Mindrove Armband

Cody Huynh

Contents

[Useful links 2](#_Toc145980873)

[Setup Guide 3](#_Toc145980874)

[Connecting armband 3](#_Toc145980875)

[Weekly report 3](#_Toc145980876)

[Week 1 3](#_Toc145980877)

[Python script notes 3](#_Toc145980878)

[Week 2 4](#_Toc145980879)

[Week 3 5](#_Toc145980880)

[Week 4 7](#_Toc145980881)

# Useful links

* Mindrove Armband website
  + <https://mindrove.com/armband/>
* Mindrove SDK
  + <https://docs.mindrove.com/>
* Mindrove visualizer
  + <https://mindrove.com/downloads/>
* Mindrove manual
  + <https://mindrove.com/wp-content/uploads/2023/04/UserManual_v2_3_2.pdf>
* Python api reference
  + <https://docs.mindrove.com/UserAPI.html#python-api-reference>

# Setup Guide

This part of the document serves as a guide outlining the step-by-step process of setting up and utilizing the Mindrove armband. It covers essential instructions on connecting the armband to your computer and extracting data from it using Python.

## Connecting armband

1. The Mindrove armband uses wifi to connect to the PC. It is recommended to use the WiFi dongle provided (so that you can continue using the internal WiFi of your machine for internet access).
2. To connect to the device turn it on and look for “MindRove\_ARB”   
   A close up of a number

   Description automatically generated
3. When you connect to it the password is “#mindrove”
4. To use the visualizer, go to the link section in this document under Mindrove visualizer, there will be the link to download the visualizer.
5. Upon the first start-up, Windows asks about communication permissions of the app. Please allow communication on both private and public networks.
6. The app also prompts the user to choose a notch filter that can be 50 or 60 Hz according to the line frequency that is applied in the given country. If the user does not want to set the filter (choosing None).

# Weekly report

## Week 1

This week I got the armband to connect to my pc using the include wifi dongle, use the MindRove Desktop App to see the outputs of the sensor, and run a python script to connect and output the data form the armband. The data is still in a form that is not understandable.

### Python script notes

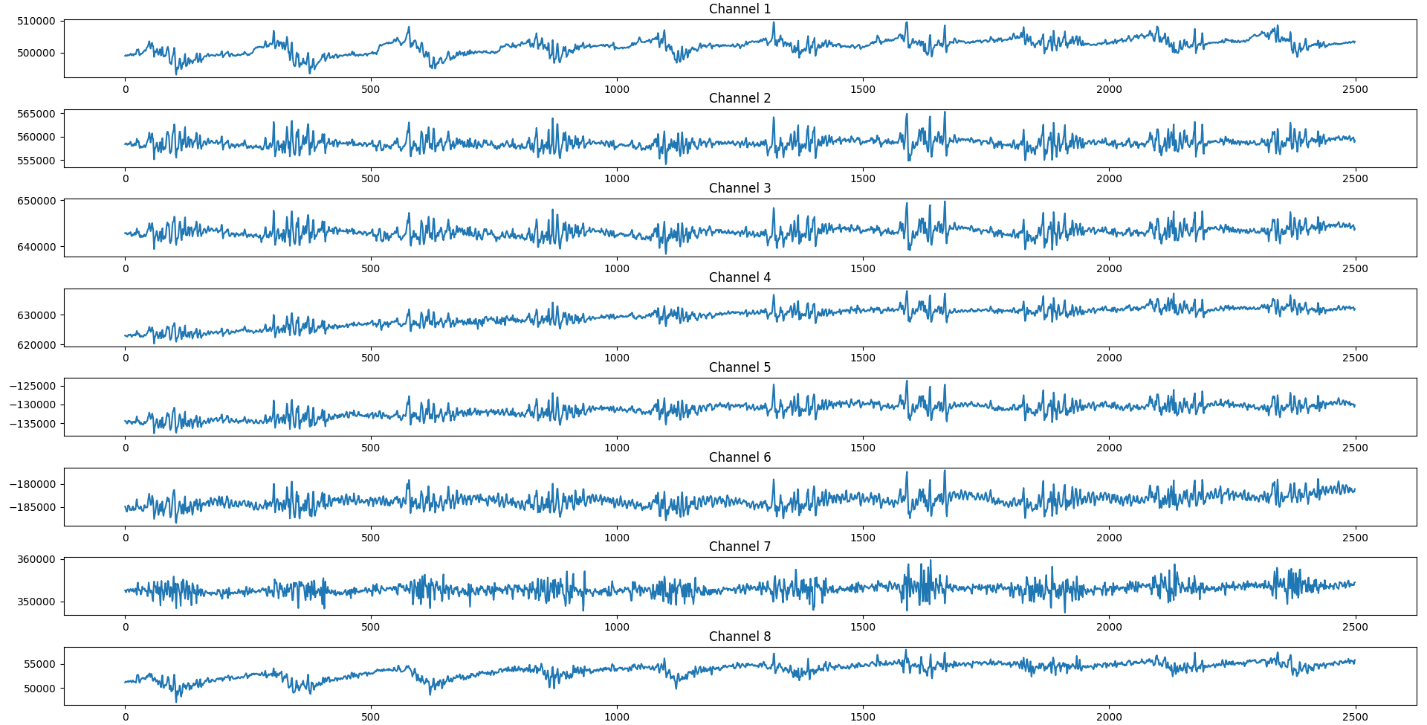
* Use the python sample code (<https://docs.mindrove.com/Examples.html#python-get-data-from-a-board>)
* Use (python armband.py --board-id 0) to run the code. 0 is the board id for the armband.
* Output of the code  
  A computer screen shot of a program

  Description automatically generated

## Week 2

This week I got the data and separate the eeg data and stored it in a csv file.

## Week 3

This week I use pandas and matplotlib to display the data on a graph for each channel.

Code:

import argparse

import time

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import mindrove

from mindrove.board\_shim import BoardShim, MindRoveInputParams, BoardIds

from mindrove.data\_filter import DataFilter, FilterTypes, AggOperations

def main():

    period = 5  # time to record in seconds

    fileName = "data.csv"

    BoardShim.enable\_dev\_board\_logger()

    params = MindRoveInputParams()

    board\_id = BoardIds.MINDROVE\_WIFI\_BOARD

    board\_shim = BoardShim(board\_id, params)

    board\_shim.prepare\_session()

    board\_shim.start\_stream()

    eeg\_channels = BoardShim.get\_eeg\_channels(board\_id)

    accel\_channels = BoardShim.get\_accel\_channels(board\_id)

    sampling\_rate = BoardShim.get\_sampling\_rate(board\_id)

    time.sleep(period)

    num\_points = period \* sampling\_rate

    data = board\_shim.get\_current\_board\_data(num\_points)

    board\_shim.stop\_stream()

    board\_shim.release\_session()

    eeg\_data = data[eeg\_channels]

    accel\_data = data[accel\_channels]  # output of shape (3, num\_of\_samples)

    print(eeg\_data)

    print(sampling\_rate)

    df = pd.DataFrame(eeg\_data)

    df.to\_csv(fileName, index=False)

    df = pd.read\_csv(fileName)

    ####### plot the data #######

    # Create a single figure with subplots

    fig, axes = plt.subplots(8, 1, figsize=(20, 14))

    for row in range(0, 8):

        data = []

        for x in range(df.shape[1]):

            data.append(df.iloc[row, x])

        axes[row].plot(data)  # Use the respective subplot for each row

        axes[row].set\_title(f"Channel {row + 1}")

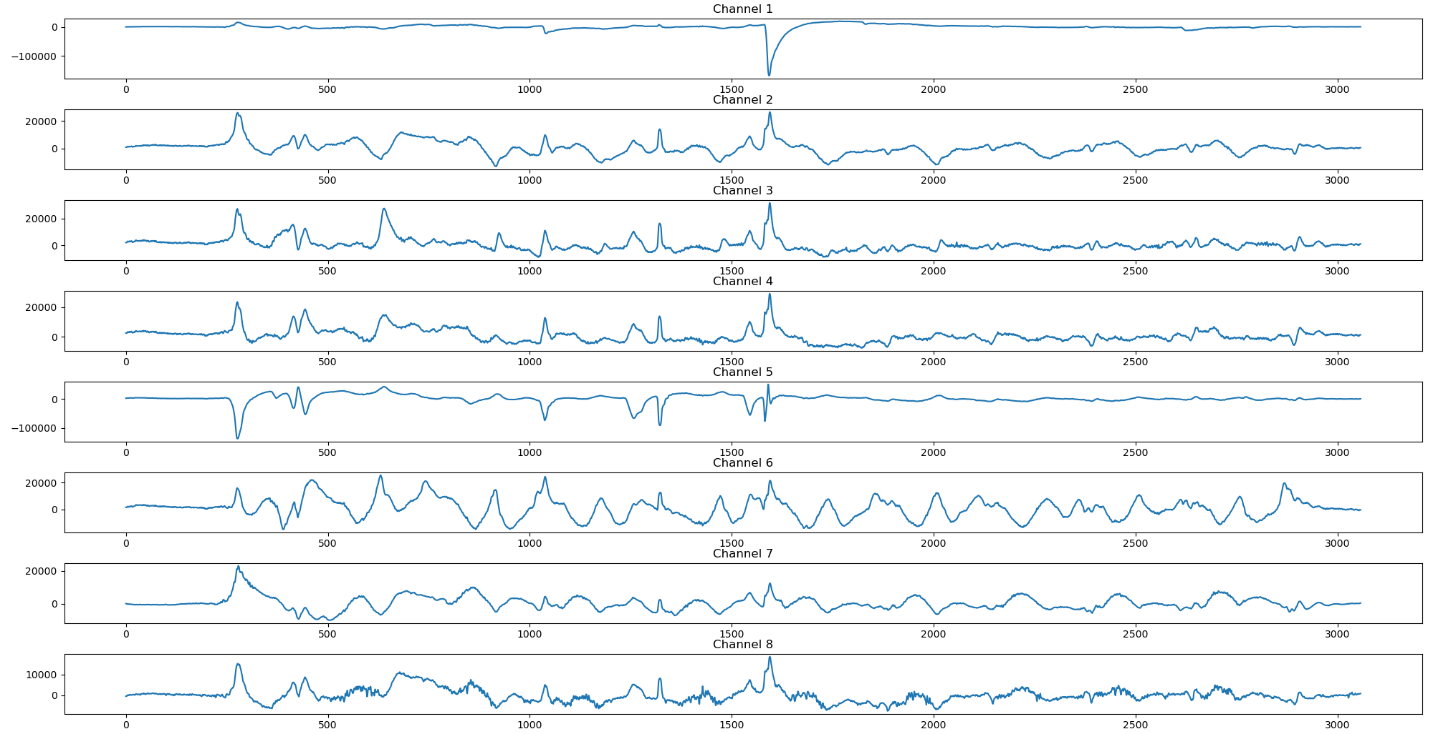
    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

## Week 4

This week I use the Mindrove app to record about 5 seconds of eeg data and display it using pandas and matplotlib



Data from Mindrove app

A graph of blue lines

Description automatically generated

Data from my python script

As you can see the data does not look too similar. This could be because that in my python script I do not use any filters. While I can’t see the code for the app it does look like it has some kind of filter for less noise.

Note: the app output a csv file with a different shape then the script and the app also has a different csv format, uses ; to separate.

Code:

import argparse

import time

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

def main():

    fileName = "test2\_20\_29\_07.csv"

    df = pd.read\_csv(fileName)

    ####### plot the data #######

    # Create a single figure with subplots

    fig, axes = plt.subplots(8, 1, figsize=(20, 14))

    for col in range(0, 8):

        data = []

        for x in range(df.shape[0]):

            data.append(df.iloc[x, col])

        axes[col].plot(data)  # Use the respective subplot for each row

        axes[col].set\_title(f"Channel {col + 1}")

    plt.tight\_layout()  # Adjusts spacing between subplots

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    main()