NeuroController

Authors: Jonathan Ke, Wendy Mo, and Jean Udompanyawit

Department of Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—For individuals with difficulty controlling their muscle functions, performing tasks on a desktop computer is virtually impossible. Given that the traditional mouse and keyboard would be very difficult to use in this scenario, we aim to create an alternative platform that allows a more accessible control through brain signal acquisition techniques, specifically electroencephalogram (EEG) and electromyography (EMG). This platform increases computer accessibility through an integrated control interface system that is easily customizable for users.**

***Index Terms*—Arduino, Bluetooth, EEG, EMG, Emotiv, Fast Fourier Transform (FFT), Logistic Regression, Python, Random Forest, Support Vector Machine (SVM)**

I. Introduction

Nowadays, the computer is prevalent and critical for people in various ways, ranging from working to easing general daily lives tasks. Even though computers are being more accessible to the general body of people, one of the limited user groups are those who lack physical control abilities. For individuals who suffer from physical disabilities such as amputees, there is no platform in the market that offers an alternative controllable interface on the computer for this user group. Therefore, our project will aim to increase accessibility of desktop devices for individuals who do not have control of their upper limbs or have difficulty controlling muscle functions. We will create an application that serves as the mouse and keyboard interface and will be controlled through electroencephalography (EEG) and electromyography (EMG) signals.

Even though there is various research studying the control of cursor movement, there are currently no competing technologies in the market that offer an integrated interface allowing a full interaction on the computer desktop. Therefore, we are developing this platform to allow the user to both control a cursor as well as to use a keyboard on a device, through mental and augmented physical control. Our goal is to provide the individuals with a minimum shoulder movement ability and with full mental control abilities to be able to use this platform to navigate tasks on the computer wirelessly. This will provide full and unhindered access for people with disabilities to use a traditional desktop device such as a laptop on a user-friendly control.

II. Use-Case Requirements

Our main goal in this project is to provide an interface for those with disabilities to use desktop devices. The most important requirement is that the platform works with low latency, in real time, and with high accuracy for unhindered control on a desktop application.

One use-case requirement we have is to aim for above 90% accuracy in converting user intentions into the corresponding output within the interface. This will be tested through one of our testing methods, called the Task test.

To ensure a satisfying speed of our platform across the software and hardware implementations, another requirement is that a user, on average, can score 500 ms on reaction time in the human benchmark test using our device.

Further, we will require that our user hit five targets within 30 seconds with a 75% accuracy. This user accuracy and speed requirement will be tested through a point-and-click test.

Another use case requirement comes from our signals; for our application, two types of control data are required: continuous control data and single-time control data. Continuous control data will be obtained from the EMG, which measures the muscle movement signals from the shoulders. On the other hand, seven types of distinct single-time control data will be acquired from the EEG device. These distinct signals will be used to serve different features on our interface. For example, the EMG sensors will allow for control in either an up to down or left to right direction and double blinking will mimic clicking. These feature mappings will be set in this manner when the user puts on our device for the first time but will be customizable to allow for maximum flexibility.

Finally, the use-case requirement is that we will have an onscreen accessibility keyboard that provides the ability to send all standard keyboard inputs, modified with text autocomplete and customizable widgets based on the user preferences.

III. Architecture and/or Principle of Operation

We are building our application using Python as the programming language. From there, we are using various packages and libraries, including Tkinter and PyAutoGUI. For our backend signal processing, we are creating and training ML models using the Scikit machine learning library. Python provides a development environment that can be used and accessed by both the developer and user on any laptop.

After the user puts on the EEG headset, is attached to the EMG sensors, and opens our application, the user will be prompted to a screen that has an accessibility keyboard. For the EMG sensors, we will connect an Arduino with a Bluetooth module to transfer the data wirelessly. Both the headset and sensors serve as our data acquisition platforms. When the user performs various actions to accomplish tasks (eg. double blinking to click), the data will be processed and classified through ML pattern recognition, and the application will trigger the mapped action within the desktop.

The EMG shoulder movement will allow the user to control moving the cursor or going across the keyboard either in a left to right motion or in an up to down motion. The user will be able to switch between these two methods through one of our extracted features. The gestures used to control different functions will be made customizable based on our user preferences.

Further descriptions of our system architecture are described with our figures.

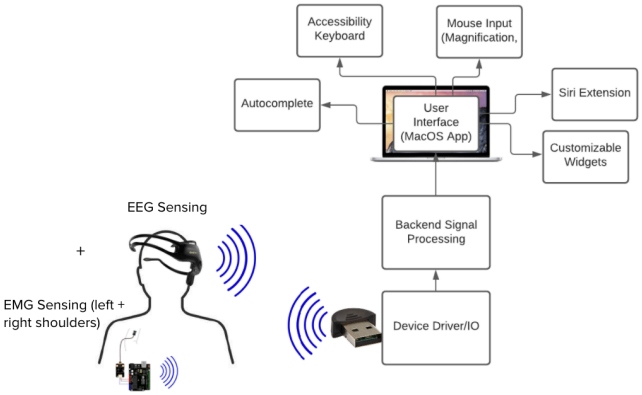


Fig. 1. Overall System Layout. Users will generate signals via EEG and EMG. The signal is sent wirelessly to backend processing that funnels recognized signals to the user interface.

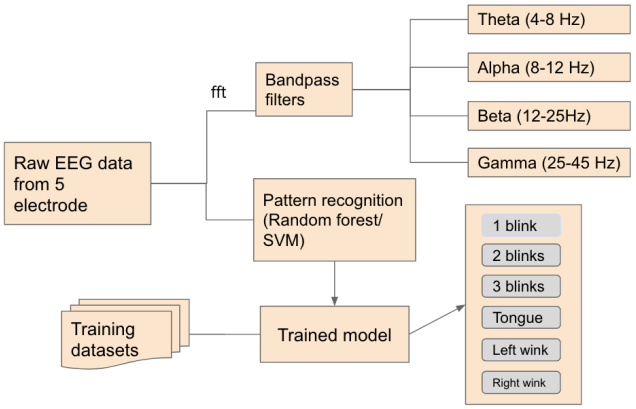


Fig. 2. Signal Processing Subsystem. Live raw EEG data is fed into a fast fourier transform (FFT) to ensure user actions are only forwarded to the frontend when the user is focused and an ML model to detect user commands

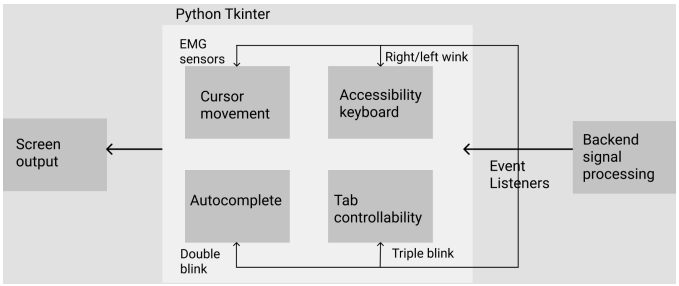


Fig. 3. Frontend Block Diagram. The processed signals from the backend will be connected to event listeners that the frontend can respond to. Users will activate certain features of our application, which will be displayed on our screen output.

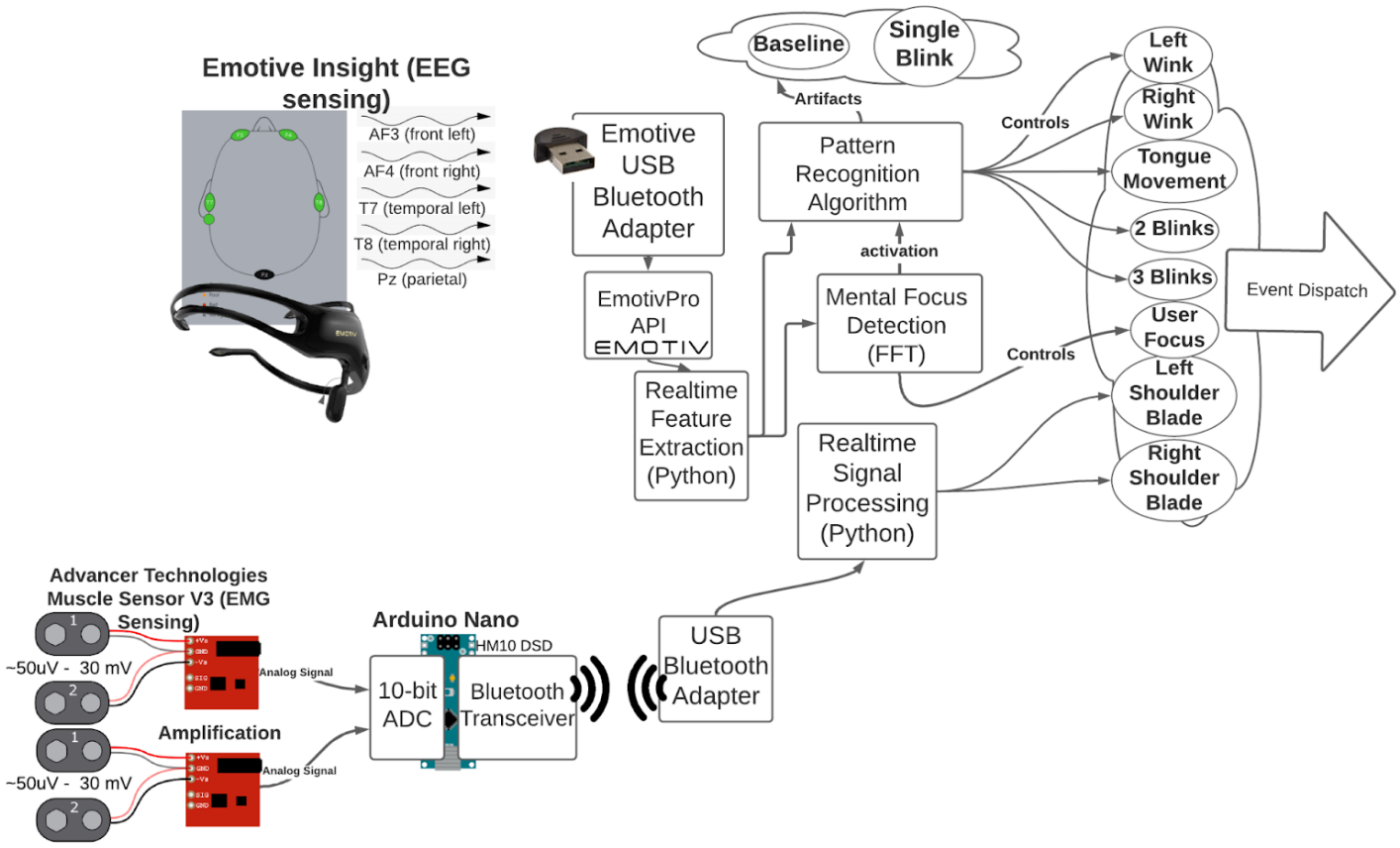


Fig. 4. Backend System Layout. Both EEG and EMG signals from the two sensing systems are funneled through separate wireless transmitters and unified under the same backend layer.

IV. Design Requirements

Speed and accuracy are our two most important requirements. As mentioned in the use-case requirements section, we aim to have over 500 ms latency from acquiring EEG and EMG signals to the end process of the computer input. Our data will be transmitted via Bluetooth, which generally has a latency of approximately 100 to 300 ms. We require that our Python backend processing does not have a latency over 50 ms. Python backend processing refers to the signal processing, machine learning data classification, and communication between the data acquisition device to backend program as well as to the user interface.

Our accuracy will be speculated based on two aspects: signal classification accuracy and overall user control accuracy. The first design requirement is to have our signal classification accuracy of more than 90% using a combination of support vector machine (SVM), random forest classification, and linear classification models. We require that the collected training data be contained from over 200-300 samples data sets per each signal data type.

The data will be sampled at 128 Hz. To save the expensive calculation, the Fast Fourier Transform (FFT) analysis will be calculated every 4.5 seconds to detect the focus state and to either activate or deactivate our ML classification functions. Since our window of interest will be running through the samples of approximately 1.5 second span as our longest data, the triple blinking has the features that last up to around 1.5 seconds.

EEG data will be analyzed in real-time to detect the most fitting classification for double blinking, triple blinking, left wink, right wink, clenching/tongue movement, normal baseline, and unclassified noisy signals.

Real-time EMG data from the two shoulder blades will be our main controlling method for the two directional control planes: left/right and up/down. Therefore, we aim to have an accuracy of 95% for detecting the shoulder movement that will be used as the continuous controlling data.

The design requirement for our interface is to offer our user both the cursor and the mouse control on the desktop by EMG, with additional controls through EEG. The user would be able to navigate through the entire desktop page easily with the cursor triggered by EMG signals from the shoulders or send key inputs like tabbing through a webpage using a combination of EEG signals.

V. Design Trade Studies

A. *Emotiv Insight Model 1.0*

The Emotiv Insight is a 5 channel, wireless headset that records the wearer's brainwaves and translates them into meaningful data. The Emotiv Insight is preferable to other EEG headsets (both produced by Emotiv and other companies) for its lower price point and its ability to provide the data necessary for our modeling. While other headsets contain more channels and provide more information, after looking at our project budget, the other EEG headsets would have cost more than the

money allotted. Lastly, there was already an Emotiv Insight available in the capstone inventory, so we did not need to spend any money.

This device also provides an out-of-the-box platform for interfacing through the Emotiv app for our application development.

B. Hardware Design

A major and early decision made in the designing of our application is to use the commercially available Emotiv headset. With this, we are somewhat limited in how we can pursue low latency and easy operation within our system. We initially considered adding a RaspberryPi to offshore the ML and signal processing to a second computing device. However, the tradeoff here is that we desire our system to be wireless with respect to the headset. The Emotiv headset uses a Bluetooth wireless dongle to send data from the headset. To reach the target of 500 ms between a user's response to a stimulus and generating the corresponding output, we already need to account for a latency of around 200 ms for the Emotiv Bluetooth receiver to acquire from the headset, process, and setup for API access [1]. If there exists another remote device like a RaspberryPi to acquire, process, and send data again through a wireless system like Bluetooth or WiFi, we expect around an additional latency of 200 ms. This leaves approximately 100 ms for the user to react to a stimulus and for our signal processing and application inter-communication to react to the user's signal, which is a lower flexibility time window for the user. We need more slack in meeting the latency requirements so instead are opting for our primary compute to be performed within the desktop or laptop which the application is being used from to ensure that the majority of the latency is from the Bluetooth connection of the Emotiv Insight and wired latencies within the desktop operating system, providing greater slack for users to react. This will include some work to make sure our software critical path is short and inexpensive as to prevent our application from overburdening the desktop hardware.

C. Model Selection

With the headset chosen for acquiring EEG signals, the Emotiv Insight also comes with software packages to process and interpret signals, which we opt to not use and instead to construct our own.

From experimentation of the EmotivBCI package, which is provided with the Emotiv Insight, we found very low reliability and minimal ability to tune or update the package to meet our accuracy needs. Primarily due to the lack of tuning and the closed source software licensing of EmotivBCI, which shrouds the ability to tune or test the system to meet the 90% accuracy metric, we decided to instead ingest raw EEG data from the headset and process the data using our own models and filters to ensure we have the flexibility to meet the metric. Furthermore, since we only have one headset on hand, we desire our recognition system to be trained specifically for this headset.

Given that the software will be hosted on a desktop computer potentially running other applications, we desire to find models that are lightweight, quick to train, and deploy. Hence, we will be using an assortment of random forest, SVM, and logistic regression models over more complex and expensive models like a neural network. Neural networks are not optimized for CPU processing and require a large amount of data to train successfully. Our project is limited by the small samples of data we need to manually collect. To allow for rapid retraining and use, we cannot deploy a neural network. In comparison, random forests and logistic regression are cheap to train and quick to build given we are choosing a set of features that should be easy to differentiate on and more complex feature sets can be modeled using SVM.

D. Python Tkinter

Our user interface is an essential part of our project because it influences a user's experience while using our application. An inviting, easy to use interface will increase a user's satisfaction.

Initially, we looked into various desktop application development platforms and had decided on using Flutter for its reduced code development time, "hot reload" feature, and its ability to deploy visually appealing apps from a single codebase. However, we decided to go against this option because none of us had experience building an app from scratch and there was a steep learning curve.

After settling on using Python as the language for our project, we considered using Tkinter for the creation of our user interface for its various built-in features. This will allow us to easily interface the frontend with the backend.

VI. System Implementation

A. Signal Processing Algorithm

Our data will be sampled at 128 Hz. The two main branches of processing are spectral analysis (Fast Fourier Transform, or FFT) and temporal analysis. Both will be fed into an ML model.

On the pattern recognition side, the incoming real-time signals will be analyzed continuously and will be classified into different pattern recognition according to our signal input. Since we have real-time data running constantly, the data will be kept in a 30 second data buffer, in which we will analyze the data using a sliding 1.5 second window. Our ML model will be in charge of classifying the data into double blinking, triple blinking, left wink, right wink, and clenching/tongue movement, normal baseline, and unclassified noisy signals. Outstanding features of the patterns in terms of amplitude and data variance will be used to classify the signals into the following categories. A more detailed explanation of the procedure training and classification will be explained in section C, *EEG Classification*.

On the FFT side, the data will be decomposed into 4 different focus signals according to their frequency bands: theta, alpha, beta, and gamma signals. These signals will give us information about the level of focus/relaxation state the user is currently in. This information is what we will be using to activate our pattern recognition system. The lower frequency signals refer to a loss in focus and the higher frequency signals would refer to a higher focus. Since our real-time data will be kept in a 30 second data length buffer, we will FFT for every 4.5 seconds length of data. In this period, if a loss in focus is detected, we will ignore the next 4.5 seconds length of data to save the system from operating on unintentional data. Two main brain waves that we

will be looking for are the theta and alpha range, which measure between 4-8 Hz and 8-12 Hz respectively. Theta and alpha range refer to a slightly sleeping and a calm/relaxed state. Therefore, if there is a substantial increase of amplitude in frequency range 4-12 Hz, for the next 4.5 seconds length of data, we will deactivate our processing algorithm. The exact value of the threshold will be determined through experimentation.

B. Signal Acquisition

The brain signals we will be acquiring are: AF3, which is located on the front left of the forehead; AF4, which is located on the front right of the forehead; T7, which is located above the ear on the left side of skull; T8, which is located above the ear on the right side of the skull; and Pz, which is located on the top of the head. These signals will be acquired through the Emotiv Insight headset which sends signals via Bluetooth to a locally installed EmotivPro desktop application accessible through the Emotiv API interface.

A custom Python script will be created with a thread directly responsible for calling the Emotiv API to set up a connection to the Emotiv device and communicate connection issues to the frontend. Once a connection is established, the same thread will poll sensor and device data from the API stream, strip context data, and dump the relevant sensor data into a local buffer which will have the capacity to hold the previous 30 seconds of polled data to handoff to the classification subsystem. A second thread within the running process will use the currently buffered dataset for classification and logging. Another thread within the process will be actively Fourier transforming incoming data.

C. EEG Classification

The classification subsystem running on a second thread will classify raw EEG data stored within the buffer into double blinking, triple blinking, left wink, right wink, and clenching/tongue movement.

To do this, the classification thread maintains a sliding 1.5 second window on the previous 30 seconds of data within that shared buffer and live computes features from the 1.5 second window. The features computed will be described in more detail below.

The process for distinguishing different blinking and winking patterns is done by extracting features including maximums, slopes, average signal strength, frequency strength, variance, and differences in amplitude in the AF3 and AF4 electrode data and using a random forest model to predict the occurrence and number of blinks and winks.

Teeth clenching and tongue movement will be processed by extracting local maxima and slopes from the T7 and T8 electrode data and using a logistic regression classifier to detect the occurrence of strong clenching or tongue movement.

The models described above will be trained using manually collected data from the Emotiv headset before deployment. There will be at least 100 samples of each training data point for building the model. The ML models will be trained in Scikit-Learn to optimize algorithm performance over smaller data sets. Some visualizations of the sample data we will use for training are provided in Appendix A. This data has already been cut to only have the feature we are detecting within the sample window.

The resulting outputs from the model will be dispatched to a data bus shared by event listeners on the frontend.

D. EMG Signal Acquisition and Implementation

EMG signal is the continuous control input signal that we will be using to produce continuous movement to activate the cursor. We purchased the EMG sensor that will be connected to Arduino and send the data wirelessly to the computer. EMG ranges from 0-15 mV prior to amplification [2]. Shoulder movement would increase the amplitude of the signal which will be used as the right and left digital signals. These signals will be sent wirelessly through the Bluetooth module that is attached to the Arduino and sent to the backend for processing. Processed signals will then be dispatched to the frontend as an event. The right signal will be used for right/up control in the front-end interface, whereas the left signal will be used as the left/down control.

E. Software Interface (Backend and Frontend)

A bus system will be established in which the backend will publish detected features and event listeners on the front-end application will react and activate based on the incoming feature stream. The frontend application will dispatch interface movements accordingly. The features of our application will be displayed on our screen output. Users will be able to perform various tasks, including typing on a keyboard, moving a cursor across the screen, and performing left and right clicks. This will all be coded in Python.

VII. Test, Verification and Validation

The design of our interface is mainly broken down into three parts: signal acquisition, backend signal processing, and the front-end interface. Most of our tests will be focused on the latency, accuracy, and the user satisfaction. As mentioned in the other sections, we require our system to work on real-time data analysis, thus, we would like to ensure that the latency of our two routes of data, EEG and EMG, are within an acceptable range that the user will be satisfied with.

A. EEG signal classification accuracy

For our EEG signal processing testing, we will test on the pattern recognition accuracy. From research paper [3], the general linear technique such as SVM has accuracy up to 97-98% of analyzing the EEG data. Therefore, our goal is to be able to have above 90% accuracy by using either the SVM and/or random forest classification. From our current data sample, we will split 75% of the collected data into a training set and 25% of the collected data into a test set such that the test and training sets both have the same proportion of each movement data point. For different models with different hyperparameters, we will use the training set to train the model and verify the model meets the 90% accuracy metric by seeing if it predicts the test data with below 10% error.

Of the subset of the models we select from this process, we will add each model into the real-time prediction system, add functions to parse the appropriate features that are used by the model to predict, and then attempt to predict features live. For each signal we need to predict, we have the subject we are monitoring in the session produce that signal 20 times in sequence with at least the second pause between each and track

the number of times the model predicted the movement occurring correctly. We track the number of times the model predicted incorrectly, including falsely predicting a feature existing when it did not occur and not recognizing the movement when it occurred, and divide it by 20 to get a percentage error for each movement we are testing. We then take an average across all the different movements to get an overall error prediction percentage. We aim for this number to be below 10%, meaning 90% of the time the model detected the movement in streamed EEG data or did not detect a movement that did not occur.

B. EEG full system pipeline latency of <500 ms

A human benchmark test that is available online will be used to test speed of the control from the EEG signal acquisition to the output on the interface. The user will react to the stimulus, in this case clicking the screen in the test without any planar control from the EMG. This will be a good approximation to see the latency sum of acquiring the data from EEG headset, wireless transfer, backend signal processing, and the sending of data over to the front end. The entire process is expected to be less than 500 ms. To ensure the least and consistent latency from the other computer programs, we will be running only our API in the background and all other applications have to be closed.

C. Integrated system: accuracy and speed

This test will be used to test the accuracy and speed of the control from the EMG signal with the clicking ability from the EEG signal. We will be using the point and click test that is created by ourselves. Circles with 100 pixel-radii will appear randomly on the screen every five seconds and stay there for 10 seconds. The user will need to click on those circles within the 10 seconds before they disappear. The test will be for a one-minute period, and we will aim for the user to be able to click on 75% of the overall targets.

D. User Experience

User experience is one of the important areas of focus of our product. Even though our interface is not able to allow a multidirectional control (eg. diagonal movement), we hope that the experience of controlling an interchangeable directional plane provides satisfaction to our subject group. Since there is a challenge in finding test subjects of our interest, we will conduct a user experience survey based on the users with no disabilities. They will be asked to rate our product based on a scale from one to ten on whether they would recommend our product to someone in our use-case group or not. This scale will be linear, with 1 being a no, 5 for neutral and 10 for a definite yes. We aimed to have 15 subjects and hope to receive 70% confidence in our product in this test.

VIII. Project Management

A. Schedule

Currently, we are transitioning between our design/setup phase and our first execution phase. We have run and recorded several sets of data from the EEG device, and that data is being used to train and test our machine learning models. This is a part of our signals research phase, as we are currently constructing an effective way to build and train different ML models and testing their accuracy in live detection. We have also begun playing around with the EMG sensors and connecting them with our Arduino.

We are moving into the implementation of the software phase, where we will be building out the frontend of our application. Currently, we have a keyboard with tab, arrow, and cursor features. In the coming weeks, we will be adding new features and connecting the frontend with the backend. See Appendix B for more details.

B. Team Member Responsibilities

Our work is split into a few different categories, including signal processing, hardware components, and software environment/application. Although our project tasks are roughly split along these categories, we will work together to complete the tasks. The assignment of responsibilities serves more as an indicator for leadership of the task.

Jean has the most experience with signals and has taken advanced digital signal processing courses, so she will be mainly in charge of creating the signal processing procedure and algorithm, as well as conducting research on EEG and EMG signals. She will work on creating the signal processing structures for the model and will be working in analyzing data to extract the main features that will be used to train the ML model.

Jonathan will be working on the hardware/software interfacing. After obtaining the data from the Emotiv Insight headset, he will build pipelines for parsing the data and build a testbed for training and testing ML models. This includes testing within the data set and a live detection testbench where the model will attempt to detect continuous live inputs. He will then create the acquisition and processing production backend that will communicate to the user frontend.

Wendy will be working on the software end, primarily on the frontend side. She will create the interface and add various features to our project. She and Jonathan will work to connect the frontend and backend components together for the final application.

C. Bill of Materials and Budget

To date, we have purchased all the components outlined in our design, and around half of the components have arrived. Even though we ordered a few components that might not make it into our final system, this was a conscious decision as they were purchased as spares. We also ordered replacement components in the event of component failure. Fortunately, there was already an EEG device in the 18-500 inventory, so we did not have to spend a significant portion of our budget purchasing that. We were given a budget of $600 and have spent just above $226. See Appendix C for more details.

D. Risk Mitigation Plans

One of the major risks we might encounter is how accurately we can detect the desired features from the EEG sensors. Furthermore, we also risk having an unsmooth and difficult to use interface if the signals are difficult to pinpoint from the user.

To combat this, we are already including two EMG sensing devices which will only require a simple threshold cutoff for

triggering display interactions. These can be rigged to generate continuous movement and be configurable as the primary controllers of desktop movement, especially the mouse, with EEG headset acquired signals used to trigger less commonly used shortcuts.

As a minimum target, if our system can reliably detect two user-controllable signals from the EEG headset, we have enough flexibility to define how patterns of those signals will correspond to interface interactions. While this would be convoluted and difficult to pick-up in the beginning, we can guarantee high accuracy on detecting these signals and our system will still be usable, albeit with each action being a unique combination of the 2 binary signals we acquire.

Another potential risk we may face is if the same set of features we would like to detect are very different from person to person. We can mitigate this by using the EEG headset and manually calibrating a set of data from a user and only using their unique signals to train the recognition model before it is used. We can also port this as an extra feature of our interface, so the user doesn't have to navigate to another application before using the system.

If the EEG headset acquired signals prove to be extremely unreliable, we also plan to pivot to using a desktop camera and using object detection with OpenCV to recognize facial movements that we can map to user actions.

IX. Related Work

There are currently no other projects or products that incorporate both EEG and EMG signals in the way that we have planned.

A lot of research has been done to study the 2D directional control from motor imagery for people who have fully lost their arm muscles. There are similar projects that take in EEG signals to study and control the cursor on the BCI program [4][5]. However, there is no product that is available in the market yet that allows the integration of the BCI discovery to create a platform on the desktop computer. Moreover, with how expensive a standard EEG cap costs, we chose to use an EEG headset as it is more commercialized. With the trade-off between the price and data acquisition to the spatial and temporal resolution, we are integrating EMG sensors to have precise control data from the shoulder muscle movement instead. We hope that our interface will allow amputees or people with limited physical ability to use computers on a daily basis and without much difficulty.

Previous Capstone projects involving EEG or EMG signals include a game called *Myorun,* from Fall 2020, which allowed users to physically workout their forearm using a game and another game called *Traveling Mind,* from Spring 2021, which implemented a mind-controlled iOS game. Our project hopes to integrate mind-control and EMG to not just play a game but to control more general computer interfaces.

X. Summary

The design and plan of work for our project has undergone many iterations. The systems and components have been planned and changed to best fit our design goals, as well as be feasible to implement. Our plan has also changed as we have reevaluated our skill sets and areas of expertise. So far, however, we have not done any testing with all our components linked together. This is our next biggest challenge, as we have dedicated time in our schedule in the upcoming weeks to do testing.

We are excited to implement our design and look forward to improving desktop accessibility for those suffering from upper body disabilities. If there is anything we have learned in the first half of this course, it is that we will maintain an open mindset project and be open to any changes or modifications to our design.

Glossary of Acronyms

API- Application Programming Interface
CPU- Central Processing Unit
EEG- Electroencephalography
EMG- Electromyography
FFT- Fast Fourier Transform
ML- Machine Learning
SVM- Support Vector Machine

References

[1] "The latency of Emotiv data streams," *EMOTIV*, 24-Aug-2021. [Online]. Available: https://www.emotiv.com/knowledge-base/the-latency-of-emotiv-data-streams/. [Accessed: 28-Feb-2022].

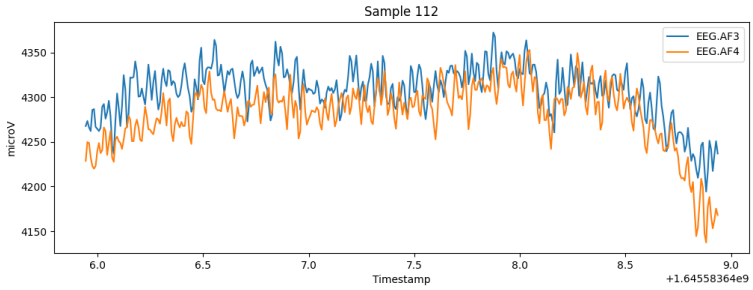
[2] M. B. I. Raez, M. S. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: Detection, processing, classification and applications," *Biological procedures online*, 2006. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1455479/. [Accessed: 02-Mar-2022].

[3] H. U. Amin, W. Mumtaz, A. R. Subhani, M. N. M. Saad, and A. S. Malik, "Classification of EEG signals based on Pattern Recognition Approach," *Frontiers*, 01-Jan-1AD. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fncom.2017.00103/full. [Accessed: 03-Mar-2022].

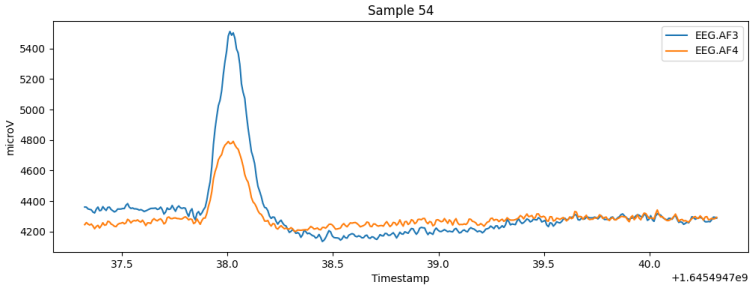
[4] Yuanqing Li, Chuanchu Wang, Haihong Zhang and Cuntai Guan, "An EEG-based BCI system for 2D cursor control," 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) [Online]. Available: https://ieeexplore.ieee.org/document/4634104?arnumber=4634104 [Accessed: 03-Mar-2022].

[5] J. R. Wolpaw, D. J. McFarland, G. W. Neat, and C. A. Forneris, "An EEG-based brain-computer interface for Cursor Control," *Electroencephalography and clinical neurophysiology*. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/1707798/. [Accessed: 03-Mar-2022].

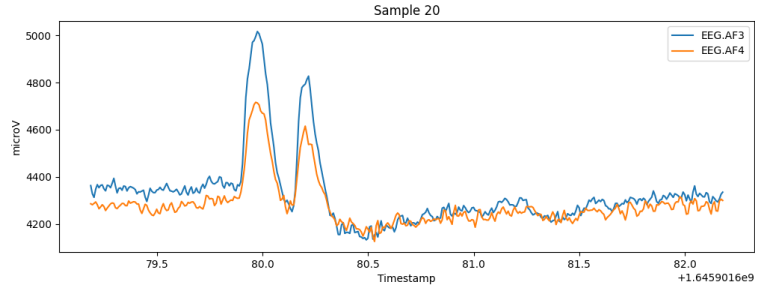
Appendix A



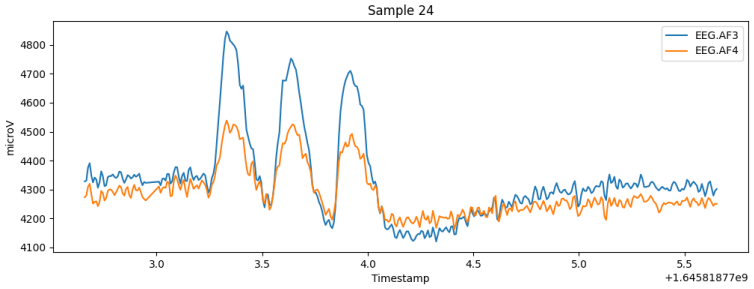
(a) Typical baseline sample with no features



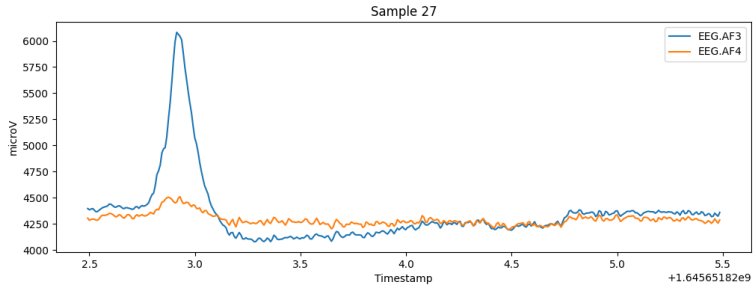
(b) Typical blink sample



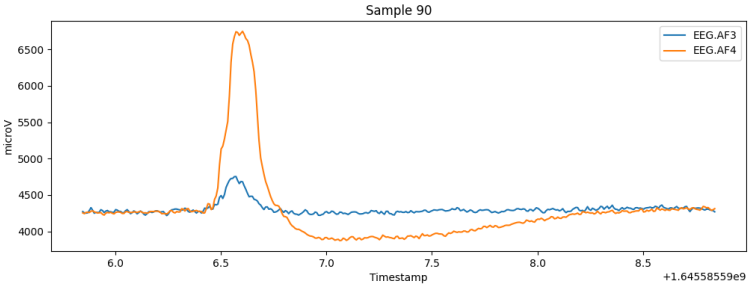
(c) Typical double blink sample



(d) Typical triple blink sample



(e) Typical left wink sample



(f) Typical right wink sample

Fig. 5. Visualizations of the sample data from the EEG device that we are using for training.

Appendix B

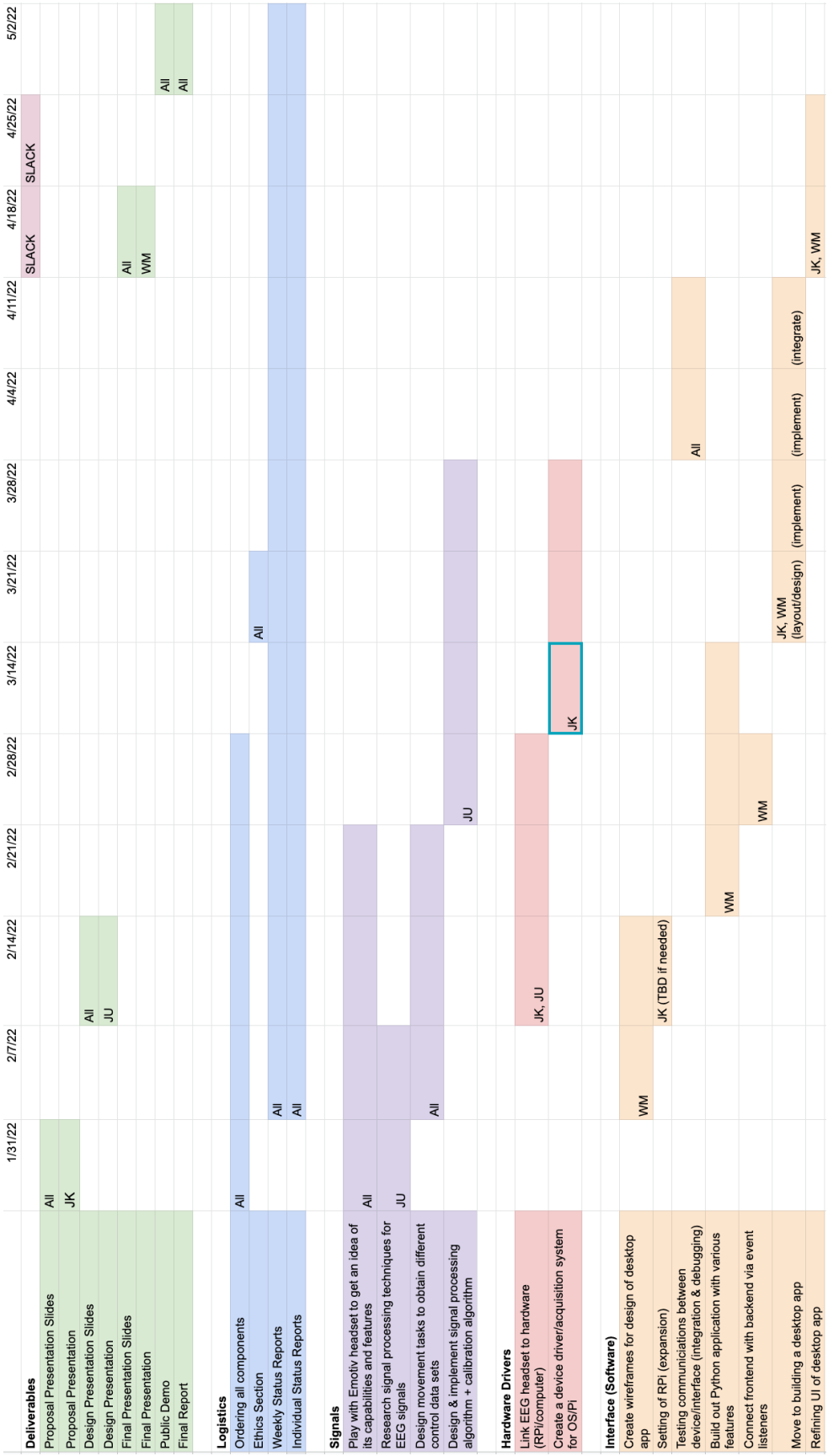


Fig. 6. Schedule

Appendix C

Item	Quantity	Manufacturer	Source	Price	Arrived?
Emotiv Insight Model 1.0	1	Emotiv	18-500 Inventory	$0.00	Yes
Replacement EEG sensors	2	Emotiv	Emotiv	$61.94	No
EmotivPRO Student License	1 license ($30/month x 3 months)	Emotiv	Emotiv	$90.00	Yes
EMG Sensors	2	N/A	Ebay	$44.61	Yes
EMG gel electrodes	1 pack of 100 electrodes	3M	Amazon	$18.50	No
9V battery	1 pack of 8	Amazon	Amazon	$11.59	No
Arduino Nano	1	Arduino	18-500 Inventory	$0.00	No
				Grand Total	**$226.64**

Table I. Bill of Materials