Non-causal Beat Tracking for Rhythm Games

Bram van de Wetering
NHTV Breda University of Applied Sciences

March 14, 2016

Abstract

This paper introduces a non-causal tempo tracking method designed for applications in rhythm game development. The proposed method analyses a musical signal and produces one or more tempo estimates, each comprised of a BPM and offset value, which can be used to compute predicted beat positions. An evaluation is provided which compares the proposed method to existing tempo tracking methods, using a dataset of one thousand songs with tempo values determined by human listeners. The evaluation shows promising results, and indicates the proposed method has the ability to consistently produce accurate estimates for music typically featured in rhythm games.

Contents

1 Introduction

Tempo tracking (also referred to as tempo detection, tempo extraction or BPM detection) is used to estimate the tempo of a piece of music. Tempo is usually expressed in beats-per-minute (BPM), which can be intuitively understood as the number of times a human listener would tap to the beat of a song in one minute. Although tapping is natural for human listeners and requires no musical knowledge, tempo tracking is a non-trivial problem in computing (Dixon, 2001). The field of tempo tracking is closely related to beat tracking, which attempts to find the positions of individual beats, and onset detection, which attempts to find onset positions of individual notes.

Rhythm games generally use tempo data to ensure that input triggers are synchronized with musical events. Examples of games that feature this type of gameplay include Dance Dance Revolution (Konami, 1998-present), Rock Band (Harmonix, 2007-present), Hatsune Miku: Project DIVA (Sega, 2009-present), and Crypt of the NecroDancer (Brace Yourself Games, 2015). All of the example games feature game mechanics that penalize the player for missing inputs and/or performing inputs with inaccurate timing. Since the sync between inputs and musical events often has a direct effect on the evaluation of player performance, rhythm games require highly accurate tempo estimates.

2 Motivation

The objective of this project was to develop a non-causal tempo tracking method that is suitable for tempo estimation in rhythm games. The method is developed specifically for electronic music genres typically featured in rhythm games, and is expected to produce tempo estimates that are highly accurate and do not need human adjustment. Specific criteria used during the evaluation are discussed in section 4.

The proposed method can be applied to content generation for rhythm games in several ways. First, tempo estimates produced by the method can be used to calculate beat positions. The beat positions can then be combined with self-similarity information (Foote & Uchihashi, 2001) to divide a song into logical units such as measures or sections. Beat positions can also be used in the quantization of note positions (possibly also derived from the musical signal with onset detection) to make sure that notes occur on logical fractions of beat positions. The proposed method can also serve as a tool that aids content development for rhythm games, either by verifying tempo values of songs or by automatically filling in tempo values in an editor environment.

3 Method

This section discusses the implementation details of the proposed tempo tracking method. First, the representation of the produced tempo estimates is specified. Second, the three individual phases of the tempo tracking procedure are explained. In the first phase, note onset positions are extracted from a musical signal based on frequency/phase information. In the second phase, a predetermined BPM range is searched for promising BPM values based on how well the predicted beat positions align with extracted onset positions. In the third phase, an offset value is computed for each found BPM value based on note onset positions and slope information from the input signal.

3.1 Representation

In the proposed method, a tempo estimate is represented by a BPM, offset, and fitness value. The BPM determines the period between two beat positions. The offset determines the position of the first beat, which indirectly also determines the position of every following beat. For example, a song with a tempo of 120 BPM and an offset of 0.2 seconds will have beats occurring at $t = 0.2, 0.7, 1.2, \ldots$ This representation assumes a constant BPM value, a limitation that is further discussed in section 5.3. The fitness value gives a relative indication of how preferable an estimate is. The fitness of the primary estimate is always 1, and alternative estimates have a fitness ranging from 0 to 1. In practice, the primary estimate is often significantly more likely to be correct than the alternatives (if presented at all), which tend to have fitness values less than half of the primary fitness.

3.2 Onset extraction

The first phase of the procedure, onset extraction, produces a list of onset positions that is later used in the evaluation of BPM and offset candidates. The test implementation uses onset detection provided by the Aubio library (Brossier, 2003), which implements the note segmentation algorithm presented in (Brossier et al., 2004), but this algorithm can be replaced by other onset detection algorithms with minimal changes to the rest of the procedure.

3.2.1 Onset detection function

First, a phase vocoder computes the Short Time Fourier Transform (STFT) of the input signal. The frequency and phase information from the STFT is used in the onset detection function. Several onset detection functions are available, including High Frequency Content (Masri, 1996), Complex

Domain (Duxbury et al., 2003), Phase Based (Bello & Sandler, 2003), Spectral Difference (Foote & Uchihashi, 2001), Kulback-Liebler (Hainsworth & Macleod, 2003), Modified Kulback-Liebler (Brossier, 2006), and Spectral Flux (Dixon, 2006). To determine the most suitable onset detection function, each function was evaluated with first 100 songs from the dataset according to the criteria specified in 4.2, the results of which are provided in table 1.

Method	HFC	CD	PB	SD	KL	MKL	SF
Accurate	100/100	100/100	83/100	98/100	100/100	97/100	100/100
SSE	0.008	0.007	0.214	0.012	0.005	0.045	0.003

Table 1: Results of the onset detection functions evaluation, showing accuracy and sum of squared errors (SSE).

The evaluation results suggest that Spectral Flux (SF) is the most suitable onset detection functions for the chosen song selection, due to its high accuracy and low error sum. The results agree with the findings presented in (Dixon, 2006), which also suggests that SF is the most promising onset detection function among the tested candidates. Spectral Flux measures the sum of the positive changes in magnitude across all frequency bins of the STFT. The detection function $SF(n)$ for the nth frame in the input signal is defined as:

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

where $X(n,k)$ represents the kth frequency bin of the nth frame, N is the window size, and $H(x) = \frac{x+|x|}{2}$ is the half-wave rectifier function. In the test implementation, a window size of 1024 and hop size of 256 were used.

3.2.2 Temporal peak picking

To determine onset times, the chosen detection function is processed by a temporal peak-picking algorithm. The peak-picker selects onset times from local maxima of the detection function above a certain threshold. This threshold is calculated according to the thresholding operation presented in (Brossier et al., 2004), which uses a small sliding window. The threshold value $\delta_t(n)$ for the nth frame is defined as:

$$\delta_t(n) = \text{median}(D(w)) + \alpha\langle D(w)\rangle$$

where $D(w)$ is the detection function over the set of frames in the window w, median(x) is the weighted median of x, α is a positive weighting factor, and $\langle x \rangle$ is the mean of x. In the test implementation, w was comprised of the frames $\{n-5, n-4, \ldots, n+1\}$ and 0.1 was used for α. Figure 1 shows the onset positions returned by the onset extraction procedure for a given input signal.

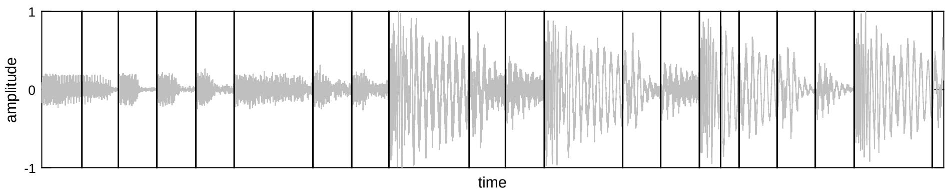

Figure 1: A plot of a 2.5 second fragment from "Vultures" by Kitcaliber (#956), showing the detected onset positions (black) against the input signal (gray).

3.2.3 Silence gate

As the final step in onset extraction, a silence gate is used to eliminate false positives in areas of low energy. Onset times that were previously selected by the peak-picker are rejected if the mean energy of the frames in the current STFT window is below a given threshold loudness, which indicates a silent region. In the test implementation, a threshold of -70 dB was used.

3.3 BPM estimation

The second phase of the tempo estimation procedure, BPM estimation, selects one or more promising candidates from a predetermined BPM range. The chosen implementation is a modified version of the BPM interval testing routine used in Dancing Monkeys (O'Keeffe, 2003), a non-causal approach based on sampling a set of onset positions at regular intervals. This approach is similar to the method presented in (Arentz, 2001), but obtained better results in the evaluation.

3.3.1 Broad interval test

First, an interval test is performed over a broad range of BPM values to get a general estimate of the location of promising BPM candidates. The interval range $[i_{\min}, i_{\max}]$ is computed according to:

$$i_{\min} = F\frac{60}{T_{\max}} \qquad\qquad i_{\max} = F\frac{60}{T_{\min}}$$

where $[T_{\min}, T_{\max}]$ represents the chosen BPM range, and F is the frequency of the input signal. The BPM range of [89, 205] used in the test implementation results in an interval range of [12907, 29730] for input signals with a frequency of 44.1 kHz. To avoid an exhaustive search over thousands of intervals, only every 10th interval in the range is evaluated. A fitness value is computed for each selected interval, which represents the likelihood that the BPM value corresponding to the interval is close to the correct BPM.

To compute the fitness value of an interval i, a histogram with i bins is constructed, where $H(n)$ corresponds to the nth bin of the histogram. Then, for each onset found during onset extraction phase, the onset position p is counted in the bin $H(p \bmod i)$. An example of two histograms can be seen in figures 2 and 3, in which the interval corresponding to 140 BPM (the correct BPM value) shows a stronger concentration of onset positions than the interval corresponding to 180 BPM.

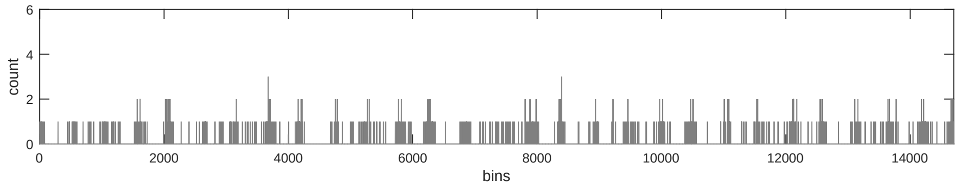

Figure 2: A plot of the onset histogram of “Vultures” for 180 BPM ($i = 14700$), showing a relatively weak concentration of onsets positions.

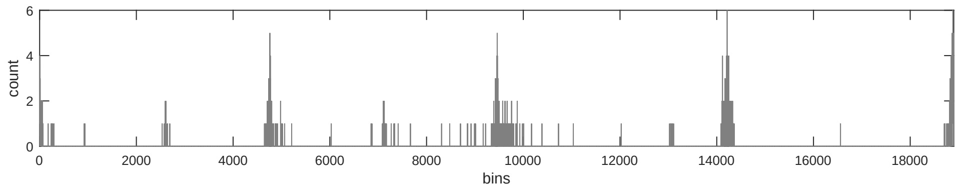

Figure 3: A plot of the onset histogram of “Vultures” for 140 BPM ($i = 18900$), showing a strong concentration of onsets at sixteenth note positions.

After counting all onset positions in the histogram, the confidence value of each onset position is calculated according to:

$$C(p) = f(p) + \frac{f(p + 0.5i)}{2}$$

where $C(p)$ represents the confidence value at position p and f is the evidence function. The inclusion of evidence from $f(p)$ and $f(p + 0.5i)$ allows for onsets that occur near primary beat positions as well as off-beat positions to contribute to the overall fitness of the interval i. The highest confidence value $C(p)$ of all onset positions p for interval i is selected as fitness value of i. The evidence function $f(p)$ represents the sum of the bins surrounding p multiplied by a Hamming window, and is defined as:

$$f(p) = \sum_{n=1}^{w} \left(0.54 - 0.46 \cos\left(\frac{2\pi n}{w-1}\right)\right) H((p - 0.5w + n) \bmod i)$$

where w is the Hamming window size. An example of fitness values resulting from a broad interval test can be seen in figure 4.

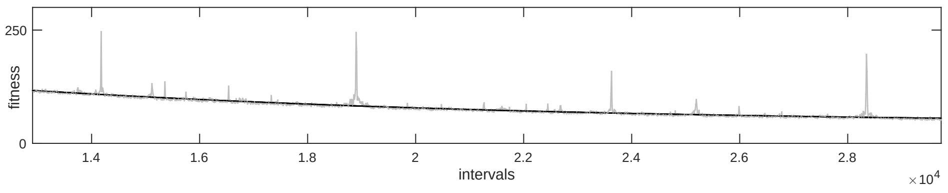

Figure 4: A plot of the broad interval test of "Vultures", showing the fitness values of every 10th interval (gray) and the polynomial curve approximation (black). The peak at $i = 18900$ corresponds to the correct BPM value.

As figure 4 illustrates, the fitness values are biased towards the low end of the interval range, which corresponds to high BPM values. This is caused by the relatively small histogram size for short intervals, which increases the average onset count per bin. To remove the bias, the fitness values are fitted to a 3rd degree polynomial curve approximation, which is then subtracted from the computed fitness values.

3.3.2 BPM candidate selection

After the fitness values are adjusted with the curve approximation, the maximum fitness value of the broad interval test is determined. Then, fitness values are computed for the 'missing' intervals in the range $[i-9, i+9]$ for every interval i with a fitness above 40% of the maximum fitness. The BPM value corresponding to interval with the highest fitness in this range is then added to a list of BPM candidates. Table 2 shows the BPM candidates resulting from the fitness data in figure 4.

Interval	14175	18901	23625	28350
BPM	186.666	139.992	112.000	93.333
Fitness	160.7	250.6	96.1	147.34

Table 2: BPM candidates in the interval test of "Vultures", which were selected from the 19 intervals surrounding the four peaks above the 40% fitness treshold.

To further increase the accuracy of the tempo estimates, the BPM candidates are processed using a set of hand-tuned heuristics. The BPM candidates selected during the interval test are usually either close to the correct BPM or a simple fraction of the correct BPM such as $\frac{1}{2}$ or $\frac{2}{3}$. To remove 'duplicates', BPM candidates that differ by less than 0.1 from a multiple of a candidate with a higher fitness value are rejected. Small differences between the correct BPM and the closest BPM candidate are also common, due to inaccuracies in the detected onset positions and the fact that not all BPM values can be represented by integer intervals. To address this, BPM candidates that differ by less than 0.05 from an integer value are rounded and re-evaluated with a (possibly fractional) interval. If the fitness value of the rounded BPM exceeds 99% of the original fitness value, the rounded BPM is assumed to be correct and replaces the original BPM candidate. Finally, if the ratio between the fitness values of the primary and secondary BPM candidates is less than 1.05, the fitness values of all BPM candidates are recomputed with accurate, fractional intervals.

3.4 Offset estimation

The third phase of the tempo estimation procedure, offset estimation, computes the most optimal offset value for each of the remaining BPM candidates selected in the second phase. A modified version of the interval fitness computation is used to compute an initial estimate, followed by a slope computation which aims to reduce the number of off-beat estimates.

3.4.1 Initial estimate

For offset estimation, the onset histogram used in the interval testing procedure is reconstructed for each BPM candidate. Since prominent bass notes tend to occur on beat positions, bins close to the correct offset value will have a relatively high count of onsets. This means the fitness function used during the board interval test can also be used during offset estimation, but this time the function evaluates the onset position with the highest fitness value instead of the fitness value itself. The initial offset estimate o is computed as:

$$o = \frac{p_{\max} \bmod i}{F}$$

where o is the predicted position of the first beat in seconds, F is the frequency of the input signal, i is the interval of the BPM candidate, and $p_{\max}$ is the onset position p with highest confidence $C(p)$ for the interval i according to the confidence function presented in 3.3.1. Figure 5, an onset histogram similar to figure 3, shows the relation between the correct/off-beat offset positions and bins with a high onset count.

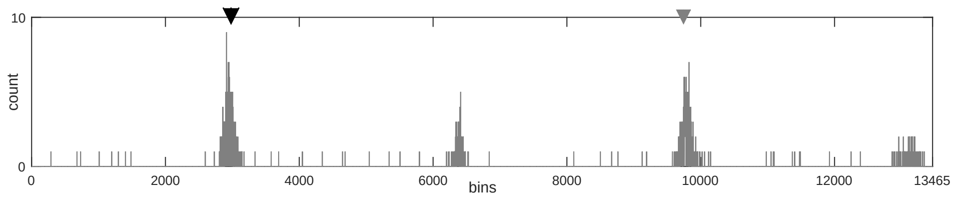

Figure 5: A plot of the onset histogram of "Magic Cycles" by DJ Sharpnel (#541), with two arrows indicating the correct offset value (black, $o = 0.067$) and the off-beat offset value (gray, $o = 0.220$).

3.4.2 Off-beat reduction

In order to decrease the number of offset estimates that are off by a half-beat interval, an off-beat reduction step is performed. First, a sample buffer that approximates the slope function of the input signal is constructed. The amplitude of each sample in the slope buffer is computed by sliding a window over the input signal, such that:

$$S(p) = \max\left(\sum_{n=1}^{w}\left|I(p+n)\right| - \sum_{n=1}^{w}\left|I(p-w+n)\right|, 0\right)$$

where $S(p)$ is the slope function at sample position p, $I(p)$ is the amplitude of the input signal at p, and w is the window size (50 milliseconds of samples in the test implementation). Each sample in the slope buffer records the difference in energy between the w samples preceding and w samples following the sample position; a large increase in relative energy output of the signal will result in high amplitudes in the slope buffer. An example of a slope function can be seen in figure 6, which shows several peaks that correspond to high energy onsets in the input signal.

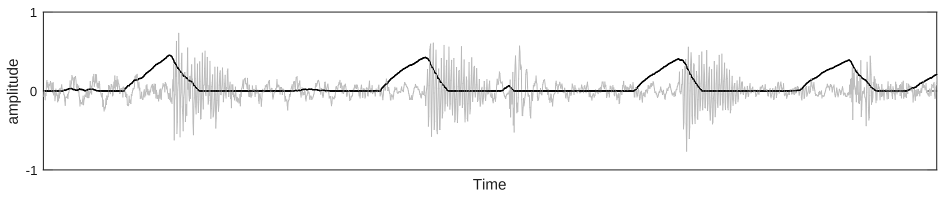

Figure 6: A plot of a fragment from "Drift" by Metrik (#260), showing the normalized slope function (black) against the input signal (gray).

After the slope buffer has been computed, it is sampled at the predicted beat positions $\{o, o+i, o+2i, \ldots\}$ and the half-beat offset positions $\{o + 0.5i, o+1.5i, o+2.5i, \ldots\}$, where o is the predicted position of the first beat in seconds and i is the interval between two beat positions in seconds. If the average amplitude sampled at the half-beat offset positions exceeds the average amplitude sampled at the predicted beat positions, the half-beat offset position $o + 0.5i$ is used for the final estimate instead of the initial estimate o.

4 Evaluation

This section discusses the data and criteria that were used to evaluate the accuracy of the proposed tempo tracking method in comparison to existing methods. The evaluation makes use of content and mechanics featured in the rhythm game In The Groove (Roxor Games, 2006), also referred to as ITG. First, the dataset of songs is described. Second, the criteria used to classify the results of the BPM and offset tests are specified and motivated.

4.1 Dataset

The chosen tempo tracking methods were evaluated with a dataset comprised of 1000 songs, selected from community content available for ITG. All selected songs have a single, consistent tempo (estimated by the content creator) and were selected from song packs that are known to have accurate tempo estimates. The dataset contains music typically found in rhythm games; most of the selected songs belong to electronic music genres such as Trance, Dance, Techno, Drum and Bass, Chiptune, Dubstep and Pop Music. A complete list of songs with title, artist and pack information is provided in the appendix.

4.2 Test criteria

In the evaluation results, all BPM and offset estimates are classified as either accurate, close, or wrong, based on the distance between the estimate and the original value. There is no real objective way to determine if an estimate is 'close enough' to the original; a timing difference of several milliseconds might go unnoticed by a novice player, but not by an expert player aiming for a perfect score. However, sensible threshold values can be worked out by using the timing windows of ITG as a baseline.

For evaluation purposes, an estimate is considered accurate if it fits the smallest timing window used in ITG. To receive full score for a note in ITG on standard settings, player input has to occur within 21.5 milliseconds of the note onset. Therefore, estimates resulting in beat positions that differ by less than 21.5 milliseconds from the correct beat positions are considered accurate. If larger differences occur, tapping precisely on beat would result in a score penalty according to the game mechanics of ITG. Estimates that do not qualify as accurate are considered close if the difference is less than 102 milliseconds, the largest timing window ITG still considers a 'hit', and wrong otherwise.

4.3 BPM threshold

Since timing errors induced by an incorrect BPM value accumulate over time, the acceptable range of a BPM estimate depends on both a threshold value and song duration. The song duration used in the evaluation was 134 seconds, the average duration of all songs in the dataset. The allowed range of BPM values for accurate and close estimates can be derived from the correct BPM based on the following two formulas:

$$T_{min} = \frac{T_{base} \cdot d}{d + t} \qquad T_{max} = \frac{T_{base} \cdot d}{d - t}$$

where T_{base} is the correct BPM, T_{min} and T_{max} specify the range of BPM values that are considered accurate/close, d is the chosen duration (134 seconds), and t is the chosen threshold (0.0215 seconds for 'accurate', 0.102 seconds for 'close'). BPM estimates that are unit fractions or multiples of the correct BPM are also accepted in the difference calculations, since those estimates still result in correctly aligned beat positions.

4.4 Offset threshold

Since offset estimates are expressed in seconds, the chosen threshold values apply directly. An offset estimate that differs by less than 21.5 milliseconds from the correct offset is considered accurate, and similar rules apply for close and wrong offsets. Distance calculations between offset values are wrapped within the range of $[0, i]$, where i is the interval between two beat positions in seconds, since adding or subtracting multiples of i from the offset will result in identical beat positions. Offset estimates that are close to the correct offset shifted by $0.5i$ are also accepted in the difference calculations. However, those estimates are marked as off-beat, since this offset corresponds to the halfway point between correct beat positions.

5 Results

This section discusses the results of the evaluation according to the criteria specified in section 4. First, the BPM detection results are discussed, followed by the offset detection results. For the proposed method, only the primary estimate was considered, even in cases where one of the alternative estimates provided a more accurate result. Afterwards, the limitations of the proposed method are discussed.

5.1 BPM estimation results

For the evaluation of the BPM estimates, the proposed method was compared against 4 existing tempo tracking methods. The first method is the

Matlab implementation of Dancing Monkeys (O'Keeffe, 2003), which formed the basis of the BPM estimation phase discussed in section 3.3. The second method is part of the LabROSA project, which uses an implementation of the dynamic programming method described in (Ellis, 2007) and was developed using human tapping data for the 2006 MIREX Beat Tracking evaluation. The third method is the beat tracking implementation used in the open source DJ software Mixxx (Andersen & Andersen, 2003), which uses a hybrid of the two state beat tracking model described in (Davies & Plumbley, 2005, 2007) and the dynamic programming method described in (Ellis, 2007). The fourth method is the beat tracking implementation used by Virtual DJ (Atomix Productions, 2003), of which the implementation details are not publicly available. Table 3 shows the results of the comparison.

Method	Accurate	Close	Wrong	SSE	AE
Proposed method	970/1000	25/1000	5/1000	0.133	0.004
Virtual DJ	873/1000	47/1000	80/1000	1.403	0.014
Mixxx	797/1000	71/1000	132/1000	2.661	0.023
Dancing Monkeys	633/1000	330/1000	37/1000	1.393	0.025
LabROSA	111/1000	85/1000	804/1000	12.301	0.102

Table 3: BPM estimation results showing accuracy, sum of squared errors (SSE), and average error (AE).

The evaluation results show that the proposed method has the highest number of accurate results and lowest error values compared to the other methods. Table 4 shows details of the five BPM estimates that were wrong.

Song	Real BPM	Est. BPM	Alt. BPM	Fitness ratio	BPM ratio
#228	135.000	202.495	135.000	0.67	2/3
#275	178.000	133.495	178.000	0.79	4/3
#363	140.000	105.000	140.000	0.85	4/3
#369	110.000	164.973	109.989	0.68	2/3
#440	69.000	103.501	138.000	0.59	2/3

Table 4: Comparison of the BPM estimates that were classified as 'wrong' in the evaluation, showing the song number, correct BPM, primary estimate BPM, alternative estimate BPM, ratio of the alternative fitness to the primary fitness, and ratio of the correct BPM to the primary estimate BPM.

Inspection of the estimates in table 4 reveals that the correct BPM values are all $\frac{2}{3}$ or $\frac{4}{3}$ fractions of the (incorrect) primary estimate. In all five cases the alternative estimate would have been classified as accurate, but the primary estimate was picked due to a higher fitness value. The relatively low fitness of the correct BPM can be attributed to the use of many triplet notes, which occur at $\frac{1}{3}$ and $\frac{2}{3}$ beat positions. Because the confidence function presented in 3.3.1 gathers evidence from the off-beat positions, which occur at $\frac{1}{2}$ beat positions, the onsets of triplet notes do not contribute to the fitness value of the correct BPM estimate.

5.2 Offset estimation results

For the evaluation of the offset estimates, the proposed method is compared against Dancing Monkeys by K. O'Keeffe (O'Keeffe, 2003), which is the the only other implementation that provides explicit offset estimates. Table 5 shows the results of the comparison.

Method	Accurate	Close	Wrong	Off-beat	SSE	AE
Proposed method	974/1000	19/1000	7/1000	142/1000	0.141	0.006
Dancing Monkeys	764/1000	217/1000	19/1000	227/1000	0.893	0.019

Table 5: Offset estimation results showing accuracy, off-beat results, sum of squared errors (SSE), and average error (AE).

The evaluation results show that while more than 97% of the offset estimates from the proposed method are accurately aligned to the correct beat positions or off-beat positions, the number of instances in which the off-beat position is chosen over the correct position is still relatively high. This can be attributed to a large number of onsets corresponding to eighth notes that occur on off-beat positions. The off-beat reduction step described in 3.4.2 attempts to address this problem by factoring in the amplitude of the onsets, but additional processing is required in order to further reduce the number of off-beat estimates.

5.3 Limitations

In order to achieve the high level of accuracy required for tempo estimates in rhythm games, several limitations are imposed on the range of input signals for which an accurate estimate can be computed.

The most significant limitation is that both the BPM detection and offset detection algorithms can only produce reliable estimates for songs with a constant tempo. This limitation is inherent to the method used, since it uses onsets from the entire input signal to gather evidence for a single tempo value. Beat tracking methods that address this limitation have been developed in the past. The causal beat prediction algorithm described in (Davies & Plumbley, 2004; Davies et al., 2005) handles tempo changes, and the dynamic programming method described in (Ellis, 2007) is able to track changes within ±10% of the target tempo. However, an initial evaluation of these methods suggested that the produced tempo estimates are not accurate enough to be used in rhythm games.

The second limitation of the proposed method is that it is designed to produce good results for songs from the chosen dataset, which represents music typically featured in rhythm games. Most songs from the dataset have low frequency, high energy bass notes on or near beat positions that are relatively easy to identify. It is unclear how reliable the tempo estimates of this method will be for songs without distinct bass notes, which appear in genres such as Classical, Jazz, or acoustic music in general.

References

Andersen, T., & Andersen, K. (2003). *Mixxx.* Retrieved from `http://mixxx.org/`.

Arentz, W. A. (2001). Beat extraction from digital music. *Proc. of NORSIG, Trondheim, 2001.*

Atomix Productions. (2003). *Virtual DJ.* Retrieved from `http://www.virtualdj.com/`.

Bello, J. P., & Sandler, M. (2003). Phase-based note onset detection for music signals. In *Acoustics, speech, and signal processing, 2003. proceedings. (icassp'03). 2003 ieee international conference on* (Vol. 5).

Brossier, P. M. (2003). *Aubio, a library for audio labelling.* Retrieved from `http://aubio.org/`.

Brossier, P. M. (2006). *Automatic annotation of musical audio for interactive applications* (Unpublished doctoral dissertation). Queen Mary, University of London.

Brossier, P. M., Bello, J. P., & Plumbley, M. D. (2004). Real-time temporal segmentation of note objects in music signals. In *Proceedings of the ICMC.*

Davies, M. E., Brossier, P. M., & Plumbley, M. D. (2005). Beat tracking towards automatic musical accompaniment. In *Audio engineering society convention 118.*

Davies, M. E., & Plumbley, M. D. (2004). Causal tempo tracking of audio. In *Ismir.*

Davies, M. E., & Plumbley, M. D. (2005). Beat tracking with a two state model. In *Acoustics, speech, and signal processing, 2005. proceedings. (ICASSP'05). IEEE international conference on* (Vol. 3, pp. 241–244).

Davies, M. E., & Plumbley, M. D. (2007). Context-dependent beat tracking of musical audio. *Audio, Speech, and Language Processing, IEEE Transactions on*, *15*(3), 1009–1020.

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

Dixon, S. (2006). Onset detection revisited. In *Proceedings of the 9th international conference on digital audio effects* (Vol. 120, pp. 133–137).

Duxbury, C., Bello, J. P., Davies, M., Sandler, M., et al. (2003). Complex domain onset detection for musical signals. In *Proc. digital audio effects workshop (dafx)* (pp. 6–9).

Ellis, D. P. (2007). Beat tracking by dynamic programming. *Journal of New Music Research*, *36*(1), 51–60.

Foote, J., & Uchihashi, S. (2001). The beat spectrum: A new approach to rhythm analysis. In *IEEE, international conference on multimedia & expo* (p. 224).

Hainsworth, S., & Macleod, M. (2003). Onset detection in musical audio signals. In *Proc. int. computer music conference* (pp. 163–6).

Masri, P. (1996). *Computer modelling of sound for transformation and synthesis of musical signals.* (Unpublished doctoral dissertation). University of Bristol.

O'Keeffe, K. (2003). Dancing monkeys. *Masters project*, 1–66.