UNIVERSITY OF MARYLAND

Neuromechanics Laboratory

Fischell Department of Bioengineering

University of Maryland

HoH Python Documentation

Michael Penafiel

Directed by Dr. Li-Qun Zhang

Supervised by Giovanni Oppizzi

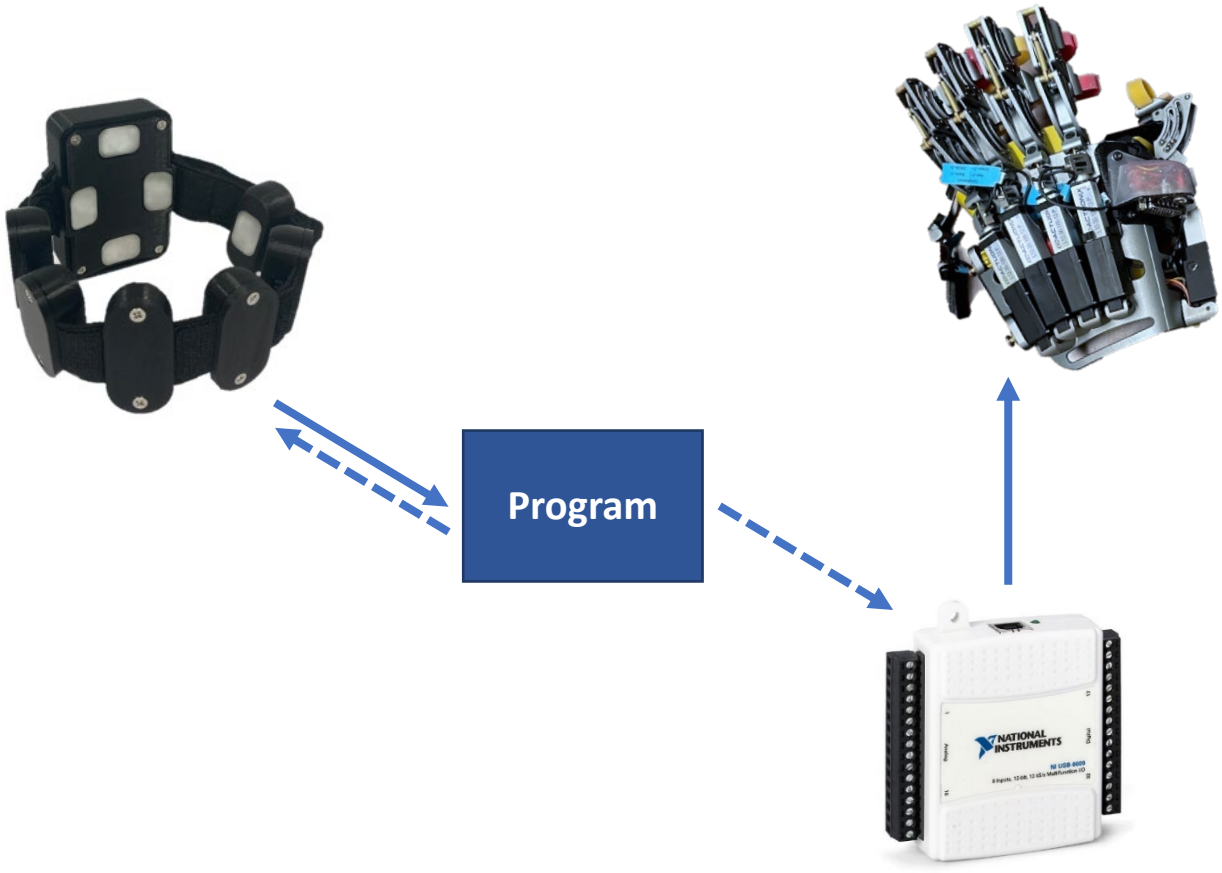
Contents

Introduction

EMG Pose Classification is a system developed in the Neuromechanics Laboratory with the intended use for stroke rehabilitation. This document highlights the development of the Python code used for data collection and training. The current iteration of code has two main functions, to collect data and to test the Hand of Hope (HoH). The first function relies on the MindRove EMG Armband, which will collect data based on different hand poses that the patient assumes. This data will then be fed into a machine-learning environment to train a model. The second function is to test that model against a protocol fed into the HoH. It will receive commands based on the model. Currently, the protocol is hard-coded; however, the goal is to input a protocol and test the model directly against it.

Technical Documentation

The program serves two main functions and requires multiple levels of communication between all the hardware. It sends and receives signals between the MindRove Armband, to communicate when to collect data. The program must also communicate with the NI USB-6009 Multifunction I/O DAQ system. This powerful device can read and write analog signals, which creates a simple path to communicate with the HoH. This required a driver to be installed onto the computer before it could function as needed. The diagram below shows which signals are read and written by the program and hardware.



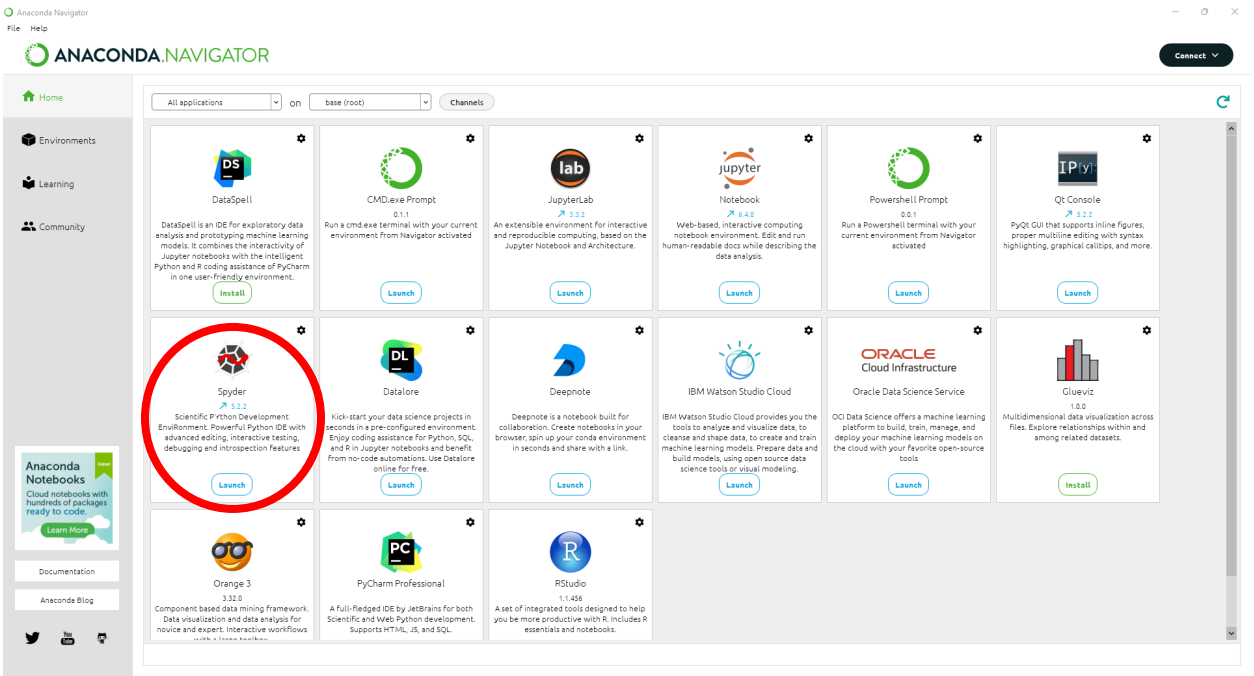
The code was developed in a Spyder IDE using a Conda Environment. This provided several additional libraries that were necessary to process data. We also used several different packages that had to be imported into the environment. This included **nidaqmx**, **PyQt5**, and **mindrove_brainflow**. The GUI was written using **PyQt5**, which incorporates seamless integration of threading. **mindrove_brainflow** was used to communicate with the EMG Armband and acted as a wrapper for brainflow to act. **nidaqmx** was used to communicate with the I/O. These three packages make up the bulk of the written code and create multi-level communication with the Armband, I/O, computer, and HoH robotic hand.

Installing

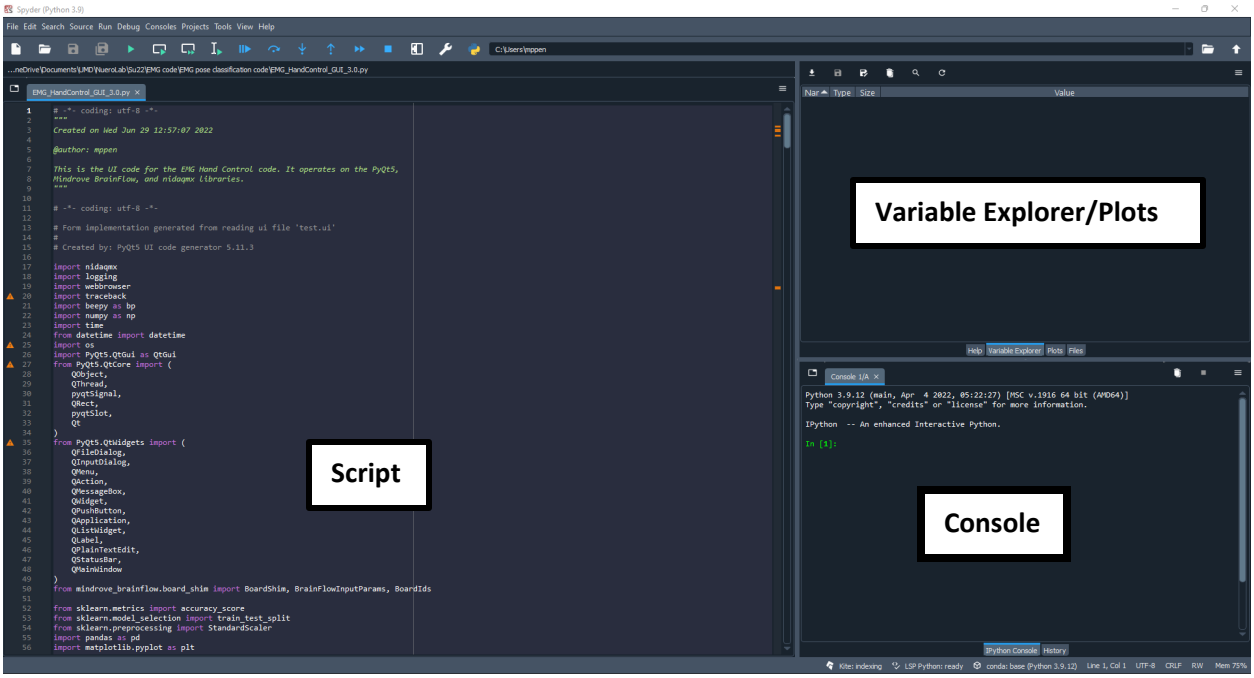
The following software is required.

- Anaconda
  - Installing on Windows
- Spyder IDE (w/ Python 3.7-3.10)
  - Spyder is preinstalled with Anaconda and can be accessed using Anaconda Navigator. Use version 5.2 or later while operating the program.
- NI-DAQmx driver
  - Note: Install programming environments such as NI LabVIEW or Microsoft Visual Studio® (Community release) before installing this product.

The program is run using Spyder and accessed using Anaconda Navigator. When opening Anaconda Navigator, a hub of Python and R applications will appear. Spyder can be launched from this page.



When Spyder launches, the following Integrated Development Environment (IDE) appears.



Many packages come pre-installed in anaconda, however, not everything is included. Install the following modules by copying each line into the console of Spyder and pressing **ENTER**.

```
pip install brainflow
```

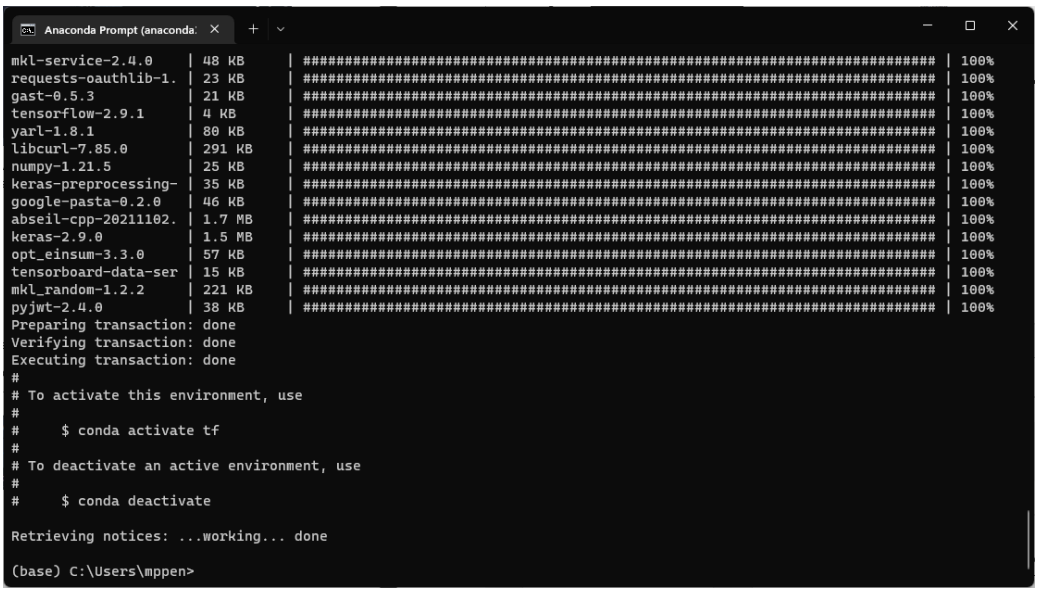
```
pip install mindrove_brainflow
```

```
pip install nidaqmx
```

```
pip install beepy
```

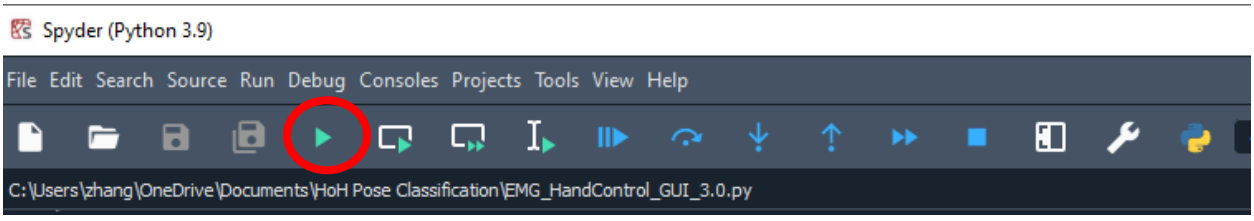
```
pip install simpleaudio
```

The last requirement is [TensorFlow](), which can be installed in the Anaconda Command Prompt.



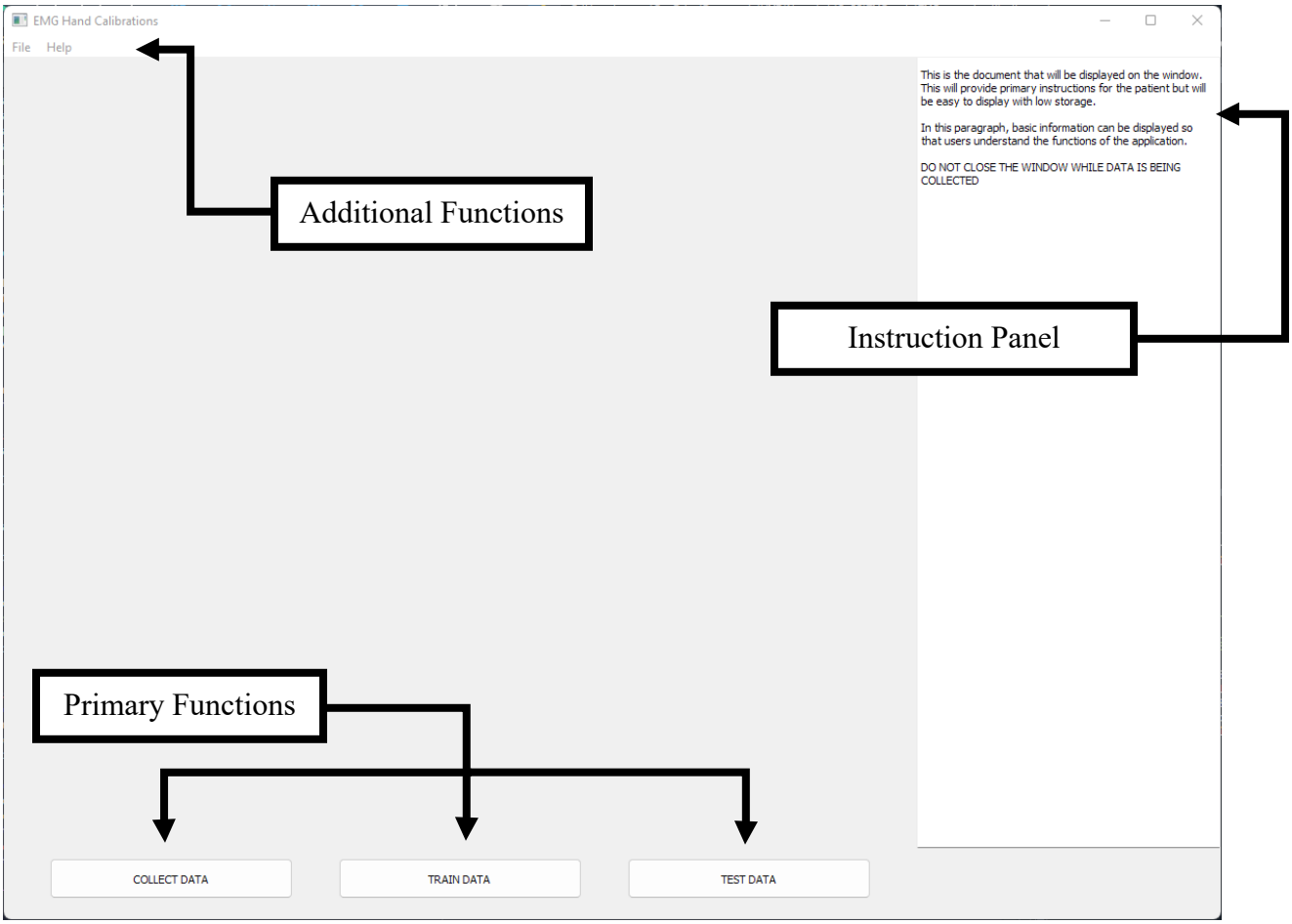
Before running the script, connect all hardware to the computer. This includes the NI DAQ, which connects via USB and the MindRove Armband, which connects via Wi-Fi. To connect, turn it on, then look at the available Wi-Fi networks in settings. It will appear with the format “MindRove_ARB_######” followed by a serial code. If a password is required, use *#mindrove*. Once a connection is made, the user should be able to stream and collect data now.

To run the entire script, press the play button located on the menu bar.

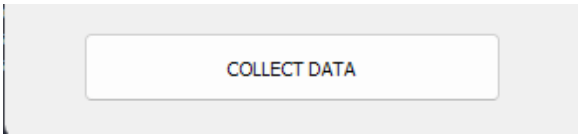


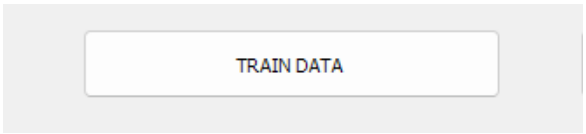
User Guide

When opening the application, a window should appear. This is the User Interface (UI). The instructions for running the test will be in the right-hand portion of the screen, as well as a message board. There are also three buttons located a**t the bottom of the window.**



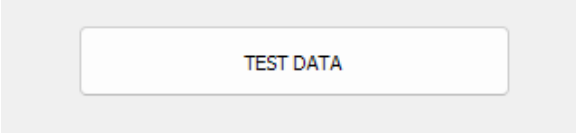
There are three primary functions that are located at the bottom of the UI. The one on the left initiates communication between the MindRove Armband and the interface. It will stream and collect EMG data to the program that will later be used to train a model. NOTE: there must be a Wi-Fi connection between the MindRove armband and the computer. If there is no stable connection, there will be an error and the program may crash.





The button in the middle is used to train the data that was collected during the current session. When pressed, it will create a model and train it with given parameters. The progress can be seen in the Spyder IDE. Once the model has been trained, the program prompts the user to create a name for the working model.

The button on the right will test that data while training the patient against the same or new prompt. The interface is connected to the HoH and will assist the patient to assume the pose on the screen.



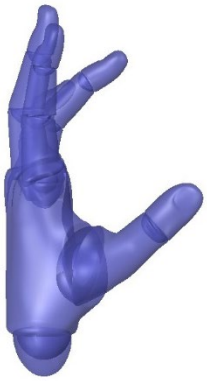
When clicking one of the buttons, a countdown will begin that will prepare the patient. A series of images will then be displayed on the UI that the patient must replicate. The MindRove Armband will only be used when the program is collecting data. When testing the data, the HoH will assist the patient to assume the poses. **Table 1** shows all the available poses and their associated code.

Pose 0: rest

Pose 1: open

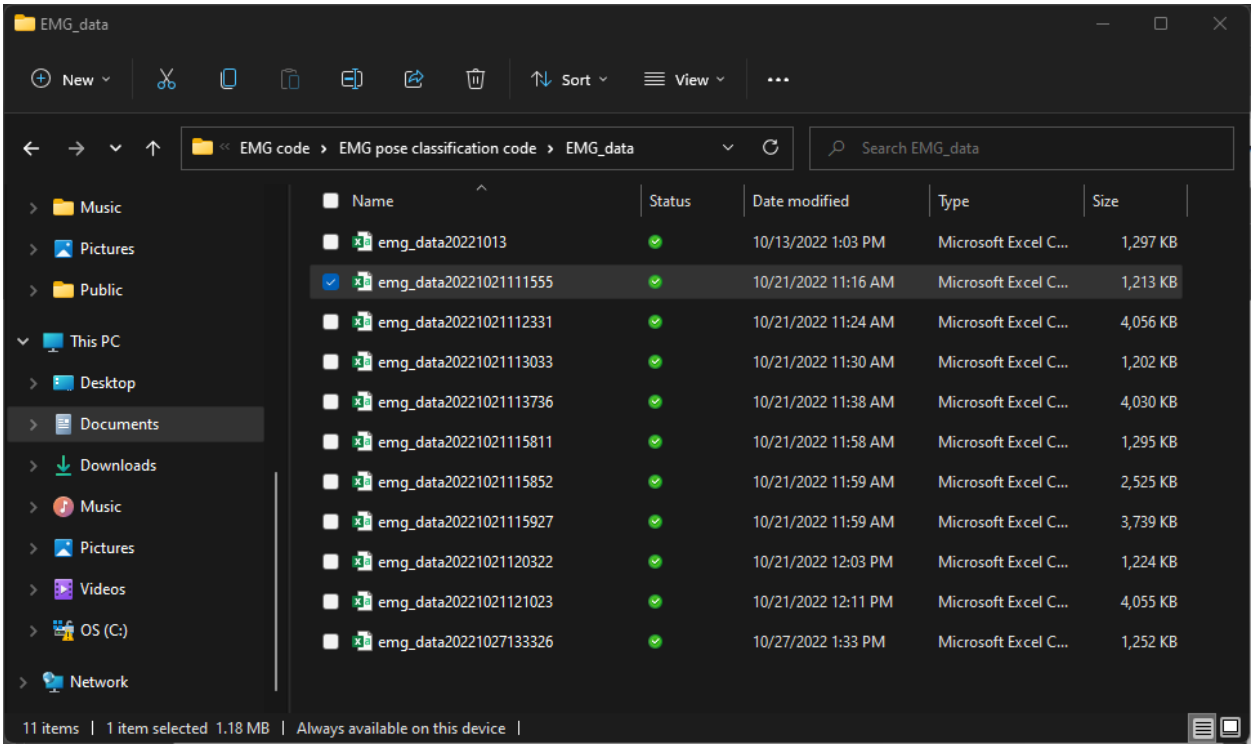
Pose 2: close

Pose 3: tripod

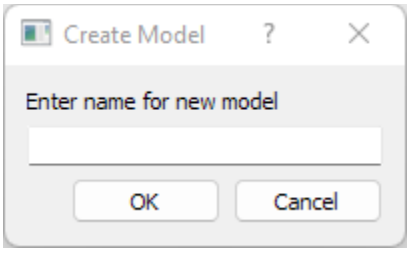


Pose 4: tripod open

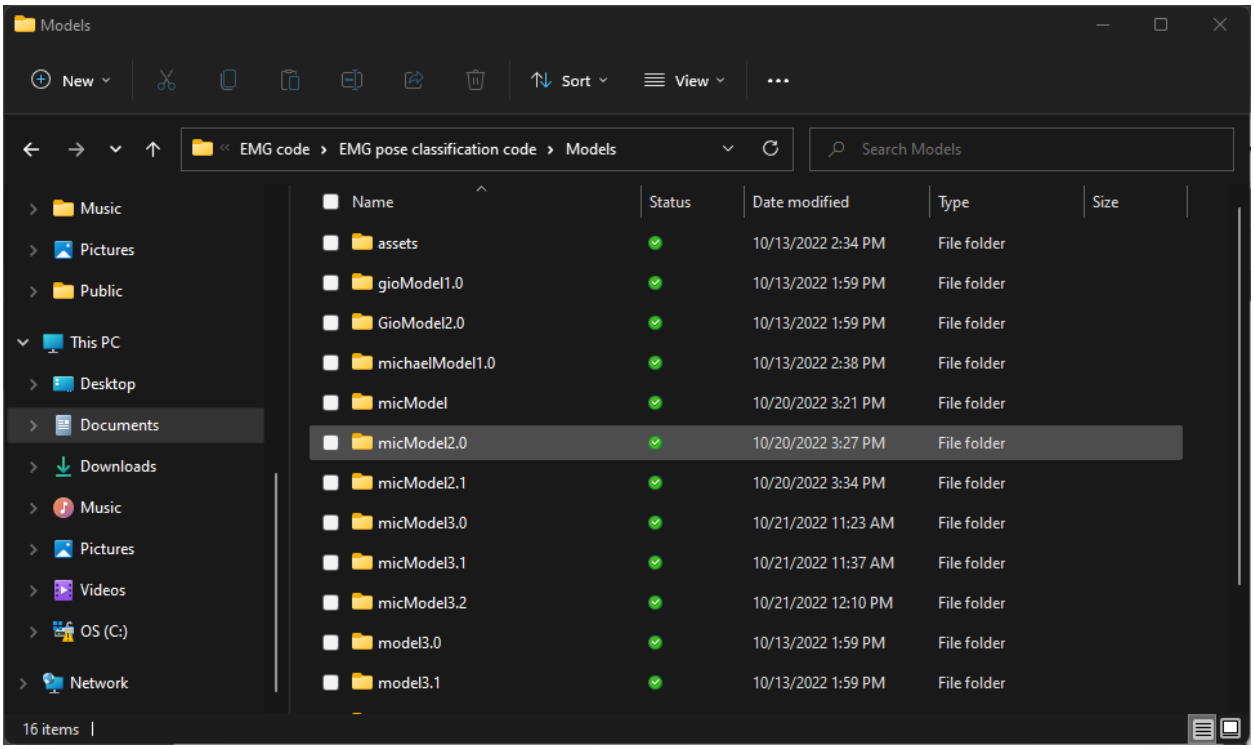
Collected data from the MindRove Armband will be saved as an Excel file in a specific destination on the user's computer. To ensure that the program works as intended, create a folder titled "EMG_data" as this is the designated location. It should be in the same folder as the main python program *EMG_HandControl_GUI_#.#*. This will limit errors that occur when saving data. Each trial run will be saved with the timestamp when the trial started. Below is a screenshot of the EMG_data folder with several different trials saved.



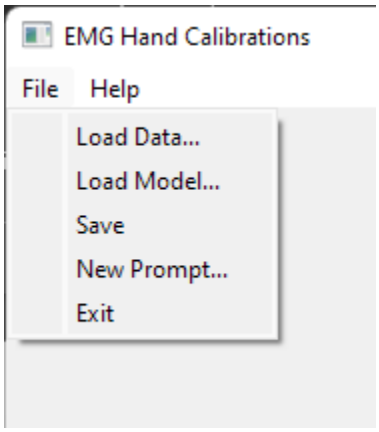
After the model has been trained, a prompt will appear to name it.



Like the data, the model will then be saved in a folder. Make sure there is a folder titled "Models". It should be in the same folder as the main program.



The default prompt at the beginning is [0,2,0,1], which codes rest, close, rest, and open. This prompt can be changed under *File → New Prompt...*.



Under the same file tab, previous data sets and models can be loaded into the program. This is done either to retrain a model or test a model against a previous data set.

Troubleshooting

Resources

The following links provide the documentation for the three imported libraries and can be used as a reference when reviewing code segments.

1. https://brainflow.readthedocs.io/en/stable/UserAPI.html#
2. https://nidaqmx-python.readthedocs.io/en/latest/index.html
3. https://doc.qt.io/qtforpython/api.html