

Real Time Tempo Analysis of Drum Beats

Author: **Philip Hannant**, Supervisor: **Professor Steve Maybank**

Wednesday 30th March, 2016

Contents

1	Introduction & Background	3
1.1	Beat Detection Background	3
1.1.1	Onset Detection	4
1.1.2	Short-Term Fourier Transform	4
1.2	Audio Analysis using the Discrete Wavelet Transform	4
2	Aims and Objectives	5
2.1	Core Project Features	5
2.2	Non-Core Project Features	5
3	Development Plan for the Solution	5
3.1	Beatroot	5
3.1.1	JWave and Beat Detection algorithm	6
3.2	Live Audio Processing	6
3.3	Comparative analysis of Beatroot and JWave	7
3.4	Develop GUI & logging system	7
3.5	Build Extensive drum sample set	7
4	Project Schedule	8
5	References	9

1 Introduction & Background

Having played the drums for nearly twenty years I have experienced a number of different training tools to help improve my timing, these tools have always been exclusive to midi driven electric drum kits. These drum kits are able to include training tools within their drum modules which utilise the midi events being triggered by the player and give live feedback to the exact timing and instrument being played. An example of such a system is the Roland DT-1 V-Drums tutor which is a software package that can be connected to a Roland V-Drum module in order to provide the player with an interactive experience that helps a drummer improve their rhythm, coordination and sight reading. Such an extensive and accurate range of tutoring functions is only made possible by the midi events that are intrinsic to an electronic drum kits function. (an example of the GUI can be found in figure 1) [roland ref]. These tutoring tools are not however available to a drummer who does not own an electronic drum kit, they are restricted to using a metronome and using their ear to determine their timing and rhythm. To address this, I intend to investigate the current audio tempo analysis and beat tracking algorithms available solely for the purpose of being implemented within an drum tutoring software package for the use with acoustic drums.

Figure 1: Example of Roland DT-1 V-Drums tutor GUI



1.1 Beat Detection Background

Detecting musical time is a skill which is not only a fundamental to musicians [1] but also something that seemingly comes naturally to humans, the majority being able to analyse and reproduce discrete metrical structures of a piece of music [2]. Producing algorithms to replicate this nature human ability is probably first attempted by Longuet-Higgins [1], where he began to consider that rhythm as a binary tree with each node representing a note or rest. This theory developed into a system which would use a static tolerance limit on how much the downbeats varied, enabling the perceived tempo to be adjusted accordingly [add longuet references]. Since Longuet-Higgins first work there have been a number of different approaches to beat detection in audio, M. Goto and Y. Muraoka created a system which would learn the frequencies of the bass drum and snare drum, in order to then detect events triggered by these instruments during a piece of music [Goto & muroaka]. In 2001 Simon Dixon presented a system which processed a piece of audio in order to generate a tempo hypotheses at various metrical levels, multiple agents were then employed to find the sequence of beat times which best matched the salient rhythmic events which were originally detected.

Due to the rapidly expanding research being carried out on beat detection, in 2005 the 1st annual Music Information Retrieval Evaluation eXchange (MIREX) was held. MIREX has been

set up as a contest with the goal of comparing state-of-the-art algorithms and music information retrieval [MIREX website]. The topics to be evaluated were proposed by the participants and in the first year, three of the nine topics concerned beat detection (Audio Drum Detection, Audio Onset Detection and Audio Tempo Extraction). To date, the majority beat detection software has focused mainly on the dj industry where tempo and beat matching are popular aspects to be included in modern dj programs. There are however other sectors of the music industry would find beat detection software a useful, in particular, drummers could find it an enormously useful training tool. There are currently many systems available to a drummer,

1.1.1 Onset Detection

The majority of the audio beat detection systems developed to date rely on an onset detection function, where an audio onset is considered to be the beginning of each music note within a piece of music [wiki onset] and therefore can be used to find time-locations within all sonic events in a piece of music [MIREX].

1.1.2 Short-Term Fourier Transform

The Short-Term Fourier Transform (STFT) is based on the original Fourier Transform (FT) which was developed by Joseph Fourier in 1822, which can be used to find out how much of each frequency exists in a signal. The STFT was developed in order to solve provide a solution to the fact that the Fourier Transform does not work on non-stationary signals which results in it not being possible to used to find out when a frequency exists in the time-domain [polikar 2]. The STFT solved this issue by dividing the signal into a number smaller segments using a window function, which effectively created a series of stationary signals which the Fourier Transform could be successfully used on. This however did not fully solve the problem as the size of window function used had an effect on the the quality of frequency resolution and time resolution: [polikar3]

- Narrow Window Function =, Good Time Resolution, Bad Frequency Resolution
- Wide Window Function =, Bad Time Resolution, Good Frequency Resolution

This was eventually solved by development of Wavelet Transforms which are capable of providing time and frequency information simultaneously in order to give a time-frequency representation of a signal, which are described in further detail in the next section.

1.2 Audio Analysis using the Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) is a form of wavelet transform which foundations go back to 1976 when Crosier, Estaban and Galand devised a technique to decompose discrete time signals [polikar]. The DWT uses digital filtering techniques like the original Continuous Wavelet Transform, however it is significantly easier to implement. The DWT works by passing the signal through a series of high pass filters to analyse the high frequencies and low pass filters to do the same to low frequencies. The low pass filtering results in the resolution being halved but leaves the scale unchanged and the high filtering halves the time-resolution but doubles the frequency resolution, which effectively reduces the uncertainty in the frequency by half [polikar 4]. Due to this, the DWT is considered to share similar time-frequency resolution characteristics with the human ear [DWT Tzanetakis et al].

Tzanetakis, Essl and Cook described how the DWT could be used to extract information from non-speech audio, their beat detection algorithm was based on detecting the salient periodicities of the audio signal and was made up of the following steps[tzanetkis].

1. Signal decomposed into a number of octave frequency bands using DWT

2. The subsequent time domain amplitude envelope is extracted for frequency each using low pass filtering, full wave rectification and downsampling
3. Each band is then normalised
4. The envelopes of each band are then summed together and an autocorrelation function is computed

2 Aims and Objectives

As part of my project I propose to build a real time drumbeat tempo analyser, this will use a combination of existing libraries in order to provide a basis for the creation of a live drumming tempo training tool. I will elaborate on the key features of this work in the following sections.

2.1 Core Project Features

- Live audio capturing and processing
- Adapt Eng Eder de Souza Matlab implementation of the beat tracking algorithm described by Tzanetkis et al
- Log detailed comparative information regarding the efficiency and accuracy of the Beatroot and DWT implementation using JWAVE
- GUI to provide the user with current tempo played and notification of beat detection
- Build and test an extensive sample set of drum beats at varying tempos and velocities in order to ensure the system is tested against as close to a real drummer as possible

2.2 Non-Core Project Features

- Live tempo status bar created and tested with a live player to demonstrate the systems ability to handle varying tempos
- Create a calibration mode within the system in order to allow for the system to learn the relevant frequencies of all of the drums being played
- Extend GUI to provide the user with real time notification on when a certain drum is played by using the stored calibrated frequencies

3 Development Plan for the Solution

The main part of this project involves building a beat detection system in order to provide temporal analysis of the live audio signal. To do this I will use the Short-Term Fourier Transform based Beatroot and the algorithm described by Tzanetkis in [paper ref] which will use JWAVE for the Discrete Wavelet Transform.

3.1 Beatroot

Beatroot is an audio beat tracking system which was first presented by Simon Dixon in the 2002 paper “On the analysis of musical expression in audio signals”. The system is described as “an off-line beat tracking system which finds the times of musical beats and tracks changes in tempo throughout a performance” [dixon 2002]. Beatroot uses two interacting processes in order to process

audio; 1) Tempo induction calculated from the rate of detected beats and 2) Beat tracking produced by the synchronising the pulse sequences detected within the audio signal. These processes are performed using the following steps:

1. The audio signal is first passed through a spectral flux onset detection function which uses the short-term Fourier transform (further detailed provided in the following section).
2. A clustering algorithm is then employed to find all of the significant metrical units within the signal, these clusters are then compared and ranked accordingly to produce a set of tempo hypotheses.
3. A multiple agent architecture is utilised to match the sequences of detected beats, where each agent represents a specific tempo value and alignment of beats with the audio signal.
4. Finally, the agents are then evaluated against the original hypothesised beats and the highest ranked sequence is then returned as the solution.

3.1.1 JWave and Beat Detection algorithm

JWave is a Java implementation by Christian Scheiblich which provides a number different transforms including the Haar wavelets, this first DWT invented by Alfred Haar and the Daubechies wavelets which are the more commonly used discrete wavelet transforms[wiki]. JWave only provides the DWT component needed for this project, therefore the beat detection algorithm described by Tzanetkis et al will need to be implemented which will be adapted from the Matlab implementation by Eng Eder de Souza.

3.2 Live Audio Processing

In order to capture and process live audio the Javax Sound package will be utilised, where the audio will be captured using the Java Line hierarchy; specifically the TargetDataLine class, which requires an instance of the Javax Sound AudioFormat class to be parsed in to work. The AudioFormat class is made up of a number of constructed fields which are defined as follows:

- Encoding - This will be set to "PCM.signed", representing audio encoded to the native linear pulse code modulation, where quantization levels are linearly uniform [wiki].
- Sample Rate - 44,100, set to match CD quality for the number of analogs samples which will be analysed per second.
- Sample Size in Bits - 24, based on a sound card with a 24 bit sample depth.
- Channels - 2, audio will be captured using a stereo microphone.
- Frame Size - 6, where frame size equals the number of bytes in a sample multiplied by the number of channels.
- Frame Rate - 44,100, same as sample rate.
- Big Endian (boolean) - false, as the project will be developed on an Intel core which uses a little endian architecture. Where endianness refers to the order of bytes which make up a digital word, with big endianness storing the most significant byte at a certain memory address and the remaining bytes being stored in the following higher memory addresses. The little-endian format reverses the order storing the least significant at the lowest and most significant at the highest memory address [<https://en.wikipedia.org/wiki/Endianness>].

3.3 Comparative analysis of Beatroot and JWave

A key part of this project is to ascertain whether the technologies currently available are accurate enough to create a training tool similar to the midi based software described in section 1. To achieve this the 2 libraries will need to be run in parallel with the same captured audio and a predetermined set of parameters recorded. For the tempo analysis this will simply be the bpm recorded at the end of each frame, all of the bpm's within the sample set will be exact and in order to match the current midi software packages accuracy any returned bpm's should be exact. However, very small deviations in tempo can be very hard to detect to the human ear so some margin of error will be applied. To account for the fact that at the slower bpm's any beat out of place will be a lot easier to notice than at a higher bpm, the margin for error will be weighted by using the following equation;

$$m = bpm - 60 * 0.0001$$

where m represents the weighted value for the margin for error to be applied.

3.4 Develop GUI & logging system

3.5 Build Extensive drum sample set

The success of this project will rely heavily on the amount of data that the system can produce and in order to do this a large set of drum beat samples will need to be created. The drum beats themselves will be created using the installed midi drumbeats with the Apple software package, Garageband. The sample set will need to contain beats from a variety of styles as well as samples with fluctuating tempos. The main aspects of the sample set can be found below:

- The tempo range will be 60 - 160 beats per minute (bpm), although some samples will be created with varying tempos in order to closely reflect a real human player
- The time signatures will be mainly in common time (4/4) to reflect playing styles, however some beats will use swing time which will either be created by feel within the sample or by using triplets with a duple meter in 12/8
- Velocities will be varied across the sample set, accent and ghost notes will be used in line with a real playing style
- The sample set will contain a wide range of drum beats, including; beats using one drum, those only with a back beat, beats incorporating single rests within a bar or for the whole duration of a bar & beats with large solos and cymbal work

Musical styles will comprise of rock & pop, blues and jazz, with the main emphasis falling on rock & pop beats. Each style will have between 5 - 10 unique samples created, which will in turn each be sampled with a bpm value starting at 60 and increasing in steps of 5 up until the maximum, which will account for approximately 500 separate samples. There will also be a final set of simple single and multiple drum beats included which will produce another c. 10 unique samples, which will give a total sample set size of c. 750 samples.

4 Project Schedule

The start date for the project is Monday 13th June 2016 and the end date is Monday 19th September 2016, the non-core project features will only be completed if the project is ahead of schedule.

Table 1: Project Timeline

Dates	Task
w/c 13th June 2016	Develop live audio capturing system
w/c 20th June 2016	Set up Beatroot library in development environment
w/c 27th June 2016	Implement DWT tempo and beat detection algorithm in Scala/Java
w/c 4th July 2016	Test systems with single live audio source
w/c 11th July 2016	Develop system to store data generated during tempo analysis
w/c 18th July 2016	Research and build appropriate framework to process captured audio in parallel e.g. akka actors
w/c 1st August 2016	Create sample set of drum beats
w/c 8th August 2016	Test system using full sample set
w/c 15th August 2016	Develop UI
w/c 22nd August 2016	Analyse logged data
w/c 29th August 2016	Write up report
w/c 5th September 2016	Present findings to project supervisor
w/c 12th September 2016	Finalise report
w/c 19th September 2016	Submit report

5 References

<http://www.roland.co.uk/blog/exploring-roland-dt-1-v-drums-tutor-software/>
https://en.wikipedia.org/wiki/Pulse-code_modulation
http://www.jsresources.org/faq_audio.html#frame_rate
[https://en.wikipedia.org/wiki/Onset_\(audio\)](https://en.wikipedia.org/wiki/Onset_(audio))
http://www.music-ir.org/mirex/wiki/2016:Audio_Onset_Detection
<http://users.rowan.edu/~polikar/WAVELETS/WTpart2.html>
https://en.wikipedia.org/wiki/Discrete_wavelet_transform