# Brain Controlled Drone: Final Report

Jialun Bao, Zirui Qiu,

Nicholas Gao, and Qing Xu

*Supervisor:*

Professor Toby Cumberbatch

Professor Dirk Martin Luchtenburg

**Abstract**

This paper explores the design process behind the Brain-Controlled Drone, a quad-copter system controlled by measurements of a user's cerebral intentions. First, we examine the approach of this project, which uses electrooculography (EOG) via a portable headset with multiple active-electrode circuits to acquire a user's neural activity. Second, we delve into the interpretation and classification of EOG signal data as directional commands. The signal processing and machine learning techniques. Finally, we examine how the translated commands are used to pilot a modified commercial drone, which is embedded with self-stabilization and other safety protocols. In all, the result is a successful system that tracks and interprets electrical signals and transmits the determined command to the drone, allowing it to navigate around a horizontal plane by moving forward and turning left or right.

*Keywords* - Electrooculography, neural network, machine learning, quad-copter control

# Contents


**1 Introduction** **1**

**2 Hardware Design** **4**

2.1 Electrode . . . . . 4

2.2 Customized Wearable . . . . . 5

2.3 Acquisition Circuit . . . . . 6

2.4 Prototype Board . . . . . 8

2.5 Customized PCB . . . . . 9

2.6 Discussion . . . . . 10

**3 Data Collection** **12**

3.1 Data Sampler . . . . . 12

3.2 Data Transfer . . . . . 14

3.3 Data Filtering . . . . . 15

3.4 Data Readout . . . . . 15

**4 Data Analysis and Feature Extraction** **17**

4.1 Training Neural Networks . . . . . 17

4.2 Neural Network Architecture and Test Score . . . . . 18

4.3 Principal Component Analysis . . . . . 21

4.4 Wavelet Decomposition . . . . . 22

4.5 Comparisons of Different Methods . . . . . 23

**5 Drone** **24**

5.1 Hardware Selection . . . . . 24

5.2 Developing a Controllable Computer Mouse Cursor . . . . . 25

5.3 Hacking the AR Drone 2.0 . . . . . 26

5.4 Drone in Action . . . . . 27

**6 Conclusion** **30**

**7 Acknowledgements** **31**

**Appendices** **34**

**A Background on EEG** **34**

A.1 Functional Areas of Brain . . . . . 34

A.2 Generation of EEG . . . . . 35

A.3 Frequency Decomposition of EEG . . . . . 36

A.4 Electrode . . . . . 37

A.5 Electrode Placement . . . . . 38

**B Simulation** **40**

B.1 Data Generation . . . . . 40

B.2 Feature Extraction . . . . . 41

B.3 Neural Networks . . . . . 41

B.4 Simulation Results . . . . . 42

B.5 Other Machine Learning Considerations . . . . . 43

**C Old Prototype** **46**

C.1 Electrode . . . . . 46

C.2 Ultracortex "Mark IV" EEG Headset . . . . . 48

C.3 On-headset PCB Design . . . . . 49

C.4 Tests on INA118 Instrumentation Amplifier . . . . . 52

C.5 Main Board PCB Design . . . . . 52

C.6 PCB Fabrication . . . . . 55

C.7 Discussion . . . . . 55

# 1 Introduction

In today's world, technology is pushing the boundaries of what humans can achieve and modern developments are making technology accessible to increasingly larger portions of the human population. With this picture in mind, this project is centered on the development of a mind-controlled quadcopter drone system. Overall, it is a multifaceted challenge involving the acquisition of electrooculograms (EOGs) and electroencephalograms (EEGs), signal processing, and drone control. This project attempts to explore these areas by: first (i), examining the nature of brain signals, how EOG and EEG measurement occurs, and how measurement hardware is assembled, second (ii), investigating how EOG and EEG data can be parsed and analyzed, and then how it can be classified or characterized to reflect the brain activity that actuated it, and third (iii), evaluating the applications of such a technology, which in our case, is developing a flight-ready drone that can interface with the bioelectrical signals. While the concept of electroencephalograms has been around for decades, their use to identify specific brain activity and thought, or let alone to control other technology, is still an open challenge, with many modern researchers failing to show the necessary specificity when interpreting EEGs to make them useful.

The impact of this research is strong and widespread. A compelling case is made by modern prosthetics that can be controlled by the wearer. The ability for the prosthetics to measure the neural activity and move according to the wearer's intentions adds significant improvements to the quality of life of a disabled individual. They can possess the autonomy that normal individuals possess, now that they have a limb that can move naturally. However current prosthetics use electromyography (EMG), which measures motor or muscle activity. What if we want to control something that is not naturally part of our bodies? There would be no EMG to measure for that, so this is where EEG becomes important. It serves as a measure of brain activity. Using this, the goal is to derive an individual's intentions and thoughts. A technology like this could increase people's productivity and give them broader control over their environment with a faster, more natural response. Along these

lines, it is helpful to discuss an interesting use case for this prospective technology. One study that developed mind-controllable computer cursors stated that their inspiration for this area of work was astronauts operating in space [1]. Astronauts are often in tight areas and wearing clunky pressurized suits and gloves. However, they are still required to use computers and perform complex tasks. An EEG interface would help improve the astronaut's limited accessibility to their spacecraft's controls and give them the wider range of actions that is normally restricted by the confines of their suits.

During the course of the project, we faced some major challenges. While we managed to assemble a PCB board, preliminary tests showed it could only measure a clean EMG signal. It performed unsuccessfully for detecting the EEG signal due to various noise sources. As a result, we considered the possibility of adding shielding methods to combat the noise, but this would come at the cost of making our device very bulky. Hence, we decided to shift focus and measure signals through electrooculography (EOG). This decision was inspired by a Korean study that attempted to achieve the same goal of an EEG controlled drone [2]. In this study, EEG proved to be equally as intractable, so they stopped attempting to measure electrical signals. Instead, they used cameras to track eye-motions as input. However, unlike the Korean study, we wanted to stay true to the goal of EEG and push the current technology that is used to measure neuro-electrical signals. Consequently, we chose to track eye motions without a camera, but instead through EOG, which measures the electrical signals passed from the brain to the eye muscles. Thus this project brings humans another step closer to interpreting the neuro-electrical signals from EEGs, and will be directly applicable when measurement technologies can successfully record clean EEG signals.

Meanwhile, the signal processing and classification aspects of the project showed promising results. Besides confirming the in-feasibility of EEG, it was able to consistently classify the EOG signals correctly. Lastly, in regard to how the commands are communicated to the drone, the project uses a higher-level approach, in which, open-source wrapper methods (detailed later) are used to execute user's commands. In all, this project shows the viability of using neuro-electrical signals as inputs in

order to dependably control other technologies.

The remainder of the paper is devoted to our accomplished work and will follow a modular format similar to the scheme of our project. This scheme is depicted below in Fig 1. The order of the subsequent sections reflect this same sequence of events that the system uses to function. The signal is first acquired from the user via our bioelectrical signal acquisition board. The design and implementation of this hardware is discussed in Section 2. The signal is then sampled and converted into digital data and classified into a directional command using a neural net algorithm (CPU in Fig 1). The details of these steps are placed in Sections 3 and 4, respectively. The last step is the transmission of the selected command to the drone over WiFi. The quadcopter drone control and test results are presented in Section 5. Then, towards the end, Section 6 concludes the paper with a review of our accomplishments and potential future work. Lastly, the appendices present some alternative system designs from previous iterations, along with related background information.

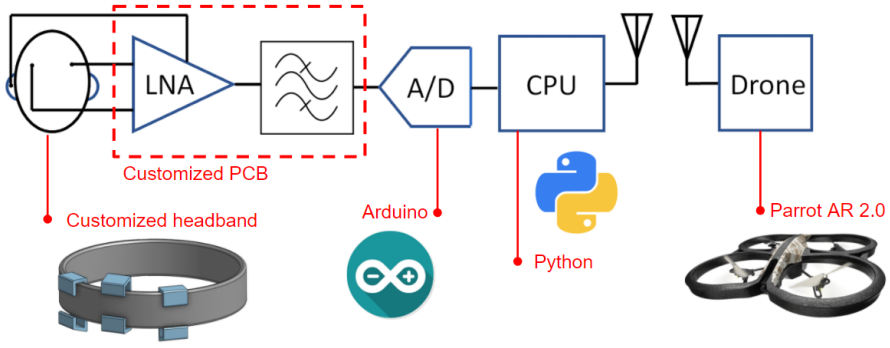


Figure 1: Overall system architecture

# 2 Hardware Design

In this section, we will discuss the first component of the system: the signal acquisition circuit. This makes up the left-hand portion of the system architecture given in Fig 1.

## 2.1 Electrode

Over the course of the project, we had many design iterations, using many different electrodes in attempt to acquire clean bioelectrical signals. One large problem we had with a previous prototype (discussed in Appendix C) was the instability of flex EEG sensors. To avoid this problem, we proposed gold cup electrodes as the replacement for the flex sensors. The cup electrodes are smaller in size compared to the flex sensors, which makes a more compact design possible. They are also reusable, although a clean-up of electro-gel is needed after each use. Other than the necessary maintenance effort, gold cup electrodes proved to be a very good alternative for the flex sensor.

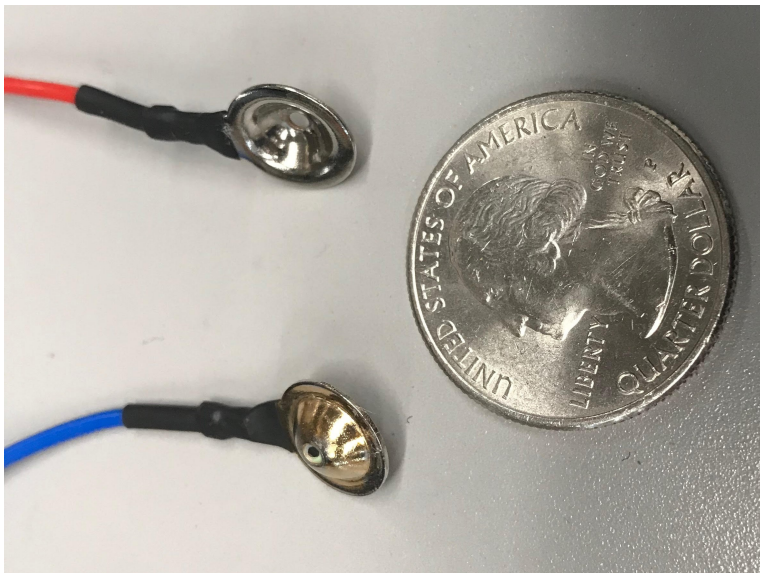


Figure 2: Gold cup electrodes

The cup sensors solved many of the problems that we encountered with the first prototype. First, when electrogel is applied, the electrode adheres to the skin and therefore provides an adequately stationary point of contact between the skin and the electrode. This allows the input waveform to remain on the same level across different experiment trials and future uses. In other words, we can rule out the chance that different degrees of contact caused any observed differences in the data across trials. Also, in terms of user experience, this electrode does not cause

discomfort, even after periods of extended use.

## 2.2 Customized Wearable

As explained in the last section, the gold cup electrodes with electrogel can adhere relatively well to the user's skin. However, shaking of the wires connected to the sensors and movements of the user's head can cause a change in the electrodes' positions, and even cause the electrodes to be detached from the user's skin. For preliminary experiments, electrical tape was used to ensure the gold cup electrodes remained in a fixed position, but electrical tape is disposable instead of reusable. In addition, the removal of the tape at the end of use will cause a certain degree of discomfort for the user. Therefore, a customized wearable was designed to secure the electrodes.

Fig 3 shows the positional placements of the gold cup electrodes on a user's head, with each red dot on the figure corresponding to an electrode. This arrangement of electrodes consists of two channels, each with 2 inputs, and therefore allows 4 classes of outputs. Channel one is comprised of the electrode on the forehead and the electrode above the left eyebrow, while channel two is comprised of the electrode on the forehead and the electrode above the right eyebrow (note that the electrode placed on the forehead is used as a input in both channels). The remaining two electrodes placed on the ear lobes of the user provide signal references.

Figure 3: Electrode Placement

As discussed earlier, we had to design a wearable that could fix the positions of the gold cup electrodes. Fig 4 shows a CAD of the headband assembly. The electrode holders, indicated in blue, are 3D printed parts. Each electrode holder has one open side (outward-facing) to make it removable in order to increase serviceability and allow for a quick assembly and disassembly. On the other side (skin-facing), a

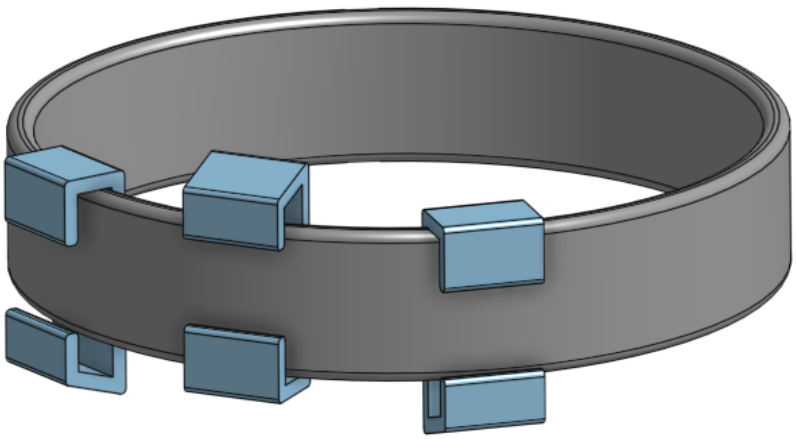

Figure 4: CAD Drawing for Headband

thin piece of foam is attached, which secures the gold cup electrodes in place, and simultaneously, provides comfort to the user because of its softness. The asymmetrical design allows the headband assembly to reach further out and cover more area if needed. Meanwhile, the headband, indicated in gray in Fig 4, is a commercialized elastic sport headband, which provides contraction to hold the electrode holders against the head. Furthermore, given the electrode holder's design, it can slide freely in a lateral fashion along the headband. This served as a convenient solution to when we experimented to find the optimal spots for electrode placement and for when we generalize to other users. In addition, two 3D printed clips are used to secure the reference electrodes on each of the user's ear lobes. Fig 5 demonstrates the manufactured customized wearable on a real user's head.

## 2.3 Acquisition Circuit

Once the sensors take a measurement, the raw analog data needs to be sent to a circuit for preprocessing, where the desired components can be amplified and extracted. The schematic for the acquisition circuit for one single channel can be found in Fig 6. 4 AA batteries with a total voltage of 6 V power this circuit. The battery voltage is regulated through a regulator to provide a stable 5 V voltage

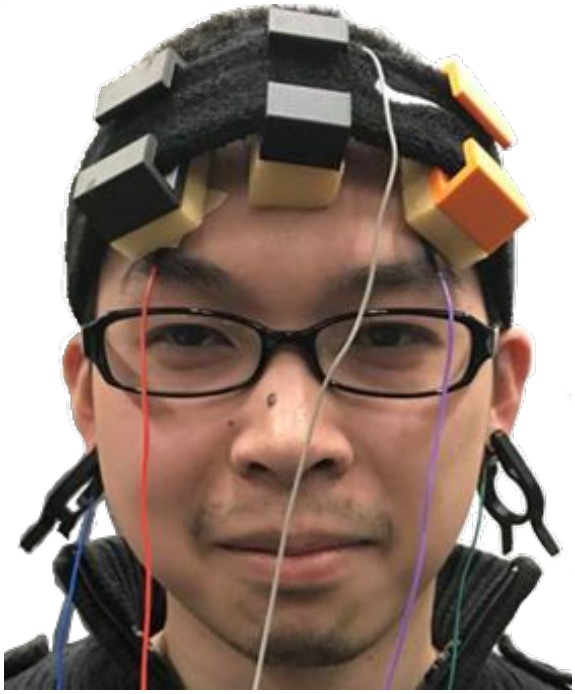

Figure 5: Customized Wearable Assembly

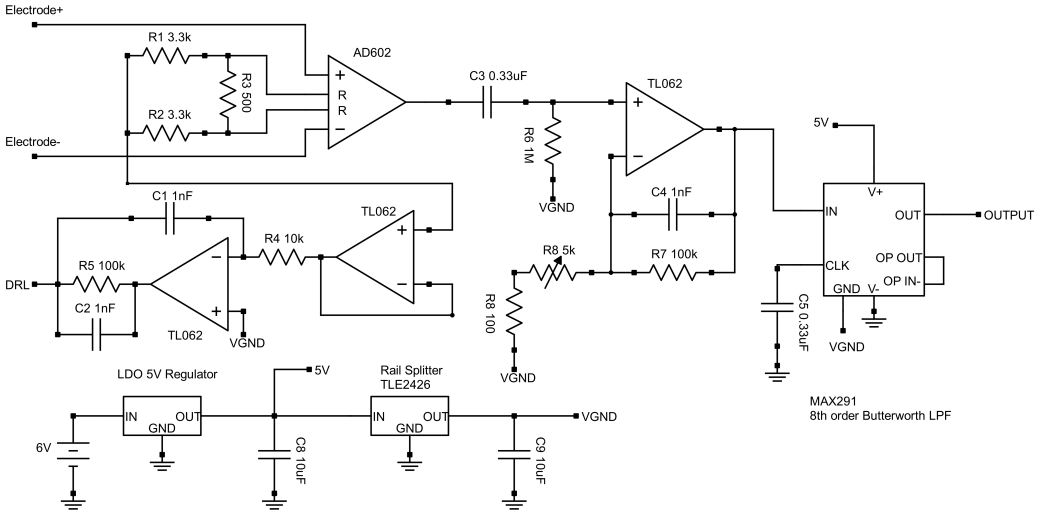


Figure 6: Overall schematic for the EEG acquisition board

output. The rail-splitter here generates a virtual ground of 2.5 V for the entire system, which is essential for a single supply system. The power supply system is shifted from double supply to single supply for the following reasons. Two coin cells of 3 V each were used in the original design, but they could only last for a few hours, then the voltage dropped. In addition, in a double supply system, a voltage

level shifter was required to shift the voltage level of the signal so that it could be detected by a unipolar analog-to-digital-converter (ADC). The differential paired signal is fed into the instrumentation amplifier (IA) AD602 with gain of roughly 40 dB. Because the output of the IA saturates around $\pm$ 1.7 V, the gain is slightly lowered than before to compensate the non-zero offset voltage. The passive DC high pass filter following the IA filters out the DC offset voltage. Then a variable gain amplifier further amplifies the signal, with a gain ranging from 26 dB to 60 dB.

As for the filter, we decided to use a Maxim 8th-order Butterworth low pass filter IC with a cut-off frequency set at 40 Hz. Compared to our filter design in the initial prototype, this new solution is much smaller in size and has significantly less component variation. Conveniently, the cut-off frequency is also adjustable in this version. After running tests with this design, the results showed this filter could effectively filter out the 60 Hz line noise exhibited in the previous iteration.

Lastly, the drive right leg (DRL) circuit negatively feeds the common signal back to the brain-skin to reduce the DC half-cell voltage between skin and electrode (details are discussed in Appendix C). It significantly improves the common mode rejection ratio (CMRR). In addition, the common mode signal is buffered in the DRL circuit, and thus it can be used for input guarding, a technique where the common mode signal is fed back to the shield the coaxial cable to reduce the capacitive effect of the shield.

## 2.4 Prototype Board

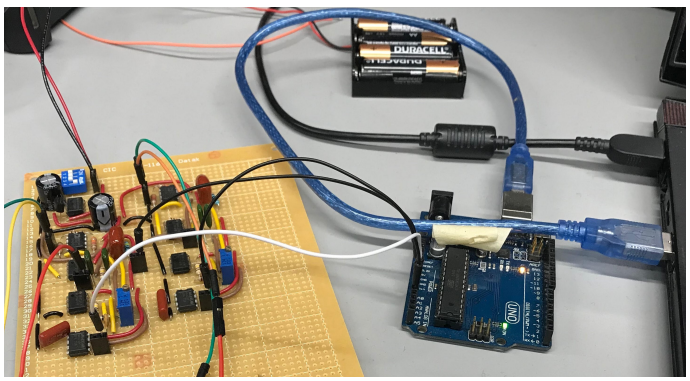


Figure 7: Second prototype board

Fig 7 shows our second prototype board, which is powered by a 6 V battery pack, and the two channel outputs are sent to a laptop via a micro-controller.

The components used for a 2-channel prototype board are tabulated in table 1.

| Component | Tolerance | Quantity |
|---|---|---|
| LDO 5V regulator | - | 1 |
| TLE2426 rail splitter | - | 1 |
| AD602 instrumentation amplifier | - | 2 |
| TL062 low power op-amp | - | 6 |
| MAX7480 butterworth LPF | - | 2 |
| 5 k precision potentiometer | - | 2 |
| 3.3 k resistor | 1% | 4 |
| 1 M resistor | 1% | 2 |
| 100 ohm resistor | 5% | 4 |
| 500 ohm resistor | 5% | 2 |
| 10 k resistor | 5% | 2 |
| 0.1 uF capacitor | 10% | 4 |
| 10 uF capacitor | 10% | 2 |
| 1 nF capacitor | 10% | 6 |
| 0.33 uF capacitor | 10% | 4 |

Table 1: Component lists for the second prototype board

## 2.5 Customized PCB

In order to improve the noise performance and to make the project more portable, we designed a board with EAGLE CAD. The boards were fabricated by a Chinese vendor called *jlcpcb.com*, and assembled in the senior lab. Figure 8 shows one fully assembled PCB with 2 channels.

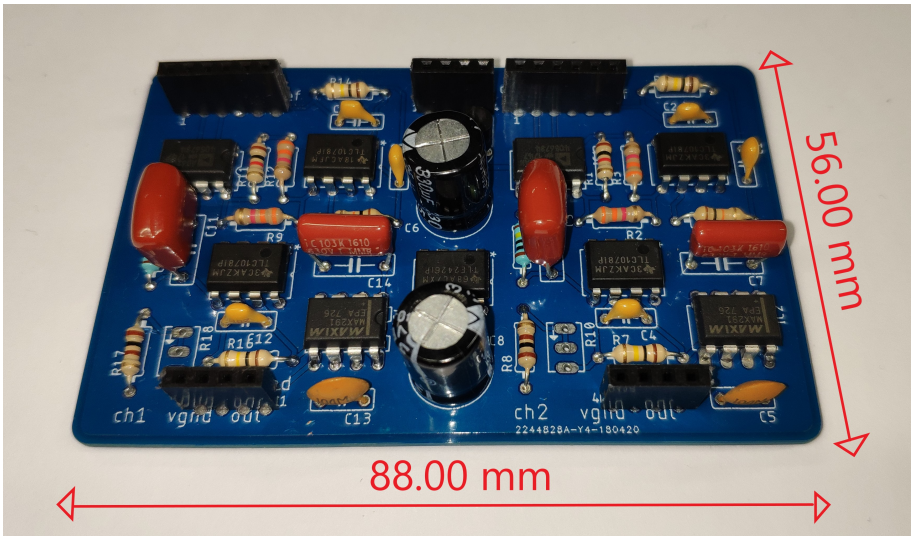


Figure 8: Assembled PCB

## 2.6 Discussion

This board is much smaller than the previous one (Appendix C), and we were able to obtain clean signals using this board. The 8th order LPF effectively filtered out the 60 Hz noise. The back of this board was electrically taped to prevent shorting between soldering contacts.

Despite the better overall performance, the final circuit triggered oscillation in some trials. We suspected that the electrode was the root cause of the instability because everything else in the circuit was unchanged across different trials. The skin-electrode interface impedance easily varies from day to day, based on the subject's changing skin conditions, the varying amount of electrogel placed in the gold cup, and even small changes in the placement of electrode. This varying impedance might move the system pole of our circuit to the right-handed plane and thus cause the instability. To improve the stability of the circuit, we can break 2 large gain stages into many small gain stages, but that would add more complexity to the circuit and require more power as well.

Another thing we noticed was that all active devices have flicker noise, which has an amplitude proportional to 1/f. In other words, the flicker noise is the most dominant at low frequencies. For bioelectrical signals in the microvolt range, its influence could be very significant. In the future, we can implement a chopping

amplifier to remove the flicker noise. In this case, we will bring the bio-electrical signal to some chopping frequency, amplify it, and then bring it back to the baseband and remove the flicker noise.

# 3 Data Collection

This section will discuss how the analog data are sampled and converted into digital data. Some design choices are also elaborated upon.

## 3.1 Data Sampler

Once the hardware extracts the desired components from the raw signal, it feeds it to the data sampler. We used an Arduino Zero board as our data sampler, and it mainly acts as a digital to analog interface by acquiring the signal at some desired sampling frequency. This board was chosen because it has an on-board 12-bit ADC. This allows for higher accuracy in the sampled voltages, in comparison to lower bit Arduino models, which can only store smaller, less precise values. For example, we observed that our signals exist in a 3.3 V input range. This means that the Arduino Zero can achieve a precision of less than 1 mV.

To illustrate and confirm this point, we ran a comparison test, attempting to sample simulated signals. Figure 9 shows the results obtained from a 10-bit and 12-bit ADC, respectively. The testing source was a sinusoidal wave at 15 Hz with 100 mV peak to peak. The left waveform in red was obtained from the 10-bit ADC, and at low voltage there was some form of glitches. The difference was not very clear at 100 mV when comparing peak to peak. However we were expecting to see more glitches, since in the case of our actual EEG signal, the voltages would be very low.

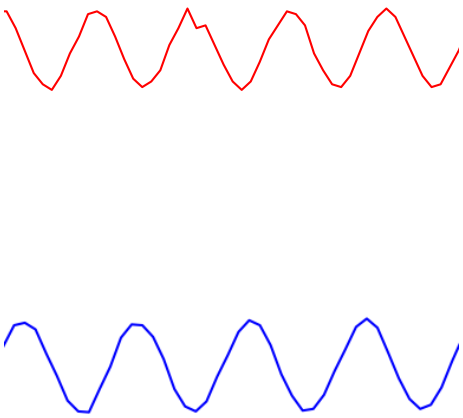

Figure 9: Comparison between 10 bits and 12 bits ADC

Furthermore, in our initial design, we had a simple DC biasing circuit to shift the DC voltage level of the output signal in a dual supply system. This meant that

the unipolar ADC could read the signal. However, in our second prototype, we switched from a dual supply system to a single supply system. The output signal was always sitting around the virtual ground, which was half the supply voltage or 2.5 V. Therefore, no DC biasing circuit was required for the final prototype board.

However, due to the presence of a non-zero offset voltage, the output signal was not exactly centered on 2.5 V. More so, the signal often varied, meaning we couldn't simply subtract the output signal by 2.5 V to decouple it from DC. Thus, instead of looking for a hardware solution, we solved this problem by subtracting out the raw signal's average DC value from the overall raw signal to obtain the pure AC signal form.

Moreover, an experiment was conducted to see the effects of sampling rate on the quality of the signal. Theoretically, we would like to sample as fast as possible to obtain a smooth signal with the best quality, but a higher sampling frequency means more samples per second. This translates to a high data transfer rate that can lead to congestion at the serial port and result in a significant transmission delay. This is something we wanted to avoid because we want a low latency system. You shouldn't have to wait too long to see the drone respond to your command. Also we don't want a queue of commands to accumulate, as the user would then no longer feel the real-time one-to-one control experience that we are seeking to create. Thus, a sampling rate of 300 Hz was experimentally chosen to compromise between signal quality and transmission delay.

Figure 10: 15 Hz vs. 30 Hz

Figure 10 shows the results obtained from 15 Hz and 30 Hz testing sources, respectively. Each source was running at 100 mV peak to peak. The left picture shows a perfect sinusoidal wave, but the right picture is slightly distorted, showing

the potential drop in signal quality that higher frequencies can cause. Nevertheless, a 300 Hz sampling rate was able to capture enough useful information from the signal without leading to any significant transmission delay or backup.

## 3.2 Data Transfer

Since Arduino has a very limited computational capability, the actual data were processed in a computer. Specifically, the data were first sampled by the Arduino board at 300 Hz, and then they were transferred through the serial port at 115200 bits per second to a python program running on a laptop. The total time delay in one cycle is given by Eq 1

$$\begin{aligned} T_{total} &= T_{analogread} + T_{transmissiondelay} + T_{transmissionoverhead} \\ &\approx 2 * 0.1ms + \frac{8 * 2 * 2 * 1000}{115200} ms \approx 0.5ms \end{aligned} \tag{1}$$

First at all, for the Arduino Uno with 10 bit ADC, there is 100 $\mu$s delay in each analog read, and each channel sends out 2 bytes of data at 115200 bps every cycle. For 12 bit ADC, the delay of AnalogRead command is expected to be longer. Due to the lack of detailed timing information of the ADC Zero, we just assume the delay remains at 100 $\mu$s. Since there are two channels in total, the total time delay is about 0.5 ms, assuming no transmission overhead. In reality, it is expected to be a little bit longer. In other words, the maximum sampling rate allowed is less than 2 kHz. This is one of the reasons why 300 Hz sampling rate is chosen to leave some flexibility to add more channels in the future.

Moreover, in order to avoid the buffer overflow problem as well as to synchronize the clock between Arduino and computer, a simple handshake protocol is implemented here. The Arduino only sends data to the serial port when an instruction is issued from the python program when it is ready to read data. Furthermore, to have a better visualization, this python program also plots the real-time data as shown in Figure 10.

## 3.3 Data Filtering

Because a huge gain was required to amplify the EEG signals, any motion artifacts, such as touching the wire or moving the forehead would cause large spikes in the data. Therefore, we would like to filter out the unwanted data. However, it was very difficult to come up with specific rules to automatically filter out the undesired signals due to the nature of the signal. Instead, we manually discarded the bad signals.

This interface not only reads data from the acquisition circuit board to the computer, but also allows one to monitor the acquired data, and then decide whether or not to store them. Every time data becomes available, the computer plots the sample data and waits for the user's decision. Upon a decision being made, it reads another block of data. This semi-automatic program helped us to collect meaningful data.

## 3.4 Data Readout

When we first began data collection, we put electrodes at various spots on Nicholas' head, hoping to locate a source of strong and distinguishable signals that could be classified with high accuracy. Since at this stage we were still pursuing EEG, we first tried the left parietal lobe, specifically the motor cortex, which is near the left ear. The left motor cortex is responsible for any right hand side body motion. Unfortunately, when Nicholas was asked to lift, move, and shake his right hand, no significant response from the left parietal lobe was observed. Instead, the signal seemed very arbitrary. The first few attempts were not successful, and not until we moved the electrodes to the frontal lobe, near the forehead, did we observed some spikes when Nicholas was blinking his eyes regularly. There was a strong observed correlation between the blinking of the eyes and the voltage spikes in the signal. This was another major influence in the decision to shift the project to focus on EOG, as a precursor for later work on EEG.

Hereafter, we made some further explorations and determined five distinguishable classes of signals corresponding to the five subsequent actions: blinking both

eyes, blinking the left eye, blinking the right eye, rolling the eye up and down, and keeping eyes opened/idle. An example of the measured voltage over time from each side, left and right eye muscles, can be seen in Figure 11.

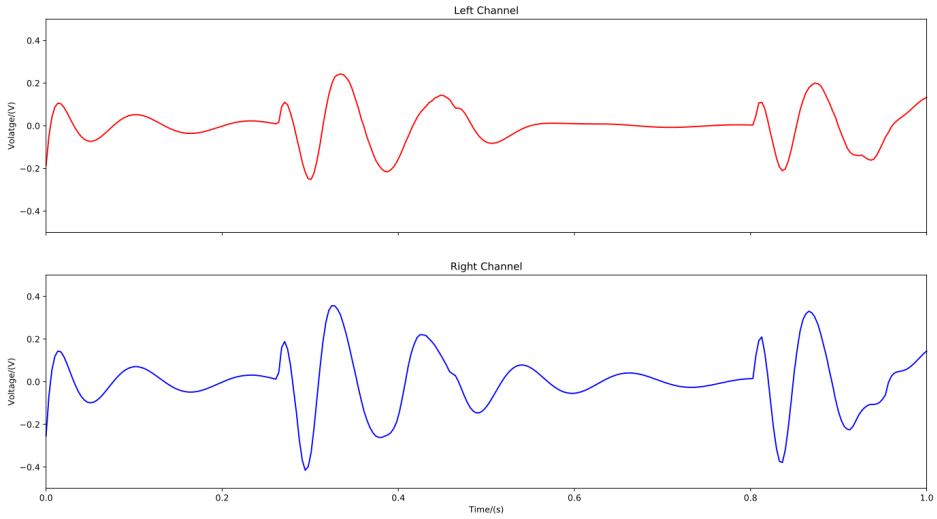


Figure 11: Jialun's blinking both eyes

Convinced that this EOG design was viable, we then compiled a training/testing dataset by recording Jialun performing the five aforementioned eye actions. The extraneous components of the data were manually filtered out as discussed earlier. In the end, 300 samples of data, each with a 1 second duration, were collected for each class of signals. In total, with 5 classes, we had 1500 samples of labeled data that were representative of their respective eye movements.

# 4 Data Analysis and Feature Extraction

Though we already discussed some filtering and extraction steps for the signal, these were all preprocessing steps. At this point, we have a signal that can be used for classification into commands, but before that, we need to choose what aspects of the signal should be used.

In this regard, even though our initially proposed feature extraction method was wavelet decomposition, simple fast Fourier transform (FFT) worked surprisingly well on the EOG signal data. Consequently we decided to stick with FFT, and leave wavelet as an alternative feature extraction method. The output of the transform was then inputted into a neural net for classification, as will be discussed in the next section.

## 4.1 Training Neural Networks

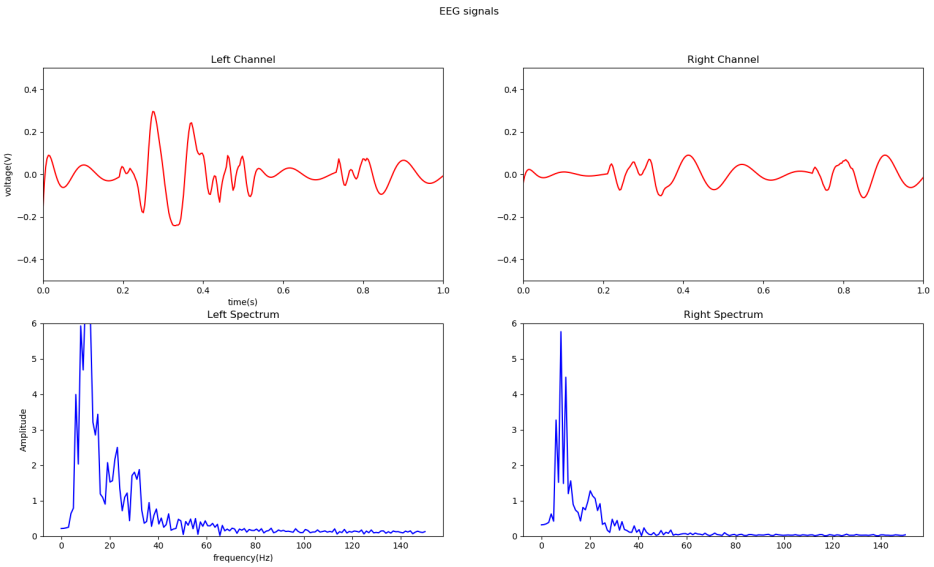


Figure 12: Measurements from Jialun blinking left eyes

As mentioned, each collected sample is processed using FFT. Figure 12 and Figure 13 show the signals and their corresponding frequency spectrums for when Jialun is blinking only his left eye and when he is rolling his eyeballs up and down, respectively. Fortunately, all 5 classes of signals are seemingly very distinguishable

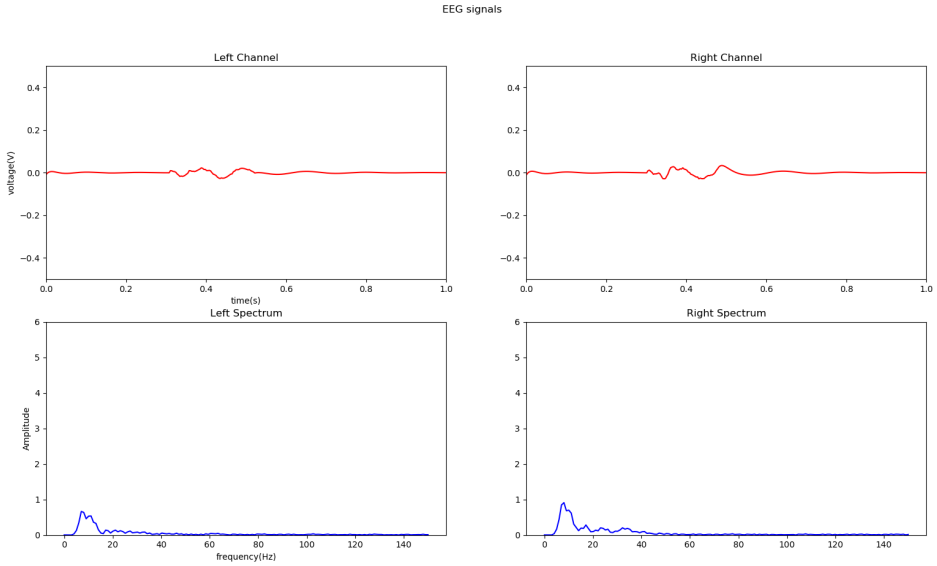


Figure 13: Measurements from Jialun moving his eyeballs up and down

through a simple observation by eye.

Unfortunately due to the low sampling rate, the frequency resolution isn't very great, as one can see in Figure 12 and Figure 13. However, instead of increasing the sampling rate, which would potentially affect our system architecture, a moving average is applied to the frequency spectrum to smooth out the response. Specifically, a moving average is implemented with a window size of 5 (i.e. each frequency's amplitude is a sum of the amplitude values of the 5 frequencies measured around the desired frequency. The smoothed spectrum is then decimated every two points, or in other words, down sampled by 3. Figure 14 illustrates such an action. When moving from the top spectrums (blue) to the lower spectrums (red), only every third point is kept. Afterwards, 12 frequency-amplitude pairs are obtained from each channel (i.e. each red graph contains 12 points). With two channels, these pairs are concatenated to form a 24-feature vector for each 1-second observation.

## 4.2 Neural Network Architecture and Test Score

Subsequently, this means that each sample in our labeled 1500 sample dataset had these 24 features. We then used 90 % of the samples to train a neural network and used the remaining 10 % to test the performance of the now trained neural net.

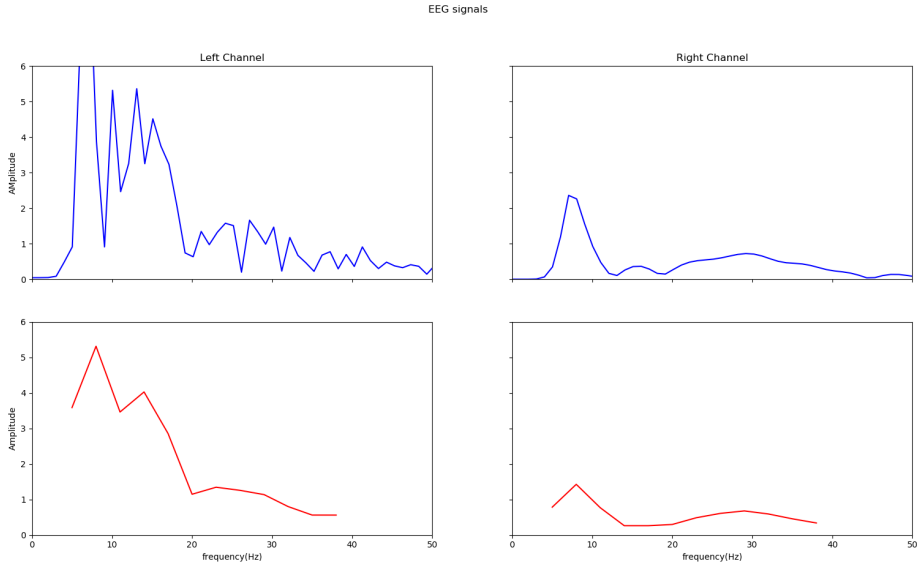


Figure 14: Applying moving average and down sampling

The design of the neural net (e.g. number of hidden layers and nodes) was rather arbitrary and parameters were chosen based on what produced the best results during validation.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.90 | 0.82 | 0.86 | 22 |
| 1 | 1.00 | 0.93 | 0.96 | 28 |
| 2 | 0.89 | 0.94 | 0.92 | 36 |
| 3 | 1.00 | 1.00 | 1.00 | 34 |
| 4 | 0.94 | 1.00 | 0.97 | 30 |
| avg / total | 0.95 | 0.95 | 0.95 | 150 |

Figure 15: Test Score for 5 classes

Fig 15 shows the classifier achieving an average accuracy of around 95 % for 5 classes of signals. A "lbfgs" solver is used in this neural network classifier to assist in the learning process. In addition, the neural network inherits its structure from previous simulations carried out prior to us being able to record bioelectrical signals: 6 hidden layers and 130 hidden neurons (details are in Appendix B).

Although the training result looks very promising, it fails to classify and predict all of the real-time signals. In particular, it could detect four of the classes: the blinking of both eyes, the blinking of the left eye, the vertical rolling of the eyeballs, and keeping the eyes open, but on the other hand, it couldn't distinguish the final class, corresponding to the blinking of only the right eye. It often confused it with the blinking of both eyes. Later on, however, we found out that Jialun couldn't properly blink his right eye without blinking his left eye to some degree. Thus we decided to attempt using only the 4 successful classes of signals. Consequently, in order to provide the maximum freedom of motion, we decided to map those 4 signal classes to 4 different drone commands: moving forward, turning left, turning right, and stopped/idle. With these commands, the drone can still achieve full 2-D planar motion, despite using only 4 commands.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.00 | 0.00 | 0.00 | 37 |
| 1 | 0.00 | 0.00 | 0.00 | 30 |
| 2 | 0.27 | 1.00 | 0.43 | 25 |
| 3 | 1.00 | 1.00 | 1.00 | 28 |
| avg / total | 0.29 | 0.44 | 0.32 | 120 |

Figure 16: Test Score for 4 classes (underfitting case)

After the removal the blinking right eye class, we performed another round of training. However, the training result was much worse than the previous one, as shown in Fig 16. It turns out that the model is under-fitting, since there are only 1200 samples.

After experimenting with different structures and simplifying the neural network, we finally arrived on 2 hidden layers and 20 hidden neurons. With this setup, the average correction rate is about 94 %, almost matching the success of the first neural network with 5 classes. This shows that the four-class trained model can viably classify real-time signals. For instance, this means that when Jialun is blinking both of his eyes, the neural net correctly identifies it as belonging to the first class

and the respective command is transmitted to move the drone forward.

## 4.3 Principal Component Analysis

To further verify the chosen features and validate the 94 % correction rate of our neural network, we performed principal component analysis (PCA) on the data to examining whether or not they are separable. PCA is essentially a dimensionality reduction tool, that calculates the first few largest variances along with their directions. In our experiments, only the first two principal components are used in the interest of producing a 2D scatter plot. From Fig 17, one can see that 4 classes of signals form 4 separable clusters. From this plot, it is easy to infer that in a higher dimensional space, the 4 classes would be easily separable, verifying that our neural network's performance is reasonable.

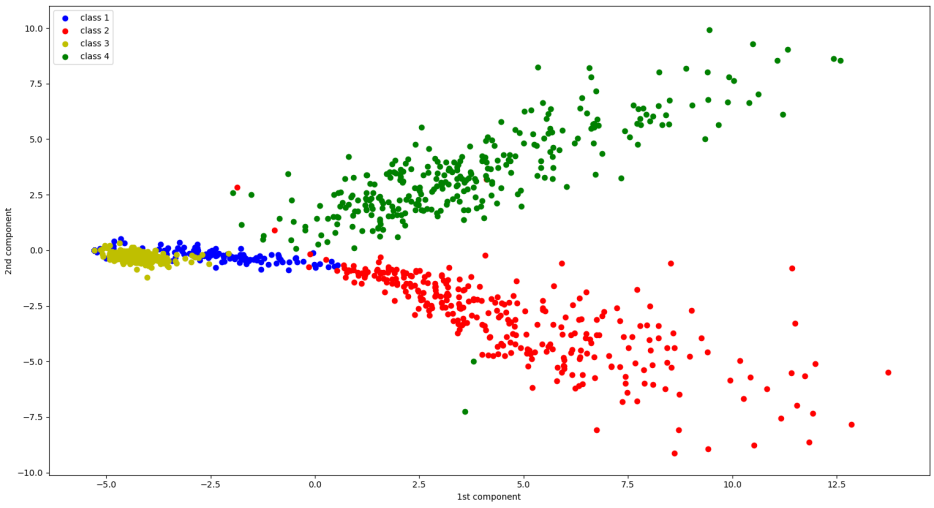


Figure 17: PCA on extracted features

Inspired by this, we decided to use this dimension reduction tool to reduce the dimension of the features. From Fig 17, it is possible to add a few more principal components to separate the clusters. Since the dimension of the features is greatly reduced by PCA, a support vector machine (SVM) might be a good candidate for the classifier.

After several trials, it was found that the classification with the first 4 principal components yielded the best result. This classification scheme achieved average correction rate of 73.5% in 20 training runs. Since this did not beat our neural net's performance, we decided to keep using the neural net. A more detailed comparative analysis will be provided later.

## 4.4 Wavelet Decomposition

Our original proposed feature extraction method was wavelet decomposition, which would decompose the signal into 4 levels. Fig 18 shows an example of using wavelet to decompose the signal. As mentioned in the earlier section, only detail space 2, 3, and 4 are used for further analysis.

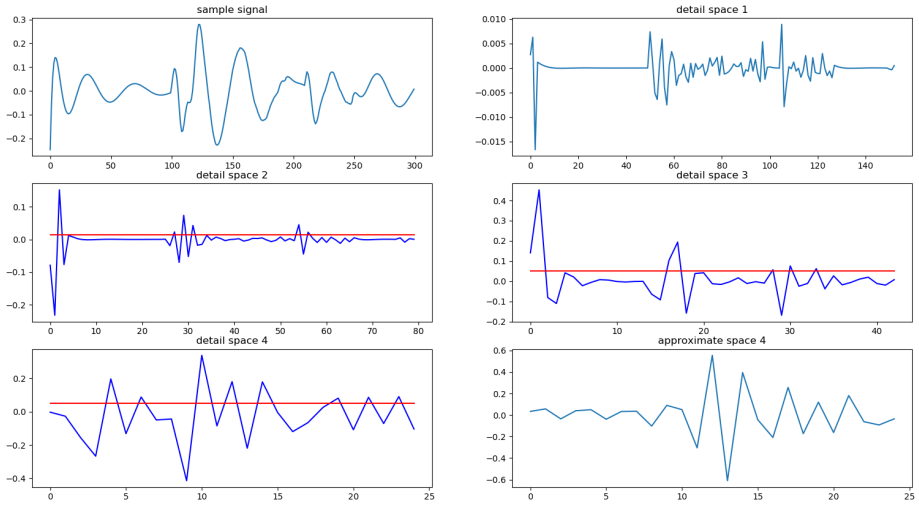


Figure 18: Wavelet decomposition on a sample signal

The detail coefficients from those bands are then simplified through a simple statistical trick: mean absolute value. The red line in Fig 18 indicates the mean absolute value at the corresponding frequency band.

Finally, the concatenated mean absolute values(a vector of length 6) from both left and right channels were used to train a neural network with only one hidden layer (5 hidden neurons). An average correction rate of 95.7% was achieved over 20 training runs.

## 4.5 Comparisons of Different Methods

| Method | FFT, and NN | FFT, PCA, and NN | Wavelet decomposition, and NN |
|---|---|---|---|
| Average correction rate in 20 training runs | 97.25% | 73.5% | 95.7% |

Table 2: Comparisons of different methods.

With limited sample data and limited training runs, it was very hard to conclude anything. Solely based on the data we had, it seemed that FFT and wavelet decomposition performed equally well, but FFT is much simpler and efficient than wavelet decomposition. This gives FFT a slight edge over wavelet.

As for PCA, it is meant to reduce the dimension of features so that it could potentially reduce the classification time by using simpler classifier such as SVM. The goal is to do more preprocessing on feature extraction and therefore do fewer computations in classification. However, the average correction rate for the PCA and SVM combo wasn't really too great. It is very likely due to the fact that only limited sample data was used to train SVM.

As a result, the simple FFT method coupled with a neural network was chosen as our feature extraction method.

# 5 Drone

The third and last part of this project is to use the classified signal to navigate the drone via Wi-Fi.

## 5.1 Hardware Selection

The last part of this project is to transmit the classified signal to navigate the drone via Wi-Fi. The Parrot AR Drone 2.0 Power Edition was chosen to be the actuation of the brain- controlled drone system (Figure 19). Firstly, it is a quad-copter. Quadcopters, as opposed to fixed-wing UAVs, have the advantage of ease of control and fast response to position controls. It also has an easier dynamic modeling than the fixed- wing UAVs. Secondly, because quad-copters do not have an energy-efficient way of creating lift, they usually have poor stamina. Parrot AR Drone 2.0 Power Edition gives the drone three times more battery life compared to other commercialized drone models. With 1500mAh HD batteries, the drone can fly for up to 36 minutes per run.


Figure 19: Parrot AR Drone 2.0 Power Edition

## 5.2 Developing a Controllable Computer Mouse Cursor

Before any design decisions were made about the drone or about hardware and software, we first investigated methods others used to control a mouse cursor with EEG signal.

At a higher level, by bridging the gap between the brain and computers, our project pushes an area of research that lends itself to developing new technologies that offer a more intuitive and natural interaction [2]. Moreover, there was an interesting takeaway from the aforementioned study that was inspired by the need to aid astronauts [1]. They used an EEG-based computer-brain interface to control a 1D and 2D cursor on a display. The paper outlines two main goals. The first being to "augment human–system interaction in wearable, virtual and immersive systems by increasing bandwidth and quickening the interface", and the second being to "enhance situational awareness by providing direct connections between the human nervous system and the systems to be controlled" [1]. Since these objectives align closely with those of this project, it serves as a good project to reproduce. As a result, an appropriate short-term goal would be to reproduce the EEG-controlled cursor. This allows us to focus on one dimension at a time. We can then test and evaluate the EEG control before the drone component is finished. Lastly, but equally as important, we can avoid unnecessary damage to our drones. With this idea in mind, we have begun developing an interface that allows you to control a computer's cursor. This was done with assistance from Al Sweigart's guide on automating computer tasks [3]. Using this tutorial, we reproduced his exercise, which is depicted below (Figure 20). It shows how you can guide the cursor with Sweigart's Python library in order to draw out a square spiral. With this interface, it will be easier to connect the EEG-signals to commands for the cursor.

This experiment with Sweigart's python library was conducted as a preliminary step and a quick verification tool for the classified signal from the previous component of the system.

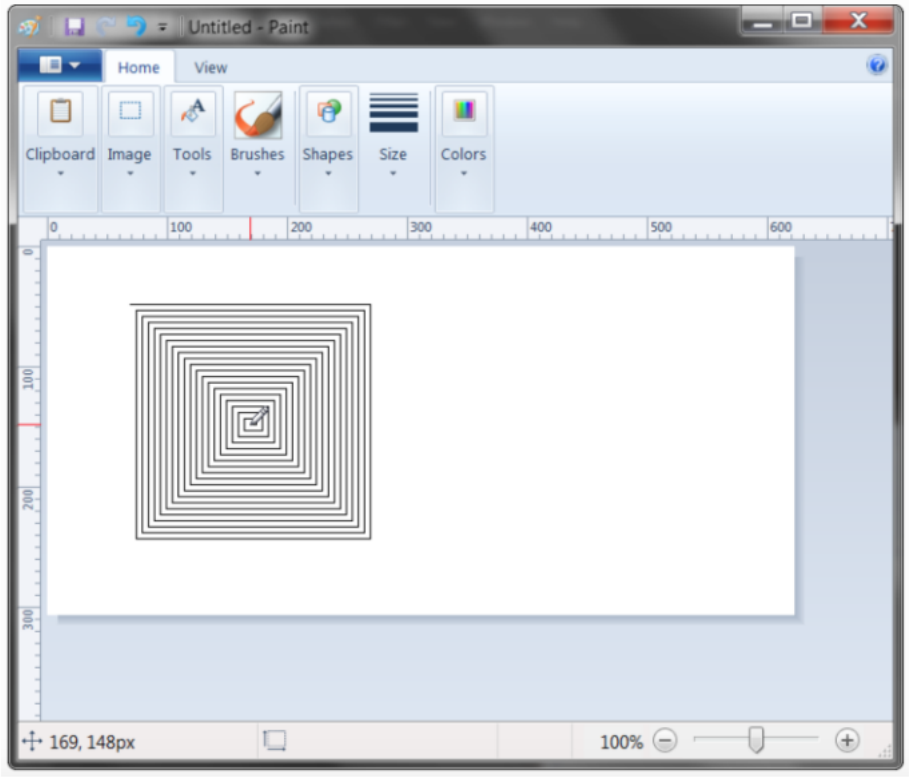


Figure 20: 2-Dimensional Mouse Cursor Control with Sweigart's Python Library [3].

## 5.3 Hacking the AR Drone 2.0

The first option for the new drone was to use the software development kit (SDK) provided by Parrot. This allows for data to be taken from the drone's sensors and for commands to be given to the drone. The downside to this is that all the control algorithms we had written in Matlab would need to be manually re-coded in C++. All the built-in control code and MATLAB matrices would need to be reformulated as new C- style methods and data structures.

On the other hand, another available option was to use the provided flight control algorithm that comes with the AR Drone 2.0. With this as a base, we can simply apply a command interface layer on top. In practical terms, this means that the drone will be able to hover in place autonomously using its existing firmware. Then, when the user gives a directional command, the drone will respond and transition to the new location. This approach can also be solved using the Parrot SDK, however unlike the earlier approach, there are many application programming interfaces (APIs) that simplify the use of Parrot's SDK for these purposes. The APIs allow a

programmer to request directional commands that are converted, by the APIs, into the necessary methods from Parrot's SDK, so that the drone can understand the instructions it is receiving. If we wanted to follow the original plan of having our control algorithm handle the drone completely, we would need to start at the SDK level, since APIs aren't made for designing new custom control algorithms.

Through experimenting with flying the drone, we found the built-in controls on the Parrot AR Drone fulfills the basic safety requirement with its outside hull separating the propellers from the ambient environment.

Nevertheless, while we found many API libraries, we chose to use PS-Drone, a Python library developed by J. Philipp de Graaff [4]. This choice was made because the neural net that classified the electrical signals was developed in Python. Using the same computer language eases the process of combining the two processes, specifically the neural net classification algorithm to the respective drone control commands. Moreover, this package not only provides methods to make translational movements, but also provides an interface for the computer to send real-time, in-flight commands to the drone. To have more freedom of control over the drone, we programmed in Python using this API to control the drone through keystrokes. Using these tools, we recreated the interface, assigned computer keys to each of the 6 directions in 3D, and attempted to fly the drone. The motivation behind the test was that the keystrokes are essentially an emulation of controlling the drone using classified bioelectrical signals. This trial was done merely to verify that the PS-Drone library works reliably on its own. The next step was to remove keystroke commands and instead have the Python program call the piloting functions, depending on the observed output of the classification algorithm. For example, when the neural net classifies a double blink signal as class 1, the Python script should call the "Move Forward" function in the PS-Drone library.

## 5.4 Drone in Action

Again, as briefly mentioned previously, the 4 signals tied to drone motions are: blinking both eyes (moving forward), blinking left eye (turning right), vertically rolling the eyeballs up and down (turning left), and keeping the eyes opened or at

rest (idle).

To make the drone fly more smoothly, we intentionally stop the drone for a half second after every command to ensure the system has enough time to collect and process the most current data. This is necessary because otherwise the drone would just keep flying according to its last command until it receives the new command, which would arrive at least a second after the first command due to the latency of the system.

In this regard, the drone should fly in a stable manner, especially in the interest of safety. Thus each command that leads to an actuated motion, only leads to an incremental change in position or orientation. To control each of these movements, there are two parameters that can be set: gain and time duration. Gain determines the velocity after a command is received, the higher the gain is, the faster the drone moves. Time duration determines how much time the drone is in that movement. Together, these two values determine how large a distance or angle the drone moves when a command is received. As a result, these two parameters can be tuned for different test environments. For example, to operate in a confined space like a classroom, both gain and time duration should be reduced to minimal values to ensure precise control of drone's position.

In addition, to build out the robustness of the flying experience, we implemented a moving average on the flying commands. We make each command a one-hot vector of length 4, and also keep track of the current and previous two commands. The combined weighted average would make the current decision for the drone. For example, under the original scheme, if the drone is currently stationary, and then Jialun unintentionally blinks both of his eyes, the drone would in theory move forward. However, with this moving average feature, we could avoid this kind of erroneous situation. To illustrate this idea, we consider the case where the drone has been stationary; since the previous two states affect the current decision, Jialun has to blink his eyes for two consecutive commands in order to shift the moving average toward the "move forwards" classification. The result of this addition is a robust system that can differentiate between involuntary eye movements, such as reflexive and spontaneous blinking, and voluntary eye movements, such as intentional

blinking.

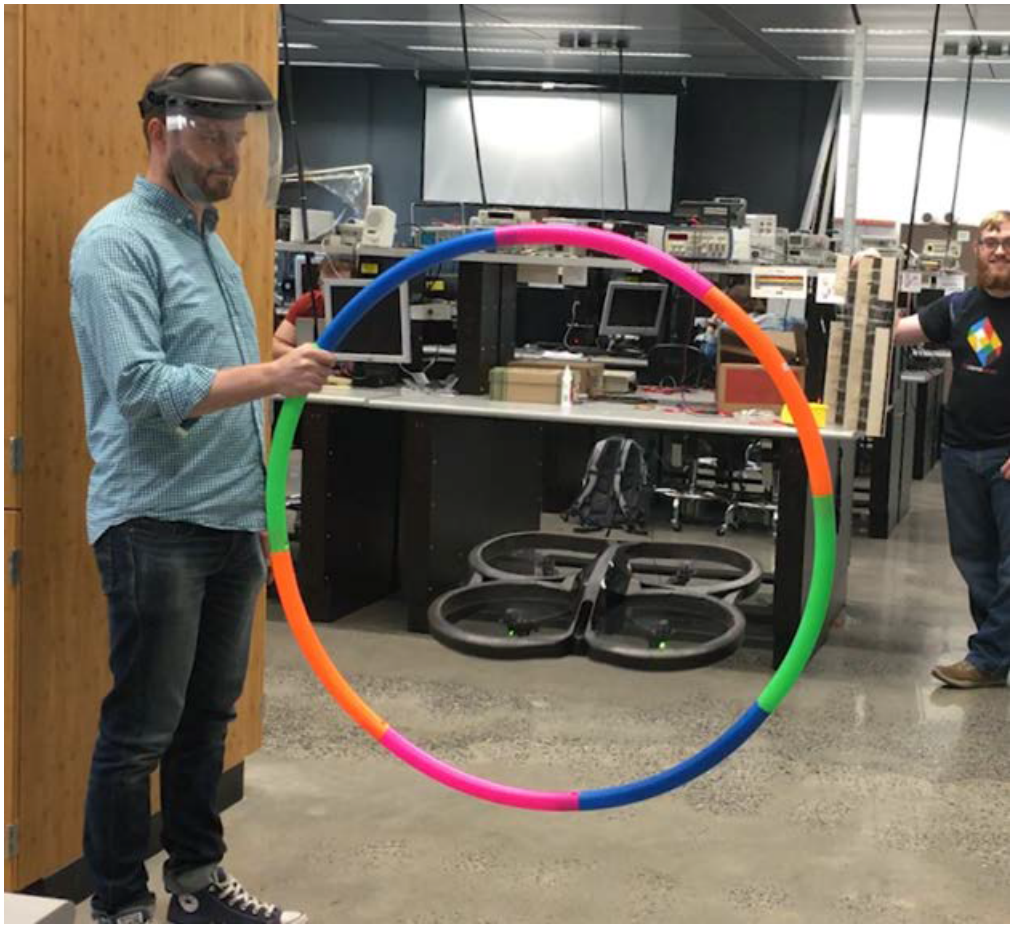

Figure 21: Operating the Drone through a Hula Hoop

Altogether, the finished system was tested in multiple lab experiments. We found that the system works effectively to take advantage of EOG in order to navigate the drone through various paths toward a target. We demonstrated its success through several trials consisting of flying the drone around obstacles and corners to reach a hula-hoop target. A depiction of the successful trials can be found in Figure 21. As well, a demonstration video of the trials can be found at the following link *https://youtu.be/43b-u2KSLVI*. In the video, several successful trials of the drone flying through the hula-hoop are shown with the synchronized footage of Jialun controlling it through EOG. In addition, the video also includes a demonstration of the individual commands as received by the drone, as well as another section demonstrating the voltage and frequency responses synchronized to the footage of Jialun's eye movements.

# 6 Conclusion

Given the successes of the system, we show that control through bioelectrical signals is not just a theoretical possibility. This project serves as a proof-of-concept that EOGs, one of the many forms of bioelectrical signals, can effectively control another piece of technology (e.g. drones). In the future, when the technology advances such that humans can measure EEGs reliably, this system design can be extended to generate commands based on the signals corresponding to a user's thoughts.

In the meantime, potential future work pertaining to this project would be to generalize the system to more users. To achieve this, we would need to gather sample data from new users, so that the system can recognize the same four classes of signals, but account for differences among individuals. Likewise, another route to pursue would be to optimize the electrodes and their placement. To this effect, it would also be useful to characterize our system's extendibility to other bioelectrical signals, such as EMG. We could then assess if the new data can augment the performance of our system.

In conclusion, it is interesting to consider the ethical challenges that this project introduces. Some may argue in opposition to this project, asking if this technology is necessary. Will it not encourage laziness or perhaps remove the privacy of one's own mind? We answer this by saying that the technology is still too immature to introduce ethical conflicts. At the current state, the benefits to humans, in augmenting our capabilities, outweigh the current possible downsides. In this regard, it is useful to reemphasize that the drone is merely an actuator. Given the modular design of the project, it can easily be swapped out for another piece of technology. From this fact alone, it's clear that this project's impact is widespread. It can be extended as a system to achieve tasks as common as turning on house lights or as complex as piloting a vehicle.

# 7 Acknowledgements

We would like to personally thank our advisors at The Cooper Union, Professor Toby Cumberbatch and Professor Dirk Luchtenburg. The progression of both this project and of our professional careers must be attributed to their reliable and faithful guidance.

# References


[1] L. Trejo, R. Rosipal, and B. Matthews, "Brain–computer interfaces for 1-d and 2-d cursor control: Designs using volitional control of the eeg spectrum or steady-state visual evoked potentials," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 225–229, 2006.

[2] B. Kim, M. Kim, and S. Jo, "Quadcopter flight control using a low-cost hybrid interface with eeg-based classification and eye tracking," *Computers in Biology and Medicine*, vol. 51, pp. 82–92, 2014.

[3] A. Sweigart, *Automate the Boring Stuff with Python*. No Starch Press, 2015.

[4] J. P. de Graaff, "Ps-drone," Available at http://www.playsheep.de/drone/index.html (2014).

[5] G. C. Ribas, "The cerebral sulci and gyri," *Neurosurgical Focus*, vol. 28, no. 2, 2010.

[6] "Brain lobes," Available at https://www.mayoclinic.org/brain-lobes/img-20008887.

[7] A. A. S. I. Arman and A. Syed, "Cost-effective eeg signal acquisition and recording system," *International Journal of Bioscience, Biochemistry and Bioinformatics*, vol. 2, no. 5, pp. 301–304, 2012.

[8] A. B. C. O. D. C. E. P. M. Osullivan, J. P. Pena and A. Temko, "Comparison of electrode technologies for dry and portable eeg acquisition," *2017 7th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2017.

[9] C. V. H. R. Y. J. Xu, S. Mitra and K. A. A. Makinwa, "Active electrodes for wearable eeg acquisition: Review and electronics design methodology," *IEEE Reviews in Biomedical Engineering*, 2017.

[10] "International 10-20 system," Available at https://en.wikipedia.org/wiki/10%E2%80%9320_system_%28EEG%29.

[11] N. U. of Sciences and P. Technology, "Eeg / erp data available for free public download," Available at https://sites.google.com/site/projectbci/ (2017/09/17).

[12] M. Osullivan, J. P. Pena, A. Bocchino, C. Omahony, D. Costello, E. Popovici, and A. Temko, "Comparison of electrode technologies for dry and portable eeg acquisition," *2017 7th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2017.

[13] "Ultracortex-makr-iv," Available at https://shop.openbci.com/collections/frontpage/products/ultracortex-mark-iv.

[14] J. Qiu, "Senior project proposal, brain-controlled drone," 2017.

# Appendices

## A Background on EEG

### A.1 Functional Areas of Brain

Based on the functionality difference of the different part of the brain, the human brain can be divided into 6 sections, and each section is called a lobe. The 6 lobes are frontal lobe, parietal lobe, occipital lobe, temporal lobe, limbic lobe, and insular cortex [5]. In terms of volume and functionality, the former four lobes are much more dominating than the latter two. Due to the significant size difference, the latter two lobes are very unlikely to be measured the EEG sensors. Thus, only frontal lobe, parietal lobe, occipital lobe, and temporal lobe will be further discussed in this paper.

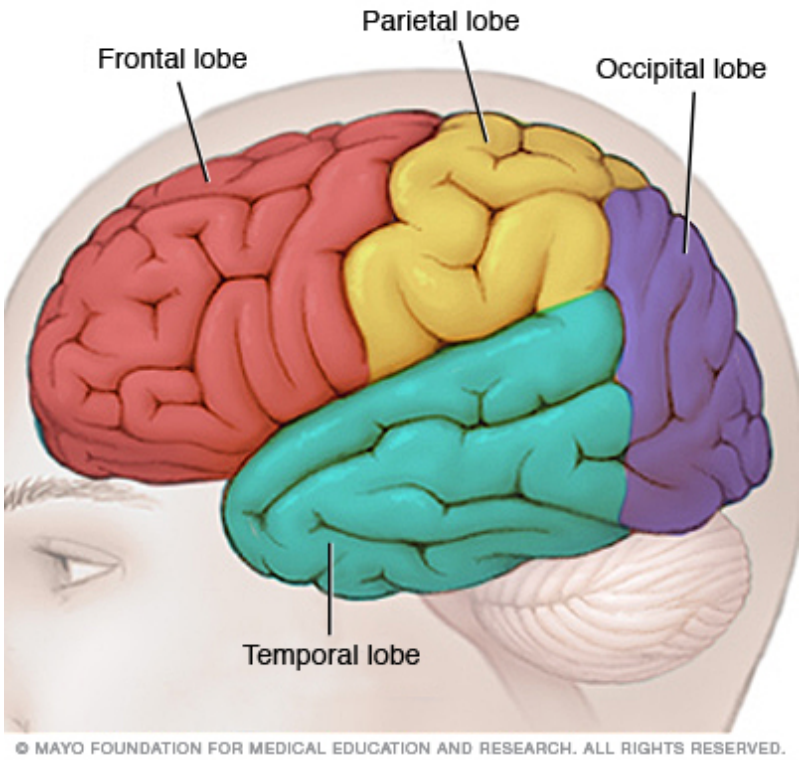


Figure 22: Four major lobes of the human brain [6].

The frontal lobe is located in the front part of the brain. It is in charge of

most of the execution of movements and high-level decision making. The frontal lobe has four important gyri, which are precentral gyrus, superior frontal gyrus, middle frontal gyrus, and inferior frontal gyrus. The precentral gyrus contains motor neurons that extend to the spine so it is mainly in charge of sending commands to different motor units in the human body. The later three gyri are called the prefrontal cortex, and they take part in more complicated motor control and decision making.

The parietal lobe sits immediately behind the frontal lobe and above the temporal lobe. It serves as a sensor integration center. The human body is equipped with a huge amount of sensors in order to observe the surrounding world. For example, human skin is mainly used to detect the temperature and any potentially dangerous stimulation. The parietal lobe is the destination of most of those sensors, and it analyzes the sensory data and helps to make decisions. Part of the parietal lobe also plays a very important role in language processing

The occipital lobe is located at the back of the brain. It is mainly in charge of visual processing.

The temporal lobe is located at the bottom of the brain. Since the temporal lobe is the closest to the ears, one of its key functions is processing audio data, such as interpreting the language. Another main function of temporal lobe is associated with human's memory.

In the application of a drone control system, directional commands are the most important information. Since it is very likely that most of useful information comes from the frontal lobe, it is more efficient to localize electrode distribution in only frontal lobe area.

## A.2 Generation of EEG

The origin of EEG signals is neurons in the human brain. Each time a neuron is transmitting a command to the adjacent neuron or motor units, it will generate a potential difference encoding the information and propagate the command to the next stage. However, measured EEG cannot precisely represent the information of a single signal path. The first main reason is the amount and size of neurons.

The number of neurons in the human brain is in the order of $10^{11}$, and the size of a neuron is significantly smaller than an electrode. Thus, if an electrode is used to measure EEG signals, what it actually captures is the superposition of electric potential generated by the thousands of neurons nearby. The second reason is the physical objects between the sensors and the neurons. Conventionally, scientists use surface electrodes to measure EEG signals, meaning between the sensor and the neurons that are being measure, there are usually three degrees of insulation: the skull, skin, and hair. As a result, the EEG signals captured by the sensors are usually the significantly attenuated sum of synchronous activity of thousands of neurons. Because it is nearly impossible to localize neuron activities, it has been a challenge to extract useful and clear information from EEG signals.

## A.3 Frequency Decomposition of EEG

The frequency components of EEG signals is ranged from DC to around 40Hz. Based on the frequency corresponding human behavior, the EEG spectrum in divided into 5 bands.

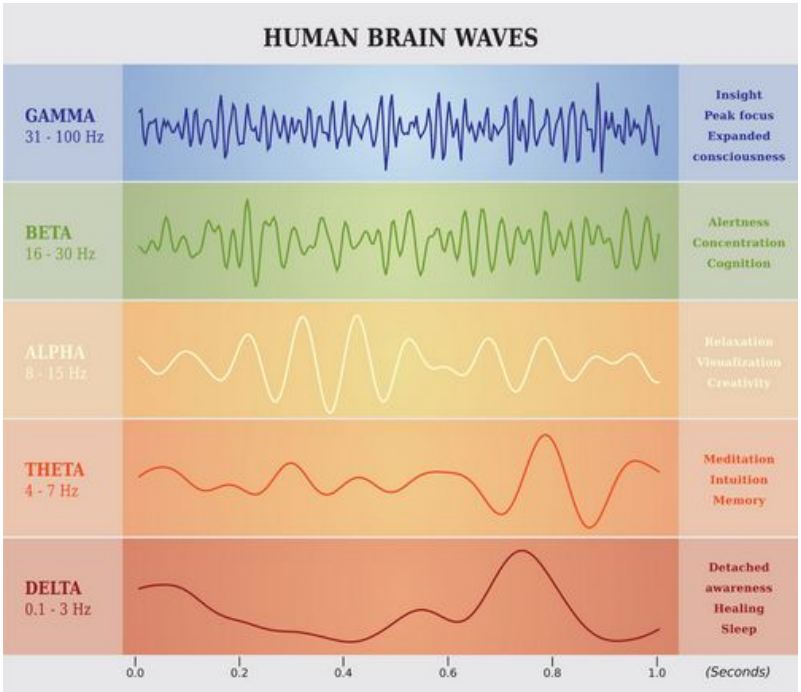


Figure 23: System architecture for prototype I

Delta wave is ranged from DC to 4Hz. It has the highest amplitude compared to the signals in other bands. It is the dominant rhythm in infants. For a normal adult, delta wave only appears in deep sleeping cycles.

Theta wave expands from 4 to 8Hz. It is the dominant rhythm of children up to 13. For an adult, theta wave only appears in sleeping or special activities, such as meditating.

Alpha wave is ranged from 8 to 12Hz. It is the major rhythm seen in an adult in relaxed state. It disappears when eyes open or when light is shined onto closed eyes.

Beta wave sits between 12 and 30Hz. It appears when the human is in alert state or concentrating on something. It also appears when there is significant mental activities. However, beta wave has a relatively low amplitude, so it is very difficult to accurately capture it.

Gamma wave is above 30Hz. It only appears when the brain is functioning at peak performance.

## A.4 Electrode

A good electrode design can also improve the quality of the EEG signals captured. This section compares electrodes built with different materials and working mechanisms.

The first commonly-used electrode technology is Ambu Neuroline 700 electrode [7] [8]. This is a disposable silver-chloride based electrode. It can capture EEG signals outside the hair zone. However, if it is used over hair, gel needs to be applied in order to acquire the accurate EEG signals.

The second option is flex electrode by Cognionics Inc. This electrodes has multiple flexible legs [8]. When pushing the sensor toward the skull, the legs brush hair aside, allowing the electrode in the middle to have direct contact with the skin. By removing the effects from hair, this type of electrode is able to measure EEG signals more accurately.

Another electrode technology is microneedle array electrode [7] [8]. This sensor has an array of needles at 500 $\mu$m to penetrate the dead cells on the surface of skin.

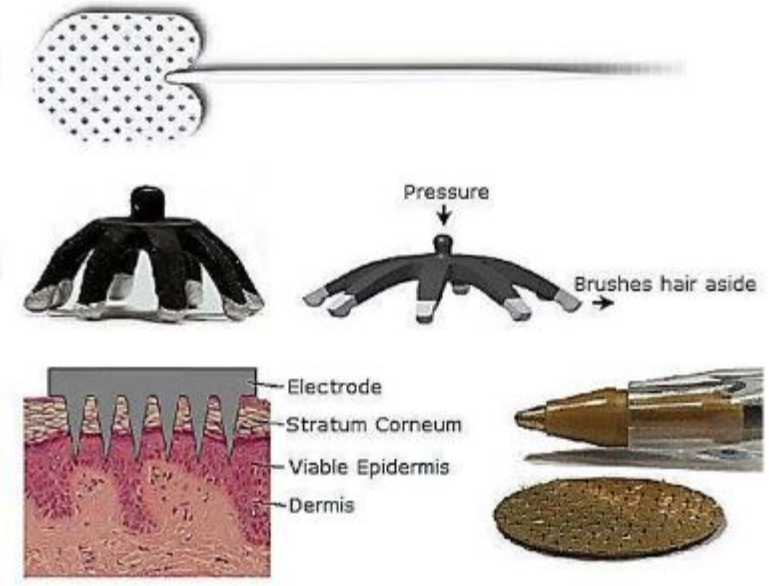


Figure 24: Various EEG electrodes

The advantage of this technology is a relatively high signal-to-noise ration (SNR).

In order to protect the signals from noise coupled in along the signal path, sometimes an active electrode is used [8] [9]. A common active electrode circuit is a unity gain buffer, which acquires the signal from a high-output-impedance system (human brain), and transmit the signal to the next stage with very low output impedance. Other active electrodes may have gain stage close to the sensors.

## A.5 Electrode Placement

In many past EEG acquisition researches, headsets were designed to allow the user to wear and remove the sensors with more ease. More importantly, they help to stabilize each sensor during measurement. The stability of the sensors during measurement is essential because it reduces the motion artifacts in the measured signals. Furthermore, the headset provides structure and al-

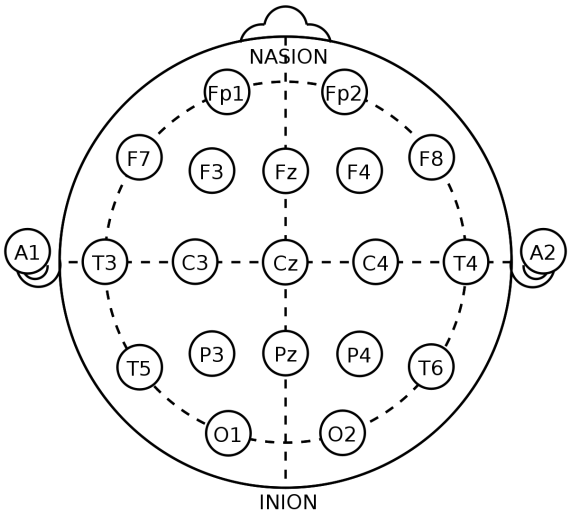


Figure 25: International 10-20 system

lows for a pressure to be applied downwards onto each of the flex sensors, allowing them to brush the hair aside.

The original plan was to create a customized headset to hold EEG sensors. However, the placement (location) of EEG sensors is standardized as shown in Figure 25, which is also known as the international 10-20 system [10]. It is not only time consuming to recreate a headset model, but it is also very difficult to make a model that matches the standard. Therefore, we decide to look for some existing headsets.

# B Simulation

We ran various of simulations to confirm the feasibility of our proposed data processing and classification methods on EEG signals. Specifically, the features were extracted from the signals using wavelet decomposition and classified into outputs using neural networks.

## B.1 Data Generation

In order to generate the data, we assumed the signal was periodic over a small time interval. Seven classes of bio-signals were generated by superimposing three sinusoidal waves with frequency, amplitude, and phase values chosen from pre-defined distributions. We assumed the frequency and the amplitude had Gaussian distributions, while the phase had a uniform distribution. The complete list of parameters are detailed in Table 3. On top of these simulated bio-signals, a zero mean Gaussian noise was added to ensure a SNR of 20 dB. This helps to make the signals reflect the expected noise from actual EEG measurements. Finally as a result of these parameters, the sampling frequency was 160 Hz.

| Class | Frequency (Hz) | Frequency variance | Amplitude (V) | Amplitude variance |
|---|---|---|---|---|
| 1 | 10, 15, 20 | 0.5 | 0.8, 0.1, 0.8 | 0.01 |
| 2 | 11, 13, 25 | 0.5 | 0.5, 1, 0.4 | 0.01 |
| 3 | 13, 18, 19 | 0.5 | 1, 0.3, 0.5 | 0.01 |
| 4 | 8, 14, 21 | 0.5 | 0.7, 0.8, 0.9 | 0.01 |
| 5 | 10, 13, 17 | 0.5 | 0.5, 0.9, 1.2 | 0.01 |
| 6 | 12, 20, 22 | 0.5 | 1, 0.7, 0.9 | 0.01 |
| 7 | 14, 16, 18 | 0.5 | 0.5, 0.5, 1 | 0.01 |

Table 3: Parameters for EEG signal generator

For each class, 200 samples were generated randomly based on each respective set of parameters.

## B.2 Feature Extraction

Once the data were obtained, the next step was to compress them. We mapped the time sequence data into a feature space while preserving the original data through an orthogonal transformation, namely a discrete-time wavelet transform. Specifically, we used Daubechies 4 wavelet function, provided by the PyWavelets module in Python, to decompose the signals into 4 sub-levels. Each sub-level corresponds to a different frequency band, as well as a different frequency resolution. Detailed information for each sub-level is listed in Table 4.

| Decomposed signal | Frequency Range(Hz) |
|---|---|
| D1 | 40-80 |
| D2 | 20-40 |
| D3 | 10-20 |
| D4 | 5-10 |
| A4 | 0-5 |

Table 4: Frequency ranges of the different sub-levels produced by the wavelet decomposition at a sampling rate of 160 Hz.

Next, the detailed coefficients from sub-level 2, 3, and 4 were used to represent the signal. To further reduce the dimensionality of those coefficients, the absolute mean average value was calculated for each set of coefficients. This provides a rough estimate of the average power of each frequency band.

Thus for two channels of data, a feature vector of length 6 is expected. In the simulation, a total of 1400 samples are processed, meaning a $1400 \times 6$ feature matrix is generated.

## B.3 Neural Networks

Neural networks were chosen as our classifier for two reasons: first, it is scalable, having the ability to add more neurons and hidden layers without changing the underlying structure; and second, it can realize a non-linear transformation function.

The neural networks in this simulation consists of 6 input nodes, 7 output nodes, and 21 hidden layers. The number of hidden layers is decided by trial and error.

In particular, the neural networks from an open-source library Scikit_learn are used in this project. A ReLU activation function is used to introduce non-linearity for the networks. Moreover, this classifier uses the "lbfgs" solver, which estimates the Hessian (second order derivative) matrix when performing deepest descent. Thus it can avoid getting stuck at a local minimum.

The major problem during testing neural networks is that some of our data is actually very large, and the networks cannot handle big input data very well. To solve this problem, we pre-process the data by standard normalizing them.

## B.4 Simulation Results

In the simulation, the EEG signal is first generated as aforementioned, and then it is classified by the pre-trained neural networks.

To give a better visualization, a simple GUI is made as shown in Figure 26. In this GUI, the user can choose to generate one of seven types of EEG signals in order to emulate a user's potential intention (i.e. a neuron firing). Subsequently the signal is processed and a classification is predicted by the neural network. As one can see in the figure, this case shows a class-4 EEG signal being chosen, generated, and finally, displayed. More importantly, this shows the network is correctly predicting that the EEG signal is type-4!

In order to obtain the overall correction rate of the network, an automatic simulation is made, in which, the neural networks are trained multiple times. At each overall training iteration, the networks are evaluated with 700 random test samples to obtain a correction rate. The overall correction rate for this 7-classes classifier is around 70 %. Although fine tuning could be done to achieve an even better rate, it is unnecessary to spend more time on simulated data. The purpose of this simulation was to prove that the proposed feature extraction and classification methods work reasonably well.

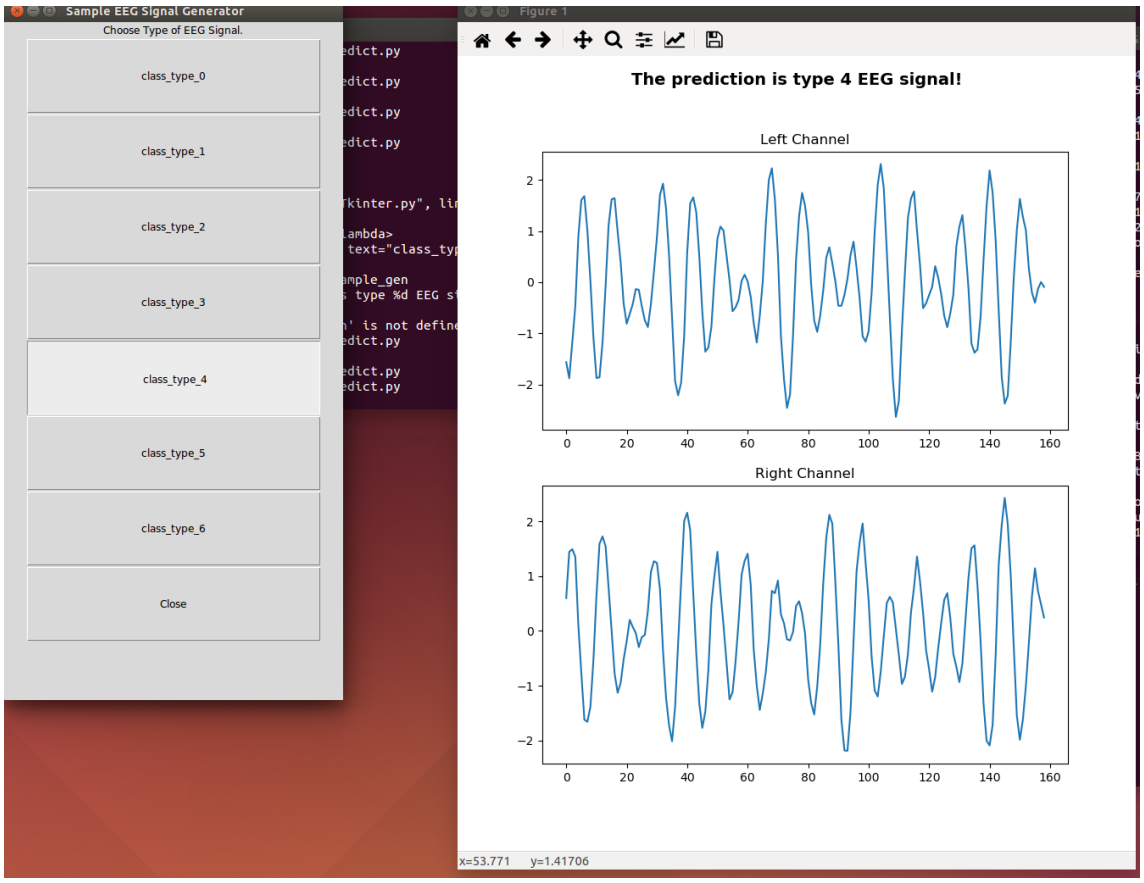


Figure 26: GUI for simulation

## B.5 Other Machine Learning Considerations

In order to gain a broader picture of the problem, it is important to consider both the possibility of classifying the raw data straight from the headset and the possibility of using other classification models. We decide to explore this idea simultaneously. First we choose to use a simple Gaussian generative model. Meanwhile, we found a binary EEG data set from a public database offered by the National University of Sciences and Technology, Pakistan [11]. They provided a binary (1-bit) data set that consists of EEG signal samples with 19 different electrode features. The data were obtained from a participant who either thought about moving his left hand or moving his right hand. While the database also provides two dimensional data, the main focus initially is being able to discern a single binary command (left/right).

Returning to the classification problem, we combine the EEG data corresponding

to the participant moving either their right or left foot. Interestingly, the Gaussian generative model was able to discern 99.98% of the observations correctly. While this is strong evidence that other models besides a neural network may be suitable, there are still numerous concerns that may cause this model to fail. Firstly, we have not yet been able to collect usable data. Consequently, it is not known if our observations will produce similar EEG readings. Next, the group in Pakistan used a commercial EEG headset (Neurofax EEG System), while we are designing our own headset [11]. Their EEG readings may be more specific to the exact brain areas/waves that need to be observed. As well, the noise they experience is likely less, making the quality of their data much better.

Moreover, as mentioned previously, there is a systematic difference in that the Pakistani group was able to observe 19 different EEG values per reading. For our headset, we currently only have 2 EEG sensors. The presence of more data allows for more features for the classification model and this may explain why the Gaussian generative model performed so well on the public data. On the other hand, with our data, the model will have to classify the signals based on only two main features. Despite this problem, the Pakistani data happened to be ordered and labeled with the region of the brain from which the EEG reading came. With this in mind, it may be useful to re-classify the data using the same model, but using on a 2 feature subset of the original 19 features. By varying the different combinations, we can compare which pair of brain areas led to the best classification rates. Subsequently, we can design the headset to have the two sensors focus on the two respective brain regions.

Lastly, a final difference occurs in the fact that this model only had to execute a binary classification of the data into left and right. In our final design, we aim to be able to discern all six translational directions (seven when including an idle command). The presence of multiple classes can cause overlap between classes in the model and decrease its accuracy. Nonetheless, these tests have shown that other machine learning methods are worth significant consideration. Furthermore, the experiments confirm the feasibility of using sampled time-domain EEG data, without any complicated signal processing, as the input for the classification model.

Once the EEG acquisition circuit is complete, it will prove useful to compare these different approaches: neural networks versus other machine learning models and raw EEG versus signal-processed EEG data.

# C Old Prototype

In our first prototype, we have two separate boards. The first board is used for active electrodes, so we could amplify the acquired signal locally. The second board is designed for further amplification and additional signal filtering. Those two boards would have BNC connectors so that we could transmit signal through shielded coaxial cable, which could prevent electromagnetic interference from the surrounding environment.

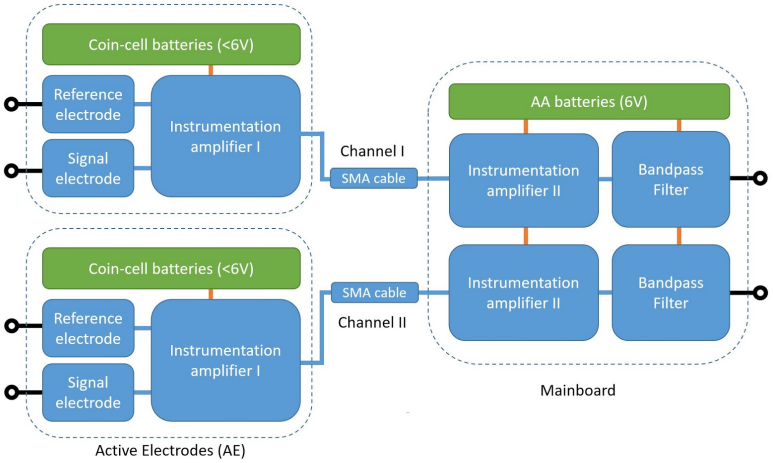


Figure 27: System architecture for prototype I

## C.1 Electrode

For our application, where the signal level is very small while the noise could be dominating, the electrode needs to meet the following requirements [12]:

1. The electrode has to be non-invasive because it is very dangerous to insert metal into the human brain, and it is also very difficult to drill through the skull.

2. The electrode needs to be non-disposable so that it can be reused for low cost.

3. The output impedance needs to be at most $10^7$ $\Omega$ in order to make sure the amplifier can measure the EEG signal correctly.

4. The electrode should be able to work without gel with unprepared skin for better user experience.


Figure 28: Cognionics Flex sensor

Based on the requirements above, we chose the flex sensor from Cognionics as our electrode, which is shown in Figure 28. The selected sensor has an output impedance up to 2000 $k\Omega$, which is less than our targeted maximum value. When force is applied to the top of the sensors, the flexible legs of the sensor spread out and brush the hair aside in order to provide direct contact between the metal (silver/silver chloride) tip of each flex leg and the scalp under the hair. This results in compatibility of the designed headset with unprepared skin, meaning that subjects do not need to shave their heads.

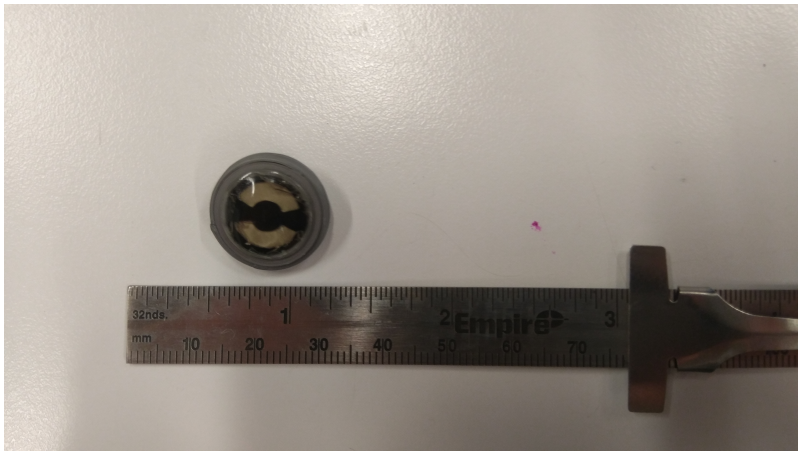


Figure 29: Cognionics Drypad sensor

In addition to the flex sensors for all the through-hair channels, we also selected the drypad sensor from Cognionics for hairless skin regions. This sensor will be placed on the user's forehead in order to obtain an reference signal for the other channels. Fortunately, this sensor has an output impedance of only 100 kΩ and is able to work without gel.

## C.2 Ultracortex "Mark IV" EEG Headset

After some research, we found the OpenBCI EEG headset [13] to be a great candidate for this project's headset. It's designed based on the 10-20 system, and it is open source.

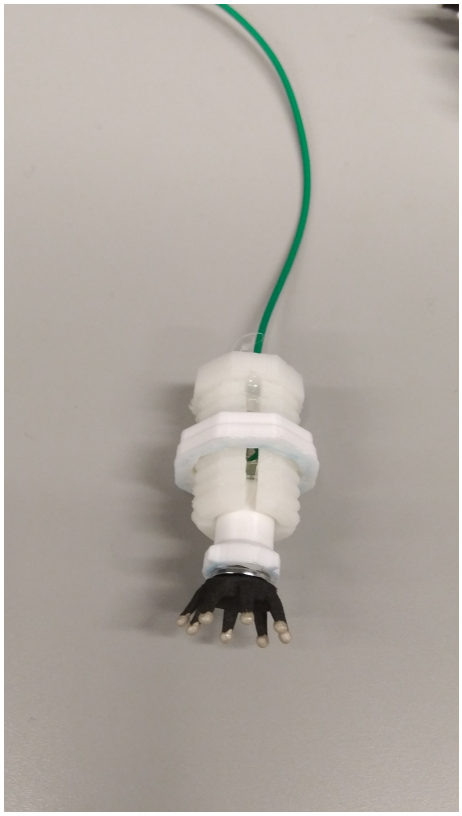

Figure 30: Sensor holder

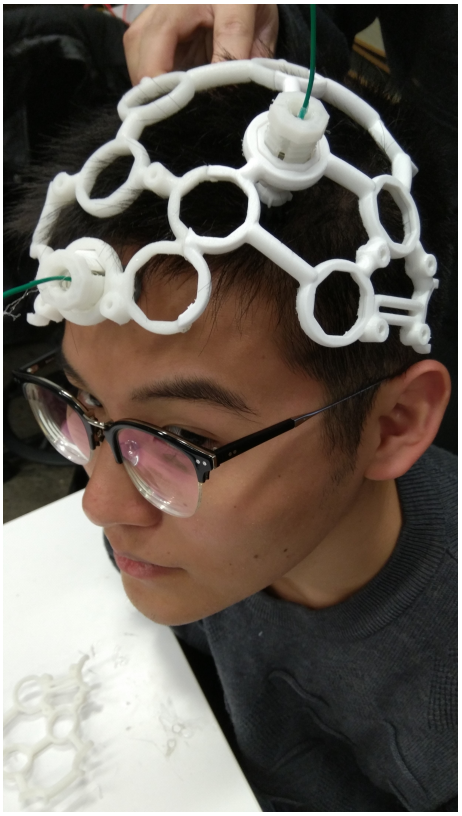

Figure 31: Ultracortex's frame

Despite the convenience of open source material, there were still a few adjustments that needed to be done. First, we were told by the operator at Cooper Makerspace that the frame is too big to be printed. To solve this problem, we cut the frame into six smaller pieces that could be connected together after print-

ing. Secondly, the sensor holders this frame uses are designed for needle electrodes. Thus, the shape of the holder is modified so that it can hold both our flex sensor and drypad sensor.

As shown in Figure 30, each sensor is attached to a spring-actuated system. The spring and the screw-like cylinder attached to the end of the sensors allow the user to adjust both the radial position of the sensors and the pressure force on the flex sensors according to his/her head size.

The headset has multiple octagon-shaped holes as shown in Figure 31. The sensor can be placed at any of the holes. This design allows us to test the effect of sensor location on the measured EEG signal, and thus to optimize the quality of EEG signal by placing the sensor to the right location.

## C.3 On-headset PCB Design

The main function of the on-headset PCB is to provide a gain of 40dB to make the EEG signal more resilient to the noise along the signal path.

As for the instrumentation amplifier, there are some specifications to meet:

| | |
|---|---|
| Gain | $>$ 40 dB |
| $V_{supply}$ | $<$ 36 V or $\pm$ 18 V |
| Size | $<$ 100 mm x 100 mm |
| Weight | $<$ 100 g |
| $Z_{in}$ per channel | $>$ 10 $M\Omega$ |
| CMRR | $>$ 100 dB |

Table 5: Design specifications for the on-headset circuit

There are two implementation plans for the instrumentation amplifier circuit: one is the single-IC solution and the other is the discrete component solution. With both solutions, a gain of 40 dB can be easily achieved. The supply voltage of the discrete component circuit is the supply voltage of each op amp, which can be as low as 1.5V. For the single-IC operational amplifier, the INA118 from Texas Instruments has the lowest minimum supply voltage of 3.0 V. In terms of the supply voltage, both

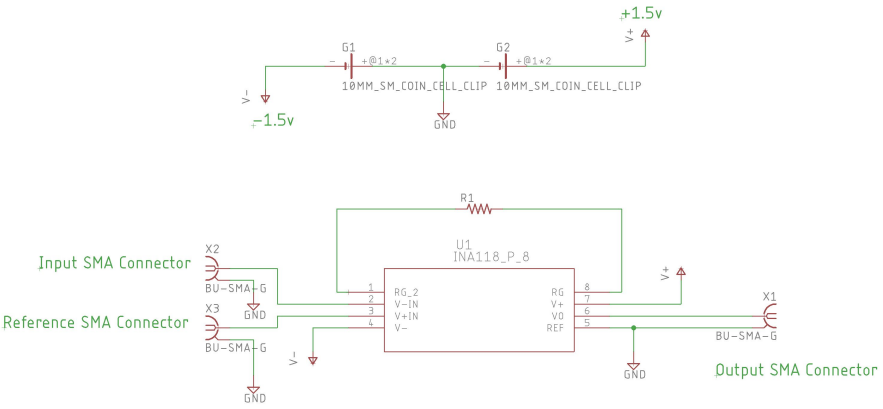


Figure 32: Schematic capture of the on-headset circuit

solutions are in the safety range. Moreover, both solutions have input impedances larger than $10^{10}$ Ω, which is significantly higher than what is necessary.

However, the single-IC solution is better in terms of component size and component value matching. A discrete component instrumentation amplifier normally requires 3 op amps and 7 resistors, while for the single-IC solution, only 1 op amp and 1 resistor are required. Thus, in order to minimize the size of this PCB, the single-IC solution is preferred and selected. Moreover, integration of components into one wafer can reduce the component value spread. Therefore, it should provide more stable gain.

In order to minimize the supply voltage, the INA118 from Texas Instruments was selected since this model has the lowest minimum supply voltage of 3.0 V among all the instrumentation amplifier ICs that can provide a gain of 60dB and noise voltage less than $12\text{nV}/\sqrt{Hz}$. In addition, since the instrumentation amplifier requires 3 V supply voltage, we decide to use two 10-mm coin cell batteries as the positive and negative power supplies.

Based on the design specification discussed above, the schematic and design layout of the on-headset PCB is completed in EAGLE software. The size of the PCB is 36mm by 36mm. The schematic is shown in Figure 32.

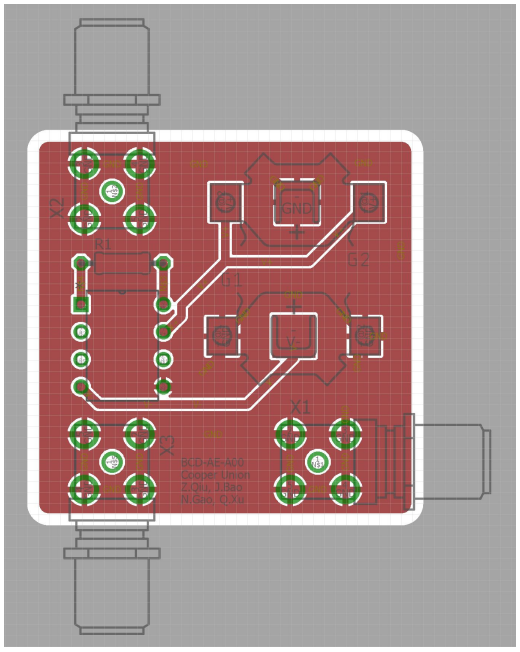

Figure 33: Top layer of the on-headset PCB

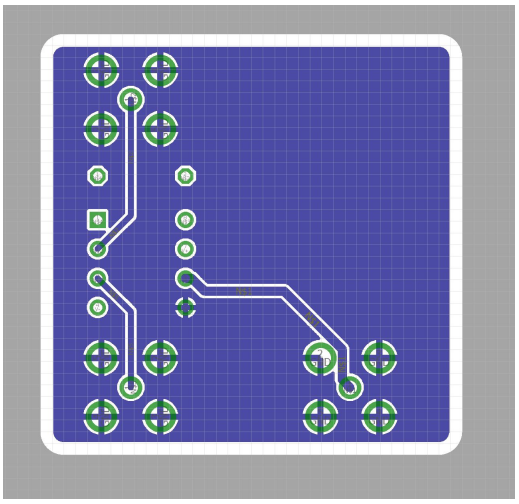

Figure 34: Bottom layer of the on-headset PCB

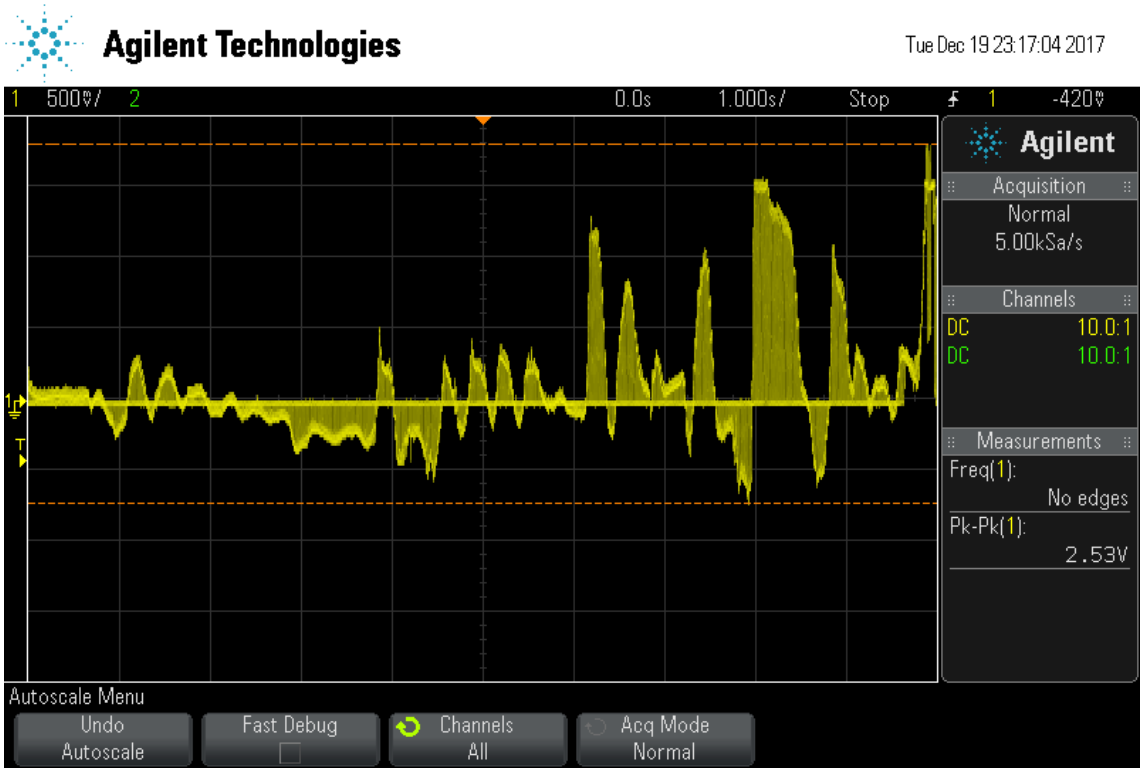


Figure 35: Measurement of EMG signal

## C.4 Tests on INA118 Instrumentation Amplifier

To test the INA118 chip, we measured the EMG signal using this amplifier with gain of 40 dB. As the user moves his/her fingers the EMG signal also changes correspondingly on top of 60 Hz line noise as shown in Figure 35. The 60 Hz line noise is expected, because we haven't include any filter to suppress the noise during the experiment. To suppress the 60 Hz interference, we implement driven right leg (DRL) circuit to cancel out the effect of common mode signal. The result obtained after implementing DRL is shown in Figure 36. This suggests that DRL significantly improves the common mode rejection ratio of the amplifier circuit. Moreover, input guarding technique is also considered for later implementation with shielded cable to further reduce common mode effects.

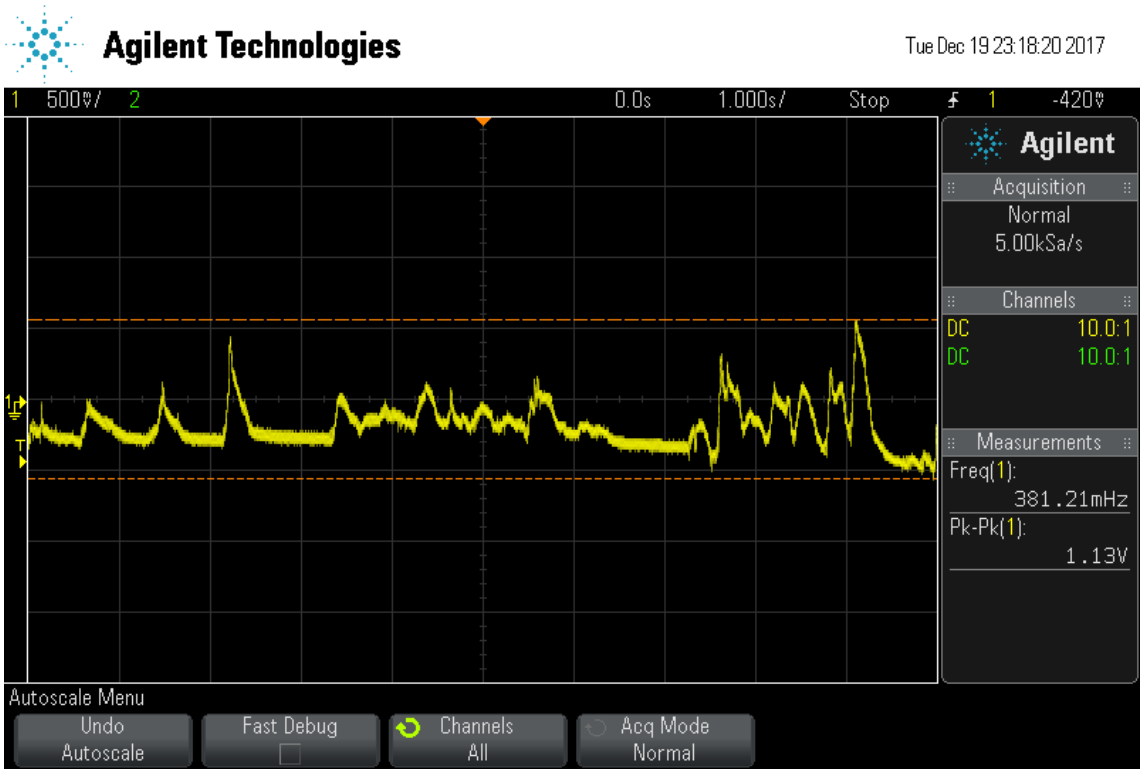


Figure 36: Measurement of EMG signal with driven right leg

## C.5 Main Board PCB Design

The main-board has two main function blocks. The first block is a secondary gain stage. This circuit takes in the pre-amplified EEG signal from the on-headset

PCB through the SMA cable, and amplifies it again to reach an amplitude that can be accurately sampled by the ADC. As discussed in the previous chapters, the first gain stage provide a gain of 40dB, so in order to achieve an overall gain of 100dB, the gain stage on the main-board has to provide a gain of 60dB. In order to achieve such a high gain with minimal components, an instrumentation amplifier is used in this stage. The second block is a band-pass filter to eliminate the noise that is not in the desired frequency ranges.

In order to minimize fabrication process variance and keep the bill of materials as short as possible, the INA118 from Texas Instrument is used again on this PCB. INA118 has a maximum gain of 10,000 (80dB) which meets our gain requirements, and it also has a wide supply range (up to 36V) to guarantee the desired output swing.

As discussed in the previous paper [14], the target EEG band in our application is the beta wave, which occurs when a person is awake and actively thinking. The beta wave extends from 12 Hz to 30 Hz, and this range should be the pass-band of our band-pass filter. The noise that we want to eliminate mainly comes from two sources: the first source is head motion artifacts, which are usually under 5 Hz, and the second source is the 60-Hz line noise. In order to reject the motion artifacts and the line noise, our filter must have enough attenuation at each critical frequency.

Given such a narrow pass-band and short transition bands, we decided to use a 1-dB-ripple Chebyshev filter, because of its relatively fast roll-off and relatively flat pass band response. The 3-dB frequency of the filter is selected to be 10 Hz and 30 Hz in order to let beta wave pass through, while obtaining a 30-dB attenuation at 5 Hz and 60 Hz.

In order to realize the design specification, a third-order system must be used, and the parameters for each stage are tabulated below:

To implement the filter with analog circuitry, we decided to use the multiple feedback topology to minimize the number of passive components and moreover to minimize the noise gain. The component values are tabulated in Table 7.

In order to ensure the filter meets all the design specifications, a simulation was run in LTSpice software, and the simulation result is shown below.

| Stage | $F_o$(Hz) | Q | $A_o$ |
|---|---|---|---|
| 1 | 10.6 | 4.3122 | 4.5342 |
| 2 | 28.3 | 4.3122 | 4.5342 |
| 3 | 17.3 | 1.9190 | 1 |

Table 6: Parameters for each stage of band-pass filter

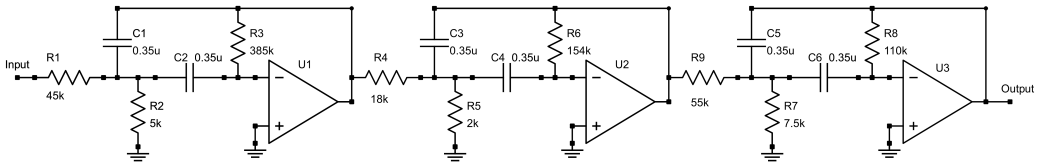


Figure 37: Schematic for Chebyshev multiple feedback band-pass filter

| Components | Values | Components | Values | Components | Values |
|---|---|---|---|---|---|
| R1 | 45 kΩ | R4 | 18 kΩ | R7 | 7.2 kΩ |
| R2 | 5 kΩ | R5 | 2 kΩ | R8 | 110 kΩ |
| R3 | 385 kΩ | R6 | 154 kΩ | R9 | 55 kΩ |
| C1 | 0.35 $\mu$F | C3 | 0.35 $\mu$F | C5 | 0.35 $\mu$F |
| C2 | 0.35 $\mu$F | C4 | 0.35 $\mu$F | C6 | 0.35 $\mu$F |

Table 7: Component values for multiple feedback band-pass filter

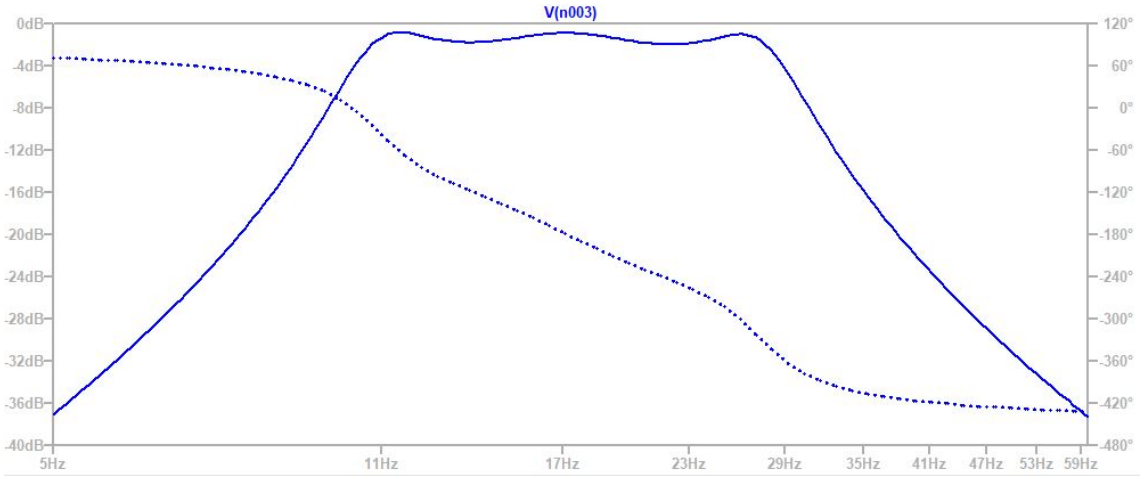


Figure 38: Frequency response of Chebyshev multiple feedback band-pass filter

As shown in Figure 38, the 3 dB attenuation occurs around 11 Hz and 30 Hz

as we expected, and at 5 Hz and 60 Hz the attenuation is greater than 36 dB. The simulation proves that our design in theory should be able to preserve beta waves and reject the expected noise.

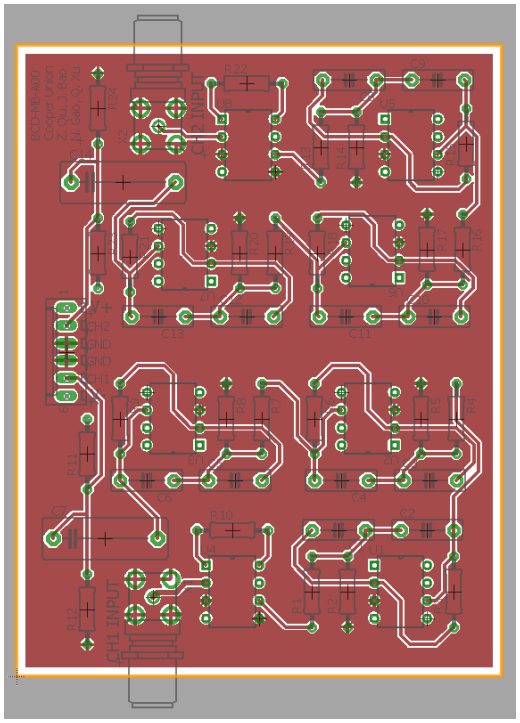

Figure 39: Top layer of the mainboard PCB

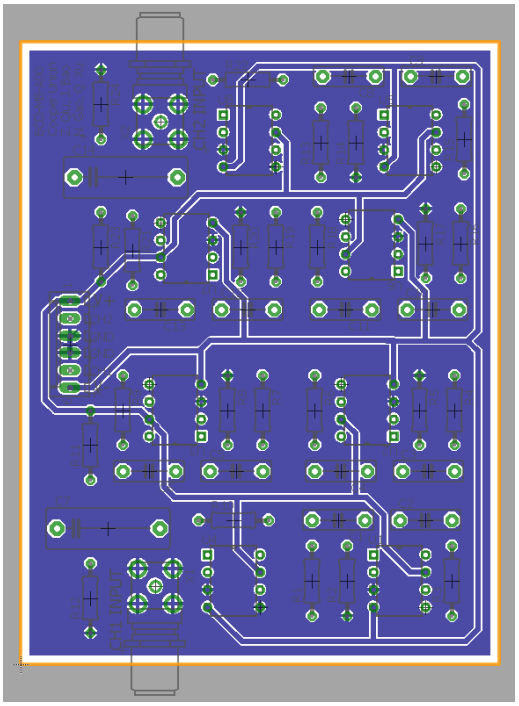

Figure 40: Bottom layer of the mainboard PCB

## C.6 PCB Fabrication

After consulting with some PCB experts, we selected JLCPCB, who makes prototype PCBs for only $2, as our PCB fabrication supplier. Figures 41 and 42 show the PCB boards we received. Each component is hand soldered.

## C.7 Discussion

After we put all the parts together and try to take some EEG measurement, few problems were encountered. The most obvious one was perhaps the 60 Hz line noise. The bandpass filter we designed has a stop band attenuation of 30 dB, so in an ideal case, the noise should be suppressed.

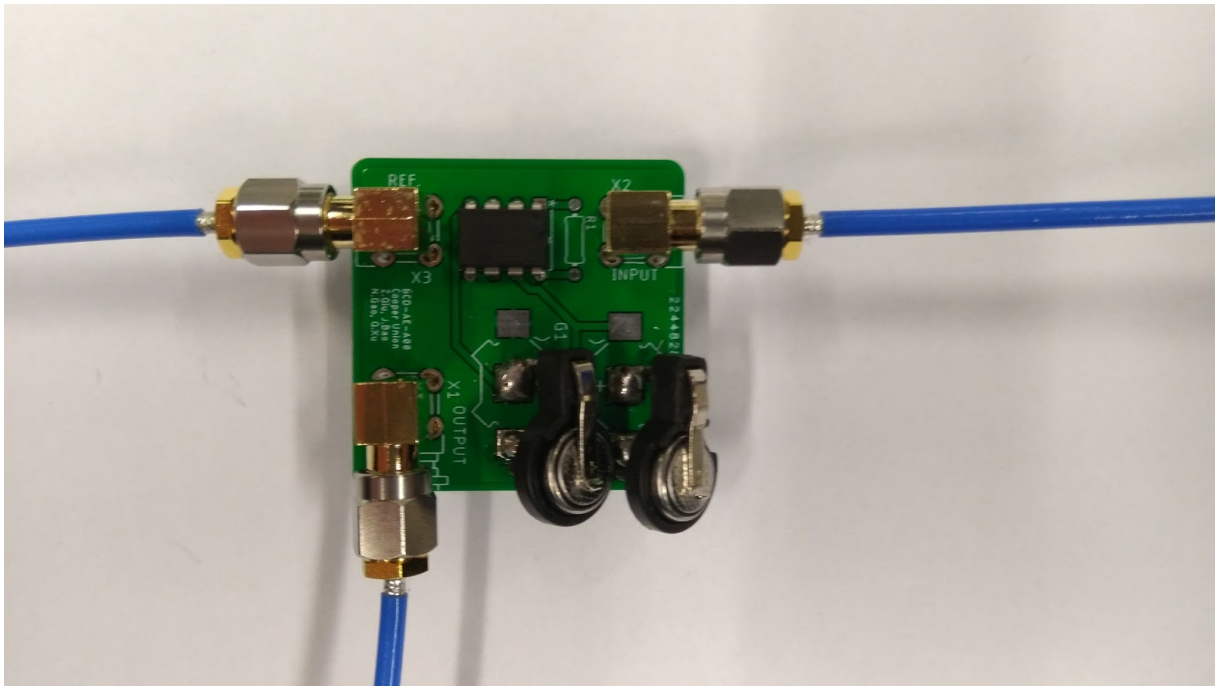


Figure 41: On-head PCB board

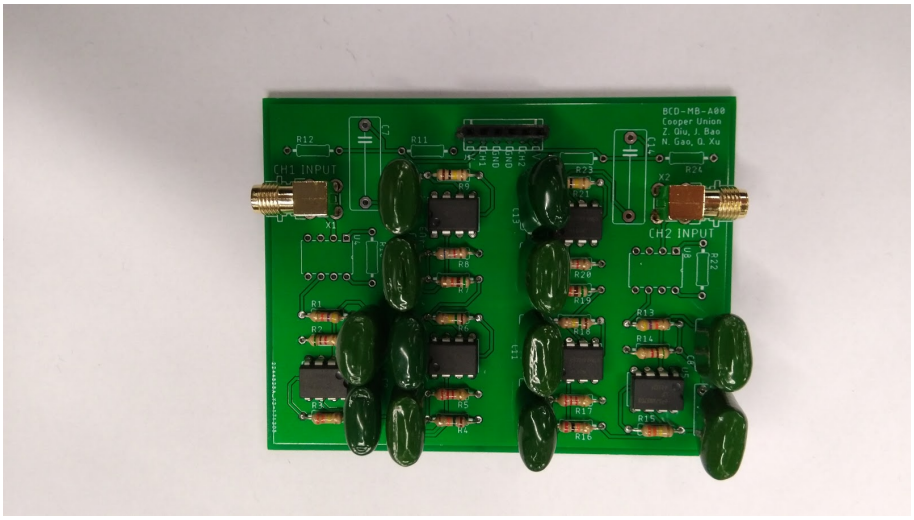


Figure 42: Main PCB board

The biggest issue we had was undesired filter response due to the component variation. In order to achieve high roll-off rate in the stopband, we chose a Chebyshev filter. However, this filter type was known to be very sensitive because each of the ripple in the passband was caused by a high-Q bandpass stage. The Q and the gain of each stage needed to be precisely controlled in order to construct the equiripple shape. If there was a abnormal Q or gain in any of the stage, the overall frequency

response the filter would be totally destroyed. During our PCB assembly process, we used capacitors with 10%-variation. This huge variation caused the Q factor and the gain of each stage shifted from the designed values, and eventually resulted in wrong cut-off frequencies and magnitude of the passband ripple of the filter. As a result, we were able to clearly see the 60-Hz line noise at the output of the filter.

The second problem we faced was the instability of flex EEG sensors. The electrodes were placed in the head frame through screw-bolts mechanism. Pressure is needed to make a stable skin-electrode interface. However, the pressure needed to maintain stable signal reading would cause discomfort on user's scalp after wearing the headset for a short while. More importantly, applying different pressure to the electrode against scalp would cause a change the input characteristics of the electrodes. In other words, it contributed huge variation in input impedance based on how much pressure was applied. When the pressure applied to the electrodes against the scalp increase, we observe a significant increase in waveform as observed in the oscilloscope. Attempts to regulate the same amount of pressure on the headset was made but was not successful. For this reason, a calibration that relates certain brain activity to a extracted feature is needed every time when the user puts on the device, and every time when small adjustments are made to the wearing of the headset. This is not ideal because the unrepeatability of testing and calibration would jeopardize the reliability of the device.

Lastly, the head frame itself was little bit too big to fit onto one's head. In practice, there should be at least three of four electrodes spread around the head frame. Each electrode applied pressure force in different direction so that the net effect could stabilize the head frame. However, in our application, we only placed electrodes at the front side of the head frame; as a result, the frame was very loose, unless we used some other meanings to constrain the head frame on one's head such as a rubber band or a simple string.