# HANDS-FREE DRAWING USING ELECTROENCEPHALOGRAM (EEG) SYSTEM AND EYE TRACKING

By

NURDHIYA HUSNA BINTI HUSSEIN

14434

**FINAL REPORT**

Submitted to the Electrical & Electronics (EE) Engineering Programme

In Partial Fulfillment of the Requirements

For the Degree

Bachelor of Engineering (Hons)

(Electrical & Electronics Engineering)

Final Year Project II, September, 2014

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

## Table of Contents

## List of Figure

## List of Table

## Abstract

This project aims to develop alternative way to help patient suffering with communication disabilities that prevent them from communicating using the normal way such as speech, body language, and etc. This project utilizes EEG and eye movement, and translating them into a meaningful message or action. This task is performed using EMOTIV interfaced with BCI Application.

# Chapter 1: Introduction

## 1.1 Background

Electrical signals are emitted by the brain each time it conduct any activity. However, each signal being emitted is different in frequencies for each activity that are carried out by the brain. Recording of these signals is called Electroencephalography (EEG). Various ways and methods have been developed to manipulate these signals and translate them into set of commands. Eye tracking using EEG are currently being widely explored using numerous Brain-Computer Interfacing (BCI) technique. Figure 1 shows a basic block diagram on how the EEG signals are retrieved and integrated with BCI application. These expected classes of signals are being decoded into several control commands for defined external application based on the BCI configuration [2].

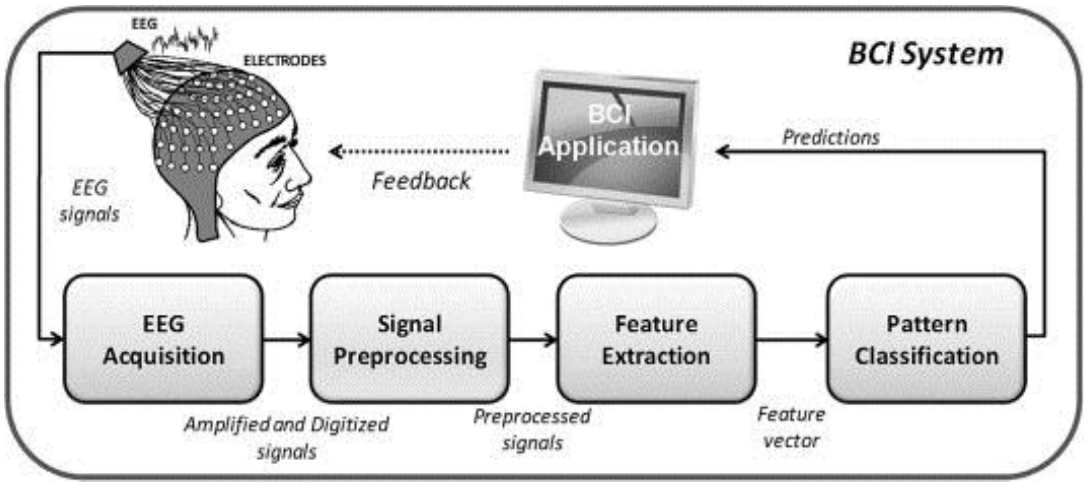


**Figure 1: General block diagram of an EEG-based BCI system [2] .**

Based on these set of signals, this project focused on translating these brain signals with the aid of eye movement tracking, into drawings.

## 1.2 Problem Statement

For a patient that suffers any kind of disabilities that restricted them from communicating with other person using the normal ways (speech, body language, etc.), this paper aim to develop a new method to deliver the patient's intention and translate it into drawing via eye movement using manipulation of EEG signals.

## 1.3 Objectives

- To establish interface that can track the eye movement and convert into desired shape/drawing.

## 1.4 Scope of Study

Scope of study for this project is to have deep understanding on how the signals and activities from the brains are recorded using EEG system. Thus, by using devices develop from EMOTIV EEG Headset, it is easier to observe the patterns, cognitive abilities, and also record each activity carried out by the brain.

Researches on EEG have been rapidly increases during these past few years. Thus, it is crucial to read and keep an update about the latest technologies and interfaces used by other researchers for more accurate results for this project paper.

# Chapter 2: Literature Review

## 2.1 Brain-Computer Interface (BCI)

Over the last 2 decades, mankind has developed a neurotechnology that link brain's activity with devices and control or command it without using external neuromuscular pathway (such as hands or legs).  Such interfaces are called brain–computer interface (BCI).  Figure 2 shows the essential part in a BCI system [1]:

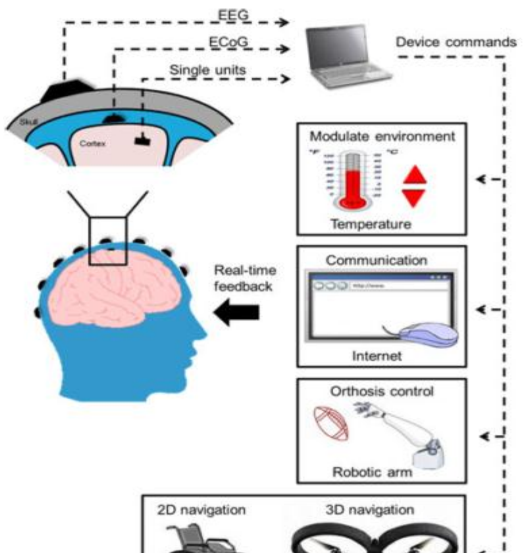
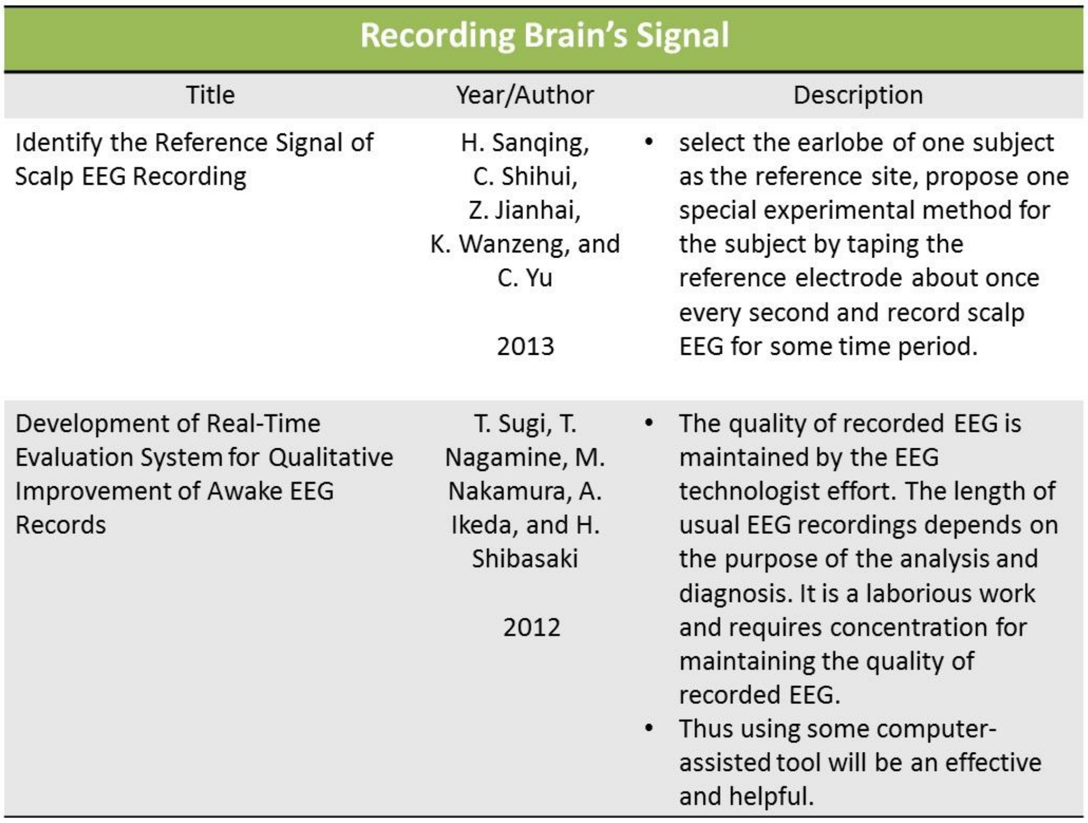


Figure 2: Schematic Diagram of Essential Components in BCI System [1].

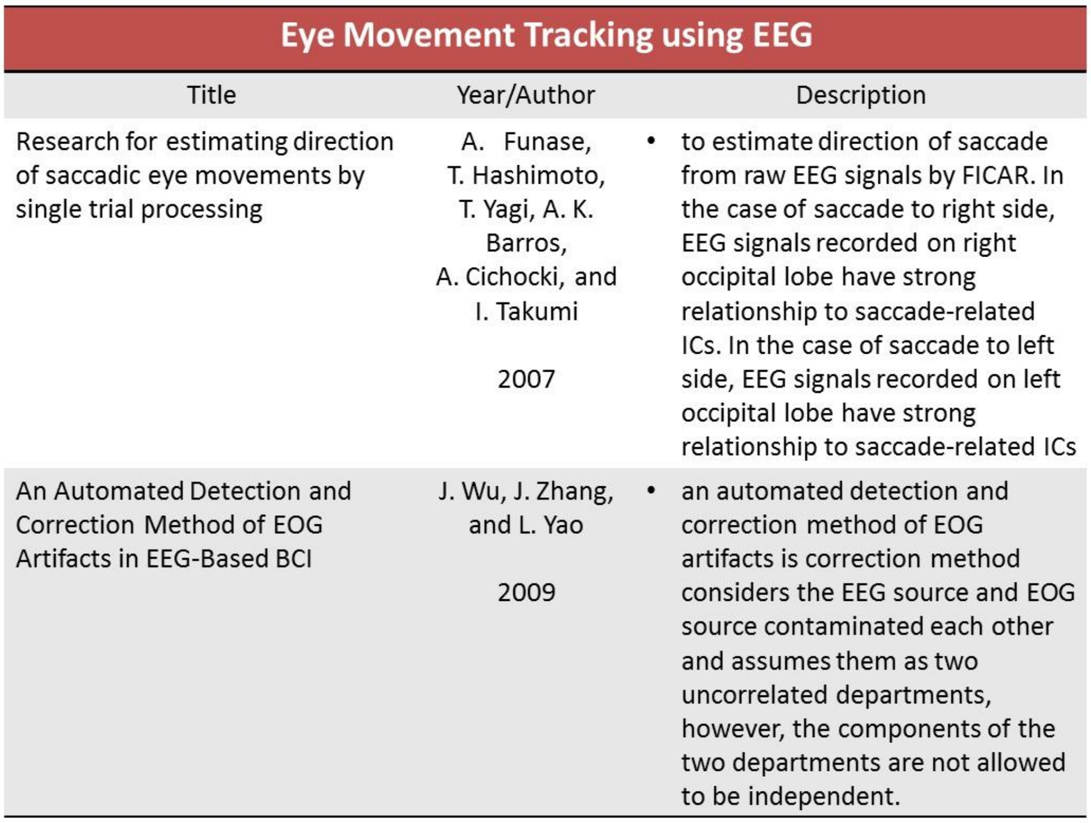
## 2.2 Brain's Signal from Patients and Eye Tracking using EOG

Major parts of the worlds are suffering from chronic neurological disorder. According to [3] , almost 80% of people in developing countries suffers from epilepsy which is one of the worrisome neurological disorder and this statement also are agreed by [4] which state that almost 60 million people across the globe are affected by this disease making it second most chronic neurological disorder after stroke. Methods to treat these diseases using EEG are being developed. As mentioned by [4], they are extracting the EEG signal to detect potential epilepsy brain tissue using High Frequency Epileptiform Oscillation from a few potential patients. Therefore, to help patients or potential patients that suffers from diseases by using EEG. However, for people with physical disabilities and also speech skills problem such as paralyzed patient, communicating with people the normal way might be troublesome. Thus, eye tracking movement might be one of the solutions. In some research conducted by [5], the EOG (electrooculography), which are the signals emitted from the eye, are extracted and remove from EEG signals to avoid interference with the actual EEG signals. But, a research carried out by [6] make use of these EOG signals and perform a simulation in BCI application that follows the same eye movement perform by the subject. Though the eye movement simulation can be perform by the BCI, but the movement only restricted to left and right only.



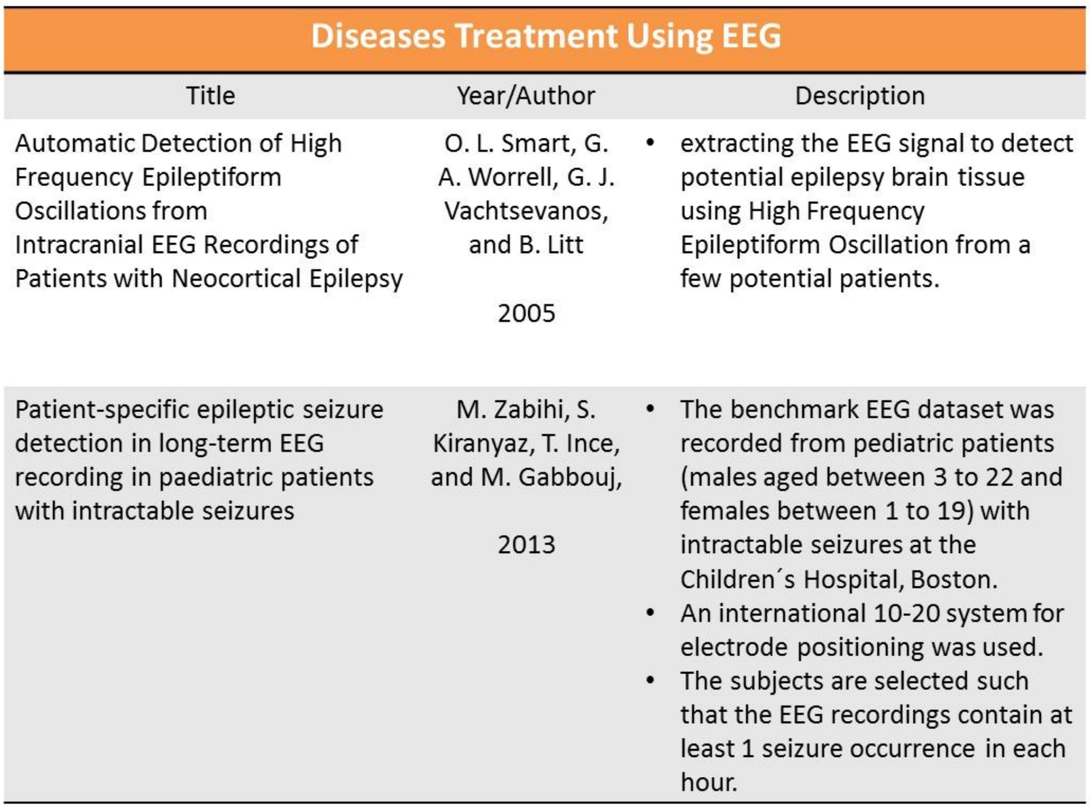
| Recording Brain's Signal | | |
|---|---|---|
| Title | Year/Author | Description |
| Identify the Reference Signal of Scalp EEG Recording | H. Sanqing, C. Shihui, Z. Jianhai, K. Wanzeng, and C. Yu<br><br>2013 | • select the earlobe of one subject as the reference site, propose one special experimental method for the subject by taping the reference electrode about once every second and record scalp EEG for some time period. |
| Development of Real-Time Evaluation System for Qualitative Improvement of Awake EEG Records | T. Sugi, T. Nagamine, M. Nakamura, A. Ikeda, and H. Shibasaki<br><br>2012 | • The quality of recorded EEG is maintained by the EEG technologist effort. The length of usual EEG recordings depends on the purpose of the analysis and diagnosis. It is a laborious work and requires concentration for maintaining the quality of recorded EEG.<br>• Thus using some computer-assisted tool will be an effective and helpful. |

**Table 1: Summary of Literature Review for Recording Signal**



| Eye Movement Tracking using EEG | | |
|---|---|---|
| Title | Year/Author | Description |
| Research for estimating direction of saccadic eye movements by single trial processing | A. Funase, T. Hashimoto, T. Yagi, A. K. Barros, A. Cichocki, and I. Takumi<br><br>2007 | • to estimate direction of saccade from raw EEG signals by FICAR. In the case of saccade to right side, EEG signals recorded on right occipital lobe have strong relationship to saccade-related ICs. In the case of saccade to left side, EEG signals recorded on left occipital lobe have strong relationship to saccade-related ICs |
| An Automated Detection and Correction Method of EOG Artifacts in EEG-Based BCI | J. Wu, J. Zhang, and L. Yao<br><br>2009 | • an automated detection and correction method of EOG artifacts is correction method considers the EEG source and EOG source contaminated each other and assumes them as two uncorrelated departments, however, the components of the two departments are not allowed to be independent. |

**Table 2: Summary of Literature Review for Eye Movement Tracking using EEG**



| Diseases Treatment Using EEG | | |
|---|---|---|
| Title | Year/Author | Description |
| Automatic Detection of High Frequency Epileptiform Oscillations from Intracranial EEG Recordings of Patients with Neocortical Epilepsy | O. L. Smart, G. A. Worrell, G. J. Vachtsevanos, and B. Litt<br><br>2005 | • extracting the EEG signal to detect potential epilepsy brain tissue using High Frequency Epileptiform Oscillation from a few potential patients. |
| Patient-specific epileptic seizure detection in long-term EEG recording in paediatric patients with intractable seizures | M. Zabihi, S. Kiranyaz, T. Ince, and M. Gabbouj,<br><br>2013 | • The benchmark EEG dataset was recorded from pediatric patients (males aged between 3 to 22 and females between 1 to 19) with intractable seizures at the Children´s Hospital, Boston.<br>• An international 10-20 system for electrode positioning was used.<br>• The subjects are selected such that the EEG recordings contain at least 1 seizure occurrence in each hour. |

**Table 3: Summary of Literature Review for Diseases Treatment using EEG**

## 2.3 SSVEP

Steady State Visually Evoked Potentials (SSVEP) is signals that will be produced when the visual stimulation of the brain are exposed to specific frequencies. The brain will produce electrical signals that contain the same frequencies with the one the retina is being exposed to. The retina will be excited usually by a visual stimulus ranging from 3.5 Hz to 75 Hz,[7]. Thus, when our eyes are watching a visual with 3.5Hz frequency, our brain will produce the same electrical signal with the same frequency which is 3.5Hz.

Electroencephalographic researches regarding vision are usually manipulating this type of signal in their studies. It is useful in research because of the signal-to-noise ratio efficiency and comparative invulnerability to artifacts [7].

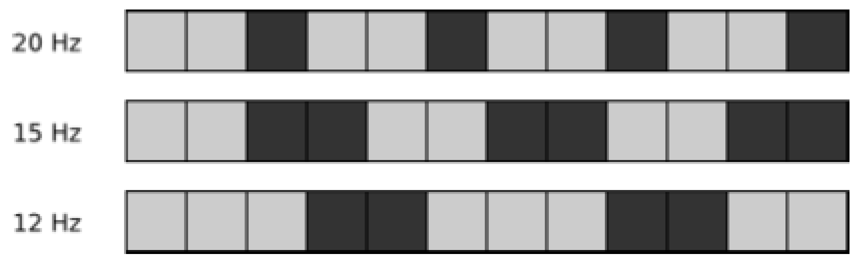


**Figure 3: Example of Several Frequencies**

By using SSVEP based BCI application, visual stimulus with different frequencies is simultaneously shown to the subject. The subject will focus his attention  into a screen that display certain patterns of visual  and the matching stimulating frequencies will appear in the recording of the EEG signal's spectral. After the EEG signal are analyzed and processed, it will then be turned into desired action or output.

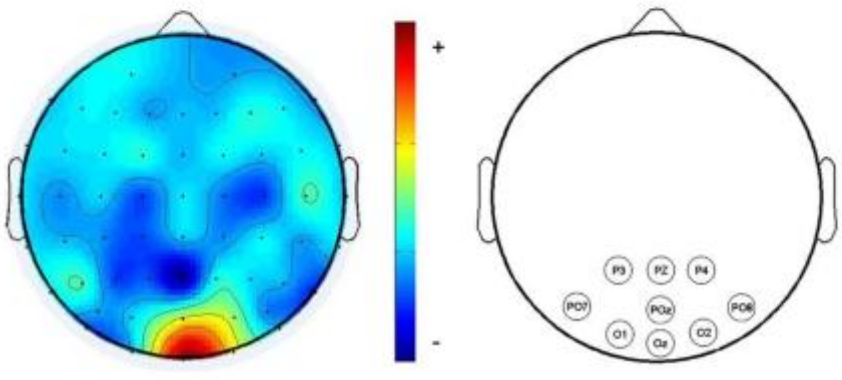


**Figure 4: EEG Topography of the SSVEP Effect on visual cortex**

## 2.4 Open vibe

OpenViBE is a platform that is established to design, test and use the brain-computer interfaces. It is software for real-time neurosciences meaning that the process of acquiring, filtering, processing, classifying and visualizing brain signals are always conducted in real time. OpenViBE is also free and open source software that works perfectly with Windows and Linux operating systems [8]. Mainly OpenViBE application focuses on medical fields. It aims to assist disabled patients with real time biofeedback, neurofeedback and diagnosis. It also provides function for other application field such as multimedia for developing virtual reality program or video games, robotics and all other fields that related [8].

Openvibe is an easy platform as all the signal processing algorithms are denoted by blocks. Each block can be simulating with each other by just linking them with a connector. Simulink Boxes in MATLAB also serves the same purpose but Openvibe offers more convenient and user friendly interface especially when dealing with real-time neurofeedback.

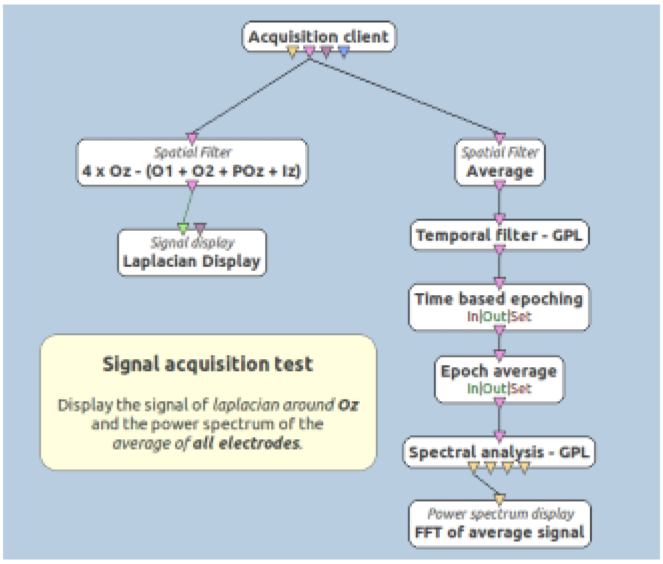


Figure 5: Open vibe Graphical Interface

### 2.4.1 Classification and Feature extraction

EEG signals retrieved are usually low in frequencies and it is hard to distinguish between one signal with another signal. Thus to increase the signal change from one another, Common Spatial Pattern (CSP) algorithm are used.

Box below that provided in Openvibe figure spatial filters according to the CSP algorithm that helps to increase the discrimination between two types of events that occurs simultaneously.

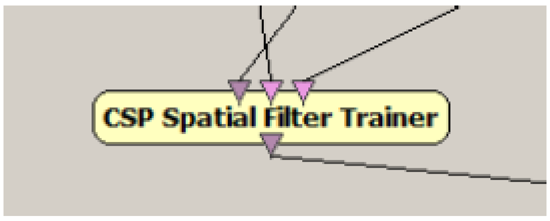


Figure 6: Spatial Filter Trainer

# Chapter 3: Methodology/Project Work

## 3.1 Research Methodology

The methodology implemented in this project is based on responsive development approach. It aims to find the best yet dynamic approach solutions with operation of iterative and incremental development that promotes adaptive planning, imaginative progress and well-organized delivery.

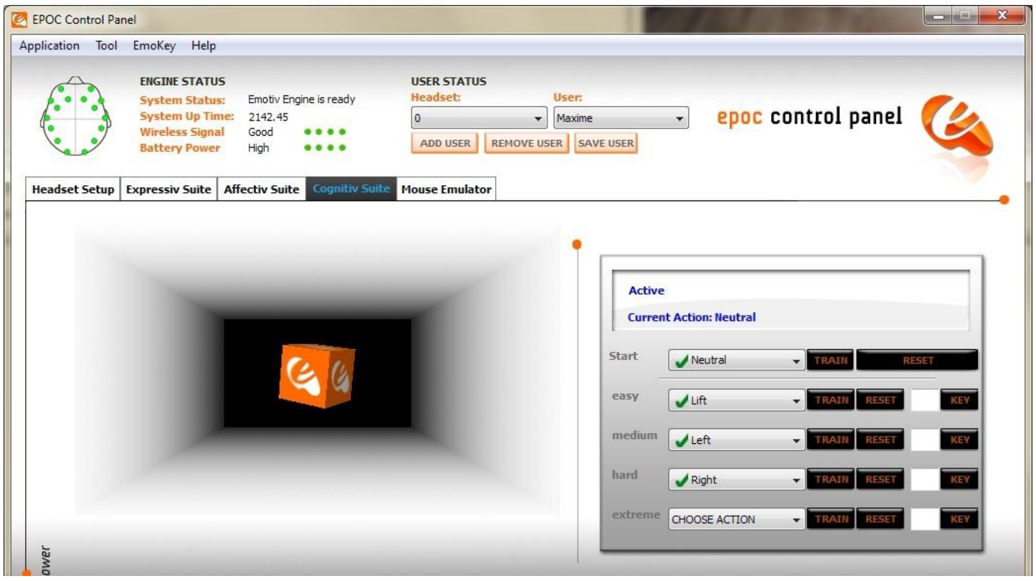
## 3.2 Tools

### 3.2.1 Software:

There is some software and development tools will be used to complete the project

 **Emotiv SDK (Software Development Kit)**

EMOTIV EEG Headset provided in the lab is one of the important equipment that needed to be use. There are 3 suites that can be used in this application. The expressive suite, affective suite and cognitive suite.  This SDK is completed with a high resolution images and wireless neuroheadset connection. The API of this SDK can be used with numerous programming languages such as Visual studio, java and python that can produce numerous creative application of neuroscience.


ENGINE STATUS
System Status: Emotiv Engine is ready
System Up Time: 2142.45
Wireless Signal: Good
Battery Power: High

USER STATUS
Headset: 0
User: Maxime

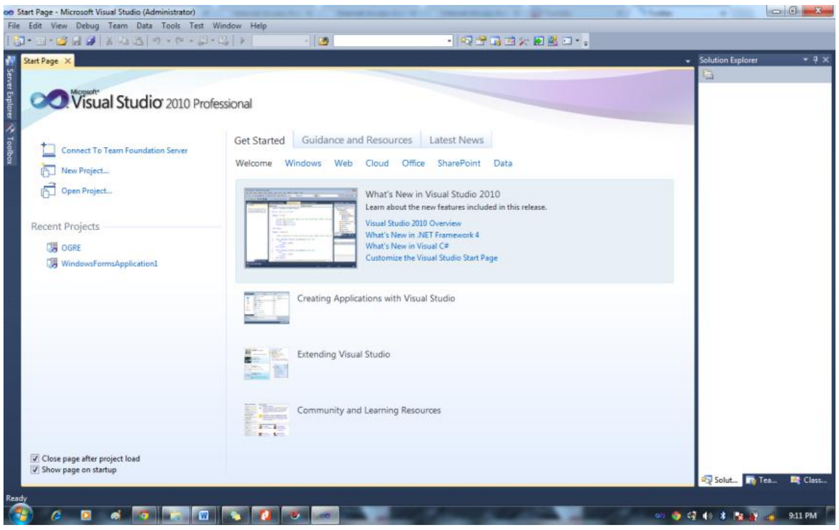
Active
Current Action: Neutral

| | | |
|---|---|---|
| Start | Neutral | |
| easy | Lift | |
| medium | Left | |
| hard | Right | |
| extreme | CHOOSE ACTION | |

Figure 7: Cognitiv Suite

 **Visual Studio (2010)**

LUA programming language is mainly used in this project. It can be viewed and edited using Microsoft Visual Studio (2010) . LUA programming is easy and user-friendly. Nowadays, many developers especially the one that dealing with neuroscience technologies prefers to write in LUA programming language because of its simplicity and dynamic response.



Start Page - Microsoft Visual Studio (Administrator)
File Edit View Debug Team Data Tools Test Window Help
Start Page
Visual Studio 2010 Professional
Connect To Team Foundation Server
New Project...
Open Project...
Recent Projects
OGRE
WindowsFormsApplication1
Get Started
Guidance and Resources
Latest News
Welcome Windows Web Cloud Office SharePoint Data
What's New in Visual Studio 2010
Learn about the new features included in this release.
Visual Studio 2010 Overview
What's New in .NET Framework 4
What's New in Visual C#
Customize the Visual Studio Start Page
Creating Applications with Visual Studio
Extending Visual Studio
Community and Learning Resources
Close page after project load
Show page on startup
Solution Explorer
Ready
9:11 PM

Figure 8: Microsoft Visual Studio 2010

- **Open Vibe:**

OpenViBE is a platform that is established to design, test and use the brain-computer interfaces. It is software for real-time neurosciences meaning that the process of acquiring, filtering, processing, classifying and visualizing brain signals are always conducted in real time.

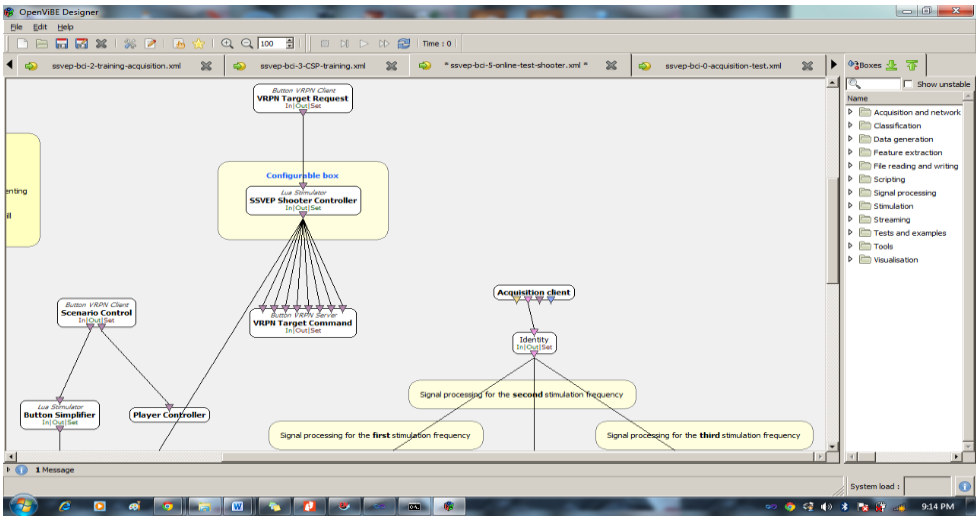


Figure 9: Openvibe Designer

 **VRPN (Virtual-Reality Peripheral Network)**

The Virtual-Reality Peripheral Network (VRPN) is a set of classes within a library and a set of servers. For a virtual-reality (VR) system, VRPN are used as a transparent network that connect the application program to external application regardless if they are physical or within the network itself.  A declared host at each VR station controls the peripherals and VRPN provide the connection between the application and all devices [9].  For this project, VRPN are used to connect keyboard stimulator with the Lua Stimulator in Openvibe. It also is used to connect Openvibe with external application which is OGRE.

- **OGRE**

OGRE (**O**bject-Oriented **G**raphics **R**endering **E**ngine) is a platform created to develop 3D graphics. It scene-oriented with flexible 3D engine written in C++ that is designed to make it simpler and easier for developers to produce applications that employ hardware-accelerated 3D graphics [10]. In this project, OGRE is used to develop a graphic simulation of a spaceship with flickering stimulus. It is then integrated with Openvibe that uses EEG signal as an input and perform certain action based on the input signals.

### 3.2.2 Hardware:

 **Emotiv  EPOC**

The Emotiv EPOC is a 14 channels headset. It uses 14 different electrodes that are scattered in different but specific parts around the scalp to read and retrieve electrical signals emitted by the brain so that it can be monitored and detected the user's thoughts (Cognitive), feelings and expressions (expressive).[11]

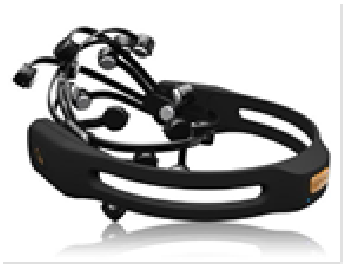
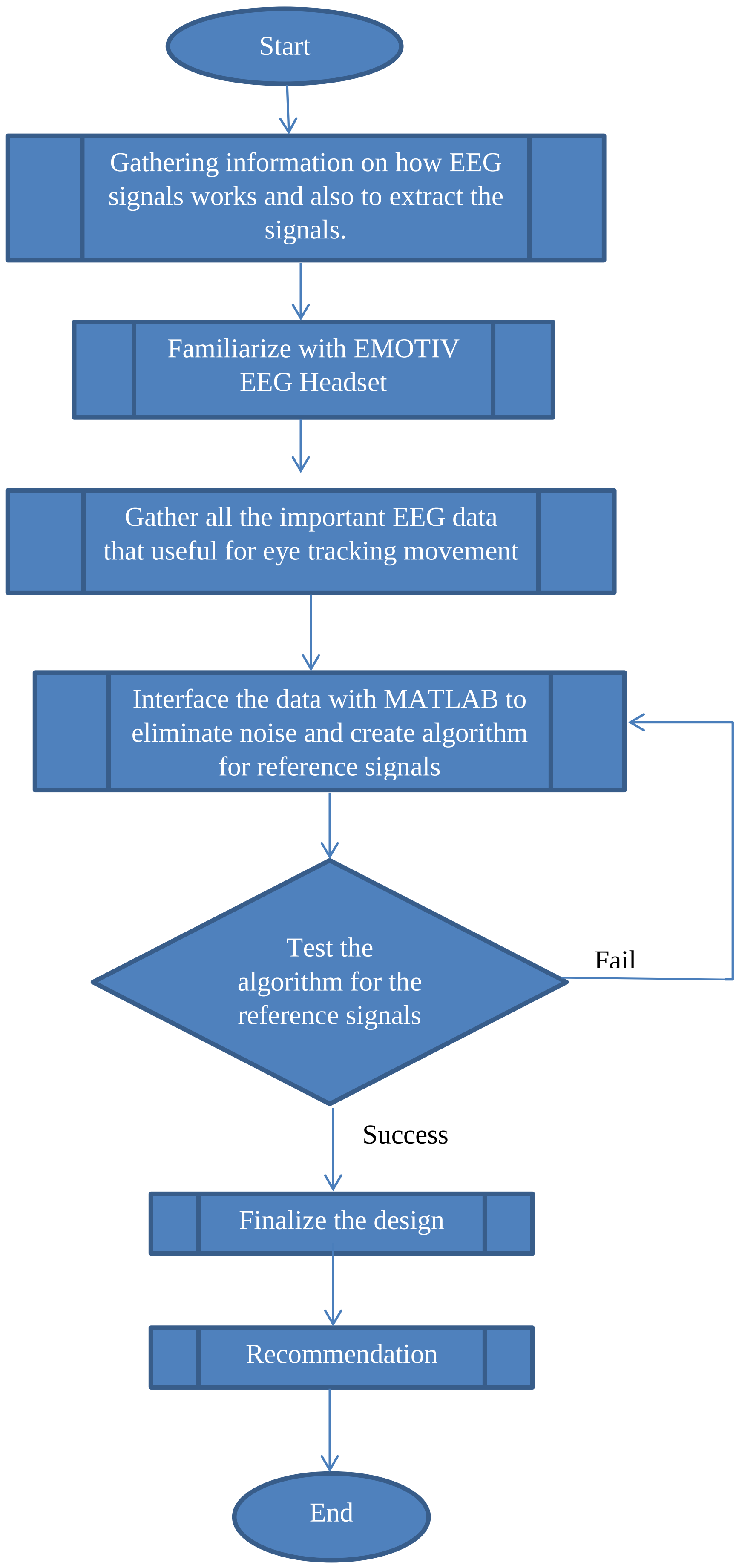


Figure 10: EMOTIV EPOC Headset

## 3.3 Project Flowchart


Start
Gathering information on how EEG signals works and also to extract the signals.
Familiarize with EMOTIV EEG Headset
Gather all the important EEG data that useful for eye tracking movement
Interface the data with MATLAB to eliminate noise and create algorithm for reference signals
Test the algorithm for the reference signals
Fail
Success
Finalize the design
Recommendation
End

## 3.4 Gantt Chart and Key Milestone

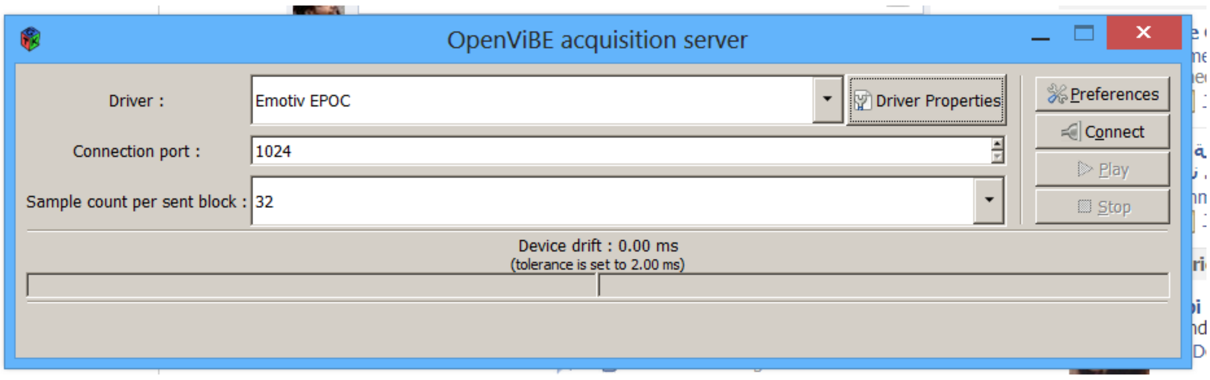
Gantt chart and Key Milestone for FYP 1 and FYP 2:

| Activities | FYP 1 | | | | | | | | | | | | | | FYP 2 | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Weeks No. | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| Title Proposal | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Preliminary Research/ Data Collection | | | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | | |
| Extended Proposal | | | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | | | | | | | | |
| Data Gathering from EMOTIV EEG Headset | | | | | | | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | |
| Proposal Defend | | | | | | | | 🟦 | | | | | | | | | | | | | | | | | | | | |
| Analyzing Data Gathered | | | | | | | | | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | | |
| Interfaces with MATLAB | | | | | | | | | | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | | | | | | | | | | |
| Interim Report | | | | | | | | | | | | | 🟨 | 🟦 | | | | | | | | | | | | | | |
| Progress Report | | | | | | | | | | | | | | | 🟦 | | | | | | | | | | | | | |
| Finalize algorithm | | | | | | | | | | | | | | | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟦 | | | | | | |
| Testing Algorithm | | | | | | | | | | | | | | | | | | | | | | | 🟨 | 🟨 | 🟨 | 🟨 | 🟨 | 🟦 |
| Project Dissertation | | | | | | | | | | | | | | | | | | | | | | | | | | | | 🟦 |

# Chapter 4: Result and Discussions

## 4.1 Extracting EEG Signal from EMOTIV using OpenVibe

OpenVibe is one of user friendly platform especially for BCI application. Using this tool, EEG signal can be easily extracted and manipulated according to the desired environment.


OpenViBE acquisition server

Driver : Emotiv EPOC

Connection port : 1024

Sample count per sent block : 32

Device drift : 0.00 ms
(tolerance is set to 2.00 ms)

Figure 11: Acquisition Server

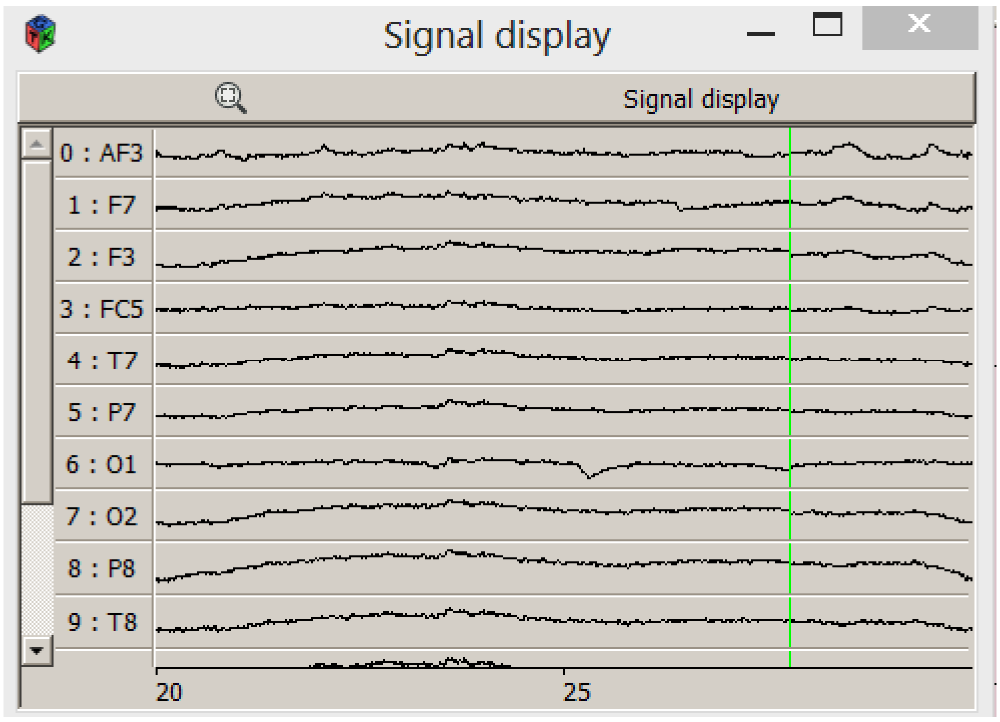
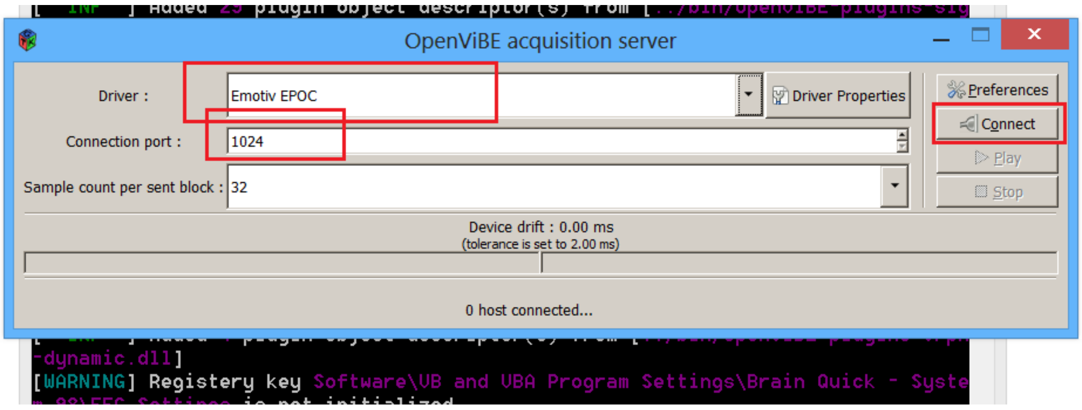


Figure 12: Sample of extracted EEG signal

The type of the acquisition device should be carefully chosen as it will be used and then ported to the Design framework. As Openvibe's acquisition server is already equipped with EMOTIV EPOC driver, user can easily select the interface and declare the local host for this connection. It is important to declare the local host accordingly, as the same local host is used when designing and running the scenarios from Openvibe.



```
[  INF  ] Added 29 plugin object descriptor(s) from [../bin/OpenViBE-plugins-sig
-dynamic.dll]
[WARNING] Registery key Software\UB and UBA Program Settings\Brain Quick - Syste
```

OpenViBE acquisition server

Driver : Emotiv EPOC

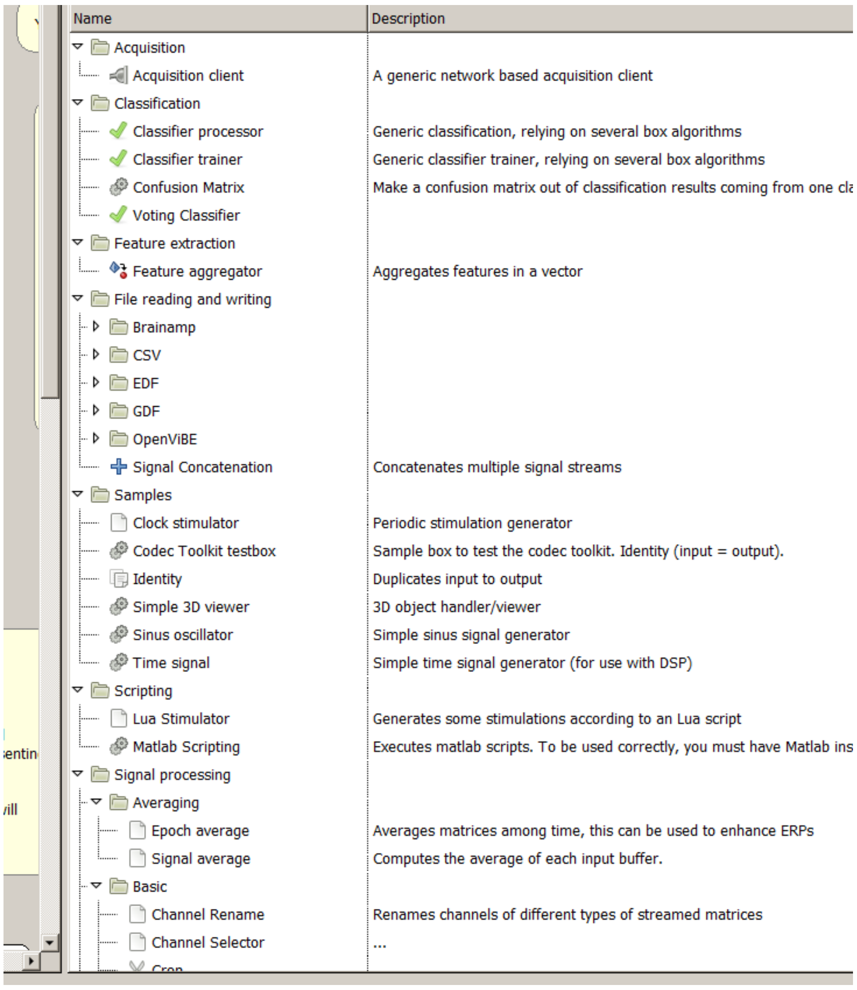
Connection port : 1024

Sample count per sent block : 32

Device drift : 0.00 ms
(tolerance is set to 2.00 ms)

0 host connected...

Figure 13: OpenVibe Acquisition Server



| Name | Description |
|---|---|
| Acquisition | |
| Acquisition client | A generic network based acquisition client |
| Classification | |
| Classifier processor | Generic classification, relying on several box algorithms |
| Classifier trainer | Generic classifier trainer, relying on several box algorithms |
| Confusion Matrix | Make a confusion matrix out of classification results coming from one cl |
| Voting Classifier | |
| Feature extraction | |
| Feature aggregator | Aggregates features in a vector |
| File reading and writing | |
| Brainamp | |
| CSV | |
| EDF | |
| GDF | |
| OpenViBE | |
| Signal Concatenation | Concatenates multiple signal streams |
| Samples | |
| Clock stimulator | Periodic stimulation generator |
| Codec Toolkit testbox | Sample box to test the codec toolkit. Identity (input = output). |
| Identity | Duplicates input to output |
| Simple 3D viewer | 3D object handler/viewer |
| Sinus oscillator | Simple sinus signal generator |
| Time signal | Simple time signal generator (for use with DSP) |
| Scripting | |
| Lua Stimulator | Generates some stimulations according to an Lua script |
| Matlab Scripting | Executes matlab scripts. To be used correctly, you must have Matlab ins |
| Signal processing | |
| Averaging | |
| Epoch average | Averages matrices among time, this can be used to enhance ERPs |
| Signal average | Computes the average of each input buffer. |
| Basic | |
| Channel Rename | Renames channels of different types of streamed matrices |
| Channel Selector | ... |

Figure 14: Sample of OpenVibe Module

## 4.2 Training the data

Below configured environment are used to play a scenario that helps to get the desired EEG signals.

1) VRPN client are used to control the scenario. Keyboard stimulator is interface with Openvibe using VRPN client. The scenario will only be played when the spacebar of the keyboard are pressed.
2) SSVEP Training Controller is used to control the scenario played. It can be used to modify the frequency of the flickering stimulus, color of the boxes and the delay between each box.
3) Generic Stream Writer is used to record the acquired signals after the scenarios finish playing.

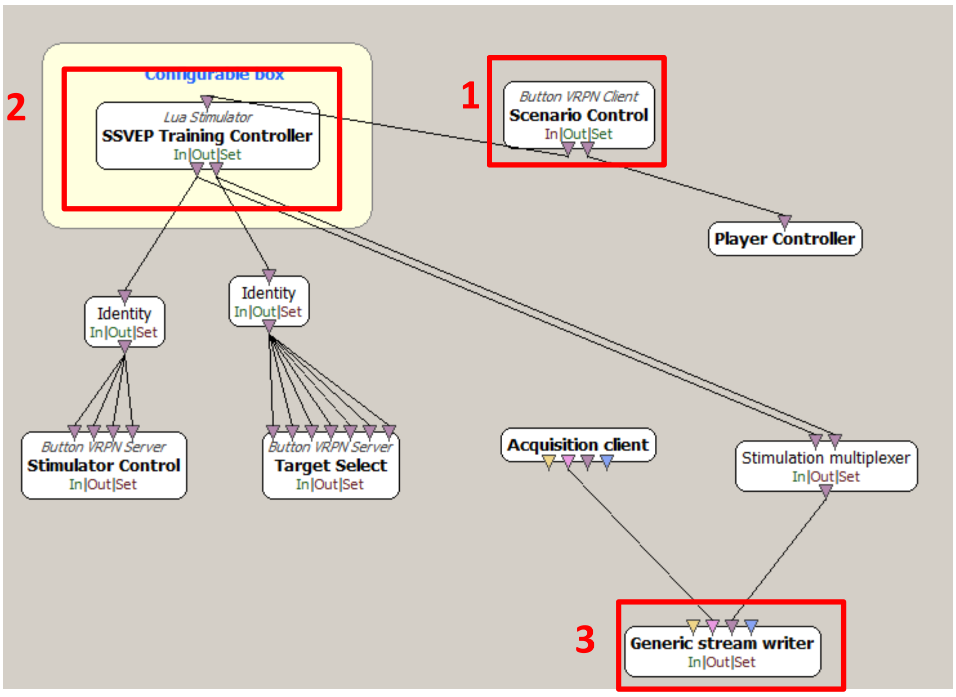


Figure 15: Environment created to train the data

### 4.2.1Training Acquisition

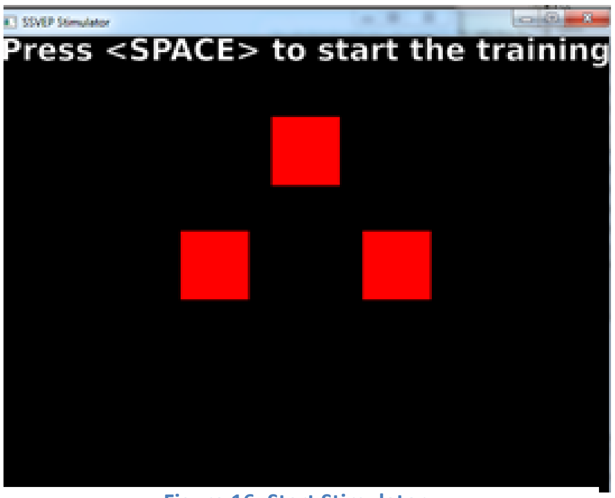


Figure 16: Start Stimulator

Subject will be asked to focus their visual to different flickering frequency object. Each flickering frequencies are actually giving out different SSVEP. With different SSVEP the eye movement can be track and distinguish. This training will take about 10 to 15 minutes to be completed and it also required high focus intensity from the subject.

## 4.3 Feature Extraction

Certain features are extracted from the digitized EEG signal in this stage. The desired frequency range is extracted and the amplitude relative to some reference level is measured. The extracted features can be denoted as some frequency bands on the power spectrum. It will be extremely difficult to classify those mental tasks even with a well-executed classifier if the feature combinations are representing mental overlap each other. But, if the feature sets are not over lapping and different, most of the classifiers can be used efficiently to classify them.

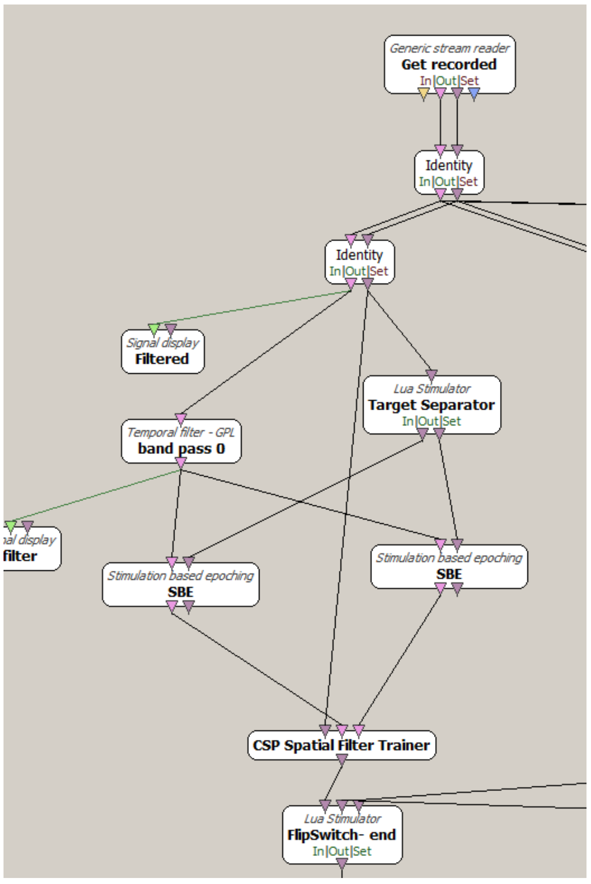


Figure 17: Environment for Feature Extraction

## 4.4 Classifier Training

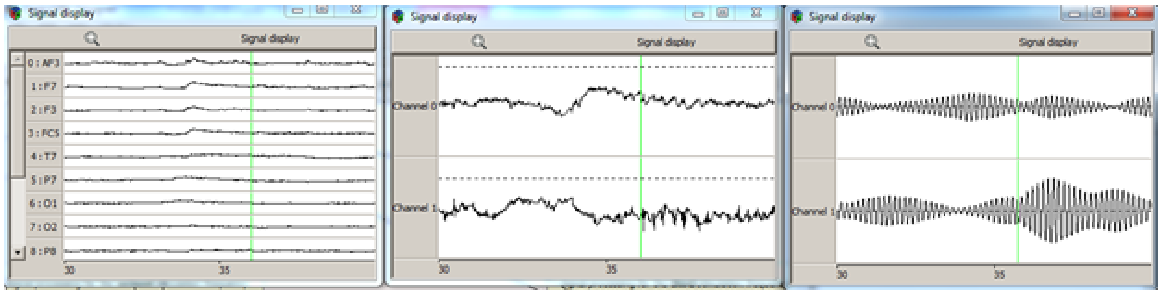
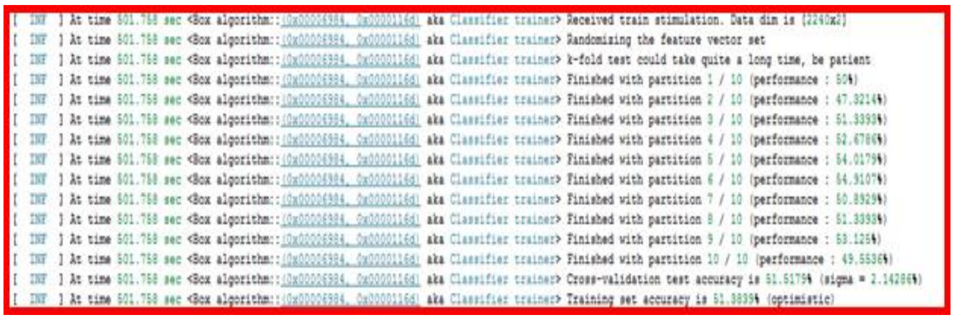


**Figure 18: From left (Signal acquired, Filtered Signal using CSP filter, Filtered Signal using Low Pass Filter)**

The classifier training is used to classify the useful signal. Common Spatial Pattern (CSP) filter is a filter that is developed from Common Spatial Pattern algorithm. The goal of the algorithm is to improve the discrimination of two types of signals. The spatial filters are constructed in a way they maximize the variance for signals of the first condition while at the same time they minimize it for the second condition. This can be used for discriminating the signals of two commonly used motor-imagery tasks (e.g. left versus right hand movement). It can also be used for two-class SSVEP experiments or any other experiment where the discriminative information is contained in the variance (or power in a certain band) of the signal conditions. Thus, CSP filter is used to increases the signal variance for one condition while minimizing the variance for the other condition. This training will take around 10 minutes to complete.

## 4.5 Performance Results



```
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Received train stimulation. Data dim is (2240x2)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Randomizing the feature vector set
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> k-fold test could take quite a long time, be patient
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 1 / 10 (performance : 50%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 2 / 10 (performance : 47.3214%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 3 / 10 (performance : 51.3393%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 4 / 10 (performance : 52.6786%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 5 / 10 (performance : 54.0179%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 6 / 10 (performance : 54.9107%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 7 / 10 (performance : 50.8929%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 8 / 10 (performance : 51.3393%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 9 / 10 (performance : 53.125%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Finished with partition 10 / 10 (performance : 49.5536%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Cross-validation test accuracy is 51.5179% (sigma = 2.14286%)
[  INF  ] At time 501.758 sec <Box algorithm::(0x00006984, 0x0000116d)> aka Classifier trainer> Training set accuracy is 51.3839% (optimistic)
```

Figure 19: The performance of the trained signals

After classifier training, the performance of the signal will be displayed. Higher percentage of the performance indicates that the signal is very useful and vice versa. However, the highest signal can be obtained from now is around 70% which is still lacking as 80% performance is needed.

## 4.6 Online Test

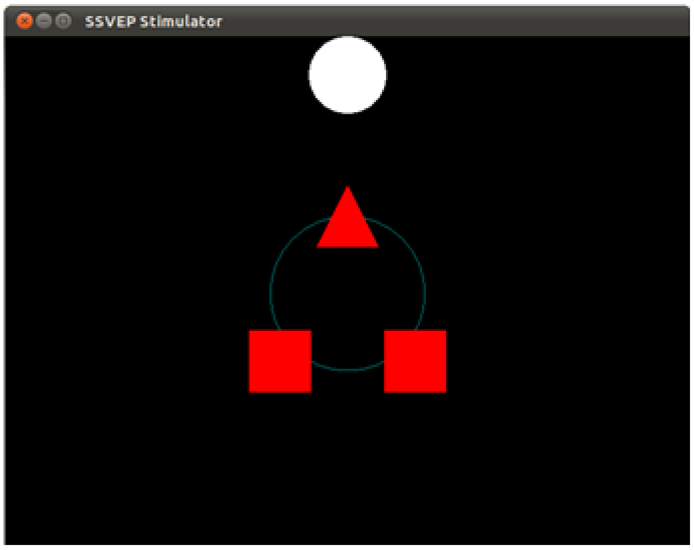


Figure 20: SSVEP Stimulator

A scenario is created to test the signal acquired from previous step.  With different flickering frequency, subject are expected to move the object accordingly using only their focus; left square (clockwise), right square (anti-clockwise), triangle (to shoot the white target). However, with poor training performance (<80%), it is quite hard to move the object accordingly.

# Chapter 5: Conclusion and Recommendations

## 5.1 Conclusion

In conclusion, interfaces that can track eye movement and translate it into the desired movement are successfully built. However, it takes more times to work on the programming language that can create and simulate the eye movement into a drawing.

Also, instead of using MATLAB, new software that is user friendly, more dynamic, cost-free, and focused only neurofeedback signals are successfully integrated and implemented in this project.

It is also noticed that even with the same subject, the performance of the signal can be vary according to their state of emotion and ability to remain focused on a long time. Thus the system is not universal and a lot of users will face issue when using the system.

## 5.2 Recommendations

Future works that can be worked on in other to fulfill the objective are:

1) Develop algorithm that can complete the training acquisition data in short time with accurate result of the EEG signals.

2) Develop another framework using OGRE that is reliable and efficiently working for a hands-free drawing using EEG system.

3) Find a new method and algorithm that can distinguish the variation of the signal more effectively other than Common Spatial Pattern filter.

4) Make a new programming course that focused on LUA programming that is really useful for BCI application in the future.

## References

[1] Y. Han and H. Bin, "Computer Interfaces Using Sensorimotor Rhythms: Current State and Future Perspectives," *Biomedical Engineering, IEEE Transactions on,* vol. 61, pp. 1425-1435, 2014.

[2] Z. N. M. R.J. Huster, S. Enriquez-Geppert, C. Herrmann, "Brain-computer interfaces for EEG neurofeedback: Peculiarities and solutions," *International Journal of Psychophysiology,* , vol. 91, pp. 36–45, 2014.

[3] M. Zabihi, S. Kiranyaz, T. Ince, and M. Gabbouj, "Patient-specific epileptic seizure detection in long-term EEG recording in paediatric patients with intractable seizures," in *Intelligent Signal Processing Conference 2013 (ISP 2013), IET*, 2013, pp. 1-7.

[4] O. L. Smart, G. A. Worrell, G. J. Vachtsevanos, and B. Litt, "Automatic detection of high frequency epileptiform oscillations from intracranial EEG recordings of patients with neocortical epilepsy," in *Technical, Professional and Student Development Workshop, 2005 IEEE Region 5 and IEEE Denver Section*, 2005, pp. 53-58.

[5] J. Wu, J. Zhang, and L. Yao, "An automated detection and correction method of EOG artifacts in EEG-based BCI," in *Complex Medical Engineering, 2009. CME. ICME International Conference on*, 2009, pp. 1-5.

[6] A. Funase, T. Hashimoto, T. Yagi, A. K. Barros, A. Cichocki, and I. Takumi, "Research for estimating direction of saccadic eye movements by single trial processing," in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, 2007, pp. 4723-4726.

[7] Fabrizio Beverina, Giorgio Palmas, Stefano Silvoni, Francesco Piccione, and S. Giove, "User adaptive BCIs:SSVEP and P300 based interfaces " *PsychNology* vol. 1, pp. 331-354, 2003.

[8] Openvibe.inria.fr.OpenViBE. Software for Brain Computer Interfaces and Real TimeNeurosciences [Online]. Available: http://openvibe.inria.fr/ [

[9] V. R. P. Network. (22nd September). *VRPN 07.32|Virtual Reality Peripheral Network*. Available: http://www.cs.unc.edu/Research/vrpn/

[10] OGRE. (2001, 23rd September). *OGRE*. Available: http://www.ogre3d.org/

[11] A. Kawala-Janik, M. Podpora, M. Pelc, P. Piatek, and J. Baranowski, "Implementation of an inexpensive EEG headset for the pattern recognition purpose," in *Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), 2013 IEEE 7th International Conference on*, 2013, pp. 399-403.

## Appendices

### 1) Coding for training acquisition controller using LUA,

```
sequence = {}

number_of_cycles = 0


stimulation_duration = nil

break_duration = nil

flickering_delay = nil


target_width = nil

target_height = nil


target_positions = {}

number_of_targets = {}


stimulationLabels = {

        0x00008100,

        0x00008101,

        0x00008102,

        0x00008103,

        0x00008104,

        0x00008105,

        0x00008106,

        0x00008107

}


function initialize(box)


        dofile(box:get_config("${Path_Data}") .. "/plugins/stimulation/lua-stimulator-stim-
codes.lua")
```

```
	-- load the goal sequence
	s_sequence = box:get_setting(2)



	for target in s_sequence:gmatch("%d+") do
		table.insert(sequence, target)
		number_of_cycles = number_of_cycles + 1
	end

	box:log("Info", string.format("Number of goals in sequence: [%d]",
number_of_cycles))

	-- get the duration of a stimulation sequence
	s_stimulation_duration = box:get_setting(3)

	if (s_stimulation_duration:find("^%d+[.]?%d*$") ~= nil) then
		stimulation_duration = tonumber(s_stimulation_duration)
		box:log("Info", string.format("Stimulation Duration : [%g]",
stimulation_duration))
	else
		box:log("Error", "The parameter 'stimulation duration' must be a numeric
value\n")
		error()
	end

	-- get the duration of a break between stimulations
	s_break_duration = box:get_setting(4)

	if (s_break_duration:find("^%d+[.]?%d*$") ~= nil) then
		break_duration = tonumber(s_break_duration)
		box:log("Info", string.format("Break Duration : [%s]", s_break_duration))
	else
		box:log("Error", "The parameter 'break duration' must be a numeric
value\n")
		error()
	end


	-- get the delay between the appearance of the marker and the start of flickering
	s_flickering_delay = box:get_setting(5)


	if (s_flickering_delay:find("^%d+[.]?%d*$") ~= nil) then
		flickering_delay = tonumber(s_flickering_delay)
		box:log("Info", string.format("Flickering Delay : [%s]", s_flickering_delay))
	else
		box:log("Error", "The parameter 'flickering delay' must be a numeric 
value\n")
		error()
	end


	-- get the target size


	s_targetSize = box:get_setting(6)


	s_width, s_height = s_targetSize:match("^(%d+[.]?%d*);(%d+[.]?%d*)$")
	target_width = tonumber(s_width)
	target_height = tonumber(s_height)


	if s_width ~= nil and s_height ~= nil then
		box:log("Info", string.format("Target dimensions : width = %g, height = 
%g", target_width, target_height))
	else
		box:log("Error", "The parameter 'target size' must be in format float;float")
		error()
	end


	-- get the targets' positions


	s_targetPositions = box:get_setting(7)
	number_of_targets = 0
	for s_target_x, s_target_y in s_targetPositions:gmatch("(-?%d+[.]?%d*);(-
?%d+[.]?%d*)") do
		box:log("Info", string.format("Target %d : x = %g y = %g",
number_of_targets, tonumber(s_target_x), tonumber(s_target_y)))
		table.insert(target_positions, {tonumber(s_target_x), tonumber(s_target_y)})
		number_of_targets = number_of_targets + 1

	end



	-- create the configuration file for the stimulation-based-epoching
	-- this file is used during classifier training only
	cfg_file_name =
box:get_config("${Player_ScenarioDirectory}/configuration/stimulation-based-
epoching.cfg")
	cfg_file = io.open(cfg_file_name, "w")
	if cfg_file == nil then
		box:log("Error", "Cannot write to [" .. cfg_file_name .. "]")
		box:log("Error", "Please copy the scenario folder to a directory with write
access and use from there.")
		return false
	end

	cfg_file:write("<OpenViBE-SettingsOverride>\n")
	cfg_file:write("  <SettingValue>", stimulation_duration, "</SettingValue>\n")
	cfg_file:write("  <SettingValue>", flickering_delay, "</SettingValue>\n")
	cfg_file:write("  <SettingValue>OVTK_StimulationId_Target</SettingValue>\n")
	cfg_file:write("</OpenViBE-SettingsOverride>\n")

	cfg_file:close()
```

```
	-- create the configuration file for the training program
	cfg_file =
io.open(box:get_config("${CustomConfigurationPrefix${OperatingSystem}}-ssvep-demo-
training${CustomConfigurationSuffix${OperatingSystem}}"), "w")
```

```
        cfg_file:write("# This file was automatically generated!\n\n")
        cfg_file:write("# If you want to change the SSVEP trainer configuration\n")
        cfg_file:write("# please use the box settings in the training scenario.\n\n")

        cfg_file:write("SSVEP_TargetCount = ", number_of_targets, "\n")
        cfg_file:write("SSVEP_TargetWidth = ", target_width, "\n")
        cfg_file:write("SSVEP_TargetHeight = ", target_height, "\n")

        for target_index, position in ipairs(target_positions) do

                cfg_file:write("SSVEP_Target_X_", target_index - 1, " = ", position[1],
"\n")
                cfg_file:write("SSVEP_Target_Y_", target_index - 1, " = ", position[2],
"\n")
        end

        cfg_file:close()

end

function uninitialize(box)
end

function process(box)

        while box:keep_processing() and box:get_stimulation_count(1) == 0 do
                box:sleep()
        end

        current_time = box:get_current_time() + 1

        box:send_stimulation(1, OVTK_StimulationId_ExperimentStart, current_time, 0)
```

```
	current_time = current_time + 2


	for i,target in ipairs(sequence) do
		box:log("Info", string.format("Goal no %d is %d at %d", i, target,
current_time))
		-- mark goal
		box:send_stimulation(2, OVTK_StimulationId_LabelStart + target,
current_time, 0)
		-- wait for Flickering_delay seconds
		current_time = current_time + flickering_delay
		-- start flickering
		box:send_stimulation(1, OVTK_StimulationId_VisualStimulationStart,
current_time, 0)
		-- wait for Stimulation_duration seconds
		current_time = current_time + stimulation_duration
		-- unmark goal and stop flickering
		box:send_stimulation(1, OVTK_StimulationId_VisualStimulationStop,
current_time, 0)
		-- wait for Break_duration seconds
		current_time = current_time + break_duration
	end


	box:send_stimulation(1, OVTK_StimulationId_ExperimentStop, current_time, 0)


	box:sleep()
end
```

## 2) Code for classifier training

```
targets = {}
non_targets = {}
sent_stimulation = 0


function initialize(box)
```

```
		dofile(box:get_config("${Path_Data}") .. "/plugins/stimulation/lua-
stimulator-stim-codes.lua")

		-- read the parameters of the box

		s_targets = box:get_setting(2)

		for t in s_targets:gmatch("%d+") do
			targets[t + 0] = true
		end

		s_non_targets = box:get_setting(3)

		for t in s_non_targets:gmatch("%d+") do
			non_targets[t + 0] = true
		end

		sent_stimulation = _G[box:get_setting(4)]

end

function uninitialize(box)
end

function process(box)

		finished = false

		while box:keep_processing() and not finished do

			time = box:get_current_time()

			while box:get_stimulation_count(1) > 0 do
```

```
            s_code, s_date, s_duration = box:get_stimulation(1, 1)
            box:remove_stimulation(1, 1)

            if s_code >= OVTK_StimulationId_Label_00 and s_code <= OVTK_StimulationId_Label_1F then

                received_stimulation = s_code - OVTK_StimulationId_Label_00

                if targets[received_stimulation] ~= nil then
                    box:send_stimulation(1, sent_stimulation, time)
                elseif non_targets[received_stimulation] ~= nil then
                    box:send_stimulation(2, sent_stimulation, time)
                end

            elseif s_code == OVTK_StimulationId_ExperimentStop then
                finished = true
            end
        end

        box:sleep()

    end
```

## 3) Code for Shooter Classifier

```
class_count = 0


function initialize(box)
        dofile(box:get_config("${Path_Data}") .. "/plugins/stimulation/lua-stimulator-stim-codes.lua")
        class_count = box:get_setting(2)
end

function uninitialize(box)
end

function process(box)

        while box:keep_processing() do

            time = box:get_current_time()

            while box:keep_processing() and box:get_stimulation_count(1) > 0
do

                local decision = 0
                local decided = false

                -- check each input
                for i = 1, class_count do
                    -- if the frequency is considered as stimulated
                    if (box:get_stimulation(i, 1) -
OVTK_StimulationId_Label_00 == 1) then
                        if not decided then
                            decision = i
                            decided = true
                        else
                            decision = 0
                        end

                    end
                    box:remove_stimulation(i, 1)
                end
```

```
                                        if decision ~= 0 then

                                                  box:send_stimulation(1,
OVTK_StimulationId_Label_00 + decision - 1, box:get_current_time() + 0.01, 0)

                                        end


                              end

                              box:sleep()

                    end

end
```