Real Time Tempo Analysis of Drum Beats

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

Birkbeck, University of London
Department of Computer Science and Information Systems

Project Proposal
MSc Computer Science

April, 2016

Contents

Abbreviations

BPM	Beats Per Minute
DWT	Discrete Wavelet Transform
FT	Fourier Transform
JSON	Javacript Object Notification
IDE	Integrated Development Environment
STFT	Short-Time Fourier Transform
TDD	Test Driven Development

Definitions

Acoustic Drum Kit	A collection of drums and cymbals which do not have electronic amplification. Typically made up of a bass drum, snare drum, tom-toms, hi-hat and 1 or more cymbals.
Beat	For the purpose of this project a beat will be defined as the sequence of equally spaced pulses used to calculate the tempo being played by the drummer.
Downbeat	Refers to beat one of a measure of music, called a downbeat to correspond to the motion a conductor's arm.
Drum Module	The device which serves as a central processing unit for an electronic drum kit, responsible for producing the sounds of the drum kit.
Electronic Drum Kit	An electrical device which is played like an acoustic drum kit, producing sounds from a stored library of instruments and samples.
Bar/Measure	A measure or bar is a segment of time made up by a predetermined number of beats, for example a piece of music with a 4/4 time signature will have 4 beats in every measure/bar.
MIDI	Musical Instrument Digital Interface is a protocol developed in the 1980's to allow electronic instruments and other digital musical tools to communicate with each other [22].
Time Signature	Used in musical notation to represent the number of beats in a measure or bar of music.

1 Introduction and Background

Having played the drums for nearly twenty years, I have used a number of different training tools to help improve my timing. Until recently, these tools have been exclusive to MIDI driven electronic drum kits, which are able to include training tools within their drum modules. These tools use the events being triggered by the drummer to give live feedback on the exact timing and part of the drum kit 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 drummer with an interactive experience that helps improve their rhythm, coordination and sight reading [1]. 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). These tutoring tools are not however available to a drummer who does not own an electronic drum kit. Such a drummer is restricted to using a metronome[1] and his/her ear to determine his/her timing and rhythm. To address this, I intend to investigate two of the available audio tempo analysis and beat detection algorithms to ascertain if they are suitable to be implemented in a piece drum tutoring software for acoustic drums.

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

1.1 Beat Detection Background

Detecting musical time is a skill which is not only fundamental to musicians [2] but also something that seemingly comes naturally to humans. The majority are able to analyse and reproduce the metre[2], tempo and rhythmic aspects of a piece of music [3]. Longuet-Higgins [2] was one of the first to produce an algorithm to replicate this human ability. He constructed a binary tree with each node representing a note or rest [4]. He then developed his theory into a system that was able to measure the variations in the downbeats of a piece of music and adjust the perceived tempo according to whether the note was later or earlier than expected [2]. Since Longuet-Higgins' first work there have been a number of different approaches to beat detection. M. Goto and Y. Muraoka developed a system which learned the frequencies of the bass drum and snare drum, in order to detect events triggered by these instruments during a piece of music [5]. Simon Dixon [6] presented Beatroot, an interactive beat tracking and visualisation system which is able to estimate the tempo and times of musical beats in performed music [6]. In the same year Tzanetakis, Essl and Cook [12] described an algorithm based on the Discrete Wavelet Transform which is capable of detecting the beat attributes of music.

[1] Device which produces a regular metrical tick or beat which is settable in beats per minute
[2] Metre is the repeating pattern that provides the pulse of beat of a piece of music

1.1.1 Beatroot

Beatroot is an audio beat detection system presented by Simon Dixon in 2001 and is described as a "beat tracking system which finds the times of musical beats and tracks changes in tempo throughout a performance" [13]. Beatroot works by first processing digital audio data to produce a list of onset (see figure 2) or event times. The time intervals between these events are then analysed to generate tempo hypotheses concerning the rate and location of beats. Using the tempo hypotheses, searches are carried out to test the different hypotheses about the rate and timing of beats. The results of these searches are ranked and the beat times found in the highest ranked search are returned [27].

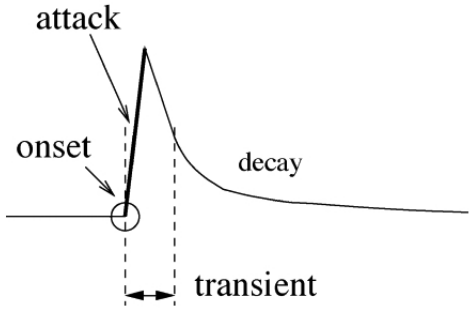

Figure 2: The onset of a note is the instant which marks start of the variation in the frequency of a signal [25]

1.1.2 Short-Time Fourier Transform

To detect the onsets of a piece of digital audio Beatroot uses an onset detection function based on the Short-Time Fourier Transform (STFT). The STFT is a form of 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. A drawback of the FT is that it is unable to provide any details of when a frequency component occurs in time for non-stationary signals[3]. A solution to this was to split a non-stationary signal up into a number of smaller segments using a window function, which effectively created a series of stationary[4] signals which the original FT could then be successfully applied to. However, this did not fully solve the problem as the size of window function affects the quality of frequency resolution and time resolution:

- Narrow Window Function $\longrightarrow$ Good Time Resolution, Bad Frequency Resolution
- Wide Window Function $\longrightarrow$ Bad Time Resolution, Good Frequency Resolution [10]

1.1.3 Discrete Wavelet Transform

The first literature regarding the wavelet was provided by the mathematician Albert Haar in 1909 [26]. The wavelet transform is a technique for analysing signals which was developed as an alternative to the STFT [12]. Like the STFT the Discrete Wavelet Transform is able to provide time and frequency information. However, it is able to do this without a window function and unlike the STFT which is only able to provide uniform time resolution for all frequencies where as the DWT provides time resolution and frequency resolution as follows [12]:

[3]Non-stationary signals are signals whose frequency contents changes over time [10].
[4]The frequency contents of a stationary signal does not change over time

	Frequency Resolution	Time Resolution
High Frequencies	Poor	Good
Low Frequencies	Good	Poor

1.1.4 Tzanetakis, Essl and Cook Beat Detection Method

In 2001, Tzanetakis *et al* described how the Discrete Wavelet Transform (DWT) could be used to extract information from non-speech audio [12]. Their beat detection algorithm was based on detecting the most prominent signals which are repeated over a period of time within the analysed audio. The algorithm itself was split into the following steps:

1. Signal decomposed into a number of octave frequency bands using the DWT
2. The subsequent time domain information is extracted for each frequency band
3. The data from each band are then summed together and a function to find repeating patterns is applied

As there is no current open source Java implementation of this algorithm. I will attempt to implement a version of this myself, which will be the DWT based beat detection component of this project.

2 Aims and Objectives

For my project I propose to build a real time drum beat tempo analyser. This will use the Beatroot system described previously and an implementation of the Tzanetakis *et al's* beat detection algorithm [12] as 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
- Implement the beat detection algorithm described by Tzanetakis *et al* [12]
- Build a system to record detailed comparative information regarding the efficiency and accuracy of Beatroot and Tzanetkis, Essl and Cook's algorithm in order to ascertain if they are accurate enough to be used in a training tool for drummers.
- Design a User Interface (GUI) to provide real time feedback on the current tempo played and beats detected
- An extensive sample set of drum beats will need to be created to ensure the system is tested sufficiently.

2.2 Non-Core Project Features

- Investigate if it is possible for the system to learn the frequencies of the different drums being played
- Extend GUI to provide the user with real time notification on when a certain drum is played

3 Development Plan for the Solution

The first part of my project involves the designing my own implementation of the beat detection algorithm described by Tzanetakis *et al* [12]. This will then be incorporated in a beat detection system along with the Beatroot system in order to provide real time tempo analysis of an audio signal.

3.1 Proposed Architecture

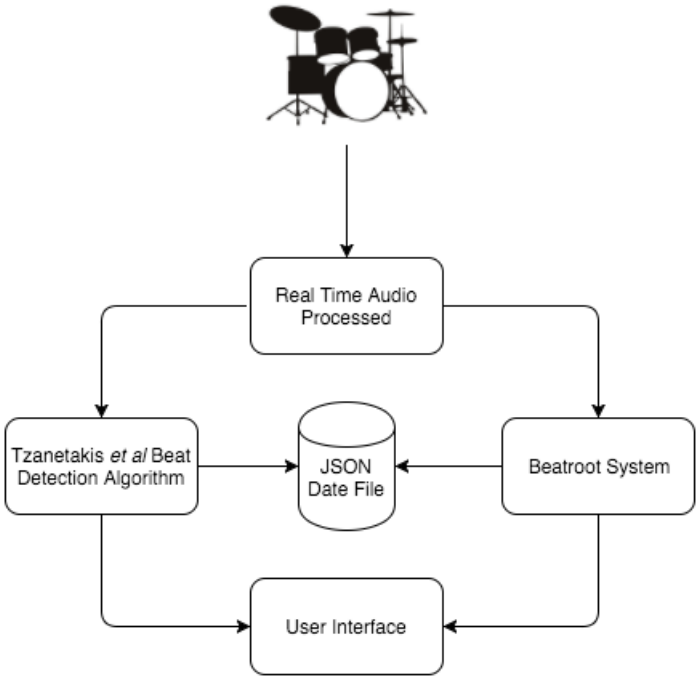

Figure 3:

3.2 Live Audio Processing

The live audio will be processed using the Javax Sound package. The audio will be captured using a stereo microphone and processed to match CD quality with the Javax Sound AudioFormat class. The Beatroot system was not originally intended to be used as a real time system [29] so currently only works with prerecorded audio. It will therefore be necessary to modified the audio input it accepts in order for it to work with live audio.

3.3 Implementation of Tzanetakis *et al* Beat Detection Algorithm

As a there is a Matlab implementation which has been created by Eng Eder de Souza [20]. The implementation of Tzanetakis *et al* [12] beat detection algorithm will require the use of a library which provides DWT functionality. This will be provided by JWave.

JWave is a Java library created by Christian Scheiblich which provides a number of different transform packages, including a wavelet package. For the purpose of this project the Daubechies wavelets classes within JWave will be used. The Daubechies wavelets are a form of discrete wavelet transform which were developed by Ingrid Daubechies in the late 1980's and are frequently used in applications [27].

3.4 Comparative Analysis of Beat Detection Algorithms

A key part of this project is to ascertain whether the two beat detection methods described in section 1 are accurate enough to create a training tool similar to the MIDI based software seen in figure 1. To achieve this the two systems will be run in parallel using real time audio and return details on the calculated tempo in beats per minute (bpm), the number of beats detected and the time it took to calculate the results.

The data collected during this process will be recorded and stored in an appropriate format to allow it to be queried and analysed. Due to the relatively small amount of data which will be collected it seems excessive to create a database to do this. Instead the data will be stored in a JSON file. JSON was chosen over XML as it has a much simpler grammar and is able to map directly onto the data structures used in modern programming languages [21].

3.5 Build Extensive Drum Sample Set

The success of this project will rely heavily on the amount of data that the system produces. A large set of drum beat samples will be required. The drum beats will be created using the Apple software package, Garageband. The sample set will contain beats from a variety of styles as well as samples with fluctuating tempos. The main aspects of the sample set are as follows:

- The tempo range will be 60 - 160 beats per minute. Some samples will be be created with varying tempos in order to model human playing more closely
- The time signatures will be mainly in common time (4/4) to reflect normal playing styles, however some drum beats will use swing notes[5] which will either be created by feel[6]
- Accenting[7], ghost notes[8] and missed beats[9] will be added 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 and beats with large solos and cymbal work

Musical styles will comprise rock and pop, blues and jazz, with the main emphasis falling on rock and pop beats. Between 5 - 10 unique samples created for each style, which will be sampled with a range of bpms. Each sample will have approximately 20 different bpm values, accounting 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 result in a total sample set size of approximately 750 samples.

3.6 User Interface

The user interface will be developed using the JavaFX framework, to enable that the system will be cross-platform compatible. Its features will be kept simple initially as it will only be required to provide a real time output of the current two tempo calculations and accumulative count of beats detected.

[5] A swing note is style where notes with equal time are played with an unequal durations, usually as alternating long and short durations [wiki]

[6] By adapting the strength a drum or cymbal is being played at it is possible to infer a different style of music using feel, e.g. swing time can be created this way.

[7] Accents are an emphasis placed on a specific note

[8] Ghost notes are a note which is played at a very low volume

[9] Missed beats will be used to represent the drummer missing a drum or cymbal

3.7 Non-Core Project Features

The non-core project features will be included only if the project is developed ahead of the schedule described in section 4.

3.7.1 Frequency Learning

This non-core feature will investigate if it is possible to create a similar system to the one developed by Goto and Muraoka, which learnt the frequencies of the snare and kick drum in order to detect events triggered when these instruments were played [5]. The goal will be to enable the system to learn the frequencies of all of the separate parts of a drum kit (snare drum, bass drum, tom-tom, hi-hat and cymbals) being played and use this event to extend the GUI.

3.7.2 Extended User Interface

Developing this additional feature will only be possible if the previous non-core feature (3.7.1) is successfully developed. If so, the user interface will be extended to look similar to the drum kit visual seen in figure 1, where a colour indicator will be displayed to represent which instrument was last played. The development of this final feature depends heavily on the accuracy and efficiency of the tempo analysis algorithms used.

3.8 Project Methodology

This project will be developed using the Test Driven Development (TDD) framework. TDD is a development process that combines the test-first development and refactoring. Test-first development involves writing a test before just enough production code is written to fulfill that test [30]. Refactoring is the process of making small changes to code in order to improve its design, making it easier to understand and modify [31]. I intend to use TDD due to the relatively short development time of this project. TDD will enable small steps to be taken during the process which is considered to be a more productive approach to writing software [30].

3.9 Development Languages and Testing

The project will be developed in a combination of Java and Scala. Java is required as both the Beatroot and JWave libraries are in this language, however any functional aspects of the project will be developed in Scala, as it offers a superior functional tool set when compared to those offered in Java 8. As previously discussed JSON will be used to store the collected data.

I will use the Intellij IDE to develop this system. Git will be used for version control, testing will be completed using JUnit for Java, and Scala Test for any functionality developed in Scala. In order to ensure that the parsed JSON is well formed it will be tested using the JSON validator found at - http://jsonlint.com/.

4 Challenges, Probability, Impact and Mitigation

Table 1: Key Project Challenges and Mitigation

Challenge	**Mitigation**	**Probability**	**Impact**
Own implementation of Tzanentkis, Essl and Cook's [12] beat detection algorithm takes longer than expected	Adapt the Eng Eder de Souza Matlab implementation instead	Medium	High
Beatroot does not configure to work with real time audio	Find and use an alternative	Low	High
The two beat detection systems do not integrate well in a concurrent system	Create a two separate systems	Low	High
Not enough time to develop the GUI	Develop simple command line based user interface	Low	Low

5 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 2: 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 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 2026	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

6 Summary

I propose to build a real-time drum beat tempo analysis system using the Short-Time Fourier Transform based Beatroot library and an implementation of the Tzanetkis *et al* [12] beat detection algorithm. An extensive set of drum samples will then processed through the system to represent live audio and the results recorded. These results be analysed to determine if the two beat detection methods are suitable to be implemented in a real time training tool for drummers.

7 References

1. http://www.instructables.com/id/What-is-MIDI/
2. http://www.roland.co.uk/blog/exploring-roland-dt-1-v-drums-tutor-software/
3. Allen and Dannenberg (1990), Tracking Musical Beats in Real Time, International Computer Music Conference, International Computer Music Association, pp. 140-143
4. P. Desain and H. Honing (1989), The Quantization of Musical Time: A Connectionist Approach, Computer Music Journal, Vol. 13 No. 3 (Autumn), pp. 56-66
5. H. C Longuet-Higgins (1976), Perception of melodies, Nature Vol. 263, pp. 646-653
6. M. Goto and Y. Muraoka (1994), A beat tracking system for acoustic signals of music, in Proceedings of the Second ACM International Conference on Multimedia, pp. 365-372
7. S. Dixon (2001), Automatic Extraction of Tempo and Beat from Expressive Performances, Journal of New Music Research, 30 (1), pp. 39-58.
8. G. Tzanetakis, G. Essl and P. Cook (2001), Audio Analysis using the Discrete Wavelet Transform, Proc. WSES International Conference on Acoustics and Music: Theory and Applications (AMTA)
9. S. Dixon (2003), On the Analysis of Musical Expression in Audio Signals, Storage and Retrieval for Media Databases, SPIE-IS&T Electronic Imaging, SPIE Vol. 5021, pp. 122-132
10. Andreas de Vries (2006), Wavelets
11. http://www.nyu.edu/classes/bello/MIR_files/2005_BelloEtAl_IEEE_TSALP.pdf
12. http://users.rowan.edu/ polikar/WAVELETS/WTpart2.html
13. Gao, Robert X, Yan, Ruqiang (2011), Wavelets, Theory and Applications for Manufacturing
14. dixon 2007
15. https://github.com/ederwander/Beat-Track
16. http://www.json.org/xml.html
17.
18.
19. http://www.music-ir.org/mirex/wiki/2005:Main_Page
20. https://en.wikipedia.org/wiki/Onset_(audio)

21. http://www.music-ir.org/mirex/wiki/2016:Audio_Onset_Detection

22. http://users.rowan.edu/ polikar/WAVELETS/WTpart4.html

23. https://en.wikipedia.org/wiki/Discrete_wavelet_transform

24. https://en.wikipedia.org/wiki/Pulse-code_modulation

25. https://en.wikipedia.org/wiki/Endianness

26. http://www.jsresources.org/faq_audio.html

27. https://en.wikipedia.org/wiki/Test-driven_development

28. A. Robertson, A. Stark and M. E. P. Davies (2013), Percussive Beat tracking using real-time median filtering, 6th International Workshop on Machine Learning and Music

29. http://disp.ee.ntu.edu.tw/tutorial/WaveletTutorial.pdf

30. http://www.ieor.berkeley.edu/ ieor170/sp15/files/Intro-to_Sonic_Events_Campion.pdf

31. Simon Dixon (2001), An Interactive Beat Tracking and Visualisation System

32. Graham Percival, Member, IEEE, and George Tzanetakis (2014)

33. http://dictionary.onmusic.org/appendix/topics/meters