**CogniSync: Milestone 5**

**TO**: Aaron Maus

**FROM**: Justin Haysbert, Gabriel Sagrera, Shayne Shelton, Ryan Stevens

**DATE**: 02/07/24

**GitHub**: <https://github.com/stevensryanw/BCI_Infinity>

**FALL 2023 SEMESTER RECAP**

* Developed and Outlined GUI using Tkinter
  + User training (Our user training prompting scripts)
  + Model selection (SciKit-Learn, PyTorch, TensorFlow)
  + Model output (keyboard strokes based on predictions)
    - Snake Game
    - Model Wheelchair
* Created data collection protocol
  + Step-by-step instructions for participants to use on collection days
* Filed IRB
  + Filed initial IRB and now revising for human research approval
* Researching data input methods
  + Open BCI stream eeg module via serial (PyOpenBCI, NeuroPype)
* Exploring Python modules that aid in BCI research
  + NeuroPype - a platform with processing components for BCI
    - Allows for custom data filtering, real-time data streaming, and visualization
    - Pipeline design
    - Custom algorithmic design
* Printed a large headset and glued it all together, ready for data collection
  + Will begin printing medium and small in the Spring
* Tools and software researched but put aside for now
  + Gathertown
  + NeuroPype
  + PyAutoGUI

**GOAL PROGRESS**

*IRB Updates*

After meeting with the IRB managers, it was determined that our project does not contain human subjects research. We have been given the green light to begin training models on our data. We have also been given the go-ahead to collect the data of others to train models as long as the data is used in the scope of the Expo. If we seek publication for our work, we must complete an IRB as we are “adding to general knowledge.” We have received a verbal confirmation but must complete a quick IRB form to obtain written approval.

*GUI*

After developing a basic framework for the CustomTkinter GUI, the team developed individual pages and connected them to the main GUI.

User Training Page  
The team designed a user training page to complete the GUI so that training sessions can begin. This page should allow the user to begin a training session by starting a data stream in concurrence with a prompting script. The user is prompted to complete a movement for 10 seconds, wait a 10-second rest period, wait a 5-second preparation period, and then repeat the process 40 times for each moment. The prompting script is also written to randomize the order of the movements. This prompting script has been integrated within the GUI; however, we are still working on adding labeling functionality and openBCI GUI integration.​ This page should be completely done within 1-2 more coding sessions.

Model Selection

The team has developed a model selection GUI that allows the user to load a CSV from a previous data collection session to train a classification model. The GUI allows users to select models such as LDA, SVM, and CNN. Training will begin once the user picks a model type, CSV data file, and a name for the saved file. This model will then be used to classify movements in real-time during the snake game and with the model wheelchair.

Snake Game

The team developed a working Snake Game using Python and Tkinter. We then utilized its logic and incorporated the code into a frame that fit onto the pages of the joint GUI. We now have code for a functional snake game fully integrated into the GUI framework that will be used with headset data.

The only improvements needed in the game are to incorporate a scoreboard to track performance and give the user a goal to achieve when operating the object. Additionally, the team may eventually add online training to train the model further while playing. If so, we may need to change some of the game logic, and key binds to feed and/or stream data elsewhere.

*Connecting Headset to GUI*

We have been working on two possible solutions to connect our OpenBCI Cyton headset to the GUI. The first we tried was Neuropype, which required a student license to use it. After working on using Neuropype for the GUI connection, the deprecations of the connection between OpenBCI Cyton and Neuropype made us change paths and start on a new package. The new package we are working with is PyOpenBCI. Unfortunately, it is also deprecated but still usable.

NeuroPype

We have decided not to use the Neuropype program to assist us in the live data collection due to the multiple issues we found when using it. The neuropype software currently only works on Windows or Linux computers and not on Mac, allowing only one of the group members to work on the neuropype code. There were also issues with the Neuropype software as it relied on a Python package that did not exist anymore and had changed to a different package, requiring significant changes to the files in Neuropype. The software also needed to recognize the port the computer was receiving the data stream from but could not recognize the ports on the Windows laptop we had access to. Due to the issues, we decided to look into different options in the meantime to allow us to continue the project quickly.

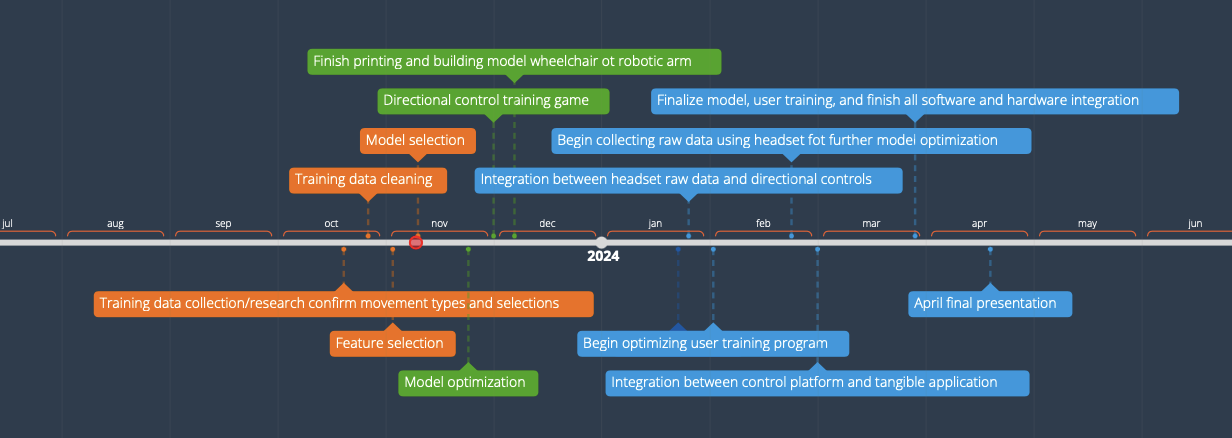
PyOpenBCI

In the struggles with Neuropype, we started researching additional ways to integrate our headset and GUI. We quickly found a Python 3 package provided by OpenBCI called PyOpenBCI. This package provides us with a way of connecting Python and our headset both automatically or manually with the serial port. The established connection uses a callback function to send the data where we choose. Our current implementation for a callback function uses the example lslstreamer() function from the package GitHub. For use with GUI, we have begun modifying the function to write to a CSV with the channel\_data, aux\_data, and our added label from the GUI prompter.

*Control Output (Model Wheelchair)*

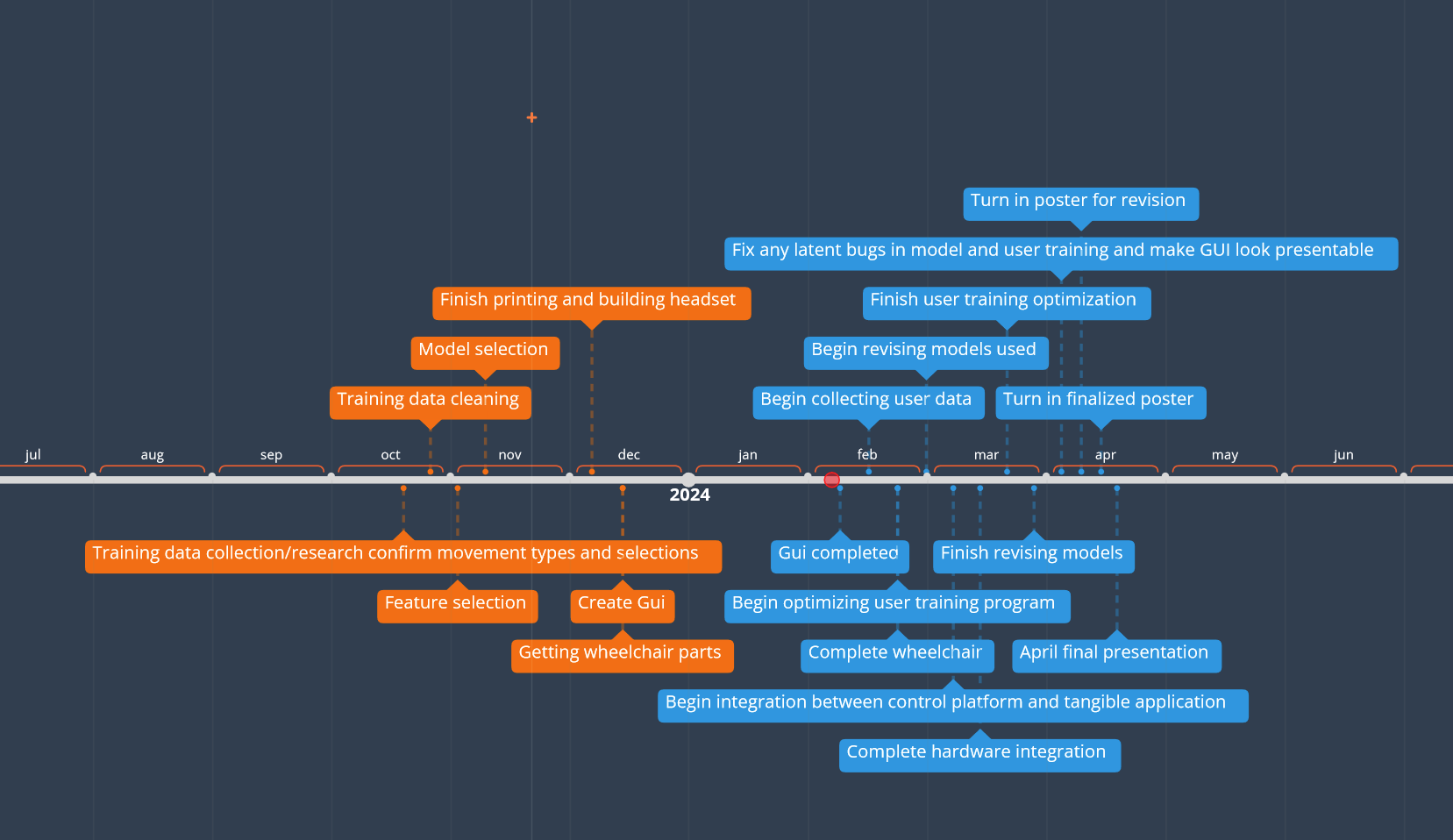
Most of the team’s time and attention was dedicated to finishing the GUI, so we could start recording sessions. Because of this, we devised a list of needed parts, ordered them, and now have them in their position. We have everything we need to construct the device and, eventually, code it to work with a controller.

**PLAN UNTIL SSE EXPO (04/23/24)**



***Figure 1*** - Milestone 3 Updated Timeline

In *Figure 1*, the orange bubbles represent the goals we had completed by milestone 3 with the green being what was immediately next and blue showing goals in the second semester. At this point in the project we were thinking of doing both a wheelchair and a robotic arm. We had also planned to have started collecting data in October by gathering data from online sources and using that to optimize our model. We ran into some issues with using data from online sources as with most of the datasets we could not confirm where the researchers were placing the electrodes and ones who said where the electrodes were being placed did not match the headset we planned on using. Due to this we pivoted away from online data sets and decided to use data we collected on the headset we had gotten.



***Figure 2*** - Newly updated project timeline

In *Figure 2*, the color scheme remains the same as before but has been updated to have our weekly goals for the project. Due to us only getting the headset a few days before winter break, we are only now able to collect data. Before the semester began we had begun to create the GUI we planned to use in our data collection and to have all parts for the project in before the end of the semester and to have the GUI done in the beginning of the second semester to begin data collection as soon as possible. Our next immediate goals are to begin collecting data and to complete the wheelchair. Once some data has been collected we plan on starting our model optimization and begin the hardware integration simultaneously to help save on time. We have also budgeted our time so that most of the project finishes at the end of march so that we can just make small adjustments and focus on the poster in April.