CSE490E1 Assignment 1

Boren Li
1468594

Last quarter, I took a sound and wave capstone. My partner and I designed and implemented a project called Instant guitar chord maker. What the project does is that when a guitar player plays a chord, the software captures the chord and detects strings and frets that are played in order and displays the result. I am glad that the project meets the design goals. But when I look back now, there are two things that could be done better.

1. design

I chose the UI to be a desktop application. I have a guitar displayed on the screen and when a string is detected, that string turns green and a dot shows up on the fret. If no fret is pressed, there is simply no fret to show. To make UI looks good, I decided to use front end knowledge (html, css and javaScript) to implement the UI part. The core part of the software that is ,frequency analysis part, was written in java. I simply googled it and it looked to me that JavaFx can be used as a bridge between java (from analysis part) and javaScript (UI part). But when I designed it, I did not take a look at a concrete example on what JavaFx is capable of doing. After a couple of weeks, we finished both frequency analysis and UI, then I realized that I can only call javaScript methods in java code and vice versa using javaFx but there is no way to actually send data from java to javascript. To me, it is a big design mistake. I spent a good amount of time implementing UI using javascript but all of the effort is in vain if I cannot send data to UI. Fortunately, I found another approach to get around the issue by converting java type data into JSON and save it in the local file and when data in local file is updated, code in UI executes and display the result on the screen. I think design is one difficulty when building the software. Design should be concise on every aspect, otherwise it has a big potential to spend efforts for nothing if design has to be reconsidered in the middle of implementation.

2. Implementation

As a software developer, I try to avoid surprise in the implementation phase. I like to get the logic straight and clear in the design phase. However, sometimes it is inevitable that I have to face surprise. Since I know very less about music, I did not realize that harmonic plays a big role in guitar. In my implementation, I used FFT (fast fourier transform) to detect base frequencies of pitches but at the same times, base frequencies also generate harmonics. During development of the project, harmonics problem is the most confusing and most unpredictable factors. To solve the problem, My partner and I made a histogram of magnitudes of FFT result of first 1000 bins to give ourselves a better understanding of these harmonics. We found out that for every pitch that is really being played has at least a harmonic of double of its frequency. For example, after playing 6th string empty fret with frequency of 82.4Hz, there should be a very high magnitude at frequency of 165 Hz which is double of 82.4Hz. Based on this property, we can exclude many

noise and high multiplier(5 times or more) harmonic. This process filtered out many unexpected result, but there were still many low multiplier harmonic exists. We checked the possible of existence of each low multiplier harmonic at each frame, and we removed those harmonic that same frequency pitches are unlikely to be played in frame. I think the difficulty about implementation is not actually coding but find the way to solve the problem especially when facing a problem that has never met before.

3. Testing

Testing can be tricky for this project. Many thing can go wrong so somehow it can be hard to target the problem. First challenge is the input that we feed to the frequency analyzer code. The input is created when the player plucks the string so that input might be different from time to time. Besides, different frequencies of noise varies. Therefore, when we thought we fix the bug, we actually could not test the code with the exact same input. Since the software should work for all kind of guitars but since the material that made the guitar might be different so generated frequency can be different. Besides some strings are made of nylon and some are made of steel, it can add some uncertainty. So you can tell it might be difficult to test the software. So far I have not figured a better way but I will continue to work on it.