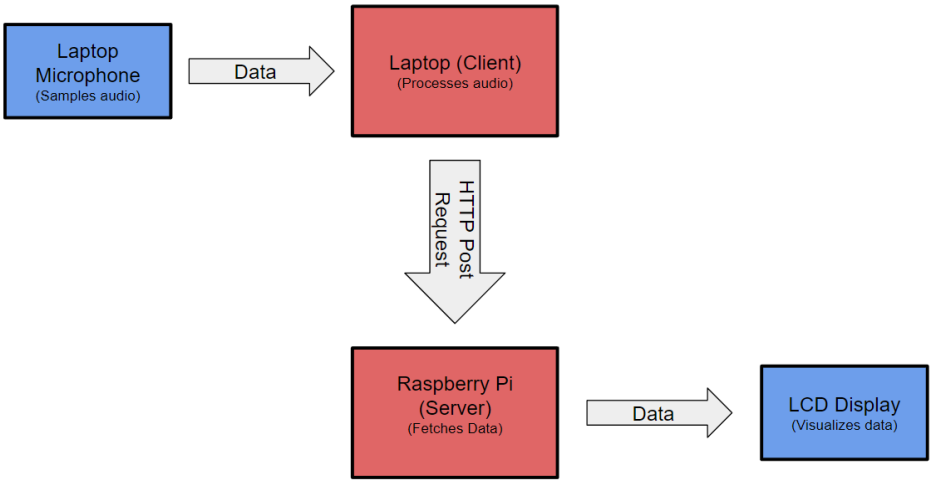
Munir Bshara
Hongyu Zhao
May 5th, 2023

EE250 Final Project: Tuner

In this final project, we aimed to develop a real-time tuner that would concurrently take audio samples using a computer's microphone and send the results of those samples to a raspberry pi. The raspberry pi would then display the frequency of the most prevalent tone and the note it corresponds to on the LCD display interface using GrovePi.

Before getting into the details of how each component of this project operates here is a Block Diagram of the design:



The workflow of the project starts at the laptop's microphone, which samples audio using the Python library PyAudio, in real-time at constant intervals. This data is then processed by the laptop using a simple FFT scheme in which we looked for the single most prevalent frequency in the sample. With this frequency, we then mapped the frequency to the note equivalent(using list comprehension). Now that the key information is extracted from the data we must send the data from the laptop to the raspberry pi. The protocol we utilized for this communication was HTTP in which the laptop acted as the client and raspberry pi as the server. The data payload of the HTTP Post request consisted of the frequency and the corresponding note separated by a single comma. Once this request was received by the server(RPI), the data was extracted and separated into the two corresponding parts that were separated and displayed on the LCD display using the I2C interface that GrovePi provides.

Overall the challenging aspect of the project was deciding on the different parameters of the FFT as well as dealing with the inaccuracies and the post-reading compression of the audio signals. In order to decide on the different parameters of the FFT, we did multiple FFTs at once changing one parameter at a time in order to test the effect of each one in our project. Once we understood the parameters we were able to come up with our "ideal-looking" FFT scheme. Regarding the audio reading, the computer used for testing decided to "enhance" the audio read

by the input microphone which actually messed up with our initial tests. This meant that some of the initial tests were inaccurate or just flat-out not being read. To fix it we had to manually turn off multiple obscure settings on this computer to get a clean read. We also changed the number of channels from 2 to 1 so that there is no conflict from computer to computer.

Overall this project provided a fun challenge with dealing with multiple different topics we encountered throughout the entire semester. In the future to improve our project, we would want to use a buzzer to play the specific tone that the client is reading. This would require the use of a timer ISR as well as an ADC-like design to get the count value for the ISR from the frequency given(something like 109). Also, we would like to improve the accuracy of our project for higher-pitched notes which are relatively really close together, so a slight inaccuracy in reading or the FFT could mess with the entire result. This is primarily caused by the lack of accuracy of the microphones, so to combat this we would have to buy more expensive and accurate microphones. We could also get user input for what instrument they wanted to tune and thus provide a guide for the tuning, making it easy for new musicians to pick up.