Real-time Frequency Analyzer in a Volumetric Display

Rev. 0

General Description

The objective of this project was to design and manufacture a volumetric display which projects a 3D representation of the frequency content of an incoming signal. This device is meant to enhance the music-listening experience with compelling visual effects. Volume-swept displays give the illusion of holography by rapidly moving an LED board. To accomplish this at arbitrary speeds, a speed controller turns a motor that spins a 64x64 LED board from its center axis. Meanwhile, an on-board USB microphone packages and sends chunks of audio data continuously to a Raspberry Pi. The Pi samples the chunk and transforms the time signal into the frequency domain. A separate controller communicates the desired frequency range to be displayed by the LED board via bluetooth. Then, the Pi divides the chunk of frequency content into 64 bins and assigns a local average amplitude. Finally, it maps each bin to a row in the board and assigns LED states accordingly.

Github Link

https://github.com/egelbtuch/3D-Live-Music-Visualizer-Guitar-Tuner.

Features

1. Power electronics for and communication between Raspberry Pi and LED Board
2. LED Board encased in a frame, axle attachment spun by a motor (via motor controller)
3. Bluetooth controller to set frequency range on the LED Board (communicates with the Pi)

Complete Electrical Block Diagram

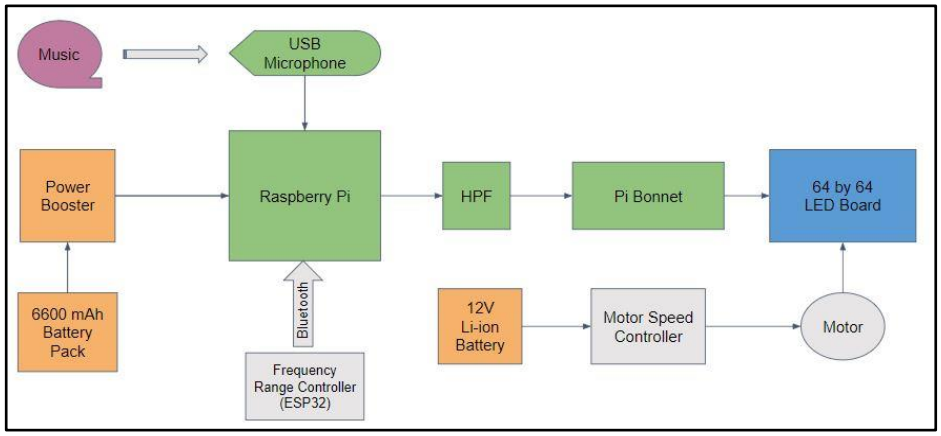

Figure 1: Complete Electrical Block Diagram

Mechanical Drawings

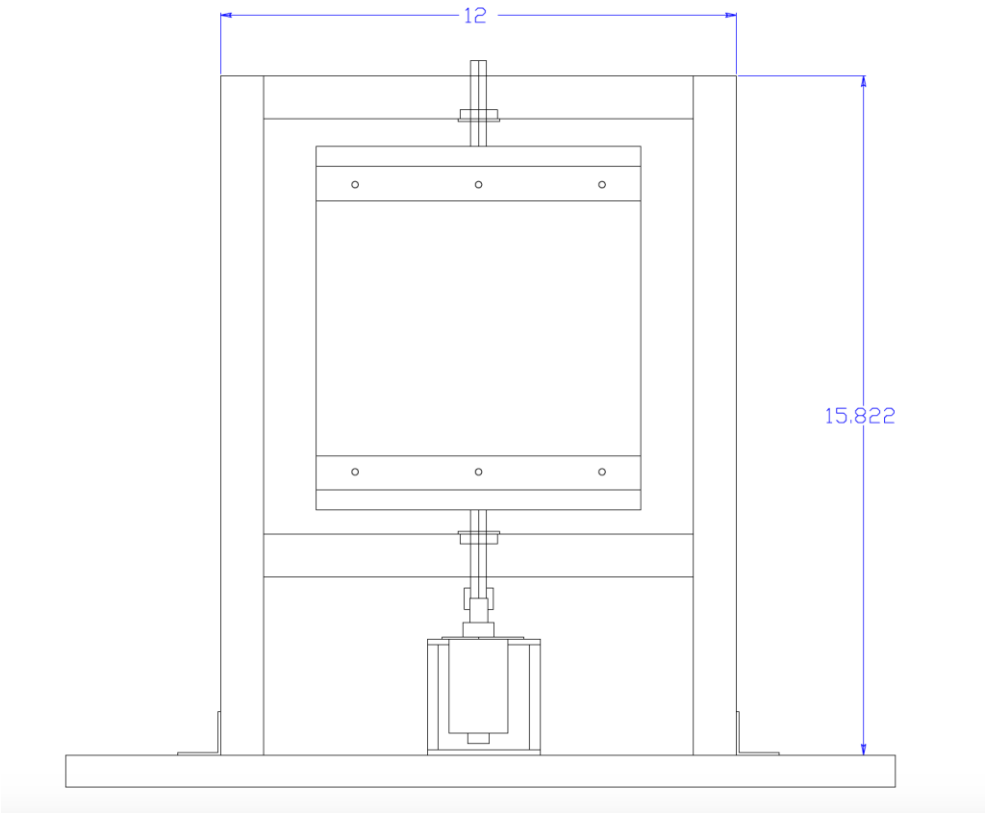

Figure 2: Mechanical Drawing for Motor, Board and Frame

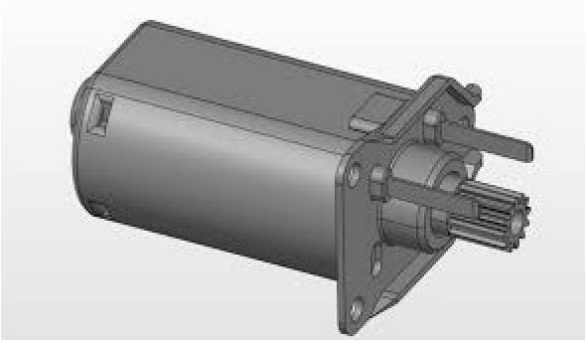

Figure 3: Denso Throttle Motor CAD mark-up

Feature 1:

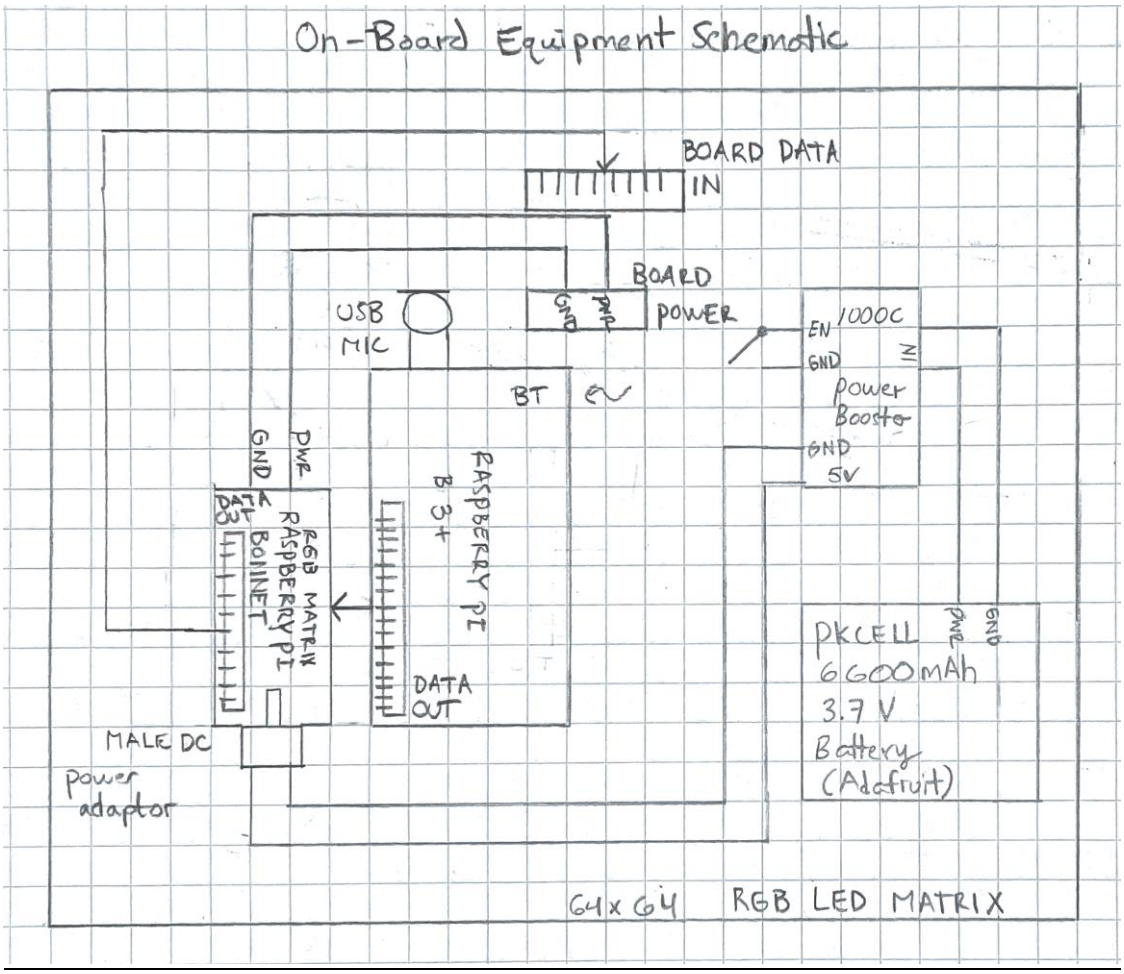

Figure 4: Power and Communication Electronics on LED Board

Description:

By the nature of the task, all power and communication electronics had to be packed behind the LED board, which imposed considerable constraints on the size and weight of each component. The single 6600mAh battery (Adafruit) powers both the Raspberry Pi 3 B+ and the 64x64 RGB LED Matrix (Adafruit). We control power using a simple switch on the 1000C Power Booster (Adafruit), which enables a connection between the battery and the components it supports. The Raspberry Pi takes in audio input from an on-board USB microphone, which is “plug and play” with the Pi. On start-up, the Pi initiates a program called Visual, written in Python, which then continuously streams and packages audio input into chunks of data. The visualization algorithm handles all signal processing techniques, including a fast-fourier transform, high pass filter, coordinate transformation, and finally mapping the LED states. At the output of the Pi are the LED state assignments, which are then translated to the LED board by the RGB Matrix Pi Bonnet, an add-on to the Pi made specifically for applications like this. The Pi Bonnet also transfers power to both the Pi and the Board.

Parameters Table

LED PWM Bits (of 11). Lower # means higher contrast, less CPU time and faster refresh rate. This is an input.	2
PWM Nanoseconds for LSB (300 high). Lower # means higher frame rate (good for recording purposes). This is an input.	50
Frequency Range per Bin from 120-500 Hz (Hz)	6
Frequency Range per Bin from 120-2500 Hz (Hz)	37
Frequency Range per Bin from 120-5000 Hz (Hz)	76
Frequency Range per Bin from 120-10000 Hz (Hz)	154
Frequency Range per Bin from 120-15000 Hz (Hz)	232
Frequency Range per Bin from 120-20000 Hz (Hz)	310
Max Current Draw by LED Board (A)	7.68
Typical Active Current Consumption by LED Board (A)	3

Feature 2:

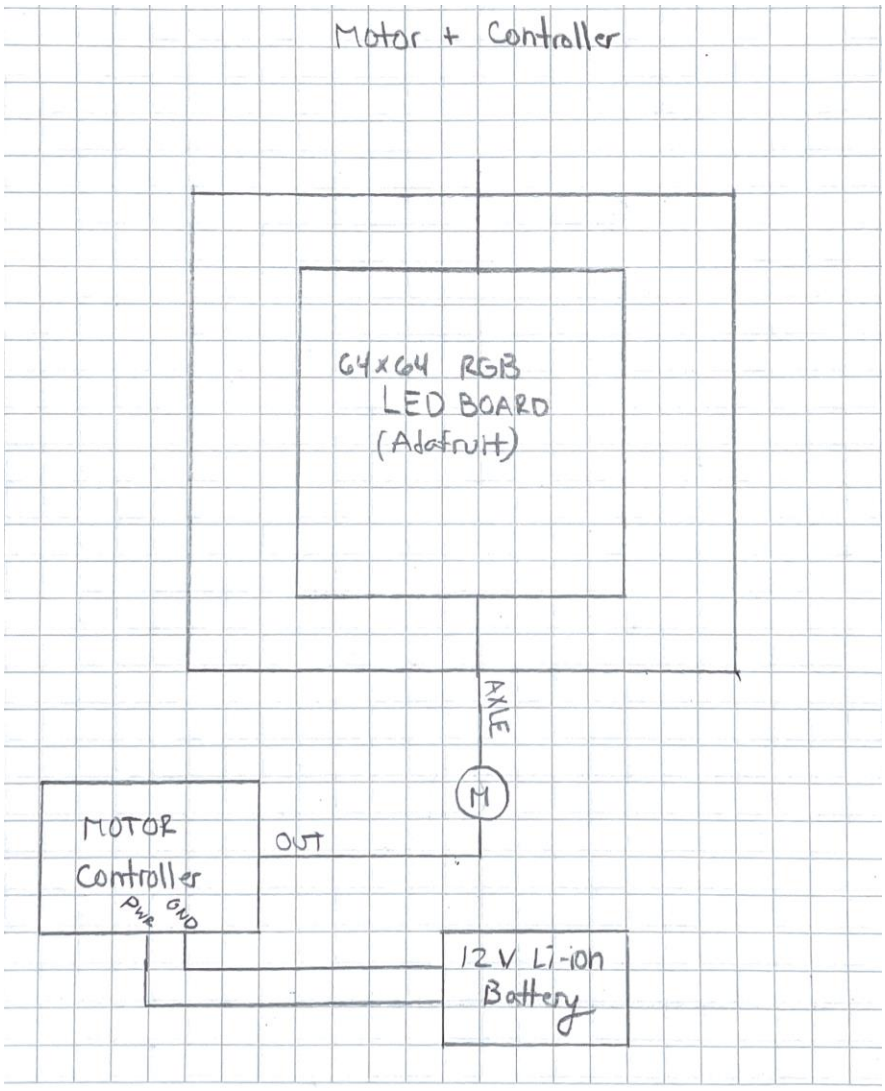

Figure 5: Motor Schematic

Description:
A 12V Li-ion battery powers the Denso Throttle Motor, part no. AE235100-0160. The motor controller is essentially a voltage divider which allows us to control rotational velocity. While the maximum speed is as high as 5000 rpm, we found that less than 2500 rpm was sufficient to get the desired visual effect while also mitigating the risk of parts flying off of the board.

Performance Parameter Table

Voltage (V)	12
Max Speed (rpm)	5000

Feature 3:

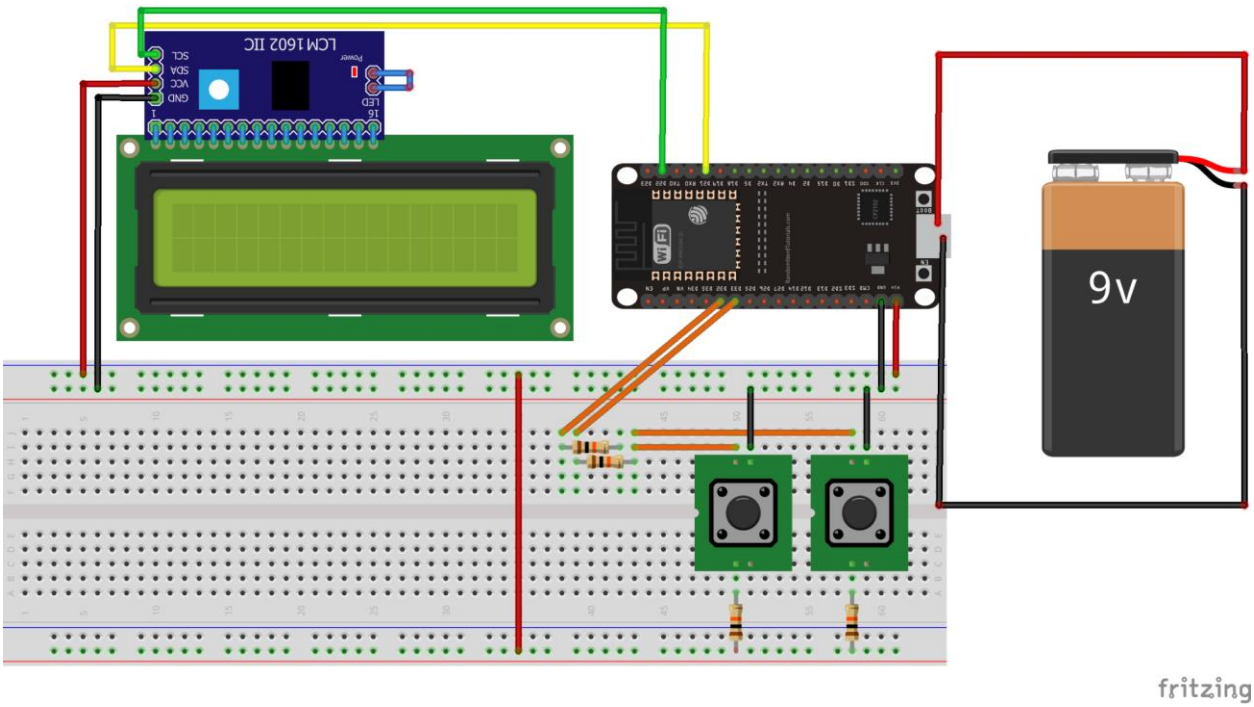

Figure 6: Bluetooth Controller Schematic

Description:
Users can use 2 push-buttons to control the maximum frequency displayable on the LED board. This is useful to find where the frequency content of input is, relatively speaking. Additionally, changing the displayed range of frequencies changes the shape of the visual. The ESP32 continuously checks the two input pins corresponding to the two push-buttons and updates the max frequency field's value when a button is pressed. A serial bluetooth communication channel exists between the Raspberry Pi and ESP32, which continuously sends the max frequency field's value to the Raspberry Pi (using the BluetoothSerial library). The Raspberry Pi checks for new frequency range values every ~10 seconds and then updates the LED matrix to display the current frequency range. This is done to prevent delay associated with constant retrieval attempts. Finally, the ESP32 also controls the output on the LCD display (which shows the range of frequencies on the LED matrix) using the LiquidCrystal_I2C library.