AUTOMATIC BASS DRUM PEDAL

A Project Report
Submitted in partial fulfillment of
the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

by

Anina Saji P	B160322EC
Arun Ganapathy	B160031EC
Clare Chris Stephen	B160805EC
Neelima Vijayan	B160157EC
S Mohamed Rayaan	B160019EC

Under the guidance of
Prof. Lyla B.Das

Department of Electronics and Communication Engineering
NATIONAL INSTITUTE OF TECHNOLOGY CALICUT
Calicut, Kerala, India – 673 601

2019-2020

Acknowledgments

We would like to share our sincere gratitude to all those who helped us in the progress of our project thus far. We hereby thank our Project Guide, Prof. Lyla B. Das, for her invaluable guidance, comments and suggestions throughout the course of the project. We would also like to show our gratitude to the B.Tech Project Evaluation Committee for their continuous help and monitoring during the project work. All of our team is thankful to the Department of Electronics and Communication Engineering and the National Institute of Technology Calicut, for providing us with this opportunity to learn from these experiences. We are also thankful to all of our classmates and most of all our parents who have inspired us to face all the challenges and overcome them all.

Anina Saji P
Arun Ganapathy
Clare Chris Stephen
Neelima Vijayan
S Mohamed Rayaan

June 2020
National Institute of Technology Calicut

Declaration

We hereby declare that except where specific reference is made to the work of others, the contents of this project report are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This project report is our own work and does not contain any outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Anina Saji P (B160322EC)

Arun Ganapathy (B160031EC)

Clare Chris Stephen (B160805EC)

Neelima Vijayan (B160157EC)

S Mohamed Rayaan (B160019EC)

NIT Calicut
Date: 28.06.2020

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Certificate

This is to certify that the project report entitled **Automatic Bass Drum Pedal** submitted by **Anina Saji P (B160322EC), Arun Ganapathy (B160031EC), Clare Chris Stephen (B160805EC), Neelima Vijayan (B160157EC), S Mohamed Rayaan (B160019EC)** to National Institute of Technology Calicut for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering, is a bonafide record of the project work carried out by them under my supervision and guidance. The content of the project report, in full or parts have not been submitted to any other institute or university for the award of any degree or diploma.

Prof. Lyla B.Das
(Project Guide)
Dept. of Electronics and
Communication Engineering
NIT Calicut

Dr. P. P. Deepthi
(Professor and Head of Department)
Dept. of Electronics and
Communication Engineering
NIT Calicut

Date: 28.06.2020

(Office seal)

Abstract

This paper aims to design and implement an Automatic Bass Drum Pedal with the help of an embedded board which will aid musicians to keep time and rhythm in accordance with a preset time signature, an input pattern taken from the user at the time or a pattern from a given sound track. An Automatic Bass Drum Pedal, as the name suggests, consists of a bass drum and its pedal which is mode controlled not only to aid musicians during practice, recordings and live performances, but also provides specific advantages to drummers. An Automated Bass Drum Pedal, with the help of an embedded board, will aid musicians to keep time and rhythm in:

- Mode One: Accordance with a preset time signature
- Mode Two: An input pattern taken from the user at the time
- Mode Three: A library with a list of songs for which the bass drum patterns have already been fixed.
- Mode Four: The fingerprint of a sample recorded using microphone is compared with existing database to recognize the song and its bass pattern is generated and given to the pedal.

Contents

List of Figures

Chapter 1

Introduction

Being a form of art and cultural activity, Music can be defined as a sound placed in time. The main elements of a song include pitch, rhythm and dynamics. Technology is an absolute need we cannot escape from. The music industry has undergone swift and dramatic changes over the years due to various advances in technology.

It is not a surprise that technology has had a tremendous effect on our access and approach towards music. To date, many musicians believe that the major effect that today's technology has had on music production, is that it has enabled artists to do more with just a handful.

In this project, we will be focusing on the concept of rhythm. The only way to produce music that would sound full is to have all the instruments compliment the vocals. It can be said without any hesitation that the drummer/drums is an integral part that majorly contributes to this notion. Today, loop pedals and digital audio interfaces enable musicians to replicate/record instruments and track/play over the recorded tracks. Thus it provides solo artists and bands with the option to achieve satisfactory results even when they don't have all the band members. This will help make it easier and cheaper for them to record and to perform.

However, looping a recorded snip of an instrument or playing along with it, does not quite give the musician a natural feeling. One of the very broad objectives of our project is to implement the same feeling by avoiding the use of digitally recorded beats or tracks. The socially supportive objective of our project targets handicapped/physically challenged drummers to continue to pursue a career in music.

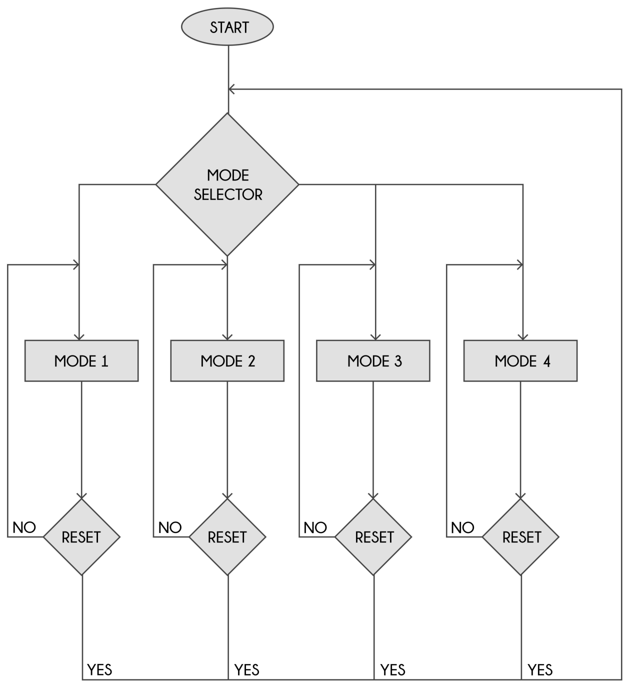


Figure 1.1: Flow Chart of modes.

Chapter 2

Literature Review and Motivation

2.1 Literature Review

"Audio Analysis using the Discrete Wavelet Transform"[1] uses DWT to analyze the spectral and temporal properties of non-stationary signals like audio signals. This paper was referred to as it describes the use of Discrete Wavelet Transform instead of FFT for beat detection.The paper "ECG beat detection using filter banks"[2] detects heart beats in electrocardiogram (ECG). Analysis is done in both time and frequency domains after the ECG is decomposed into frequency bands of equal bandwidth using filter banks.

The paper "An Industrial-Strength Audio Search Algorithm"[3] deals with algorithms implemented in the flexible audio search engine Shazam Entertainment Ltd. The algorithm uses a combinatorial hashed time-frequency constellation analysis of the audio, yielding unusual properties such as transparency, in which multiple tracks mixed together may each be identified. We have taken inspiration from the methodology they used to implement our own fourth mode of operation.

An approximation of an effective search algorithm in "Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces"[4], to identify songs based on 5 second samples promises a tremendous future development for higher song recognition rates. "Robust Sound Modeling for Song Detection in Broadcast Audio"[5] devises an audio fingerprint development algorithm which takes into account parameters such as rhythm and melodic trajectories, into the pattern matching algorithm. The prospects and benefits of a private cloud server set up in a Raspberry Pi is examined in "Implementation of Cloud Server for Real Time Data Storage using Raspberry Pi"[6]. The paper "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures"[7] implements live migration within the PiCloud through a model designed to overcome the limitations of simulation of x86 test-beds. "Proportional control of a solenoid actuator" [8] explores ways for switching solenoids to be controlled proportionally using certain non-trivial strategies. "A Basic Electromagnetic Theory for Controlling Solenoid Actuators"[9] devises theoretical ways to control actuator motion, including current changes and complex inductance reactions in coil motion. These papers help to realize bottlenecks from the early stages itself, and also give an understanding on how to control actuator force.

2.2 Motivation

The music industry is something that revolves around musicians having perfect hand eye coordination and physical capabilities to play instruments. Or so it was until the invention of electronic music. But what this failed to bring out was the essence of true music that was present before electronically reproduced sounds. Another significant detail is that most musicians lose their career in music and are rendered unable to play any instruments for the rest of their life due to an accident or disability that they probably developed later on in their lives. Especially in the case of drummers who exert a tremendous amount of stress on their arms and legs. The automatic bass drum pedal is an attempt to solve the two issues mentioned above. What this ultimately does is, it enables disabled musicians to still create music despite their inability to do so physically and also helps individual artists create the experience of a full band without the necessity of the reproduction of electronic sound. This method can further extended to be used in other instruments as well.

Chapter 3

Hardware Design

3.1 Raspberry Pi 3(Model B)

Raspberry Pi, also known as 'RPi', is a group of small-single board computers that run on a '1.5 GHz 64/32-bit quad-core ARM Cortex-A72' CPU consisting of the 'Broadcom BCM2711' chip. Generally, it comes with a 1GB, 2GB or 4GB 'LPDDR4-3200 RAM', on board Micro SD Card Slot and a 'Broadcom VideoCore VI 500 MHz' Graphic Card.

3.2 Solenoid Motor(6N) with ciruit

The Solenoid Motor put to use is of the push-pull type with a throw length of 1mm. It provides a starting force of 6N (12V DC, 50% duty cycle.) Actuation required for automating the machine requires easy on and off and also very detailed and intricate sequencing.For products that require linear or rotary motion, solenoids are convenient to use based on their price, size, ease of installation and use. The actual force provided by the (solenoid motor) pedal on the bass drum can be calculated as

$$F = (N * I)^2 \mu_0 A/(2g^2) \tag{3.1}$$

3.3 Waveshare Raspberry Pi LCD Display 3.2 LCD-V4

A 3.2" touchscreen monitor, 320 x 240 display connected via an adapter board which is for managing the conversion of signal and power. The Pi communicates with the touchscreen through SPI(Serial Peripheral Interface).

The display requires connections from the Pi which is for power from the GPIO port of the Pi and also a ribbon cable from the Display to the Pi's DSI port.

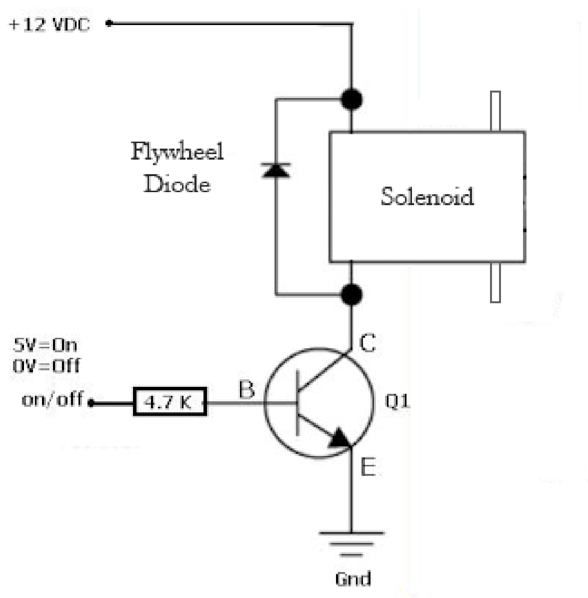


Figure 3.1: Schematic of Solenoid Motor Driver Circuit.

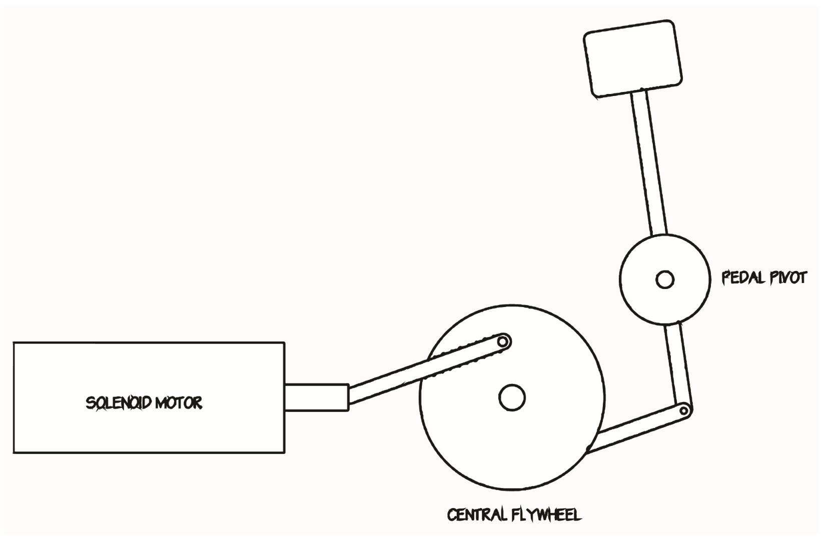


Figure 3.2: Schematic of the Product

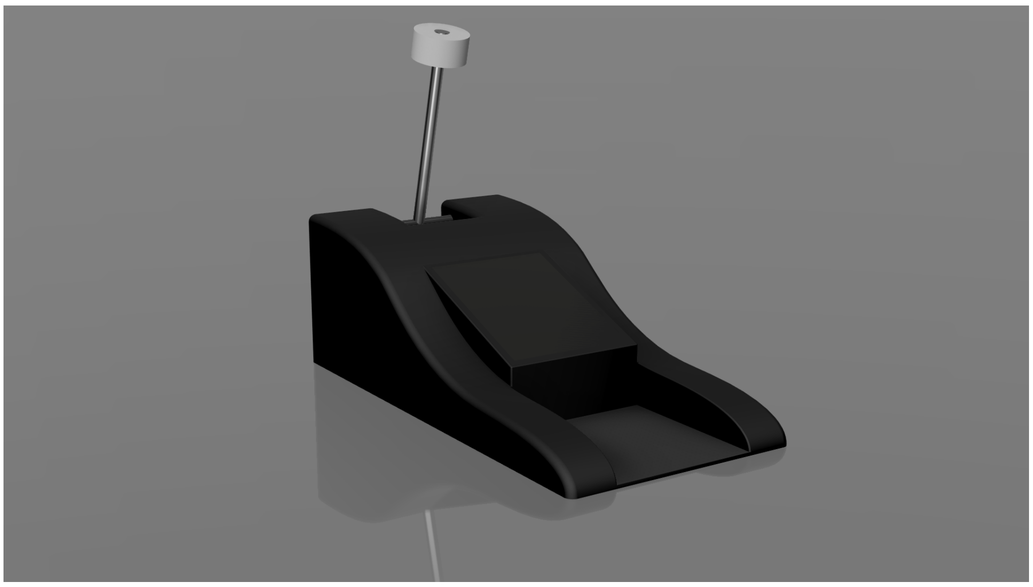


Figure 3.3: Proposed physical model (view 1)

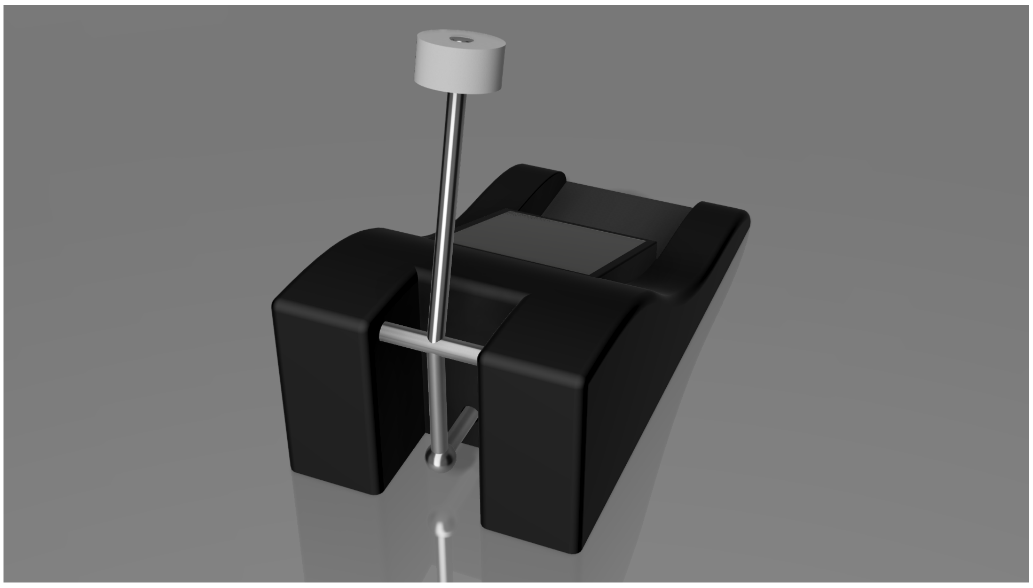


Figure 3.4: Proposed physical model (view 2)

Chapter 4

Softwares Used

4.1 Processing v3.5.3

Processing is an IDE (Integrated Development Environment) with a graphical library available as open source. It is mainly used to understand and represent basic programming making use of graphical aspects. Processing uses JAVA and also provides a graphical user interface for simplifying the compilation and execution.

4.2 MATLAB

MATLAB is a numerical programming language that enables matrix manipulations, graphical representation of functions and data, implementation of algorithms, generation of user interfaces and interfacing with programs coded in other programming languages. We used this platform to process sound input and analyze the constituent frequency bands. Various signal processing steps involved in this are pulse smoothening, differentiating, rectifying and convolution with different comb filters to realize the tempo accurately.

4.3 Proteus Design Suite

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic computer aided design.Proteus is used mainly to make schematics and create digital prints which can then be used for making PCBs (Printed Circuit Boards). We have tried to simulate a sub-woofer active low pass filter, which works on low power audio signal and is intended as a filtering element before a power audio amplifier driving a sub-woofer loudspeaker.

Chapter 5

Modes of Working

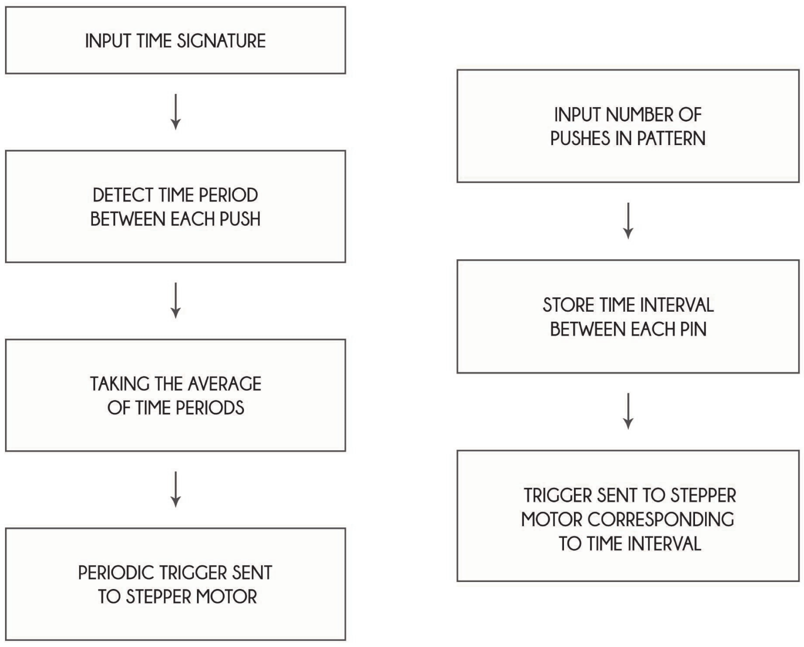
5.1 Mode 1

Aim : The Mode one design was originally supposed to wait for the user to select a particular time signature from a given menu that would include a total of five time signatures with preset accents. Apart from the selection of a time signature, the user was also supposed to tap and define the tempo.

Implementation : Mode 1 of the project is replicating a normal metronome. It works the same as a normal metronome app in your phone or keyboard where you tap the first few beats and it goes on at the same tempo. Here you get the beat on actual drums and the initial input from the user is just as easy i.e a few taps. The coding for this has been done and includes taking the successive time intervals between four taps.
To get the input taps from the user, we've used a HEX Keyboard. The time intervals between the taps are then averaged and the mean time interval is taken and displayed on an LCD Display. Further beats on the drum then go on at the same tempo. This is much better than merely taking one time interval as users will not always be able to input the right tempo in one or two beats. Repeated taps and averaging time interval will help reduce the error. This average time interval is used to move the servo motor. The servo motor had to go forward at each beat and get back to its initial position and then it has to wait i.e a delay of the averaged time that has been found earlier.

5.2 Mode 2

Aim and Implementation: Mode two allows the user to input a pattern which will be repeated as such. For this 2 sets of inputs are taken. Each set of input has a fixed number of button pushes and each of these time intervals will be stored in variables. And the corresponding interval from each set is averaged and the pattern times are fixed accordingly. This provides us with minimal error in the pattern. Coding has been done where two sets of input are taken from the user and the servo motor interfacing is done as well.



INPUT TIME SIGNATURE
DETECT TIME PERIOD BETWEEN EACH PUSH
TAKING THE AVERAGE OF TIME PERIODS
PERIODIC TRIGGER SENT TO STEPPER MOTOR
INPUT NUMBER OF PUSHES IN PATTERN
STORE TIME INTERVAL BETWEEN EACH PIN
TRIGGER SENT TO STEPPER MOTOR CORRESPONDING TO TIME INTERVAL

Figure 5.1: Mode 1 and 2

5.3 Mode 3

Aim and Implementation : Mode 3 presents the user with a list of songs for which the bass drum patterns have already been fixed. However, to provide the user with a more versatile program, we have also induced the mode 1 functionality into this mode, post-the song selection.

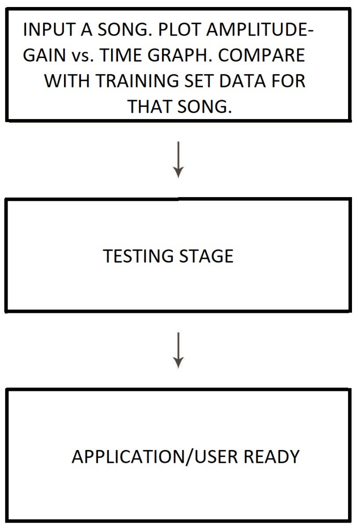

INPUT A SONG. PLOT AMPLITUDE-GAIN vs. TIME GRAPH. COMPARE WITH TRAINING SET DATA FOR THAT SONG.
TESTING STAGE
APPLICATION/USER READY

Figure 5.2: Mode 3

5.4 Mode 4

Aim and Implementation : The fingerprint of a sample recorded using microphone is compared with existing database to recognize the song and its bass pattern is generated and given to the motor.

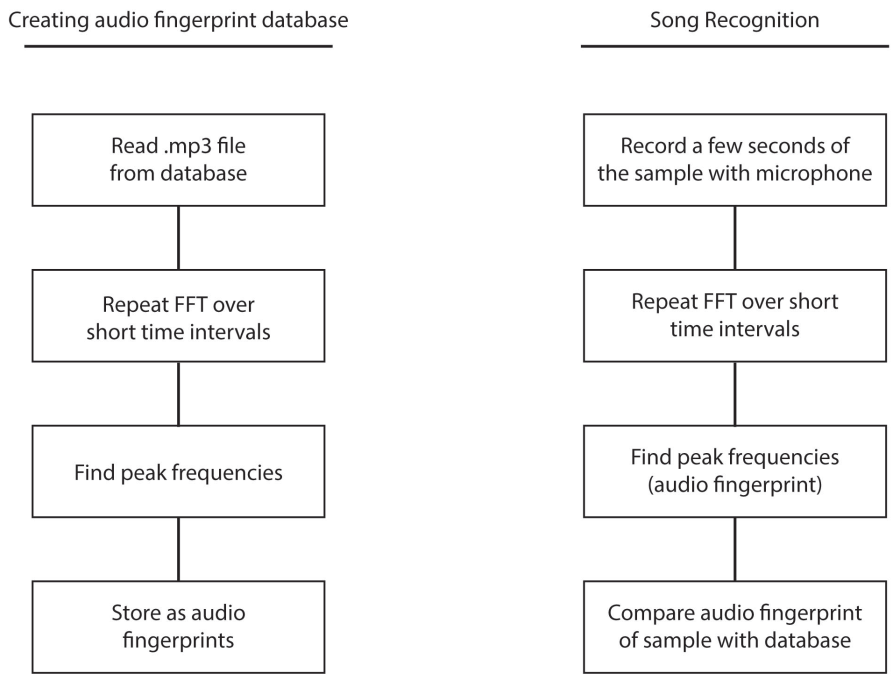

Creating audio fingerprint database
Read .mp3 file from database
Repeat FFT over short time intervals
Find peak frequencies
Store as audio fingerprints
Song Recognition
Record a few seconds of the sample with microphone
Repeat FFT over short time intervals
Find peak frequencies (audio fingerprint)
Compare audio fingerprint of sample with database

Figure 5.3: Mode 4(a) and 4(b)

Chapter 6

Working of Mode 4

6.1 Working of Song Recognition(Software)

- A spectrogram of each song is created using repeated FFT over small windows of time. Hence, we obtain amplitude as a function of time and frequency. (Fig. 6.1)

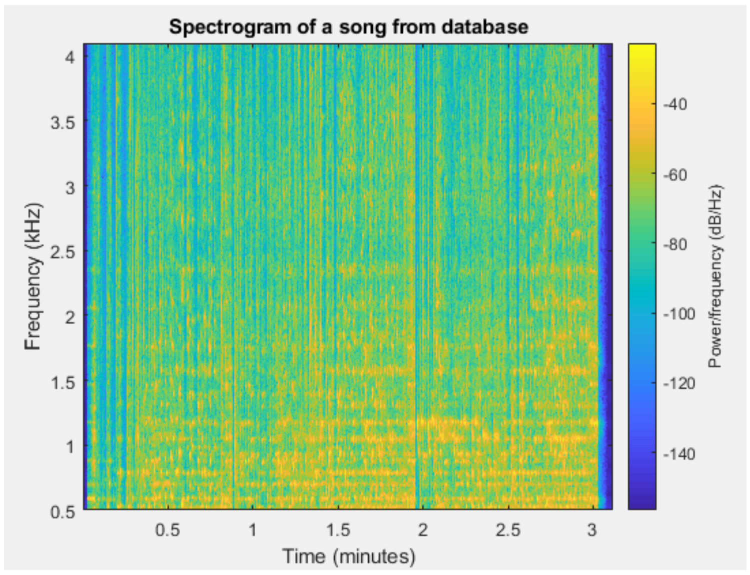


Figure 6.1: Spectrogram of a song from Database

- We find the local maxima for the amplitude values and obtain the frequency-time pairs associated with it. Thus, we have converted the song into a set of points that are relevant for the song and denote the

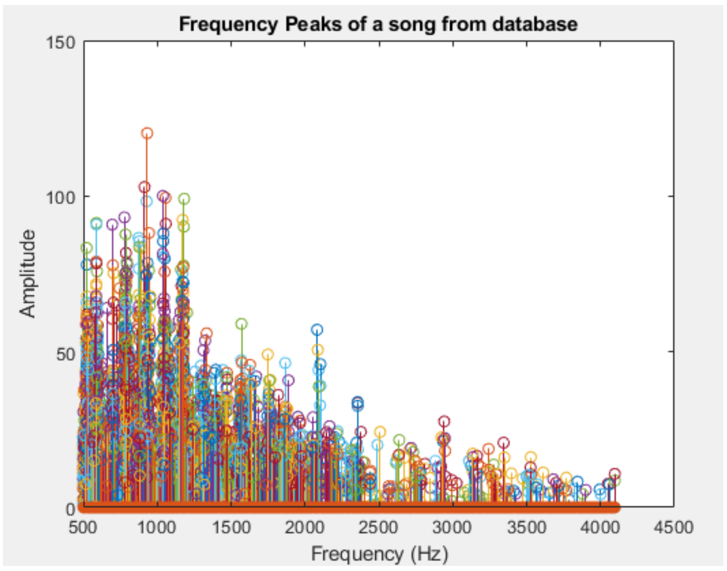


Figure 6.2: Frequency peaks of a song from Database

points of interest in the song. This set of points will help us to identify the song. (Fig. 6.2)

- These points are stored using a hash function which uses a combination of peak frequencies and time as keys.
- Along with the fingerprint of each song, the song details are also stored to show after recognition. This is repeated for every song in the database.
- For recognition, a few seconds of the song to be recognized which is being played out loud is recorded using a microphone and then the fingerprint for this audio clip is generated. (Fig. 6.3 and Fig. 6.4)
- It is compared to the audio fingerprints available in the database and will be recognized, if present.

6.2 Working of Song Recognition(Hardware)

6.2.1 SD Card

- We have a sufficiently sized Micro-SD Card attached to the Raspberry Pi module.

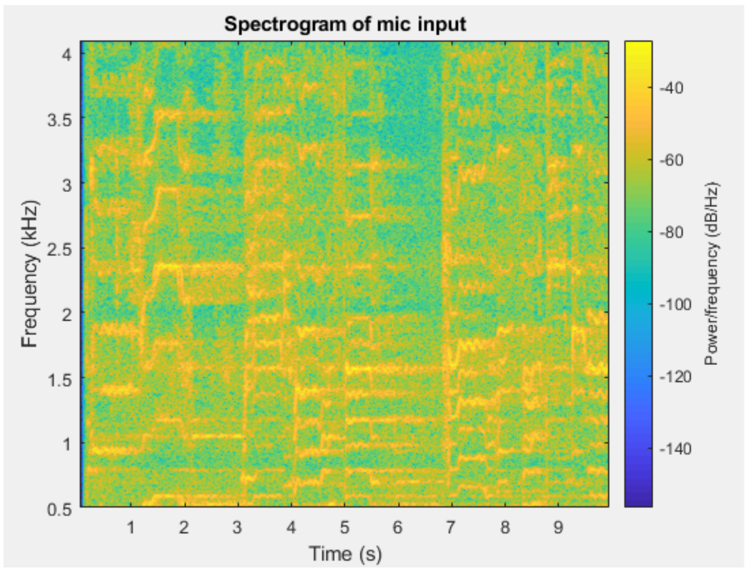


Figure 6.3: Spectrogram of the Microphone Input

- As mentioned above, a 'fingerprint' is obtained from each song.
- The above-mentioned 'fingerprint' for each song is then stored on the SD Card for offline use.
- The file is then taken from the SD Card for immediate and/or future song recognition.

6.2.2 Cloud Storage

- A vast and diverse library of songs will be stored on the cloud.
- The Raspberry Pi module has connection/access to this data and has to be connected to a network during the initial phase of downloading the song in order to obtain its fingerprint.
- Apart from the library, the cloud will also be used as the primary software update option for users.
- Every few minutes, the software code automatically checks the cloud server for a change in the version number of the running software.
- If the software has been updated, it will proceed to prompt the user to initiate an immediate software update or to delay the update to a fixed time.

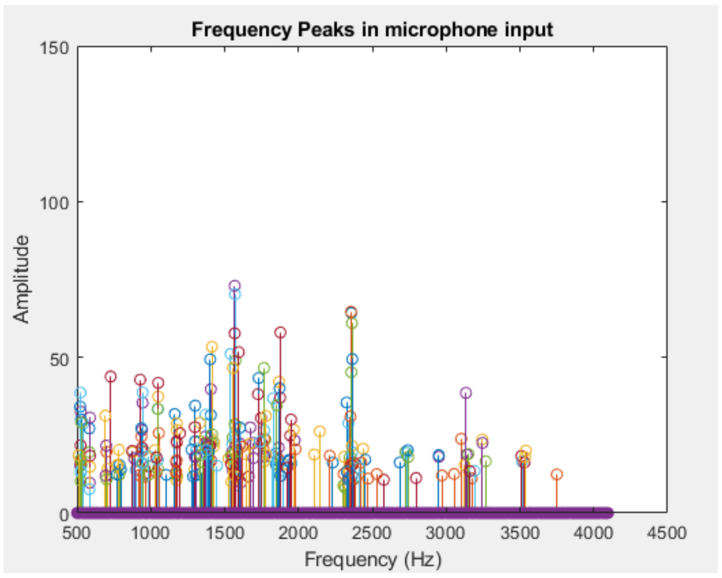


Figure 6.4: Frequency peaks of the Microphone Input

- When the user does complete the update, the module will automatically initiate a complete reboot.

6.3 Detecting the Fundamental frequency

To find the song's fundamental time, the song's sudden sound impulses are illustrated and then the fundamental duration in which those triggers appear are pinpointed. For this, the signal is divided into frequency bands and their envelopes are extracted. These envelopes are differentiated to accentuate sudden sound changes and run through a comb filter bank to extract as our tempo, the highest energy output. 2.2 seconds of music are retrieved from the middle of the recording, processed and the tempo corresponding to the music piece is detected. This 2.2-second sample interval is chosen because it is the minimum amount of information that we can work with to get at least two beats for the slowest tempo we permit that is, 60 beats per minute. To guarantee that the tempo-choosing process does not exaggerate the energy of 60 bpm, we prefer 2.2 seconds rather than 2 seconds. A two-second sample might have a 60bpm search strategy interpreting the beginning and end as beats. Since 60bpm is the lowest search tempo it is feasible to use any sample longer than two seconds.

- *Step 1: Filter bank*

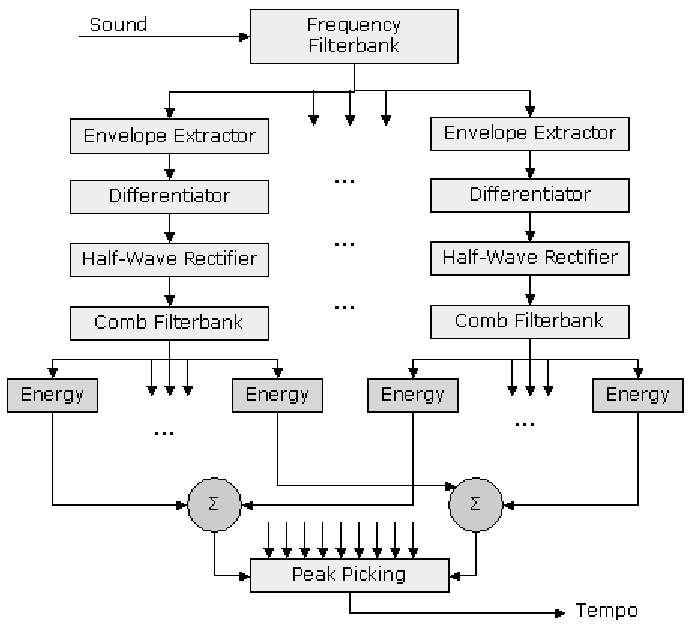


Figure 6.5: Beat Detection Block Diagram

The signal is broken down into different signals, each comprising of a frequency spectrum. This distinguishes notes from different instrument groups, and can be analyzed accordingly. FFT of the signal is taken to isolate the instruments, and then clusters of frequencies are allocated to various frequency bands. The range for each frequency spectrum depends on the frequency of sampling and the length of signal. After taking the inverse FFT, the time domain representations of those signals are sent to the smoothing function.

- *Step 2: Smoothing*

  We reduce the signal to a form in which we can perceive sudden sound changes so that we can comprehend the tempo, we are limiting the signal to its envelope. This can be seen as the sharp increase in the amplitude of the sound and not the frequencies it holds. Each of the frequency-banded signals is low-pass filtered. To this end, the signals are full wave rectified to reduce high-frequency information, and so we would just have to deal with the positive side of the envelope that we were looking for. The right half of a Hanning window with a length of .4 seconds is then convolved with each of the signals. To this effect,

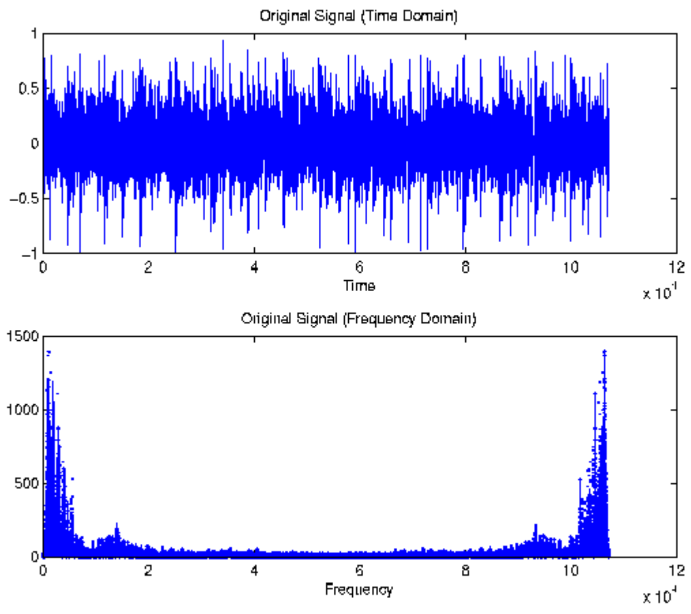


Figure 6.6: Filter bank output of one frequency band (Step 1)

the signals are converted into the frequency domain, multiply and find the inverse FFT to minimize computation time.

- *Step 3: Differentiation - Rectification*

  The envelopes detected are differentiated for accentuation when the amplitude of the sound varies. The largest changes will lead to beats, as the beat is merely a regular sound emphasis. The frequency-banded signals are differentiated in time and then half-wave rectified, so that we can only see rise in sound.

- *Step 4: Comb Filter*

  A comb filter is a sequence of impulses that regularly occur at a given tempo. To ascertain which yields the highest energy, the differentiated frequency-banded signals are convolved with various comb filters.

  The comb filter is specified by a range of tempos that are to be searched for and the resolution or spacing between them. These comb filters are to be convolved with the signal. The FFT of each frequency-banded signal is multiplied by each comb filter's FFT, and each of these are stored. And the frequency-band energies are rounded up and we've

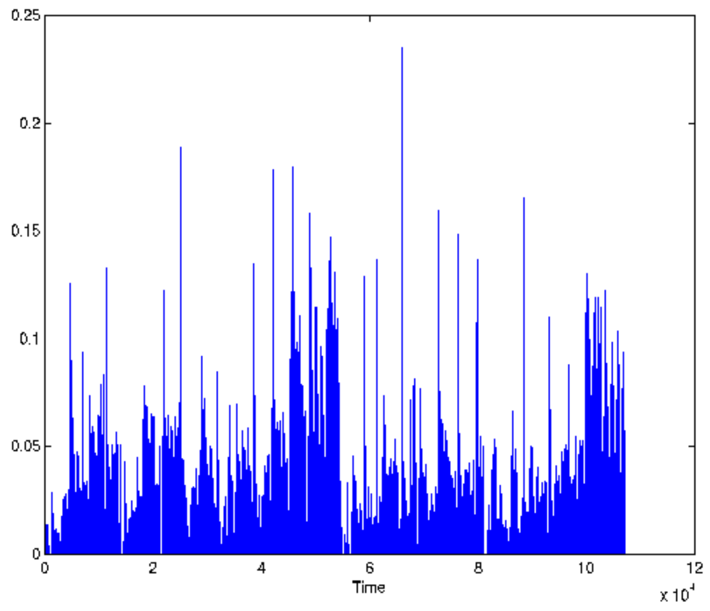


Figure 6.7: Smoothened output (Step 2)

been left with a vector of tempo energies. This output has a peak at the song's fundamental tempo followed by smaller, broader peaks at multiples of this tempo. The maximum value of those energies is taken as the music piece 's fundamental tempo.

6.4 Simple Sound Energy Algorithm

Sound energy variations are detected by comparing the average sound energy of the signal with the instant sound energy. Consider the stereo mode with two lists of values, a_n and b_n. The list of sound amplitude values captured every T_e seconds for the left channel is contained in a_n while that of the right channel is contained in b_n. The instant energy is the energy contained in 1024 samples of a[n] and b[n], 5 hundred of a second is average energy is not computed for the entire song. The instant energy is compared with the nearby average energy. When the instant energy is superior to the local average energy, a beat is detected. The average energy is computed on 44032 samples which is about 1 second to the human ear energy persistence model. The recent 44032 samples are stored in the history buffer which contains the list of samples for the left and right channel history. *Algorithm*

- The instant energy 'e' is calculated with the 1024 samples in a[n] and

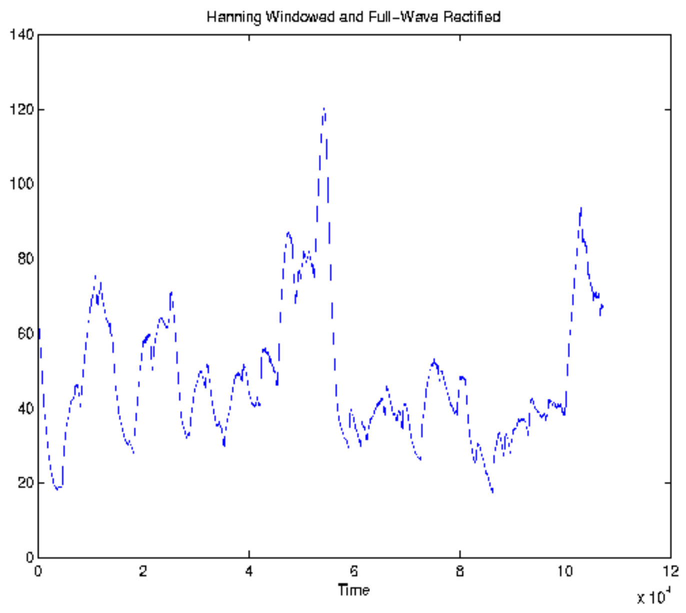


Figure 6.8: Differentiated and rectified output (Step 3)

b[n].

$$e = e_{stereo} = e_{right} + left = \sum_{k=i_0}^{i_0+1024} a[k]^2 + b[k]^2 \tag{6.1}$$

- The local average energy'¡E¿' is computed on the 44100 samples of the history buffer.

$$< E >= \frac{1024}{44100} \times \sum_{i=0}^{44032} (B[0][i])^2 \tag{6.2}$$

- The oldest 1024 samples are deleted to make room for the newest samples by shifting the history buffer to the right.

- The new 1024 samples are stacked onto the top of the history buffer.

- The instant energy computed is compared to $C\times < E >$, where C is a constant which determines the sensitivity of the algorithm. An acceptable value for the constant is 1.3. If 'e' is superior to $C\times < E >$, then a beat is detected.

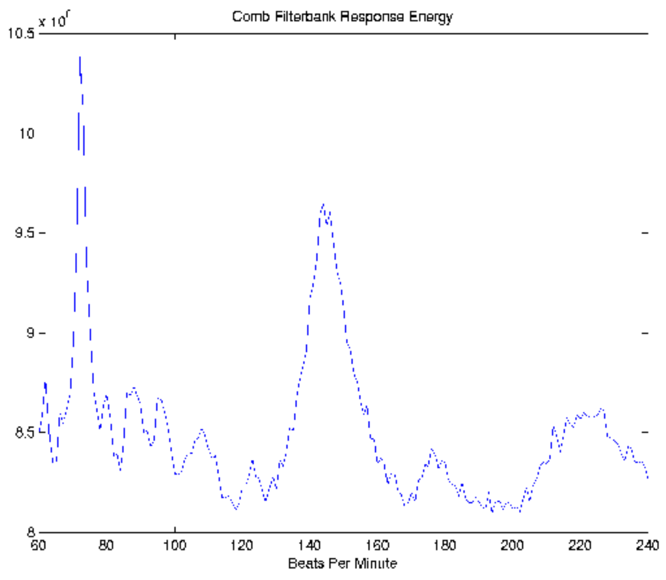


Figure 6.9: Energy response of Comb filter (Step 4)

6.5 Frequency Selective Sound Energy Algorithm

The source signals could be extracted from a .wav file or straight from a streaming microphone as opposed to taking from a_n and b_n. 1024 samples are accumulated and a 1024 frequency spectrum is obtained in the frequency domain through a fast Fourier transform. This spectrum is then divided into many sub bands. The sound energy contained in each sub band is computed and compared with the corresponding recent energy average. A beat is detected when at least one sub band has an energy superior to its average. Increasing the number of sub bands would bring about more precision for the beat detection. To improve on the accuracy of the algorithm, we use the defaults of human ears. The transfer function of the human hearing system is very similar to that of a low pass filter. Humans can perceive low pitched noises more clearly and easily than the high-pitched noises. Thus, a logarithmic repetition is preferred.

Algorithm

- Take FFT of the 1024 samples taken from a[n] and b[n].
- Square the modulus of the 1024 complex numbers to get the frequency amplitudes of the signal and store them in a buffer of size 1024.

- Square the modulus of the 1024 complex numbers to get the frequency amplitudes of the signal and store them in a buffer of size 1024.

$$E_S[i] = \frac{32}{1024} \times \sum_{k=i\times 32}^{(i+1)\times 32} B[k] \tag{6.3}$$

- Define a buffer for Energy history (say Ei[]) and keep storing the immediately past 43 energy computations for the i-th sub band in Ei[i].
- Compute average energy for the i-th sub band.

$$< Ei >= \frac{1}{43} \times \sum_{k=0}^{42} Ei[k] \tag{6.4}$$

- Find the new energy value for sub band i (Es[i]) and place it at Ei[0] after shifting the rest of the values to the right.
- If $Es[i] > (C\times < Ei >)$, i.e. if the new energy value for a sub band is greater than the average value for it, a beat is detected.

Chapter 7

Results and Observations

7.1 Song Recognition

Repeated FFT over small windows of time gives the spectrogram of each song. Fingerprint of each song is obtained on hashing using a combination of peak frequencies and time and stored in the SD Card for offline use. High accuracy was observed while recognizing a song by comparing fingerprint generated from the song recorded using microphone and that from database.

7.2 Beat Detection Algorithm

On dividing the signal into bands based on frequency ranges and then getting their envelopes, emphasis can be made on the sudden impulses in the track. This can be used to obtain the fundamental period. Running a comb filter bank through these envelopes, gives the result with the greater energy result that is taken as the required tempo.

7.3 Mode-wise Observations

Mode 1 of the project replicates a normal metronome and it works the same as a normal metronome app in the phone or keyboard where the first few beats are tapped and it goes on at the same tempo. Mode 2 allows the user to input a pattern which will be repeated as such. Mode 3 presents the user with a list of songs for which the bass drum patterns have already been fixed. In mode 4 the fingerprint of a sample recorded using microphone is compared with existing database to recognize the song and its bass pattern is generated.

Chapter 8

Challenges Faced and our Approach

8.0.1 Lack of access to hardware components

Solution : We had to limit the project to perfecting the software as we did not have the hardware components that we had purchased in hand. We used simulation softwares to try out our algorithms.

8.0.2 Overheating of pedal due to continuous use over a long period of time

Problem Faced : Stepper motors with different current ratings dissipate different amounts of power. The most apt motor was to be chosen for efficient working of the bass drum pedal mechanism. The interfacing was to be done in the most efficient manner and the driver's should be chosen accordingly. The shield or driver should be able to handle the current flowing through it and it shouldn't get heated up.

Solution : This problem was sorted by using an appropriate servo motor shield that supports our motor's specifications.

8.0.3 Overheating of circuit components from prolonged use can lead to the damage of the setup entirely

Solution : Components with proper rating were chosen to avoid this. The software should also be optimised. i.e the delays should be given carefully so as not to overheat the driver.

8.0.4 Wear and tear of the pedal due to repeated to and fro motion at varied intervals and speeds

Problem Faced : This will lead to poor execution of pedal, regardless of whether the required signature is simple or complex.

Solution : The problem faced during our MPMC project was amplified due to poor use of the motor driver on our part which was rectified. The model of the pedal used is completely different this time. It is way more realistic and we have no such problems this time.

8.0.5 Small error in the detected or inputted pattern

Problem Faced : This might result in a large variation from the desired beat when played along with a song if it's working for a long time. This is because the slight error will get magnified when the same error occurs repeatedly for every beat.

Solution : This error was minimised by taking in multiple inputs and averaging over the intervals.We also did our best to reduce unnecessary software delays.

8.0.6 The input from the button might not be precise

Problem Faced : Due to the difference in sensitivity of the type of push button being used. A very accurate push button should be used and it should also be one that caters the specific needs of the project.

Solution : This particular problem was tricky to solve. After conducting many trials with no success, we resolved to using a HEX Keyboard. This meant that we had to interface the HEX Keyboard to the Arduino UNO and also the LCD Display which was done with ease.

8.0.7 Constants are different for average energy calculations of different genres

Problem Faced: For the Simple Sound Energy algorithm, the constant with which the average local energy is to be multiplied changes with the type of music. Even for one song, the constant should ideally change based on the type of music that is going on. For beats which are intense and precise, the constant should be high and for songs with a lot of noise, the constant should be lower.

Solution: Find the variance of the current or instantaneous energy and use this value of variance to find the suitable value using linear regression.

8.0.8 Beats are accidentally skipped while using Simple Sound Energy Algorithm

Problem Faced: For music with not so precise beats and noise, two beats might seem to be one long note as this algorithm cannot detect pitch. This will lead to a beat being skipped.

Solution: We used the Frequency Sound Energy algorithm to determine which frequency sub band has a beat.

Chapter 9

Conclusion and Future Scope

9.1 Conclusion

As seen from the progress so far we have managed to successfully complete four modes of operation completely in software and partially in hardware for the bass drum pedal and test it on a small scale prototype. The issues that arose due to the stepper motor and servo motor were fixed with the use of a solenoid motor. The issues that arose with beat detection with simple sound energy algorithm were fixed using frequency sound energy algorithm. Further work that is remaining includes the implementation of the solenoid motor circuit that has been designed.

9.2 Future Scope

- To extent the project to a snare drum as well, if the algorithm works optimally for the base drum.
- To ensure that the algorithm works well for wide genre of music by filtering and amplifying the initial sound track as necessary.
- To increase the library of songs available in Mode 3.

References

[1] G.Tzanetakis, G.Essl, P. Cook, Audio analysis using the discrete wavelet transform (Proceedings of the AMTA, WSEAS), 2001

[2] V. X. Afonso, W. J. Tompkins, T. Q. Nguyen, and S. Luo, “ECG beat detection using filter banks,” IEEE Trans. Biomed. Eng., vol. 46, pp. 192–202, Feb. 1999.

[3] Wang, A. An Industrial Strength Audio Search Algorithm. In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Baltimore, MD, USA, 26–30 October 2003; pp. 7–13.

[4] M. L. Miller, M. A. RODRIGUEZ, I. J. Cox, “Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces”, Journal of VLSI Signal Processing 41, 285–291, 2005

[5] Pedro Cano , Eloi Batlle , Harald Mayer, Helmut Neuschmied, “Robust Sound Modeling for Song Detection in Broadcast Audio”, AES 112th Convention, Munich, Germany, 2002 May 10–13

[6] S. Emima Princy ; K. Gerard Joe Nigel, “Implementation of Cloud Server for Real Time Data Storage using Raspberry Pi,” 2015 Online International Confernece on Green Engineering and Technologies (IC-GET 2015)

[7] Fung Po Tso, David R. White, Simon Jouet, Jeremy Singer, Dimitrios P. Pezaros ,“The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures”, 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops, 2013

[8] K.W. Lim, N. C. Cheung, M. F. Rahman, “Proportional control of a solenoid actuator,” Proceedings of IECON’94 - 20th Annual Conference of IEEE Industrial Electronics, 1994

[9] S. Obata,“A Basic Electromagnetic Theory for Controlling Solenoid Actuators”, 10th France-Japan/ 8th Europe-Asia Congress on Mecatronics, 2014