Final Report Part I

Guitar Fretboard Mapping (Audio to Virtual Fretboard) and Chord Finder

COMP 4801 Final Year Project

Department of Computer Science

The University of Hong Kong

Joseph Xavier Vinit Miranda, 3035243778

Supervisor:

Prof. Dirk Schneiders

14th April, 2019

Table of Contents

a. **Abstract**…………………………………………………………………………………...2

b. **List of Figures**……………………………………………………………………….3

c. **Outline**……………………………………………………………………………….4

1. **Introduction**………………………………………………………………………....5

2. **Previous Works**…………………………………………...………………………...7

3. **Methodology**……………………………………………………………………...9

3.1. Fretboard Mapping………………………………………………………….9

3.1.1. Concept…………………………………………………………....9

3.1.2. Calibration………………………………………………………..12

3.1.3. Onset Detection………………………………………….........12

3.1.4. Pitch Detection…………………………………………………...14

3.1.5. Image Processing………………………………………………...17

3.1.5.1. Extract the guitar fretboard……………………………....17

4. **References**………………………………………………………………………….19

a. Abstract

Fretboard mapping, namely the ability to translate audio data to a position on a virtual fretboard is a highly-desirable feature needed in many fields of guitar playing. It is useful in the process of writing guitar tablature, i.e. musical notation for guitar, which is quite tedious and time-consuming when performed manually. It has also been useful in developing the self-learning guitar application, *Chord Finder*, which runs on the backbone of the fretboard mapping algorithm, and helps teach chord voicings all over the fretboard in an intuitive manner.

There are a number of useful pitch detection algorithms that detect the pitch of a note played on the guitar with high accuracy. However, this is insufficient for fretboard mapping which also requires spatial information. We thus intend to perform the fretboard mapping in real time using concepts of *Music Information Retrieval* and *Computer Vision* to pinpoint the location of the notes played by the user on the guitar neck. A pitch detection model has been implemented to capture the frequency of the note being played and a computer vision model has been used to detect the position of the user's hand. Our algorithm is a *multi-modal* approach that employs a combination of the two models to help identify the string and fret on the guitar fretboard.

Preliminary testing of the algorithm shows that it performs better at fretboard mapping than existing methods when it comes to the acoustic and classical guitar.

It is hoped that the effectiveness of this approach and the research carried out during its implementation will bring a positive change to the outlook of music education, and will spark new work that applies our concepts to other instruments.

b. List of Figures

Fig 3.1……………………………………………………………………………………9
Fig 3.2…………………………………………………………………………………...12
Fig 3.3…………………………………………………………………………………...13
Fig 3.4…………………………………………………………………………………...14
Fig 3.5…………………………………………………………………………………...14
Fig 3.6…………………………………………………………………………………...15
Fig 3.7 …………………………………………………………………………………..16
Fig 3.8………………………………………………………………………………….. 16
Fig 3.9…………………………………………………………………………………...17
Fig 3.10…………………………………………………………………………………17
Fig 3.11……………………………………………………………………………….... 17

d. Outline

This report aims serves as part I to the exhaustive analysis of our final year project. The first chapter of the report includes a background to the project, the need for such an application for its target audience, guitarists, and the main objective of the project. The second chapter of the report will then detail previous works related to guitar transcription and the applications and methods available. The third chapter of this paper discusses the concept of the fretboard mapping algorithm up until the point where the guitar fretboard, bridge and headstock is extracted from the image on the user's webcam. Part II of the report will provide further information on the *fretboard mapping algorithm* and *chord finder.*

1. Introduction

An algorithm that maps raw audio from a user's laptop microphone to a string and fret on a virtual guitar fretboard is highly sought after and is quite useful for multiple reasons. One reason is the *notation of guitar tablature* (tabs) by guitarists as they try to transcribe their work so that it can be read by others easily. A robust fretboard mapping algorithm can be used by guitarists to transcribe their music conveniently and in lesser time without having to manually select the note and string of the guitar on existing transcribing software such as Sibelius [1]

Another reason why fretboard mapping is important is the creation of intuitive guitar education applications that help teach amateur guitarists to take their playing to the next level. Applications can be developed to check if the user is playing the melody of a guitar song the right way by playing the correct string and fret on the guitar for higher accuracy. Key check can be used to check if the user is playing in key or not.

Using the fretboard mapping algorithm we have made an application, *Chord Finder*, which helps amateur guitarists find chord voicings all over the fretboard. Users of the application will be able to circumvent multiple tedious aspects of songwriting, such as the location of the most convenient and ideal-sounding voicing for a given chord, and making sure that the piece being written is in key, which will enable them to exercise their creativity with little disruption. The application will also help guitarists remember the location of all the different notes on the guitar: a technique that is very difficult to master and often requires tedious memorization without any meaning.

The final deliverable will be a native desktop application, providing interactive support to novice guitarists to help play all types of chords using both visual and audio input from their instruments; the product will track the player's movements on the guitar fretboard (the guitar neck) using a combination of pitch detection algorithms and computer vision algorithms to track the user's hand position. Users of the application will not only be greatly assisted in the memorisation of chords, but they will also be able to circumvent multiple tedious aspects of songwriting, such as the location of the most convenient and ideal-sounding voicing for a given chord, and making sure that the piece being written is in key, which will enable them to exercise their creativity with little disruption.

2. Previous Works

There have been a lot of efforts made in the field of Computer Vision and Music Information Retrieval to map audio to the fretboard. Techniques that exclusively employ computer vision have been used in the past, but the accuracy of such methods is very low [2]. Since the approach employed only relies on Computer Vision, it is very difficult to actually obtain the exact string and fret number of the user's hand, especially when multiple fingers of the user's hand show up on the fretboard when the user plays a note.

There is another method that detects only the guitar audio from a microphone and displays the notes as they are being played live. In terms of tab generation, it does not have the ability to distinguish which string a note is coming from, but it makes educated guesses [3] and the technique is optimized for one guitar.

Transcribe [4] is an application that acts as an assistant for people who want to transcribe music from a recording. It is used to slow down the recording and to then identify the spectrogram of possible pitches as well as some frequency analysis. However, this application still does not automate the transcription. It is essentially a specialised video player that is optimized for its purpose. Furthermore, it is not meant for real-time use and does not detect video and audio from the webcam and microphone respectively. Advanced machine learning techniques that involve the use of deep belief neural network models have also been used before with modest success [5].

Our work aims to be as intuitive as possible; any hindrance in the action of guitar playing should be minimised. Moreover, it needs to be accessible: any ordinary personal computer with an attached webcam and microphone should be sufficient for the application's use. The fretboard mapping algorithm, that is introduced in this paper, follows these constraints and is also reasonably competent in its function.

Among existing self-learning guitar education applications, *Ultimate Guitar* [6] is one of the most prominent ones in the domain: it is an expansive library of song tabs that allows users to not only practice using said tabs to expand their repertoire, but also to make their own submissions and review existing submissions. *Yousician* [7] is another notable application; it allows users to practice songs in real time on their guitar by playing along scrolling tab visualizations that provide constant feedback on the accuracy. There are numerous other applications that provide libraries of chord shapes and musical scale shapes, both of which constitute the building blocks of harmony and melody of guitar-based songs, respectively. *Chord Finder* is different in the sense that it'll teach the user how to actually play these harmonic lines on the guitar along with multiple inversions in an interactive way.

3. Methodology

3. 1 Fretboard Mapping

3.1.1 Concept

Octave Colour Code

2	3	4	5	6

Fret Note

Open	1F	2F	3F	4F	5F	6F	7F	8F	9F	10F	11F	12F	13F	14F	15F	16F	17F	18F	19F	20F
E4	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C
B3	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G
G3	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb
D3	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb
A2	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F
E2	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C	C#/Db	D	D#/Eb	E	F	F#/Gb	G	G#/Ab	A	A#/Bb	B	C

Fret Frequency (Hz)

Open	1F	2F	3F	4F	5F	6F	7F	8F	9F	10F	11F	12F	13F	14F	15F	16F	17F	18F	19F	20F
329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99	830.61	880.00	932.33	987.77	1046.5
246.94	261.63	277.18	193.66	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99
196.00	207.65	220.00	233.08	246.94	261.63	277.18	193.66	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88	523.25	554.37	587.33	622.25
146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94	261.63	277.18	193.66	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16
110.00	116.54	123.47	130.81	138.59	146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94	261.63	277.18	193.66	311.13	329.63	349.23
82.41	87.3	92.49	97.99	103.83	110.00	116.54	123.47	130.81	138.59	146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94	261.63

Figure 3.1: A visual depiction of the notes on a fretboard/guitar neck and their corresponding frequencies [9]

The product is centered around the concept of fretboard mapping, i.e. the mapping of raw audio signals corresponding to a note played on a user's guitar and retrieved from a microphone, to a string and to a fret on said string. It is vital for the application to interpret the exact position (the string and fret combination) on the fretboard of the note being played, in order to give any meaningful advice on finger positioning for the user's next line of action in their song. Conventional pitch detection, for which there is plentiful existing work, does not provide sufficient information for this purpose, since the same frequency corresponds to multiple positions on the guitar, as can be seen in figure 1. This information, thus, needs to be supplemented with visual input. It suffices to visually detect, using the user's webcam, the general locality of the fret where the user has his fretting hand placed, without the need for determining the exact finger positions of the user. This is because, given a fretting hand position, there is only one possible fret and string combination reachable from this position that corresponds to the pitch supplied by the pitch detection algorithm, and can thus, be easily inferred.

It is worth noting that the need for computer vision could be circumvented by calling the user to manually input the string number on which they played the note detected. However this is highly unintuitive and would disrupt the user's creative flow due to having to constantly input the string number during the process. Thus, visual detection of the hand position to infer the string/fret combination is the most practical choice in this circumstance.

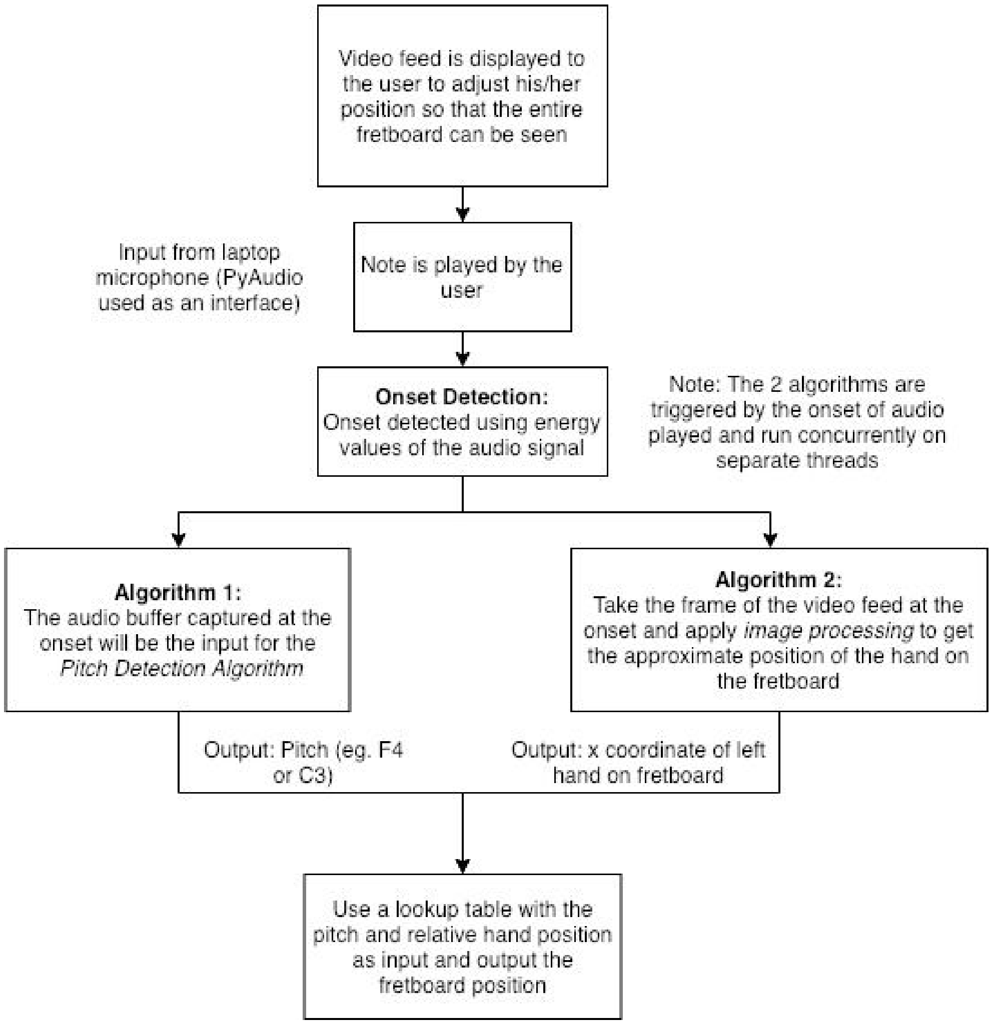

Fig 3.2 Summary of the Fretboard Mapping Algorithm

In summary, the frequency of the note played by a user is detected from the user's mic input, using a suitable pitch detection algorithm in terms of speed and accuracy. This is accompanied by the use of computer vision techniques to detect the general locality of the fret position from the user's webcam input, in order to successfully map the note played by the user to a virtual fretboard.

3.1.2 Calibration

In an application such as this, where detecting a user's hand position is important, it is reasonable to employ skin detection. A skin color histogram can be used to subtract the background from an image, only leaving parts of the image that contain skin tone.

A much simpler method to detect skin would be to find pixels that are in a certain RGB or HSV range. The problem with the above approach is that changing light conditions and skin colors can cause problems with the skin color detection. In contrast, a skin color histogram is likely to be more accurate and also takes changing lighting conditions into account.

The user of the application must initially calibrate by capturing the region of interest which covers the user's hand. The skin color histogram will then be calculated based on the HSV values of the region, which are then normalized and stored in a numpy array for consequent use.

3.1.3 Onset Detection

A survey of existing onset detection implementations from popular Music Information Retrieval libraries such as Essentia, Librosa and Aubio, was carried out. Since none of the functions found were capable of carrying out onset detection in real-time, an implementation was created using the method introduced in [10]. The algorithm used is as follows:

The first step of the algorithm involves converting the audio signal to an Onset Detection Function value (ODF value) which is a representation of the signal at a low sampling rate. Each ODF value will correspond to a single audio buffer which contain 1024 samples. The ODF is a good measure of the unpredictability of the audio signal which is helpful in detecting an onset. It is calculated using the energy values of the buffer, which are calculated using the formulae shown below.

The next step of the onset detection process is to detect the peaks of the ODF signal, i.e. detect where the value of the ODF is higher than a certain threshold. The location of the peak gives the location of the onset.

$$\mathrm{ODF}_E(n) = |\, E(n) - E(n-1) \,| \qquad E(n) = \sum_{m=0}^{N} x(m)^2,$$

Whenever a note is played, the onset is detected in real time and this initiates both the pitch detection and image processing algorithms. After adjusting the parameters, a near 100% onset detection accuracy was accomplished.

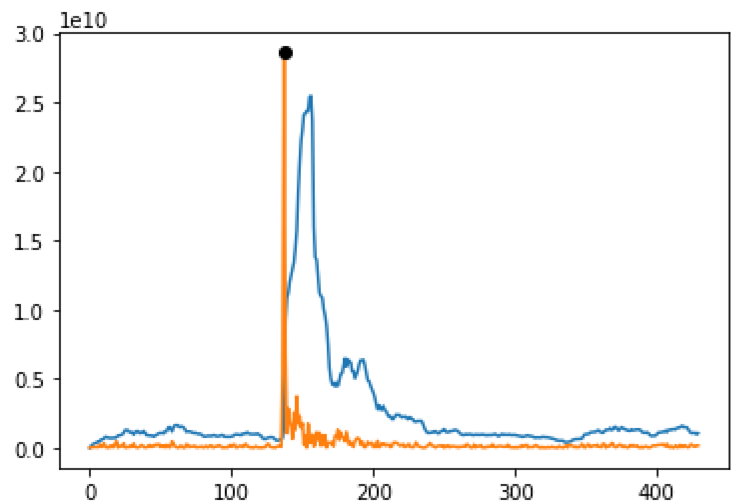

Fig 3.3 Onset Detection. The orange line represents the ODF Function of the Audio Signal and the blue line represents the Threshold Function. In the above 5 second recording, one onset is accurately recorded

Real-time peak picking (one buffer delay).
Input: ODF value
Output: Whether or not previous ODF value represents a peak (Boolean)

```
IsOnset ← False
if PreviousValue >CurrentValue and PreviousValue
>TwoValuesAgo then

    if PreviousValue >CalculateThreshold() then
        IsOnset ← True
    end

end
UpdatePreviousValues()
return IsOnset
```

Fig. 3.4 Peak picking Algorithm [10]

3.1.4 Pitch Detection

The pitch detection algorithm used is *frequency extraction using autocorrelation.* This use the audio buffer that triggered the onset as its input. This algorithm has been optimized for robust performance in this application.

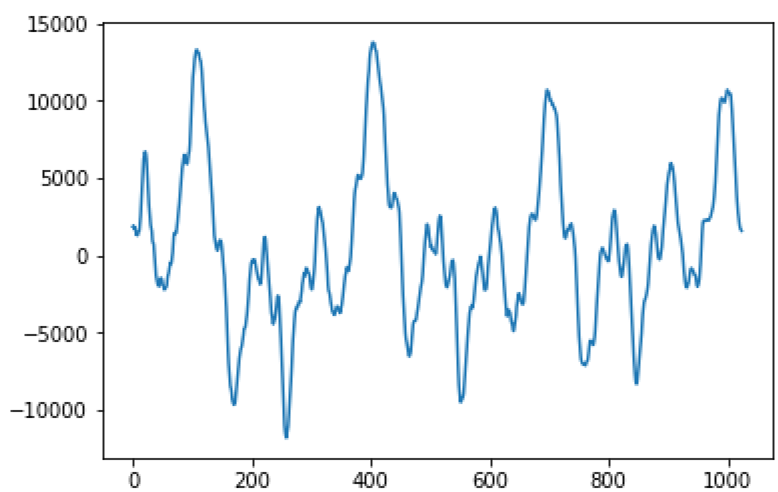

Fig 3.5 The audio buffer being used for pitch detection

```python
def freq_from_autocorr(raw_data_signal, fs=48000):
    corr = fftconvolve(raw_data_signal, raw_data_signal[::-1], mode='full')
    corr = corr[int(len(corr)/2):]
    d = diff(corr)
    start = find(d > 0)[0]
    peak = argmax(corr[start:]) + start
    px, py = parabolic(corr, peak)
    return fs / px
```

Fig 3.6 The algorithm that is being used for pitch detection[11]

3.1.5 Image Processing

3.1.5.1 Extract the guitar fretboard

Fig. 3.7 Image captured by the webcam at the onset of a note. Note that only the bottom half of the webcam is captured for simpler analysis

Once the onset is triggered, a picture is captured by the user's webcam and the picture is put through a series of filters before the hand position is detected. Only the bottom half of the image is used for analysis and the image is first converted to a grayscale image and then *Canny edge detection* is applied to detect edge pixels.

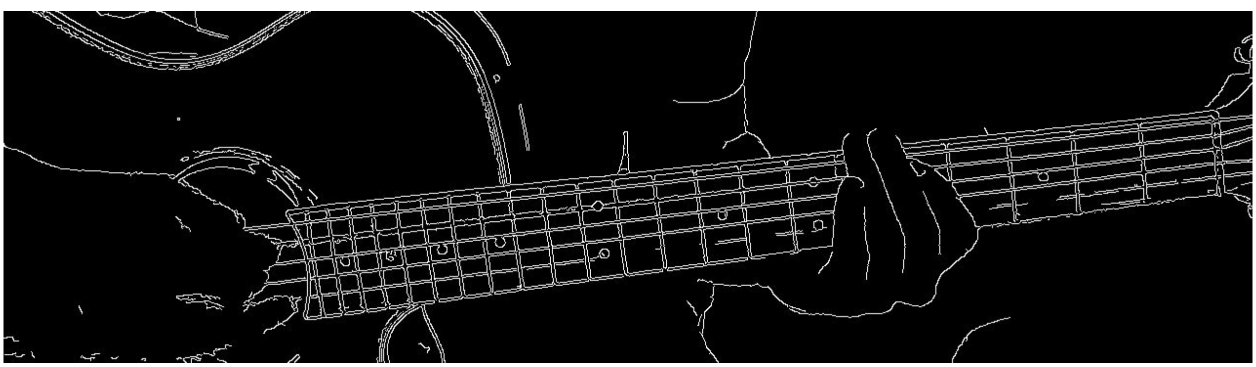

Fig. 3.8 Canny edge detection being used to detect the edges. A suitable threshold was chosen such that only the sharp edges (the lines on the fretboard) are detected

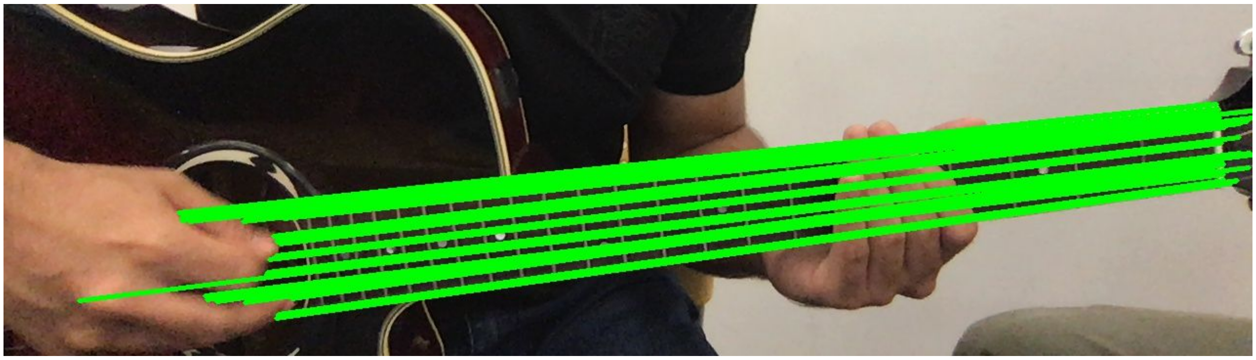

Fig. 3.9 Probabilistic Hough Transform used to detect lines on the fretboard; the minimum length parameter is set to a high value, in order to extract only the lines on the fretboard.

The output of the edge detection is sent to the *Probabilistic Hough Transform* to detect lines. This will detect and return all the lines of a certain length in the image. The length has been kept sufficiently high to capture only the lines on the fretboard. The top and bottom lines that outline the guitar fretboard remain after a filtering process: the top and bottom lines that intersect the vertical line at the middle of the screen are chosen as the top and bottom lines of the fretboard, respectively. The coordinates of the lines enables the selection and rotation of the region of interest, i.e. the fretboard, the headstock and the bridge. The next part of Image Processing will be continued in Hamza's report.

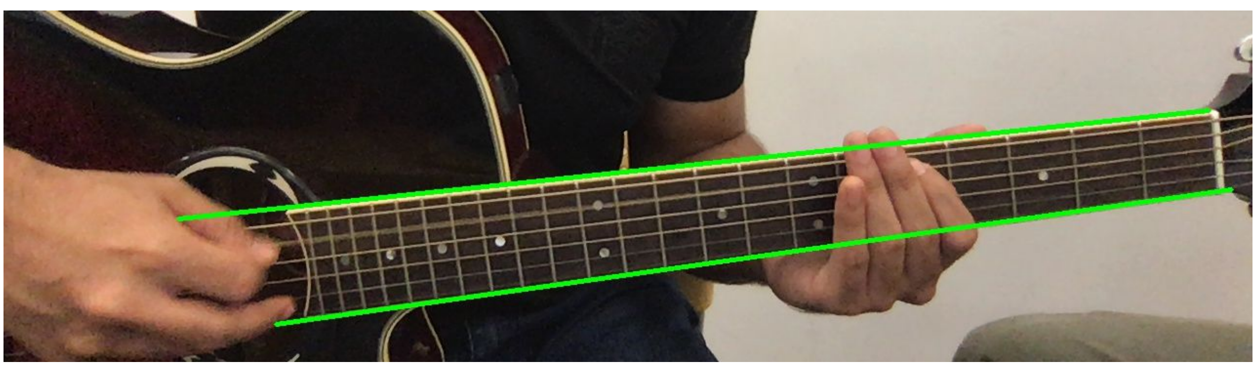

Fig. 3.10 Filtering out all the lines except the top and bottom lines to get the outline of the guitar

Fig. 3.11 The outline is cropped and rotated to get a picture of the fretboard, the headstock and the bridge. Further processing needs to be done to detect only the fretboard

6. References

1. Sibelius. https://www.avid.com/sibelius-ultimate
2. Scarr, J. and Green, R. (2018). *Retrieval of Guitarist Fingering Information using Computer Vision*. Department of Computer Science and Software Engineering University of Canterbury, New Zealand.
3. Real-time Guitar Analysis and Tab Transcription. https://www.youtube.com/watch?v=nt_kMxCK78I
4. Transcribe. https://www.seventhstring.com/
5. Burlet, G. (2015). Guitar Tablature Transcription using a Deep Belief Network. Available:https://era.library.ualberta.ca/items/4503fe8b-10ef-431b-9ac4-2096b6cb61ac/view/c02e9ae0-4528-4e39-b80c-7bd5ad0054d3/Burlet_Gregory_D_201507_MSc.pdf
6. Ultimate-guitar.com. (2018). *ULTIMATE GUITAR TABS - 1,100,000 songs catalog with free Chords, Guitar Tabs, Bass Tabs, Ukulele Chords and Guitar Pro Tabs!*. [online] Available at: https://www.ultimate-guitar.com/ [Accessed 24 Oct. 2018].
7. Yousician.com. (2018). *Yousician – Your personal music teacher*. [online] Available at: https://yousician.com/ [Accessed 24 Oct. 2018].
8. Carlton, D. (2013). I analyzed the chords to 1300 songs for patterns. This is what I found. (Part 3) Interactive Discovery. [Blog] *The Hooktheory Blog*. Available at: http://www.hooktheory.com/blog/chord-progression-search-patterns-and-trends/ [Accessed 24 Oct. 2018].
9. Guitar note frequency visualization. (2013). [image] Available at: https://stackoverflow.com/questions/18694542/how-to-detect-string-when-pitch-tracking-on-electric-guitar [Accessed 24 Oct. 2018].
10. Glover, J. , Lazzarini, V. and Timoney, J. (2011). *Real-time detection of musical onsets with linear prediction and sinusoidal modeling.* EURASIP Journal on Advances in Signal Processing
11. Note Trainer - Real-Time Musical Note Detection with Raspberry Pi and Python https://www.youtube.com/watch?v=wkZW9Hgrnao&t=23s