

Controlling and Navigating in a Virtual Environment Using Brain Control Interface System

Julien Gergi Sarkis

A thesis submitted for the degree of Master of Science in Intelligent Systems and Robotics

Abstract: A technology that was considered a science fiction in the past, is being researched in the present, and in the future, it will be a technology that humans will be highly depending on in their daily life. The Brain-Computer Interface is a pipeline between the brain and machines, it is a way of communication that utilizes the brain signals as inputs to be used in a system, and the output could be anything that humans could benefit from. The BCI has many types and multiple numbers of ways to be processed, which are being researched daily to be improved and that one day it will be a one-hundred percent reliable communication, this field brings positive additions to the humans' life and can dramatically improve the quality of life especially for the motor disabled people. The huge difference that the BCI can have on human life is a critical reason that motivates researchers and scientists to bring this field to its highest potential.

Keywords: Brain-Computer Interface, Motor Imagery, Virtual reality, Virtual Environment, Classification, Machine Learning, Lab Streaming Layer, Common Spatial Pattern.

Supervisor: Professor. Francisco Sepulveda

School of Computer Science and Electronic Engineering

University of Essex

August 2019

Contents

1. Introduction:	1
1.1. Proposed Solution:	2
1.4. Document Summary:	3
2. Updated Background Study:	4
2.1. Motor Imagery (MI) Brain Computer Interface:	4
2.2. Electroencephalogram (EEG) Signal:	4
2.3. Machine Learning and Signal Processing:	6
2.4. Unity3D	7
3. Updated Literature Review:	7
4. Implementation and Testing	10
4.1. Unity 3D:	10
4.2. OpenVibe:	10
4.3. Implementation:	10
4.4. Recording EEG Signal	18
4.5. The Dataset:	22
4.6. Virtual Environment using Unity3D:	22
4.7. Connecting OpenVibe and Unity3D:	23
4.8. OpenVibe Configuration:	23
4.9. Unity3D Configuration and Script Editing:	27
4.10. Challenges and Solutions:	31
5. Evaluation and Discussion:	33
5.1. Machine learning algorithms:	33
5.1.1. Linear Discriminant Analysis (LDA):	33
5.1.2. Multi-Layer Perceptron:	33
5.2. Confusion Matrix:	34
5.3. Wilcoxon Rank-Sum Test:	34
5.4. Results:	35
5.5. Results Summary	59
5.6. Discussion:	59
5.6.1. Testing of Linear SVM and LDA with 10k fold using Wilcoxon test:	60
5.6.2. Testing of 10k fold LDA with MI and CSP MI using Wilcoxon test:	61
6. Project Management:	63
6.1. Updated Plan:	63

6.2. Kanban Board:	65
6.3. Changes from The Proposal:	67
6.4. Reflective Discussion:	67
7. Conclusion:	69
7.1. Future Direction:	69
Bibliography	1

Figure 1: The System	3
Figure 2: Placement of the electrodes on the scalp [9].	5
Figure 3 OpenVibe Software MI System	12
Figure 4 Signal before DSP	13
Figure 5 Signal after DSP	14
Figure 6 CSP MI Train.	16
Figure 7 CSP Classifier trainer.	17
Figure 8 Enobio Connection part 1	18
Figure 9: Enobio Connection part 2	19
Figure 10: Signals Captured by Enobio	19
Figure 11: OpenVibe Design for Signal Recording	20
Figure 12: Placement of the Sensors	21
Figure 13 BCI MI Online Design Part 1	24
Figure 14 BCI MI Online Design Part 2	25
Figure 15 Graz Visualization	26
Figure 16 Matrix Display	26
Figure 17 LSL Export Configuration Box.	27
Figure 18 Unity Script for LSL.	27
Figure 19 Unity Stream Name and Type	28
Figure 20 Fixed Update Code for The Movement of The Character.	29
Figure 21 GetInput Function.	30
Figure 22 Communication process between OpenVibe and Unity3D	32
Figure 23: Kanban Board part 1	65
Figure 24 Kanban Board part 2	66

1.Introduction:

In recent years, Brain-Computer Interface (BCI) has become a promising field for the future, especially that it has many capabilities and applications, such as improving the quality of life of motor disabled people, providing the game industry with a more real-life experience while gaming and who knows maybe in the future the ability to drive a spaceship in space just by using brain signals. The Brain-Computer Interface is still a research in development as it has difficulties and obstacles that blocks it from being a reliable technology in the application that humans do day by day, for example a motor disabled person crossing the road using BCI technology is not safe, because the current accuracy of the BCI choosing the right option is not enough to be a reliable technology that will serve on a day by day basis. However, in the near future the BCI could change the way that humans live and provide an easier and better life.

Living in the era of technology, human's dependency on computer systems is increasing by the day, which makes the BCI field a popular field and the improvement of the BCI is an interest for all researchers and companies specializing in the human-computer interaction [1]. The Brain-Computer Interface has the ability to translate brain signals to become a sort of communication between the user and the environment, which makes it an excellent tool for the motor disabled people [2]. The BCI works on electrical signals called EEG (Electroencephalogram), which are recorded from a certain region of the brain, the Brain-Computer Interface utilizes many types of paradigms [3], in this project Motor Imagery (MI) is the paradigm that will be used. Motor Imagery will be used to control the movement of a first-person perspective character inside the Unity3D software, which will either move left or right.

In the world, there is a big number of disabled people who has physical challenges with walking or moving and they are in need of help to have a better quality of life, however, the ability to connect the brain with a machine could of great use to everyone and especially the motor disabled people [4]. With the classification of the brain signals, the correct training and further researches in the field of BCI, a dream of many around the world would come true, this is mainly a huge motivation for this project and the main reason of working on it. Moreover, this process, unfortunately, is not an easy one, it requires choosing the right classifier, the right features, multiple numbers of testing and many other factors that affect the performance of the system. However, in this project, the testing of many configurations is

carried out and evaluated with the purpose of improving the performance of the BCI system. In the next paragraph, the goals and objectives achieved in this project are discussed.

1.1. Proposed Solution:

This project has a vast depth and what can be accomplished in this field is far beyond what can be done in a short period of time, for this reason goals and objectives that can be achieved in the given time to work on this project had to be set, with the ability to improve and take the project to the next level in further studies. As like any project that needs to be successful, goals and objectives are set and worked on to be achieved, here is a list of the goals and objectives for this project.

1.2. GOALS:

- Development of a BCI system that is capable of deciding which direction is needed.
- Development of a virtual environment where a character will move left or right depending on the brain signals.
- Using machine learning techniques, classification of the brain signals is done.
- Using Lab Streaming Layer, a communication between OpenVibe and Unity3D is done.
- Recording of the brain signal.

1.3. Objectives:

- Using the Motor imagery paradigm left or right movement is decided by the system.
- Building a suitable virtual environment using Unity3D software.
- The BCI system is built using OpenVibe software.
- Signal processing techniques are used to improve the performance of the system.
- Many machine learning techniques are tested using multiple numbers of configurations, compared and evaluated in the purpose of choosing the best system that can be achieved.
- Evaluation of the whole project.
- Using Enobio20G to record EEG brain signal.

In the Fig.1 below, a brief look at the system can be seen.

As seen in Fig.1, Pre-recorded or live brain signals can be used to enter OpenVibe, the latter software take care of the signals, by processing them and extracting the needed features, applies machine learning and classifies them, then the configuration is used in an online design that will send data to a stream called Lab Stream Layer (LSL) which will be then

available for Unity3D to read it from and thus use that data to control what is needed to be controlled from the brain signals, which in this case is a character.

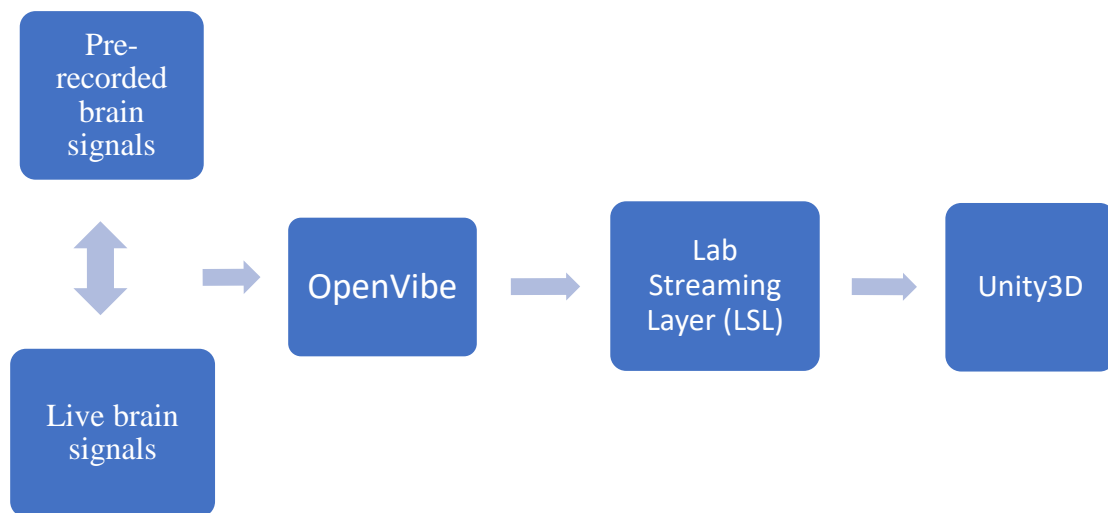


Figure 1: The System

1.4. Document Summary:

After the background study and literature review done in the Project Proposal, to understand the aspects of the project, have a general knowledge on the subject and look through similar projects, an updated literature review and background study will be carried out. Then, all the configurations used and tested will be shown with their performance in each of the machine learning technique that was tried in this project. However, after all the tests and configurations, a discussion about the results and the settings that were used is done to further understand what was done in the project and to able to discover ways to improve the project. After that, and before the summary and conclusion, a section for how the project was managed, and what was changed from the proposal and applied in this project.

2.Updated Background Study:

In the project proposal, an overview of the BCI principles, history and uses in field. In this project, further reading was done, and an updated version of the background study is presented below.

2.1.Motor Imagery (MI) Brain-Computer Interface:

Brain-Computer Interface (BCI) is a way of linking the human brain with any computer-related machine. There are many types of BCI paradigms and frameworks, Motor Imagery (MI) is an excellent paradigm for a BCI system especially when related to movement.

Depending on the EEG signal of the brain, input for a needed activity can be sent from the brain [5]. However, Motor Imagery (MI) is when the user imagines a movement of a body part which will make the EEG signals patterns on the sensorimotor region of the scalp. [6]

The BCI-MI has proven to be successful in many applications such as robot control, moving a mouse on a computer screen, wheelchair control, and many more. Moreover, a critical point in BCI that includes EEG signals and Motor Imagery type is the extraction of the right features from the EEG signal which has noise and low spatial resolution. [7] Common Spatial Pattern (CSP) is a common way of feature extraction which will find the spatial filters with a maximum of variance to a class and minimum to another, but it has some problems with overfitting and being sensitive to outliers. [7] Because of the advancement of the hardware and software in recent years, the implementation of the BCI has become easier, however, the main goal of the BCI is to help people with severe disabilities related to motor tasks, Motor Imagery BCI paradigm is one of the most popular for this task, it is chosen to arouse rhythmic cortical activities when recording EEG signals. The difference that happens in the neural connectivity is to be extracted from the scalp using a spatial filter which is the common spatial pattern, this method has been used in many MI BCI to improve the recognition of the pattern. [8]

2.2.Electroencephalogram (EEG) Signal:

After the discovery of the EEG signal and the first recording of it in 1924, a path to analyze the human brain activity became clearer, however, after all the researches and experiments the EEG signal is now used to control equipment such as a wheelchair, a cursor on a screen and many more uses. [9]

Moreover, the placement and position of the electrodes that will measure the EEG signal from the brain, is 10-20 international electrode standards such like from an ear to another. The 10-20 system is just a distance between the sensors which is ten or twenty percent from the total distance forward and backward or from right to left of the scalp. Looking at Fig.2 an example of the placement of the electrodes on the scalp can be seen. [9]

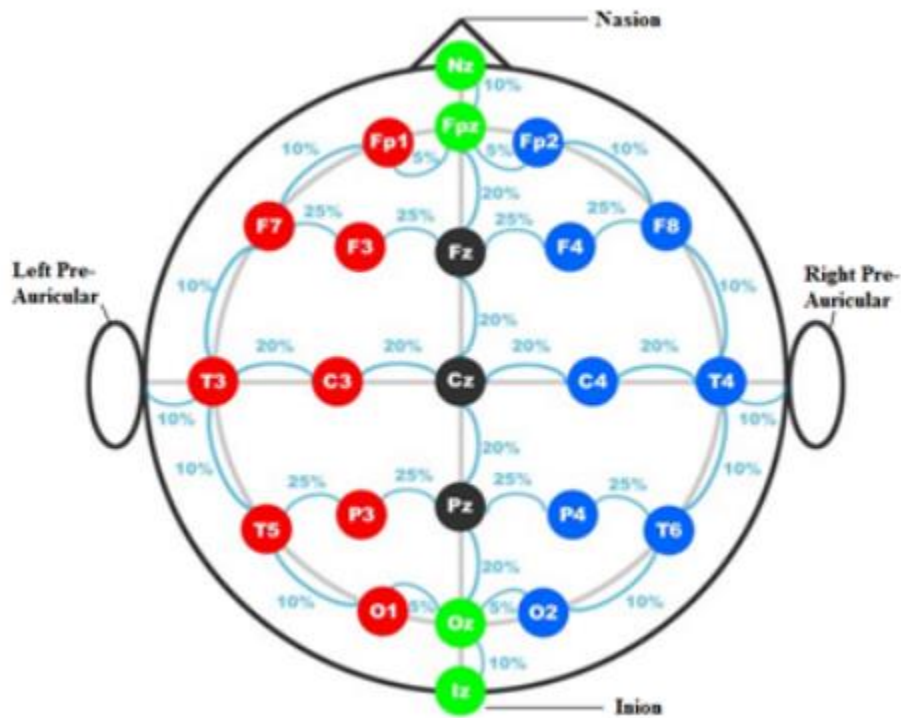


Figure 2: Placement of the electrodes on the scalp [9].

Electroencephalogram (EEG) is a non-invasive way for the measurement of the brain signals. However, recording the EEG is efficient in cost, can be done anywhere and is harmless to humans, the signal can then be used for many purposes, mostly to communicate with a machine using the motor function of the brain or other functions. The advancement in the EEG recording technology is making it easier to use the BCI in the real-world application. [10]

The EEG Signal has many properties, such as Steady-State Visual evoked potentials (SSVEP) Motor Imagery (MI) and P300. The MI is the most commonly used paradigm, as it is an easy and natural way that is normally used in everyday life especially when moving a robot or any navigation specific task. Heavy researches are being done to make more practical and better BCI system that can be used in real-life application, which can be done

by improving the methods of the EEG signal processing, the three main signal processing are [11]:

- EEG signal artifact removal
- Feature extraction
- Classification

2.3. Machine Learning and Signal Processing:

In this project, three machine learning algorithms were used which are:

- Linear Discriminant Analysis (LDA)
- Support Vector Machine (SVM)
- Multi-Layer Perceptron

After the signal processing in the BCI system, classification of the signal obtained is done using machine learning algorithms. However, in a machine learning way, the feature selection step takes care of the computation time while having a high accuracy in the classification, in a research many feature extraction technique was tried and compared with each other, it was concluded that utilizing all the features does not necessarily increase the performance of the classification but will increase the computation time. [12]

One of the most important factor to make the BCI system a successful one is an effective signal processing which makes the translating of the brain signals more clear for the classifier and then to the machine [13]. Moreover, the processing of an EEG signal will make use of the signal to create patterns that will be recognized by the classifier algorithm that will be used [14].

However, the signal processing part of the BCI is a challenging and important part specially used in real-time applications. The feature extraction part of the signal processing is the part where a feature is extracted from the signal and then used as pattern to be recognized by the classifier. [15]. So, the signal processing in simple words encodes the message that is been sent from the brain for the machine to read.

Common Spatial Patterns (CSP) is a commonly used method for Motor Imagery BCI, it is a popular spatial filter which extracts discriminative spatial patterns by training. The CSP algorithm projects the EEG signal to a direction which is most discriminative. However, CSP has some negative effect on the big variance that occurs during the training sessions [16].

2.4. Unity3D

Unity3D is the worlds most famous and leading real-time designing platform, it is a tool for the creation of more than half of the games that exist in the world. [17]

Unity3D supports 2D and 3D, it operates on Windows, Mac, and Linux with a user-friendly interface to navigate in the software. It has a built-in artificial intelligence NPCs that can be used to move around the virtual world. With the prefabs that can be found online, a smooth workflow can be achieved. Unity3D utilizes Box2D and NVIDIA PhysX as its physics engine. [17]

As this software has a vast functionality and potential, there are many online courses that can be taken to be able to utilize Unity3D's best functionalities and features. However, in this project a course provided by Lynda has been taken that covered mostly all the features of Unity3D. [18]

3.Updated Literature Review:

The Brain-Computer Interface connects the brain with the outside world, it creates a communication between the signals of the brain and any computer-related machine, by translating these signals into commands to control devices [19], which opens a wide variety of applications and ways to make use of this field.

In a project, EEG signals were recorded from humans performing four different tasks, the subjects are told to read a sentence in their head without saying it out loud and with no muscle movements, and the sentence was depending on the movement of a robot in the room, so if forward they will say forward and so on. However, the overall performance after the signal processing and the classifier trainer was between 87.50% to 95.00% depending on the subject. [19]

In another project, the BCI was used in an Active and Assisted Living system which makes a smart environment and it can help the person living in it daily. The system has three important units, one for the signal acquisition, one for the signal processing feature extraction and classification, and another one for the output of the system [20].

In a similar project, BCI system was used for navigation and control of a semiautonomous mobile robot in an indoor environment, it used an online SSVEP BCI, which uses four brain states to output four different control commands using MSI for the classification. The project turns out to be feasible as subjects tried and were able to navigate the robot only by using their thoughts [21].

Another project worked on knowing the mental states of humans using the EEG BCI using machine learning algorithms. A dataset of 25 hours of EEG signals recorded from five users that were doing low-intensity tasks, such as controlling a train on a computer for about 35 to 55 minutes. The mental states studied in this project are focused, passive attention and not focused but awake. However, the BCI system is to distinguish between these states. The SVM classifier was used in this project and the accuracy results of the experiments done on the healthy users using that classifier range between 88.60% to 96.70% (depending on the user) which is averaged in five folds. [22]

On a similar work, a virtual reality environment is controlled using the Brain-Computer (BCI), BCI2000 is used to record EEG signals from users, using networking tools commands

are sent from this software to Unity3D. Unity3D is used here to create the virtual environment. While BCI2000 is running in the background the user is communicating with Unity [23]. In another study, a 2-class Motor Imagery (MI) BCI system is worked on, which utilizes visual feedback showing left and right-hand imagery, in this project, two experiments were done, the first one had the volunteers using Motor Imagery (MI) without the feedback and the second one with the visual feedback. Five volunteers participated in this project with only one participant with an experience of Motor Imagery training. The signal acquisition was recorded using three electrodes C3, C4 and Cz placed on the sensorimotor cortex where the MI happens, after the acquisition step, signal processing, and classification step is done using Common Spatial Pattern (CSP), then after the filtering Principal Component Analysis (PCA) is implemented to extract features, and finally for the classification method, SVM is used on the two experiments [24].

In a home automation system, Brain-Computer Interface paradigm is used based on auditory Selection Attention. While home automation is very popular that provides the ability to control home device and equipment, the BCI could be a big addition to that technology, especially for people with disabilities, the overall system will have an addition to the quality of life for the people. In the project EEG signal is used, converted to analyze and then used as inputs to the home automation system. Many tasks were done successfully such as dimming and brightening the smart light bulb, and other commands. The tasks accuracy of turning on the light was 91.66%, for the dimming task an accuracy of 66.70% is achieved, to brighten the light 58.33% and to turn it off 50.00% [25].

A similar project that controls an avatar in a virtual environment is done using 3D VE to create the 3D environment using computer graphics, this software is used since the 70s for many purposes but not until recent years that BCI is being used with this software. However, the Unity engine is used with open source software Blender 3D to create the whole experience. For the BCI part of the system, BrainNet36 equipment is used to collect the signal at a 200hz sampling rate. The dataset was recorded from a right-handed man with the age of 30, performing five tasks. The subject was told to sit resting and then wait for the tasks, which involved a screen with arrows which navigated the subject to imagine movement of right and left hand, both feet and the imagination of rubbing a cube with both of the hands. The first task had an accuracy of approximately 93.75%, the second task had an accuracy of approximately 86.75%, the task with both feet had approximately 80.00% and the last one had 73.75% [26].

In a different project that used Motor Imagery (MI) Brain-Computer Interface; seven users volunteered in an experiment which signals recorded from their scalp with 10/10 EEG configuration which cover the sensorimotor region that is used in MI. after the pre-processing, deep learning classification is done on the dataset with a Multi-Layer fully connected Perceptron, which scored a 98.20% accuracy on the MNIST classification task [27].

In another study, a BCI system in virtual reality is implemented using a wearable headset that records EEG signals, a VR headset, and a visual interface. The tasks in this project are to rotate or scale objects in the VR using commands from the brain or facial expression. Unity3D software is used for the visualization of the virtual environment [28].

4.Implementation and Testing

In order to be familiar with the software required for the project, many online courses and demos from Lynda, YouTube, and the OpenVibe software were taken.

4.1.Unity 3D:

For the Unity software, “Unity 3D Essential Training” provided by Lynda was very beneficial to help start and advance in building the virtual environment needed for this project, it covered many important aspects and features of the Unity3D software, such as the setting up of the project, knowing how to use the unity interface, working with assets, applying materials on objects, the physics, lighting, audio, the post-processing and many more useful features that the software provides [18].

4.2.OpenVibe:

Regarding the OpenVibe software, tutorials can be found on OpenVibe’s website [29], however, changes had to be done for the purpose of the project. But the tutorials and forums were very helpful to understand the basics of the software and how to make it work. The tutorials together with the examples available in the software can show how the BCI system works starting from viewing the signal to training the classifier then using the system for the actual work [30].

4.3.Implementation:

The first step taken in the project after all the reading and background study in the proposal is developing a virtual 3D environment, after taking the online course on Lynda about the Unity3D software, Building a virtual environment became possible; however, a small simple environment was built with lights, walls and first-person perspective character that can move inside the environment and jump.

After that, OpenVibe was used to work on the BCI system, from the OpenVibe website a ten minute of a dataset with two features, moving left and moving right was downloaded, which is a start on developing the system.

Moreover, after reading the CSV file in the software, many steps were ahead. At first choosing the reference channel which is the first data set used, it was called “Ref_Nose” it subtracts a reference channel from every channel to remove the electromagnetic noise, then a

channel selector block is implemented which chooses which channel to use from all the sensors on the head, in our case the middle area will be used. After selecting the channels, a surface Laplacian block is used which is a spatial filter, that subtracts a channel from its surroundings and outputs the useful channels, the coefficients used in the beginning is to output two channels C3 and C4, later on more channels were added for better performance. The next step is to use a temporal filter which selects the frequency band, the method used is Butterworth with filter order 6, regarding the frequency band, many tests are done to try most of the frequency bands combined and separated.

Moreover, Time Based Epoching is needed, so looking at the dataset, the time for each epoch can be observed, and then written in the box of the Time Based Epoching. Then, simple digital signal processing is done, in this case, $x \times x$ was used which means that the energy of the signal was obtained, after we obtain that, signal averaging is done to have more stable information, after averaging the obtained energy of the signal another simple digital processing box was used but this time using $\log(1+x)$ which will result in the end with the power of the averaged energy of the signal. This obtained signal will be used as a feature for the system.

The feature aggregator box, checks the event ID in the dataset, the event ID has a unique number for each event happening such as left or right movement, however, the box in the software will create a matrix with left and right movement and every event ID will be read by the OpenVibe as a different direction.

All the boxes discussed earlier can be seen in Fig.3, plus the signal display which was used to observe the signal before and after the processing and feature extraction. You can see below the signal before and after processing.

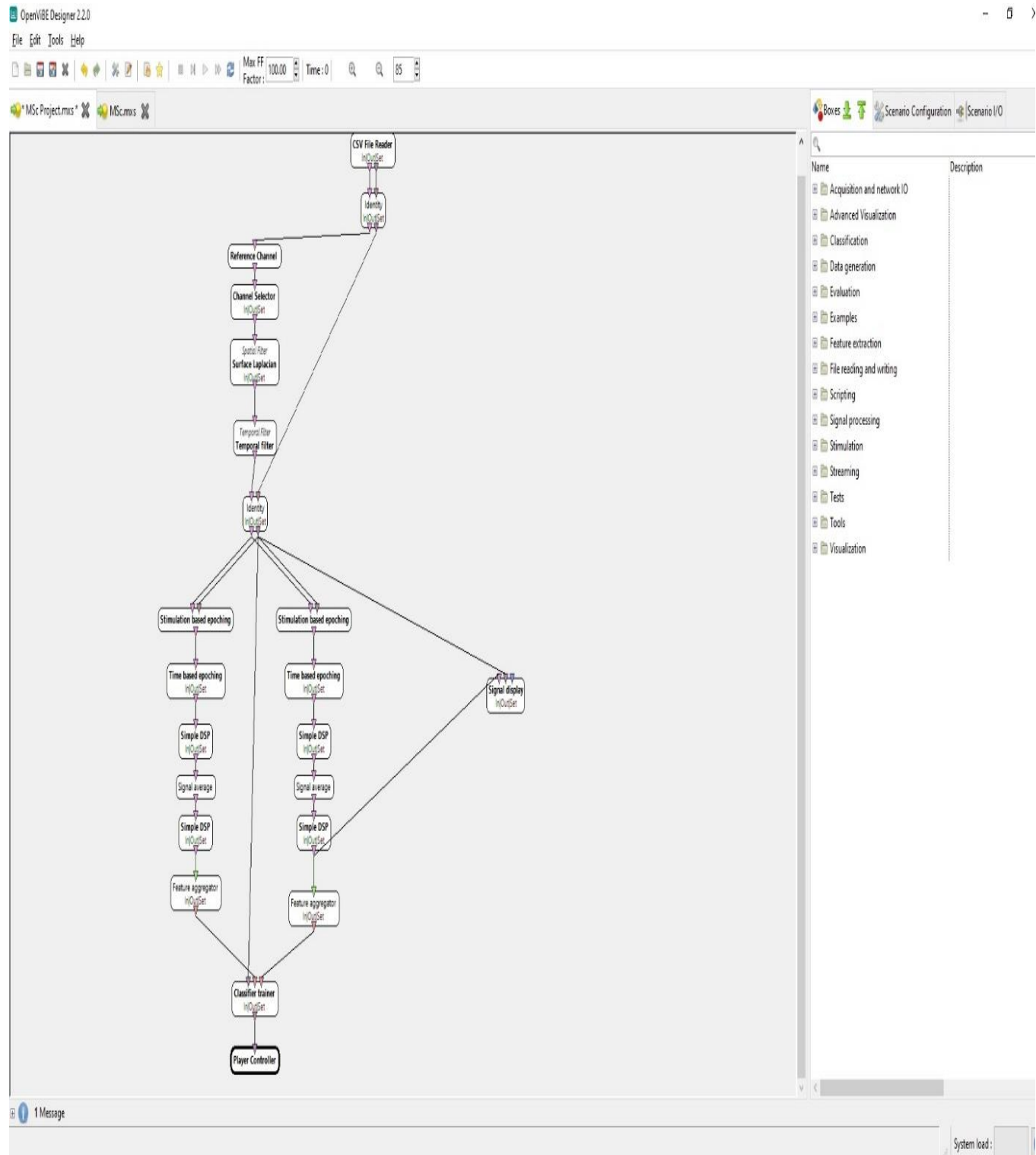


Figure 3 OpenVibe Software MI System

This in Fig.4 is the brain signal before processing and averaging and seen after the temporal filter. There are 8 filtered channels in this figure, which was chosen in the spatial filter configuration. The X-axis represents time, the amplitude of the signals is measured in voltage and the Y-axis is representing the channels

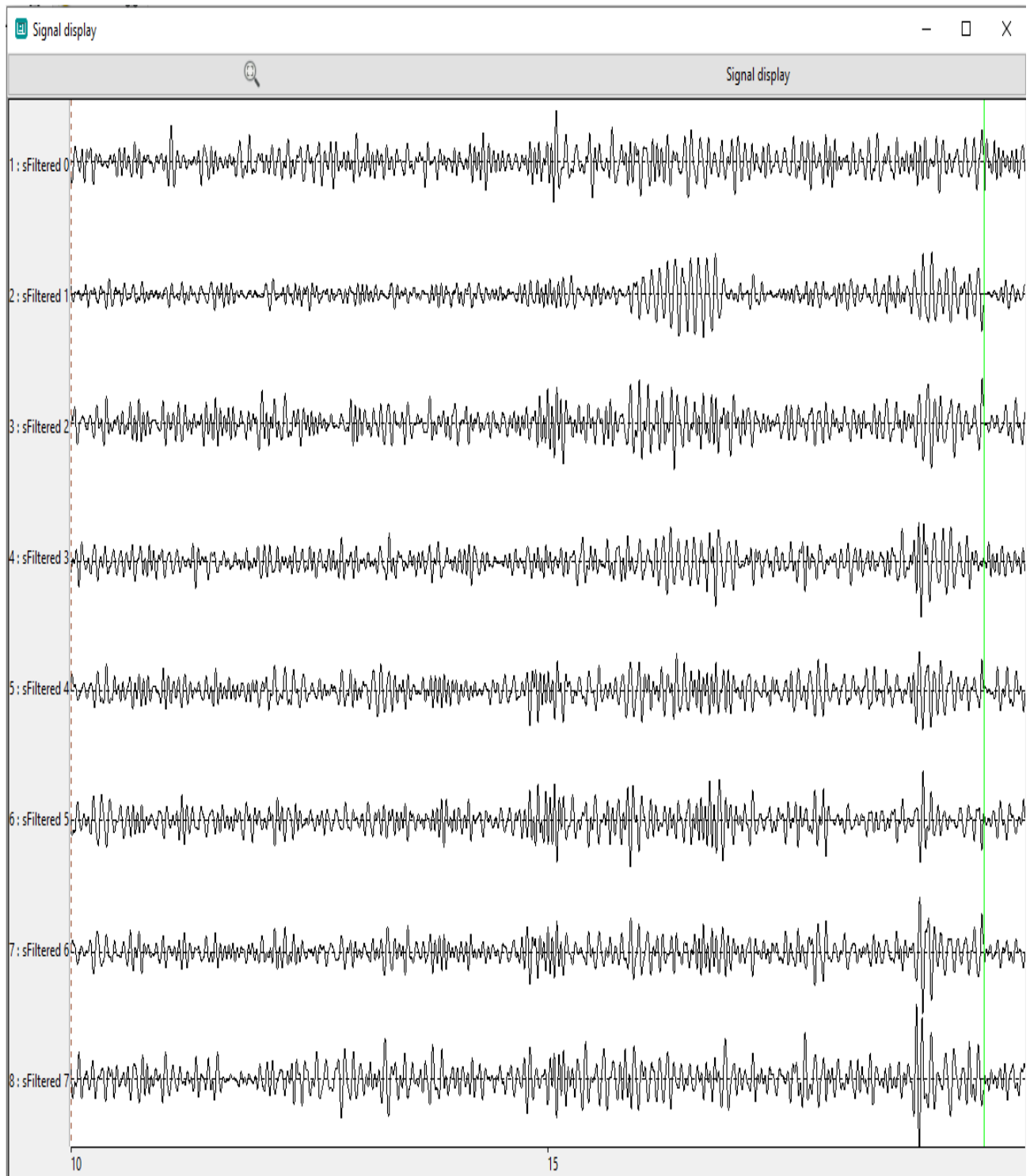


Figure 4 Signal before DSP

In Fig.5 what you can see is the brain signal after processing and feature extraction, seen after the second DSP box. The X-axis represents time, the signal amplitude is measured in voltage and the Y-axis represents the channels.

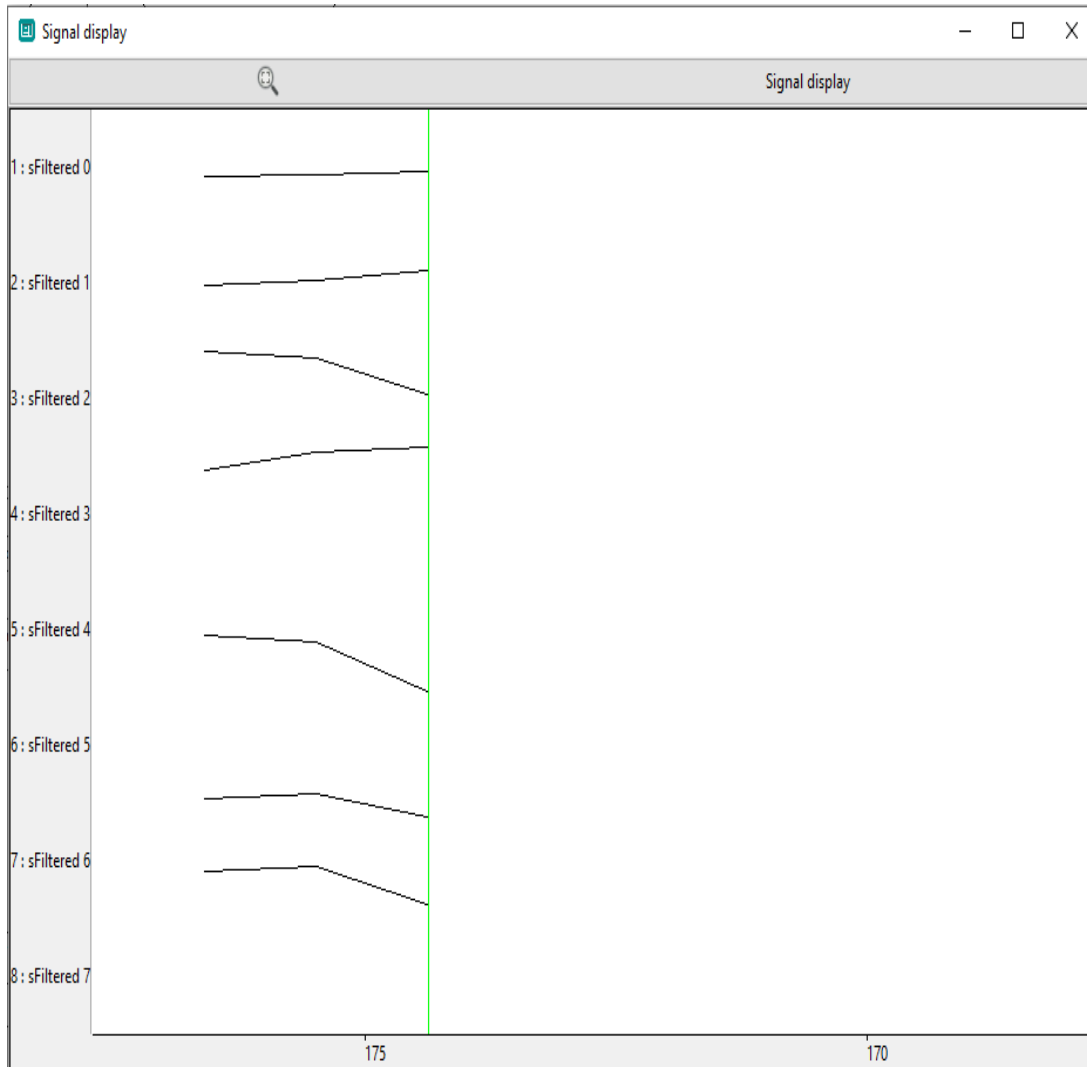


Figure 5 Signal after DSP

Furthermore, after the feature extraction a classifier trainer box is implemented, which has two classes, left the movement and right movement. The classifier trainer in the OpenVibe software has three algorithms to use, Linear Discriminant Analysis (LDA), Multi-Layer Perceptron, and Support Vector Machine (SVM).

However, to test the system many configurations were tried, and results of each configuration were written down then compared to check which configuration was the best for the system. Firstly, the frequency range was changed to test many frequency bands separated and combined. Since the Alpha plus Beta frequency range (8 – 24Hz) was the optimal frequency range, it was used in other different configuration, such as without a spatial filter and with

more output channels and different spatial filter coefficient, which was tried in a trial and error method.

After achieving an optimal configuration for the current system, a 10-k fold LDA and a 10-k fold SVM were tested using the Wilcoxon test which is also called Statistical Significance Test which is a non-parametric test that will determine if there is a statistical difference between these k-fold results. This test can be found online as a calculator [31], where the k values are the input and the needed output is a p-value. However, if the p-value is larger than 0.05 it means that there is no statistical difference between the two results, which in other words means that the result which had a better performance percentage is not actually better than the other result with lower percentage.

Moreover, Common Spatial Pattern Motor Imagery (CSP-MI) was also implemented and tested in the various configurations, this system requires training for the spatial filter, which is trained in a different design using the dataset that was downloaded from the OpenVibe software [30]. And then, this trained configuration is loaded in the classifier design and used as a spatial filter configuration.

In Fig.6, the design to train the CSP spatial filter can be seen. Below in figure seven, the classifier trainer design which utilizes the CSP spatial filter box is shown.

4. Implementation and Testing

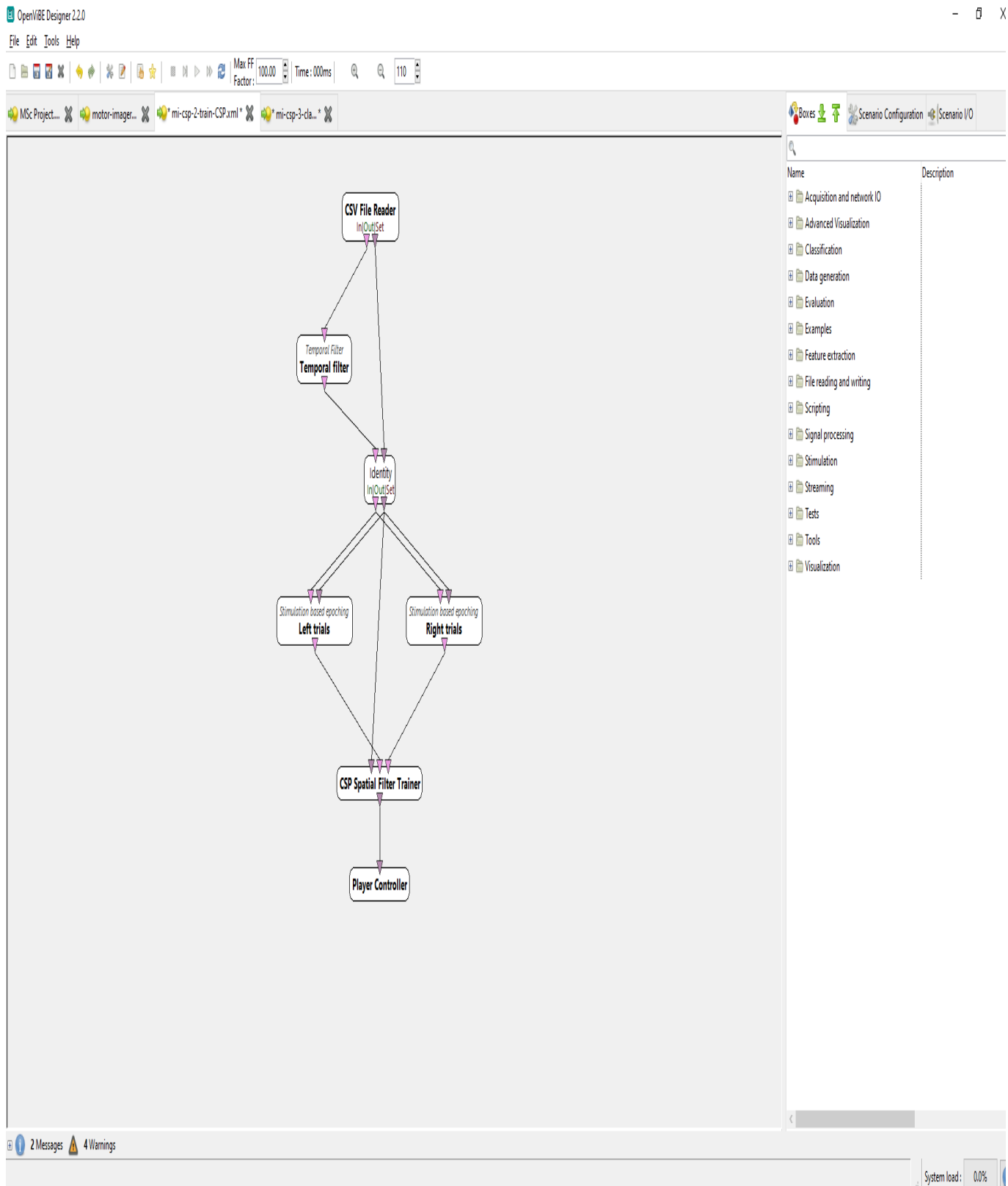


Figure 6 CSP MI Train.

In the CSP classifier design, which is seen in Fig.7, almost the same design as the normal MI design is used but without a reference channel and a channel selector, because these will be done automatically during the training of the CSP Spatial Filter in the previous design. However, the results acquired from this system was also tested using the Wilcoxon test mentioned before.

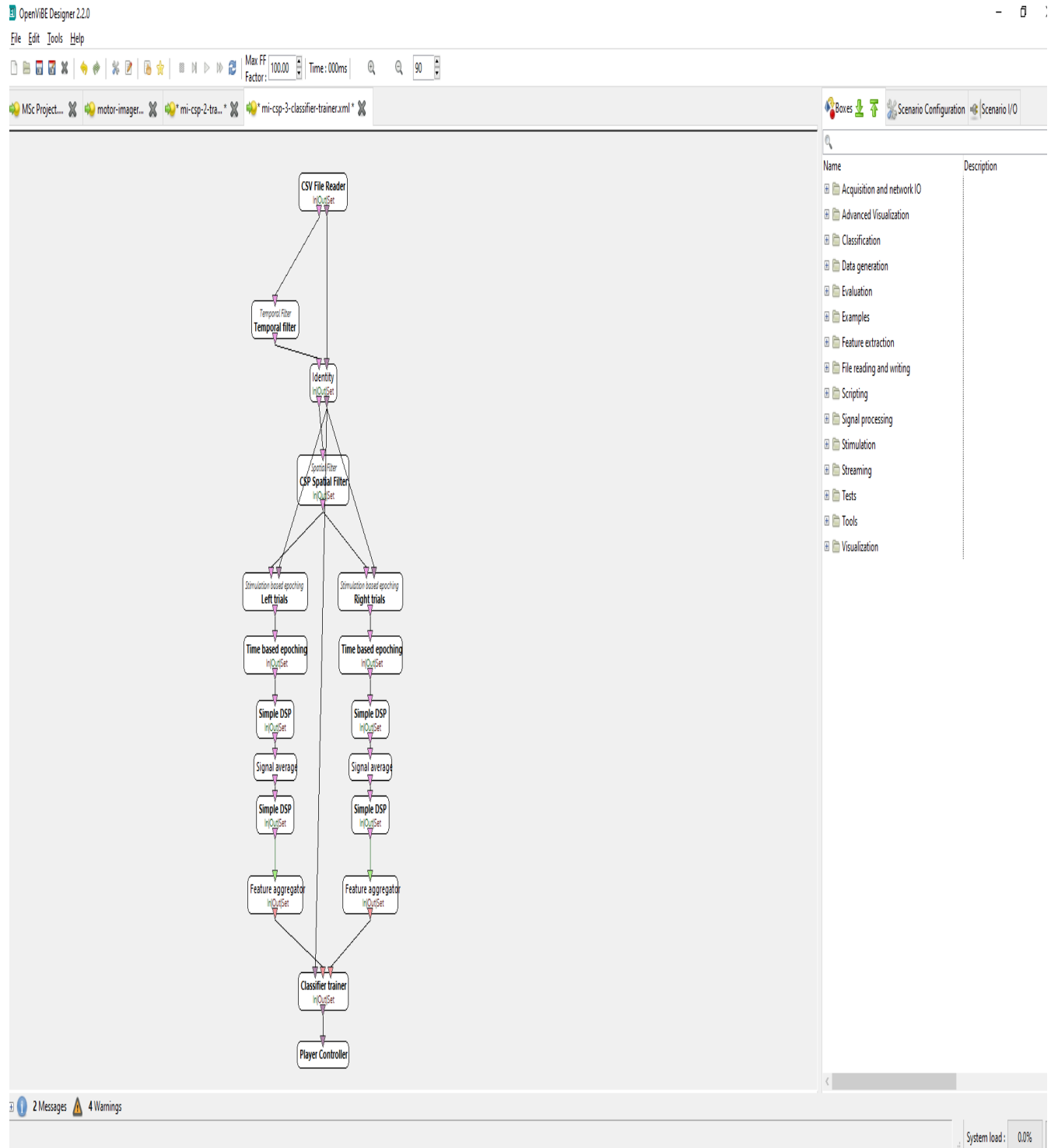


Figure 7 CSP Classifier trainer.

4.4. Recording EEG Signal

For the recording of the EEG signal, Enobio20G was used, which is a wireless electrophysiology sensor system that is able to record brain EEG signals, it is a cap that is placed on the scalp containing sensors [32].

Firstly, the Enobio equipment is connected to the laptop via Bluetooth, then using the software NIC the Enobio is connected to the laptop and the signals can be seen after synchronizing the input signals, during this time the cap is placed on the scalp with the sensors in the correct place.

However, when connecting the Enobio to the NIC software, a confirmation that a clear signal is being received is needed. As seen in Fig.8, at the right side of the figure, all the signals are turned red which means that there is no clear signal is being received.

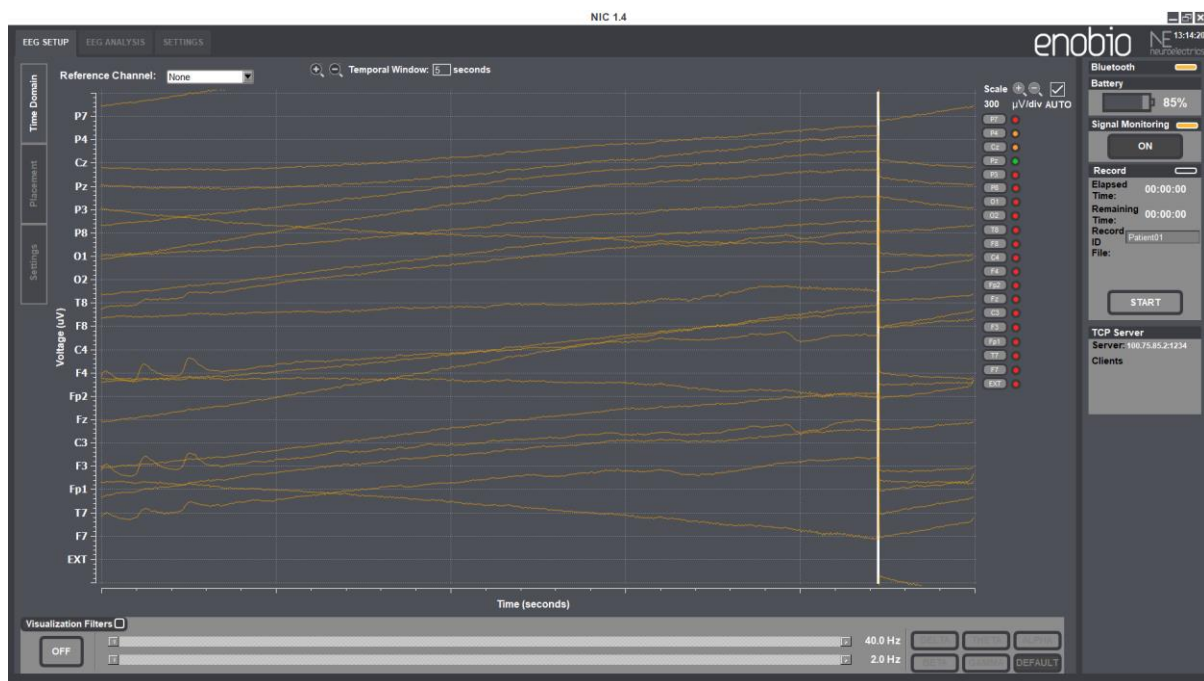


Figure 8 Enobio Connection part 1

To fix this issue, adjustment of the sensors that are placed on the scalp is required, and gel is needed on some specific sensors. As seen in Fig.9 all the lights turned green which means that a clear signal from those sensors is being sent.

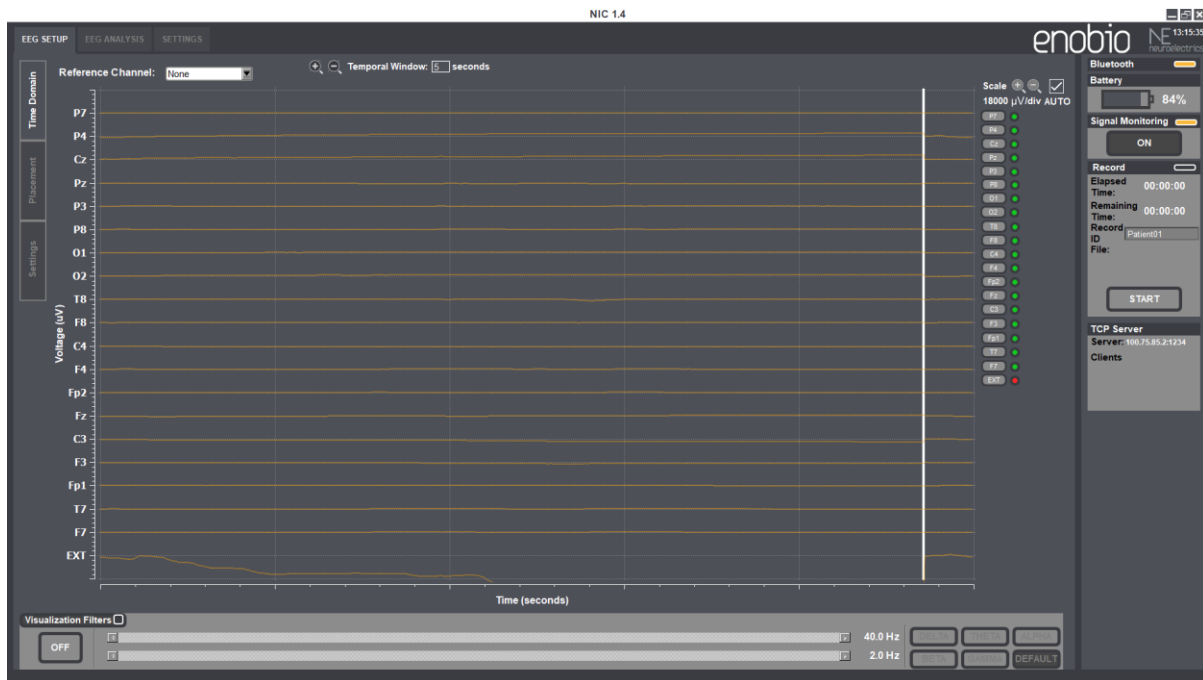


Figure 9: Enobio Connection part 2

After making sure that the signals are clear, signal monitoring is available in the software. Looking at Fig.10, a signal can be seen which was being recorded live. The X-axis refers to the frequency, and the Y-axis refers to the voltage. Looking at the Fig.10, a rise in the signal is seen and it is because of an eye blink. In the same Fig.10, another axis can be seen below the larger one, which is the same signal but in the time domain.

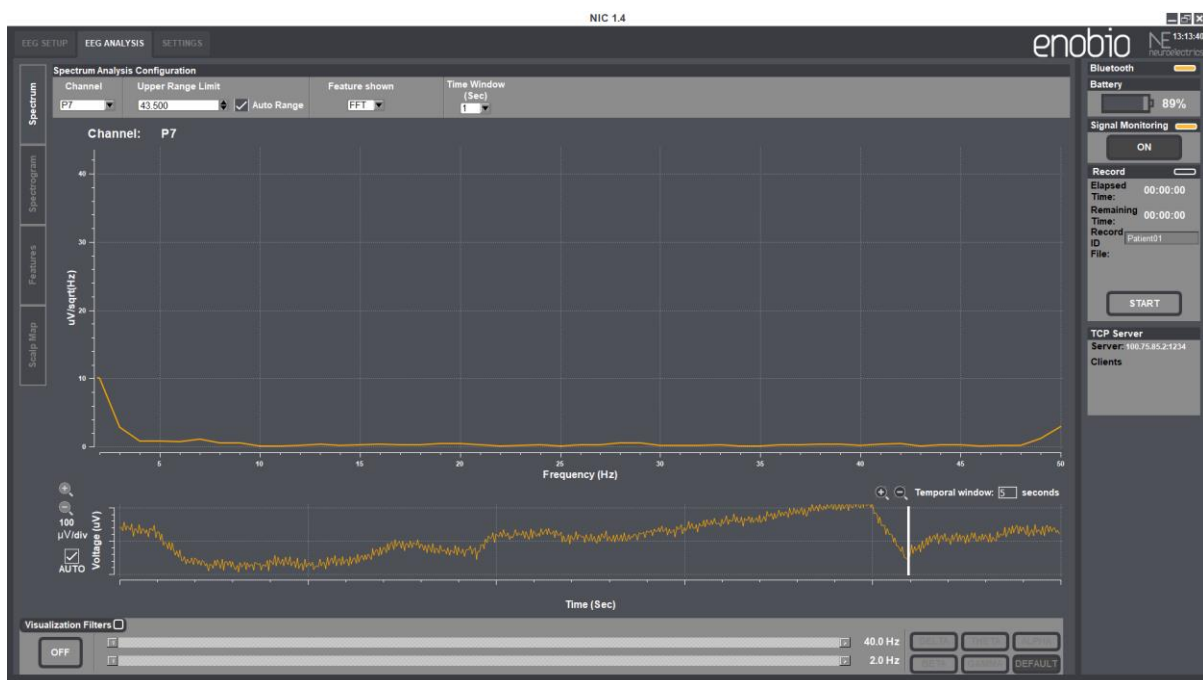


Figure 10: Signals Captured by Enobio

Moreover, after making sure that a clear signal is being received from the equipment, connecting it to OpenVibe is required to continue with the experiment, via the acquisition server, Enobio is connected to OpenVibe, then the recording is started with a simple design in OpenVibe which can be seen in Fig.11. The Graz Motor Imagery BCI Stimulator is the box that will generate the required stimulation, the Graz Visualization is the box that visualizes the stimulation, however, the signals are written using the Generic stream writer box and they are written in .ov file, this file will later be converted into CSV and trained on the system.

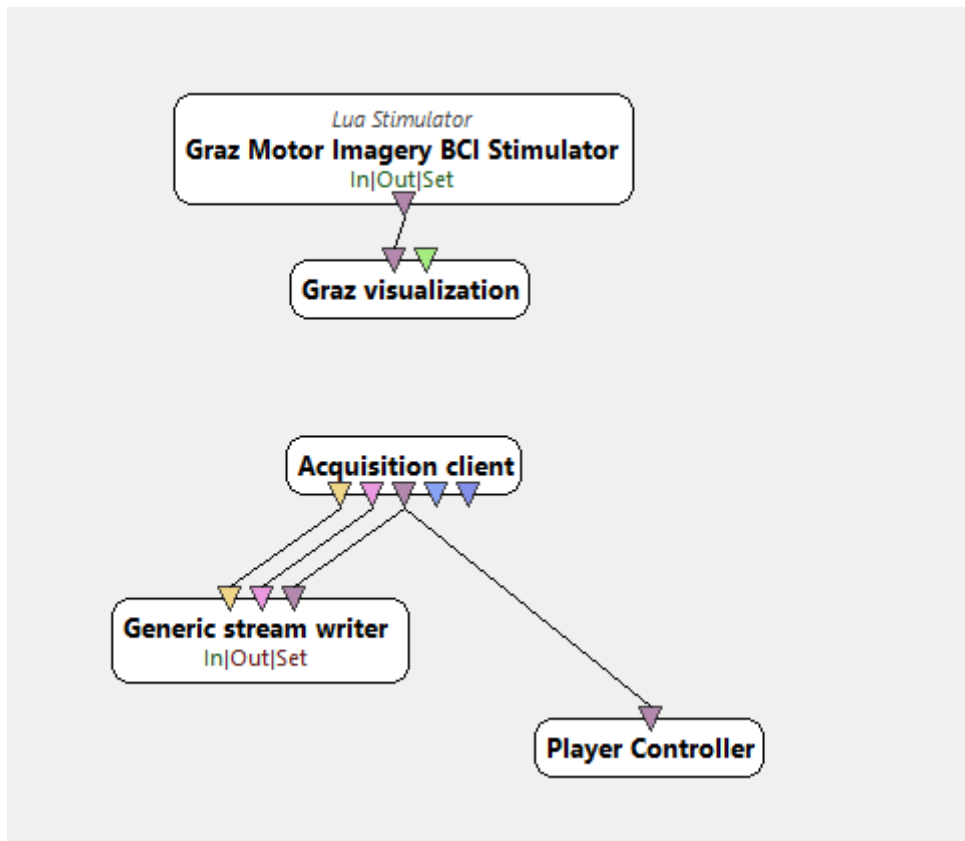


Figure 11: OpenVibe Design for Signal Recording

After obtaining a CSV file, a training of that file is to be done, however, a new configuration that suits this new dataset of recorded brain signals needs to be set. To be able to have the right settings for the dataset, a look into documentation is needed to be able to fix the dataset and make it suitable for training, this is done by editing the dataset and setting the correct name of the channels in the dataset and knowing which channel is the reference channel. [33]

Moreover, the reference channel in the design is set, and the channels are written in the channel selector, but a new surface Laplacian coefficient is needed. To be able to achieve the best coefficient for the Laplacian filter, a look into Fig.12 is required. Every green circle is a channel used, the Laplacian subtracts a channel from its surrounding, after looking at the

Fig.12 a new Laplacian coefficient is extracted and used, having 19 outputs for all the 19 channels. Multiplication of the number of channels surrounding the channel that is being worked is done on the latter channel, and then a subtraction of the surrounding channels is done. For example, looking at Fig.12 Cz channel is multiplied by four and then subtracted by C4, Fz, C3, and Pz.

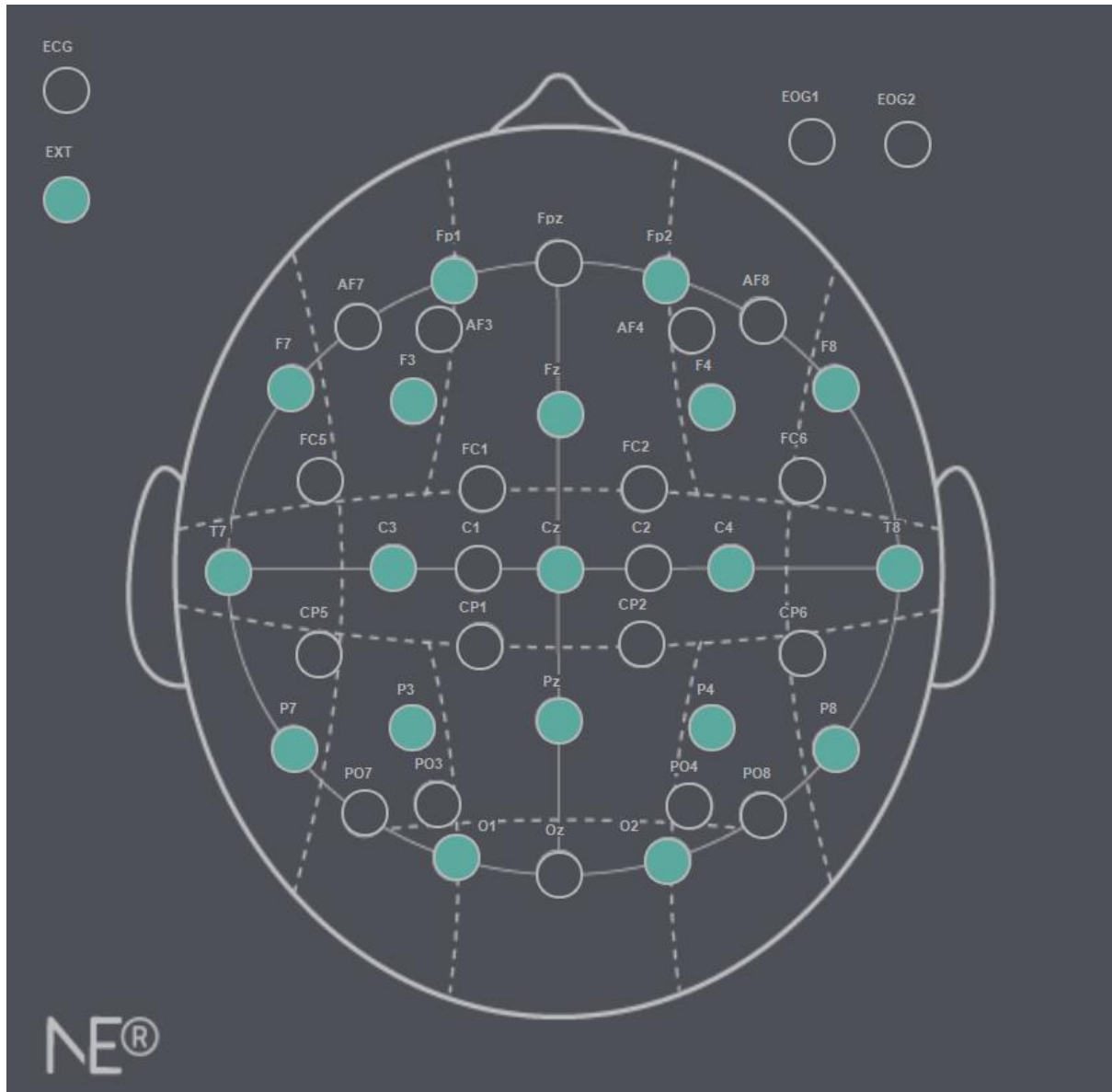


Figure 12: Placement of the Sensors

After configuring the Laplacian filter, the rest of the boxes needs to be configured based on the new dataset.

4.5. The Dataset:

The dataset used to train the system is downloaded from the OpenVibe website with a CSV format [30]. It is a dataset of a pre-recorded signals of a person thinking either left-hand movement or right-hand movement depending on what was asked from the person. It has ten channels C3, C4, FC3, C5, C1, C2, C6, CP3 and CP4, as well as a reference channels called Ref_Nose which makes a total of eleven channels, a time for every 0.001953 seconds, one epoch for every 0.0625 seconds and an event ID for every event occurring during a certain time.

The dataset contains fourteen records of right and left-hand imagination, the channels have been recorded in common average mode and Ref_Nose is to be used as reference when needed. [30]

Each dataset that is available on the website is made of forty trials, asking the user to imagine either right or left-hand movement, twenty imagination of each direction. The experiment followed the protocol of Graz University protocol. The signals were recorded on three different days in the same month. The dataset used from the website is called signal which has the raw signal of the experiment. [30]

4.6. Virtual Environment using Unity3D:

After getting familiar with the software, following the online courses; a small virtual environment suitable for testing of the BCI system must be built. The Unity3D software offers many prefabs online, some of them are free to purchase and some are not. However, after searching on the online asset store, house building tools were found and implemented in the software, after placing the house, a first-person character was implemented inside the house with the ability of moving forward, left, right, backward and jumping, another thing that had to be taken into consideration is the physics inside the house which is not built-in with the prefabs, so collisions for the character and the walls were created, in a way that the character cannot pass through the walls.

After testing on this simple environment, a more suitable environment for disabled people will be built manually together with simple functionalities to use inside the house.

Moreover, in order to test the system, a way to connect Unity3D with the OpenVibe software is to be found, there is various way to do this, but in this project Lab Streaming Layer (LSL) will be used which is downloadable for Unity3D and is built in the OpenVibe software.

4.7. Connecting OpenVibe and Unity3D:

In order to make Unity3D's character to move according to the brain signals commands from OpenVibe, data had to be sent from one software to another. There are many ways to send data from OpenVibe to external software, such as TCP/IP, Lab Streaming Layer (LSL), writing and reading from a text file or many other ways that can be found on the OpenVibe forums [32]. In this project, LSL was chosen to be the communicator between the software. The communicator opens a stream which the other software will listen to using an internal LSL addon. However, each software needs to be configured accordingly for it to be able to receive the correct stream and to utilize the data sent to Unity3D, scripts had to be edited.

4.8. OpenVibe Configuration:

After training the classifier and saving the chosen configuration that will be used in the system, another design needs to be created in order for the Brain-Computer Interface Motor Imagery system to be online and working.

Moreover, the design is a combination of boxes built-in the OpenVibe software, starting from reading a dataset to sending data to Unity3D via LSL. The dataset used in this design is different from the dataset used to train the classifier but noting that it has the same channels, reference channel, and structure, it is also downloaded from the OpenVibe website [30], however as seen in the Fig.13, a Reference Channel and Channel Selector box are used to choose the Reference channel and the rest of the channels from the dataset, then a Surface Laplacian box is implemented with the same settings and coefficients as the one used in the classifier design, as well as the rest of the boxes shown in the Fig.14 until the Classifier processor box which chooses the configuration that was saved during the classifier trainer design, This box will output a streamed matrix output to a simple DSP box that will transform the output into an equation of the form "x", and then will be sent to the Matrix Transpose box and finally to the LSL export (Gipsa) and matrix display box.

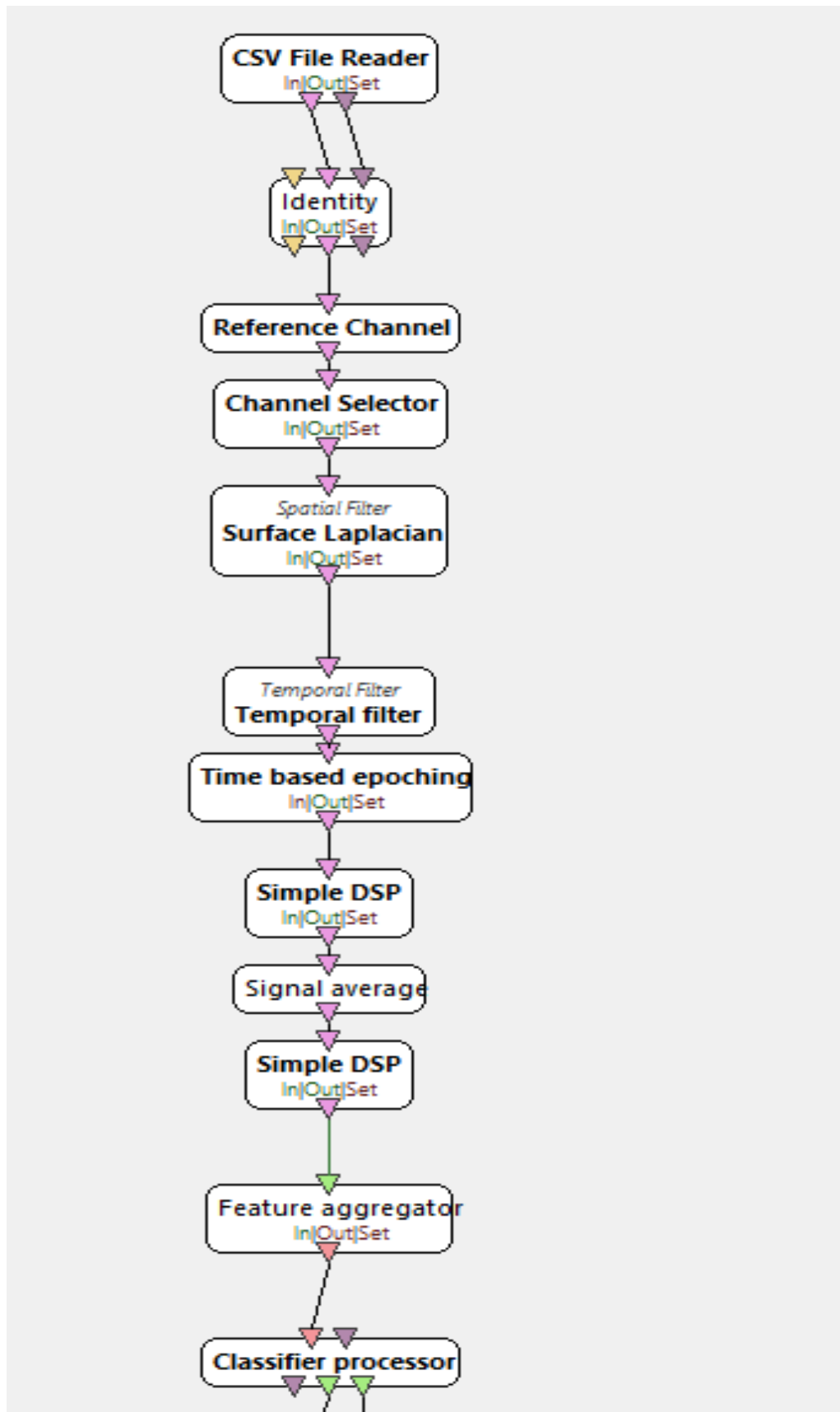


Figure 13 BCI MI Online Design Part 1

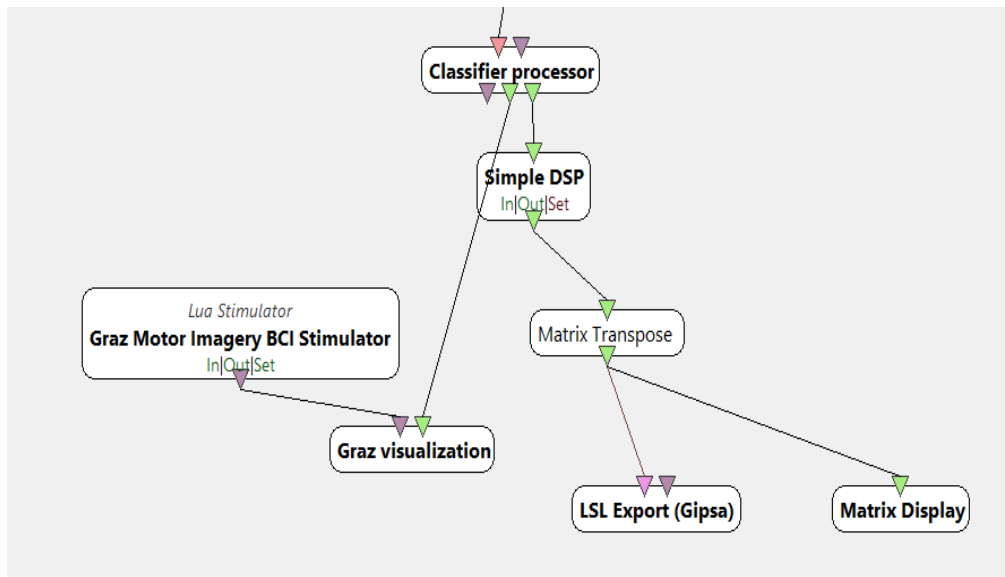


Figure 14 BCI MI Online Design Part 2

The Graz Motor Imagery BCI Stimulator and the Graz Visualization are used to visualize arrows that will indicate which direction is being stimulated at the instance as shown in Fig.15. And the matrix display box will display the probabilities of each direction, the feature with the highest probability is the feature that is being stimulated, see Fig.16, and finally the LSL Export box will open a stream for the external software to listen to, as seen in the Fig.17 a name for the stream needs to be set and used in the external software to be able to receive the same stream.

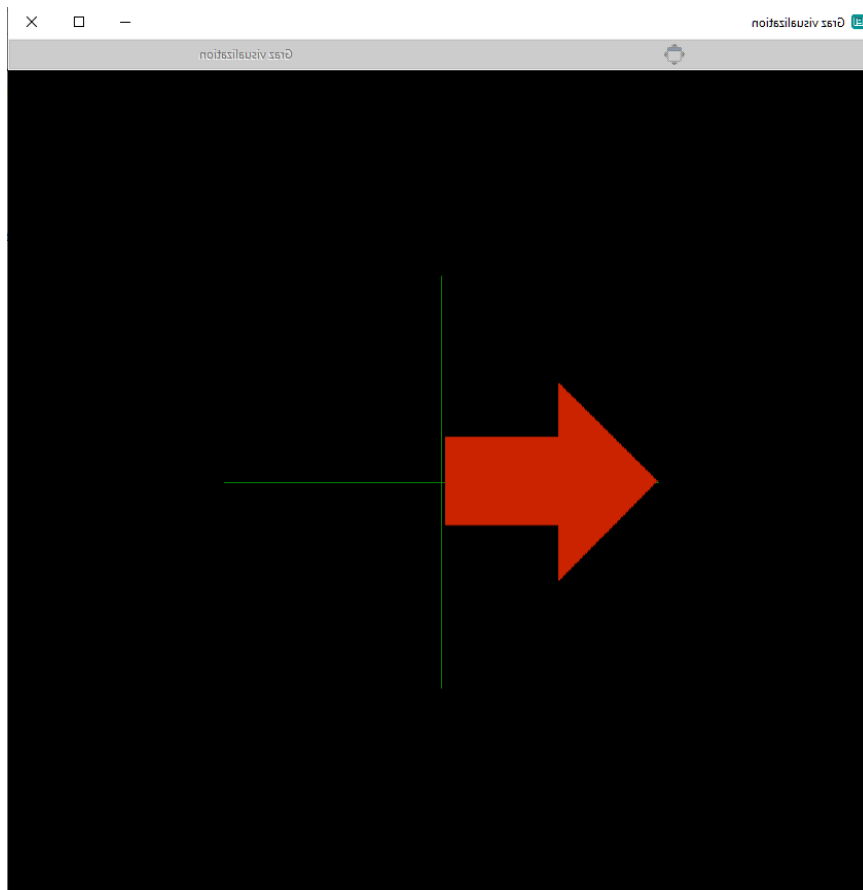


Figure 15 Graz Visualization

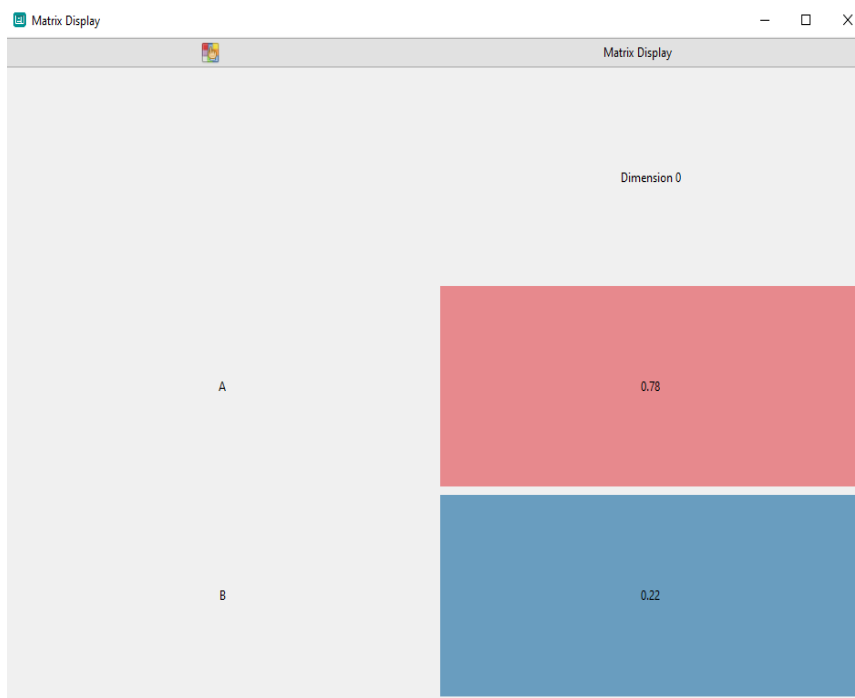


Figure 16 Matrix Display

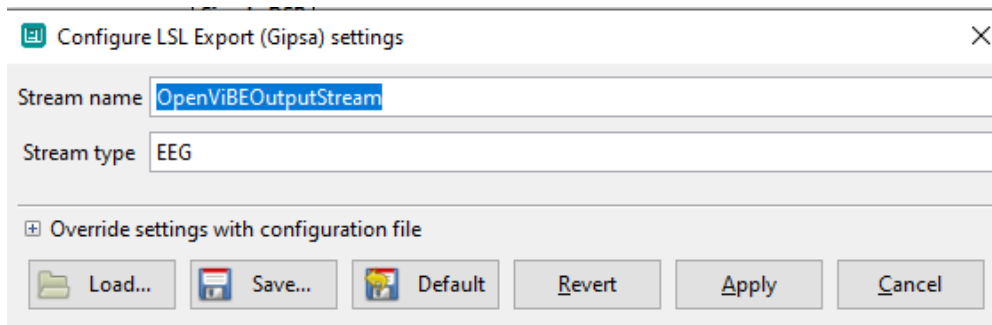


Figure 17 LSL Export Configuration Box.

4.9. Unity3D Configuration and Script Editing:

For the Unity3D software, an external addon has to be downloaded, which is called LSL4Unity, it can be found on GitHub [33]. After downloading the addon, it is dragged and dropped in the Unity3D project and can be used after editing the scripts according to what is needed to be done with it.

Firstly, inside the LSL4Unity there is a script called ExampleFloatInlet which needs to be edited. As seen in the Fig.18, a float called lastSample is added which will be the container for the values that will be sent from OpenVibe.

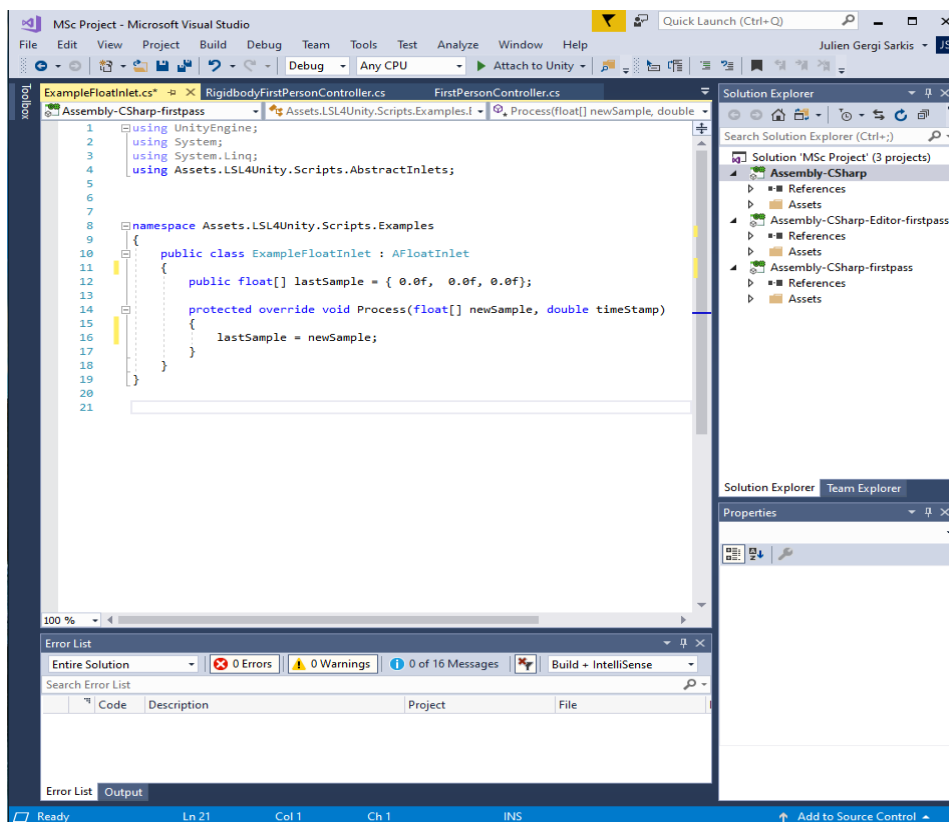


Figure 18 Unity Script for LSL.

Then this script will be added to the game object in unity, in this case it is the FPSController which is the controller of the first-person perspective character, however, after adding the script to the FPSController, the name of the stream that is written in the LSL Export box in OpenVibe needs to be written inside the script box in Unity as seen in the right side of the Fig.19, as well as the stream type.

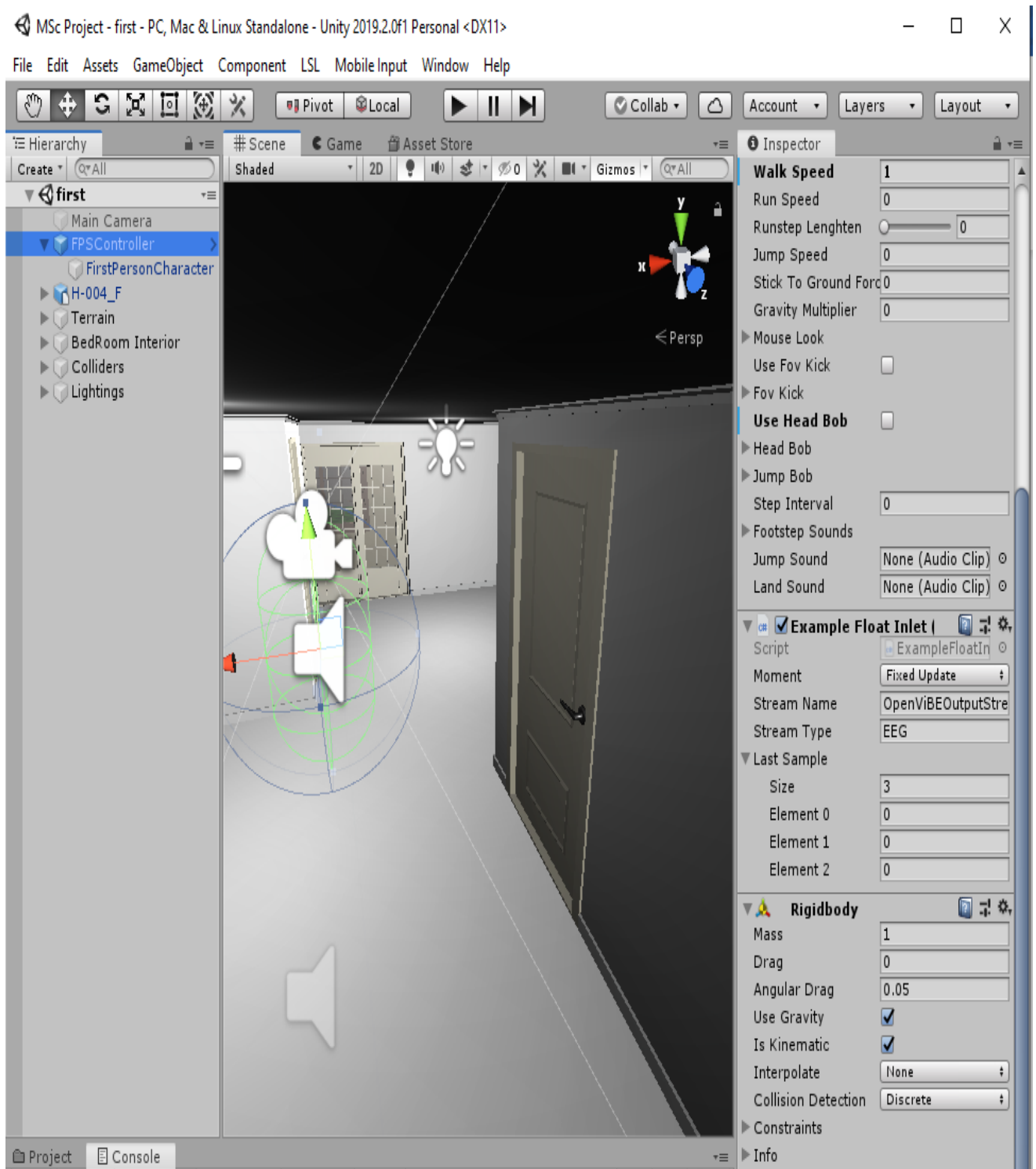


Figure 19 Unity Stream Name and Type

Moreover, another script needs to be edited which is the first-person controller script, which originally was made for moving the character using keyboard functions, in this case, the character needs to be moving using brain signals. Firstly, the LSL stream needs to be included and added to the code, then in the start() function “inlet = FindObjectOfType<ExampleFloatInlet>();” is added. In the fixed update function, the code is edited to able to use the data that is being received from the stream for the purpose of moving the character, the code can be seen in Fig.20.

```
private void FixedUpdate()
{
    float speed;
    GetInput(out speed);
    // always move along the camera forward as it is the direction that it being aimed at
    Vector3 desiredMove = transform.right*(m_Input.y- m_Input.x);

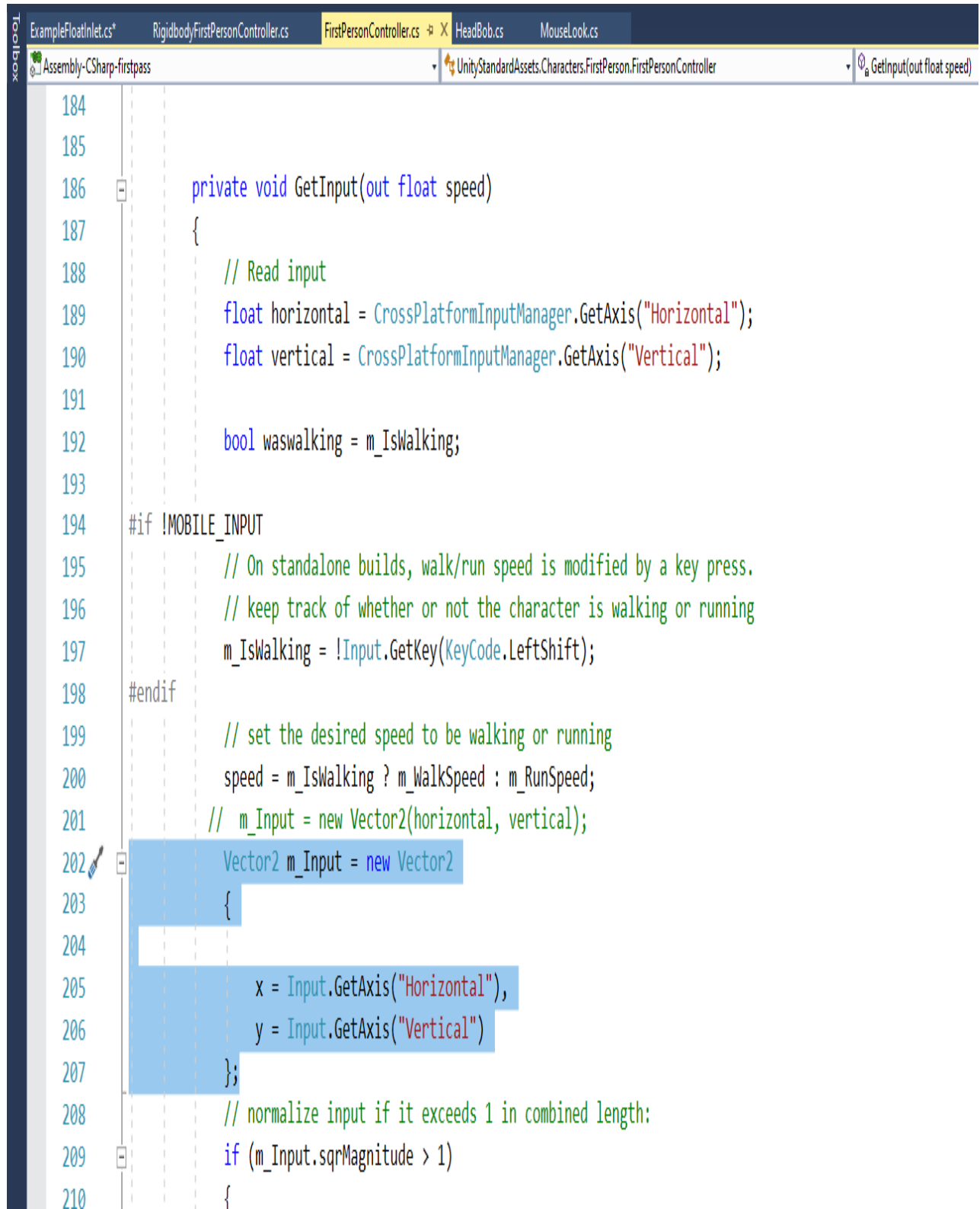
    // get a normal for the surface that is being touched to move along it
    RaycastHit hitInfo;
    Physics.SphereCast(transform.position, m_CharacterController.radius, Vector3.down, out hitInfo,
        m_CharacterController.height/2f, Physics.AllLayers, QueryTriggerInteraction.Ignore);
    desiredMove = Vector3.ProjectOnPlane(desiredMove, hitInfo.normal).normalized;

    if (inlet.lastSample.Length > 0)
    {
        m_Input.x = inlet.lastSample[0];
    }
    if (inlet.lastSample.Length > 0)
    {
        m_Input.y = inlet.lastSample[1];
    }

    m_MoveDir.x = desiredMove.x*speed;
    m_MoveDir.z = desiredMove.z*speed;
```

Figure 20 Fixed Update Code for The Movement of The Character.

However, the inputs that are used in this function are added in a different function, which is the `GetInput` function. The input that we need is added as a 2D vector as seen highlighted in Fig.21.



```

184
185
186     private void GetInput(out float speed)
187     {
188         // Read input
189         float horizontal = CrossPlatformInputManager.GetAxis("Horizontal");
190         float vertical = CrossPlatformInputManager.GetAxis("Vertical");
191
192         bool waswalking = m_IsWalking;
193
194     #if !MOBILE_INPUT
195         // On standalone builds, walk/run speed is modified by a key press.
196         // keep track of whether or not the character is walking or running
197         m_IsWalking = !Input.GetKey(KeyCode.LeftShift);
198     #endif
199
200         // set the desired speed to be walking or running
201         speed = m_IsWalking ? m_WalkSpeed : m_RunSpeed;
202         // m_Input = new Vector2(horizontal, vertical);
203         Vector2 m_Input = new Vector2
204         {
205             x = Input.GetAxis("Horizontal"),
206             y = Input.GetAxis("Vertical")
207         };
208
209         // normalize input if it exceeds 1 in combined length:
210         if (m_Input.sqrMagnitude > 1)
211         {

```

Figure 21 `GetInput` Function.

Looking again at Fig.20, there is a line in the code “Vector3 desiredMove = transform.right*(m_Input.y – m_Input.x);”, this line translates the movement that is required to happen. Basically, two values are being received from the OpenVibe LSL stream, and these values are probability of each direction (left or right), probability A (left) is input.x, and probability B (right) is input.y, the function here is transform.right so when subtracting the two probabilities, a positive value is needed for the character to move right and a negative value for the other direction. That is why a subtraction of probability right minus probability left is done in the code, in a simpler form, if probability B is less than probability A then move left and if probability B is larger than probability A then move right.

4.10. Challenges and Solutions:

While attempting to reach the final system, having a BCI design that connects to Unity3D software and moves a character depending on the classification of the two classes in OpenVibe, many challenges and obstacles were encountered and fixed.

Moreover, the biggest challenge encountered in this project is to connect Unity3D software with OpenVibe via LSL, especially that there are no tutorials about it online, so pieces of information were collected separately and then joined together to make the project work. In the beginning, an acquisition server was tried to connect the two software, however, the connection was successful through the server, but the system confused the input and the output.

With editing in the script of OpenVibe, the right connection of inputs and outputs was connected in the design, which made it possible to get an output from OpenVibe, that will be an input in Unity3D using LSL export in OpenVibe and LSL4Unity in Unity3D. in the end the values of the probability of each class were streamed from one software to another and could be read in the Unity3D software.

However, the correct way to connect Unity3D and OpenVibe is shown simply in Fig.22. When information enters the LSL export box in the OpenVibe software, the latter opens a stream with the values that entered the box, however, the stream is to be used in the Unity3D software. Unity3D reads the stream through an addon called LSL4Unity which

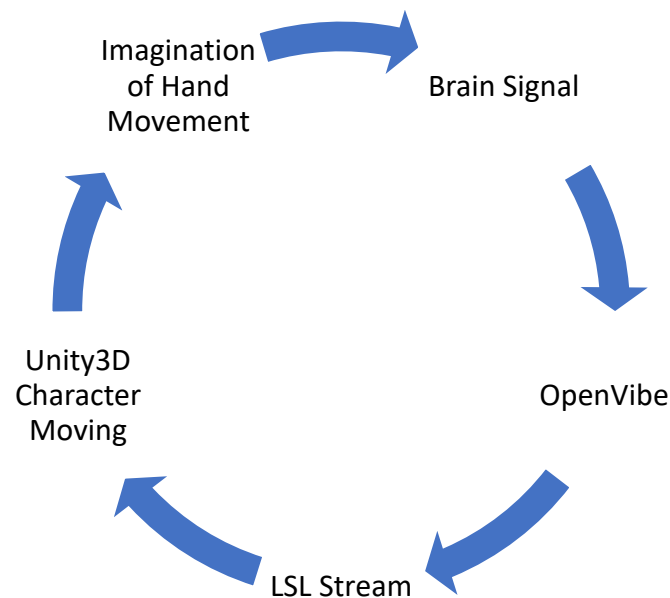


Figure 22 Communication process between OpenVibe and Unity3D

Another challenge encountered in the project is the editing of the character controller in Unity3D, the scripts used in Unity3D are poor commented which made it harder to understand every function in the script and to know which function is controlling the direction of the character. However, after analyzing the script and doing trials on it, editing the script to be moving left and right using the brain signals in the OpenVibe software was successful.

5.Evaluation and Discussion:

For the evaluation of the system, more than ten configurations were set and tested, and their result can be seen in the tables below. To evaluate, cross-validation accuracy results are shown on the three machine learning algorithms used in this project.

In each configuration, some settings regarding the signal processing, the method for the BCI and other settings were modified and tested for a clear evaluation of what improves the system and what decreases the performance of the system.

However, many results were extracted from all those configurations and the numbers are not similar in each configuration, which is why a Wilcoxon rank-sum test and confusion matrix is done on the results of the configuration that had the best potential to be the final configuration.

5.1.Machine learning algorithms:

In this project, three machine learning algorithms were used to classify the data, into two classes: Left or Right. Linear Discriminant Analysis (LDA), Multi-Layer Perceptron and Support Vector Machine are the machine learning techniques used.

5.1.1. Linear Discriminant Analysis (LDA):

Discriminant Analysis is a popular algorithm for the classification, it is used in different fields such as statistics, machine learning, and pattern recognition. It searches for the combination of features that is different from a number of classes. It belongs to the group of parametric classification, which has assumptions such as that the data has a certain distribution [34].

Linear Discriminant Analysis (LDA) is a common algorithm that is used in the extraction of features and the reduction of dimension. It is used in many applications such as face recognition, image retrieval, and many other applications. One of the limitations of the LDA is that it relies on the well estimated non-singular covariance matrices [35].

5.1.2. Multi-Layer Perceptron:

Multi-Layer Perceptron is neural network belonging to Artificial Neural Network (ANN) which is a strong process to model data, it has the ability to capture and deal with complicated input and output relationships [36].

In general, the ANN has the ability to learn the relationship between the input and the output which is usually hidden, it is contained in a distributed way and has known features [37].

5.1.3. Support Vector Machine:

SVM is one the most popular and most used supervised machine learning method for the classification which is because of the good performance and the range of application [38].

The Support Vector Machine (SVM) can be used when all the information has a total of two classes. This algorithm classifies the data by looking for perfect hyperplane that will isolate the data purposes of a class from another. This method is known to be an economical strategy, it is successful because of the good performance in the ability of generalization. It has shown good potential in Binary Classification and other classification [39].

5.2. Confusion Matrix:

A confusion matrix is used as a standard way to show the thematic accuracy of sensed information. A common way of doing the error matrix is by means of global indexed which is like the overall accuracy or the Kappa coefficient. A confusion matrix is a tool for evaluating data; it is a statistical tool for analyzing paired findings. The contents of a confusion matrix a set of numbers that belongs to a degree of similarity between the classes. Moreover, the confusion matrix is made from a control sampling on the data and a response to the design. [40]

5.3. Wilcoxon Rank-Sum Test:

The Wilcoxon ran-sum test is a non-parametric test [41] which is applied to evaluate if the results are statistically significant or just different by luck or chance. The Wilcoxon test is one of the most common hypothesis tests to know if two groups are equally distributed. It is commonly used in medicine and in other fields [42].

In order to know if the results are statistically significant or not, a calculation of the P-value is required, which can be done using t-test or Wilcoxon-Mann-Whitney test. These tests can be done if the two groups are of equal distributions. However, depending on the p-value, the two results can be marked as statistically significant or not; if the p-value is larger than 0.05 than the results is different by chance [42] .

There are many hypothesis testing to evaluate if results are found by luck or not, such as the t-test which is a popular way of knowing if the results are statistically different or not but the difference between the other tests and the Wilcoxon Rank-Sum test does not assume that the

results are normally distributed. And this is why it can be used in biological researches, basically because results that are extracted from biological source such as the brain, has a different system and pattern than other fields. [42]

To summarize, In the Wilcoxon Rank-Sum test, the p-value is evaluated by the calculation of the rank concerning the two samples, which is comparing the rank of all the possible permutations of the samples. [42]

5.4. Results:

After setting up the system, test runs are done to able to see the results and improve the system. Many configurations are tested, and results are shown in tables for each configuration.

Configuration 1:

As for every system, performance should be done to evaluate the results and improve the system whenever possible. For this system, many configurations were done, and then tested; the first configuration had a spatial filter coefficient of “4;0;-1;0;-1;-1;0;0;-1;0;0;4;0;-1;0;0;-1;-1;0;-1” with a two outputs and ten inputs, the filter method is set as Butterworth with a filter type of Band Pass having a filter order of 6. Now for the first few configurations, only the frequency range was varied, the first configuration was the default one which had a low-cut frequency of 8Hz and a high-cut frequency of 24Hz which is the alpha (8-12Hz) combined the Beta(13-24Hz). However, three classifier algorithms were tested and evaluated using cross-validation.

The first algorithm was Linear Discriminant Analysis (LDA) and by varying the K number in each test run starting from k=3 to k=10, the test accuracy result varied from 69.5918% to 73.4694%, eight results in total. Results can be seen in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	70.00	77.09
4	73.46	77.09
5	70.81	77.09
6	73.10	77.09
7	72.85	77.09
8	70.35	77.09
9	72.13	77.09
10	69.59	77.09

Table 1 LDA accuracy results for configuration 1

The second algorithm was Multi-Layer Perceptron, with varying the K number from k=3 to k=5, and by varying with the number of neurons in the hidden layer, using 30,40 and 50 neurons the test accuracy results varied from 63.9898% to 70.6100%, nine results in total. Results can be seen in the table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
4	40	0.0001	0.1	65.71	75.51
4	30	0.0001	0.1	66.78	76.22
4	50	0.0001	0.1	70.61	76.42
3	50	0.0001	0.1	69.13	76.47
3	40	0.0001	0.1	69.24	75.91
3	30	0.0001	0.1	63.98	76.83
5	30	0.0001	0.1	66.98	75.81
5	40	0.0001	0.1	66.88	76.02
5	50	0.0001	0.1	67.09	76.22

Table 2 Multi-Layer Perceptron accuracy results using configuration 1

The third and final algorithm used was Support Vector Machine (SVM), by changing the k number from three to four, changing the SVM type to either C-SVC and NU-SVC, and

changing the kernel type from linear, polynomial, radial basis function and Sigmoid the accuracy-test results varied from 38.92% to 73.6224%, having fifteen results in total. Results can be seen in the table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	71.89	77.29
4	C-SVC	Linear	3	73.62	77.29
5	C-SVC	Linear	3	72.39	77.29
3	NU-SVC	Linear	3	54.48	49.28
4	NU-SVC	Linear	3	55.35	66.17
5	NU-SVC	Linear	3	63.46	66.07
3	C-SVC	Polynomial	3	72.09	76.17
4	C-SVC	Polynomial	3	72.55	76.17
5	C-SVC	Polynomial	3	72.80	76.12
3	C-SVC	Radial basis function	3	68.88	78.06
4	C-SVC	Radial basis function	3	72.85	78.06
5	C-SVC	Radial basis function	3	69.18	78.06
3	C-SVC	Sigmoid	3	54.24	58.97
4	C-SVC	Sigmoid	3	43.97	59.03
5	C-SVC	Sigmoid	3	38.92	58.97

Table 3 SVM accuracy test results using configuration 1.

Configuration 2:

The second configuration, having the same spatial filter coefficient, number of outputs, number of inputs and the same temporal filter configuration, just with a different frequency range, this configuration has a low-cut frequency of 2Hz and high-cut frequency of 40Hz.

Using the LDA algorithms, and by varying the K number, the accuracy-test results varied from 69.5918% to 73.4694%. Results can be seen in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	56.33	70.25
4	64.69	70.25
5	58.26	70.25
6	62.45	70.25
7	60.61	70.25
8	60.96	70.25
9	63.26	70.25
10	57.29	70.22

Table 4 LDA accuracy results using configuration 2.

Using the Multi-Layer Perceptron, and varying the same settings as configuration, the test accuracy results varied from 45.6633% to 54.7957%. Results can be seen in the table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	51.43	57.04
3	40	0.0001	0.1	54.79	68.26
3	50	0.0001	0.1	49.54	68.46
4	30	0.0001	0.1	48.57	68.41
4	40	0.0001	0.1	51.02	68.31
4	50	0.0001	0.1	45.66	46.22
5	30	0.0001	0.1	54.03	67.50
5	40	0.0001	0.1	54.23	56.88
5	50	0.0001	0.1	52.34	68.41

Table 5 Multi-Layer Perceptron accuracy test results using configuration 2.

Using the SVM classifier, with the same variations as the previous configuration, the accuracy-test results changed from 39.7959% to 64.6939%. A table with the results can be seen below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	57.25	70.30
4	C-SVC	Linear	3	64.64	70.15
5	C-SVC	Linear	3	64.69	70.20
3	NU-SVC	Linear	3	56.33	50.00
4	NU-SVC	Linear	3	40.15	50.00
5	NU-SVC	Linear	3	40.00	50.00
3	C-SVC	Polynomial	3	59.27	70.30
4	C-SVC	Polynomial	3	63.82	70.30
5	C-SVC	Polynomial	3	62.39	70.35
3	C-SVC	Radial basis function	3	52.71	70.15
4	C-SVC	Radial basis function	3	60.25	70.15
5	C-SVC	Radial basis function	3	53.87	70.20
3	C-SVC	Sigmoid	3	43.31	50.00
4	C-SVC	Sigmoid	3	48.77	50.00
5	C-SVC	Sigmoid	3	39.79	50.00

Table 6 SVM accuracy results using configuration 2.

Configuration 3:

In this configuration, the Alpha frequency range (8-12Hz) was tested. Using the LDA classifier, the results of the accuracy tests ranged from 48.2653% to 57.0427%. a table of the results can be seen below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	52.19	65.10
4	54.18	65.10
5	53.62	65.10
6	57.04	65.10
7	51.73	65.10
8	50.15	65.10
9	51.74	65.10
10	48.26	65.10

Table 7 LDA accuracy test results using configuration 3.

As for the Multi-Layer Perceptron, the accuracy-test results varied from 43.3676% to 51.3265%. A table of the results can be seen below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	43.36	64.38
3	40	0.0001	0.1	43.98	57.14
3	50	0.0001	0.1	46.32	64.94
4	30	0.0001	0.1	46.53	64.64
4	40	0.0001	0.1	44.94	65.15
4	50	0.0001	0.1	49.28	64.94
5	30	0.0001	0.1	51.32	64.48
5	40	0.0001	0.1	48.87	64.59
5	50	0.0001	0.1	48.16	65.00

Table 8 Multi-Layer Perceptron accuracy test results using configuration 3.

Using the SVM classifier, the accuracy-test results varied from a range of 38.3168% to 60.1020%. a table with the results can be seen below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	51.88	64.84
4	C-SVC	Linear	3	54.54	64.94
5	C-SVC	Linear	3	52.24	64.74
3	NU-SVC	Linear	3	38.82	50.00
4	NU-SVC	Linear	3	50.35	50.00
5	NU-SVC	Linear	3	36.63	50.00
3	C-SVC	Polynomial	3	44.58	58.41
4	C-SVC	Polynomial	3	45.00	58.31
5	C-SVC	Polynomial	3	42.80	58.31
3	C-SVC	Radial basis function	3	58.77	66.93
4	C-SVC	Radial basis function	3	60.10	66.83
5	C-SVC	Radial basis function	3	55.51	66.83
3	C-SVC	Sigmoid	3	38.31	50.71
4	C-SVC	Sigmoid	3	41.98	51.98
5	C-SVC	Sigmoid	3	47.14	51.73

Table 9 SVM accuracy test results using configuration 3.

Configuration 4:

In configuration four, the Beta frequency is tested which has a range of 12Hz - 24Hz. Using the LDA classifier the results of the accuracy-test ranged from 68.1122% to 73.0102%. results of the classifier can be seen in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	68.88	75.51
4	73.01	75.51
5	68.26	75.51
6	71.57	75.51
7	72.55	75.51
8	68.38	75.51
9	69.99	75.51
10	68.11	75.51

Table 10 LDA accuracy test results using configuration 4.

Using the Multi-Layer Perceptron, the results ranged from 60.3169% to 67.9582%. a table below shows the results of the classifier.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	66.74	73.06
3	40	0.0001	0.1	64.44	73.77
3	50	0.0001	0.1	60.31	73.87
4	30	0.0001	0.1	67.55	72.44
4	40	0.0001	0.1	67.24	72.80
4	50	0.0001	0.1	67.95	73.01
5	30	0.0001	0.1	67.29	73.82
5	40	0.0001	0.1	67.39	73.06
5	50	0.0001	0.1	65.45	72.80

Table 11 Multi-Layer Perceptron accuracy test results using configuration 4.

The SVM classifier, on the other hand, achieved a range of 65.8163% to 74.1327%. results can be seen in a table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	70.21	75.40
4	C-SVC	Linear	3	74.13	75.40
5	C-SVC	Linear	3	70.05	75.45
3	C-SVC	Polynomial	3	70.46	73.87
4	C-SVC	Polynomial	3	71.32	73.87
5	C-SVC	Polynomial	3	69.79	73.87
3	C-SVC	Radial basis function	3	68.98	76.47
4	C-SVC	Radial basis function	3	69.64	76.37
5	C-SVC	Radial basis function	3	65.81	76.53

Table 12 SVM accuracy test results.

Configuration 5:

For configuration five, the gamma frequency range was used which is 24Hz - 40Hz. The accuracy test of the LDA classifier in this configuration ranged from 23.8776% to 41.1243%. results of the classifier can be seen below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	41.12	53.26
4	39.84	53.26
5	40.91	53.26
6	39.55	53.26
7	31.88	53.26
8	25.56	53.26
9	28.25	53.26
10	23.87	53.26

Table 13 LDA accuracy test results using configuration 5.

The Multi-Layer perceptron results ranged from 37.1939% to 47.0918%. the results are shown in a table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	38.31	51.62
3	40	0.0001	0.1	42.29	52.65
3	50	0.0001	0.1	41.37	50.00
4	30	0.0001	0.1	47.09	54.28
4	40	0.0001	0.1	47.04	52.85
4	50	0.0001	0.1	43.62	52.44
5	30	0.0001	0.1	37.19	41.37
5	40	0.0001	0.1	40.86	52.75
5	50	0.0001	0.1	40.71	45.00

Table 14 Multi-Layer Perceptron accuracy results using configuration 5.

The results of the SVM classifier ranged from 32.6020% to 47.2143%. Results are shown in a table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	38.31	53.21
4	C-SVC	Linear	3	40.00	53.26
5	C-SVC	Linear	3	35.81	53.21
3	C-SVC	Polynomial	3	38.01	51.53
4	C-SVC	Polynomial	3	37.70	51.63
5	C-SVC	Polynomial	3	32.60	51.42
3	C-SVC	Radial basis function	3	47.21	63.21
4	C-SVC	Radial basis function	3	46.07	63.21
5	C-SVC	Radial basis function	3	46.22	63.26

Table 15 SVM accuracy test results using configuration 5.

Configuration 6:

The frequency range tested in this configuration is the Delta + Theta range, which is 2Hz – 8Hz. The LDA Classifier scored an accuracy test results between 36.3776% and 49.5443%. results can be seen in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	42.65	56.02
4	46.47	56.02
5	43.52	56.02
6	49.54	56.02
7	44.74	56.02
8	41.12	56.02
9	42.03	56.02
10	36.37	56.02

Table 16 LDA accuracy test results using configuration 6.

The Multi-Layer Perceptron classifier scored an accuracy test results ranging from 37.9592% to 48.0918%. the results are shown in a table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	40.15	55.96
3	40	0.0001	0.1	40.35	48.52
3	50	0.0001	0.1	43.26	57.19
4	30	0.0001	0.1	44.69	56.93
4	40	0.0001	0.1	48.09	53.26
4	50	0.0001	0.1	46.22	56.42
5	30	0.0001	0.1	37.95	48.36
5	40	0.0001	0.1	40.66	55.86
5	50	0.0001	0.1	39.13	53.36

Table 17 Multi-Layer Perceptron accuracy test results using configuration 6.

The SVM classifier scored results ranging from 36.3265% to 45.0000%. The results can be seen in the table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	39.69	56.73
4	C-SVC	Linear	3	44.64	53.68
5	C-SVC	Linear	3	40.61	56.68
3	C-SVC	Polynomial	3	42.19	55.71
4	C-SVC	Polynomial	3	40.96	55.71
5	C-SVC	Polynomial	3	36.32	55.76
3	C-SVC	Radial basis function	3	42.65	57.60
4	C-SVC	Radial basis function	3	45.00	57.70
5	C-SVC	Radial basis function	3	40.15	57.60

Table 18 SVM accuracy test results using configuration 6.

Configuration 7:

Since the Alpha + Beta frequency range (8-24Hz) achieved the best results in the previous tests, it will be used for the rest of the variations in the next configurations. In this configuration, there will be no spatial filter used. The LDA classifier in this configuration had an accuracy test results that range from 56.5816% to 62.0449%. results can be seen in a table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	62.04	72.95
4	60.71	72.95
5	59.28	72.95
6	60.35	72.95
7	60.20	72.95
8	56.58	72.95
9	59.27	72.95
10	59.13	72.95

Table 19 LDA accuracy test results using configuration 7.

The Multi-Layer Perceptron in this configuration had an accuracy test results ranging from 36.2755% to 42.2449%. The results can be seen in the table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	38.31	54.84
3	40	0.0001	0.1	39.28	54.54
3	50	0.0001	0.1	38.52	49.38
4	30	0.0001	0.1	41.42	54.23
4	40	0.0001	0.1	42.24	55.40
4	50	0.0001	0.1	40.56	55.00
5	30	0.0001	0.1	36.83	53.77
5	40	0.0001	0.1	38.26	53.06
5	50	0.0001	0.1	36.27	56.47

Table 20 Multi-Layer Perceptron accuracy test results using configuration 7.

The SVM classifier scored an accuracy test results in the range of 44.4898% to 61.2803%. the results can be seen in the table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	61.28	72.75
4	C-SVC	Linear	3	59.69	72.75
5	C-SVC	Linear	3	58.21	72.75
3	C-SVC	Polynomial	3	60.56	75.35
4	C-SVC	Polynomial	3	60.00	75.30
5	C-SVC	Polynomial	3	60.25	75.35
3	C-SVC	Radial basis function	3	48.83	71.47
4	C-SVC	Radial basis function	3	53.36	71.58
5	C-SVC	Radial basis function	3	44.48	71.63

Table 21 SVM accuracy test results using configuration 7.

Configuration 8:

Configuration 8 added a spatial filter to the design but with 3 output channels instead of 2, using the frequency range of Alpha + Beta (8-24Hz). The Spatial filter coefficient used in this configuration are (4;0;-1;0;-1;-1;0;0;-1;0;0;4;0;-1;0;0;-1;-1;0;-1;0;0;3;-1;0;-1;0;-1;0;0). The LDA classifier in this case scored an accuracy test results that ranges between 70.5102% to 73.7245%. Results can be seen in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	71.38	78.31
4	73.72	78.31
5	71.02	78.31
6	73.72	78.31
7	73.31	78.31
8	71.27	78.31
9	72.24	78.31
10	70.51	78.31

Table 22 LDA accuracy test results using configuration 8.

The Multi-Layer Perceptron accuracy test results in configuration 8 are in the range of 63.0727% to 67.6020%. Results can be seen in the table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	66.94	76.58
3	40	0.0001	0.1	63.07	76.73
3	50	0.0001	0.1	65.41	76.73
4	30	0.0001	0.1	64.54	76.53
4	40	0.0001	0.1	65.15	76.37
4	50	0.0001	0.1	67.60	76.47
5	30	0.0001	0.1	64.74	76.37
5	40	0.0001	0.1	65.40	76.53
5	50	0.0001	0.1	66.32	76.53

Table 23 Multi-Layer Perceptron accuracy test results using configuration 8.

The SVM classifier scored an accuracy test results in the range of 67.9592% to 74.5408%. the results can be seen in the table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	72.14	78.41
4	C-SVC	Linear	3	74.54	78.52
5	C-SVC	Linear	3	72.19	78.41
3	C-SVC	Polynomial	3	70.77	78.31
4	C-SVC	Polynomial	3	73.36	78.46
5	C-SVC	Polynomial	3	71.47	78.36
3	C-SVC	Radial basis function	3	71.07	79.74
4	C-SVC	Radial basis function	3	70.51	79.79
5	C-SVC	Radial basis function	3	67.95	79.79

Table 24 SVM accuracy test results using configuration 8.

Configuration 9:

Configuration 9 added one more output channel to the spatial filter reaching four channels in total, the frequency range used is Alpha + Beta (8-24Hz). The spatial Filter coefficient chosen in this configuration are (4;0;-1;0;-1;-1;0;0;-1;0;0;4;0;-1;0;0;-1;-1;0;-1;0;0;3;-1;0;-1;0;-1;0;0;0;-1;0;3;0;0;-1;0;0;-1). The LDA classifier had an accuracy test results which ranged from 71.0773% to 74.6429%. Results are shown in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	71.07	78.36
4	74.64	78.36
5	71.27	78.36
6	74.22	78.36
7	74.18	78.36
8	72.04	78.36
9	72.85	78.36
10	71.63	78.36

Table 25 LDa accuracy test results using configuration 9.

The Multi-Layer Perceptron classifier scored an accuracy test results ranging from 50.0574% to 63.5714%. Results are shown in a table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	61.59	75.91
3	40	0.0001	0.1	50.05	75.76
3	50	0.0001	0.1	61.84	76.37
4	30	0.0001	0.1	58.41	75.30
4	40	0.0001	0.1	53.87	76.27
4	50	0.0001	0.1	63.01	76.17
5	30	0.0001	0.1	52.04	74.74
5	40	0.0001	0.1	61.32	75.91
5	50	0.0001	0.1	63.57	76.37

Table 26 Multi-Layer Perceptron accuracy test results using configuration 9.

The SVM classifier scored an accuracy test results that range from 69.5918% to 76.1224%. Results are seen in a table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	72.65	78.57
4	C-SVC	Linear	3	76.12	78.57
5	C-SVC	Linear	3	73.01	78.57
3	C-SVC	Polynomial	3	72.60	80.35
4	C-SVC	Polynomial	3	75.30	80.35
5	C-SVC	Polynomial	3	72.95	80.35
3	C-SVC	Radial basis function	3	72.19	81.12
4	C-SVC	Radial basis function	3	73.77	81.07
5	C-SVC	Radial basis function	3	69.59	80.96

Table 27 SVM accuracy test results using configuration 9.

Configuration 10:

In this configuration, the Alpha + Beta frequency range (8-24Hz) is used with a spatial filter that has 5 output channels. The spatial filter coefficients used are (4;0;-1;0;-1;-1;0;0;-1;0;0;4;0;-1;0;0;-1;-1;0;-1;0;0;3;-1;0;-1;0;-1;0;0;0;-1;0;3;0;0;-1;0;0;-1;0;0;-1;0;4;0;-1;-1;-1;0). The LDA classifier in this configuration scored an accuracy test results ranging from 70.5665% to 75.9694%. Results can be seen in a table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	70.56	78.97
4	75.96	78.97
5	70.76	78.97
6	73.31	78.97
7	73.36	78.97
8	71.93	78.97
9	72.85	78.97
10	71.63	78.97

Table 28 LDA accuracy test results using configuration 10.

The Multi-Layer Perceptron classifier scored an accuracy test results that range from 62.2065% to 65.2041%. Results can be seen in the table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	62.20	77.34
3	40	0.0001	0.1	62.56	77.65
3	50	0.0001	0.1	62.97	77.44
4	30	0.0001	0.1	63.92	76.93
4	40	0.0001	0.1	64.18	77.65
4	50	0.0001	0.1	65.20	77.65
5	30	0.0001	0.1	63.11	77.14
5	40	0.0001	0.1	63.92	77.14
5	50	0.0001	0.1	64.54	77.65

Table 29 Multi-Layer Perceptron accuracy test results using configuration 10.

SVM classifier scored an accuracy test results ranging from 71.8398% to 77.3469%. Results are seen in a table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	73.01	79.38
4	C-SVC	Linear	3	77.34	79.48
5	C-SVC	Linear	3	73.21	79.38
3	C-SVC	Polynomial	3	71.83	81.17
4	C-SVC	Polynomial	3	75.86	81.22
5	C-SVC	Polynomial	3	72.95	80.35
3	C-SVC	Radial basis function	3	72.19	81.12
4	C-SVC	Radial basis function	3	73.77	81.07
5	C-SVC	Radial basis function	3	72.90	81.12

Table 30 SVM accuracy test results using configuration 10.

Configuration 11:

In this configuration, the Alpha + Beta frequency band (8-24Hz) was used with a spatial filter that has eight output channels. The spatial filter coefficient used in this configuration are (4;0;-1;0;-1;-1;0;0;-1;0;0;4;0;-1;0;0;-1;-1;0;-1;0;0;3;-1;0;-1;0;-1;0;0;0;-1;0;3;0;0;-1;0;0;-1;0;0;-1;0;4;0;-1;-1;-1;0;0;0;0;0;-1;3;0;-1;-1;0;-1;0;0;0;0;-1;0;3;-1;0;-1;0;0;0;0;-1;-1;3).

The LDA classifier scored an accuracy test results ranging from 68.8265 to 77.9592%. Results can be seen in a table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	71.99	80.10
4	77.95	80.10
5	71.42	80.10
6	72.59	80.10
7	72.24	80.10
8	68.01	80.10
9	70.76	80.10
10	68.82	80.10

Table 31 LDA accuracy test results using configuration 11.

The Multi-Layer Perceptron classifier scored an accuracy test results that ranged from 43.9736% to 64.3367%. Results can be seen in a table below.

K Number	Number of neurons in the hidden layer	Learning stop condition	Learning coefficient	Test accuracy result (%)	Training set accuracy result (%)
3	30	0.0001	0.1	62.41	72.70
3	40	0.0001	0.1	62.56	79.64
3	50	0.0001	0.1	62.97	77.44
4	30	0.0001	0.1	52.85	53.61
4	40	0.0001	0.1	58.87	79.74
4	50	0.0001	0.1	53.72	79.59
5	30	0.0001	0.1	43.97	60.76
5	40	0.0001	0.1	58.67	79.89
5	50	0.0001	0.1	64.33	80.05

Table 32 Multi-Layer Perceptron accuracy test results using configuration 11.

Using SVM classifier in this configuration scored an accuracy test results ranging from 70.1020 to 79.3878%. Results are shown in a table below.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	74.59	81.07
4	C-SVC	Linear	3	79.38	81.07
5	C-SVC	Linear	3	73.77	79.38
3	C-SVC	Polynomial	3	73.26	86.02
4	C-SVC	Polynomial	3	78.82	86.02
5	C-SVC	Polynomial	3	75.00	85.96
3	C-SVC	Radial basis function	3	72.24	85.51
4	C-SVC	Radial basis function	3	77.60	85.40
5	C-SVC	Radial basis function	3	70.10	85.45

Table 33 SVM accuracy test results using configuration 11.

Configuration 12:

In this configuration, a different MI process is tested, which is the Common Spatial Pattern Motor Imagery (CSP MI), it utilizes a different spatial filter that needs training. In this configuration eight output channels were trained and used. However, the frequency band that was tested on is Alpha + Beta (8-24Hz). Due to Multi-Layer Perceptron poor and random results, it is not tested in further configurations. The LDA classifier in this CSP configuration scored an accuracy test results ranging from 67.1429% to 77.2449%. The results are shown in the table below.

K Number	Test accuracy result (%)	Training set accuracy result (%)
3	75.56	81.58
4	77.24	81.58
5	75.10	81.58
6	73.05	81.58
7	69.43	81.58
8	67.14	81.58
9	69.58	81.58
10	69.58	81.58

Table 34 LDA accuracy test results using configuration 12.

The SVM classifier scored an accuracy test results that range from 35.9694% to 74.2347%. Results can be shown in the table below. Noting that the low results only came from the polynomial SVM.

K Number	SVM Type	Kernel Type	Degree	Test accuracy result (%)	Training set accuracy result (%)
3	C-SVC	Linear	3	73.52	78.16
4	C-SVC	Linear	3	74.23	78.21
5	C-SVC	Linear	3	72.90	78.16
3	C-SVC	Polynomial	3	47.94	50.00
4	C-SVC	Polynomial	3	52.60	50.00
5	C-SVC	Polynomial	3	35.96	50.00
3	C-SVC	Radial basis function	3	71.33	73.57
4	C-SVC	Radial basis function	3	71.78	73.52
5	C-SVC	Radial basis function	3	70.10	72.52

Table 35 SVM accuracy test results using configuration 12.

5.5. Results Summary

Configuration	LDA Best Accuracy (%)	Multi-Layer Perceptron Best Accuracy (%)	SVM Best Accuracy (%)
1	73.46	70.61	73.62
2	64.69	54.79	64.69
3	57.04	51.32	60.10
4	73.01	67.95	74.13
5	41.12	47.09	47.21
6	46.47	48.09	45.00
7	62.04	42.24	61.28
8	73.72	67.60	74.54
9	74.64	63.57	76.12
10	75.96	65.20	77.34
11	77.95	64.33	79.38
12	77.24	Nan	74.23

Table 36: Results Summary of All the Configurations

5.6. Discussion:

Looking at the tables of the accuracy results, the number varies dramatically from one configuration to another and from one algorithm to another. The first configuration is the default configuration of the MI BCI in OpenVibe, which had two outputs in the spatial filter and a frequency band of alpha plus beta, in this default configuration the results were not bad, around 73% in LDA classifier, 70% for the Multi-Layer Perceptron and 73% for the SVM. After trying all the combinations of frequency bands separated and together, it was obvious that the default Alpha plus Beta range is the optimal range for the Motor Imagery BCI.

After trying the combination of the frequency bands, the setting of the spatial filter was configured, starting with removing the spatial filter to adding more outputs until 8 outputs. Removing the spatial filter had a negative effect on the results, LDA classifier achieved around 60% which is 10% less than with a two output spatial filter, Multi-Layer perceptron around 40% and SVM around 60%.

Moreover, with increasing the number of outputs of the spatial filter and trying many spatial filter coefficients the results were improving. The more outputs in the spatial filter the more accurate the system was achieving until reaching eight outputs, after that the performance was not improving and but decreasing. In configuration eleven, eight output channels were used in the spatial filter and the Alpha plus beta frequency range was used, however the best result that was achieved in the testing is in this configuration. For the LDA classifier, around 78% accuracy result was achieved, around 64% for the Multi-Layer Perceptron and around 80% for the SVM which is why 4 k-fold linear SVM is to be used as the main classifier in this project.

Moreover, Common Spatial Pattern (CSP) Motor Imagery (MI) BCI is tested, especially that it is very common in the use of navigation as seen in the background study, however, in this project it had very similar result to the normal MI BCI.

However, the results achieved were dramatically changing from one setup to another and having random numbers does not evaluate the system accurately, this is why Wilcoxon test is done on the results that had the best potential.

5.6.1. Testing of Linear SVM and LDA with 10k fold using Wilcoxon test:

The setup of the most successful configuration is used which is configuration 11 that had eight output in the spatial filter and Alpha plus Beta frequency band.

Cross-Validation Performance with k= 10

Method	K	Cross-validation Accuracy
Linear SVM	10	73.2653
LDA	10	68.8265

Table 37: Cross-Validation Accuracy of Linear SVM and LDA

Partitioned Performance k=10

Linear SVM	LDA
39.7959	32.6531
76.5306	75.5102
88.2653	86.7347
95.4082	95.4082
79.5918	71.4286
69.3878	68.3673
76.0204	76.0204
77.0408	73.4694
73.4694	63.7755
57.1429	44.8980

Table 38: Partitioned Cross-validation

p-value: 0.3622291

The p-value is larger than 0.05, which implies that the difference between the two results is by chance and there is no statistical difference between them.

5.6.2. Testing of 10k fold LDA with MI and CSP MI using Wilcoxon test:

Another Wilcoxon test is done on a 10 k-fold LDA of the normal MI BCI and the CSP MI BCI, using the configuration eleven settings.

Cross-Validation Performance with k= 10

Method	K	Cross-validation Accuracy
LDA MI	10	68.8265
LDA CSP MI	10	69.8503

Table 39 10 k-fold cross-validation test for LDA MI and LDA CSP MI.

Partitioned Performance k=10

LDA MI	LDA CSP MI
32.6531	27.0408
75.5102	65.6061
86.7347	94.3878
95.4082	95.4082
71.4286	68.8776
68.3673	62.7551
76.0204	65.8163
73.4694	82.6531
63.7755	53.5714

44.8980	52.0408
---------	---------

Table 40: Partitioned Performance.

p-value: 0.5935504

There is no statistical difference between these two results which means as well that the results are different by chance.

6. Project Management:

In this chapter, an overview of the way this project was managed and how tasks were organized to reach the final version of the project. Many steps were taken, some took more time than planned and some didn't. an updated plan on the project can be found next as well as what changed from the previous plan plus a reflective discussion on the project.

After the project proposal, a clear idea on the project was planned, the path to finish the project was with doing a task after another, starting from taking online courses of software that are needed in this project to finalizing the project.

6.1. Updated Plan:

A plan for the project was decided in the project proposal, and before starting on the final project. Mostly all the tasks were worked on and finished. Below there is a list of tasks that were worked on during this project.

- Further reading on the topic and further understanding of similar projects that was done before was an important task that helped understanding how to start and how to continue planning every step.
- Exploring the tools that are needed for this project is essential, of them is OpenVibe, many tutorials and documentation was read and at the end of this task, a design of a Motor Imagery BCI was done.
- Another necessary tool for this dissertation was Unity3D, it is a software with a wide range of options and potentials, an online course from Lynda was helpful and could give the ability to start and build an environment including the physics and the script editing and writing. However, noting that the online course covered mainly all the aspects of the software which is why it took more time than planned.
- Creating a basic virtual environment with a first-person perspective character that can move inside the environment.
- Designing the Motor Imagery (MI) BCI in the OpenVibe software.
- Training the classifier on the dataset.
- Trying three machine learning algorithms in the design.
- Testing the same system on different frequency bands
- Testing the same system without a spatial filter.

- Trying many combinations of spatial filter coefficients until having the best results which were done in a trial and error approach.
- Adding outputs to the spatial filter.
- Testing a different type of BCI, the Common Spatial Pattern (CSP) Motor Imagery (MI) BCI.
- Designing the online system that will use the configuration of the classifier trained in the previous design and apply it on a new dataset or on a live recording of brain signals.
- Writing down all the results from all the configurations that are tried which include the steps talked mentioned in the tasks before this.
- Evaluating the results using the Wilcoxon rank-sum test.
- Attempting to connect OpenVibe with Unity3D using LSL via acquisition server, which did not work.
- Attempting to fix the problem with LSL while at the same time considering other communication methods.
- Connecting both software using LSL, which was a time-consuming task because there is no clear information and tutorials on how to properly connect the software.
- After establishing the connection between OpenVibe and Unity3D, a test on moving a ball was done to make sure that the method is working.
- After successfully being able to read the values of the class probability from OpenVibe to Unity via LSL, using the numbers on the character is done by editing the movement script of the character in Unity3D.
- Writing the dissertation, after doing all the work and tasks that lead to finishing the project writing the dissertation could be done.

6.2. Kanban Board:

In this section, an overview of the Kanban board will be seen, however, two figures will be shown, Fig.23 shows the task in progress and the tasks done, and Fig.24 shows how the task that was in progress moved to the done tasks when they were done.

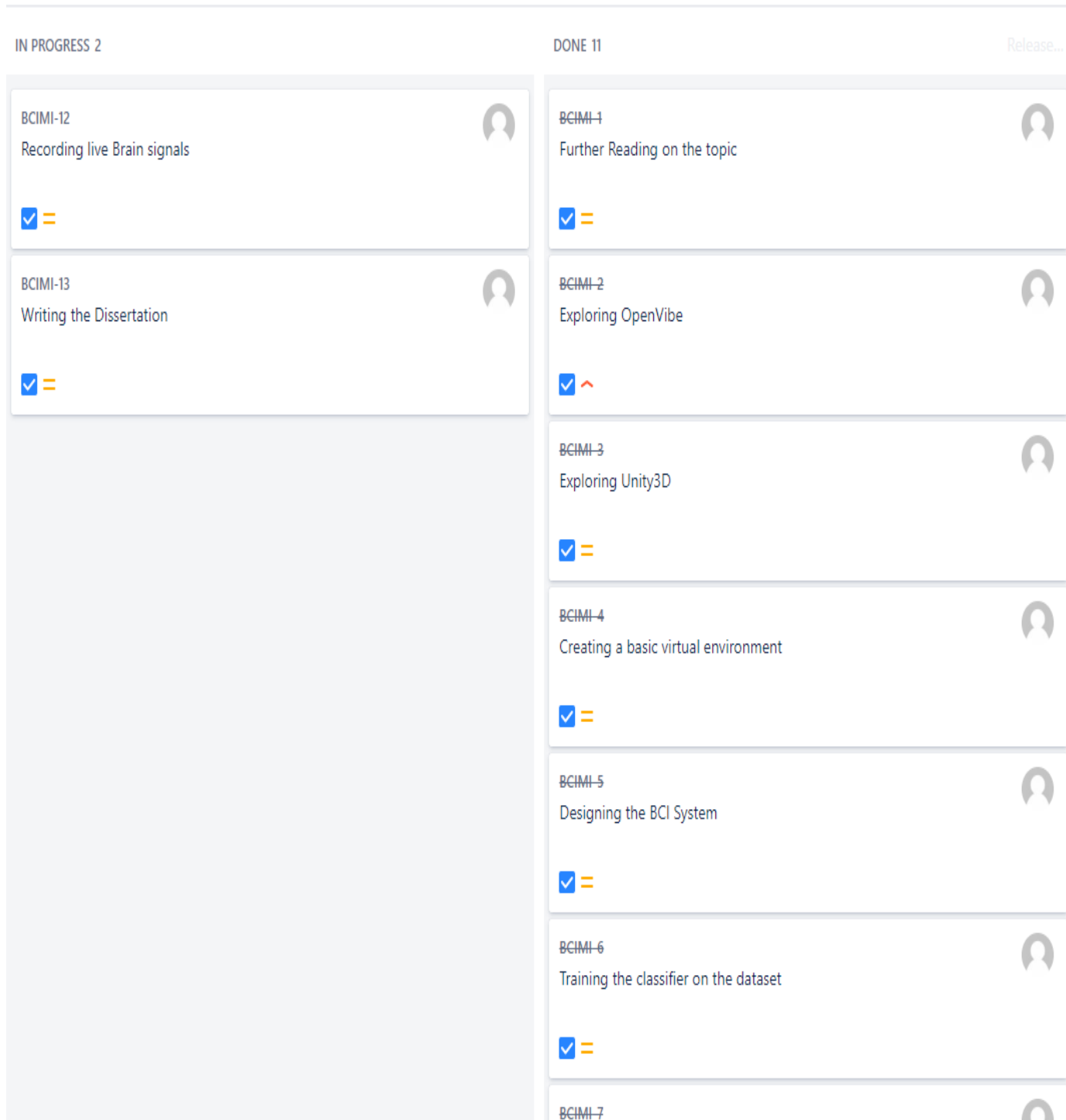


Figure 23: Kanban Board part 1

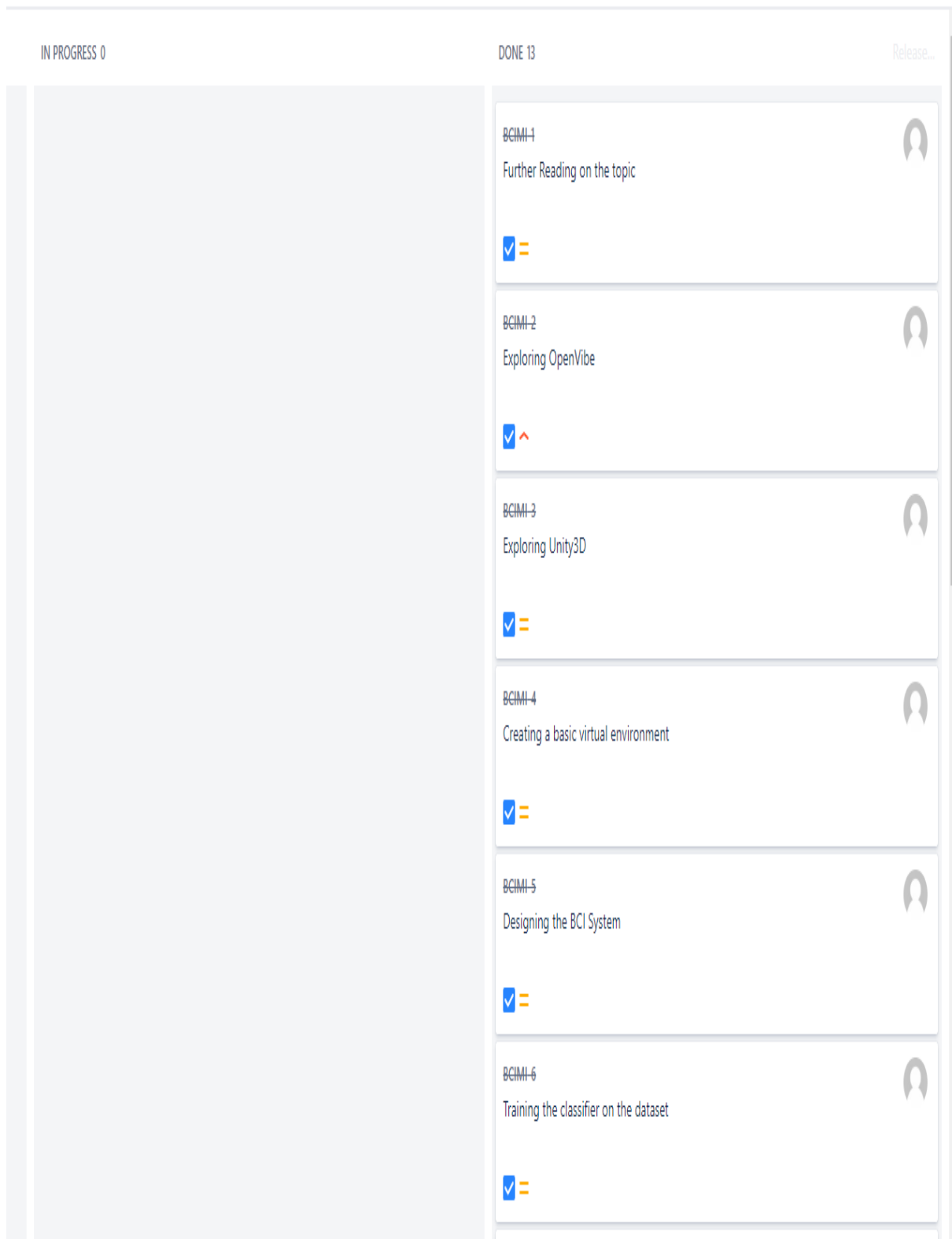


Figure 24 Kanban Board part 2

6.3. Changes from The Proposal:

There are some changes from what was decided in the proposal and what was done in the final project, and these changes are stated and discussed in this section.

The first and main change from the proposal to the final project is the ability to control functionalities which will require more than two classes in the system, which is not a hard task to do but because of using a dataset that had only two classes (left and right) a third class was not made although it is possible to have more than two classes but it will require more time, especially that a big amount of the time on this project was invested in the testing of many configurations and evaluating the system plus establishing a connection between OpenVibe and Unity3D.

In the feature extraction part of the project which is done using the OpenVibe software, as planned in the proposal Common Spatial Pattern filter was trained and was used in one of the configurations, however the features used in the project is the average energy of the power of the signal.

For the Machine learning part of the project, SVM and LDA were used as planned but a third algorithm was added which is the Multi-Layer Perceptron.

One of the Objectives that was set in the project proposal was not implemented in the final project which is producing an SSVEP BCI system and combining the SSVEP and MI in the same design. A Motor Imagery design was made with many configurations and another type of MI BCI was designed which is the CSP Motor Imagery BCI.

6.4. Reflective Discussion:

Overall the final project is a completed project that has a working Brain-Computer Interface system that can control a character in another software using the communication tool LSL, which was the main goal of the project. However, this project can be improved and worked on in many different aspects. Starting from adding more classes to the project which will open the possibility of controlling equipment in the virtual environment, another thing that can be improved in the project is using a different type of BCI such like the SSVEP type and if possible do a hybrid system that uses Motor Imagery for the navigation in the house and SSVEP for controlling the functionalities, which can be done with an interface that can be controlled using the brain signals, the interface can give the user the option to choose between navigating or controlling functionalities.

Another idea that can improve this project, is to have the possibility to navigate automatically, for example using SSVEP and by focusing on a light that will have a certain frequency, the user will automatically navigate to a pre-programmed location, which will require the use of localization which is not a hard thing with today's technology, especially with the existence of routers and access points.

Many settings and machine learning algorithms were tested in this project which gave a clear idea of what improves the BCI system and what is needed to increase the performance of the design.

The nature of this project, a multi-disciplinary project provided a big experience in the project management aspect, managing the tasks of different subjects and being able to connect these fields together is not possible without a good project management ability which was learned and improved dramatically during this project. As the project was advancing the more precise the project management was needed to be, as time passed the deadline of the project was getting closer, however, the intention in this project was to use Jira to manage the report but because there was a project in the software, in the beginning, a manual approach was used to manage all the tasks and the goals until near the end of the project Jira was then used to manage the remaining task and to check if all the tasks were done. This project improved my ability to manage a big number of tasks and was able to give the ability to have a better time and task management.

7. Conclusion:

To summarize, A Brain-Computer Interface system was designed to control a first-person perspective character that exists in a virtual environment that was created using Unity3D software. The BCI system was designed using OpenVibe, two types of BCI have tested a standard Motor Imagery system and another Common Spatial Pattern Motor Imagery BCI.

However, the BCI system had two classes, one is left and the other is right; these classes were extracted from a pre-recorded brain signals dataset, then processed and features were extracted to be classified using three machine learning algorithms, Linear Discriminant Analysis (LDA), Multi-Layer Perceptron and Support Vector Machine (SVM). Results of the classifiers can be seen in tables in the evaluation section of the dissertation.

Moreover, twelve Configurations were tested each had a variation in the settings, changing the frequency band, changing the setting of the spatial filter and testing a different type of spatial filter. These configurations can all be seen with their results in this project.

Results were evaluated using the Wilcoxon rank-sum test which can identify if the results were statistically different or just different by chance, plus a confusion matrix is shown.

However, the Support Vector Machine (SVM) with 4 k-fold using the Alpha plus Beta frequency band and using eight outputs in the spatial filter (Configuration 11) had the best accuracy result.

Brain signals were recorded live using the Enobio equipment, signals were extracted converted to CSV format and a new training designed was used to train this new dataset acquired.

This project was managed using Kanban Board and by planning tasks and working on each task until finished.

7.1. Future Direction:

This project and this field, in general, have a very big potential to make the life of humans easier and to increase the quality of life of the people. However, this project can be improved and worked on to be finalized and tested on real systems, with the addition of many features to the system, such as the ability to control functionalities and to navigate through a pre-programmed environment using localization.

An interface can be worked on using android SDKs and a tablet that can be attached to a wheelchair, however, this interface can be friendly to BCI systems and can have a hybrid BCI system that includes MI and SSVEP.

Moreover, with more time more work can be dedicated to improving the accuracy of the system and using datasets that can be recorded live.

Bibliography

- [1] S. Z. Diva, R. A. Prorna, I. I. Rahman, A. B. Islam and M. N. Islam, "Applying Brain-Computer Interface Technology for Evaluation of User Experience in Playing Games," *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, 2019.
- [2] B. Lei, X. Liu, S. Liang, W. Hang, Q. Wang, K.-S. Choi and J. Qin, "Walking Imagery Evaluation in Brain Computer Interfaces via a Multi-View Multi-Level Deep Polynomial Network," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 3, pp. 497-506, 2019.
- [3] J. Olias, R. Martin-Clemente, M. A. Sarmiento-Vega and S. Cruces, "EEG Signal Processing in MI-BCI Applications With Improved Covariance Matrix Estimators," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 5, pp. 895-904, 2019.
- [4] D. Paul and M. Mukherjee, "Automation of wheelchair using brain computer interface (BCI) technique," 2019.
- [5] A. S. M. Miah, S. R. A. Ahmed, M. R. Ahmed, O. Bayat, A. D. Duru and M. I. Molla, "Motor-Imagery BCI Task Classification Using Riemannian Geometry and Averaging with Mean Absolute Deviation," *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)*, 2019.
- [6] P. Cheng, P. Autthasan, B. Pijarana, E. Chuangsuwanich and T. Wilaiprasitporn, "Towards Asynchronous Motor Imagery-Based Brain-Computer Interfaces: a joint training scheme using deep learning," *TENCON 2018 - 2018 IEEE Region 10 Conference*, 2018.
- [7] V. Mishuhina and X. Jiang, "Feature Weighting and Regularization of Common Spatial Patterns in EEG-Based Motor Imagery BCI," *IEEE Signal Processing Letters*, vol. 25, no. 6, pp. 783-787, 2018.
- [8] S. Kanoga, A. Kanemura and H. Asoh, "A COMPARATIVE STUDY OF FEATURES AND CLASSIFIERS IN SINGLE-CHANNEL EEG-BASED MOTOR IMAGERY BCI," *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2018.
- [9] X. Wan, K. Zhang, S. Ramkumar, J. Deny, G. Emayavaramban, M. Siva Ramkumar and A. F. Hussein, "A Review on Electroencephalogram Based Brain Computer Interface for Elderly Disabled," *IEEE Access*, vol. 7, pp. 36380-36387, 2019.
- [10] K.-C. Chuang and Y.-P. Lin, "Cost-Efficient, Portable, and Custom Multi-Subject Electroencephalogram Recording System," *IEEE Access*, vol. 7, pp. 56760-56769, 2019.
- [11] A. Jafarifarmand and M. A. Badamchizadeh, "EEG Artifacts Handling in a Real Practical Brain-Computer Interface Controlled Vehicle," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 6, pp. 1200-1208, 2019.

- [12] R. Chatterjee, T. Maitra, S. Hafizul Islam, M. M. Hassan, A. Alamri and G. Fortino, "A novel machine learning based feature selection for motor imagery EEG signal classification in Internet of medical things environment," *Future Generation Computer Systems*, vol. 98, pp. 419-434, 2019.
- [13] G. Cisotto, S. Pupolin and F. Piccione, "Comparison about EEG signals processing in BCI applications," *2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems (VITAE)*, 2014.
- [14] M. Varela, "Raw EEG signal processing for BCI control based on voluntary eye blinks," *2015 IEEE Thirty Fifth Central American and Panama Convention (CONCAPAN XXXV)*, 2015.
- [15] J. Arnin, D. Kahani, H. Lakany and B. A. Conway, "2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)," *Evaluation of Different Signal Processing Methods in Time and Frequency Domain for Brain-Computer Interface Applications*, 2018.
- [16] C. Chen, W. Song, J. Zhang, Z. Hu and H. Xu, "An Adaptive Feature Extraction Method for Motor-Imagery BCI Systems," *2010 International Conference on Computational Intelligence and Security*, 2010.
- [17] "Products - Unity," Unity, 2019. [Online]. Available: https://unity3d.com/unity?_ga=2.158755388.724224847.1566391693-909955988.1553272268.
- [18] "Lynda," [Online]. Available: <https://www.linkedin.com/learning/unity-3d-essential-training/next-steps?u=51088249>.
- [19] L. Junwei, S. Ramkumar, G. Emayavaramban, D. F. vinod, M. Thilagaraj, V. Muneeswaran, M. Pallikonda Rajasekaran, V. Venkataraman and A. F. Hussein, "Brain Computer Interface for Neurodegenerative Person Using Electroencephalogram," *IEEE Access*, vol. 7, pp. 2439-2452, 2019.
- [20] N. Mora, I. De Munari, P. Ciampolini and J. del R. Millán, "Plug&Play Brain-Computer Interfaces for effective Active and Assisted Living control," *Medical & Biological Engineering & Computing*, vol. 55, no. 8, pp. 1339-1352, 2016.
- [21] Y. Yuan, W. Su, Z. Li and G. Shi, "Brain-Computer Interface-Based Stochastic Navigation and Control of a Semiautonomous Mobile Robot in Indoor Environments," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 11, no. 1, pp. 129-141, 2019.
- [22] Ç. İ. Acı, M. Kaya and Y. Mishchenko, "Distinguishing mental attention states of humans via an EEG-based passive BCI using machine learning methods," *Expert Systems with Applications*, vol. 134, pp. 153-166, 2019.
- [23] C. G. Coogan and B. He, "Brain-Computer Interface Control in a Virtual Reality Environment and Applications for the Internet of Things," *IEEE Access*, vol. 6, pp. 10840-10849, 2018.

- [24] S. Jo and J. W. Choi, "Effective motor imagery training with visual feedback for non-invasive brain computer interface," *2018 6th International Conference on Brain-Computer Interface (BCI)*, 2018.
- [25] V. K. K. Shivappa, B. Luu, M. Solis and K. George, "Home automation system using brain computer interface paradigm based on auditory selection attention," *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2018.
- [26] B. B. Longo, A. B. Benevides, J. Castillo and T. Bastos-Filho, "Using Brain-Computer Interface to control an avatar in a Virtual Reality Environment," *5th ISSNIP-IEEE Biosignals and Biorobotics Conference (2014): Biosignals and Robotics for Better and Safer Living (BRC)*, 2014.
- [27] S. A. Yohanandan, I. Kiral-Kornek, J. Tang, B. S. Mshford, U. Asif and S. Harrer, "A Robust Low-Cost EEG Motor Imagery-Based Brain-Computer Interface," *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018.
- [28] R. Abbasi-Asl, M. Keshavarzi and D. Y. Chan, "Brain-Computer Interface in Virtual Reality," *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2019.
- [29] "Documentation," OpenViBE, 2019. [Online]. Available: <http://openvibe.inria.fr/documentation-index/>.
- [30] "Dataset Downloads," OpenViBE, 2019. [Online]. Available: <http://openvibe.inria.fr/datasets-downloads/>.
- [31] "Wilcoxon-Mann-Whitney Test Calculator -," Ccb-compute2.cs.uni-saarland.de, 2019. [Online]. Available: <https://ccb-compute2.cs.uni-saarland.de/wtest/>.
- [32] "OpenViBE forum • Index page," Openvibe.inria.fr, 2019. [Online]. Available: <http://openvibe.inria.fr/forum/>.
- [33] "xfleckx/LSL4Unity," GitHub, [Online]. Available: <https://github.com/xfleckx/LSL4Unity>.
- [34] N. Auguin, D. Morales-Jimenez and M. R. McKay, "Robust Linear Discriminant Analysis Using Tyler's Estimator: Asymptotic Performance Characterization," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.
- [35] H. Xiong, W. Cheng, J. Bian, W. Hu, Z. Sun and Z. Guo, "<inline-formula> <tex-math notation='LaTeX'>\mathcal{DBSDA}</tex-math> </inline-formula>: Lowering the Bound of Misclassification Rate for Sparse Linear Discriminant Analysis via Model Debiasing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 3, pp. 707-717, 2019.
- [36] M. Abdullah-al-mamun and T. Alam, "An approach to empirical Optical Character Recognition paradigm using Multi-Layer Perceptorn Neural Network," *2015 18th International Conference on Computer and Information Technology (ICCIT)*, 2015.
- [37] H. Czap, "Construction and interpretation of multi-layer-perceptrons," *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat.No.01CH37236)*, 2001.

- [38] F. Liu, K. J. Lee and J. Hong, "A Feature-Aware Online Learning Approach for Support Vector Machine Classification," *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018.
- [39] M. Singh, R. Pamula and S. k. shekhar, "2018 International Conference on Computing, Power and Communication Technologies (GUCON)," *Email Spam Classification by Support Vector Machine*, 2018.
- [40] A.-L. F.J, R. g.-A. J and A.-F. n. M. V, "Complete Control of an Observed Confusion Matrix," *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 2018.
- [41] "Supplements for "Chance Encounters" (GENERAL)," Stat.auckland.ac.nz, 2019. [Online]. Available: <https://www.stat.auckland.ac.nz/~wild/ChanceEnc/>. [Accessed 2019 08 16].
- [42] A. Marx, C. Backes, E. Meese, H.-P. Lenhof and A. Keller, "EDISON-WMW: Exact Dynamic Programing Solution of the Wilcoxon–Mann–Whitney Test," *Genomics, Proteomics & Bioinformatics*, vol. 14, no. 1, pp. 55-61, 2016.