell] = \frac{\rho[\ell] - \min_n \rho[n]}{\sum_{n=0}^{N-1} \rho[n] - N \min_n \rho[n]}. \qquad (2)$$

See Fig. 3(a) for an example normalized autocorrelation frame. The figure shows also the correct beat period B, 0.5 seconds, and tatum period T, 0.25 seconds, as vertical lines.

Next, accent periodicity is estimated by means of the N-point discrete cosine transform (DCT)

$$R[k] = c_k \sum_{n=0}^{N-1} \bar{\rho}[n] \cos \frac{\pi(2n+1)k}{2N} \qquad (3)$$

$$c_0 = \sqrt{1/N} \qquad (4)$$

$$c_k = \sqrt{2/N}, \qquad 1 \leq k \leq N-1. \qquad (5)$$

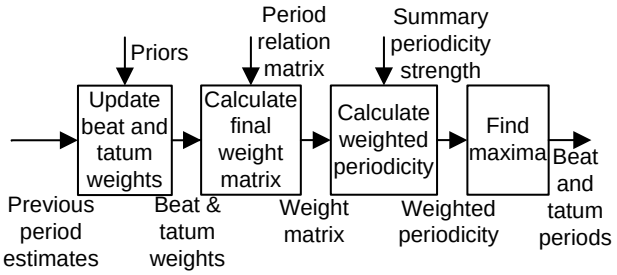

Figure 4. The period estimator.

Similarly to an IOI histogram [5], accent peaks with a period p cause high responses in the autocorrelation function at lags $\ell = 0$, $\ell = p$ (nearest peaks), $\ell = 2p$ (second-nearest peaks), $\ell = 3p$ (third-nearest peaks), and so on. Such response is exploited in DCT-based periodicity estimation, which matches the autocorrelation response with zero-phase cosine functions; see dashed lines in Fig. 3(a).

Only a specific periodicity window, $0.1 \text{ s} \leq p \leq 2 \text{ s}$, is utilized from the DCT vector $R[k]$. This window specifies the range of beat and tatum periods for estimation. The subband periodicities $R_i[k]$ are combined into an M-point summary periodicity vector, $M = 128$,

$$S[k] = \sum_{i=1}^{4} \rho_i[0]^\gamma \widetilde{R}_i[k] \quad 0 \leq k \leq M-1, \qquad (6)$$

where $\widetilde{R}_i[k]$ has interpolated values of $R_i[k]$ from 0.5 Hz to 10 Hz, and the parameter $\gamma = 1.2$ controls weighting. Figure 3(b) shows an example summary periodicity vector.

2.5. Beat and Tatum Period Estimation

The period estimation stage finds the most likely beat period $\hat{\tau}_n^B$ and tatum period $\hat{\tau}_n^A$ for the current frame at time n based on the observed periodicity $S[k]$ and primitive musicological knowledge. Likelihood functions are used for modeling primitive musicological knowledge as proposed by Klapuri *et al.* in [3, p. 344–345], although the actual calculations of the model are different. An overview of the period estimator are depicted in Fig. 4.

First, weights $f^i(\tau_n^i)$ for the different beat and tatum period candidates are calculated as a product of prior distributions $p^i(\tau^i)$ and "continuity functions":

$$f_C^i\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right) = \frac{1}{\sigma_1\sqrt{2\pi}} \exp\left[-\frac{1}{2\sigma_1^2}\left(\ln \frac{\tau_n^i}{\tau_{n-1}^i}\right)^2\right], \qquad (7)$$

as defined in Eq. (21) in [3, p. 348]. Here, $i = A$ denotes the tatum and $i = B$ denotes the beat. The value $\sigma_1 = 0.63$ is used. The continuity function describes the tendency that the periods are slowly varying, thus taking care of "tying" the successive period estimates together. τ_{n-1}^i is defined as the median of three previous period estimates. This is found to be slightly more robust than just using the estimate from

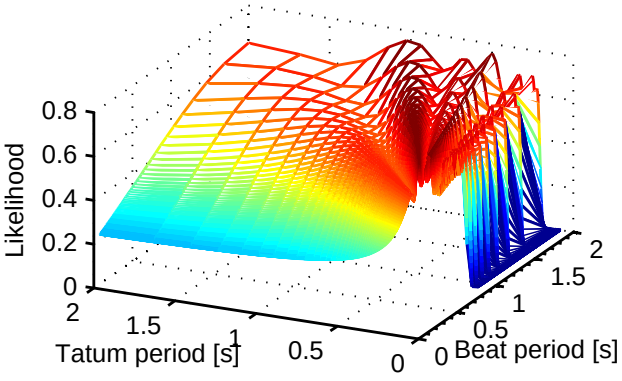

Figure 5. Likelihood of different beat and tatum periods to occur jointly.

the previous frame. The priors are lognormal distributions as described in Eq. (22) in [3, p. 348].

The output of the *Update beat and tatum weights* step in Fig. 4 are two weighting vectors containing the evaluated values of the functions $f^B(\tau_n^B)$ and $f^A(\tau_n^A)$. The values are obtained by evaluating the continuity functions for the set of possible periods given the previous beat and tatum estimates, and multiplying with the priors.

The next step, *Calculate final weight matrix*, adds in the modelling of the most likely relations between simultaneous beat and tatum periods. For example, the beat and tatum are more likely to occur at ratios of 2, 4, 6, and 8 than in ratios of 1, 3, 5, and 7. The likelihood of possible beat and tatum period combinations τ^B, τ^A is modelled with a Gaussian mixture density, as described in Eq. (25) in [3, p. 348]:

$$g(\tau^B, \tau^A) = \sum_{l=1}^{9} w_l \mathcal{N}(\frac{\tau^B}{\tau^A}; l, \sigma_2) \quad (8)$$

where l are the component means and σ_2 is the common variance. Eq. (8) is evaluated for the set of $M \times M$ period combinations. The weights w_l were hand adjusted to give good performance on a small set of test data. Fig. 5 depicts the resulting likelihood surface $g(\tau^B, \tau^A)$. The final weighting function is

$$h(\tau_n^B, \tau_n^A) = \sqrt{f^B(\tau_n^B)\sqrt{g(\tau_n^B, \tau_n^A) f^A(\tau_n^A)}}. \quad (9)$$

Taking the square root spreads the function such that the peaks do not become too narrow. The result is a final $M \times M$ likelihood weighting matrix $\mathbf{H}$ with values of $h(\tau_n^B, \tau_n^A)$ for all beat and tatum period combinations.

The *Calculate weighted periodicity* step weights the summary periodicity observation with the obtained likelihood weighting matrix $\mathbf{H}$. We assume that the likelihood of observing a certain beat and tatum combination is proportional to a sum of the corresponding values of the summary periodicity, and define the observation $O(\tau_n^B, \tau_n^A) = (S[k_B] + S[k_A])/2$, where the indices k_B and k_A correspond to the periods τ_n^B and τ_n^A, respectively. This gives an observation matrix of the same size as our weighting matrix. The observation matrix is multiplied pointwise with the weighting matrix, giving the weighted $M \times M$ periodicity matrix $\mathbf{P}$ with values $P(\tau_n^B, \tau_n^A) = h(\tau_n^B, \tau_n^A) O(\tau_n^B, \tau_n^A)$. The final step is to *Find the maximum* from $\mathbf{P}$. The indices of the maximum correspond to the beat and tatum period estimates $\hat{\tau}_n^B$, $\hat{\tau}_n^A$. The period estimates are passed on to the phase estimator stage.

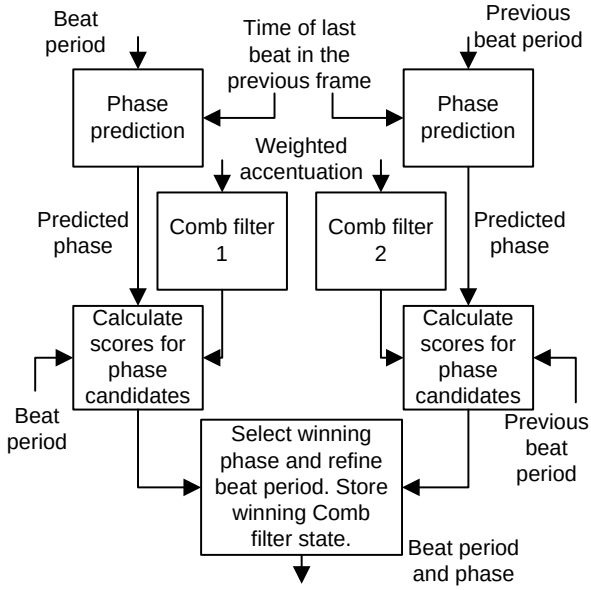

Figure 6. The phase estimation stage finds the phase of the beat and tatum pulses, and may also refine the beat period estimate.

2.6. Beat Phase Estimation

The phase estimation stage is depicted in Fig. 6. The tatum phase is the same as the beat phase and, thus, only the beat phase is estimated. Phase estimation is based on a weighted sum $v[n] = \sum_{i=1}^{4}(6-i)a_i[n]$ of the observed subband accent signals $a_i[n]$, $0 \leq n \leq N-1$. Compared to Eq. (27) in [3, p. 350], the summation is done directly across the accent subbands, instead of resonator outputs.

A bank of comb filters with constant half time T_0 and delays corresponding to different period candidates have been found to be a robust way of measuring the periodicity in accentuation signals [3] [4]. Another benefit of comb filters is that an estimate of the phase of the beat pulse is readily obtained by examining the comb filter states [4, p. 593]. However, implementing a bank of comb filters across the range of possible beat and tatum periods is computationally very expensive. The proposed method utilizes the benefits of comb filters with a fraction of the computational cost of the earlier methods. The phase estimator implements two comb filters. The output of a comb filter with delay τ and

gain α_τ for the input $v[n]$ is given by

$$r[n] = \alpha_\tau r[n-\tau] + (1-\alpha_\tau)v[n]. \quad (10)$$

The parameter τ of the two comb filters is continuously adapted to match the current ($\hat{\tau}_n^B$) and the previous ($\hat{\tau}_{n-1}^B$) period estimates. The feedback gain $\alpha_\tau = 0.5^{\tau/T_0}$, where the half time T_0 corresponds to three seconds in samples.

The phase estimation starts by finding a prediction $\hat{\phi}_n$ for the beat phase ϕ_n in this frame, the step *Phase prediction* in Fig. 6. The prediction is calculated by adding the current beat period estimate to the time of the last beat in the previous frame. Another source of phase prediction is the comb filter state, however, this is not always available since the filter states may be reset between frames.

The accent signal is passed through the Comb filter 1, giving the output $r_1[n]$. If there are peaks in the accent signal corresponding to the comb filter delay, the output level of the comb filter will be large due to a resonance.

We then calculate a score for the different phase candidates $l = 0, \ldots, \hat{\tau}_B - 1$ in this frame. The score is

$$p[l] = \frac{1}{|I_l|} \sum_{j \in I_l} r_1[j] \quad (11)$$

where I_l is the set of indices $\{l, l+\hat{\tau}_B, l+2\hat{\tau}_B, \ldots\}$ belonging to the current frame, $\forall i \in I_l : 0 \leq i \leq N-1$. The scores are weighted by a function which depends on the deviation of the phase candidate from the predicted phase value. More precisely, the weight is calculated according to Eq. (33) in [3, p. 350]:

$$w[l] = \frac{1}{\sigma_3\sqrt{2\pi}} \exp\left(-\frac{d[l]^2}{2\sigma_3^2}\right), \quad (12)$$

but the distance is calculated in a simpler way: $d[l] = (l - \hat{\phi}_n)/\hat{\tau}_B$. The phase estimate is the value of l maximizing $p[l]w[l]$.

If there are at least three beat period predictions available and the beat period estimate has changed since the last frame, the above steps are mirrored using the previous beat period as the delay of comb filter 2. This is depicted by the right hand side branch in Fig. 6. The motivation for this is that if the prediction for the beat period in the current frame is erroneous, the comb filter tuned to the previous beat period may indicate this by remaining locked to the previous beat period and phase, and producing a more energetic output and thus larger score than the filter tuned to the erroneous current period.

In the final step, the best scores delivered by both branches are compared, and the one giving the largest score determines the final beat period and phase. Thus, if the comb filter branch tuned to the previous beat period gives a larger score, the beat period estimate is adjusted equal to the previous beat period. The state of the winning comb filter is stored to be used in the next frame as comb filter 2.

After the beat period and phase are obtained, the beat and tatum locations for the current audio frame are interpolated. Although this reduces the ability of the system to follow rapid tempo changes, it reduces the computational load since the back end processing is done only once for each audio frame.

3. Implementation

The authors have written a real-time implementation of the proposed method for the S60 smartphone platform. The implementation uses fixed-point arithmetic, where all signals are represented as 32-bit integers and coefficients as 16-bit integers. The power estimation low-pass filter is implemented simply as a first-order IIR due to the arithmetic used. Increasing the filter order would have a positive impact on performance, but the given filter design causes that the coefficients exceed 16-bit dynamic scale. Naturally, the accent power compression is realized by a 200-point lookup table. Tables are used also in the period and phase estimation for efficiently computing weight function values. The continuity function, the priors, and the likelihood surface shown in Fig. 5 are stored into lookup tables. Lookup tables are also utilized for storing precalculated feedback gain values for the comb filters. For efficiency, both the autocorrelation and discrete cosine transform processes are implemented on top of a fast Fourier transform (FFT).

For low-latency real-time implementation, the algorithm is split into two execution threads. Referring to Fig. 1, a high-priority "front-end" thread runs the *resampling* and *accent filter bank* stages, feeding their results into a memory buffer. The front-end runs synchronously with other audio signal processing. *Periodicity estimation* and following stages are run in a low-priority "back-end" thread, which is signaled when a new accent frame is available from *buffering* stage. The lower priority allows the back-end processing to take a longer time without interrupting the audio processing, unlinking audio frame length and accent frame length.

4. Evaluation

The proposed algorithm is evaluated in two aspects, beat tracking performance and computational complexity. The methods of Klapuri *et al.* [3] and Scheirer [4] are used as a comparison, using the original authors' implementations. [1]

4.1. Performance

The performance was evaluated by analyzing 192 songs in CD audio quality. Songs with a steady beat were selected from various genres. The majority of songs were rock/pop (43%), soul/R&B/funk (18%), jazz/blues (16%), and electronic/dance (11%) music, and all except two songs were in 4/4 meter. The beats of approximately one minute long song

[1] We wish to thank Anssi Klapuri and Eric Scheirer for making their algorithm implementations available for the comparison.

Table 1. Beat tracking accuracy scores.

Method	Continuity required			Individual estimates		
	Correct	Accept d/h	Period	Correct	Accept d/h	Period
Proposed	60%	70%	76%	64%	76%	79%
Klapuri	66%	76%	73%	72%	85%	81%
Scheirer	29%	34%	30%	53%	65%	59%

Table 2. Processor usage profiles.

Method	Mcycles	Mcycles/s
Proposed	678	2.3
Klapuri (MATLAB)	125000	420
Scheirer	136000	450
Scheirer without `malloc` etc.	119000	390

excerpts were annotated by tapping along with the song playing. The evaluation methodology followed the one proposed in [3], assessing both the period and phase estimation accuracy of the proposed method. A correct period estimate is defined to deviate less than 17.5% from the annotated reference, and the correct phase to deviate less than 0.175 times the annotated beat time. The following scores were calculated and averaged over the duration of the excerpts and over all 192 songs:

- Correct: Beat estimate with correct period and phase.
- Accept d/h: Beat estimate with period matching either 0.5, 1.0, or 2.0 times the correct value, and correct phase.
- Period: Beat estimate with correct period, phase is ignored.

We calculated the scores for both the longest continuous correctly analyzed segment and individual estimates without continuity requirement. For comparison, the methods proposed in [3] and [4] were run on the same data. The results are shown in Table 1. In summary, the proposed method approaches the Klapuri *et al.* method performance in all of the cases. The biggest deviations are in the Scheirer method scores with continuity requirement, reflecting the lack of beat period smoothing in the Scheirer method.

4.2. Complexity

We compared the computational complexity of the three algorithms on a PC having 1.86 GHz Pentium M processor and 1 GB of memory. The proposed and Scheirer methods were implemented in C++ in floating point and compiled with the same compiler settings; function inlining intrinsics were added into Scheirer's original algorithm. The Klapuri method is a combination of MATLAB and C++ code.

A 300-second audio clip was processed five times with each of the three methods and the algorithm CPU time was measured (excluding file access and decoding). The median CPU cycles of the five runs are shown in Table 2, divided by 10^6 (Mcycles), and normalized with audio clip length (Mcycles/s). The Klapuri method is not strictly comparable to the others because it is mostly MATLAB processing: 61% of the CPU is used in MATLAB code. The Scheirer method cycles break down into 82% for comb filtering and 13% for runtime functions (e.g. `malloc`). A second Scheirer profile in Table 2 has the runtime functions subtracted. The proposed algorithm is found over 170 times more efficient.

We also evaluated the computational complexity of the proposed method on a Nokia 6630 smartphone having a 220 MHz ARM9 processor. An instruction profiler was configured to sample the processor program counter on a 1 kHz rate, yielding 302500 data points in total. During playback, 13% of processor time was spent in the beat and tatum tracker implementation and 8% in MP3 format decoding. The profile shows the algorithm to perform very efficiently, comparable to the complexity of the MP3 decoder.

5. Conclusion

A beat and tatum tracker algorithm can be made computationally very efficient without compromising beat tracking performance. We introduced a novel beat and tatum tracker for music signals, consisting of multirate accent analysis, discrete cosine transform periodicity analysis, and phase estimation by adaptive comb filtering. The complexity of the proposed method is less than 1% of Scheirer's method, and its beat tracking accuracy approaches Klapuri's method. The authors have created a real-time implementation of the proposed method for the S60 smartphone platform.

References

[1] J.A. Bilmes. "Timing is of the Essence: Perceptual and Computational Techniques for Representing, Learning, and Reproducing Expressive Timing in Percussive Rhythm." M.Sc. Thesis, Massachusetts Institute of Tech., Sep. 1993.

[2] F. Gouyon and S. Dixon. "A review of automatic rhythm description systems." *Comp. Music J.*, 29(1):34–54, 2005.

[3] A.P. Klapuri, A.J. Eronen, and J.T. Astola. "Analysis of the meter of acoustic musical signals." *IEEE Trans. Audio, Speech, and Lang. Proc.*, 14(1):342–355, Jan. 2006.

[4] E.D. Scheirer. "Tempo and beat analysis of acoustic musical signals." *J. Acoust. Soc. Am.*, 103(1):588–601, Jan. 1998.

[5] J. Seppänen. "Tatum grid analysis of musical signals." In *Proc. IEEE Workshop on Applic. of Signal Proc. to Audio and Acoust. (WASPAA)*, pp. 131–134, New Paltz, NY, USA, Oct. 2001.

[6] C. Uhle, J. Rohden, M. Cremer, and J. Herre. "Low complexity musical meter estimation from polyphonic music." In *Proc. AES 25th Int. Conf.*, London, UK, 2004.

[7] P.P. Vaidyanathan. *Multirate Systems and Filter Banks.* Prentice-Hall, Upper Saddle River, NJ, USA, 1993.