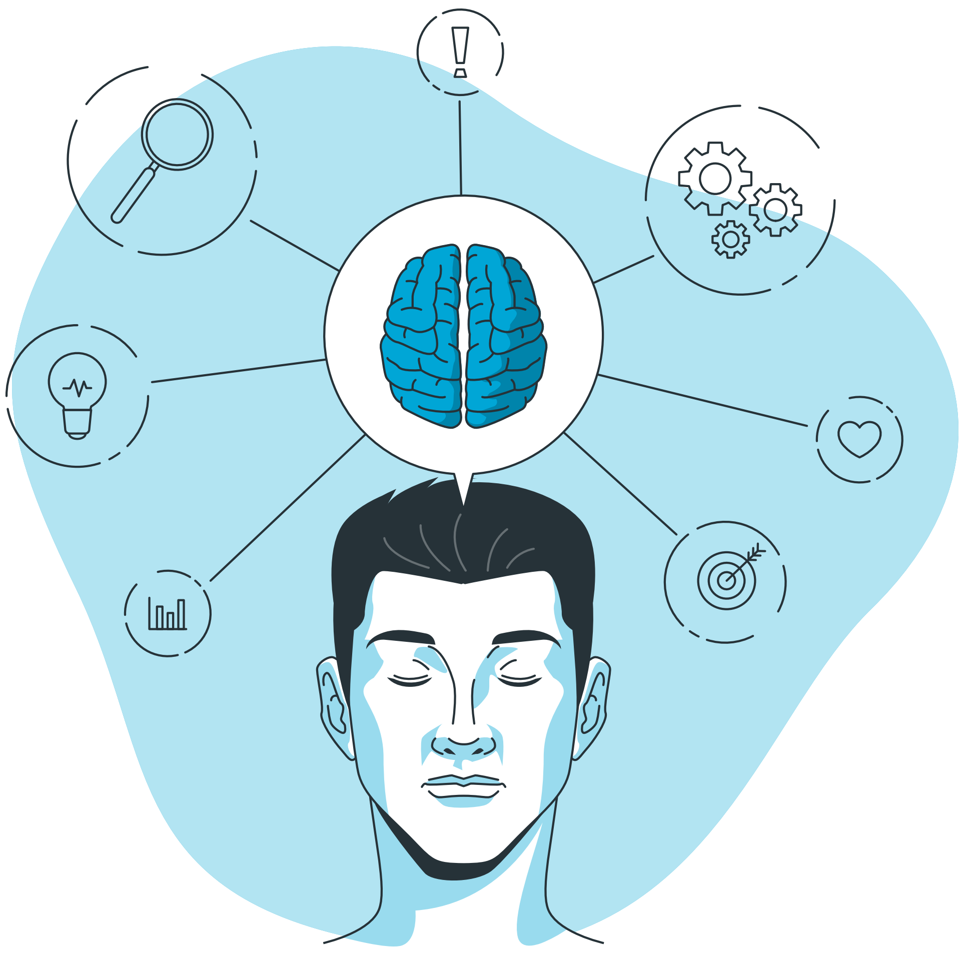
EEG-Based Brain Computer Interface

Measurement and data collection

Jack Chen
Martin Little



EEG-Based Brain Computer Interface

Measurement and data collection

by

Jack Chen
Martin Little

to obtain the degree of Bachelor of Science in Electrical Engineering
at the Delft University of Technology.

Students:	Jack Chen (5544513)
	Martin Little (5242258)
Supervisor:	Dr. B. Hunyadi
Thesis Committee:	Dr. R.A.C.M.M. van Swaaij
	Dr. G. Joseph

Cover:	'Mind map Cartoon Illustrations' by Storyset, free for personal and commercial purpose with attribution.
Style:	TU Delft Report-Thesis Template by Daan Zwaneveld (with additional modifications by Pragun Srivastava).

Abstract

This thesis investigates whether an EEG headset can be used to distinguish motor imagery signals in real time for a Brain Computer Interface (BCI).The specific EEG headset used for this project is the *gtec Unicorn Hybrid Black*. The aim of this subgroup is to stream the data in real time, preprocess the data, and create a training dataset using recordings from subjects. The sample from the EEG headset is streamed to a laptop using a live data streaming framework called Labstreaminglayer. The data is then filtered using frequency filters and ICA to remove noise and artifacts. Finally, ERDS plots are used to check the signal quality of the recordings. Recordings of sufficient quality should have (de)synchronisation peaks after the prompt is displayed. Before the recordings are made, an experimental setup is set up. This includes prompts with four different movements that the subject is asked to imagine. The data is sent to a GUI to be visualized with various graphs. It is also sent to a machine learning model to classify the movements. It was concluded that the subsystem could successfully stream data from the cap, process the data, and verify the quality of the data. Using ERDS plots, it was verified that some, but not all, MI actions are distinguishable for one individual. However, further verification of other individuals is required to conclude whether this is a systematic problem or is due to the individual.

Preface

This is the bachelor's thesis, 'EEG-Based Brain Computer Interface: Measurement and Data Collection'. It is written to obtain the bachelor's degree in Electrical Engineering at the Delft University of Technology. The main objective of this project was to develop a brain-computer interface (BCI) that uses brain waves to distinguish motor imagery signals from test subjects. These signals can be converted into a digital output to control, for example, a computer cursor. There were six people working on the project, divided into three subgroups. We were part of the measurement and data collection group.

We thought it was a fascinating project that could really make a difference in the future. It is pretty incredible to be able to control a computer with your mind, and this could be the breakthrough technology that makes it possible. We hope that this technology can be researched on a larger scale and potentially brought to the general public to help people who are physically disabled and/or paralyzed.

We would like to thank our supervisor, Dr. Bori Hunyadi, for guiding us through the BAP. We would also like to thank Prof. Dr. Leon Abelmann for his help during the project. Finally, we would like to thank our team members, Tijn Bakkum, Davi Spiller Beltrao, Victor van der Doorn, and Pragun Srivastava, for their hard work over the last two months.

Jack Chen
Martin Little
Delft, June 2024

Contents

1 Introduction

1.1. Brain computer interfaces

A brain-computer interface (BCI) enables communication between the brain and a computer using paradigms. This is done by collecting brain signals and translating them into specific digital commands for computers or other devices. This allows people to control their computers with their brain alone and without any movement [1]. This would be revolutionary for paralyzed people who are unable to control a computer. In addition to medical applications, BCIs have potential in the robotics and entertainment industries.

One of the most commonly used methods for measuring brain signals is electroencephalography (EEG). Compared to other methods, it is easy to use and inexpensive [2]. Today, several companies have brought user-friendly EEG headsets onto the market. Most are simple, non-invasive headsets that are easy to wear and inexpensive. This is in contrast to the invasive methods that involve implanting a chip in the brain. The EEG headset used in this project is the *Unicorn Hybrid Black* from *g.tec* [3], which is able to take non-invasive measurements by attaching electrodes to the user's scalp.

Different types of signals or paradigms can be used in BCIs. Some examples are SSVEP (Steady-State Visually Evoked Potentials) and P300. These paradigms use external stimuli to evoke a response from the brain that can be recorded by the BCI. While these systems are more robust to noise and easier to train, they require a constant stimulus, which can reduce performance over time as the user gets used to the stimulus [1].

Another paradigm is motor imagery, and this paradigm will be used for this project. This method does not require a constant stimulus and requires mental activity from the user, which makes it more flexible. Motor imagery involves imagining the movement without actually performing the intended movement. Motor imagery can be a good method of controlling external devices, as MI brain activity has been found to be very similar to real movement [4]. It is also a more natural process than the other paradigms, which can make it easier for users to use [2].

1.2. Problem definition

However, EEG has some limitations. The EEG headset is very sensitive to noise from external sources. Head movements, eye blinks, and heartbeats can cause artifacts in the EEG signal. EEG signals are also susceptible to noise from electromagnetic interference from power lines and electronic devices [1]. Signal processing and machine learning techniques can be used to improve signal quality, but artifacts cannot be completely removed in most cases.

Most BCIs are still focused on specific applications, such as medical. There is also signal variability between users with motor imagery, so there is no universal system. It needs to be extensively trained and calibrated based on the user's brain waves, as different people have unique brain patterns. Training on the user's data ensures that the system can accurately detect the intended brain signals. In addition, some people are not even able to produce the desired brain waves from motor imagery experiments [5]. Also, the mood and physical state of the user can affect the EEG signal [6]. Therefore, it is crucial to collect a training data set per subject to obtain more reliable motor imagery data.

1.3. Goal definition

The aim of the project is to develop an EEG-based Brain Computer Interface (BCI) that can effectively distinguish motor imagery signals in real time. To achieve this goal, the project has been divided into three sub-groups. This thesis will focus on the measurement and training data collection part of the project, which is responsible for streaming live EEG signals, collecting training data from subjects, and pre-processing the signals to be sent to the other subgroups. The decoding subgroup will use the signals to train a machine learning model and use this model to classify motor imagery signals. Finally, the interface subgroup is responsible for developing a user interface to visualize the data from the measurement group with various plots. The end product will be an interface that can distinguish MI signals in real time. This functionality can be verified using demos such as a game or moving a computer cursor. There should be several measures to determine if the signal quality is sufficient. Figure 1.1 shows a complete overview of the system with the connections between the three sub-groups.

1.4. Structure of thesis

The structure of this thesis is as follows: In chapter 2, the requirements of the project will be defined, including those for both the complete system and subsystem. All of the choices will be justified and can be traced back to the program of requirements. In chapter 3, the measurement setup will be described, which includes the EEG headset and live data streaming. In chapter 4, the implementation plan will be described, outlining the methodology for data recording and integration with other subgroups. In chapter 5, the preprocessing of the streamed data will be discussed. In chapter 6, the recordings will be analyzed using plots and other measures. Finally, in chapter 7, some concluding remarks will be provided, along with suggestions for future work.

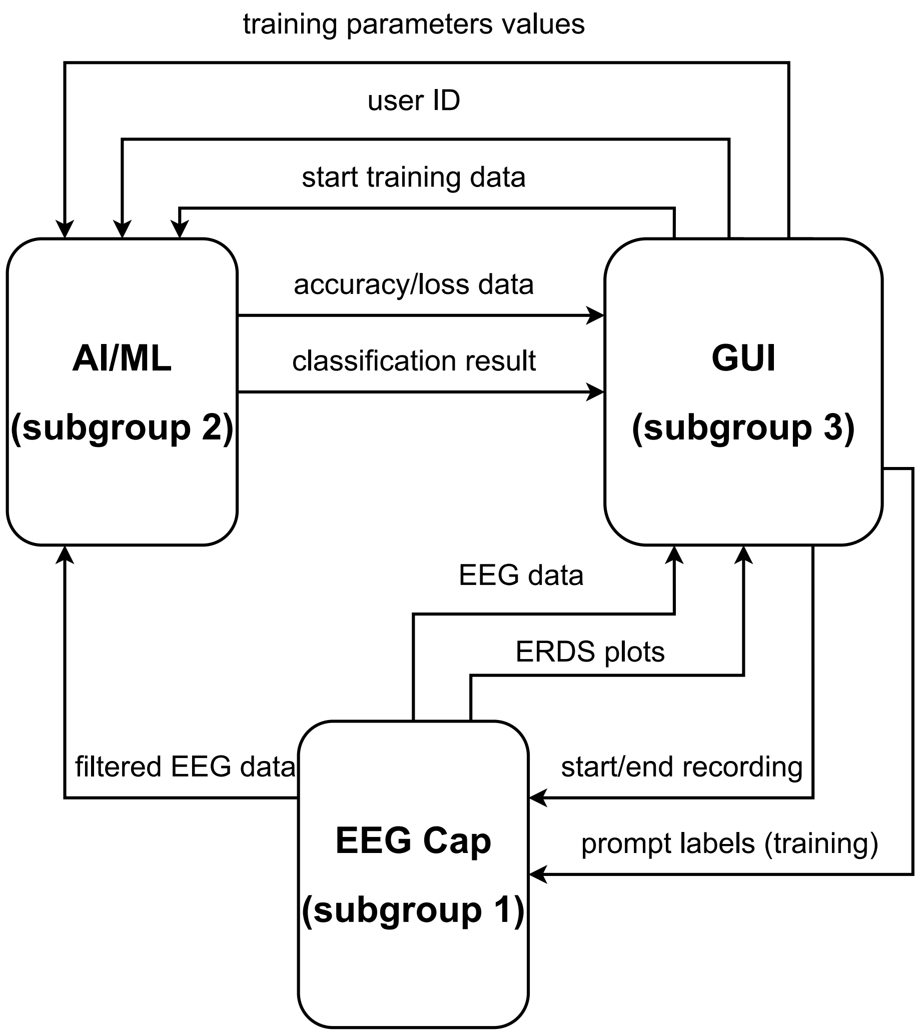


Figure 1.1: Overview of the complete system

2 Program of requirements

2.1. Top level goal and requirements

The system should demonstrate the possibility of distinguishing motor imagery (MI) signals using the g.tec Unicorn Hybrid Black.

Put more verbosely the top level goal is to make a system that demonstrates the possibility of distinguishing MI signals using the g.tec Unicorn Hybrid Black. The demo that was formulated for this purpose involves moving a cursor in 4 cardinal directions with each direction corresponding to a different MI signal. This specific demo was selected due to it's balance between feasibility and clear indication of how this technology could be used in the real world. From this goal a series of requirements was formulated. These were later refined and amended through conversations with the client.

Functional requirements

[A.1] The system must be able to decode MI-EEG signals to determine the intended movement, in accordance with requirements B1 – B3.

[A.2] The system must be able to visualize the decoded MI-EEG signals through interactive demonstration(s).

[A.3] The system must be able to plot EEG data in real-time (note: we define a real-time plot as one which satisfies requirement B4).

[A.4] The data handling performed by the system must be individualized to each user.

Performance requirements

[B.1] The minimal sensitivity of prediction for each of the 4 motor imagery actions separately must exceed 50%.

[B.2] The average overall accuracy of prediction across all 4 motor imagery actions must exceed 70%.

[B.3] The action latency (see definition below) must be less than 5 seconds.

[B.4] The plot latency (see definition below) must be less than 1 second.

Implementation requirements

[C.1] The data utilized by the system must be measured using the g.tec Unicorn Hybrid Black.

[C.2] All software developed for the system must be written in Python. This includes software for measurement, decoding, and visualization.

Definitions

- We define the *action latency* as the time between the moment at which the prompt is shown on the GUI, and the moment at which the corresponding classification is shown.
- We define the *plot latency* as the time between the measurement of a sample and its corresponding output to the plot.

2.2. Subgroup Requirements

2.2.1. Functional requirements

- The subsystem must be able to stream and filter the data from the EEG headset in real-time. This means that the sample must be collected within 1 second by the interface. Preferably in a Python environment
- The subsystem must be able to record the data from the EEG headset and save the data into a csv file.

- The output signals of the subsystem must have a signal-to-noise ratio of more than 1.25 for all of the four movement classes.
- The electrodes must be in such way secured that it fits tightly on the scalp
- The filtered data must be sent to the other subgroups with the use of a sliding window.

2.2.2. Performance requirements

- The subsystem should contribute no more than 1 second of overhead to the action latency
- The subsystem should contribute no more than 0.25 seconds to the plot latency

The performance requirement for the action latency is specified as overhead, since recording samples for a window adds latency equal to the length of the window. This contribution to the latency is not counted as part of the overhead.

The delay of the subsystem contributing to the action latency represents the time interval between a user of the system performing MI and the filtered sample being received by the decoding group.

The delay of the subsystem contributing to the plot latency represents the time interval between a measurement of the cap being made and the corresponding raw sample being received by the interface group.

2.2.3. System requirements

- The subsystem must measure EEG signals with an EEG headset with dry electrodes
- The subsystem must have instructions to set up the EEG cap measurements and real-time streaming.
- The subsystem must measure signals with the exact same procedure for different test subjects.
- The subsystem must perform at least 24 motor imagery trials from each of the four movements per test subject. The four movements are left hand, right hand, tongue and feet. The collected data will be used for the training data set.

2.2.4. Input-output requirements

- The output data for the decoding group must be a labelled array of the selected window size with a counter element. The counter element is extracted from the internal counter of the EEG headset.
- The output data for the interface group must be a filtered sample of each of the eight EEG channels for real-time plotting.
- The output data for the interface group must be an array of size 500 to which the FFT is applied for the FFT and band power plots.

3 Measurement setup

This chapter describes the measurement setup for the project. First, some background information about motor imagery will be provided. Also, it will be described how an EEG headset is utilized to stream the data in real-time. Lastly, a set of experiments with motor imagery trials is planned to assemble training data for the decoding subgroup.

3.1. Motor imagery

Motor imagery is defined as the representation of a movement without actually performing the physical movement. It is also considered to be conscious access to the unconscious process of motor movement preparation [7]. It is also found that motor imagery is associated with learning and controlling movement [8]. During motor imagery, certain parts of the brain show activity that is similar when the real movement is performed [4]. These patterns can be interpreted by a BCI using a technique called electroencephalography (EEG). This technique is explained later in subsection 3.2.1. The signals are then translated into a digital output using a classification model and can be used for demonstrations such as games.

It is recommended that subjects focus on the imagined movement in order to obtain consistent results, as EEG recordings are susceptible to noise and artifacts [2]. Providing subjects with a visual or auditory cue can help with visualization of the movement [4]. A visual cue could be text or an arrow indicating the movement to be imagined. Examples of movements commonly used in motor imagery trials are moving the hands, feet, and tongue. Recording multiple trials allows the subject to practice motor imagery, which may lead to improved results [9].

3.1.1. Brain patterns

It is important to know what brain patterns to expect during motor imagery tasks before using the EEG headset. The brain regions that are most active during these tasks and the headset channels that are most active during these tasks should be identified. It is found that the motor cortex region is most active during MI tasks [10]. This area of the brain plays an important role in the planning and execution of movements and would therefore be expected to be active during motor imagery tasks. Figure 3.1 shows the location of the brain areas. The motor cortex is located in the frontal lobe of the brain and is divided into three parts. For the purposes of this project, it is not important what each of the sub-areas does, and, for simplicity, each area is assumed to play the same role.

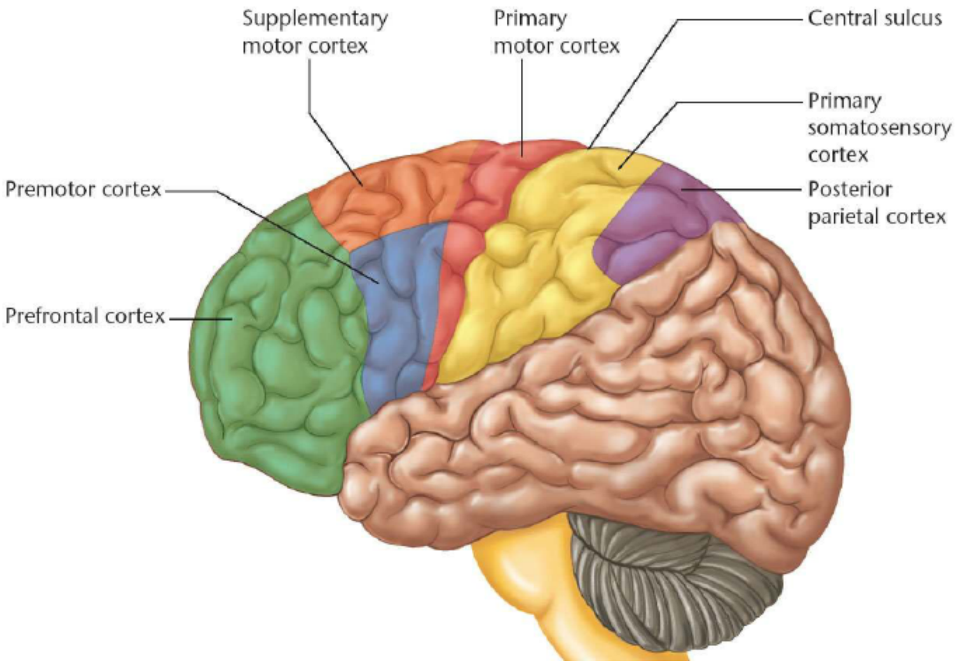

Supplementary motor cortex
Primary motor cortex
Central sulcus
Primary somatosensory cortex
Posterior parietal cortex
Premotor cortex
Prefrontal cortex

Figure 3.1: Cortical areas of the human brain [11]

A meta-analysis visualized the brain regions active during motor imagery and execution tasks [11]. And it can be seen in Figure 3.2 that similar brain regions are active during tasks. When compared to Figure 3.1, it predominately corresponds with the motor cortex and parietal brain areas that are active during those tasks. This activity in the motor cortex is as expected, whereas the activity in the parietal area deviates from expectations. However, research shows that this area also contributes to motor programming [12].

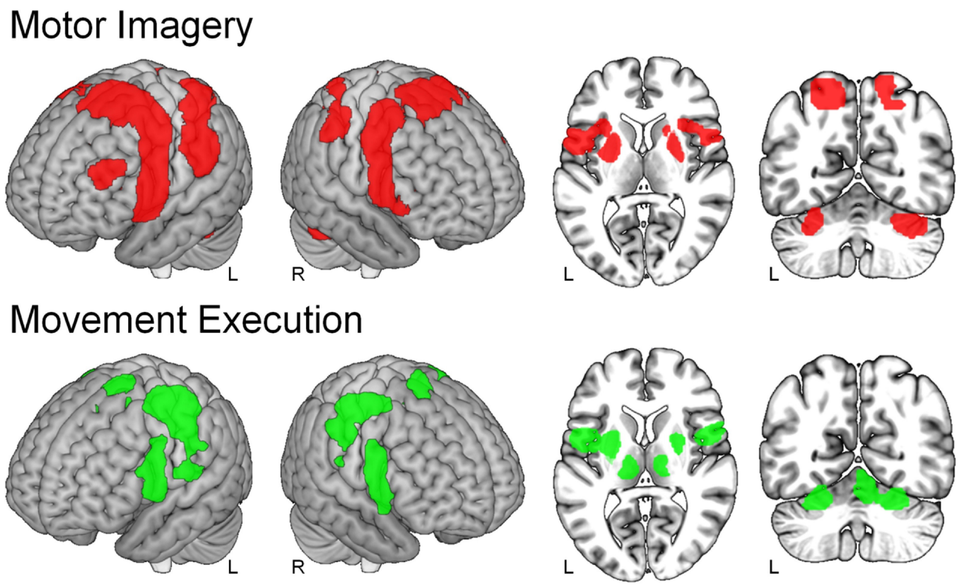


Figure 3.2: Meta analysis of motor imagery and execution. The active brain areas during these tasks are visualized. Modified image from [11]

3.2. EEG headset

3.2.1. Electroencephalography (EEG)

A common technique for measuring brain waves is to use electroencephalogram (EEG) signals. When a person is performing motor imagery tasks, then neurons in the motor cortex will become active and will send electrical currents. By placing electrodes on a person's scalp, it is possible to accurately measure the potential difference caused by the current flow [13].

EEG signals are made out of oscillations and can be divided into specific frequency bands, which are the Delta, Theta, Mu, Alpha, Beta, and Gamma bands. The frequency division of the bands can be seen in Table 3.1. Some actions and conditions can lead to an increase in activity in a certain band. For example, closing the eyes evokes a peak in the alpha band [14].

The mu and alpha rhythms reside in a similar frequency band. However, the rhythms are associated with different behaviors [15]. The mu band is associated with movement execution, planning, and observation, while the alpha band is associated with relaxation and calmness. Also, mu waves are mostly found over the motor cortex, and alpha waves are found around the back of the head.

Table 3.1: EEG Frequency Bands [16]

Band	**Frequency [Hz]**
Delta	< 4
Theta	4 to 8
Mu	8 to 12
Alpha	8 to 13
Beta	13 to 30
Gamma	30 to 100

3.2.2. EEG patterns

One pattern that is widely used in EEG research is ERDS. ERDS stands for event-related desynchronization/synchronization. In the event of a stimulus, the neurons can (de)synchronize with each other, and this results in an increase/decrease in the EEG amplitude. This phenomenon can be plotted, and these plots display changes in the spectral power of certain frequency bands of EEG signals [17]. A more in-depth description of ERDS plots will be described in section 6.2.

For motor imagery, the frequency bands of mu and beta are the most important to look into[4], so between the frequencies of 8 Hz to 12 Hz and 13 Hz to 30 Hz. The mu rhythms can be observed when measuring over the motor cortex. It causes a (de)synchronization in the mu rhythm when a person wants to imagine or perform a movement [18]. Imagining hand movements causes a desynchronization, while imagining foot and tongue movements causes a synchronization in most cases. The beta rhythms originate from the same location as the mu rhythms and also have (de)synchronization when a movement is imagined. However, it is found that these two rhythms have a different behavior during (de)synchronization [4]. Both of the bands are desynchronized when preparing the movement, but the beta rhythms show synchronization after movement execution, while the mu rhythms stay desynchronized. After that, both bands will recover to their baseline values. Figure 3.3 shows a visual representation of the mu and beta rhythm (de)synchronization. However, it should be noted that ERDS behavior may vary from person to person. Some people may have weaker responses or even show no (de)synchronization when imagining movements.

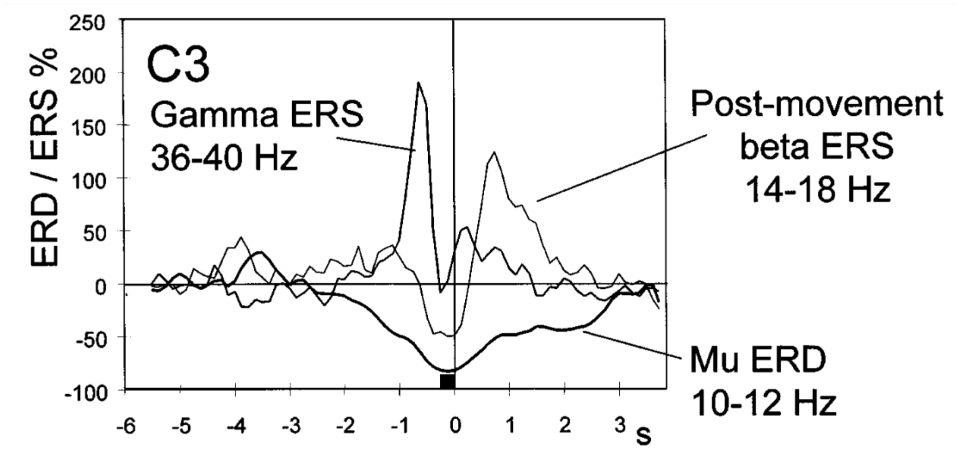


Figure 3.3: Example of an ERDS plot for three different frequency bands. The axes represent relative band power over time. The time is relative to the movement execution[4].

3.2.3. Headset

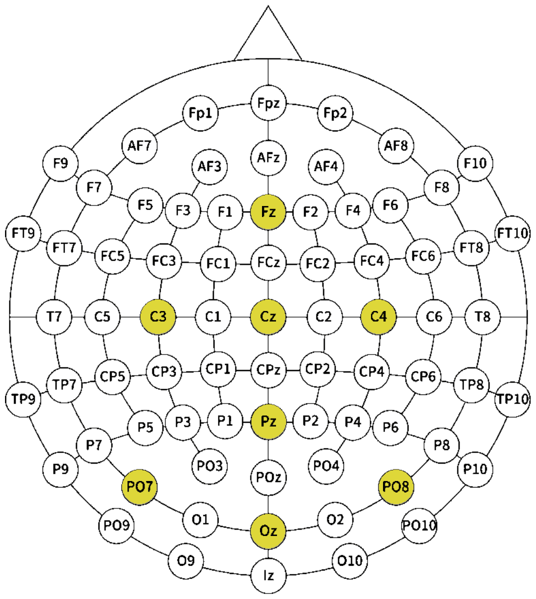
The EEG headset used for this project is the *gtec Unicorn Hybrid Black*, which is capable of making both dry and wet measurements using eight electrodes. The sampling frequency of the headset is 250 Hz, which is sufficient for the real-time measurement of test subjects. The maximum frequency that is relevant for motor imagery is 30 Hz, which is the maximum frequency of the beta band. The headset satisfies the Nyquist criteria because it samples a rate higher than 60 Hz. The headset can be connected via Bluetooth to facilitate the collection of EEG recordings.

Electrodes

The headset is capable of measuring EEG with dry and wet electrodes. Using gel for measurements could improve the signal quality by reducing the electrode-skin impedance. However, dry electrodes can be used for research with similar accuracy to wet electrodes [19]. A dry measurement also has a shorter set-up time as a result of the lack of gel application. Therefore, dry electrodes were used during the project for convenience.

Figure 3.4a and 3.4b illustrate the headset itself and its electrode positions on the scalp. The electrodes are positioned in eight different positions on the head according to the 10-20 system, which is an international system for electrode placement [20]. The electrode placements for this headset are FZ, C3, CZ, C4, PZ, PO7, OZ, and PO8. The electrode slots are attached to the mesh cap, and they cannot be moved to another position. When examining the active brain regions during motor imagery, it can be stated that the strongest response can be expected from the Fz, C3, CZ, and C4 channels during motor imagery trials, as these channels are positioned at the motor cortex areas. It is important to note that the positioning of the electrodes may vary depending on the individual and the setup. Therefore, it is essential to provide consistent instructions for electrode placement to ensure that the same positions are used each time.

(a) Unicorn Hybrid Black EEG headset [3]



(b) Electrode positions of the headset. The triangle at the top indicates the nose. Modified image from [20]

Figure 3.4: Figures of EEG headset and electrode positions

3.3. Data live streaming

3.3.1. Lab streaming layer

The Lab Streaming Layer (LSL) is used for live data streaming. This is an open source software framework that can collect streaming data from hardware devices such as EEG headsets like the *Unicorn Hybrid Black*. The software is quick and easy to set up, and it is compatible with multiple programming languages for smooth integration. It automatically collects data streams from the local area network (LAN) and ensures that the streams are synchronized in time. However, this method has some limitations. There is a delay in the communication between the receiving device and the LSL software. This delay is different when switching devices, as it depends on the CPU performance of the device. Tests have shown that latency can increase to about 20 ms. However, this latency can be compensated by software before starting to record. [21]

3.3.2. Headset integration

The original plan was to connect the headset directly via Bluetooth, but this did not allow connections to be made without the use of specific software, like the LSL interface. The LSL interface is fully integrated into the software of the EEG headset in use and is able to collect raw data from the headset at a sampling frequency of 250 Hz. It then sends the data to the LSL, where it can be collected using a Python environment. Figure 3.5 shows an overview of the data live streaming.

The headset outputs a sample of 17 elements, including raw data from the eight EEG channels, the three dimensions of the gyroscope and accelerometer, an internal counter, a validation indicator, and a battery level. For this project, the output is adjusted to remove the gyroscope, accelerometer, and battery level. What remains for the Python environment is a sample of 11 elements in total.

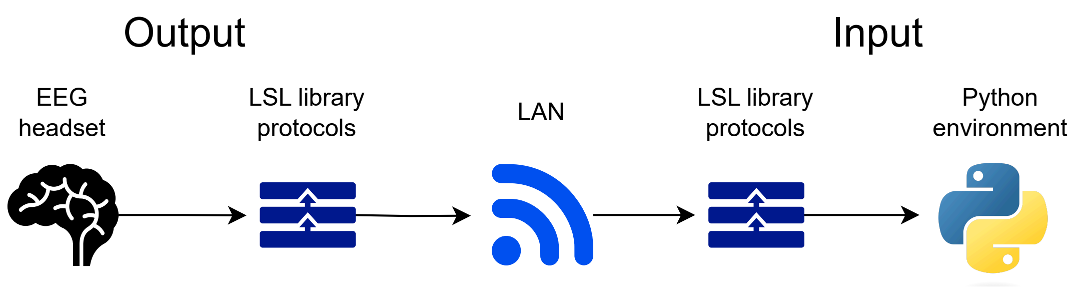


Figure 3.5: Overview of LSL data streaming

3.4. Experiment setup

3.4.1. Objective

The objective of this experiment is to construct a training data set for the decoding subgroup. Initially, the machine learning model is trained on an online dataset [22]. However, the dataset utilizes a different type of EEG headset, which may exhibit different patterns from the headset used in this project. To address this, a training data set derived from test subjects is assembled. By adding personalized data, it will let the model train on the MI patterns of each test subject. This approach is expected to enhance the accuracy and precision of the predictions.

3.4.2. Prompts

Several subjects will be tested with the EEG headset using the setup of the experiment. The setup consists of a specific prompt that will appear on a computer display. When the subject sees the prompt, then the subject is asked to imagine the movement. There are four movements to imagine, corresponding to four directions: left hand (left), right hand (right), tongue (up), and feet (down). Each movement is shown six times in a randomized order during the recording, and thus a full recording consists of 24 prompts. Each prompt starts with a fixation cross for six seconds to stabilize the EEG signal, and then the movement appears on the screen for six seconds. A timing scheme for the experiment can be seen in Figure 3.6. A complete recording therefore lasts 4.8 minutes. The experiment is repeated at least four times for each subject for a total of 96 trials, or 24 for each motor imagery action, meeting the requirement. The subject is allowed to rest between recordings. However, the subject is not allowed to remove the cap or change location to avoid unnecessary signal interference. Pictures of the prompts can be found in Appendix A.

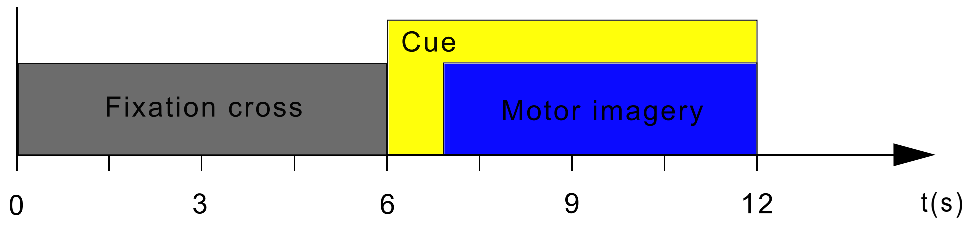


Figure 3.6: Timing scheme of the experiment

3.4.3. Instructions

To minimize further noise and artifacts, the subject is instructed to sit as still as possible and in a relaxed position. The subject should place their feet on the floor and remain in place during the recording to avoid grounding artifacts. The subject is also advised to avoid extensive head and eye movements, which cause artifacts in the EEG measurements. Finally, blinking should be avoided as much as possible, but unfortunately, this is unavoidable. This artifact should be filtered out in the preprocessing.

The four movements are the same movements that the ML subgroup uses for its training data. This makes the integration smoother. To help the subjects imagine the four movements, there were instructions on how to imagine the four movements. For the left hand, it was suggested to imagine touching each of the fingers with the thumb. The same was done for the right hand. For the tongue, it was suggested to imagine the tongue moving up and down, from the palate to the floor of the mouth. Finally, for the feet, the subject was asked to imagine moving each of the toes up and down.

Before the experiment begins, the EEG headset should be set up. The subject can place the headset on the scalp and secure it with the neck strap. The electrodes should be positioned so that they are in direct contact with the scalp and in the exact same position as described in subsection 3.2.3. The electrodes can be twisted to pass through hair and improve signal quality. Then sticky electrodes are placed behind the ear's mastoid bones as reference and ground signals. Next, the stability of the setup is tested using the *Unicorn Suite Hybrid Black Recorder* shown in Figure B.1. This software can be used to visualize the data from the EEG headset. It is necessary to wait until the channels have reached a stable state. This is determined by looking at the signal quality of the software. The channels in the scope will turn green when the signal quality is reliable. Figure 3.7 shows what the quality scope looks like in the software. When all eight channels are stable, the actual recording can start by running the LSL interface.

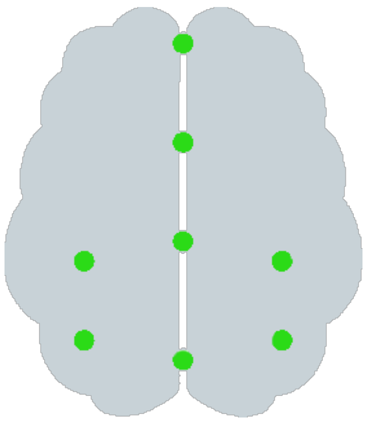


Figure 3.7: Quality scope when all channels are reliable.

4 Data collection and integration

This chapter discusses the implementation plan. It describes how all the separate techniques will be combined into a subsystem. This chapter also describes the connections to the other subsystems. Finally, it shows how to validate the prototype, i.e., to determine when the recordings are of good enough quality.

4.1. Data collection

To collect the EEG data, a Python script was written to record 72005 samples. This is equivalent to 288 seconds of recording time, with five samples as a buffer. It was integrated with the GUI to display prompts for the MI actions at predetermined times, allowing for proper analysis by labeling the data with the corresponding movements. The left hand is labeled as 1, the right hand as 2, the tongue as 3, and the feet as 4. The samples collected during the stabilizing cross prompt are labeled as 0. This data is then stored in a CSV file for future processing.

4.2. Connections

As discussed earlier in subsection 3.3.1, the LSL is used to stream data from the cap to the computer. Specifically, the EEG cap comes with a software suite containing several programs seen in **??**. One of these programs simply connects to the cap via Bluetooth and sends the data to the LSL, as shown in Appendix B. The data is then received from the LSL in several Python scripts using the PyLSL library. Specifically, it is used in the GUI to create the live plots of the signals and the data collection script described in section 4.1. As the decoding is done by Pytorch, the data collected as CSV files had to be converted to Pytorch tensors and saved as Pytorch files. A very simple script was used to take the recording from the CSV files, filter it, convert it to the format required by the machine learning network, and save it as a PyTorch file. The filtering steps taken were removal of the DC component, bandpass filtering, and application of a common average reference filter.

Additionally, a file that connects the live streamed data from the cap to the decoding machine learning model is created. The plan is for the GUI to start the file as a subprocess and communicate with it using the built-in pipe system. The file will first establish the LSL connection. It will then be told by the GUI which of the personalized machine learning models to use. It will then load that model. When a demo is started, a synchronization step is performed to avoid a build-up in the LSL buffer. Then a loop collects data from the LSL in a window, as described in subsection 5.2.3. As the window fills, it goes through the filtering steps described in section 5.2. It is then converted into a Pytorch tensor and passed to the machine learning model that has been loaded. The classification result from the model is then output to the GUI. The GUI then uses the classification result to perform the appropriate action for the demo. In addition, the filtered window is used to calculate the FFT and power bands of the EEG signal. This data is then sent to the interface group, which visualizes the data into plots.

4.3. Analysis

In order to analyze and verify the validity of the measurements, it was necessary to create some ERDS plots. To accomplish this, a simple Python script was created to generate these ERDS plots. It was created by taking the code from [23], which was then modified to take the data from the csv files and display all 8 channels instead of just three. The plots will also be integrated into the user interface to verify whether the signal quality is sufficient. The plots created by this code and their significance can be found in chapter 6.

5 Signal Preprocessing

This chapter focuses on the preprocessing of the streamed signals. The signals from the LSL are collected in a Python environment and then preprocessed to remove noise and artifacts using several techniques.

5.1. Fast Fourier Transform (FFT)

A common technique that is used for EEG signal analysis is the Fast Fourier Transform (FFT), and it will be briefly explained what it is and how it can be used in EEG signals. The FFT is a more efficient and faster method to calculate the Discrete Fourier transform (DFT). The DFT is the discrete version of the Fourier transform, which is able to transform original signals into the frequency domain. The difference is that it uses discrete-time samples instead of a continuous-time signal. It can be given by this equation:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{kn}{N}} \tag{5.1}$$

Where $x[n]$ represents a time signal of length N. The index k ranges from 0 to $N-1$. The discrete series $X[k]$ represents the DFT of the input signal, where the value at a given k is the k-th frequency component of the transformed sequence. The sum is taken over all the samples n which range from 0 to $N-1$.

As already described in subsection 3.2.1, EEG signals have rhythms that appear in different frequency bands. With the FFT, it can be visualized if a certain frequency band has higher brain activity. Also, when a specific frequency band should be suppressed in the case of, for example, noise, it can be attenuated in that spectrum. The signal can then be transformed back to the time domain using the inverse Fourier transform, thereby eliminating the noise [24].

5.2. Filtering

The original signal from the LSL is affected by noise and artifacts. This is mostly from eye blinks, head movement, and electromagnetic interference. This is solved by applying some signal processing techniques.

5.2.1. DC-component removal

The signal has a large linear DC component, which can be seen in the top subplot of Figure 5.1. This component can be removed by taking the mean of the signal and subtracting it from the data. However, this would not be ideal in this case because simply subtracting the mean of the signal does not account for any linear trends present in the data. Linear detrending takes into account the changes in the data over time, and this is the method used. A function from the Scipy library is used to implement the detrending [25]. This function performs a linear least squares fit to the data, finding the best-fit linear trend to represent the signal. In addition, detrending is performed per prompt, allowing linear trends to be removed segment by segment. This approach effectively captures and removes linear trends segment by segment, providing a more accurate detrending method when linear trends are present during certain prompts.

The linear least squares fit can be mathematically represented as the following: It is first assumed that there is a linear trend in the form of $y = ax + b$. It then creates a matrix A that is of size $N \times 2$, where N represents the number of data points in a segment. The first column of A is filled with values from 1 to N, which is then divided by N and the second column is filled with only ones. The least squares fit is then performed on A and the data segment points, which will calculate the optimal a and b values in the linear trend.

The least squares fit is calculated using Equation 5.2. Where A represents the input matrix and B the original data points of the segment. It will compute a value of x, and that approximately solves the Equation 5.2.

$$A \times x = B \tag{5.2}$$

The values from the fitted trend can be subtracted from the original signal to obtain the detrended data points using Equation 5.3. Where y_i' represents the detrended data point, y_i the original data point, and $A \times x$ the calculated fitted trend data point. Figure 5.1 shows an example of an EEG channel from the EEG headset before and after DC-component removal.

$$y_i' = y_i - A \times x \tag{5.3}$$

5.2.2. Frequency filtering

Furthermore, it can be seen that low- and high-frequency noise is affecting the signal. A cutoff frequency range that was considered was between 0.5 Hz and 30 Hz. This was the frequency filtering range that was used in the training dataset of the decoding subgroup [22]. Another range that was considered was between 8 Hz and 30 Hz. This range only leaves the frequencies that are relevant for motor imagery, as mentioned in subsection 3.2.2. This range could be useful for analyzing the EEG signals. However, the currently selected frequency range is between 0.5 Hz and 38 Hz [26], so frequencies outside this range are removed using a Butterworth bandpass filter with the specified cutoff frequencies. It was discussed with the decoding sub-group that these cut-off frequencies should be used. This would increase the performance of the classification model while maintaining enough data for training and testing the model. A Butterworth filter was chosen because it does not distort the signal within the desired frequency band. This filter was constructed using the filter design function for Butterworth filters from the SciPy library. An order of 4 was chosen to provide a sufficiently sharp cutoff while maintaining stability and simplicity.

The raw data also contains a 50-Hz noise pattern caused by power lines and other electromagnetic interference. So a notch filter at 50 Hz is applied to remove this noise. This is essentially a narrow-bandwidth band-stop filter, rejecting only the narrow frequency band. A fourth-order Butterworth filter from 49 Hz to 51 Hz is chosen to reject the noise. This filter is constructed in the same way as the bandpass filter. The frequency response of the filter can be found in Appendix C, where different filter orders are compared. Also, the plots of the poles in the z-plane can be found in Appendix D to determine the stability of the filter.

Figure 5.1 shows an example of data from the cap of one of the EEG channels after applying the aforementioned filters. As can be seen, most of the noise is filtered out, and the EEG signal is preserved. In the filtered signal, it can be seen that there are several spikes per second, which could be due to heartbeats. Noise from heartbeats is found at around 2.4 Hz [27] and is therefore not filtered out by the Butterworth filters.

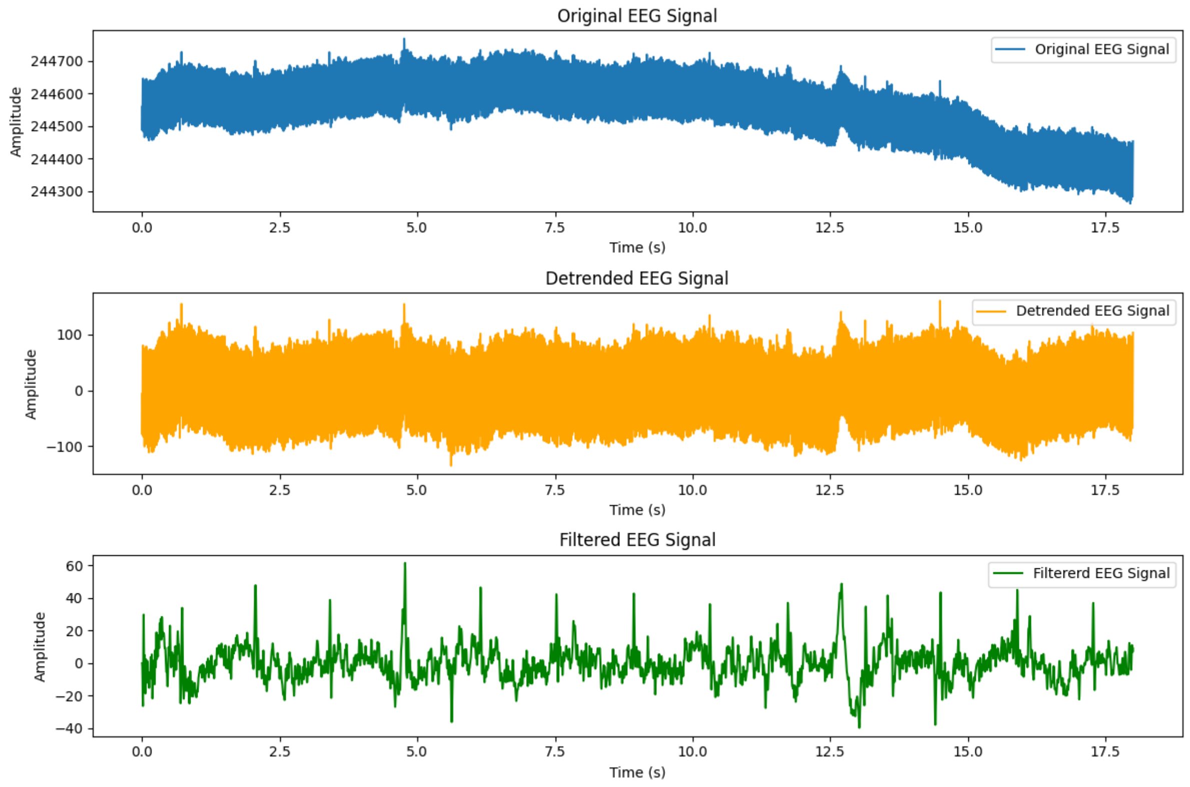


Figure 5.1: Comparison between original EEG signal from channel 1 (Fz), detrended and filtered signal. Filtered with a bandpass filter of 0.5 Hz to 38 Hz and a bandstop filter of 49 Hz to 51 Hz.

5.2.3. Live filtering

It is necessary to filter the signal in real time in order to classify and plot it correctly. It is computationally very intensive to filter large windows of data when real-time performance is needed. Therefore, the signal will be divided into smaller blocks of a certain fixed size. These are then processed one at a time and then combined into one complete processed output signal. Two methods that can be used for this are the overlap-save and overlap-add methods [28].

In the case of overlap-save, a block of length $L + M - 1$, where L is the segment that is processed and M is the length of the filter. The filter can then be applied, and after filtering, the first $M - 1$ samples are removed because they are corrupted by aliasing. These blocks are then concatenated to form the output signal. The method is visualized in Figure 5.2.

For the overlap-add method, the signal is divided into blocks of length L. Then it is zero-padded to a block of length $L + M - 1$. Where M is again the filter size. The block will be filtered, and the overlapping part will be added to get the output signal.

The overlap-save method is more efficient because it does not require processing the overlapped samples again. While the overlap-add method is a simpler implementation, it will have an efficiency loss due to the reprocessing of the same samples [28]. Thus, the overlap-save method is chosen for its efficiency.

Implementation

When the data from the EEG headset comes in, it will be filtered in real-time using the overlap-save method. A window of 500 samples is used to achieve a good balance between the number of samples used per filtering operation and the number of times the filtering operation needs to be performed to achieve good performance. Also, frequency and time resolution are taken into consideration, as a longer time window increases frequency resolution but reduces time resolution. An overlap of 75% is used to have a better time resolution and reduce the latency of plotting the FFT plot. Additionally, a Hann window is applied to reduce spectral leakage, as sidelobes may mask smaller peaks. Lastly, the signal is zero-padded to increase the apparent resolution of the FFT plot. The filtered blocks are concatenated

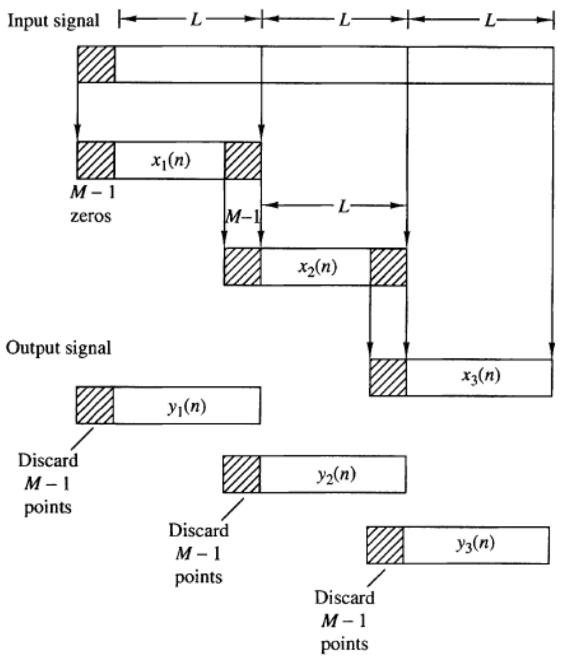


Figure 5.2: Visual representation of the overlap-save method filtering [28]

and sent to the other subgroups. In the case of the interface group, the filtered block is sent directly to visualize the data with plots. The plots include a real-time plot of the filtered data, an FFT plot, and a frequency band plot, each showing the power within a specific band. For decoding, the desired output data for the decoding group is a filtered array of size 529. This array is used for classification and for performing the demos. So the blocks are first concatenated to the desired size and then sent to the decoding subgroup.

5.3. ICA filtering

Artifacts such as heartbeats, eye blinks, and head movements cannot be completely removed by the filters mentioned in section 5.2. One common technique that could be helpful is Independent Component Analysis (ICA), which has been proven to be effective against eye movement [29]. ICA decomposes the signal into several independent components that correspond to the EEG signals but also to artifacts if they occur [30]. After applying ICA, each row represents the activity of one independent component, which has its own unique spatial and temporal properties. Then, the artifact components can be manually removed, and the remaining signals can be combined to obtain the artifact-free signal. It has been shown that it could improve the accuracy of a BCI classification model and decrease the computation time due to the reduction of channels [31].

ICA can be mathematically formulated as this model:

$$\mathbf{X} = \mathbf{A} \cdot \mathbf{S} \tag{5.4}$$

Where $\mathbf{X}$ is the observation matrix of size $N \times M$. Where N is the number of channels, which is in this case the eight EEG channels. The L represents the number of data samples in each channel. S is represented by the independent source matrix of size $M \times L$, which consists of the M number of independent components and L data samples for each component. $\mathbf{A}$ is the mixing matrix with unknown coefficients of size $N \times M$ [30]. N is the number of observed signals, which are the electrodes in this case, and M denotes the number of sources [32]. Each value in the A matrix stands for the transfer coefficient of an independent component to an EEG channel.

The purpose of $\mathbf{A}$ is that it mixes the independent components (ICs) back to observed samples. The goal of the ICA algorithm is to find the inverse of $\mathbf{A}$ called the demixing matrix, which is also denoted by $\mathbf{W}$. This way, S can be obtained from X. In other words, the independent components are extracted from the EEG signals when the demixing matrix is found. The algorithm that was selected to find the demixing matrix **W** is the Picard algorithm described in [33]. The reason for this choice is because, according to [34], ”Picard is a newer (2017) algorithm that is expected to converge faster than FastICA and Infomax, and is more robust than other algorithms in cases where the sources are not completely independent, which typically happens with real EEG/MEG data.” Through inspection and analysis of the independent components of S it is possible to identify components that correspond to certain artifacts. These artifact components are set to 0. The original signal can now be reconstructed from the independent components using Equation 5.5[30]. Since the artifacts are set to 0, this produces what the original signal would have been had the artifacts had 0 amplitude. This is equivalent to filtering out these components.

Initially, this process was conducted manually. However, an automated system that detects faulty channels was subsequently developed. The system excludes one of the components and examines the impact of these changes on each channel. The algorithm then examines the standard deviation difference and amplitude difference of each channel and removes the component from the signal if the threshold is exceeded. The algorithm sets the thresholds for the standard deviation at 9 and the amplitude difference at 200. The standard deviation difference is employed to identify the presence of long-lasting artifacts in the signal, while the amplitude difference is utilized to detect sharp peaks resulting from such artifacts. This process is repeated to exclude each component. The faulty components are then removed, and only the clean components remain.

$$x_i = a_{i1}s_1 + a_{i2}s_2 + \cdots + a_{iN}s_N = \sum_{j=1}^{N} a_{ij}s_j \tag{5.5}$$

The Python package MNE-python is used to apply ICA to the signals [35]. It is an open-source package used to visualize and analyze neurophysiological signals such as EEG and ECG. The package is also able to visualize the scalp field distribution of each independent component (IC), showing the active brain regions for the different ICs. Figure 5.3 shows an example of this. This is a way of identifying which ICs correspond to EEG signals and which are artifacts, such as blinks or heartbeats. It can be seen that the second and third ICs have a high intensity at the front of the brain, which could correspond to eye blinking. These channels should therefore be removed.

Figure 5.4 shows a recording of EEG signals before and after ICA component removal. It is clear that motion artifacts are removed, as the spikes at the beginning are less prominent. Especially in the first channel, motion artifacts are significantly reduced.

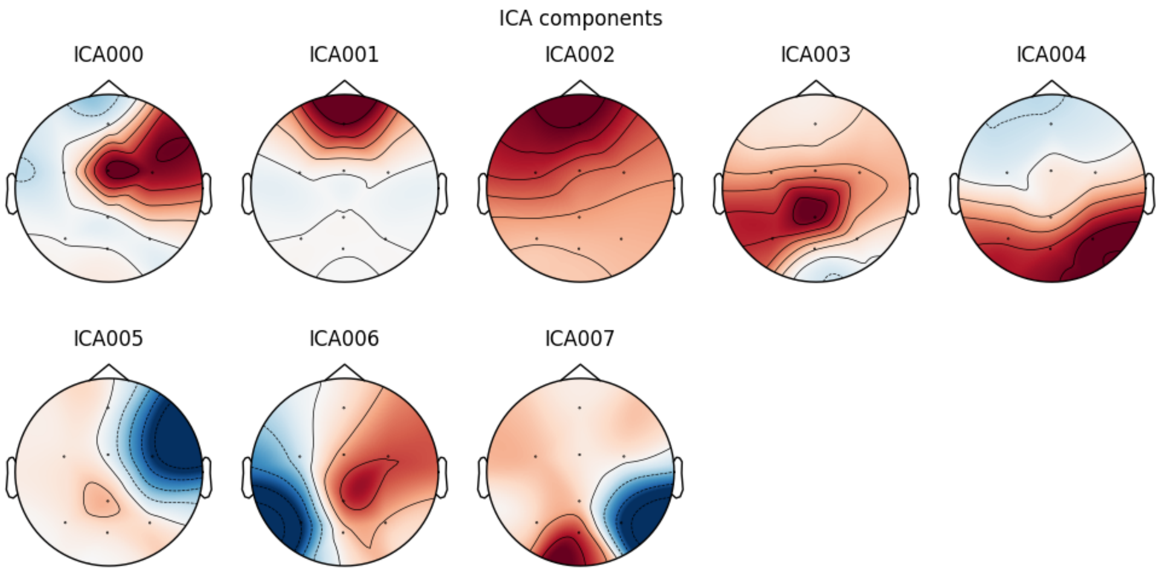


Figure 5.3: Scalp field distribution of each channel from recording in Figure 5.4.

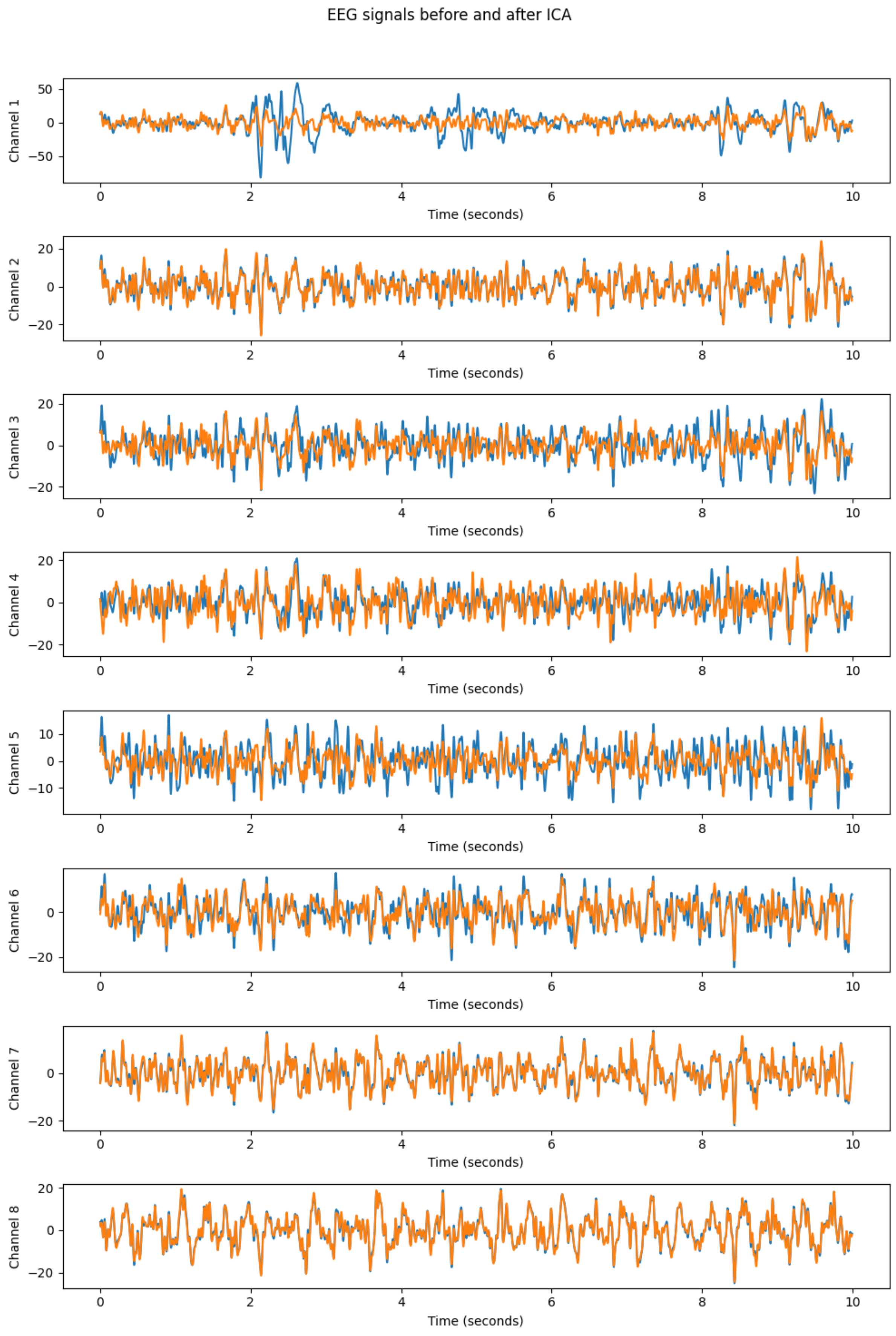


Figure 5.4: Recording of 8 channels before and after ICA. The blue line indicates the recording before ICA and the orange line after ICA. The horizontal axis represents the time in seconds, and the vertical axis represents the EEG power magnitude.

5.4. System layout

All the aforementioned steps can be combined into one subsystem. The system can be summarized into certain steps, as seen in Figure 5.5. First, the data will be acquired from the EEG headset using the LSL interface. It is then collected in a Python environment, where it is processed to remove noise and artifacts. After that, the output signal can be sent to the other subgroups, and an analysis can be performed to determine the signal quality.

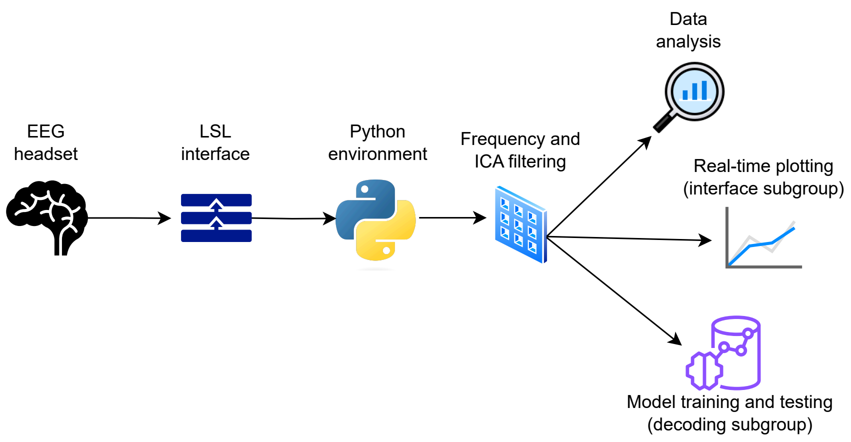


Figure 5.5: Overview of the subsystem

6 Recording analysis

This chapter will focus on the test results from the recordings. These recordings will be plotted and analyzed. The FFT plots will be analyzed to determine if the filters were sufficient. The ERDS plots will also be analyzed to determine if the recordings were of sufficient quality.

6.1. FFT plots

Several FFT plots are displayed and analysed in this section. The FFT is an effective tool for checking the functionality of the filter. It can also be used to detect certain noises and artefacts.

Figure 6.1 shows an image of the FFT of a raw and filtered EEG signal. It is clear that the low frequencies down to 0.5 Hz are attenuated, which is important for a clear visualisation of the meaningful brain signals. You can also see that the peak at 50 Hz is suppressed. This noise corresponded to interference from power lines and is now completely gone. The plots confirm that the filters are working correctly, which ensures that the data is cleaner.

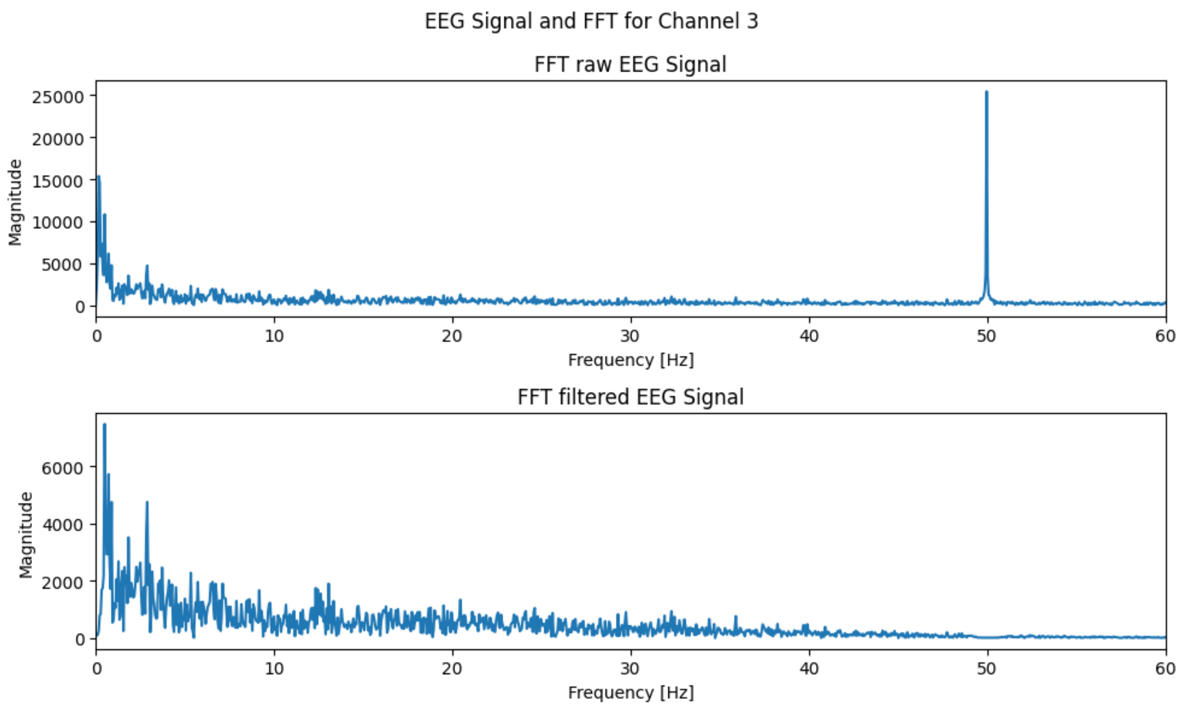


Figure 6.1: FFT of raw and filtered EEG signal of channel 3

6.2. ERDS plots

The theory behind ERDS plots is that different stimuli will create different evoked responses in brain waves at predictable delays to the onset of the stimuli. Furthermore, by averaging over multiple trials, these evoked responses can become visible and distinct from the baseline signal. The steps that were taken to compute the ERDS plots are as follows:

1. Separate recordings into segments for each trial, called epochs.
2. For each epoch, calculate the time-frequency signal power representation for each channel.
3. Baseline readjustment

4. Separate epochs into delta, theta, alpha, and beta bandwidths.
5. For each MI action, plot the average of all the epochs onto a separate plot for each power band channel pair along with the 95% confidence interval.

To outline each of these steps in more detail. The epochs are six seconds long, with 1.5 seconds before the prompt and 4.5 seconds after. The prompt is set to t = 0 for the purpose of plotting, or, in other words, the time axis on the plots is the time since the prompt happened in seconds. The epochs are cropped to -1 to 4 seconds at the same time as the baseline readjustment.

The signal power time-frequency is calculated by first performing the short-time Fourier transform (STFT) on each of the channels for each of the epochs. Then, using discrete prolate spheroidal signal (DPSS) tapers, estimate the power spectral density. This causes the data to become time-frequency power instead of time-frequency amplitude.

The baseline readjustment for each channel in each epoch finds the mean value in the baseline period, which in this case is -1 to 0 seconds. Then it applies the following formula: $y = \frac{x-m}{m}$ where x is the current signal and m is the mean value of the baseline period. The resulting signal y is what is used for the rest of the pipeline. This formula essentially causes the signal to become a relative power compared to the baseline. This, for example, means that a value of 1 corresponds to a power of twice the baseline, a value of 2 corresponds to a power of three times the baseline, and so on. Put another way, a value of 1 is a 100% increase in power compared to baseline.

Since the data is still in time-frequency form, the different frequencies are binned into delta, theta, alpha, and beta, with delta being 2–3 Hz, theta being 3–7 Hz, alpha being 7–13 Hz, and beta being 13–35 Hz.

The plotting function inherently averages values within the same time instance on the same plot. This performs the action of averaging over all epochs. Along with this, it also plots the 95% confidence interval surrounding this average. Each different MI action is plotted in a different color and is treated separately.

The core theory behind ERDS is that it can be used to identify that there is a significant enough difference between the responses of different MI actions, which proves that the measurements are usable. For this project, they are used to verify that the measurement setup and training data collected are adequate for use in the decoding part of the project.

From [36], it is known that the expectation for MI is that there is desychronization in the alpha band and synchronization in the beta band. Furthermore, specifically for the left and right arm movements, the expectation is that the response is highly localized to one side of the brain. Left-hand motor imagery is expected to have a strong response on the right side of the brain with little to no response on the left side. The inverse is true for right-hand motor imagery. Since C3 is the electrode location on the left side of the brain and C4 is the one on the right side of the brain, the evoked response for left and right hand MI should be visible in the C3 alpha and beta and the C4 alpha and beta plots.

For the sake of comparison, ERDS plots of one of the datasets from [22] have been made. This dataset is known to have useful data and has been successfully decoded by various papers. These plots are shown in Figure 6.2.

Analyzing Figure 6.3, the first thing that jumps out is the very strong response to the feet prompt. Based on other recordings of the same test subject and contrasting them to the results of other subjects, it is clear that this is due to a peculiarity of the test subject and not caused by an error in the measurements. From Figure 6.2, the expected response to the tongue prompt is a strong synchronization in the alpha band of the C3 position with weaker responses in the C4 alpha, PZ alpha, C3 beta, Cz beta, C4 beta, and PZ beta. When looking at Figure 6.3, the response can only really be seen in C3 alpha and C4 alpha. These responses have a lower amplitude than that of the test dataset and have practically no response in any of the other locations within the alpha or beta bands. These responses indicate that it should still be possible to decode the tongue signal, but a lower level of accuracy is expected. The response for both the left and right hands for subject 2 is very weak and unlikely to be distinguishable by the decoding.

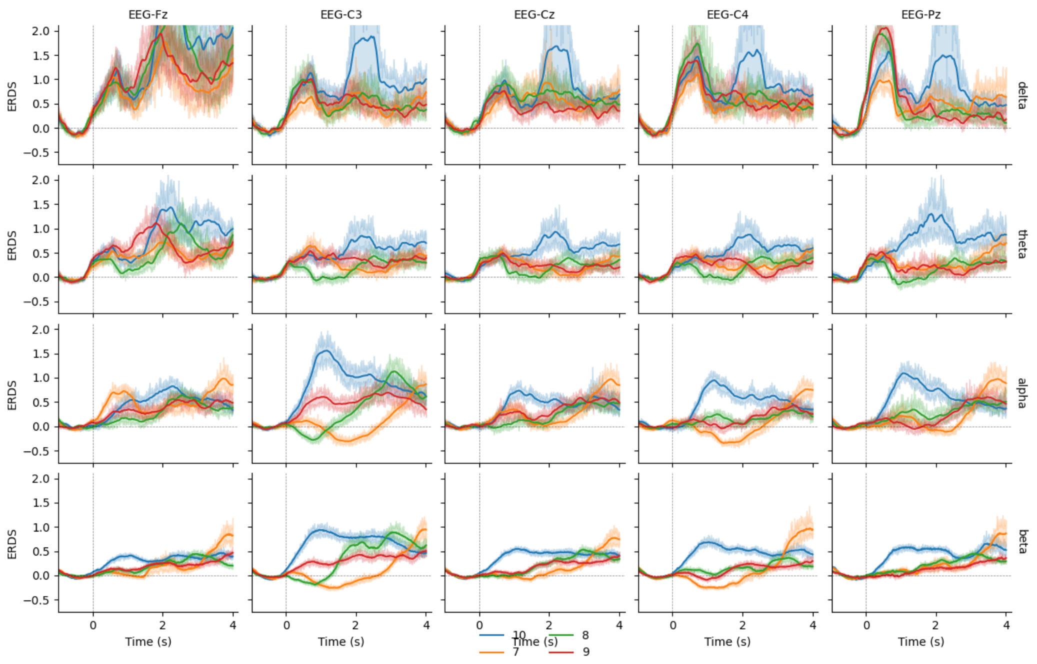


Figure 6.2: ERDS plots of dataset 2a, A09T from [22]. (Key: 7 = left, 8 = right, 9 = feet, 10 = tongue)

6.3. Signal-to-noise ratio

In neuroscience, signal-to-noise ratio (SNR) is defined as signal power divided by noise power as normal; however, measuring it is different. According to [37], the way SNR is measured is by dividing the expected value of the power of the signal by the noise power. Noise power in this way is measured as the power during rest periods. This is described by Equation 6.1 from [37], where ${r_S}^2$ is the power of the response to the stimulus and ${\sigma_N}^2$ is the noise variance. For the case of MI, the expected power from the stimulus is equivalent to the relative ERDS peaks times the baseline power plus the baseline power, and the noise power is equivalent to the baseline power. This gives the relation between the ERDS peaks and the SNR in Equation 6.2, with $ERDS$ being the peak value of the response in the channel with the strongest response and power band where the synchronization response is expected and P_B being the baseline power. From this, the SNR for the different MI actions for subject 2 can be found using Figure 6.3. The SNR values found were:

- 8.87 for feet
- 1.55 for tongue
- 1.10 for left hand
- 1.10 for right hand

From this and the requirement for SNR in chapter 2, it is clear that the requirement is met for the feet and tongue MI actions but not for the left and right hand actions.

$$SNR = \frac{P_S}{P_N} = \frac{E[{r_S}^2]}{{\sigma_N}^2} \tag{6.1}$$

$$SNR = \frac{ERDS \cdot P_B + P_B}{P_B} = ERDS + 1 \tag{6.2}$$

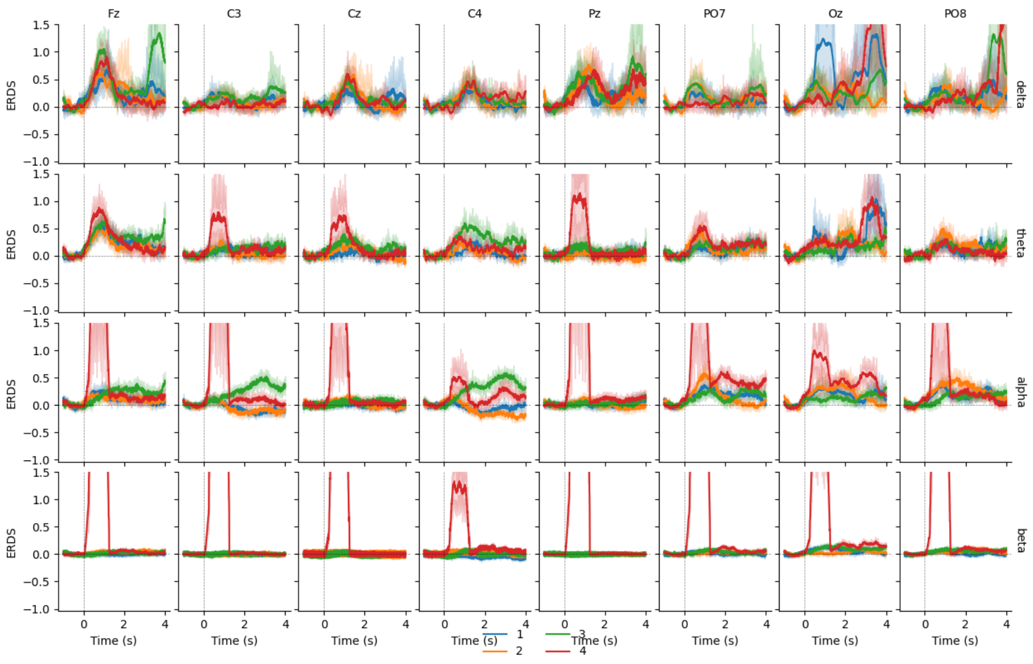


Figure 6.3: ERDS plot of subject 2 with 48 trials per MI action. (Key: 1 = right, 2 = left, 3 = tongue, 4 = feet)

7 Conclusion and discussion

7.1. Conclusion

In short, the main goal of the subgroup was to stream the EEG data in real time and preprocess it for the other subgroups. The recorded data from the subjects will be used by the decoding subgroup to train and test the ML model. For the real-time control group, the data is used to visualize the data and determine the signal quality.

In conclusion, the subsystem is able to successfully stream data from the EEG headset to a Python environment using the LSL framework. Additionally, an experimental setup is designed to have consistent motor imagery trials of four movements. These recordings will be filtered using frequency filters and ICA. Finally, using ERDS plots, it was verified that some but not all of the MI actions are distinguishable for one subject. However, further verification of other subjects is necessary to conclude whether this is a systematic issue or due to the subject. The subsystem meets all requirements except for the SNR requirement for the left and right-hand MI.

The system is able to record using the GUI and a separate user window with a prompt. After recording, it outputs a csv file with all samples, which can be used for the decoding group. However, the integration with the decoding group is not yet complete. The idea is to reshape the recorded data into the desired shape and convert it into a Pytorch array. This can then be used to train on the recorded data. For the testing phase, it is necessary to use a sliding window function in which a certain time frame is classified, but this is not yet completed.

7.2. Discussion

The project collected data from four subjects. Although the number of trials for each subject met the requirements, it may be better to record data from more subjects to obtain a larger and more diverse data set. It should be discussed with the decoding group which amount is sufficient to detect motor imagery signals. Also, the signal quality may improve if subjects are recorded over multiple days. As they get better at imaging the movements, it may produce a stronger motor imagery response. Due to time constraints, it was not possible to have subjects return after the first day of recording.

Due to time constraints, the live filtering did not work perfectly. The method used was a sliding window of the data. The window size used in this project is 500 samples with a 75% overlap. This gave sufficient frequency resolution but lacked high-time resolution. While it has been tested with different parameters, more testing should be conducted to determine the optimal parameters for the overlap and window size.

There is still a certain amount of noise in the signal after filtering. This noise can be caused by the electromagnetic field and the movement of the subject. These artifacts may be removed with alternative filtering methods, such as empirical mode decomposition, discrete wavelet transform, and wavelet packet decomposition. Thus, future research could delve deep into other signal processing techniques and other methods to improve the overall signal quality. Another thing to note is that electrostatic charges can interfere with the EEG signal. Carpet and synthetic fabrics can cause static and should be avoided for recording. It may also be useful to ground additional electrical equipment and other people in the room, such as the interface controller. This will prevent electrostatic charges from building up and potentially affecting the signal. This is not taken into account in this project, but it is something to keep in mind for future research.

Giving subjects real-time feedback when imaging the movements could improve the results. Primarily, the hope with this is that it could increase the SNR for left and right-hand MI to acceptable levels. The screen could, for example, show an arrow that is going in the imagined direction when correctly imagined. However, in this project, this is not integrated and can be something for future research [38].

References

[1] Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. "Brain Computer Interfaces, a Review". In: *Sensors* 12.2 (2012), pp. 1211–1279. ISSN: 1424-8220. DOI: 10.3390/s120201211. URL: https://www.mdpi.com/1424-8220/12/2/1211.

[2] Natasha Padfield et al. "EEG-Based Brain-Computer Interfaces Using Motor-Imagery: Techniques and Challenges". In: *Sensors* 19.6 (2019), p. 1423. ISSN: 1424-8220. DOI: 10.3390/s19061423. URL: https://www.mdpi.com/1424-8220/19/6/1423.

[3] g.tec medical engineering GmbH. *UNICORN HYBRID BLACK*. 2024. URL: https://www.gtec.at/product/unicorn-hybrid-black/ (visited on 06/12/2024).

[4] G. Pfurtscheller and C. Neuper. "Motor imagery and direct brain-computer communication". In: *Proceedings of the IEEE* 89.7 (2001), pp. 1123–1134. DOI: 10.1109/5.939829.

[5] Sangin Park et al. "Improving Motor Imagery-Based Brain-Computer Interface Performance Based on Sensory Stimulation Training: An Approach Focused on Poorly Performing Users". In: *Frontiers in Neuroscience* 15 (2021). DOI: 10.3389/fnins.2021.732545. URL: https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2021.732545.

[6] Jasmin Kevric and Abdulhamit Subasi. "Comparison of signal decomposition methods in classification of EEG signals for motor-imagery BCI system". In: *Biomedical Signal Processing and Control* 31 (2017), pp. 398–406. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2016.09.007. URL: https://www.sciencedirect.com/science/article/pii/S1746809416301331.

[7] M. Jeannerod. "Mental imagery in the motor context". In: *Neuropsychologia* 33.11 (1995). The Neuropsychology of Mental Imagery, pp. 1419–1432. ISSN: 0028-3932. DOI: https://doi.org/10.1016/0028-3932(95)00073-C. URL: https://www.sciencedirect.com/science/article/pii/002839329500073C.

[8] Marc Jeannerod and Jean Decety. "Mental motor imagery: a window into the representational stages of action". In: *Current Opinion in Neurobiology* 5.6 (1995), pp. 727–732. ISSN: 0959-4388. DOI: https://doi.org/10.1016/0959-4388(95)80099-9. URL: https://www.sciencedirect.com/science/article/pii/0959438895800999.

[9] Akira Nakashima et al. "Continuous Repetition Motor Imagery Training and Physical Practice Training Exert the Growth of Fatigue and Its Effect on Performance". In: *Brain Sciences* 12.8 (2022). ISSN: 2076-3425. DOI: 10.3390/brainsci12081087. URL: https://www.mdpi.com/2076-3425/12/8/1087.

[10] Alfons Schnitzler et al. "Involvement of Primary Motor Cortex in Motor Imagery: A Neuromagnetic Study". In: *NeuroImage* 6.3 (1997), pp. 201–208. ISSN: 1053-8119. DOI: 10.1006/nimg.1997.0286. URL: https://www.sciencedirect.com/science/article/pii/S105381199790286X.

[11] Robert M. Hardwick et al. "Neural correlates of action: Comparing meta-analyses of imagery, observation, and execution". In: *Neuroscience & Biobehavioral Reviews* 94 (2018), pp. 31–44. ISSN: 0149-7634. DOI: https://doi.org/10.1016/j.neubiorev.2018.08.003. URL: https://www.sciencedirect.com/science/article/pii/S0149763417309284.

[12] Aymeric Guillot, Franck Di Rienzo, and Christian Collet. "The Neurofunctional Architecture of Motor Imagery". In: May 2014. ISBN: 978-953-51-1203-7. DOI: 10.5772/30961.

[13] Reinhold Scherer and Carmen Vidaurre. "Chapter 8 - Motor imagery based brain–computer interfaces". In: *Smart Wheelchairs and Brain-Computer Interfaces*. Ed. by Pablo Diez. Academic Press, 2018, pp. 171–195. ISBN: 978-0-12-812892-3. DOI: https://doi.org/10.1016/B978-0-12-812892-3.00008-X. URL: https://www.sciencedirect.com/science/article/pii/B978012812892300008X.

[14] M.J. Fu, J.J. Daly, and M.C. Cavusoglu. "Assessment of EEG event-related desynchronization in stroke survivors performing shoulder-elbow movements". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* 2006, pp. 3158–3164. DOI: 10.1109/ROBOT.2006.1642182.

[15] Zhanna Garakh et al. "Mu rhythm separation from the mix with alpha rhythm: Principal component analyses and factor topography". In: *Journal of Neuroscience Methods* 346 (2020), p. 108892. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2020.108892. URL: https://www.sciencedirect.com/science/article/pii/S0165027020303150.

[16] J. A. van Deursen et al. "Increased EEG gamma band activity in Alzheimer's disease and mild cognitive impairment". In: *Journal of Neural Transmission* 115.9 (Sept. 2008), pp. 1301–1311. ISSN: 1435-1463. DOI: 10.1007/s00702-008-0083-y. URL: https://doi.org/10.1007/s00702-008-0083-y.

[17] G. Pfurtscheller and F.H. Lopes da Silva. "Event-related EEG/MEG synchronization and desynchronization: basic principles". In: *Clinical Neurophysiology* 110.11 (1999), pp. 1842–1857. ISSN: 1388-2457. DOI: https://doi.org/10.1016/S1388-2457(99)00141-8. URL: https://www.sciencedirect.com/science/article/pii/S1388245799001418.

[18] G. Pfurtscheller et al. "Mu rhythm (de)synchronization and EEG single-trial classification of different motor imagery tasks". In: *NeuroImage* 31.1 (2006), pp. 153–159. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2005.12.003. URL: https://www.sciencedirect.com/science/article/pii/S1053811905025140.

[19] Janne J.A. Heijs et al. "Validation of Soft Multipin Dry EEG Electrodes". In: *Sensors* 21.20 (2021). ISSN: 1424-8220. DOI: 10.3390/s21206827. URL: https://www.mdpi.com/1424-8220/21/20/6827.

[20] Kaoru Sumi et al. "A Cooperative Game Using the P300 EEG-Based Brain-Computer Interface". In: Apr. 2019, pp. 1–17. ISBN: 978-1-78923-883-9. DOI: 10.5772/intechopen.84621.

[21] Christian Kothe et al. "The Lab Streaming Layer for Synchronized Multimodal Recording". In: *bioRxiv : the preprint server for biology* (Feb. 2024). DOI: 10.1101/2024.02.13.580071.

[22] Clemens Brunner et al. *BCI Competition 2008 – Graz data set A*. Accessed: 2024-06-12. Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology, 2008. URL: https://www.bbci.de/competition/iv/.

[23] C. Brunner and F. Klotzsche. *Compute and visualize ERDS maps.* 2024. URL: https://mne.tools/stable/auto_examples/time_frequency/time_frequency_erds.html (visited on 05/30/2024).

[24] Mário L. Vicchietti et al. "Computational methods of EEG signals analysis for Alzheimer's disease classification". In: *Scientific Reports* 13.1 (May 2023), p. 8184. ISSN: 2045-2322. DOI: 10.1038/s41598-023-32664-8. URL: https://doi.org/10.1038/s41598-023-32664-8.

[25] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[26] Xinqiao Zhao et al. "A Multi-Branch 3D Convolutional Neural Network for EEG-Based Motor Imagery Classification". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27.10 (2019), pp. 2164–2177. DOI: 10.1109/TNSRE.2019.2938295.

[27] Gabriella Tamburro et al. "Automated Detection and Removal of Cardiac and Pulse Interferences from Neonatal EEG Signals". In: *Sensors* 21.19 (2021). ISSN: 1424-8220. DOI: 10.3390/s21196364. URL: https://www.mdpi.com/1424-8220/21/19/6364.

[28] John G. Proakis and Dimitris K. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications.* 4th ed. Pearson International Edition, 2007.

[29] Zhiguo Zhang Li Hu. *EEG Signal Processing and Feature Extraction.* Springer Singapore, 2019. ISBN: 978-981-13-9112-5.

[30] Wanzeng Kong et al. "Automatic and Direct Identification of Blink Components from Scalp EEG". In: *Sensors* 13.8 (2013), pp. 10783–10801. ISSN: 1424-8220. DOI: 10.3390/s130810783. URL: https://www.mdpi.com/1424-8220/13/8/10783.

[31] Anita Safitri, Esmeralda Contessa Djamal, and Fikri Nugraha. “Brain-Computer Interface of Motor Imagery Using ICA and Recurrent Neural Networks”. In: *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*. 2020, pp. 118–122. DOI: `10.1109/IC2IE50715.2020.9274681`.

[32] Vaibhav Gandhi. “Chapter 2 - Interfacing Brain and Machine”. In: *Brain-Computer Interfacing for Assistive Robotics*. Ed. by Vaibhav Gandhi. San Diego: Academic Press, 2015, pp. 7–63. ISBN: 978-0-12-801543-8. DOI: `https://doi.org/10.1016/B978-0-12-801543-8.00002-8`. URL: `https://www.sciencedirect.com/science/article/pii/B9780128015438000028`.

[33] Pierre Ablin, Jean-François Cardoso, and Alexandre Gramfort. “Faster Independent Component Analysis by Preconditioning With Hessian Approximations”. In: *IEEE Transactions on Signal Processing* 66.15 (2018), pp. 4040–4049. DOI: `10.1109/TSP.2018.2844203`.

[34] MNE. *Repairing artifacts with ICA*. 2024. URL: `https://mne.tools/stable/auto_tutorials/preprocessing/40_artifact_correction_ica.html` (visited on 05/30/2024).

[35] Alexandre Gramfort et al. “MEG and EEG Data Analysis with MNE-Python”. In: *Frontiers in Neuroscience* 7.267 (2013), pp. 1–13. DOI: `10.3389/fnins.2013.00267`.

[36] G. Pfurtscheller and C. Neuper. “Motor imagery and direct brain-computer communication”. In: *Proceedings of the IEEE* 89.7 (2001), pp. 1123–1134. DOI: `10.1109/5.939829`.

[37] S.R. Schultz. *Signal-to-noise ratio in neuroscience*. 2007. URL: `http://www.scholarpedia.org/article/Signal-to-noise_ratio_in_neuroscience`.

[38] Tianyou Yu et al. “Enhanced Motor Imagery Training Using a Hybrid BCI With Feedback”. In: *IEEE Transactions on Biomedical Engineering* 62.7 (2015), pp. 1706–1717. DOI: `10.1109/TBME.2015.2402283`.

A Experiment setup Prompts

A.1. Stabilize cross

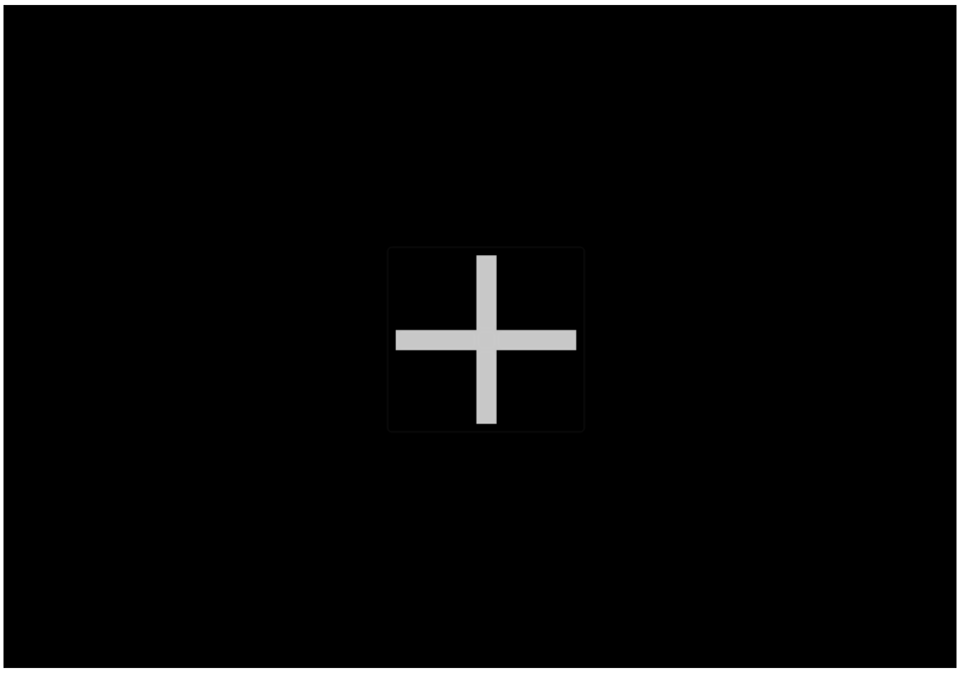


Figure A.1: Prompt of stabilizing cross

A.2. Left hand

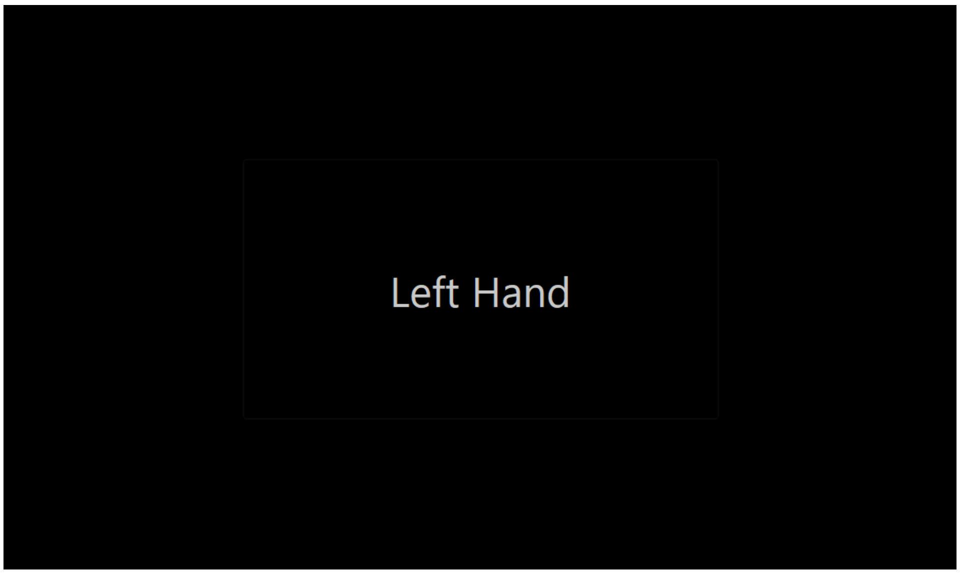


Figure A.2: Prompt of left hand

A.3. Right hand

Figure A.3: Prompt of right hand

A.4. Tongue

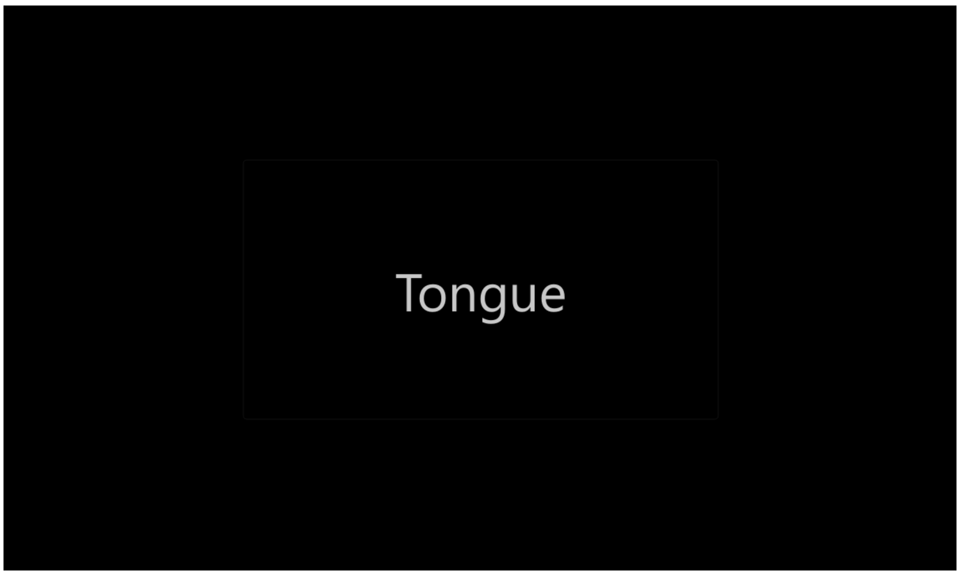


Figure A.4: Prompt of tongue

A.5. Feet

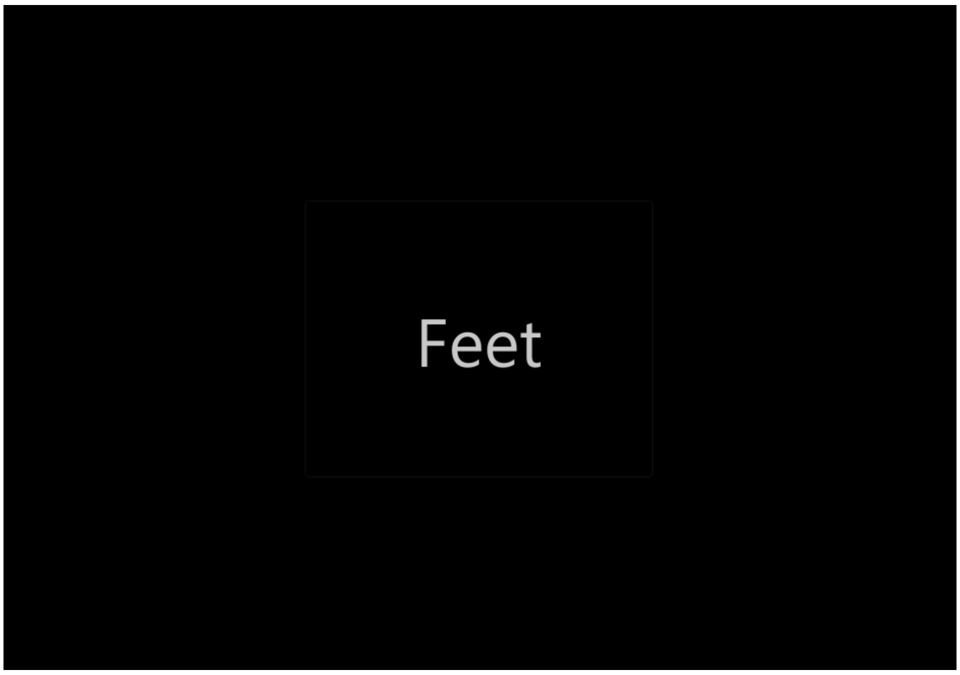
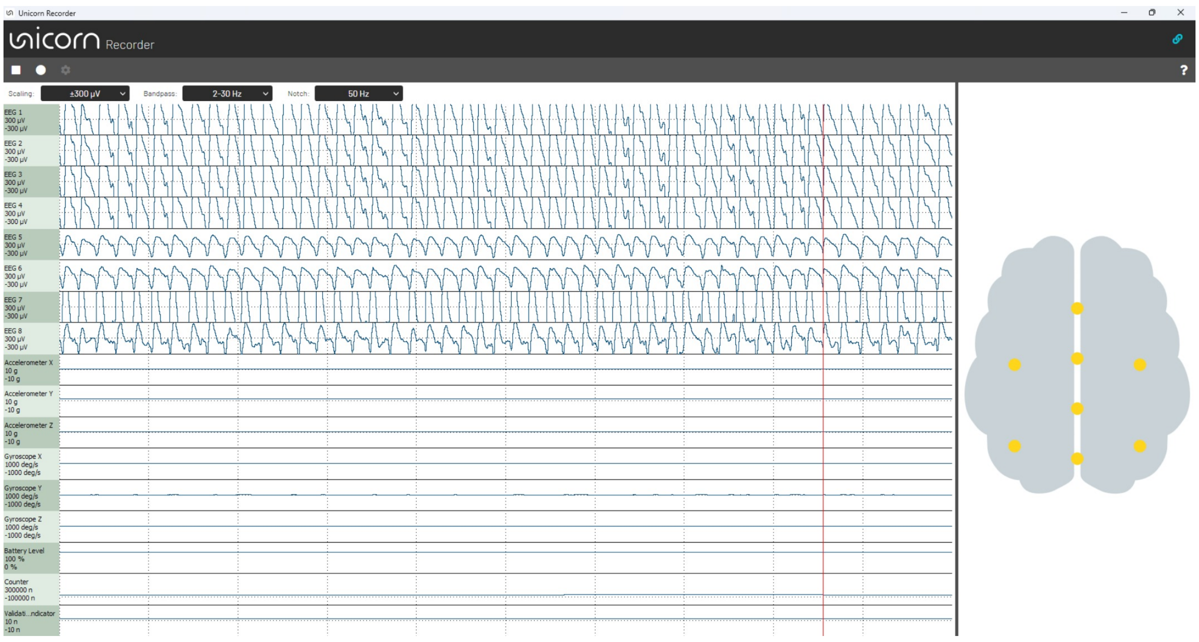


Figure A.5: Prompt of feet

B Software applications

B.1. Unicorn Recorder


Scaling: ±300 µV

Bandpass: 2-30 Hz

Notch: 50 Hz

Unicorn Recorder
unicorn Recorder
EEG 1 300 µV -300 µV
EEG 2 300 µV -300 µV
EEG 3 300 µV -300 µV
EEG 4 300 µV -300 µV
EEG 5 300 µV -300 µV
EEG 6 300 µV -300 µV
EEG 7 300 µV -300 µV
EEG 8 300 µV -300 µV
Accelerometer X 10 g -10 g
Accelerometer Y 10 g -10 g
Accelerometer Z 10 g -10 g
Gyroscope X 1000 deg/s -1000 deg/s
Gyroscope Y 1000 deg/s -1000 deg/s
Gyroscope Z 1000 deg/s -1000 deg/s
Battery Level 100 % 0 %
Counter 300000 n -100000 n
Validati...ndicator 10 n -10 n

Figure B.1: Recorder software integrated into the EEG headset

B.2. Unicorn LSL interface

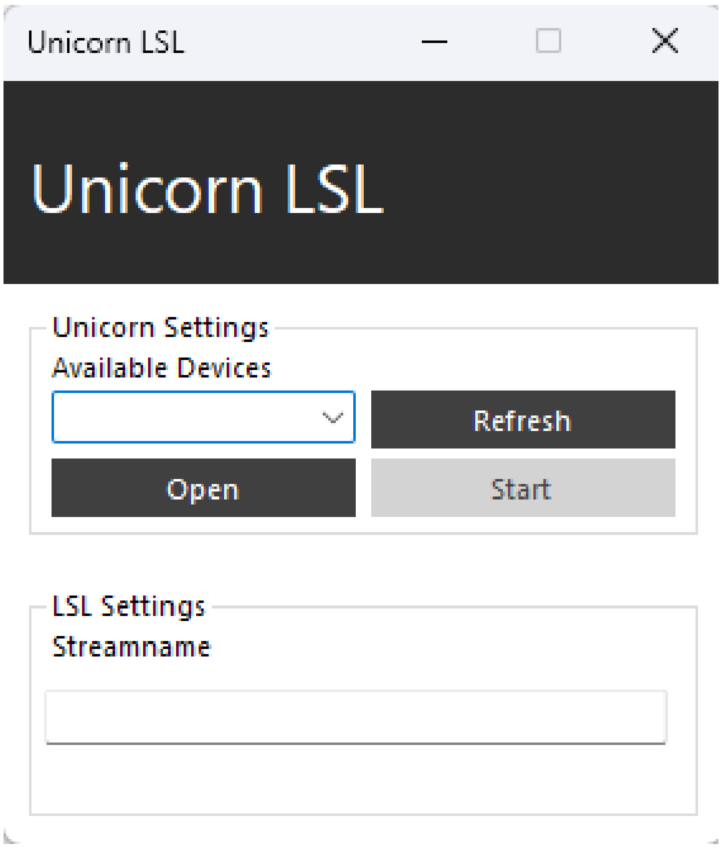


Figure B.2: LSL interface

C Filters

C.1. Frequency responses

C.1.1. Bandpass filter

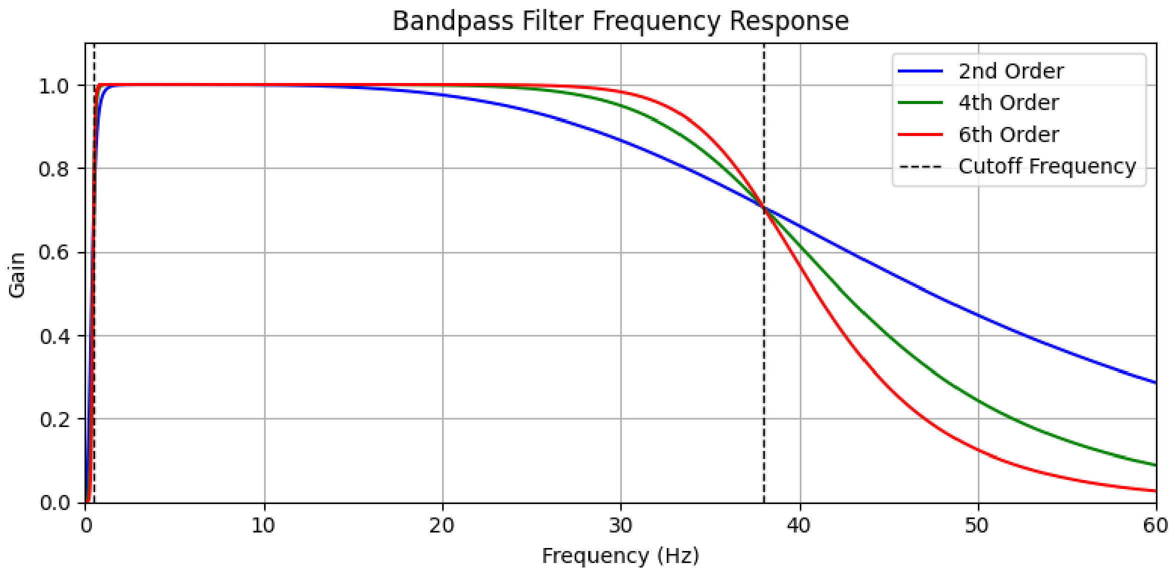


Figure C.1: Frequency response of Butterworth bandpass filter from 0.5 Hz to 38 Hz. Different orders are compared.

C.1.2. Notch filter

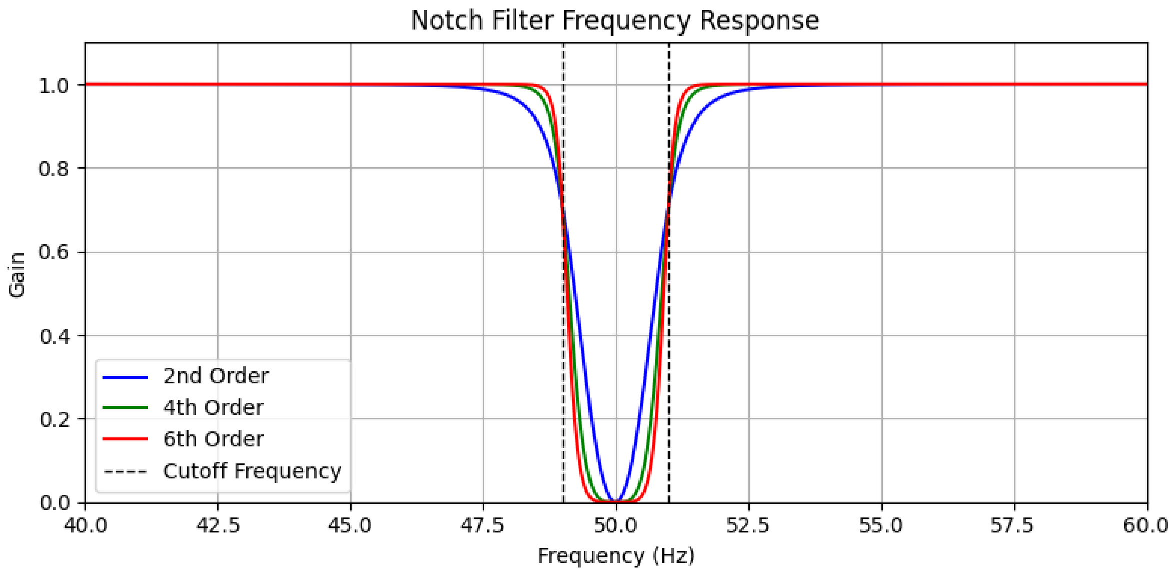


Figure C.2: Frequency response of Butterworth bandstop filter from 49 Hz to 51 Hz. Different orders are compared.

C.1.3. Filter comparison for different cut-off frequencies

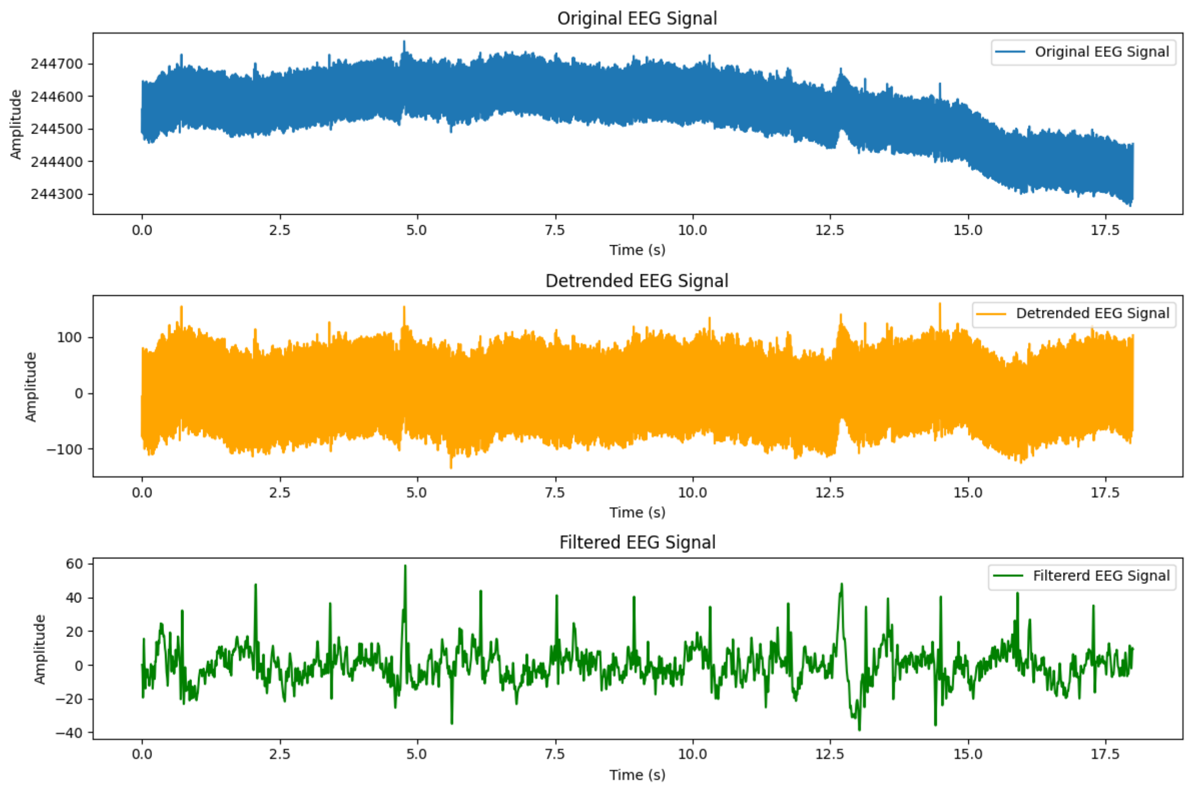


Figure C.3: Comparison between original EEG signal from channel 1 (Fz), detrended and filtered signal. Filtered with a bandpass filter of 0.5 Hz to 30 Hz.

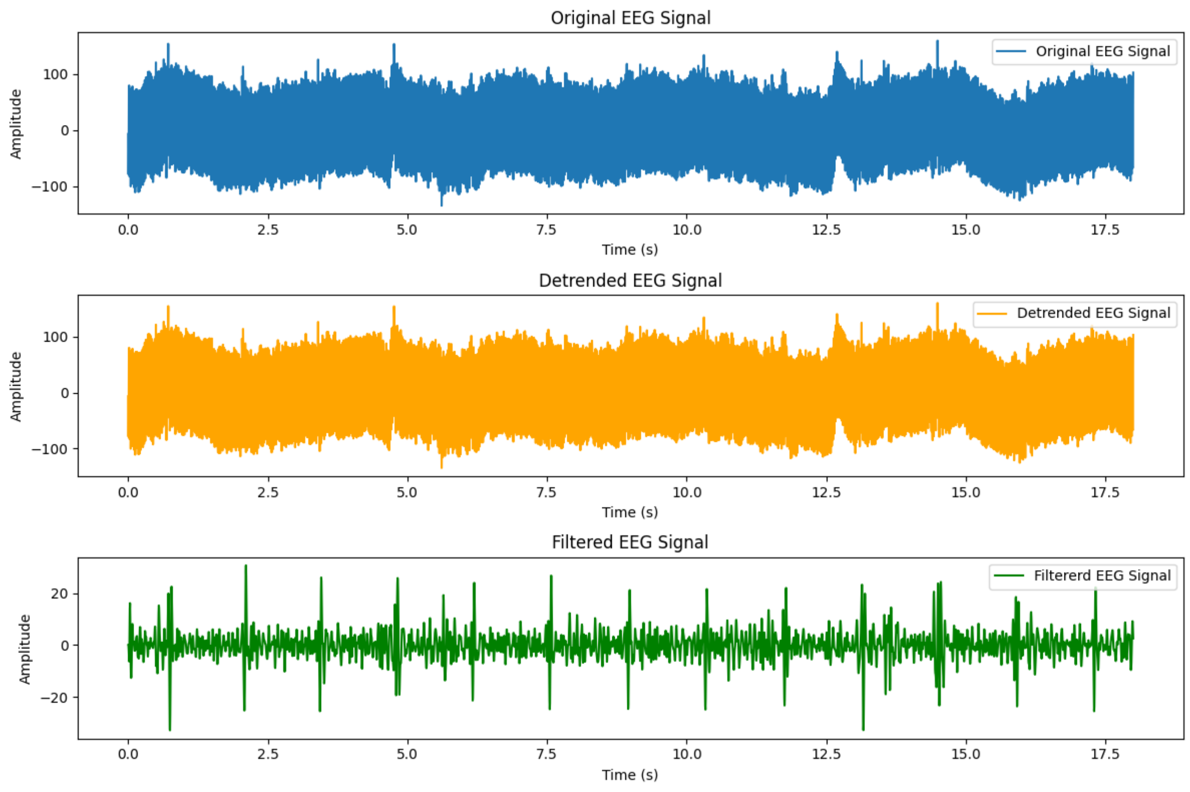
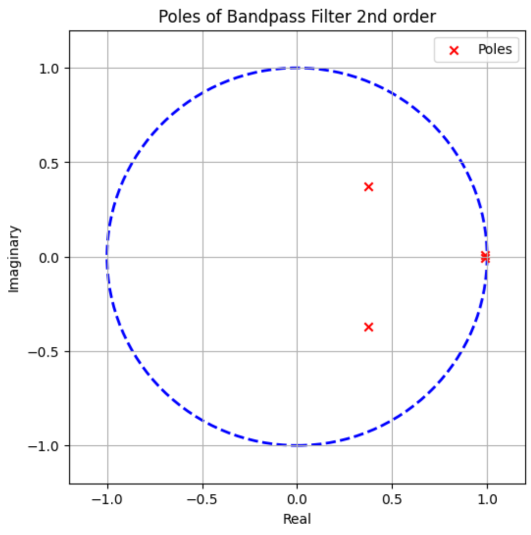


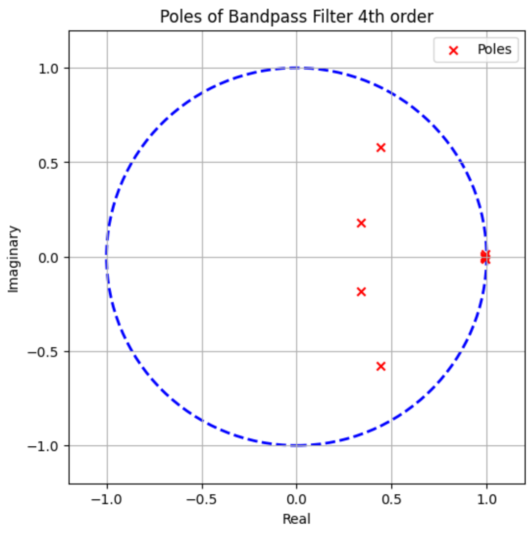
Figure C.4: Comparison between original EEG signal from channel 1 (Fz), detrended and filtered signal. Filtered with a bandpass filter of 8 Hz to 30 Hz.

D Z-plane plots filter

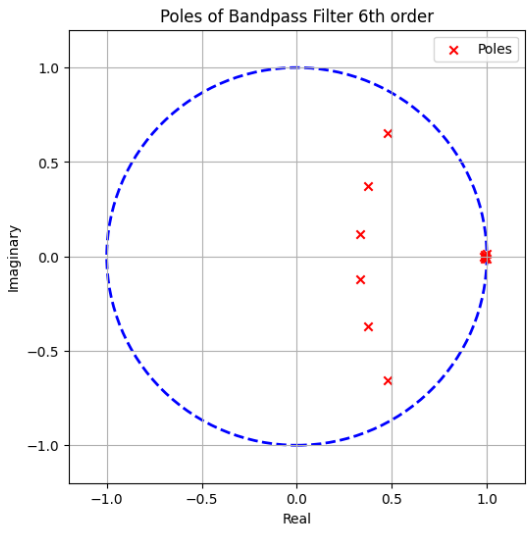
D.1. Bandpass filter



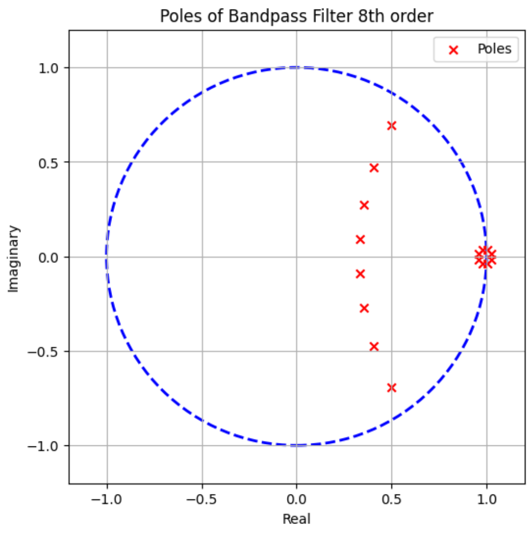
(a) Poles of the 2nd order Butterworth bandpass filter from 0.5 Hz to 38 Hz in z-plane



(b) Poles of the 4th order Butterworth bandpass filter from 0.5 Hz to 38 Hz in z-plane



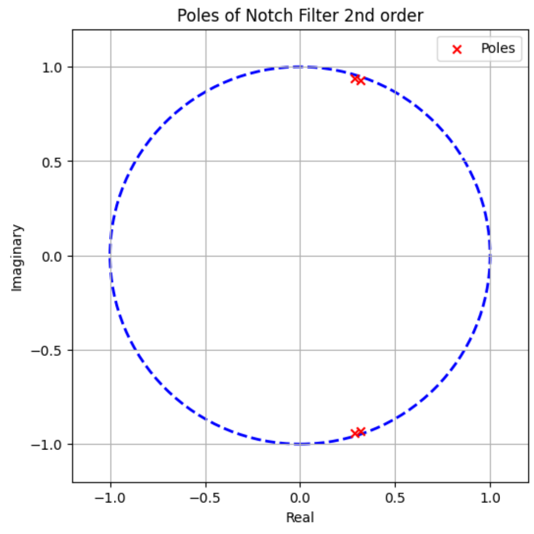
(c) Poles of the 6th order Butterworth bandpass filter from 0.5 Hz to 38 Hz in z-plane



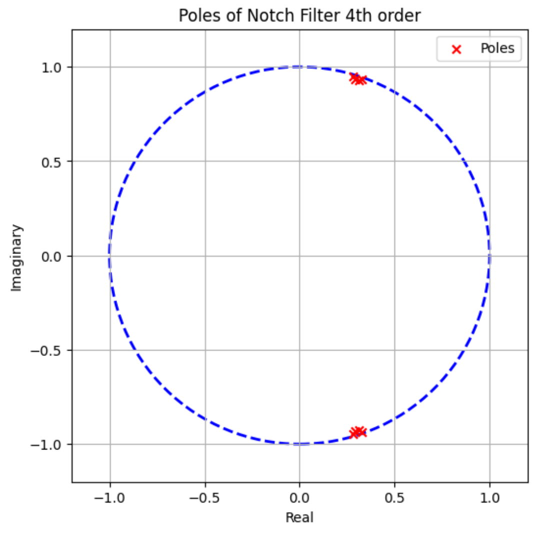
(d) Poles of the 8th order Butterworth bandpass unstable filter from 0.5 Hz to 38 Hz in z-plane

Figure D.1: Poles of Butterworth bandpass filters in the z-plane

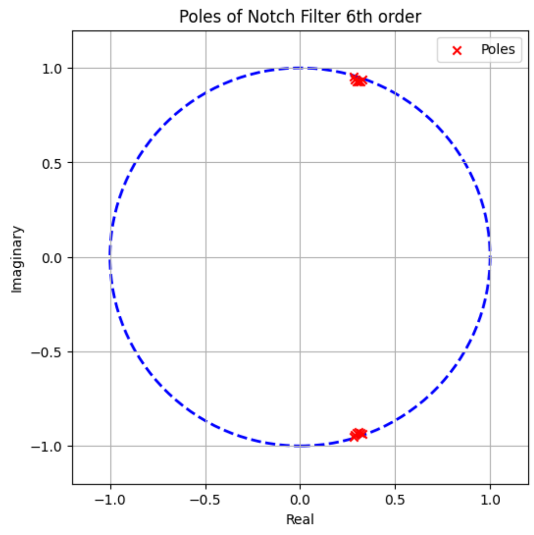
D.2. Notch filter



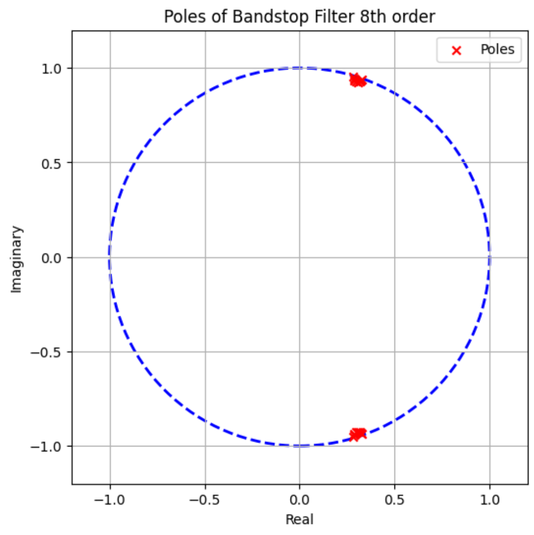
(a) Poles of the 2nd order Butterworth notch filter from 49 Hz to 51 Hz in z-plane



(b) Poles of the 4th order Butterworth notch filter from 49 Hz to 51 Hz in z-plane



(c) Poles of the 6th order Butterworth notch filter from 49 Hz to 51 Hz in z-plane



(d) Poles of the 8th order Butterworth notch filter from 49 Hz to 51 Hz in z-plane

Figure D.2: Poles of Butterworth notch filters in the z-plane