An adaptative real-time beat tracking system for polyphonic pieces of audio using multiple hypotheses

Oscar Mayor
Music Technology Group, Pompeu Fabra University
oscar.mayor@iua.upf.es, http://www.iua.upf.es/mtg

Abstract

In this paper a method for extracting beat information of a piece of music is presented, a real-time analysis is performed while the music is played or recorded from any source and the system gives the beats per minute value at each moment and beat occurrences in time. It also becomes adaptative in case of a sudden or smooth change in the tempo. It deals with multiple hypotheses, and gives the most suitable results at each time. One of the most important applications linked to this work is the automatic classification of pieces in different musical genres or finding song similarities in terms of musical rhythm.

1 Introduction

To extract rhythm information from a piece of music has become a difficult task in the scope of computer music research. To achieve a musical representation of the rhythm of a song in a polyphonic recording including not only percussive instruments is not giving good results yet, although some research has been done and there are satisfactory results for drum loops, in terms of music transcription. One of the most important topics related to this work is the automatic classification of pieces in different musical genres or finding song similarities in terms of musical rhythm. One of the first steps to achieve a high level of rhythm description is the beat level. It means to detect beat occurrences in time, the beats that a listener usually tap with his foot while listening to a song, and derive BPM (beats per minute) information from it. Beat occurrences commonly match with points of maximum energy of the sound or at least correspond with points of high energy. In this article a method for extracting BPM information of a song is presented and some conclusions about applications and future work are discussed.

2 Previous work

Previous approaches related to rhythm tracking and beat induction include several approaches.

Goto and Muraoka [1] [2] present a method, that works in real-time in a parallel-processing computer, that extracts drum patterns from a musical signal and uses a template-matching model to determine the beat of the song being analyzed. They also have presented a system that works with drum-less audio signals [3].

Scheirer [4] also presents a real-time beat-tracking system, which using a small number of band-pass filters and banks of parallel comb filters extracts the beat from musical signals of arbitrary polyphonic complexity. This system can be used to predict when beats will occur in the future.

Dixon [6] presents a bottom-up approach to beat tracking from acoustic signals deriving time signature and approximate tempo from the timing patterns of detected note onsets.

Desain and Honing [7] have made a lot of research in computational modeling of beat-tracking, their models begin looking for inter-onset intervals associating a rhythm pulse with the interval stream.

Gouyon [8] have proposed a method for classification of drum-loops detecting the minimum inter-onset interval which he calls the tick, and looks for percussive templates in the most relevant ticks of the drum loop.

3 System overview

The flow diagram in figure 1 shows the functionality of the system and can be followed to easily understand the processing of the system.

First of all we apply a smoothing window and zero-padding to the sound source which is going to be analyzed and then we do the Short Time Fourier Transform that produces a Frame-by-Frame Frequency Spectrum of the input signal. Then we apply a bank of filters over the magnitude spectrum to calculate the energy evolution in time of each filter, and multiplying this energy evolution by an arbitrary factor for each filter we get a Total Average Energy that will be the basic information where to look for BPM candidates. A pseudo-correlation method that will be explained later in section five, searches for BPM candidates and adds them to a list of hypotheses from where we derive the most probable hypothesis at each moment.

We also synthesize a perceptual rhythm from an important simplification of the input audio data and we display what is called the BPM spectrum, that is explained in detail in sections four and six respectively.

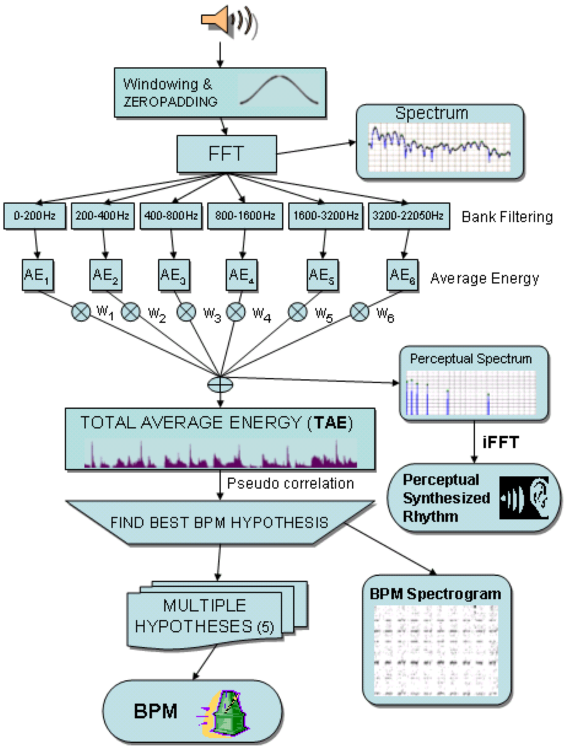

Figure 1 System overview of the beat-tracking system

4 Perceptual beat energy extraction

The system accepts any kind of audio data as input, including the compressed MP3 format, and this data is automatically converted to PCM raw audio with a quality that can be changed by the user: mono/stereo, 8/16 bit, 22050/44100Hz.

Good results are obtained dealing with mono, 16 bit, 22050 Hz, PCM raw audio, so there is no need to work with better quality because the calculation time will increase a lot and the results will be more or less the same.

Assuming that beat occurrences usually match with maximum energy points of the waveform or at least points with high energy, we can rely on energy to find beats in a musical excerpt. We perform a perceptual simplification of the audio data in order to work with few data but with enough perceptual information to perceive the rhythm of the song. The perceptual simplification is done in the frequency domain, so first of all an smoothing analysis window is applied to each frame of audio data (512 samples - 23 ms), with half overlap so we have a minimum time resolution of 11,5 ms, and then the Short Time Fourier Transform (STFT) of each frame is done applying zero padding to get an smoother spectrum with 1024 bins. Once in the frequency domain, a bank of filters like the one presented in [4] is calculated. One low-pass (0-200Hz), one high-pass (3200-22050Hz) and four band-pass filters (200-400Hz, 400-800Hz, 800-1600Hz, 1600-3200Hz) are created and the Average Energy of each filter (AE_j) is calculated in this way:

$$\sum_{j=1}^{nFilters} AE_j = \frac{\sum_{i=left}^{right} MagSpectrum[i]^2}{right-left}$$

where nFilters is the number of filters, in our case six, left and right are the left and right bins corresponding to the frequency cut-offs of each filter's spectrum and MagSpectrum[i] is the magnitude value of the i-th bin. If we listen to the synthesized sound calculated computing the inverse fft of an spectrum with six peaks, each one representing the middle frequency of each filter, modulated to the average amplitude of the filter that it represents, we can perceive a sound that inherits a reliable representation of the rhythm of the original sound. So an important simplification of the original data has been done preserving the necessary rhythm information for later processing.

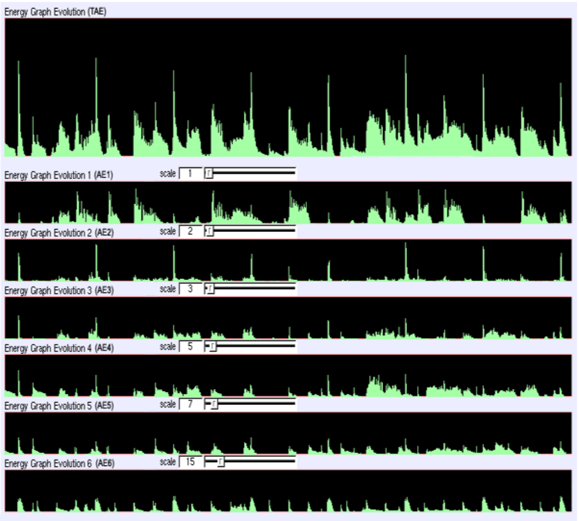
Energy Graph Evolution (TAE)
Energy Graph Evolution 1 (AE1)
scale 1
Energy Graph Evolution 2 (AE2)
scale 2
Energy Graph Evolution 3 (AE3)
scale 3
Energy Graph Evolution 4 (AE4)
scale 5
Energy Graph Evolution 5 (AE5)
scale 7
Energy Graph Evolution 6 (AE6)
scale 15

Figure 2 TAE and ponderated AE Graph Evolution for each filter, peaks in the TAE graph represent beats, we can see that high beats are equally separated, representing a constant BPM value.
In figure 2 we can see a graphical representation of the evolution of the energy for each filter along the time, so high peaks in the y-axis represent instants of time where the average energy of the filter is very high and these points are good candidates for beats.

The Total Average Energy (TAE) also shown in figure 2, is calculated as the ponderated sum of each filters' average, divided by the number of filters, in our case six.

$$TAE = \frac{w_1 \cdot \mathrm{AE}_1 + w_2 \cdot \mathrm{AE}_2 + w_3 \cdot \mathrm{AE}_3 + w_4 \cdot \mathrm{AE}_4 + w_5 \cdot \mathrm{AE}_5 + w_6 \cdot \mathrm{AE}_6}{\mathrm{nFilters}}$$

In order to extract the BPM (Beats Per Minute) information, we will focus in the Total Average Energy evolution along the time.

5 Maximum Beat Correlation

Once we have the beat information along the time (energy graph evolution, see figure 2), we try to find equally distant beats in time in order to find the correct BPM for the part of the song being analyzed. We search for candidates only when a high energy beat value is detected, this will happen only when the energy of the current beat is at least twice the average of some previous frames (last 100 frames in this implementation). To find the BPM value we use some kind of time-domain Comb-Filtering algorithm over the energy graph evolution (beats along the time) to obtain high energy and equally distant beat repetitions that will determine the most adequate BPM of the musical excerpt being analyzed.

We calculate the sum of some equally separated time-positions' energy, and then varying the separation between points into a limited range, determined by the maximum and minimum BPM value that we accept (between 50 and 200 BPM), we obtain a list of values each one representing the score of each possible BPM, then we store these values in an array, that once sorted will show the more probable and less probable BPM values at each time. This method gives the number of frames between high beats, this value is converted to a BPM value with the next conversion:

$$Tempo\,(BPM\,) = \frac{1}{GAPFRAMES\ \cdot \frac{SIZE}{2} \cdot \frac{1}{SAMPLINGRATE}} \cdot 60$$

where GAPFRAMES is the number of frames between high beats, SIZE is the number of samples in a frame (512) and SAMPLINGRATE is the sampling rate of the sound (22050/44100 Hz).

6 BPM Spectrogram

In order to represent graphically the results of the previous analysis, so a user can derive BPM information from it, several graphics have been represented showing values of energy, time position and distance between beats. In a two dimensional space we can only represent the best BPM value calculated at each moment, so y-axis will represent the distance between beats (from where we can derive BPM), and x-axis will represent the evolution in time, the problem with this representation is that we are discarding BPM values calculated with lower score but very close to the maximum that could be the correct tempo value at this moment. So the best way consists in displaying three parameters in a 3-axis representation, similar to the Beat Spectrum idea presented by Foote [5]. X-axis will represent time, y-axis will represent the BPM value or distance between high beats and z-axis, the importance of this BPM value in terms of correlation of an equally distant peaks template with the energy values at an interval of time assuming this BPM. (score of this BPM).

In figure 3 we can see this graphical representation in 2-D while representing the third axis with color depth as a gray-scale, where white represents the lowest value and black the highest value. We can see the darker horizontal line that represents the most probable BPM for the song; other lines represent half and double tempo or beats that follow a constant pattern of repetition but do not match with the multiples or submultiples of the main beat.

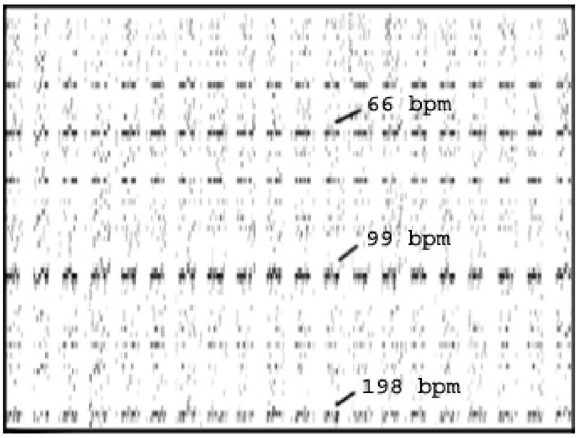

Figure 3 In this BPM spectrogram, we can see the darker horizontal line representing the correct bpm (99bpm) and other lines representing double tempo (198bpm) and 2/3 of tempo (66bpm).

7 Multiple Hypotheses

With a graphical representation like the so-called BPM Spectrogram that we have presented before, it's quite easy for a human to decide what BPM mostly identifies a song, but for a computer it's a more difficult task, the idea is to work with multiple hypotheses during the analysis process, and decide which one of the hypotheses is the best at each moment, because we're trying to get the BPM of the song in real-time when the song is being recorded or played. We deal with multiple BPM hypotheses in order to give a reliable result. In case of a changing tempo in a small interval of time that should be considered as artefacts instead of a tempo change, the system is conservative and robust enough to consider the most possible hypothesis as the correct one, we use a different approach as the one presented in [2].

Each time that a high beat occurs, the system calculates the best BPM candidates with the method described before and new hypotheses are added to the system, that decides which hypotheses are consistent, or what hypotheses are no longer valid.

The algorithm used to decide what hypothesis should be considered valid or wrong can be easily adapted in order to make the system more robust or more flexible to tempo changes, and also to tune the accuracy of the system, but this is always a compromise between quality and functionality.

For each hypothesis we have six properties: The BPMvalue, Value, Hits, HowOften, Duration and Score. BPMvalue is the number of beats per minute, Value represents the average correlation value for all the times that this hypothesis has appeared, Hits is the number of total apparitions, HowOften is the frequency of appearance in the last 100 calculations, Duration is the time since the first appearance of the hypothesis and Score is the score given to the hypothesis calculated from the rest of properties.

When an hypothesis comes up, first of all it's compared with the existing and if it's similar to one in the list, its score is recalculated or if it's new, it's added to the list replacing the worst one. The Score of the hypotheses is calculated multiplying their average value (Value property) with the HowOften value. Other properties like Hits and Duration are important to know if an hypothesis is new or old or if it's active.

At the same time that we calculate the score of the hypotheses, we decide which one is the worst, that will be replaced when a new one comes out.

Some assumptions are made that affect the scoring of the hypotheses:

- A very recently added hypothesis is scored higher than an old hypothesis.
- A very active hypothesis is never discarded even if it gets a low score.

So we try to never reject the new or very active hypotheses, even if they get bad punctuation. In these cases, these hypotheses remain in the list, and the next bad hypothesis is replaced by the new one.

8 Beat occurrences in time

Once the tempo of the song is "correctly" detected, the next step is to extract higher level information about the rhythm patterns of the song. This is a difficult task because if we work with different genres, it's not helpful to try to find spectral shapes to discriminate an instrument because a snare drum or tom sound has a different sound across genres and also different spectral shape from one style to another.

We have to deal even with music with no drums or even without any percussive instrument, which doesn't mean that it has no rhythm, it's just that the rhythm is more present in the bass lines, guitar chord progressions or in any other instrument. This is one of the problems that systems based in pattern matching [2] [8] encounter when there is no snare or bass drum to search for.

In the actual implementation we have information about when the beat occurs, so the deviation between the predicted beat and the real beat can be easily calculated and derive the level of expressivity, in this case the rubato. We have also information about secondary beats, often with less energy than the main beats, which doesn't occur at the beat level but at half, quarter or third tempo. This information characterize the rhythm pattern of the song and is the basic information to extract high level attributes that will make it easy to find, with this rhythm information, similarities between songs.

9 Conclusions and future work

The beat-tracking system has not still been systematically tested. There are a lot of parameters that can be adjusted to tune the accuracy or functionality of the system like filters' width, scaling factors, high beat threshold, frame size, type of window, zero-padding and overlap factors, hypotheses decisions, etc. Many of these parameters can be adjusted in real-time by using a complete GUI.

With a large song's database including several genres and styles, the system will be fully tested and these parameters adapted to achieve the best results for all the songs or even adapt the system during the analysis process to improve the results depending on the song that is being tracked.

Once that the BPM of a song is correctly detected, the next step is to find more information about the rhythm of the song, rhythm descriptors, transcription, finding patterns of repetition, and another high level attributes that give information about the kind of rhythm that is being analyzed. It would also be important to focus in other aspects like chord detection or changes in the melody and harmony to help the system to find similarities between songs, and not only looking for rhythm similarity but also close harmonic or melodic structures. This is part of the current work, that will lead to an extensible content based analysis system.

References

[1] Goto, M., Muraoka, Y., "An audio-based real-time beat tracking system and its applicatios", in *ICMC Proceedings 1998*.

[2] Rosenthal, D., Goto, M., Muraoka, Y., "Rhythm tracking using multiple hypotheses", in *ICMC Proceedings 1994*.

[3] Goto, M., Muraoka, Y., "Real-time rhythm tracking for drumless audio signals – chord change detection for musical decisions", in *IJCAI-97 Workshop on computational auditory scene analysis*, pp. 135-144, 1997.

[4] Scheirer, E., "Tempo and beat analysis of acoustic musical signals", in *J. Acoust. Soc. Am.* 103(1), jan 1998, pp 588-601.

[5] Foote, J., Uchihashi, S., "The beat spectrum: A new approach to rhythm analysis", in *IEEE International Conference on Multimedia & Expo 2001*, Tokyo, Japan.

[6] Dixon, S., "A beat tracking system for audio signals", in *Proceedings of the Conference on Mathematical and Computational Methods in Music*, Vienna, Austria, Dec. 1999, pp 101-110.

[7] Desain P., "A (de)composable theory of rhythm perception", in *Music Perception 9*, 439-454.

[8] Gouyon F., Herrera P., "Exploration of techniques for the automatic labelling of embedded instruments in audio drum tracks", in *Mosart Workshop*, Barcelona 2001.