Analysis of Computer Games Player Stress Level Using EEG Data

Zulfikar Dharmawan

Man-Machine Interaction Group

TU Delft

August 2007

Delft University of Technology

Analysis of Computer Games Player Stress Level using EEG Data

Master of Science Thesis Report

Zulfikar Dharmawan

1227394

Media and Knowledge Engineering Programme

Man-Machine Interaction Group

Faculty of Electrical Engineering, Mathematics and Computer Science

Delft University of Technology, Netherlands

August 2007

Graduation Committee
Dr. drs L.J.M. Rothkrantz
Dr. Ir. C.A.P.G. van der Mast
Ir. H. Geers

Analysis of Computer Games Player Stress Level using EEG Data

Zulfikar Dharmawan

Student No. 1227394

Delft University of Technology
Faculty of Electrical Engineering, Mathematics, and Computer Science
Department of Mediamatics
Man-Machine Interaction Group
Mekelweg 4
2628CD Delft
The Netherlands

Author email:
`mailto:z.dharmawan@student.tudelft.nl`
`mailto:zulfikar.dharmawan@gmail.com`

Abstract

The goal of our research is to analyze the stress level of a human player during a game session. In this research, we propose a system to classify certain human states recorded by EEG analysis during playing computer games activity. The classification procedure is off-line, thus the recording will not interfere the game. From that recording, we can differentiate certain player states. The games we used for the experiment are different challenges in racing games, chess, and first person shooter with different types of difficulty levels.

We preprocessed the data using Independent Component Analysis to remove mostly eye movement artifacts. Then, we extract several features mostly related to the frequency domain of the signal. Finally using *Waikato Environment for Knowledge Analysis* (WEKA), we tried several classifiers method to know which one give a better result.

In our experiment, we conducted three experiments in which three stress level were compared; no-stress, average and high-stress level. We were able to classify player state with an average of 79.089 % in accuracy level using Decision Tree classifier. We also performed a comparison between classifying 3 user state and with pair-wise classification (only two states). On average, we achieved 78.7864 % for distinguishing three classes of states. While, classifying two-states achieved an average of over than 80 % in accuracy level.

Acknowledgments

I would like to give my best regards and acknowledgements to all the people who helped me in this thesis work. Without their help and guidance it is impossible for me to complete the work. First of all, I would like to thank Dr. Leon Rothkrantz for guiding me in choosing and formulating the topic, stick to it and complete it. Next, I would like to thank all the volunteer participants from IN4015 Neural Network 2007 class for data acquisition.

I would also like to thank Dragos for helping me with the initial experimental setup. Also to all the initial volunteer subjects, Boy, Prama, and Pak Dedy. During the course of the thesis work, I shared the same room with Burak, Kenan, and Eric, thanks for a very nice discussion on Dutch culture and everything.

Thank you Tia for always checking my progress and giving support. Last but not least, thanks to my Mom, Dad, brother and sister at home in Indonesia for all the long-distance support while I am here in the Netherlands.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

One of the driving forces in personal computing is computer games. Computer games can be used as a test case for almost all new invention that is trying to be accepted by consumer. We have experienced a three-dimensional computer graphics these days, which sometimes we took that for granted but in early days it was not until it is implemented in computer games, we know the applicative purpose of it.

At the very basic of the computer game is how it can be controlled by its player. Computer games combine almost all the activity a person could be doing. From movement, thinking, listening, observing certain object and combination of all of that. Although, physically, a player would only be required to use keyboard or joystick or any other input device. Whether we use keyboard, keypad with arrow, joysticks, mice, or other controlling devices, the way the game can be played with enjoyment and yet still challenging is important.

Sometimes we feel that the game is way too easy for us, which eventually make us abandoning the game. In other time or other games, we feel that the game is too difficult to be played. This resulted the game being left out. This calls for a way to evaluate the usability factor of a computer game, which is more than just fill in the blank list of questionnaire. We wanted to go beyond the mind of a computer games player when he is playing the game. This means that we have to know the player current brain activity, which we refer as *state*.

Recognizing human mental state requires us to measure the activity that is performed in our brain. For this purpose we need to make use of the available brain measurement technology. Electroencephalography (EEG) fulfill this need, for it can be done by using relatively not expensive equipments and other practical reasons. From the recorded EEG, we were able to classify which brain activity tells us what state the player is currently at.

The possibilities for application derived from this research is in abundance. Essentially, we can construct an adaptive game playing experience by recognizing player's state. If we were able to determine the player state, we could give feedback directly to a player in a form of increase of difficulty level to make the game more excited.

To a more extended work derived from this research is a combination between brain-controlled application and games. Although this is not the ultimate goal, still from this research we could ensemble a system that accommodate controlling a computer game using only our thought.

Motivated by the aforementioned reasons, we proposed an environment for us to determine player state when he or she is playing a computer game using their own EEG recording or even to a broader scope using a universal recognition system.

1.2 Objectives

Our main objective in this research is to determine computer game player mental state given his or her EEG recording. For that purpose, we break down several research questions for us to tackle the main objective of this research.

1. *How to perform EEG measurement to a subject playing computer games?*

   For the author, the experience of conducting an EEG measurement was the first of its kind. It goes the same for most of the subject. Most of the experiments involving EEG are done in relation to cognitive load or working memory. For this purpose subjects are given a series of test (similar to an intelligent quotient test), in which they have to memorize certain words, shape, or numbers.

   Doing an EEG measurement to a subject playing computer game would require some degree of flexibility. Different than performing an EEG recording in a hospital or other clinical facility for medical purposes, where the subjects demanded to be treated as patients, in our experiment the subjects are healthy.

   If we want to incorporate the use of EEG device or other device to recognize player's brain activity, the setup needs to be as simple as possible but yet still produce a robust and acceptable output.

2. *How to process data gained using EEG devices?*

   The outcome of an EEG measurement device is recording of signals from brain. Although it is possible to do a visual inspection, by viewing the recorded brain signals, to notify which signals tells us what type of brain activity (or other) is performed. Performing a thorough inspection would require an expert eye with years of training and conducting activity scoring to the signals.

   The raw EEG data needs to be processed in order to retrieve information beneath the recordings. Processing (or even before processing) EEG data is performed in several steps. These require the use of basic signal processing. There will be some trade-offs in implementing certain processing steps compared to others.

3. *How to characterize brain signals based on playing computer games?*

   We want to able to know, given a certain recording of data from a player's brain activity to characterize that the player is at a certain mental state. This would require us to conduct this experiment to a different and a number of subjects.

From the acquired data, we notify certain characteristics that can distinguish player mental state. If we combine these characteristic and apply them to new data, we have a classifier that can tell us about the computer game player.

4. *Is EEG recording a suitable method to conduct usability testing in computer games?*

   Common method to do usability evaluation essentially asking the user to give some feedback about the system. Different than acquiring feedback from the user verbally, there are also other methods that tries to get feedback in the psychological or/and physiological form. Included here would be heart rate, blood pressure, and also EEG.

   The use of EEG data to measure usability in computer games is not so common. This research hopefully could propose EEG-based usability testing for an interactive and/or entertainment system such as computer games.

1.3 Methods

To solve the problems mentioned in Section 1.2, we proceed the following way:

- Literature survey from recent development in EEG and related works to find suitable tools
- Selection and adaption of tools for data acquisition and processing
- Perform experiments to record EEG data from voluntary subjects.

In a nutshell, we can differentiate this research into two scopes, the data acquisition and data processing. Data acquisition itself have taken more effort than the other part of this research. Although some study in similar topic consider data acquisition to be taken as granted, we find that it needs to be noted as different entity.

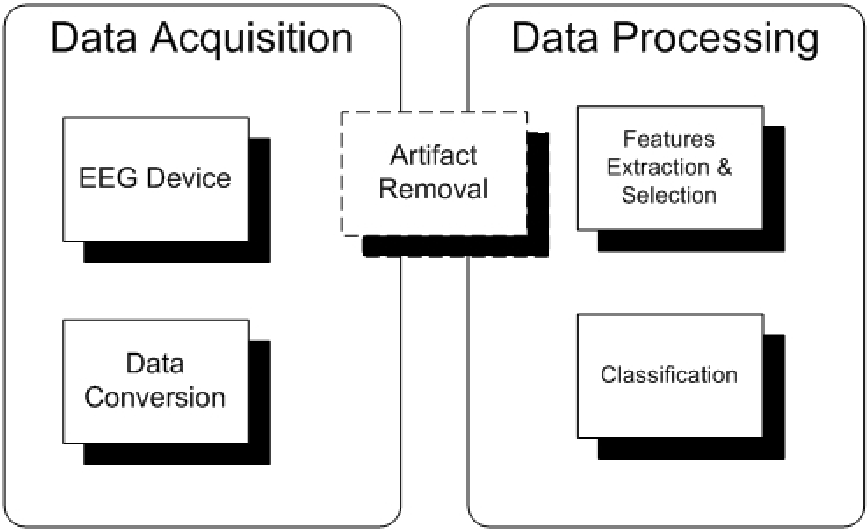


Figure 1.1: Research Blocks: Data acquisition - Data Processing

For data acquisition purpose, several experiments with different subjects are conducted. We used EEG devices and their auxiliary equipments that is already manufactured by a vendor specialized in this field. One thing to note in acquiring data using proprietary device is the data format conversion problems.

One of the important tasks we need to do is artifact removal. In Figure 1.1, we put artifact removal not in data acquisition nor in data processing, because this is done after we acquire the raw EEG data and before we further process the data. EEG data is very susceptible to artifact or noise, therefore this task is very important before we could do further analysis. Artifact identification and removal are done using the Independent Component Analysis (ICA).

For data processing, it can be break down into several blocks of task. In general, we will perform a feature selection. Features can be acquired from different characteristic of the signals. The main feature here is derived from the frequency domain of the signals. To determine the player mental state, we build a simple classifier by combining several features from a number of data. For the purpose of classification, not only do we use data from our own acquisition but also from previous research by Mark Wessel [Wess 06] for trial and error.

In this research, we evaluate and use as many available resources as possible. We use software from open-source projects (EEGLAB, WEKA, BioSig, etc) in this area and other proprietary software that was given from the EEG device we used (DEYMed). For data processing, we heavily rely on WEKA, which is commonly used in data-mining area. We twisted a bit to accommodate our purpose in classifying signals.

1.4 Scope and Limitation

This research is a combination between several disciplines of science and engineering. In fulfilling the objectives of this research, we took a very big learning curve from grasping biomedical engineering and science topics, doing the measurement by trial and error and combining it with other familiar field in computing and physics.

As aforementioned, the primary focus of this research is not on how to control computer games with only our brain without our peripheral system as the actuator. Therefore the approach we took is not to build a prototype of computer games that can be controlled without any keyboard, mouse or other control devices. In our research, due to limitation of the proprietary equipment and its outcome, we are not able to conduct this type of research.

There are so many aspects of the similar research that can not be covered thoroughly. From a biomedical engineering or measurement science perspective, we are not improving the quality of the EEG signal produced or even transferred. The computer games that we use do not occupies this signals into their game play. Some adaptation are being done to the game, but this is just for the purpose of imitating a change of difficulty level.

This research put more burdens on acquiring data as broad as possible. We tried to do as many experiment as possible with a variety of subjects. Although it is very tiresome both for the subjects and ourselves. Putting gel without knowing whether it will work or not is in itself already a challenge. We spent a lot of time acquiring brain signals, however, good quality data is not so easy to get. Consequently, since good quality data is not so lenient to acquire. The resulting data processing, which rely heavily on the data could also suffer insufficiency of good quality result.

This research also is not focusing on exploring the fundamentals element of signal processing and analysis and pattern recognition, but merely served as an example of practical implementation of those elements into practice. We picture the analysis also from the practical point of view, which is the accuracy level.

1.5 Overview

We divided this thesis into several chapters. This chapter, Chapter 1 concerns about the background and underlying reason why this research is conducted. Afterwards, we bring up the main focus of this research is our brain. Brain and its phenomenon will briefly be explained in Chapter 2.

Literature survey and comparing with other related research also being done. The next chapter, Chapter 3 deals with giving an overview of current and future research in related fields. Comparing methods and results from different parties proofed to be helpful both in conducting the data acquisition and data processing. In this chapter, we tried to give an exhaustive outlook of current state-of-the-art research but not to broad discussing parts that is not of our research concern.

After we investigate, we tried to look which area we could do more. Some parts, we even repeat the steps that other have done. In Chapter 4, we explained the underlying methods to conduct the experiments.

The measurement that we conducted can be read in Chapter 5. The steps in preparing EEG devices, head-caps, handling conductive gel, lowering impedance level, and other tips and trick in EEG recording can be read in this chapter. We also mentioned what type experiment are conducted. Moreover, what type of game that subjects will play also explained in this chapter.

In the next chapter, Chapter 6 we presented the result from several subjects out of this experiment. Data processing explained how to further process the data we have gathered previously. Furthermore, the process of removing artifacts is also explained in this chapter. The process of building a simple classifier with WEKA is also mentioned in this chapter.

In Chapter 7, we presented results from data processing. Some analytical reviews are also given. Most of this chapter will explain the result of the classifier built in the previous chapter.

We end this thesis with a conclusion in Chapter 8. One part of this conclusion consist of highlighted findings from the research and experiment conducted. Furthermore, future works derived from this research and experiment are also mentioned. To make everything more explainable, we include appendix which consist of supplementary and helpful data to this research.

CHAPTER 2

Brain

Most of our works in this thesis are related to brain. It is good idea to first of all get some know-how of our brain. Furthermore, measurements are taken place to acquire activity in the brain. Thus, we also explained some measurement techniques commonly chosen for brain activity.

2.1 Human Brain Anatomy

Brain is part of human's central nervous system with spinal cord. If we combine the other nervous system, which is the peripheral nervous system, it controls human's behavior. We are going to talk about mainly the brain, not the whole central nervous system.

Our brain consist of several parts. According to [Guyt 91], mainly it can be divided into:

- Forebrain

  In this area, we encounter the *cerebrum*, which is the largest part of the brain. Its main functions are the initiation of complex movement, speech and language understanding and production, memory and reasoning. Brain activity measurement techniques which make use of sensors placed on the scalp mainly record activity from the outermost part of the cerebrum, the *cortex*.

- Midbrain

  The midbrain forms a major part of the *brainstem*. The brainstem is the part of the brain which connects the spinal cord and the forebrain. Stroke can occur in this part of the brain.

- Hindbrain

  In this part of the brain, we see *cerebellum*. The coordination of all kinds of movements is done in the cerebellum. We see also the *pons*. The pons functions to relay signals from the cortex to assist in the control of movement and is also involved with the control of sleep and arousal. The, *medulla oblongata*, which with midbrain connects brain to spinal cord. The medulla oblongata is involved with the control of unconscious, essential functions such as breathing, blood circulation and muscle tone.

We will now focus on the forebrain, especially the cerebrum, especially its cortex, also known as *cerebral cortex*. It can be divided into several parts (lobes).

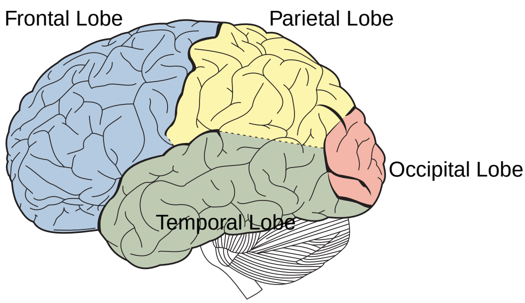


Figure 2.1: Brain's Cerebral Cortex View - Divided into Lobes

As it can be seen in Figure 2.1, the lobes of the cerebral cortex include the *frontal* (blue), *temporal* (green), *occipital* (red), and *parietal* lobes (yellow). Each has its own function. The neuroscience community has already established some common ground on what part of our brain define what is the most common task for that lobe.

1. Frontal lobe

   The frontal lobes are considered our emotional control center and home to our personality. The frontal lobes are involved in motor function, problem solving, spontaneity, memory, language, initiation, judgement, impulse control, and social and sexual behavior. There are important asymmetrical differences in the frontal lobes. The left frontal lobe is involved in controlling language related movement, whereas the right frontal lobe plays a role in non-verbal abilities. Some researchers emphasize that this rule is not absolute and that with many people, both lobes are involved in nearly all behavior.

2. Parietal lobe

   The parietal lobes can be divided into two functional regions. One involves sensation and perception and the other is concerned with integrating sensory input, primarily with the visual system. The first function integrates sensory information to form a single perception (cognition). The second function constructs a spatial coordinate system to represent the world around us.

3. Temporal lobe

   Left side lesions result in decreased recall of verbal and visual content, including speech perception. Right side lesions result in decreased recognition of tonal sequences and many musical abilities. Right side lesions can also affect recognition of visual content (e.g. recall of faces).

   Language can be affected by temporal lobe damage. Left temporal lesions disturb recognition of words. Right temporal damage can cause a loss of inhibition of talking. The temporal lobes are highly associated with memory skills. Left temporal lesions result in impaired memory for verbal material. Right side lesions result in recall of non-verbal material, such as music and drawings.

4. Occipital lobe

   The occipital lobes are the center of our visual perception system. Due to its location at the back of our head, it is not prone to injury. Disorder in occipital lobe can cause hallucinations.

2.2 Physiological Aspect of Brain

Brain is composed of many different types of cells, with its primary unit called the neuron. All the stimuli (such as touches, sound, and so on) that we perceive, pass through neurons. Neurons consist of three parts. The nucleus of a neuron is located in the cell body. Extending out from the cell body are processes called dendrites and axons. Signals then pass from the dendrites through the cell body and may travel away from the cell body down an axon to another neuron, a muscle cell, or cells in some other organ.

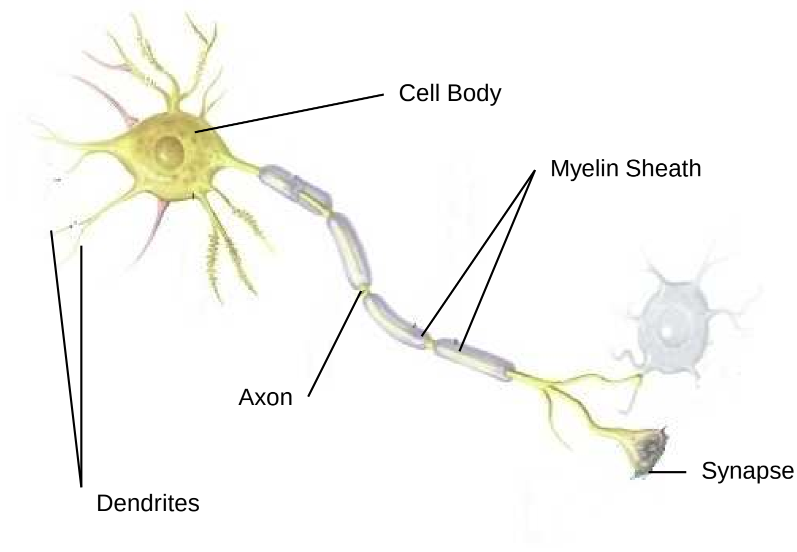


Figure 2.2: Neuron

Neurons are able to respond to stimuli conduct impulses and communicate with each other. There is an unequal distribution of charged atoms on the two sides of a nerve cell membrane (inside and outside). This unequally phenomenon leads up to what is called as resting membrane potential. This potential generally measures about 70 millivolts (with the inside of the membrane negative with respect to the outside). So, the potential is expressed as -70 mV, and the minus means that the inside is negative relative to the outside. It is called a resting potential because it occurs when a membrane is not being stimulated or conducting impulses (in other words, it's resting) [Guyt 91].

2.3 Brain Activity Measurement

There are a number of techniques to do brain activity recording. Several factors that determine which one to use are [Hona 05]:

- Invasiveness

  The more it is not invasive, the more practical the measurement technique would be. Although, some requirements, especially clinical reasons would not make it possible to neglect this factor. Thus, a surgical procedure have to be done.

- Spatial resolution

  The measure of how closely lines can be resolved in a measurement is called spatial resolution. To investigate detail of the brain, a good spatial resolution is needed.

- Temporal resolution

  Temporal resolution can also be referred as time resolution. It describes the amount of detail in a measurement by the number of samples (or image resolution for brain imaging technology) delivered over a given period of time.

- Resources required for operation of the monitoring device

  Resources needed to conduct the measurement is also put into account. This accumulates to the expensiveness of the measurement. The choice of using one device continuously and over a long period of time depends on how much resource it covers.

- Applicability as portable device

  In our research, we want the measurement device to be handy. It can be move around easily. Thus, it would be one of the factors in choosing which techniques to use.

We investigated several brain activity measurement techniques based on literature study and also we did visit some facilities that provided the technology. Initially, since the EEG device is already provided, we do not intend to use other techniques than EEG. It is worthwhile to know and get a comparison between several methods. We exclude EEG and provide it in separate section (see 2.4). Some literature studies are referred from [Wess 06].

1. Electrocorticogram (ECoG)

   Electrocorticography involves the electrophysiology of extra-cellular currents. Has both high temporal as good spatial resolution. It is a form of invasive EEG where electrodes are placed directly on the brain. This technique is invasive and therefore requires expensive surgery and comes with significant safety risks for the patient.

2. Magneto-encephalography (MEG)

   Magneto-encephalography directly measures the cortical magnetic fields produced by electrical currents. This method is non-invasive and has good spatial and temporal resolution. However the equipment is extremely expensive and due to the very weak magnetic fields requires a very impractical isolation/shielding room The real-time properties for analysis are poor. Note, the author did an MEG measurement on for a clinical visit purpose.

3. Computed Tomography (CT)

   It uses special X-Ray equipment to obtain image data from different angles around the body. It then uses computer processing of the information to show a cross-section of body tissues and organs. X-Ray brightness intensity maps in relation to brain tissue density. CT can show several types of tissue, bone, soft tissue, and blood vessels.

4. Positron Emission Tomography (PET)

   Positron Emission Tomography indirectly measures metabolism on a cellular level by tracking injected radioactive isotopes. It is based on the principle that in areas of increased activity the metabolism is on a higher level and more isotopes are supplied by the blood flow. This knowledge can be used to determine which areas are generating activity. Good spatial resolution is an advantage of PET.

The really bad temporal resolution (about 2 minutes) is a distinct disadvantage. This is due to the fact that metabolism is a relatively slow process. Moreover ionizing radiation makes this method harmful for the human body and thus unusable for applications like BCI.

5. Magnetic Resonance Imaging (MRI)

   Radio waves pass through a large magnetic field. A computer monitors the variations in the radio waves due to the electro-magnetic activity in the brain to generate a picture.

6. Functional Magnetic Resonance Imaging (fMRI)

   Functional Magnetic Resonance Imaging provides information on brain metabolism using BOLD (Blood Oxygen Level Dependent). The advantages are good spatial resolution and the non-invasiveness. But the temporal resolution is poor (about one second) and this method requires very expensive equipment and procedures.

7. Functional Near-Infrared (fNIR).

   Functional Near-infrared light penetrates the human head to sufficient depths to allow functional mapping of the cerebral cortex. Changes in tissue oxygenation cause modulation of absorption and scattering of photons, which can be measured. This response has a good temporal resolution, but is not yet feasible.

We can summarize the disadvantage of using the aforementioned techniques in Table 2.1.

Table 2.1: Brain Activity Measurement Techniques Comparison

Techniques	Disadvantage
ECoG	Highly invasive, surgical
MEG	Extremely expensive
CT	Provide only anatomic data
PET	Radiation exposure
MRI	Provide only anatomic data
fMRI	Extremely expensive
fNIR	Still in research

2.4 EEG

The electroencephalogram (EEG) is the electric potential recorded at the surface of the scalp, resulting from the electrical activity of large ensembles of neurons in the brain. The noninvasive brain imaging technique based on EEG recording is called electroencephalography.

2.4.1 Electrode Placement

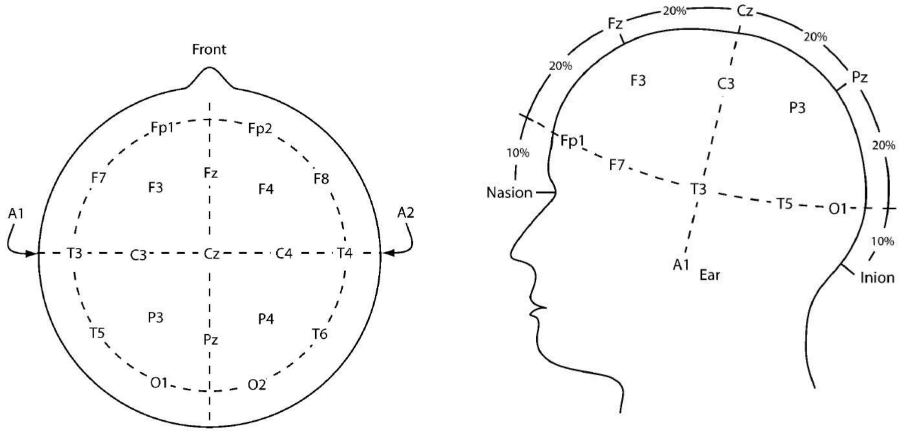
The positions for EEG electrodes should be chosen in a way, that all cortex regions which might exhibit interesting EEG patterns are covered. For most applications this is usually the whole cortex. An internationally accepted standard for electrode placements is the 10-20 system. It is called 10-20 system, because the distance between electrodes are between 20% of each other from the whole area of the scalp and it is placed 10% above the *nasion* and *inion* (will be explained next).

Reference points must be determined before the 10-20 system electrode positions can be applied. These include the *nasion*, *inion*, and ears. Nasion is located above the nose on the skull, below the forehead. While inion can be determined by a bony end marks the transition between skull and neck (see Figure 2.3).

As can be seen in Figure 2.3, we describe the naming convention of the 10-20 system. The name of the placement is according to the lobes which it tries to measure. There are the Fp electrodes, to measure pre-frontal lobe *–just above the eye*, F electrodes, to measure frontal lobe, C to measure central part of the cortex, P for parietal lobe, T for temporal lobe, and O for occipital lobe.

The scalp is also divided into three sections, the left and right hemisphere and central area. We divide this section by taking a straight line from the front to the back. The convention is that the left section will be numbered as odd and right with even numbers. The central area will be noted with 'z' addition. For example the prefrontal electrodes on the left will be named as Fp1 and at the right will be Fp2. The frontal electrode at the central will be Fz.

For the purpose of our research, we used the 19-channel version of the 10-20 system electrode placement. It consist of Fp1 and Fp2 for the prefrontal.



(a) 10-20 System (b) Nasion-Inion

Figure 2.3: Electrode Placement

F3, F4, F7, F8 and Fz for frontal area. C3, C4, and Cz for the central area. P3 and P4 for the parietal area, T3, T4, T5, T6 for the temporal. Finally O1 and O2 for the occipital lobe. There are also electrodes which are placed on the ear lobes called A1 and A2.

Besides what have been mentioned above, we also need reference point(s). Reference point is the placement of an electrode (could be more than one) to be used as a common calculation for the other electrodes.

We can say that criteria for a good reference point are:

- It should be placed far from the cerebral cortex, that would make it impossible for brain activity to interfere.
- It should be placed in a position where no other physiological activity could happen.

In our research, we tried several placements as reference points. Among them are the ear lobes and central part of the scalp.

2.4.2 Electrode Montage

Electrode montage means various ways in which the signals from different electrodes can be combined before further processes or just for viewing purpose using analytical software. We mentioned here, because throughout the experiment, it will be referred several time. There several ways to connect EEG electrodes, two of them are: common-reference montage and bipolar electrode montage.

In common-reference montage, the EEG signals represent the difference of the potentials of each electrode to one or more reference points. Here, we need to specify a reference point(s) first. For our experiment, most of our data used the ear lobes electrodes as reference points.

In bipolar montage, we used the difference between neighboring electrodes to be measured. In the medical domain bipolar recordings are used to detect the specific location of particular processes.

More on the electrode montage will be explained in Chapter 6, when we conducted the data processing part.

2.4.3 Artifacts

Since the amplitude of EEG is very small, it is susceptible to several artifacts. Those artifacts can be divided into physiologic artifacts and extra-physiologic artifacts.

Physiologic Artifacts

Physiologic artifacts are generated from the patient. These artifacts arise from other part of the body than the brain. We gather this information from [Benb 06].

1. Muscle Activity

   Our frontal and pre-frontal muscles are the common causes for these type of artifacts. Clenching our jaw can incept artifacts. An example of muscle activities artifacts can be seen in Figure 2.4.

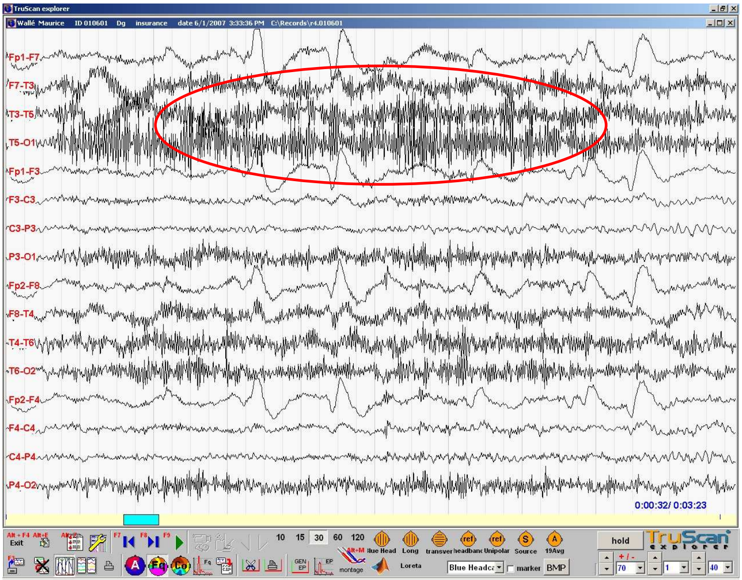


Figure 2.4: Muscle Artifact

2. Eye Movements

   There are two noticeable eye movements, the vertical eye movements typically observed by blinks and the lateral eye movements. The vertical eye movements affect the prefrontal position (Fp1 and Fp2). While the lateral eye movements affect frontal electrodes (F7 and F8). An example of eye movement artifacts can be seen in Figure 2.5.

   Figure 2.5 full of eye-blinks artifacts. Most of the blinks happen at the prefrontal or frontal channel electrodes. Those are the area where the susceptibility to eye-blinks is very high.

Extra-physiologic Artifacts

Besides the physiologic artifacts, there are also other artifacts not related to our physiological phenomenon. It is commonly referred as extra-physiologic artifacts.

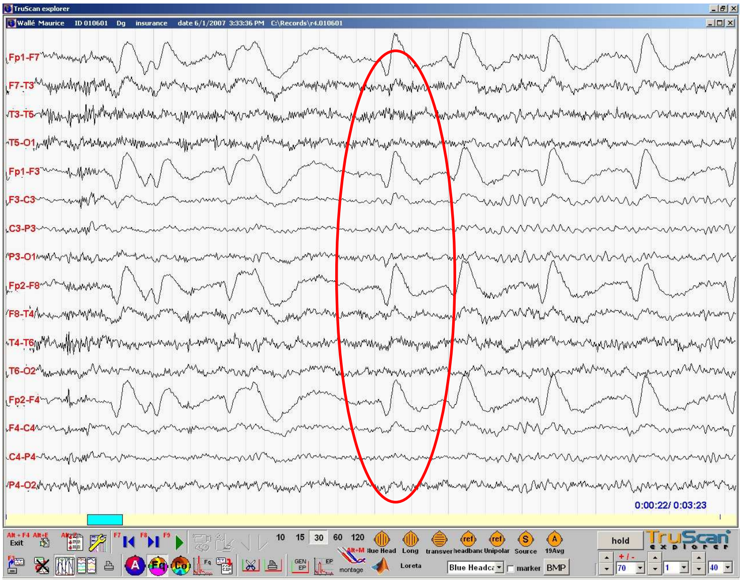


Figure 2.5: Eye Movement Artifact

1. Electrodes

   During the recording we see many unexpected phenomenon, such as the appearance of so many delta waves activity. Sweat can affect conductivity of the electrode. We see some low-frequency signals appearing. Usually, this only affect single electrode. If the impedance level of the electrode is not stable it can also cause this to happen.

   One possible artifact caused by electrode can be seen in Figure 2.6. This is referred as electrode popup. The possibility of causes can vary. Unstable impedance level can influence this. It can also because of the electrode itself becomes gradually defect, due to lack of cleaning of improper handling.

2. Movement in the environment

   Movement of other persons around the patient can generate artifacts. That is why, often in research center or hospital EEG recordings are done in an intensive-care unit room.

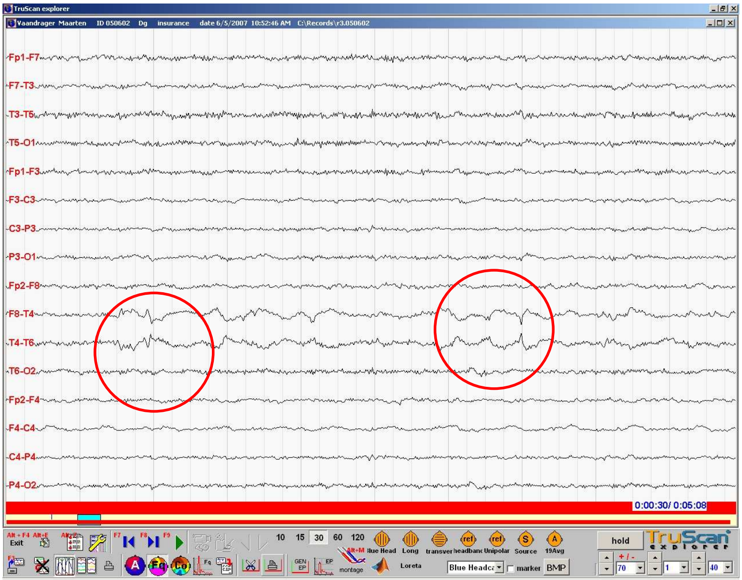


Figure 2.6: Electrode Artifact

3. Other devices

   Interference from high-frequency radiation from radio, television, or other electronic devices can interfere the device. This will generate high-frequency signals in the recording.

2.4.4 Frequency Bands

EEG frequency bands can be divided into: delta, theta, alpha, beta, and gamma. Certain lobe of the cerebral cortex has the most occurrences of certain band, although this is not definitive. Each band also has associative mental states, which can be used to determine one's condition. Summary of the EEG frequency band can be seen in Table 2.2.

EEG of adult person within delta frequency band occurs in very deep sleep, hypnosis or in coma. If is present in wake state, it is always pathological phenomenon. The higher amplitude or better locality mean more serious problems. But this frequency band is normal for very young children.

Theta frequency band is, for healthy people, especially in deep sleep and is often connected to dreaming. This waves occurs in central, temporal and parietal parts of head. Pathological occurrences are if amplitude of theta is as twice as high as alpha waves or is higher than 15 V.

Alpha waves can be typically found in back side of head and are characteristic for awake state without mental or physical activity. The highest amplitude achieves in closed eyes state and fades with opening eyes and growing activity.

Beta frequency band are connected to the front part of the head and occurs with high mental activity like concentration, thinking or emotions.

Gamma frequency band is specifically to humans and is only band found in every part of the brain. Occurs when brain needs to simultaneously process information from different areas. A good memory is associated with gamma waves activity and deficiency creates learning disabilities.

Table 2.2: EEG Frequency Bands

Band name	Frequency (Hz)	Description
Delta	0 - 4	Deep sleep
Theta	4 - 8	Creativity, drifting thoughts, dream sleep
Alpha	8 - 13	Relaxation, calmness, abstract thinking
Beta	13- 30	Focused, high alertness
Gamma	above 30	Simultaneous process

Chapter 3

Related Research

We investigated research concerned with the topic of computing human brain activities. In most of the investigation and literature studies, we found that EEG is used. This does not mean, we are neglecting other findings with other brain measurement techniques. Furthermore, we also put in to account research not directly related but could complement our research. Topics we investigated are:

- Clinical Research

  There are already a number of clinical research make use of EEG. Included in this clinical research is sleep and wake disorder research. The method used in this type of research can be applied to our research. Moreover, much of the research in the field of EEG analysis are most of the time related to clinical purpose.

- Mental State Identification

  Mental state identification tries to classify which brain activity reading tell about a certain task. Identifying human mental state is a research field that incepted the new field of Brain-Computer Interface (BCI). When we know, that a person is performing a certain task in the brain, if we can classify this reading as moving an object, we can have a basic BCI which tries to control a computer using our brain.

- Brain-Computer Interface (BCI)

  The enhancement of mental identification is BCI. BCI tries to control a computer or a device in general, by classifying certain brain reading as input to move the object. Most of the target group for this application are severely-disabled people that can not communicate with their peripheral nervous system.

- Vigilance Assessment

  Vigilance or alertness can be used as parameter in assessing whether a person is overloaded with a series of tasks. By assessing vigilance, for example in driving a car, we can predict the rate of accident caused by alertness.

- Computer Games

  Vigilance assessment can also be applied to a computer game or virtual reality system. Where we can simulate the activity before it is done in reality.

On each topic, we hope to elucidate more by giving a survey of the research in that particular topic. At the end of this chapter, from the survey we can have an outlook what to do and where the direction of our research will be.

3.1 Clinical Research

Research that has clinical purpose mainly focuses on improving the quality of the recorded brain activity measurement. We also consider the urgency to reduce and remove artifacts found in EEG recording to be a clinical purpose.

We find the work of [Gerl 06] is interesting and have an important aspect for our research. The research is about how to do a multichannel analysis of newborn EEG signals. Newborn babies usually sleep around 70 percent of their 24-hours day. Sleep in infants is significantly different from sleep in adults. Practically, the intention is to differentiate between three important states in babies: quiet sleep, active sleep and wakefulness state.

They recorded on eight EEG channels (these are Fp1, Fp2, T3, T4, C3, C4, O1, O2) combined with EOG, EMG, respiratory and ECG. Data are scored by an experienced physician to four states (wake, quiet sleep, active sleep, movement artifact). In our approach we use a method based on power spectral density (PSD) applied to each EEG channel. We also use features derived from EOG, EMG, ECG and PNG signals. They also did a comparison of the classifier based on Markov models with various learning algorithms, in particular nearest-neighbor, cluster analysis, and induction of decision rules.

In [Rome 03], the use ICA to reduce artifact in different sleep stages is explained. The different sleep stages that is scrutinized was awake, stage 2, delta and REM sleep. Artifacts, particularly certain types, may be more likely found in particular settings and stages of sleep. Artifact identification is based on time, frequency and scalp topography aspects of the independent components from the ICA. Influence of artifacts is evaluated by calculating some target spectral variables before and after the reduction, using significance probability maps.

The recordings were sampled at 256 Hz. It was recorded from 19 electrodes placed according to the 10-20 system, using averaged mastoid electrodes as a reference. Vertical EOG and Horizontal EOG were also recorded. Three male subjects aged 25, 26 and 27 years participated. Sleep stage scoring was done by three experts. Fifteen 5-second segments were selected in each of the sleep stages.

Table 3.1: Clinical Related Research

Author	**Electrodes**	**Classifier**
[Gerl 06]	Fp1, Fp2, T3, T4, C3, C4, O1, O2	Decision rules
[Rome 03]	19-channel 10-20 system EEG electrode cap	Not mentioned

3.2 Mental State Identification

The principal research in this topic was done by [Keir 90]. After this research, almost all research done in this topic used their data sets. Basically, the task that a subject would have to do are:

- **Baseline-Alpha Wave Production**

  Subjects were asked to relax and to close and open their eyes in five seconds intervals. Doing this, α-activity can be observed, at least when eyes are closed.

- **Mental Arithmetic**

  Subjects had to solve non-trivial multiplications without vocalizing or moving.

- **Geometric Figure Rotation**

  Subjects were shown a drawing of a complex geometric figure. Then the figure was moved out of sight and subjects were instructed to imagine the rotation of this figure.

- **Mental Letter Composing**

  Subjects were instructed to mentally compose a letter to a friend or relative, without moving or vocalizing.

- **Visual Counting**

  Subjects had to imagine a black board and mentally to visualize numbers being sequentially written on the board.

Even though, it is not exactly the same, but experiments conducted for the purpose of mental state identification follow the same procedure.

In [Culp 99] artificial neural networks were trained to classify segments of 12 channel EEG data into one of five classes corresponding to five cognitive tasks performed by one subject. The five tasks are the same with aforementioned. The electrodes were placed at FpZ, F3, Fz, F4, FcZ, C3, CZ, C4, Pz, P3, POz, and P4, which is not exactly the same with what we used (19-channel 10-20 system, this is uses a portion of the 32-channel 10-20 system).

ICA was used to remove artifactual EEG components. Eye blinks were detected by means of a separate channel of data recorded from an electrode placed below the subjects left eye (Vertical EOG). For the result, it obtained a classification accuracy of 94% when differentiating between the geometry and multiplication tasks. Using a 1/20th second window of EEG, it obtained a classification accuracy of 85% when differentiating between the geometry and multiplication tasks.

Pioneering the research of brain activity computing in TU Delft, [Wess 06] also did a similar experiment that is conducted by [Keir 90]. The experiments were, i.e.:

- Baseline task and Mental rotation

  Baseline task is the same as [Keir 90]. This is to generate alpha wave from the subjects. Mental rotation task is also similar to Geometric Figure Rotation from [Keir 90].

- Motor imagery

  Subjects performed the task for both the left hand and the right hand. The first task is rotating the complete limb where the hand makes small circles in front of the subject. The second task is to imagine grabbing an object in front of the subject.

- Mathematical calculation

  Several summation, subtraction, and multiplication problems were presented to the subjects. Besides visual presentation, auditive narration of the mathematical problem were also given.

- Visual presentation

  Four objects were given in this task and subjects have to look at that objects without blinking for five seconds.

- Visual self-selection

  This task is in combination with a gaze-tracker device. Given four objects at each corner of the screen, subject selects one of the objects and then concentrates on the object.

- Visual and auditive presentation

This task is very similar to the visual only task; in addition now not only is the figure shown in the center of the screen it is also auditively uttered.

- Auditive presentation

  In this task blank screen was given, without any objects. Only auditive presentation of the objects were given.

Experiments in [Wess 06] was conducted using 19-channel of 10-20 system electrode cap with additional electrode for ECG. The experiment mainly conducted to trigger brain to produce signal in the location in which we presumably the activities are happening.

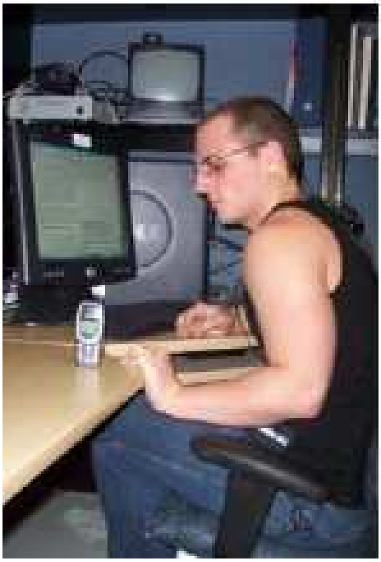
In [Lee 06], similar but lesser experiments to [Keir 90] were conducted. Two electrodes were used. They are placed at the parietal lobe (P3 and P4). Three tasks were defined. Rest, in which they instructed subjects to relax and to try not to focus on anything in particular. Mental Arithmetic, in which subjects performed mental multiplication of a single digit number by a three digit number. Mental Rotation in which subjects imagined specific objects.

Features were extracted from signal power in each of the six frequency bands for each channel, phase coherence (similarity in mean phase angle) in each band across channels, and each band power difference between the two channels. In addition to these features, they also compute the following set of more general signal properties for each input channel: mean spectral power, peak frequency, peak frequency magnitude, mean phase angle, mean sample value, zero-crossing rate, number of samples above zero, and the mean spectral power difference between our two input channels. For the classification Bayesian Network was used.

Next, we investigated an example from a real-life situation. Physiologically Attentive User Interface (PAUI) by [Chen 04], tried to measures mental load using heart rate variability signals, and motor activity using EEG analysis. It tries to distinguish four mental states and motor activity: resting (low mental activity, no movement), moving (low mental activity, sustained movement), thinking (high mental activity, no movement) and busy (high mental activity, sustained movement).

Only one electrode is used, which is placed at the Cz location if it on the 10-20 system, this is supposedly where motor activity is controlled.

(a) One Electrode Recording



(b) Tested on Mobile Phone

Figure 3.1: Physiologically Attentive User Interface (PAUI) by [Chen 04]

Features were collected by doing spectral analysis of heart variance. An increasing heart variance is an indication of an increase of mental load. While, a decrease in power for the μ-frequency band could indicate changes of motor activity.

A prototype was built where it can regulate notifications by such devices dynamically on the basis aforementioned physiological phenomenon. PAUI was applied to a mobile phone related activities where a call will be hold if a subject is currently at his/her busy state and will pass through if in a resting state, for example.

Almost correlate with the previous research, another interesting research was done by [Hona 05]. It tries to determine people's current activity in a meeting, lecture or office scenarios. Almost the same as PAUI, this research tried to make devices not interfering a person, when he or she is currently in a highly engaging task. To do this, EEG recording was used. EEG is recorded using 16-channel (Fp1, Fp2, F7, F8, F3, F4, T3, T4, T5, T6, P3, P4, O1, O2, Fz, Pz) 10-20 system cap and customized headband with 4 electrodes (Fp1, Fp2, F7, and F8) sewed to it.

Essentially, the purpose of the experiments was twofold. First is to discriminate between five activities: resting, listening, perceiving a presentation, reading an article in a magazine, summarizing the read article, and performing arithmetic operations. The second one is to predict the low and high task demand an activity is.

Features were extracted from frequency content of the signals. Then, they are passed to artificial neural-network or to a support vector machine for the classifying task. Furthermore, to be able to predict the level of task demand, it used a self-organizing map.

Table 3.2: Mental State Identification Research

Author	**Electrodes**	**Classifier**
[Culp 99]	FpZ, F3, Fz, F4, FcZ, C3, CZ, C4, Pz, P3, POz, and P4	Back-propagation algorithm
[Chen 04]	Single electrode (Cz) combined with Heart Rate Variability	Decision table
[Hona 05]	16-channel (Fp1, Fp2, F7, F8, F3, F4, T3, T4, T5, T6, P3, P4, O1, O2, Fz, Pz) 10-20 and a headband (4 electrode attached Fp1, Fp2, F7, and F8)	Artificial neural network and support vector machine
[Lee 06]	Two electrodes (P3 and P4)	Naive Bayes classifier
[Wess 06]	19-channel 10-20 system EEG electrode cap	No classification

3.3 Brain-Computer Interface

According to [Wolp 00], Brain-Computer Interface (BCI) can be defined as a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles. It involves monitoring brain activity using brain measurement and imaging technology. Furthermore, it tries to detect characteristic in the electrical signal produced. The majority of researches being conducted is based upon EEG. For our research, we narrowed down our literature research just on the EEG-based BCI. We also wanted to know which features and classification method do they use.

The motivation behind the invention of BCI is to develop a replacement of communication media for severely disabled people. People with disabled mobility, have no other way of communicating with other device in this case a computer besides using their brain. Todays focus and goal of many principal research institute in BCI technology are being directed to aid disabled target user.

BCI as a system from acquisition to the enablement of controlling a computer/device can be seen in several perspective. It can be an online or an off-line BCI. The difference between online and off-line is on the necessity that one system should process the signal in a real time manner or not. It also has consequences in the way it can have artifact or not. In off-line system, artifacts removal is being done in a separate process, while in online system is being done real-time.

Furthermore, a BCI system can be viewed as synchronous or asynchronous. Nearly the same as with the previous category, a BCI system can be a synchronous or asynchronous system in the sense that they have to be processed in an according time on-demand or user-driven and continuously. In synchronous system, it is computer-driven and the EEG has to be analyzed only in predefined time windows. While in asynchronous system is user-driven and the EEG signals have to be analyzed and classified continuously.

There are two approaches to take into consideration in constructing a BCI system. The first one is the so-called pattern recognition approach. In this approach, user has to perform different mental task aforementioned.

If the system can distinguish which task is being performed, a particular command will be executed. The second approach is the so-called operant conditioning approach. In this approach, subjects have to learn to control his/her EEG response and the system already have some characteristics that can distinguish among them. In the second approach, feedback is needed to know whether or not the subject is performing well.

In [Krep 04], the authors introduced the application of BCI for multimedia application and computer gaming. The primary focus is to have an online BCI system in the case of controlling multimedia application or computer games. The experiments conducted was the subject faced a computer screen and he/she wore a brain-cap with 128 electrodes. It distinguished between left hand movement and right hand movement. Thus, the most important electrodes would be in the central cortex part (Cz, C3, C4 in the 10-20 system).

By comparing the Fourier coefficients of the frequency band acquired (delta, theta, alpha, beta, and gamma) 4 features per channel were acquired. Further, classification methods use was support vector machine and linear programming machines.

The work of Oxford University BCI in [Robe 00] focused on using Bayesian learning. In this work, a real-time BCI was proposed. The test case was a cursor movement. For the experimental setup, this research used two electrodes positions in central cortex of the scalp (C3 and C4 in the 10-20 system). The primary observation is motor activity, which can be observed in this part of our cerebral cortex. For the processing, logistic classifier was used. Seven subjects participated and each performed moving cursor upwards followed by an attempt to move it downwards.

One of the steps in the pattern recognition approach is task classification. To do that, [Mull 00] conducted an experiment to know which electrodes has a high degree of importance to the measurement. The approach is exemplified in the frame of the brain-computer interface, which crucially depends on the classification of different brain states.

To classify two brain states, e.g. planning of movement of right and left index fingers, three different approaches are compared: classification using a physiologically motivated set of four electrodes, a set determined by principal component analysis and electrodes determined by spatial pattern analysis.

For the recording 61 electrodes was used. Unfortunately, most of the 61 electrodes used have no influence on the classification rate. Thus, it can be said, that the setup can be simplified drastically to six to eight electrodes without loss of information.

Table 3.3: BCI Related Research

Author	Electrodes	Classifier
[Krep 04]	128-channel 10-20 system EEG electrode cap with emphasize on C3 and C4	Proposed a candidate of classifier to use as support vector machine
[Robe 00]	Two electrodes at (C3 and C4)	Bayesian Net
[Mull 00]	61-channel electrodes	Principal component analysis and spatial pattern analysis

3.4 Vigilance or Stress Assessment

Assessing vigilance or alertness is closely related to our research. Most of the research in this topic, are related to maintaining concentration in a simulation environment.

We find the work of [Ishi 03] very intriguing. This research is about how to estimate a feeling from EEG data. Essentially, it tries to classify brain activities into four distinct feelings, which are joy, anger, sorrow, and relaxation. For the experiment, they used IBVA. IBVA uses headband and has three electrodes attached at the prefrontal and frontal area.

In [Lian 05], driven by growing number of traffic fatalities, it is needed to have an assessment of cognitive response in a traffic situation. The experiment is on the brain dynamics given a traffic-light situations. The system consists of a dynamic virtual-reality (VR)-based motion simulation platform. ICA algorithm is used to obtain noise-free ERP signals from the multi-channel EEG signals. Principle Component Analysis (PCA) is used to reduce dimension of features, which were then fed into a Fuzzy Neural Network classifier.

A total of three subjects participated in the VR-based traffic-light driving experiments where EEG signals were simultaneously recorded. The subject is asked to decelerate when he/she detected a red light, to accelerate when he/she saw a yellow light, and keep constant speed when he/she saw the green light. The experiments used 31-channels 10-20 system EEG and 4-channel EOG.

In the [Chop 00], two experiments were conducted. The first experiment is about the stress assessment. Nine subjects were participated. To intrigue stress, different difficulty level was proposed, the resting phase (where subject do not have to do anything), level 1 (where the difficulty starts), level 2 (where difficulty gets increased), and level 3 (the hardest).

The experiment used 10-channel 10-20 system EEG cap (Fp1, Fp2, F3, F4, P3, P4, O1, O2, T3 and T4) and two EOG channels. Features extracted were power spectral density (PSD) and inter-electrode coherence. The choice of the computer games used had been adapted so that the first factor has a minimal influence, that is, so that the experienced stress reported by subjects is lowest for game 1 and highest for other game.

The second experiment was to investigate how emotion influences the EEG, so that this information could be used in an emotion expressing interface for the disabled. Rather than understanding neural mechanisms of emotions, it was aimed at finding EEG features allowing a good discrimination between different emotional states, and possibly, a quantification of emotional intensity.

EEG recordings were done in a shielded room It used 13-channels EEG cap (Fp1, Fp2, F3, F4, F7, F8, T3, T4, P3, P4, O1, and O2) according to the 1020 system referenced at the left earlobe A1. Twenty subjects participated to the experiment. Stimuli used were sound, pictures and a combination. Like in the first experiment, features selected were PSD and inter-electrode coherence. Classification was done using Artificial Neural Network.

The work by Petr Bouchner at Czech Technical University focused on the vigilance assessment in driving. In some ways, the experiments conducted were similar to the one conducted by [Lian 05], in which traffic light scenarios were used. Moreover, it also assessed the driver's attention level. Driving simulator was used to simulate a driving scenario. Large screen display was used combined with a static car that acts as a simulator.

Other than EEG, other measurements were also put into account. Among them are response time between the stimuli was generated until the first response; self rating about the subjects themselves during the simulation took place; lane variability which can depicts drowsiness level; heart-beat rate can be used to detect if the subjects sleepy or not; finally EEG analysis.

EEG recordings were conducted using 19-channel electrode of the 10-20 system EEG cap. The analysis mostly focused on the area where artifacts will likely less to occur, such as the occipital lobe (O1 and O2) and the central cortex area (Cz). Samples were collected with a length of 3 seconds in the time just before red light signal appears during the simulated drive.

The investigation of EEG signals was done in the differences of alpha, theta and alpha/delta band ratio between drowsy and fresh driving. Initial results showed that no pattern could be depicted. One of the reasons could be that many of samples of were discarded due to the artifacts in EEG signal and it did not give enough representatives for good statistical analysis [Bouc 07].

Monitoring task loading during a command and control task was the objective of [Berk 05]. To do that, simulations were prepared. The first simulation is about identifying aircrafts as hostile or friendly and appropriately action should be taken. The second one, numbers were presented and the next consecutive digits have to be fill in. Last, recognizing memorized images in a collection of new images. All the tasks were varied in difficulty levels.

In this experiment, bipolar recordings of Fz to POz and Cz to POz were used. Other than that, common-reference recordings of Fz, Cz, and POz were also used. For the first simulation, 13 subjects were participated.

The second one, 16 subjects were gathered. The last one, recorded 19 subjects. Four states were to be predicted, high vigilance, low vigilance, relaxed wakefulness and sleepy.

Artifact removal were especially eye blinks were discarded using high-pass filter. Features extracted were log power spectrum and the relative power compared to the power for each frequency band. Prediction was done using a linear discriminant function.

Table 3.4: Vigilance and Stress Assessment Related Research

Author	Electrodes	Classifier
[Ishi 03]	Three electrodes (prefrontal and frontal area)	Feed-forward neural network
[Lian 05]	31-channel 10-20 system	Fuzzy neural network
[Chop 00]	10-channel and 13-channel EEG cap	Artificial neural network
[Bouc 07]	19-channel 10-20 system	No classification mentioned
[Berk 05]	Bipolar recording: Fz to POz, Cz to POZ and common reference: Fz, Cz, and POz	Linear discriminant function

3.5 Computer Games

The application of BCI research in computer games mostly deals with an online BCI system. But this does not stop here, because actually there are several branches implementation of BCI in computer games. As it was coined in [Nijh 07].

Those implementations are:

1. Controlling with our brain activity

   BCI can be used as a direct controller by thought. This means that inducing thoughts to manipulate brain activity that can be mapped onto game interaction commands(e.g., move cursor, click buttons, control devices);

2. Cognitive task determination

   We can also figure out some ways to determining the cognitive tasks in which the user is involved in order to evaluate (game) interfaces or game environments;

3. Adapting computer games

   using cognitive or affective state of the user to dynamically adapt the interface to the user (e.g., detect frustration or engagement and provide tailored feedback).

Simple bouncing ball game was presented in [Palk 04]. The game is called Brainathlon, which is now part of the OpenEEG project [Open 07], an open-source project to build low-cost EEG device and application. In Brainathlon, a game was developed making use of brain activities. Basically, the game is a typical neurofeedback game, that demanded the player to train his brain signals.

Brain signals were transformed into frequency bands (see Section 2.4.4). The game makes use of this frequency band. The experiment consisted of three sessions. The first session is to play the game an try to increase the band (frequency band). If for example we are at a relaxed state (a lot of alpha activity) we should increase it to deeply thinking (beta). The second session, player(s) have to maintain its current state. Whatever state he/she is at at that particular time, it needs to be maintain to win the game. The third session is to compare current state with a reference band. The player who can reach that band will win the game.

For playing the game, a player have to wear a one electrode at the Cz location according to the 10-20 system placement. One electrode attached to the left ear as a reference. The filtering and transformation were done

Figure 3.2: Brainathlon by [Palk 04]

in real-time. The EEG devices used was self-made from the OpenEEG project. The Brainathlon is a simple version of the type of game that can be controlled using our brain, as aforementioned.

Table 3.5: Computer Games Related Research

Author	Electrodes	Classifier
[Rani 05]	C1, C2	Decision Trees
[Palk 04]	One electrode, Cz	No classification
[Lalo 05]	Occipital area, O1 and O2	Linear discriminants

3.6 Summary of Related Research

For our summary, we put the related research work in a table form, as can be seen in Table 3.1, 3.2, 3.3, 3.4, 3.5. Some research are relatively spanned over several type of research. It can be put in more than one type of research.

From the literature study, we see that the placement of electrode depends on what type of measurement we want to have. It depends also on which part of the brain cortex we want to investigate. We also see that the application of ICA is used but not per se have to be applied.

Chapter 4

Methods

In this chapter, we explained how we approached the research given some literature study on related research (see Chapter 3).

4.1 Research Framework

We based this research upon Mason et. al brain interface technology framework. According to Mason et. al [Maso 05], in general we can simplify into several blocks.

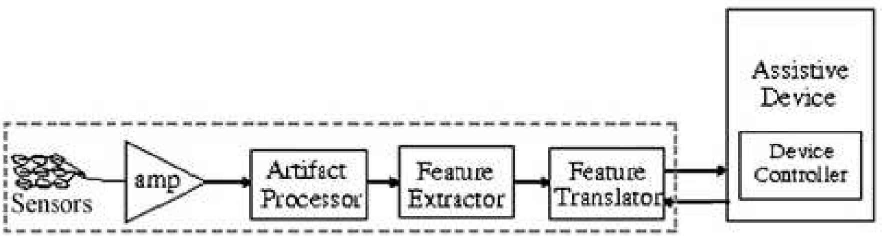


Figure 4.1: Brain Interface Technology Framework [Maso 05]

Accordingly to the general framework, aforementioned in Chapter 1. We divided our research work into data acquisition and data processing. In data acquisition, we deal with the EEG device and acquiring data through several experiments. After that, we dealt with converting all the acquired data. Artifact removal requires deep attention, since most of the data are noisy. Features need to extracted to enable us eventually classify the brain signals. Figure 4.2 clearly explains these procedures with which section to look.

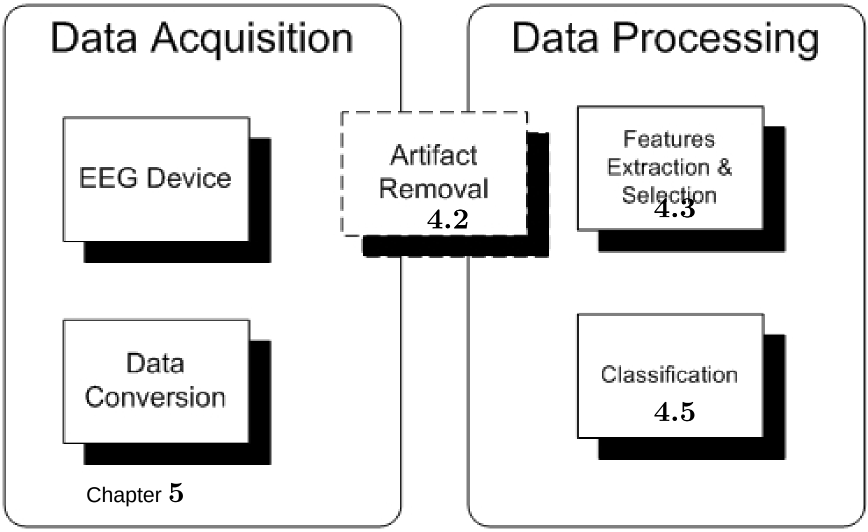


Figure 4.2: Research Blocks (with Appropriate Chapters)

4.2 Artifact Removals

In EEG recordings, electrodes are placed at the scalp with a distance of only few centimeters away from each other. These electrodes not only recorded brain activity from neural activities but also artifacts. These artifacts overlap with the neural recordings. We are concentrating on removing eye-related artifacts. First and foremost because it contaminates constantly and has the largest impact on the EEG data.

4.2.1 Visual Inspection

Visual inspection is the first step to do artifact removals. Though, it seems straight-forward, an expert eye is still needed. We have aforementioned several possibilities of artifacts in Section 2.4.3. This does not cover the whole possibilities of artifacts.

4.2.2 Independent Component Analysis (ICA)

The purpose of ICA is to decompose these overlapping recorded activities into independent component.

The main concern of ICA is to separate between several sources of signals. Principally, it can be simplify into these equation

$$\vec{x}(t) = A.\vec{s}(t)$$

where $\vec{x}(t)$ is the vector of signals we are measuring at time t and $\vec{s}(t)$ are the real sources of the signals, the independent component. Moreover, A is the mixing matrix which we want to get from performing ICA.

The goal of ICA is to find a linear transformation **W** of the dependent electrode signal **x** that makes the output as independent as possible

$$\vec{u}(t) = W.\vec{x}(t) = WA.\vec{s}(t)$$

where **u** is an estimate of the sources. The sources are exactly recovered when **W** is the inverse of **A** up to a permutation and scale change.

$$P = RS = WA$$

where **R** is a permutation matrix and **S** is the scaling matrix. The two matrices define the performance matrix **P** so that if **P** is normalized and reordered a perfect separation leads to the identity matrix. For the linear mixing and unmixing model, the following assumptions are adopted.

1. The number of sensors is greater than or equal to the number of sources N $\geq$ M.
2. The sources $\vec{s}(t)$ are at each time instant mutually independent
3. At most one source is normally distributed
4. No sensor noise or only low additive noise signals are permitted

If we apply ICA to EEG analysis, it can reveal the diversity of source information typically contained in EEG data, and the tremendous ability of ICA to separate out these activities from the recordings.

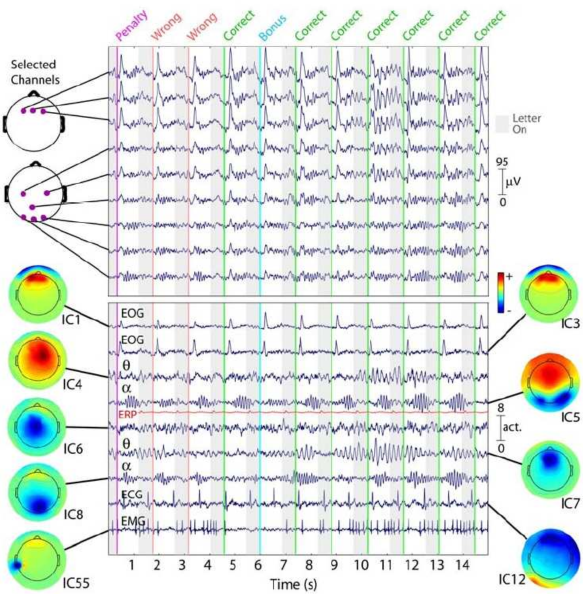


Figure 4.3: ICA Decomposition [Onto 06]

In figure 4.3, fifteen seconds of EEG at several channels with several independent components.

4.3 Feature Extraction and Selection

Generally, a feature can be arbitrary characteristic that will differentiate one particular object to another. In simple life, it can be color, taste, mean value, etc. In the context of data-mining, a feature is usually called an attribute. Furthermore, the process of acquiring features from given object is called feature extraction. Out of the available features, we may choose which to use, which is to be selected. Thus, for simplicity we call the process of generating and selecting features as feature selection. We chose features that are commonly used from our literature surveys. They will be explained in the next section.

4.3.1 Amplitude

Early Brain-Computer Interface research used amplitude differences between channel to classify EEG signal [Wolp 94].

Average Amplitude

$$A_{mean} = \frac{1}{N} \sum_{n=1}^{N} x_n \tag{4.1}$$

where x_n is a sample point

Maximum Positive Amplitude

$$A_{max+} = \max_{n=1,\ldots,N} x_n \tag{4.2}$$

Maximum Negative Amplitude

$$A_{max-} = -\max_{n=1,\ldots,N} (-x_n) \tag{4.3}$$

4.3.2 Fourier Frequency Band Spectrum Energy

Band powers of selected EEG frequency bands, for the frequency band division refer to 2.4.4.

DFT Band

Calculating Fourier coefficient band of EEG signals. Alpha band showed here.

$$DFT\_alpha(x) = \sum_{n \in alphaHzofDFTcoeficients} |DFT(x)_n| \tag{4.4}$$

where x_n is a sample point

DFT Band Ratio

Calculating ratios from each Fourier coefficient band of EEG signals. Ratios from delta band to theta band showed here.

$$DFT\_delta\_to\_theta\_ratio = \frac{DFT_{delta}(x)}{DFT_{theta}(x)} \tag{4.5}$$

4.3.3 Fourier Coefficient Statistics

Mean

$$DFT\_mean(x) = \frac{1}{N} \sum_{n=1}^{N} DFT(x)_n \tag{4.6}$$

Variance

$$DFT\_var(x) = \frac{1}{N} \sum_{n=1}^{N} DFT(x)_n - \overline{DFT(x)}^2 \tag{4.7}$$

4.3.4 Hjorthś Parameter

Bo Hjorth [Hjor 70] tried to analyze EEG signals using several parameters. These parameters occurred in several other related literature related to EEG analysis. It is therefore worth to use in our research. Those parameters are known as activity, mobility, and complexity.

The first parameter is a measure of mean power representing the activity of the signal. The second parameter is an estimate of the mean frequency and is called mobility. The last parameter gives an estimate of the bandwidth of the signal. Since calculation of Hjorth parameters is based on variance, the computational cost of this method is considered low compared to other methods. Furthermore, the time domain orientation of Hjorth representation may prove suitable for situations where ongoing EEG analysis is required [Vour 00].

Activity

This parameter gives a measure of squared standard deviation of the signal's amplitude. In statistic domain, we may refer this as variance.

The first Hjorth parameter, *activity*, is the variance σ_x^2 of the signal amplitude, where σ_x is the standard deviation.

Mobility

The second parameter is *mobility* and it is defined as the square root of the ratio of the activity of the first derivative of the signal to the activity of the original signal. This can be expressed with standard deviations as mobility $= \sigma_x^{'}/\sigma_x$.

Complexity

The third parameter, *complexity*, is defined as the ratio of mobility of the first derivative of the signal to the mobility of the signal itself: complexity $= \sigma_x^{''}/\sigma_x^{'}$.

Table 4.1: Summary of Features

No.	Features
1	MaximumPositiveAmplitude
2	MaximumNegativeAmplitude
3	Mean
4	Variance
5	DFT_band_delta
6	DFT_band_theta
7	DFT_band_alpha
8	DFT_band_beta
9	DFT_band_gamma
10	DFT_band_delta2theta_ratio
11	DFT_band_delta2alpha_ratio
12	DFT_band_delta2beta_ratio
13	DFT_band_delta2gamma_ratio
14	DFT_band_theta2alpha_ratio
15	DFT_band_theta2beta_ratio
16	DFT_band_theta2gamma_ratio
17	DFT_band_alpha2beta_ratio
18	DFT_band_alpha2gamma_ratio
19	DFT_band_beta2gamma_ratio
20	DFT_mean
21	DFT_variance
22	HJorth_Activity
23	HJorth_Mobility
24	HJorth_Complexity

4.4 Clustering

Clustering is an unsupervised learning scheme, in which data are not labeled. We wanted to organize data into groups which has similarities in some ways. Some data were not able to be labeled, therefore in our research, clustering of data needs to be done.

4.4.1 K-Means

In K-Means clustering we have to guess how many clusters should objects be placed.

The algorithm in pseudo format:

1. Initialize the value of K the number of cluster, N the number of pattern needs to be clustered.
2. Calculate μ_i as mean for each cluster also known as centroid.
3. Determine which cluster an object should be assigned to by comparing the distance between the object and centroid of the clusters.
4. Assign each object x_i to the cluster presented by the nearest distance
5. When all objects are in some clusters. Recalculate centroid to decrease the error function.
6. Repeat step 2 and step 3 until error function decreases.

4.4.2 EM Algorithm

The Expectation-Maximization (EM) algorithm is similar to the K-means procedure in that a set of parameters are re-computed until a desired convergence value is achieved. EM allows us to choose the number of clusters to be formed. As an alternative to K-Means, EM can determine a best number of clusters.

The procedure for EM roughly [Witt 05]:

1. We can assume the number of clusters (known as mixture)
2. Guessing the mean and standard deviation for each cluster and the sampling probability for the cluster
3. Use the probability density function for a normal distribution to compute the cluster probability for each instance.
4. Re-estimate the mean and standard deviation and sampling probability for the cluster

4.5 Classification

For the classification part, we are relying upon some predefined algorithm in WEKA environment [Witt 05].

4.5.1 Decision Table

The simplest way of representing the output is to make it just the same as the input which is in the form of a decision table. We can just look up the appropriate conditions to decide whether or not to decide a particular problem. Less trivially, creating a decision table might involve selecting some of the attributes. The problem is, of course, to decide which attributes to leave out without affecting the final decision.

4.5.2 J48 Decision Tree

Decision tree is an intuitive model to solve classifying problems. Sets of node will ask a question that can break down into other questions and finally ends up with a leaf. Leaves will be classes. Common use of decision tree is with nominal attribute, but it can be extended to numeric attributes.

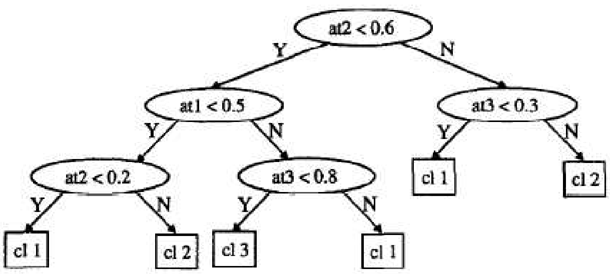


Figure 4.4: Example of a Numeric Decision Tree[Kopr 96]

It is even already proposed by [Kopr 96] to use decision tree for classifying sleep stages for infants. In Figure 4.4, a decision tree from [Kopr 96] to classify sleep stages in infant. The node contains features that need to be classify, while leaves are stages of sleep. The decision tree we used in this research takes the J48 algorithm.

4.5.3 Bayesian Network

Bayesian network is a modeling tool that combines directed acyclic graphs with Bayesian probability. Recently, Bayesian network have found increasing use in complicated problems such as medical diagnosis [Duda 01]. Each node of the network corresponds to a variable, and the edges represent causality between the nodes. The nodes are labeled as A, B, ... and their corresponding variables in lowercase letters. Each edge is directional and joins two nodes (presents influence of one node upon another). In Figure 4.5, node A directly in influences D, B in influences C, and so on.

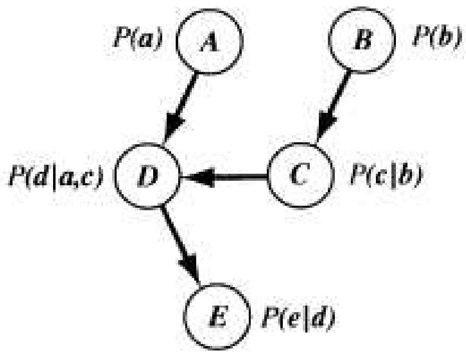


Figure 4.5: Example of Bayesian Network [Duda 01]

4.5.4 Simple Logistic Regression

Logistic regression will be compared also for classification to other method of classifications. Logistic regression is most powerful with binomial distribution of data. In classification terms, this means there are two class of data which needs to be classified. We will see the performance when we are classifying three classes of data. The data is restricted to two values such as yes/no, on/off, survive/die or 1/0, usually representing the occurrence or non-occurrence of some event (for example, bored or not).

4.6 Tools

For the purpose of data processing, we used several tools. We utilized a number of tools from proprietary software that comes with the device and also many other open-source tools.

4.6.1 Truscan Acquisition

This is the tool we use for acquiring signals from subjects. It comes with a lot of features which some of them is not used. This tools is intended for a clinical purpose. It registers subjects as patients with all their detail (including insurance and doctor's name).

Once we have registered our subjects, we can start the recording. The user-interface is somewhat functionally straight-forward. It has descent look and feel. The signal starts in real-time once the indicator level goes below or at 20 kΩ.

The great thing about this tool is that the indicator level of electrodes connected to the monitoring computer is provided. We can indicate whether the recording is good enough (indicated by low impedance). More detail about Truscan Acquisition will be explained in Chapter 5.

4.6.2 Truscan Explorer

Truscan Explorer (TE) is an utility shipped with the DEYMed Truscan 32 EEG device, like the accompanying tools Truscan Acquisition. It acts as an analytical tool from the data we have previously recorded. Truscan Explorer will show up the list of patient that we have previously registered in Truscan Acquisition. All the recordings that have been recorded in Truscan Acquisition will also automatically showed here.

If we look at Figure 4.2, TE provides us with the data conversion part. TE has the capability to transfer the recorded data into several formats. One that we use is to convert to ASCII text file. It has a *Send to MATLAB* button at its user-interface, which we have not able to do it.

We can even start analyzing the signal using TE. If we scroll the data, at an instance we can have a window to be transformed into a frequency domain. In this window, we can see the common frequency band of EEG (see Section 2.4.4).

We use TE also to slice the recordings int segments. A segmented recording will be more useful to use when we want to grab features. Furthermore, it is also much easier to control that particular signal is when a subject is at some stage. More on this tool will be explained in Chapter 6.

4.6.3 EEGLAB

EEGLAB is a collection of functions written in MATLAB, specialized for analyzing EEG data. The main usefulness of EEGLAB is the artifact removal using ICA (for detail on ICA, see **4.2.2**).

EEGLAB will be used for data preprocessing part. The preprocessing part that we need to do, mostly is artifact removal. Artifact removal is one of the strongest point for EEGLAB. In its standard distribution, EEGLAB implemented two ICA algorithms. Those are the the *logistic infomax* ICA algorithm [Bell 95] and the blind separation of signal using *JADE* [Card 97].

Even though, we can separate artifacts with the real signal by using visual inspection, we will be using this tool to help us generate cleaner data for the whole recordings. More on how we use EEGLAB is explained in Chapter 6.

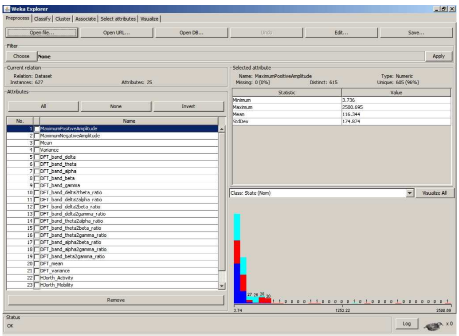
4.6.4 WEKA

WEKA stands for *Waikato Environment for Knowledge Analysis* [Witt 05]. As the name says, WEKA is collection of algorithm used for data-mining and machine-learning purposes. We already mention something about WEKA before, because indeed the classification algorithms that we use are from WEKA. As already mentioned previously, WEKA is common tool to be used in data-mining field.

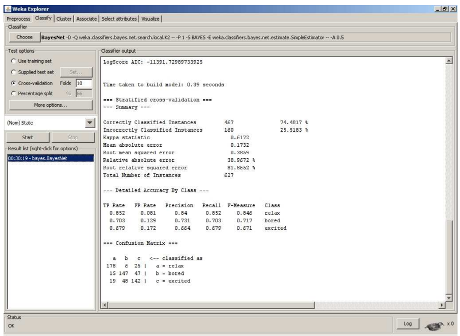
Because it already has a user-interface, we can use this to perform our classification. WEKA provide use with four user-interfaces. The first one is *Explorer*. This will be the only user-interface that we used. It provides the necessary process to input data and output results. The other three user-interface will not be used in this research. They are *Experimenter*, this is very useful if we want to make a new methods or a combination of method for classification or clustering. Next is the *KnowledgeFlow*, this is where we can define our flow of task, for example f we want to input many data from many sources such as databases and text files. It can be done in batch. Last is the command line processor.

WEKA also provided us with application programming interface (API). This enables us to extend the use of WEKA not using their own user-interface but to much broader application. WEKA enables us to connect to database, and flat text file. For this research we used text files as input.

To be able to use WEKA as a classification environment, we need to define a plain text file using their so-called Attribute-Relation File Format (ARFF). ARFF is the heart of the classification that we want to build. Same ARFF file can be applied to several different classifier.



(a) WEKA Explorer Window



(b) Classification in WEKA

Figure 4.6: WEKA Environment

Chapter 5

Data Acquisition

Data acquisition is performed to acquire data to be process in the next part of the research. In this chapter, general experimental setup and experimental setup per scenario are also described.

5.1 General Experimental Setup

Generally, we conducted the acquisition using the following setup, which can be decompose into hardware setup and software setup.

5.1.1 Hardware

In a nutshell, equipments used during the acquisition are:

- Cap

  One thing we are most familiar with in conducting EEG measurement is the use of cap with all electrodes in it. The cap follows the 10-20 system, see 2.4.1. The standardized cap has different head circumference choices. For ElectroCap^{TM}, there are three different circumference, 60cm indicated by a blue-colored cap, 55cm indicated by a red-colored cap, 50cm indicated by a yellow-colored cap.

We also come up with our own electrode placement other than the standardized cap. We created the headband for ease of preparation purposes. The headband electrode placement was also commonly used by several research [Palk 04], [Hona 05]. This headband uses single electrode that is attached to parts of the headband. Because the nature of the headband that circles around scalp, we can only record activity beneath that part of our brain hemisphere. If we refer to 2.4.1, we can only record several pre-frontal, frontal, temporal and occipital scalp.

We built the headband electrode due to the fact that we only want to inspect several activities of our brain. It is therefore, we put single electrode at Fp1, Fp2, F7 and F8 because we wanted to look at the attention level. We also put two electrodes at occipital area, O1 and O2. We also wanted to know the motor control activity, therefore we need one more electrode to be put on central locus, Cz.

(a) Head Cap with 19 Electrodes

(b) Head Band with 7 Electrodes

Figure 5.1: Cap

- Amplifier

  All the electrodes would have to go to an amplifier, sometimes refer as EEG head-box. EEG head-box is where we plug all the necessary electrode ends. If we use single electrodes, the head-box has jacks where we could plug-in the electrodes end. Figure 5.2, shows these jacks according to where the electrodes are placed in our scalp.

Figure 5.2: Truscan 32 EEG Amplifier

If we use the cap, the cap cable's end would have to be plug into a port. In Figure 5.2, this is the white port. The amplifier is powered by 4-AA sized batteries, which can be replaced easily if they out of charge. A LED indicates whether it is powered or not. We can also indicate this using software. From this amplifier, we have to connect to the analog-digital converter using an optical cable.

- Analog-Digital Converter

  EEG adapter is where the data will be connected to the computer. This means that the analog data would have to be converted to digitized version.

- Game terminal

  The game terminal is where the subject will play the game. The detail of this terminal is almost changing all the time. Depends on what type of game is played. It is a regular personal computer (all of the time, we used notebook form factor) with games installed in it.

- Acquisition terminal

  The acquisition terminal is where the EEG data will be recorded.

5.1.2 Software

- Acquisition application

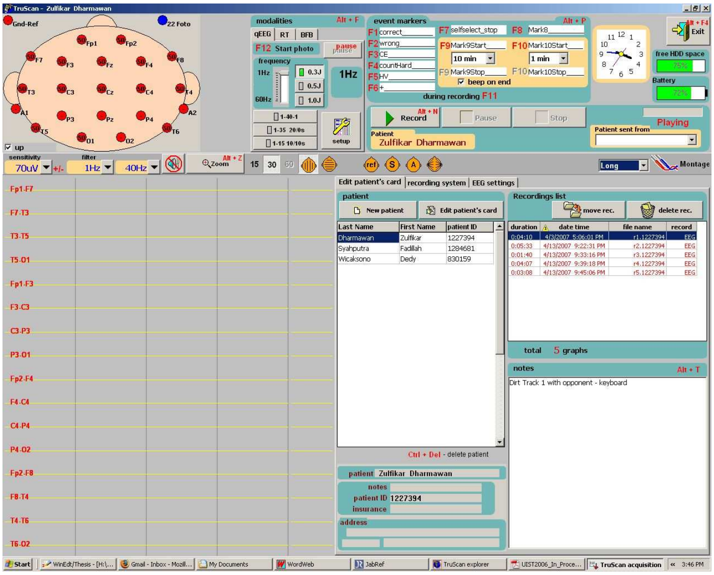


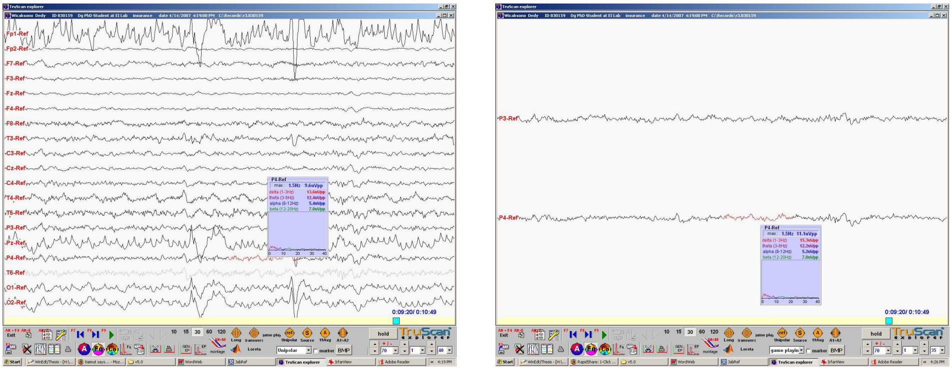
Figure 5.3: Truscan Acquisition

In acquiring the signals, we used the supplied software from the equipment that we used (Deymed). For measurement, we use Truscan Acquisition. This software comes with all the necessary interface to conduct measurement. From setting up the electrodes and also for recording the signals into its proprietary data format (Deymed DAT files).

The interface for the Truscan Acquisition provides impedance control. We can control the impedance level using this controlling user interface.

After we have acquired the signals using the Truscan Acquisition software, we continue with the next software to read the result of our acquisition, the Truscan Explorer. Using this software we can use the somewhat rather clinical user interface to guide us through do simple analysis of the signals.

When we conducted the experiment, often we can not get low impedance level. If we use the whole set of electrodes in the readings, what we get is noisy signal reading. Therefore, we only use and want to concentrate our analysis based on several electrodes only. Truscan Explorer provides us with *Montage Constructor*; it is a tool to create our own view with the choice of electrode to read. By using this we can discard unnecessary electrodes reading.



(a) Unipolar montage (with 19 electrodes) (b) Customized Montage View

Figure 5.4: Truscan Explorer

- Computer games

  Computer games we used vary from racing game, first-person shooters, and chess games. The choice of games, depends on what type of experiment that we are going to perform. We expect users involvement of the games. The game level can have varying challenging levels, which require much cognitive loads of the users.

Figure 5.5 shows the general experimental setup for all experiments. It requires two computers, one for the subject and one for the test supervisor. The test supervisor will have to monitor the signals and certain event happened, he can mark it for further investigation.

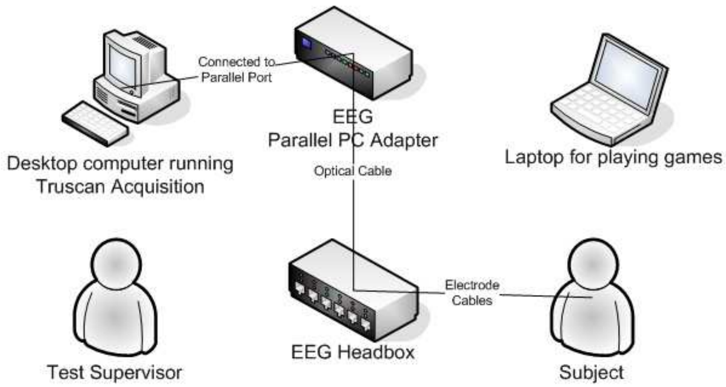


Figure 5.5: Experimental Setup

5.2 Preparation

Preparing for an EEG measurement can be dull and time-consuming, especially if we are not yet familiar with all the devices and auxiliary equipments, such as syringe, gel, paste, etc. Most of the time, it takes about 20 - 30 minutes of preparation. At first, this can be frustrating as the impedance level as indicated on the Truscan Acquisition monitor still high (red sign).

The steps we took for preparing the EEG cap for the experiments are:

1. Apply some EEG paste (i.e. Ten20^{TM}), just a small portion to touch the electrode.

2. Afterwards, each electrode have to be fully filled with the Electro-Gel^{TM} gel.

3. For the subject, cautiously apply a drop of rubbing alcohol (using a cotton bud) to the position of the electrodes.

4. After that apply small amount of *NuPrep* abrasive gel to subject's scalp.

5. There is also the so-called *One-Step AbrasivPlus*, which cuts the steps of putting different substance to our scalp or electrode. We put this on the electrodes and subject's scalp.

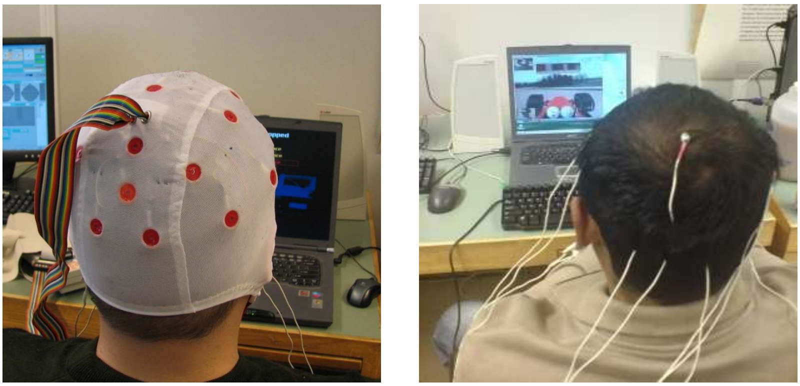
6. Put on the cap to subject's head.

7. Put a belt around the subject's chest as a place to strap the head cap.

8. Look at the Truscan Acquisition monitor, where electrode indicators of impedance level are presented.

9. If the impedance level is still high, add some of the Electro-gel using syringe through the hole. This procedure would have to be done continuously until the impedance level below 10 kΩ

After preparation and subjects are ready to performed the experiments (like shown in Figure 5.6). Subjects are requested to sit down comfortably in the experimental environment (Figure 5.7).



(a) Subject Wearing a Head Cap (b) Subject Wearing a Head Band

Figure 5.6: Recording Brain Signals

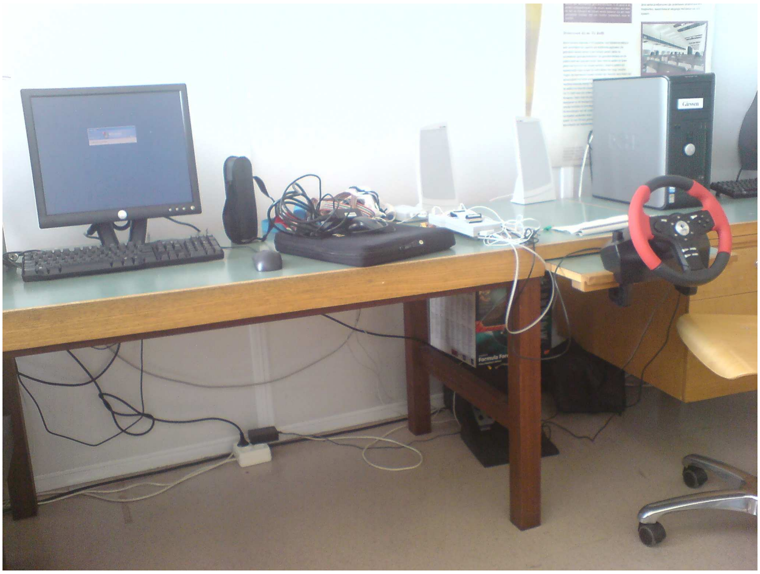


Figure 5.7: Setup for Experiments

5.3 Type of Experiments

We have three experiments. Although the experimental setup are essentially the same, the goal and method are different. Initially, we wanted to have all genre in computer games be represented in this experiment. However, due to time constraint and we do not think that doing experiment for all computer games genre would give significant different than what already proposed.

There are basically two type of games. The game that require extensive use of our motor control (our hands most of the time) and game that demanded our deep thinking. From here, it can be distinguished to several genres. Role-playing game, adventure, puzzle game goes to the more of thinking type of game. For the games that demanded more on our motor control are simulation, first person shooter game, etc. The computer game genre itself still evolving with much more invention and enhancement in computer game technology.

The experiments we did were:

1. Experiment 1 - Playing Racing Game

2. Experiment 2 - Playing Chess

3. Experiment 3 - Playing First-Person Shooter Game

We hope that these games can represent all the genre in computer games. We will explain each of the experiments and their setup in the next sections. After every experiment, we analyzed the data collected. The result will be reported in Section 5.9.

5.4 Experiment 1 - Playing Racing Game Scenario (Rc)

5.4.1 Goal

In the first experiment, we want to be able to know whether we could make a classification of general task in a playing game scenario. A computer game, essentially will have a moment where player would get frustrated because of how difficult it is to play that game and how boring a game can be if it does not offer much excitement component.

5.4.2 Method

For this purpose, we designed a scenario where subjects would be engaged in an extreme different situation.

1. Rest state

   At this stage, subjects tried to be in a relax position. Before the game is played, subject is given instruction on how to play the game, which keys to accelerate, break and go left and right. All of the scenarios below should start with this stage. This scenario lasted for 2 minutes. We helped subject to relax by giving relaxing music.

2. Driving a long straight track without opponents

   The second scenario, subjects would have to control the steering wheel to navigate cars. We also want to know how subjects react without being competed. This session takes about 15 minutes. The idea of having a long recording is that so the subject even at the start of the game was still excited, eventually will be not anymore.

3. Driving a variate and curvy track

   The third scenario also ask the subject to control a steering wheel. Different from the previous scenario, we make the tracks more excited by adding curve and some dynamic scene to the game. We also keep the user to be in a competition with other opponents. Supposedly, this is the ultimate difficulty a user could have. This experiment lasted for about 2-3 minutes depend on the driving ability of the subject (we let the game to finish).

Figure 5.8: TORCS

Racing car simulator we used are TORCS. TORCS stands for The Open Racing Car Simulator. It is an open source computer game, which is somewhat common to be used in research, especially for research in learning algorithm and vehicle control.

In order to customize the difficulty level of the game, we modify the tracks that need to be played by subjects. To modify the track, we need to modify XML file that defines the track. Detail of the XML file that define track we use can be seen in Appendix B, which is an example of defining a long straight track (we used for our experiment). There is a number of track editor that can read this particular format of XML and convert it into tracks picture. We can use this editor to aid us in building the editor.

As for the measurement, we used the electrode cap. Several few early subjects were recorded using the headband, due to technical difficulties of the electrode cap preparation.

5.5 Experiment 2 - Playing Chess (Ch)

5.5.1 Goal

In this experiment, the goal is to be able to know which brain activity tells from using different level of attention or thinking within a computer game player. We think that chess is the ultimate game to be chosen. It lets the subject to think more than to use his/her motor control.

5.5.2 Method

For the chess game, we want to look at the excitement level of subject when doing a hard calculation. Chess can be considered as hard thinking. It is a combination of visual and also planning something ahead. We expected something different than the signal produced when we are playing the more exciting game that requires more coordination of our hand and less thinking.

The experiment used the *Chessmaster* game from *Ubisoft*. It is one of the most famous chess computer games played. There are available several opponents to pick with their own characteristic. This will affect the way they move their pieces, which resulted in a though game or more easy game for subjects.

When playing the game, there is also some period where subjects are in resting state. This is the time when subject has to wait for the other player (the computer) has to do its thinking. Thus, in playing chess, incidentally we get also different scenarios.

- Resting state

  This happened when subject would have to wait for the computer before it moves its pieces. This will depend on the opponent. Some opponents used an enormous of time to do their thinking, some used little time. Supposedly this will affect the signal produced, because if subject have to wait for quite some time, he will get bored.

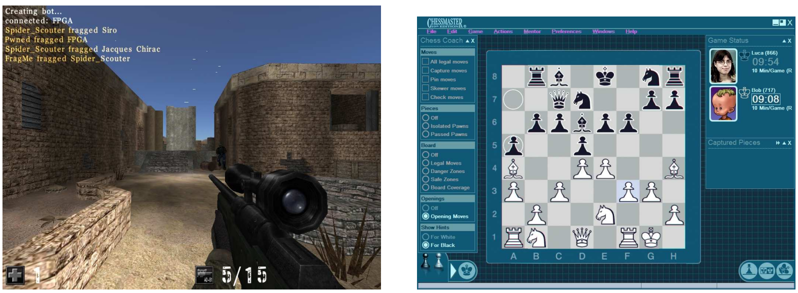
- Thinking time

  This is the period where subject would have to do their thinking. Subject's turn usually indicated by running time telling the subject the amount of time available for him to do the move.

- Almost-reflex time

  If the subject survive the game until the pieces are just a few (king with some other pieces) or if the amount of time to do his move is almost over, supposedly subject would move whatever piece they can just to survive. We call this period an almost-reflex time. We expected this period to have different signals than the previous two.

For the measurement, since this recording was done at early stage of this research, where we have not resolve problem with the electrode cap we use the headband. Besides technical problems reason, the use of headband is reasonable because we want to observe most of all the frontal and prefrontal. This area particularly is related to attention level.



(a) ActionCube (b) Chessmaster

Figure 5.9: Screen-shot of Games for Experiment 2 and 3

5.6 Experiment 3 - Playing First-Person Shooter Game (FPS)

5.6.1 Goal

Difficulty levels can come from several parameters. One of those difficulty levels is the opponent's AI level. In this experiment, we wanted to know how different opponent's level can influence brain activities.

5.6.2 Method

We are using different type of game than previous experiment. For this purpose we are using the *ActionCube* [Henk 07]. Using FPS-type of game, we would like to get an utmost degree of excitement from the user. Users would be engaged in a combat using mouse and keyboard.

In ActionCube, there is a feature to have a continuous play of game. The default setting (or even the only setting) is to play continuous game of 10 minutes. We will use this feature for our experiment in vigilance assessment. ActionCube also has a range of difficulty level that can make the opponent to be smart or dumb. The range is, in an increasing difficulty-level order, bad, worse, medium, good, best.

For the measurement, again since this recording was done at early stage of this research, where we have not resolve problem with the electrode cap, we use the headband.

5.7 Questionnaire

A simple questionnaire was asked at the end of the experiment. This questionnaire acts as a confirmation to what supposed or hope to have in the game (see Appendix C). The questionnaire was mainly asked for the Playing Racing car. Subjects were asked simple questions to confirm his/her excitement.

From the collected answers, all subjects confirm that the second session of the racing car game was boring and the third session to some subjects was very difficult. Some experienced player (have played this game before) said that the third session has the same excitement level as the second session.

5.8 Data Conversion

One of the problem of using a proprietary software is the output can not be further processed. This is also what we encountered in using Truscan Acquisition (TA). TA produced .DAT file which can not be read by other EEG analytical tools, except using its companion analytical tool, Truscan Explorer (TE). Fortunately, TE has a conversion function.

For our purpose of acquiring general analysis of the data, we block the whole recording data to be able to convert it to ASCII text format. Unfortunately all the marks will be gone during the conversion. In our research marks will be used to notify a peculiar signal or when we are doing the experiment with tasks assignment (playing computer chess game).

From the ASCII file, the generated ASCII file can be in fixed point format or floating point. All of the data we converted was in floating point format. The format of the data is row-centric format. Each row represents channel in the recordings. Depends on the sample rate we used during the recording

The generated ASCII file has columns and rows. The rows represent the channel (which is by default 19-channel). The columns represent the sample in time perspective. While, the content of the data are the amplitude.

From the generated .TXT file, we use the pop_impordata() from EEGLAB to bring about the recordings again to this analytical tool. Once it is EEGLAB, we can have it store in MATLAB variable or to other format we wish. EEGLAB comes with a tool called BioSig, which enable us to convert to virtually any biological signal format.

5.9 Raw Results

Before we go into the data processing part (or even the automated artifact removal), we wanted to see what the data looks like right now so far. From the experiments, we acquired data from 20 different subjects, the age range were between 20 - 35 years of old, with most of the subjects were male and only two female subjects. Some subjects participate in more than one experiments.

Experiment 1

Experiment 1 as defined in 5.4 involved 19 participants. As it is said previously there are three sessions. Some participants wear the headband and some wears the electrode cap (due to a problem we encountered on the preparation).

From the headband setup, which we applied to 8 participants, we acquired totally 13423 seconds of 7-channel EEG recordings. In detail, we acquire 546 seconds for the rest state (we did not acquire rest state from 4 early subjects), 8717 seconds for the driving long straight track, and 4160 second for the curvy track with opponents.

From the electrode cap setup, which we applied to 11 participants, we acquired totally 14882 minutes of 19-channel EEG recordings. The detail can be seen in Table 5.1.

Table 5.1: Playing Racing Car Games

Participants	Rest State (minutes)	Driving long straight track (minutes)	Driving curvy track with opponents (minutes)
Subject 1	188	919	295
Subject 2	280	542	199
Subject 3	137	905	313
Subject 4	223	1038	328
Subject 5	181	904	260
Subject 6	190	253	555
Subject 7	257	613	368
Subject 8	121	929	158
Subject 9	142	1102	674
Subject 10	180	527	179
Subject 11	146	1092	684

Experiment 2

Experiment 2 was conducted with 4 subjects. All wears the headbands with electrode emphasize on the prefrontal and frontal area. The collected data from subjects can be viewed in 5.2.

Table 5.2: Playing Chess

Participants	Time In minutes
Subject 1	06:58
Subject 2	07:40
Subject 3	03:44
Subject 4	05:39

For this experiment, totally we acquired more around 28 MB of EEG recordings from five subjects with 7 channels (Fp1, Fp2, F7, F8, Cz, O1, and O2).

Experiment 3

For the experiment 3, we conducted this to 2 subjects. All wears also the headband. Off course, ideally we want to have 10-minutes recording for each difficulty per subjects, but we need some preparation and all the hustle and bustle with the software and equipment totally we have more than 10 minutes per difficulty level (see Table 5.3. We also allow the subject to practice a little bit with the game, this takes some time.

Table 5.3: Playing FPS Game with Different Difficulty Levels

Difficulty Level	Subject 1 (minutes)	Subject 2 (minutes)
Bad	11:56	10:44
Medium	10:50	10:18
Best	10:19	10:12

For this experiment, totally we have 63 MB of raw EEG recordings from two subjects with 7 channels.

5.10 Summary of Data Acquisition

In data acquisition, we gain knowledge and experience in conducting EEG recordings. Even to some extent, we experimented conducting the recording with different kind of electrode setup. One with the standard Electro-Cap^{TM} and the other with a customized headband with single electrode circulating around the circumference of our head scalp.

EEG devices we used was provided by *DEYMed*. We used the acquisition software also from them, called the Truscan Acquisition. The acquisition tool was also used for preparation with the subject, for it has an impedance monitor when electrodes were attached to the device. We take the devices as is without any modification.

The data acquisitions are done throughout three experiments with three different games. The goal of the first experiment is to be able to know whether we could make a classification of general task in a playing game scenario. A computer game, essentially will have a moment where player would get frustrated because of how difficult it is to play that game and how boring a game can be if it does not offer much excitement component. Our method is to ask subjects playing a racing computer game with different tracks scheme.

The second experiment is aimed to be able to discriminate brain activity from using different level of attention or thinking within a computer game player. While in the third experiment, we wanted to know how different opponent's level can influence brain activities. For the second experiment, we asked subjects to play chess against computer. While for the third experiment, we conducted the experiment on a First-Person-Shooter game.

We gained 14882 minutes from playing computer racing games experiment from 11 subjects using head cap. Next, we acquired 30 minutes from 4 subjects using head band for playing chess against computer. Furthermore, we acquired around 60 minutes of playing first-person shooters game from 2 subjects also using head band.

After data were acquired, they are stored in a proprietary format .DAT file. To enable us further process these data, it requires data conversion. We used simple analytical tool software provided again by *DEYMed*, which enable us to have a .TXT file. Unfortunately, we lost marks we had during the recordings.

Chapter 6

Data Processing

In this Chapter, we will describe the data processing part from the Figure 4.2. Included in this part is some part of the artifact removal, feature selection, and how we try to classify the recordings with simple classifier in WEKA environment. For the purpose of data processing, we are using EEGLAB, WEKA, and some functions from MATLAB's signal processing toolbox. Detail about tools we used is described in Section 4.6.

6.1 Artifact Removal

Visual inspection of artifact removal has already been explained in Section 4.2.1. In this part, artifact removal will be done using ICA. After we have the data converted to ASCII text file and loaded in MATLAB, using EEGLAB's infomax ICA algorithm ICA independent components are acquired.

In Figure 6.1, it is shown a map of independent component, which can ease (although not so easy) the task of removing artifactual components. Still an expert eye would be needed.

For example in Figure 6.1, the component 2 is a suspect of an eye artifact. From the topographical map, we can see that much of the activities are in prefrontal area, where the artifacts is mostly happened. Before we

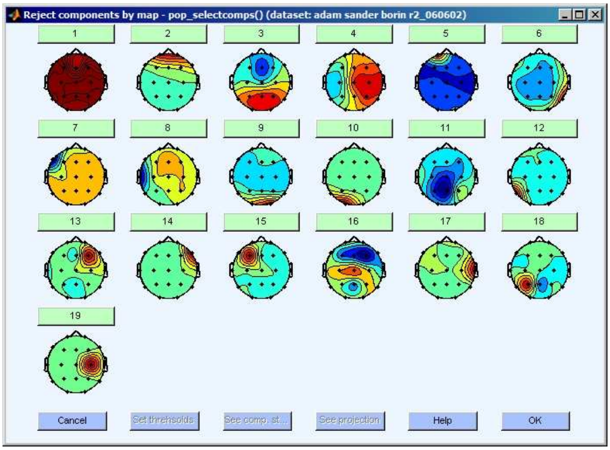


Figure 6.1: ICA's Component Map

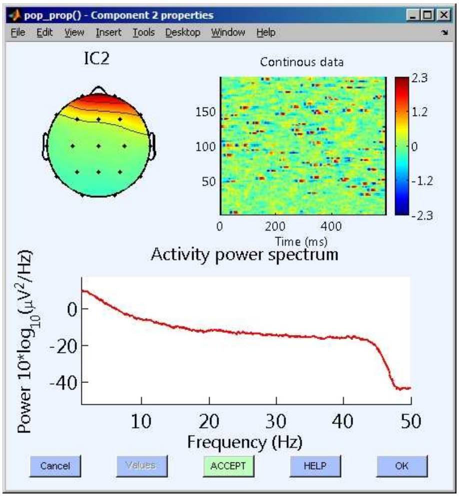


Figure 6.2: Rejecting Eye Artifact

can reject that, we could also look at the detail of that component, like in Figure 6.2. From the activity spectrum, we see that it decreases as frequency gets higher.

Besides, eye movement artifacts, there are also other artifacts. One that is stand out is muscle artifact. In [Delo 03], it is pointed out that to determine whether one component is artifactual component or not. First, we need to examine the scalp map (as shown in Figure 6.1). Next, the component time course, which in Figure 6.2 can be viewed at the right top corner of the window. Next, the component activity power spectrum, as aforementioned. Based on this inspection, we can diminish one of the components shown to acquire cleaner data.

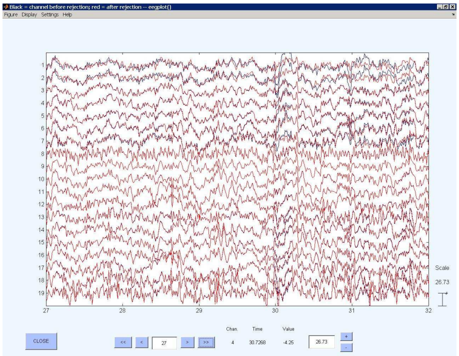


Figure 6.3: Eye Movement Artifact Removed

We did the above process for almost all of the data we acquire. Not only is it an arduous task, it is also time-consuming. We did the the steps to get the ICA weighted for one recording, using 1 GB computer with CPU speed of 1.5 GHz. Still, it takes more than 5 minutes to decompose a variety of length of data.

In Figure 6.3, we showed how discarding one of the suspected components can improve the recordings. The blue signals mean the signal before component was discarded. The red one means the signal after the component was removed. As we can see, particularly eye movement artifact is gone.

6.2 Feature Extraction

For the features we want to calculate see Section 4.3. There are several ways to extract features from brain measurement recordings. One of the common methods is to slice the data into segments. This technique is common in signal feature extraction (see Chapter 3).

Features we used are the same for all the experiments. Though, treatments of each data to acquire features are different. For the Racing experiment, we used the whole data to generate features. While, for playing chess, we use the slicing into segments approach. This is merely because the recordings are much shorter. Lastly, for playing FPS, we also used the whole recordings to generate features.

We are really depending on the data structure in EEGLAB for our recordings. The data structure is called `EEG`. In `EEG`, there is one matrix that is called `EEG.data`, which contain the data we need. Almost the same with the previous data generated by the conversion tool by Truscan Explorer (see Section 5.8), the rows are channels / electrodes, while columns represent sample according to its sample rate.

1. Experiment 1

   As aforementioned, we used the whole recording to extract features. The reason for this is practicality. There are 19 channels that need to be processed and we do not intent to blow up into more than as it is. Data from each of the channel needs to be extracted first. Afterwards, we need to calculate according to the features we wanted to extract.

2. Experiment 2

   For this experiment, we sliced the data into segments of three seconds. The reasons we do this, are basically to get more accurate prediction about the signal data. If we use the whole recording, frequent change of player state can happen. While by slicing into fewer seconds, we will have a much more specific prediction. The tradeoff is there will be an increase of features. This call for a mechanism to select which feature is more important than the other. For this purpose we used the feature selection in WEKA, which we will discuss in the next section.

3. Experiment 3

   Treatment for data from experiment 3 are similar to experiment 1. Except for the fact that we only use 7-channel compared to 19-channel we used in experiment 1.

What we do in practical is to first separate each channel. Each of this channel then go through several calculations. We do this internally in

MATLAB using functions relying upon signal processing toolbox. We do this for each channel per subject. Some of the functions to extract this features are given in Appendix.

6.2.1 Features

The implementation for the features is done using Matlab. At the start of the data processing, we implemented all the features using Java. Unfortunately, due to performance and time reasons, we switch back to use Matlab. One of the reasons why we switch back to Matlab is that we have to implement the Fourier transform ourself, which is as not as robust as with the proprietary Matlab's implementation of Fourier transforms. We will now listed snippets of code for extracting features.

Maximum Positive Amplitude

```
%% MaximumPositiveAmplitude
outData = max(timeData);

strng = [strng num2str(outData) sprintf('\t')];
```

Maximum Negative Amplitude

```
%% MaximumNegativeAmplitude
outData = -max(-timeData);

strng = [strng num2str(outData) sprintf('\t')];
```

Mean

```
%% Mean
outData = mean(timeData);

strng = [strng num2str(outData) sprintf('\t')];
```

Variance

```
%%Variance
outData = var(timeData);

strng = [strng num2str(outData) sprintf('\t')];
```

DFT band

```
function outData = frequency_powers( timeData )

len = length(timeData);

fSample = 128; fftWin = hamming(len);

fftData = fft(fftWin'.*timeData);

fStep = (fSample) / len;

fLows = [0 4 8 13 30];

fHighs = [4 8 13 30 50];

outData = zeros(5,1); for band = 1:5
    nLow = max(1,round(fLows(band) / fStep));
    nHigh = min(len/2,round(fHighs(band) / fStep));

    fftpower = zeros(nHigh,1);

    fftPower = 0;
    if (nLow < nHigh)
        for i = nLow:nHigh
            fftPower = fftPower + abs(fftData(i));
            fftpower(i)=fftPower;
        end
    end
    bandData = fftPower / len;
    outData(band) = bandData;
end
```

DFT band ratio

```
outData = frequency_powers(timeData);

%% DFT band delta
delta = outData(1);

strng = [strng num2str(delta) sprintf('\t')];

%% DFT band theta
theta = outData(2);

strng = [strng num2str(theta) sprintf('\t')];

%% DFT band alpha
alpha = outData(3);

strng = [strng num2str(alpha) sprintf('\t')];

%% DFT band beta
```

```
beta = outData(4);

strng = [strng num2str(beta) sprintf('\t')];

%% DFT band gamma
gamma = outData(5);

strng = [strng num2str(gamma) sprintf('\t')];

%% DFT band delta2theta ratio
delta2theta = delta / theta;

strng = [strng num2str(delta2theta) sprintf('\t')];

%% DFT band delta2alpha ratio
delta2alpha = delta / alpha;

strng = [strng num2str(delta2alpha) sprintf('\t')];

%% DFT band delta2beta ratio
delta2beta = delta / beta;

strng = [strng num2str(delta2beta) sprintf('\t')];

%% DFT band delta2gamma ratio
delta2gamma = delta / gamma;

strng = [strng num2str(delta2gamma) sprintf('\t')];

%% DFT band theta2alpha ratio
theta2alpha = theta / alpha;

strng = [strng num2str(theta2alpha) sprintf('\t')];

%% DFT band theta2beta ratio
theta2beta = theta / beta;

strng = [strng num2str(theta2beta) sprintf('\t')];

%% DFT band theta2gamma ratio
theta2gamma = theta / gamma;

strng = [strng num2str(theta2gamma) sprintf('\t')];

%% DFT band alpha2beta ratio
alpha2beta = alpha / beta;

strng = [strng num2str(alpha2beta) sprintf('\t')];

%% DFT band alpha2gamma ratio
alpha2gamma = alpha / gamma;

strng = [strng num2str(alpha2gamma) sprintf('\t')];

%% DFT band beta2gamma ratio
```

```
beta2gamma = beta / gamma;

strng = [strng num2str(beta2gamma) sprintf('\t')];
```

DFT mean

```
len = length(timeData);

fSample = 128; fftWin = hamming(len);

fftData = fft(fftWin'.*timeData);

x = mean(fftData);

strng = [strng num2str(x) sprintf('\t')];
```

DFT variance

```
x = var(outData);

strng = [strng num2str(x) sprintf('\t')];
```

HJorth Activity

```
function outData = hjorth_activity( timeData )

outData = var(timeData);
```

HJorth Mobility

```
function outData = hjorth_mobility( timeData )
%HJORTH_MOBILITY Summary of this function goes here
%   Detailed explanation goes here

d1 = diff(timeData);

m0 = var(timeData); m2 = var(d1);

if (m0 == 0)
    outData = 0;
else
    outData = sqrt(m2/m0);
end
```

HJorth Complexity

```
function outData = hjorth_complexity( timeData )

d1 = diff(timeData); d2 = diff(d1);

m0 = var(timeData); m2 = var(d1); m4 = var(d2);

if ((m2 == 0) || (m0 == 0))
    outData = 0;
else
    outData = sqrt((m4/m2)/(m2/m0));
end
```

6.3 Building a Simple Classifier

We need to comply with the standard format, which Attribute-Relation File Format. Features we have collected in the previous section needs to be exported or structured in a format recognized by WEKA. The implementation used to be in Java but again due to performance issue, we switched to Matlab. This can be seen in the listing below.

```
function write_arff_classification( directory, state, experiment )
%WRITE_ARFF Summary of this function goes here
%   Detailed explanation goes here

files = read_files( directory ); filename = files(1).name; dots =
strfind(filename,'.'); if (~isempty(dots))
    dots = dots(length(dots));
    if (strcmp(filename(dots:length(filename)),'.arff') == 0)
        filename = [filename '.arff'];
    end
else
    filename = [filename '.arff'];
end

relation = 'Dataset';

time = clock(); file = fopen(filename,'w');

fprintf(file,'%%ARFF File  \n');
fprintf(file,'%%Export time : \n');
fprintf(file,'%%    date (dd/mm/yy) : %d/%d/%d\n',time(3), time(2), time(1));
fprintf(file,'%%    time (hh:mm:ss) : %d:%d:%d\n',time(4), time(5), time(6));

fprintf(file,'@RELATION %s\n',relation);
fprintf(file,'\n\n%% Attributes\n');

fprintf(file,'@ATTRIBUTE MaximumPositiveAmplitude  NUMERIC\n');
```

```
fprintf(file,'@ATTRIBUTE MaximumNegativeAmplitude  NUMERIC\n');
fprintf(file,'@ATTRIBUTE Mean  NUMERIC\n'); fprintf(file,'@ATTRIBUTE
Variance  NUMERIC\n'); fprintf(file,'@ATTRIBUTE DFT_band_delta
NUMERIC\n'); fprintf(file,'@ATTRIBUTE DFT_band_theta NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_beta  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_gamma NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2theta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2alpha_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2alpha_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_beta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_mean NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_variance NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Activity NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Mobility NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Complexity NUMERIC\n');

if (strcmp(experiment,'experiment1') == 1)
    fprintf(file,'@ATTRIBUTE State {relax, bored, excited}\n');
else
    fprintf(file,'@ATTRIBUTE Difficulty {good, medium, worse}\n');
end

fprintf(file,'\n');
fprintf(file,'%%Begining of data\n');
fprintf(file,'@DATA\n'); fprintf(file,'\n');

str = classify_all( directory, files, state);

fprintf(file,'%s\n',str);

fclose(file);
```

The output of this listing are ARFF files. A sample of the .ARFF file can be seen in D.

Out of the generated data, we build a simple classifier using WEKA's implementation of well-known classifiers. (an explanation of WEKA can be read in 4.6. We used WEKA environment's explorer to choose which classifiers to use.

6.4 Feature Selection

In feature selection, we are selecting which feature will be used in the classifier. To do this, we are also relying on the WEKA. There are already some predefined feature selection method implemented in WEKA. Besides using the selected features, we also used the whole feature extracted, to know whether it will give improvement to the whole classification process.

WEKA distinguishes between two evaluation methods. The first one is attribute subset evaluator and the second one is single-attribute evaluator. Attribute subset evaluator means a subset of attributes will be taken and numeric value will be returned. After this a search method will be applied. Furthermore, single-attribute evaluator means a ranked list of attributes will be return.

From several methods available in attribute subset evaluator, we choose the *CFsSubsetEval*. Searching will be done using arbitrary methods available. Moreover for the single-attribute evaluator, we choose the *InfoGainAttributeEval*.

1. CfsSubsetEval

   This feature selector method takes a subset of features and return a number which measure a quality of the subset. *CfsSubsetEval* assesses the predictive ability of each attribute individually and the degree of redundancy among them, preferring sets of attributes that are highly correlated with the class but have low inter correlation [Witt 05].

2. InfoGainAttributeEval

   InfoGainAttributeEval evaluates attributes by measuring their information gain with respect to the class. It discretizes numeric attributes first using the MDL-based (*minimum description length*) discretization method (it can be set to binarize them instead) [Witt 05].

6.5 Summary of Data Processing

Due to activities being conducted are playing computer games, which undeniably would have a lot of muscle activities and eye blinks. This is also combined with the instability of the impedance level throughout the recordings. The collected data initially has a lot of artifacts.

We performed different types of experimentation with the data. To the raw data we first apply ICA algorithm in order to decompose into independent components. The idea is to discard unwanted artifacts such as eye blink and muscle activities. For each data, we performed ICA decomposition. To perform this, we use EEGLAB tool that enable us to visually see the impact of the removal. Unfortunately, this would still have to be done manually for each data not as batch. Thus, it took much time to preprocess the whole data.

Further in data processing, we apply some machine learning techniques to the data in order to build a simple classifier for determining player state when playing computer games. We extracted common features used in signal analysis, mostly based on Fourier coefficients and EEG rhythm waves band. For this purpose, we implemented some extracting functions under MATLAB.

Once we calculated all the features for all data, we converted the results into flat file named .ARFF file. This format is used by WEKA. WEKA is an environment for knowledge analysis. It is already came with a set of common classification techniques, clustering methods, and feature selections. We used WEKA to characterize signals from each experiment we have conducted.

Chapter 7

Results of Analysis

In this chapter the result of our experiment is given. We also evaluate the performance of the classifier and clustering algorithm we have used in the WEKA environment. We explain the procedure for processing the data in the previous chapter, here we put more emphasize on the result. Moreover, the results are compared between several processing scheme and also with or without artifact removal.

From the data acquisition, we acquired data from 20 different subjects. The age range were between 20 - 35 year of old, with most of the subjects were male and only two female subjects. Some subjects participated in more than one experiments. As has already mentioned in Chapter 5, there are 3 experiments that we have conducted. We distinguish the result from these experiments.

For the evaluation we treat the results of experiments based the dependency of the subjects as:

- Subject-Dependent (SD)

  This is a subject-dependent evaluation. We combine data from all subjects as one per session and evaluate them. Data portions of the same recording session and same subjects were used for training and testing.

- Subject-Independent (SI)

  For this type of evaluation, data from different subjects and different session were used for training and testing.

From the data portions point of view, evaluation especially for the classification can be distinguished into:

- Percentage Split

  We split the data into training and testing data. This will couple with previous treatment of evaluation. We go with the common division of training and testing data, which is 66% of data are used for training data and 33% are used to test the classifier.

- Cross-Validation

  Sometimes the amount of data used for training and testing may not be representative. We might even want to interchange the role training and testing data, which is training the system on the test data and test it on the training data. This is done with a 50:50 split between training and testing data. In cross-validation, we decide on a fixed number of *folds* or partitions of data. For our evaluation we use 10 folds. This means that data are partitioned into 10 part. Each will be used as training and test data.

Whenever is possible we divided the data until its smallest division. For example for experiment 1, we can divided the data into per session and per subject. While, experiment 2 and experiment 3 until per subject. We presented the result according to this division.

Other than that, we look up some reference how to treat the data. Some references claim that machine learning techniques does not work well with time dependent data. Thus, we divide data into segments. For this purpose we consider 30-second slice of data to be extracted as feature would be sufficient. In some references a 3-second slice is common, but since we acquire an enormous of time, we decided to make the segments as 30 second. We come up with a full-length data and segments.

7.1 Results of Experiment 1

Experiment 1 as defined in Section 5.4 involved 14 participants. Among 14 participants most of them wore the head cap, only three wore the head band, due to technical problems. To standardize the experiment, we discard the three recordings using head band. As already mentioned, the experiment consisted of three sessions. The first session is a relaxing session with no activities happening. The second one is a racing game playing session with low excitement and relatively easy difficulty level. Finally the last session involved a more exciting and challenging task.

The amount of time for this experiment had already been shown in Section 5.9. In that particular section, we acquired raw data to be processed. Processing includes applying ICA to the acquired data, extracting feature and classify and/or cluster them in WEKA.

For classification problems, it is common to measure classifiers performance in terms of the error rate. The classifier predicts the class of each instance: if it is correct, which is successfully classified, if not, it is an error.

Table 7.1: Classifiers Comparison in Experiment 1 in Full Length

Setup	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Cross-Validation	78.1499 %	78.9474 %	74.4817 %	77.6715 %
Percentage Split	72.4299 %	76.1682 %	70.0935 %	77.5701 %

In Table 7.1, we see accuracy level (error rate) of the chosen classifier compared with two test options. In general, we can say that by using cross-validation option give us a better accuracy. Compare to Table 7.2, we get a worse result than with the full-length data.

Next step we would like to evaluate the error rate for subject dependent treatment. As aforementioned, subject-dependent treatments mean that the training and testing data come from the same subject. In this experiment, we use the percentage split method. As much as 66% of the whole data in segments were used for training and 33% were used for testing.

Table 7.2: Classifiers Comparison in Experiment 1 in Segments

Setup	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Cross-Validation	67.8086 %	70.435 %	60.5834 %	67.6217 %
Percentage Split	66.6056 %	67.9792 %	59.5543 %	66.7582 %

Table 7.3: Classifiers Comparison in Experiment 1 Subject-Dependent

Participant	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Subject 1	86.6228 %	90.1316 %	85.636 %	89.1447 %
Subject 2	72.6316 %	79.2481 %	69.6241 %	77.4436 %
Subject 3	71.2206 %	69.2049 %	57.3348 %	69.6529 %
Subject 4	81.4815 %	84.7953 %	76.3158 %	84.5029 %
Subject 5	77.6608 %	78.9474 %	63.3918 %	79.7661 %
Subject 6	75.0376 %	77.4436 %	69.9248 %	80 %
Subject 7	77.0218 %	80.4878 %	73.0424 %	81.1297 %
Subject 8	66.1538 %	67.3684 %	59.5469 %	70.8502 %
Subject 9	86.8421 %	86.9674 %	85.213 %	86.0902 %
Subject 10	73.3603 %	77.4899 %	70.5263 %	70.5263 %
Subject 11	78.2456 %	77.8947 %	75.614 %	77.5439 %
Average	76.9344 %	79.0890 %	71.4700 %	78.7864 %

We see in Table 7.3, J48 Decision Tree and Simple Logistic gave the best accuracy. If we look at the stability of the classifier, Simple Logistic gave us the least fluctuation in the result.

We go further into evaluating subject independent data. Subject independent data means that we use data from all subjects as training data and use only data from one subject as testing data.

We can see in Table 7.4, again J48 Decision Tree perform the best classification. This time, surprisingly Decision Table comes second. While Simple Logistic is not that stable anymore. It even performed as the worst for Subject 6, while other classification methods have a better result.

Table 7.4: Classifiers Comparison in Experiment 1 Subject Independent

Participant	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Subject 1	85.4167 %	93.6404 %	79.386 %	77.5219 %
Subject 2	75.9398 %	91.1278 %	58.6466 %	71.5789 %
Subject 3	67.5252 %	82.4188 %	61.5901 %	63.0459 %
Subject 4	77.2904 %	90.5458 %	65.2047 %	76.7057 %
Subject 5	77.4269 %	86.9006 %	70.4094 %	74.152 %
Subject 6	47.6692 %	83.1579 %	43.3083 %	41.9549 %
Subject 7	69.9615 %	75.7381 %	64.8267 %	70.9884 %
Subject 8	66.8016 %	85.587 %	51.0931 %	59.5951 %
Subject 9	87.4687 %	94.1103 %	61.6541 %	79.3233 %
Subject 10	66.8826 %	89.0688 %	62.834 %	60.4049 %
Subject 11	76.4912 %	90.3509 %	66.8421 %	76.1404 %
Average	72.6249 %	87.5133 %	62.3450 %	68.3101 %

Table 7.5: Accuracy Level for in Experiment 1

Participant	All sessions	Session 1 vs Session 2	Session 1 vs Session 3	Session 2 vs Session 3
Subject 1	89.1447 %	99.5935 %	96.3636 %	89.8113 %
Subject 2	77.4436 %	89.0977 %	84.2105 %	79.7895 %
Subject 3	69.6529 %	92.1053 %	88.1579 %	73.4336 %
Subject 4	84.5029 %	97.552 %	93.6288 %	85.3547 %
Subject 5	79.7661 %	95.4678 %	97.5439 %	80.7018 %
Subject 6	80 %	98.0263 %	77.3279 %	95.0276 %
Subject 7	81.1297 %	95.8763 %	99.2308 %	82.6087 %
Subject 8	70.8502 %	96.6912 %	98.3425 %	72.1649 %
Subject 9	86.0902 %	93.133 %	93.0556 %	88.478 %
Subject 10	70.5263 %	98.9975 %	99.6241 %	70.3608 %
Subject 11	77.5439 %	98.4649 %	99.5614 %	77.5641 %
Average	78.7864 %	95.9096 %	93.3679 %	81.3905 %

Next step is to performed evaluation based on part of sessions. We confronted session 1 with session 2, session 2 with session 3, and session 1 with session 3. For this purpose, we use the cross-validation method with J48 Decision Tree classifier.

As we can see in Table 7.5, again it is all depending on the subject. In general, we can say that session 1 compare to session 2 and session 3 performed well in classification. While in performing classification between sessions 2 and session 3 did not performed as it did with session 1.

We already performed cross-validation, percentage split with subject-dependent and subject-independent data. Out of the whole evaluations, for this experiment we can say that the J48 Decision Tree performed well.

7.2 Results of Experiment 2

Experiment 2 as defined in Section 5.5 was conducted with 4 subjects. All wears the headbands with electrode emphasize on the prefrontal and frontal area. We only performed clustering in the experiment 2, for the label is not known. Since we do not know the label, one of the points of clustering is to know which instance of data goes to which class from the clustering.

We slice the recordings into segments of 30 seconds. Experiment 2 is a chess playing game, we know that there are at least two player states, that is thinking hard (during a player's turn) and resting (when it is the opponent's turn). Ideally, we have at least two clusters of data of these states.

Using `AddCluster` method, we can do cluster assignment to each of the instance. Using some already predefined algorithms (discussed in Chapter 4, we compare the results as shown in Table 7.6.

Table 7.6 shows the result of applying clustering algorithms to subject-dependent (SD) data. Each data treated using 10-folds cross-validation. Therefore, training and testing came from the same data. We also tried the subject-independent (SI) treatment using percentage split. We use data from all subjects as training then test it using subjects-specific data.

Table 7.7 depicted a more stable result for the EM algorithm, although using Simple K-Means a little bit indistinguishable from one cluster to the other. Yet, this is using a subject independent training data.

Table 7.6: Comparison between Clustering Method in Experiment 2 (SD)

Participants	EM Algorithm	Simple K-Means
All	3 clusters (35:149:254)	2 clusters (50:388)
Subject 12	4 clusters (39:30:27:16)	2 clusters (81:31)
Subject 13	6 clusters (26:25:2:15:32:12)	2 clusters (55:57)
Subject 14	5 clusters (42:1:12:37:6)	2 clusters (47:51)
Subject 15	7 clusters (20:10:7:11:22:8:38)	2 clusters (93:23)

Table 7.7: Comparison between Clustering Method in Experiment 2 (SI)

Participants	EM Algorithm	Simple K-Means
Subject 12	4 clusters (4:42:30:36)	2 clusters (5:107)
Subject 13	4 clusters (82:2:27:1)	2 clusters (2:110)
Subject 14	5 clusters (24:14:37:1:22)	2 clusters (16:82)
Subject 15	4 clusters (30:29:29:28)	2 clusters (27:89)

After we assign classes to the instance, we classify the data. We used the result from the subject-dependent to have a realistic result for a specific subject. For this purpose, we used J48 Decision Tree classifier, which from previous experiment proved to have a stable result. We also use only the Simple K-Means algorithm to have a compatibility in the process.

Table 7.8: Classification Accuracy Level for Experiment 2

Participants	Cross-Validation	Subject 12	Subject 13	Subject 14	Subject 15
All	94.9772%	59.8214%	75.8929%	7.1429%	52.5862%
Subject 12	94.6429%	94.8718%	31.25%	48.9796%	78.4483%
Subject 13	94.6429%	72.3214%	94.8718%	34.6939%	75%
Subject 14	92.8571%	34.8214%	30.3571%	82.3529%	33.6207%
Subject 15	97.4138%	71.4286%	50.8929%	32.6531%	97.5%
Average	94.90678%	66.65292%	56.65294%	41.16448%	67.43104%

Table 7.8 gave us the result of classification using J48 Decision Tree. We tried using several other algorithms and the results showed not much of

a different. The upper header means the testing methods, while rows are the training data used.

There is a cross-validation column, which means the testing data also comes from the same set as the training data. Furthermore, if the data from a specific subject is confronted with the same subject means that we performed a percentage split to the data, with 66% of the subject data as training data and the rest as testing data.

From the result, we have not so surprising numbers. One striking result is the result from subject 14. Even for performing classification within the same subject depicted a disappointing result, below 90 %. It also occurred if we used recordings from all subjects as training data.

7.3 Results of Experiment 3

Experiment 3 as defined in Section 5.6 was conducted to 2 subjects. The same subjects also performed task in experiments 2 (subject 14 and subject 15). All wears also the headband. Since this is a controlled task, we have totally 30 minutes from three different game difficulties. Thus, in total we have 60 minutes of recording.

In Experiment 3, the goal is to know whether we can distinguish different brain activity when a computer game player is confronted with different difficulty levels. Thus, the treatment is relatively similar to Experiment 1. We will be using full length recordings, because we wanted to know the trend during the whole 10-minutes of recording.

Table 7.9: Comparison between Classifiers in Experiment 3 Full Length

Setup	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Cross-Validation	45.2381 %	61.9048 %	45.2381 %	57.1429 %
Percentage Split	33.3333 %	40 %	33.3333 %	66.6667 %

One of the reasons why the result (shown in Table 7.9) is not so satisfying is because there is not much data to evaluate. There are only 42 instances of data. Although we performed more or less 60 minutes of recordings, 10 minutes of recording will become one instance with observing to only 7 electrodes. Due to unsatisfying results, we also try to slice the recordings into segments of 30-second.

Table 7.10: Comparison between Classifiers in Experiment 3 Segments

Setup	Decision Table	J48 Decision Tree	Bayes Net	Simple Logistic
Cross-Validation	63.3459 %	62.312 %	63.0639 %	64.3797 %
Percentage Split	62.7072 %	62.7072 %	62.1547 %	66.0221 %

Table 7.10 compares the result of classifying recordings of experiment 3 sliced into 30-second segments. The result gave a much improve results compared to the result in Table 7.9. But again, still it does not say much about distinguishing different activities due to different levels of difficulty.

Similar to experiment 1, we treated the data in comparison with different difficulty levels. Surprisingly we received subject-dependent results (as can be seen in Table 7.11).

Table 7.11: Accuracy Level for in Experiment 3 Cross Validated

Participant	All difficulties	Difficulty Good vs Difficulty Medium	Difficulty Good vs Difficulty Worse	Difficulty Medium vs Difficulty Worse
Subject 14	80.6818 %	100 %	100 %	61.9048 %
Subject 15	44.8661 %	65.3061 %	62.4585 %	59.4684 %
Average	62.77395 %	82.65305 %	81.22925 %	60.6866 %

We also performed comparison of difficulty level but using testing data from a specified subject. It turned out that the result is not much of a difference from the previous evaluation with cross-validation test option.

As we thought that the brain activities between difficulty levels are not easy to be distinguished, we also performed clustering to the data. We

Table 7.12: Accuracy Level for in Experiment 3

Participant	All difficulties	Difficulty Good vs Difficulty Medium	Difficulty Good vs Difficulty Worse	Difficulty Medium vs Difficulty Worse
Subject 14	80.5195 %	87.42 %	86.3839 %	61.9048 %
Subject 15	48.6607 %	46.5986 %	37.8738 %	61.4618 %

performed the clustering using Simple K-Means algorithm. Testing data are also being clustered using Simple K-Means to be compatible with the training data. The source of the data is the same with the previous evaluation with supervised classification.

In Table 7.13, we showed the result of the classification from clustering procedure. Testing data are derived from the pair-wise results as aforementioned (see Table 7.12, which we clustered.

Table 7.13: Accuracy Level for Experiment 3 using Clustering

Participant	3 clusters	2 clusters (Good vs Medium)	2 clusters (Good vs Worse)	2 clusters (Medium vs Worse)
Subject 14	46.5909 %	100 %	0 %	62.8571 %
Subject 15	46.4286 %	61.2245 %	44.186 %	37.8738 %

7.4 Impacts of ICA

All the data evaluated previously are done with ICA applied at the stage before data processing is taken place. After data are acquired, ICA were applied. Surprisingly, we achieved a decrease in accuracy level, as can be seen in Table 7.15.

The decrease happened for all full-length data and segments of data. We treated the data using the 10-fold cross-validation. Using 10-fold cross-validation we achieved a decreasing accuracy level. It occurred using al-

Table 7.14: Accuracy level between with ICA and without ICA

Experiment	with ICA	without ICA
Experiment 1	67.9792 %	70.4212 %
Experiment 2	93.9597 %	97.3154 %
Experiment 3	62.7072 %	64.3646 %

most all classifiers. In Table 7.15, in particular we used the J48 Decision Tree.

Possible cause of this phenomenon is that we rejected component in ICA decomposition that also contains a specific brain activities. We also think that the visual inspection at the early stage after data acquisition could improve the quality of the recordings. We can scrap the artifact without going into further process.

We then tried to apply using different training data and testing data. First, we tried the training data with ICA applied and the testing data without ICA applied. After that we also apply the opposite way. We can see in Table 7.15 the results. In brackets are the result if both ICA or non-ICA were applied to the data. Bracket in the second column means ICA was applied to the training and testing data, while the third column means no ICA were applied to both training and testing data.

Table 7.15: Accuracy level with different application of ICA to training and testing data

Experiment	ICA to training data and no ICA to testing data	ICA to testing data and no ICA to training data
Experiment 1	64.425 % (90.5458 %)	75.2437 % (93.5673 %)
Experiment 2	22.4138 % (52.5862 %)	11.2069 % (22.4138 %)
Experiment 3	76.2987 % (76.3393 %)	40.1786 % (89.2857 %)

Thus it becomes clear that the removal of one or both component in ICA containing eye activity does not necessarily improve the results and may even lead to a decrease of accuracies. Note that this is due to the fact that removal of eye activity at times would still need an expert eye to perform.

7.5 Summarizing Result after Feature Selection

Besides the application of ICA, processing of the data, feature importance could also affect the accuracy level of the classification process. It could also a method to lower the dimension of feature that needs to be put into account.

The results shown in the previous sections previously are applied on all features. In WEKA, there are several feature selection methods, which we can apply to our data. For the explanation of the method we used to select features please refer to Section 4.3. Those feature selectors are CfsSubsetEval, InfoGainAttributeEval, PrincipalComponents.

Selected features of each feature selector (see Table 7.16) then applied to the data. In Table 7.16, we select features for data in experiment 1. Selected features are different from one data to another. It is also different if we treat the data even from the same experiment differently (full-length or segments). PrincipalComponents method return a ranked attributes based on multiplication of several features.

Table 7.16: Feature Selection Method and Selected Features

Feature Selector	Selected Features (list of feature please see Chapter 6)
CfsSubsetEval	3, 11, 14, 17, 18
InfoGainAttributeEval	17, 18, 14, 11, 19, 16, 7, 9, 23, 13
PrincipalComponents	9 new features

We used the J48 Decision Tree to classify the data with 10-fold cross validation. We applied some feature selection rules provided by WEKA to our .ARFF file, which gives results in Table 7.17.

Table 7.17: Comparison of Accuracy Level per Feature Selection Method

Feature Selector	Experiment 1	Experiment 2	Experiment 3
All Features	70.435 %	94.9772 %	62.312 %
CfsSubsetEval	68.8259 %	96.5753 %	62.688 %
InfoGainAttributeEval	68.3172 %	95.4338 %	63.7218 %

The results show that there are not much of improvements in accuracy compared to the results aforementioned in the previous sections. In fact, in Experiment 1, it decreases the accuracy level.

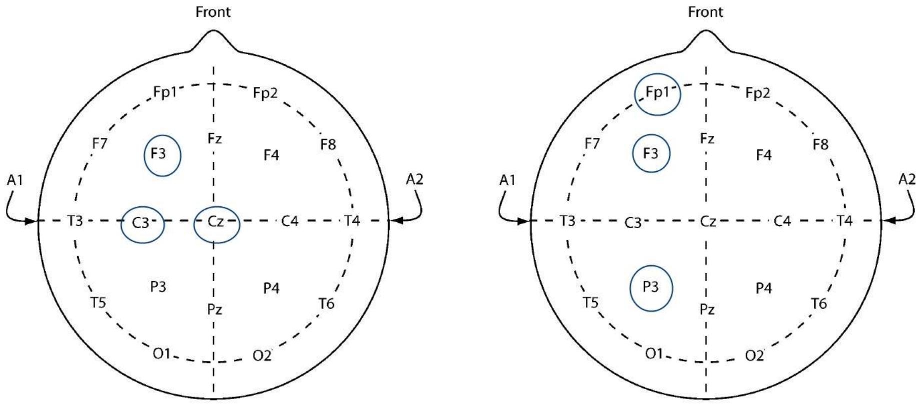
7.6 Electrode Significance

One of the difficulties in all of the experiments is to perform the same session over and over again. Conducting one session is already time-consuming and subjects are reluctant to do the same session. We used the same procedure for feature selection to know which electrode is significantly matter.

Table 7.18: Electrodes Significance for Experiment 1

Feature Selector	Selected Features (detail of number refer to Chapter 6)
CfsSubsetEval	*c3 (5), cz (18), f3 (17), f8 (8), f8 (20), fp1 (14), o1 (11), o2 (6), o2 (19), p3 (3), p4 (9), t4 (16), t6 (19)*
InfoGainAttributeEval	*fp1 (14), p3 (18), f3 (17), p3 (17), fp1 (17), o2 (6), p4 (9), t5 (5), c4 (18), pz (18)*

We compare only two selection methods *CfsSubsetEval* and *InfoGainAt-*



(a) Using *CfsSubsetEval* (b) Using *InfoGainAttributEval*

Figure 7.1: Electrode Significance for Experiment 1

tributEval. Table 7.18 depicted the result from experiment 1. We see that c3, cz and f3 are three of the most significant electrodes according to *CfsSubsetEval*. While, according to *InfoGainAttributEval*, fp1, p3 and f3 are three of the most significant electrodes.

Table 7.19: Electrodes Significance for Experiment 3

Feature Selector	Selected Features (detail of number refer to Chapter 6)
CfsSubsetEval	*cz (7), cz (18), cz (19), f8 (4), f8 (12), f8 (22), o1 (17), o1 (18), o1 (19)*
InfoGainAttributeEval	*cz (18), f8 (4), o1 (17), o1 (19), o1 (18), f8 (22), fp2 (14), f8 (17), f7 (14), fp2 (7)*

Also for Experiment 3 we compare only two selection methods *CfsSubsetEval* and *InfoGainAttributEval*. Table 7.19 depicted the result from

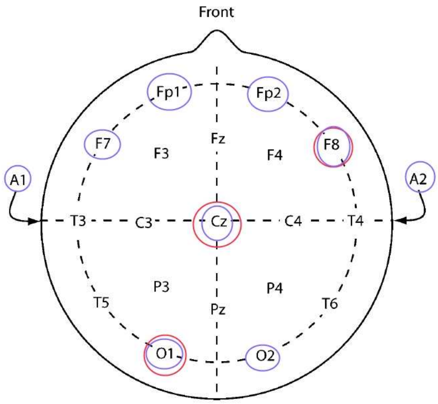


Figure 7.2: Electrode Significance Experiment 3

experiment 3. We see that cz, f8 and o1 are three of the most significant electrodes according to *CfsSubsetEval*. Exactly according to *InfoGainAttributEval*, cz, f8 and o1 are three of the most significant electrodes.

7.7 Final Summary of Results

From experiment 1, we applied classification rules to data from user playing racing games in three different sessions. We compared the result from different subjects. There are the subject-dependent and subject independent evaluation. The first one means that we trained the data and tested it on the same data but different portions. While the second one means that we trained the data and tested it per subject distinctively.

We achieved as high as 90.1 % in subject-dependent evaluation, while in subject independent we achieved 93.64 %. The subject-dependent evaluation is based in percentage split, with 66% of the data as training data while 33% as the testing data. For the subject independent data, we evaluate the whole data set for training and also one subject data as testing data.

Possible explanation why a subject independent achieve a slightly better classification result is due to the use of testing data as portion from the same data. One subject data is gathered from the whole data, which is used as the training data.

We also did a comparison between classifying the whole different sessions and only pair-wise. We achieved as high as 99.59 % for distinguishing between relaxed session versus bored session. For classifying relaxed session and excited session, we achieved as high as 99.62 %. Finally for differentiating between bored session versus excited session, we achieved as high as 95.02 %.

One possible explanation of the high accuracy, we are limiting the training data into two classes instead of three unlike the previous evaluations. Compared to all sessions classified together, indeed we achieve a better results. One thing to be noted, it is strange to see the accuracy for classifying excited session versus bored session is lower compared to relaxed session versus excited or even relaxed versus bored. We expect to have a higher accuracy in distinguishing between excited and bored. Possible cause can be inferred because relaxed session only took about 2 to 3 minutes, while bored session took about 10 to 15 minutes.

Experiment 2 displays the use of clustering method to unlabeled data. Two methods were used simple K-Means and EM algorithm. For K-Means we specifically restrict the cluster into two. This is a prior knowledge about the condition of the data, which we want to cluster. A cluster of brain activity when a person is thinking very hard (at his turn on chess) and thinking moderately (not on his turn). While, when we are applying EM algorithm, it clusters into more than two clusters.

Again, we applied the subject-dependent and subject-independent procedure for the data.For the subject-dependent procedure, we achieve a fluctuation of clusters. When we applied to all subjects, we achieve 3 clusters. While, when we applied only for a specific subject we have more than 4 clusters. If we applied the subject-independent procedure, we get a more stable clusters of 4 and 5.

We used the result of our clusters into classification. Besides classifying from cluster depicted from the same subject, we also did a classification cross-subjects. Not so surprising, the result is not so good. This is reasonable considering the clusters supposed to be fluctuating in results, even though we apply the results from simple K-Means to the classifier.

In experiment 3, we are trying to distinguish brain activity between different types of game difficulties. The results as shown in Table 7.9 and 7.10 are below 70 % in accuracy. We also distinguish using all three difficulties that we want to classify (good, medium, worse) and pair-wise as we also did with the previous experiment.

The result from classifying all difficulties and pair-wise, again, depend on subjects. One subject had astonishing results (more than 80 %), while the other one had below 50 % accuracy level. It increases when we are only classifying pair-wise for good versus medium and good versus worse. It is not the case with medium versus worse, one subject had a decrease in accuracy while the other had an increase.

Possible cause for these unstable occurrences is because of lack of data for one subject. We only conducted one session per difficulty. This could also mean that it is not so obvious to recognize change of brain activity given a different difficulty when the game played is relatively the same.

We perform artifact removal before the data is processed, either by doing visual inspection and removing part of the recording that has suspected artifacts and after performing ICA decompositions. We compared the results from without applying ICA to the data and with ICA. Surprisingly, what we had is a decrease of accuracy in data with ICA applied to it.

What we did in comparing the two results (with and without ICA) is accuracy level of classification. To get a better accuracy of classification, unique features would be affect the outcome. We see that artifacts makes extracted features become unique and specifically distinguish from one to the other. Removing artifact can mean that we are scrapping almost key features to the data. Although, the data may not be correct.

Selecting features again depends on which experiment is being conducted. As can be seen in Table 7.17. Looking at the results in that table, we see that experiment 1 had a decrease of accuracy level, while experiment 2 and experiment 3 mostly had an increase. The same possible cause as with the application of ICA, means that the key defining feature were totally scrapped.

Besides significance of features, we also did analysis of electrode significance. The same feature selection methods were used. For experiment 1, significance mostly occurred among left electrodes. While for experiment 3, electrode Cz turned out to be the most important electrodes.

Chapter 8

Conclusions

8.1 Summary of Research

In this research, we have conducted series of work that stretches from biomedical field, computing, and even psychology. Earlier in this thesis, we have broken down the work into smaller works, as can be seen in Figure 1.1. We divided the work into data acquisition and data processing blocks.

8.1.1 Data Acquisition

In data acquisition, we gain knowledge and experience in conducting EEG recordings. Even to some extent, we experimented conducting the recording with different kind of electrode setup. One with the standard Electro-Cap^{TM} and the other with a customized headband with single electrode circulating around the circumference of our head scalp.

The data acquisitions are done throughout three experiments with three different games. The goal of the first experiment is to be able to know whether we could make a classification of general task in a playing game scenario. A computer game, essentially will have a moment where player would get frustrated because of how difficult it is to play that game and how boring a game can be if it does not offer much excitement component. Our method is to ask subjects playing a racing computer game with different tracks scheme.

The second experiment is aimed to be able to discriminate brain activity from using different level of attention or thinking within a computer game player. While in the third experiment, we wanted to know how different opponent's level can influence brain activities. For the second experiment, we asked subjects to play chess against computer. While for the third experiment, we conducted the experiment on a First-Person-Shooter game.

We gained 14882 minutes from playing computer racing games experiment from 11 subjects using head cap. Next, we acquired 30 minutes from 4 subjects using head band for playing chess against computer. Furthermore, we acquired around 60 minutes of playing first-person shooters game from 2 subjects also using head band.

8.1.2 Data Processing

We performed different types of experimentation with the data. To the raw data we first apply ICA algorithm in order to decompose into independent components. The idea is to discard unwanted artifacts such as eye blink and muscle activities. For each data, we performed ICA decomposition. To perform this, we use EEGLAB tool that enable us to visually see the impact of the removal. Unfortunately, this would still have to be done manually for each data not as batch. Thus, it took much time to preprocess the whole data.

Further in data processing, we apply some machine learning techniques to the data in order to build a simple classifier for determining player state when playing computer games. We extracted common features used in signal analysis, mostly based on Fourier coefficients and EEG rhythm waves band. Once we calculated all the features for all data, we converted the results into flat file named .ARFF file. This format is used by WEKA. WEKA is an environment for knowledge analysis. It is already came with a set of common classification techniques, clustering methods, and feature selections. We used WEKA to characterize signals from each experiment we have conducted.

The performance of the classifiers are measured in accuracy level. We compared 4 predefined methods in WEKA. The results showed that J48 Decision Tree achieved the highest accuracy level in average with 79.089 % (subject-dependent data) and 87.5133 % (subject-independent data). We also performed a comparison between classifying 3 user state and with pair-wise classification (only two states). On average, we achieved 78.7864 % for distinguishing three classes of states. For pair-wise (relaxed, bored, excited consecutively) are 95.9096 %, 93.3679 %, and 81.3905 %.

We also performed clustering to unlabeled data (from playing chess game). The goal is to distinguish thinking load of our brain. The results showed that on average per subject we achieved as high as 67.43104 % of accuracy level in classifying the recording based on clustering model made.

Last, we tried to distinguish the impact of different difficulty level on our brain activities. We achieved a rather poor performance due to minimum data. The results were as high as 82.65305 % per subject.

8.2 Contributions

In Chapter 1, we indicated four research questions that we tried to tackled during the course of this research.

1. *How to perform EEG measurement to a subject playing computer games?*

   We managed to setup EEG experiments with 20 subjects to play computer games. We generated of more than 300 minutes of brain activity recordings during subjects playing computer games. We have answered this research question in Chapter 5, in which we explain the steps to conduct three experiments using EEG devices including all the preparation and how to prevent the pitfall.

   Measurement on a computer game player differs from clinical or psychological experiment purposes. Preparation steps should be as fast as possible, otherwise the joy of playing computer games would be useless. If this occurred, we would not be able to have the respond we wanted to have. Furthermore, it is somewhat arduous to control or to stimulate player to have the state we wanted to have.

2. *How to process data gained using EEG devices?*

   Data we acquired are still raw. They need to be preprocessed and processed using several steps. It all depends on how we want to do with the data. In our research, we slice the data into segments. This is to extract features, which further in the process we want to classify which task is the player is doing.

   The quality of data we gained is in low quality. Eye blinks interfere with the signals. A lot of muscle activities also occurred, such as swallowing, laughing, and so on. We consider this as artifacts.

   We explained this in Chapter 5 and 6 on data acquisition and data processing. Specific sections on removing unwanted artifact spanned on several chapters.

3. *How to characterize brain signal based on playing computer games?*

   Brain signals should help us in determining what kind of brain activities are performed. The use of tools is crucial in analyzing brain signals. We used several tools from proprietary to open-source tools.

   We were analyze brain signal visually or after being processed through pattern-recognition methods (i.e. classification or clustering). To analyze visually we use EEGLAB. It is supplied with a widely used ICA algorithm to decompose the signal into independent components.

   A much more thorough analysis is being done in WEKA. We used WEKA environment to aid us in building a classifier to determine task of playing computer games. It is already supplied with the common techniques to do this.

   We showed the results of classification and clustering to characterize brain signals in Chapter 7. From the results, it is shown that some tasks achieve good performance of accuracy level, while some would need to have more specific features to distinguish.

4. *Is EEG recording a suitable method to conduct usability testing in computer games?*

   In our experiment, we have shown the use of EEG as a result from a recording of computer game player. This can be used as additional aid to the usability testing already available right now.

Although, it is not the scope of this thesis to have an online brain-activities based computer games, it is shown that it is achievable. From the results in Chapter 7, we achieve some good results of distinguishing this. When a player is supposed to be bored, excited, or relax can be seen in their brain activities read through EEG recordings.

8.3 Recommendation

There are several things that can be done in future work. Those that can not be implemented at this stage, shall become a next step in the future research.

Much more data should be collected to get a better and much more reliable inferences for the classification. Until now, we are limited with the difficulty of setting up measurement. It takes an amount of time to prepare the equipment.

A much more realistic scenario in driving simulation would be a consideration. This can be combined with more controlled ask to achieve stimulation that we want to have. For example how brain reacts to traffic light scenarios or how brain reacts to traffic jam. The implication of this can be used to analyze and lower the level of accident on highways.

A more thorough experiment should be done. A tightly-monitored experiment with specific amount of time using games. For example when playing chess against computer, since this is a somewhat calm type of game, much more controlled brain activities can be marked.

Much of the features we put into account are taken for granted and are derived from literature study. A deep understanding of biomedical background especially on neuroscience would greatly improve the importance of using which features. In other and recent development, besides using Fourier coefficients, wavelet are used.

One of the next step from this research is to make the whole procedure done in real time. During the game is played, we can provide feedbacks to the player.

The games themselves are not fully utilized. It should be made some form of feedback to user. Again, this would require a real-time processing of the brain activities.

We believe that brain still contain one of the mysteries that human being posses. Yet, in itself there are potential aspects which we can develop. For a computer game, user involvement by knowing his or her state can leverage the level of entertainment.

Bibliography

[Bell 95] A. J. Bell and T. J. Sejnowski. "An Information-Maximization Approach to Blind Separation and Blind Deconvolution". *Neural Computation*, Vol. 7, No. 6, pp. 1129–1159, 1995.

[Benb 06] S. R. Benbadis. "EEG Atlas: EEG Artifacts". Online: `http://www.emedicine.com/neuro/topic678.htm`, 2006.

[Berk 05] C. Berka, D. J. Levendowski, C. K. Ramsey, G. Davis, M. N. Lumicao, K. Stanney, L. Reeves, S. H. Regli, P. D. Tremoulet, and K. Stibler. "Evaluation of an EEG-workload model in the Aegis simulation environment". *Proceedings of the SPIE*, Vol. 5797, pp. 90–99, 2005.

[Bouc 07] P. Bouchner. *Driving Simulators for HMI Research.* PhD thesis, Czech Technical University, Faculty of Transportation Sciences, Institute of Control and Telematics, 2007.

[Card 97] J. F. Cardoso. "Blind separation of real signals with JADE". `http://www.tsi.enst.fr/~cardoso/RR_ECG/jadeR.m`, 1997.

[Chen 04] D. Chen and R. Vertegaal. "Using Mental Load for Managing Interruptions In Physiologically Attentive User Interfaces". In: *CHI '04 extended abstracts on Human factors in computing systems*, 2004.

[Chop 00] A. Choppin. *EEG-Based Human Interface for Disabled Individuals: Emotion Expression with Neural Networks.* Master's thesis, Department of Information Processing, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, 2000.

[Culp 99] J. Culpepper. "Discriminating Mental States Using EEG Represented by Power Spectral Density". Tech. Rep., Departement of Computer Science Harvey Mudd College, 1999.

[Delo 03] A. Delorme and S. Makeig. "EEG changes accompanying learned regulation of 12-Hz EEG activity". *IEEE Transactions on Rehabilitation Engineering*, Vol. 11, pp. 133–136, 2003.

[Duda 01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification, 2nd Edition.* Wiley, 2001.

[Gerl 06] V. Gerla, L. Lhotska, V. Krajca, and K. Paul. "Multichannel Analysis of the Newborn EEG Data". In: *International Special Topic Conference on Information Technology in Biomedicine*, 2006.

[Guyt 91] A. C. Guyton. *Textbook of Medical Physiology.* W. B. Saunders Company, Philadelphia PA, 1991.

[Henk 07] A. Henke. "ActionCube website". available online through: `http://action.cubers.net/index.html`, 2007.

[Hjor 70] B. Hjorth. "EEG analysis based on time domain properties". *Electroencephalography and Clinical Neurophysiology*, Vol. 29, pp. 306–310, 1970.

[Hona 05] M. Honal and T. Schultz. "Identifying User State Using Electroencephalographic Data". In: *International Workshop on Multimodal Multiparty Meeting Processing*, 2005.

[Ishi 03] K. Ishino and M. Hagiwara. "A Feeling Estimation System Using a Simple Electroencephalograph". In: *IEEE International Conference on Systems, Man and Cybernetics, 2003*, 2003.

[Keir 90] Z. Keirn and J. Aunon. "A new mode of communication between man and his surroundings". *IEEE Transaction on Biomedical Engineering*, Vol. 37, pp. 1209–1214, 1990.

[Kopr 96] I. Koprinska, G. Pfurtscheller, and D. Flotzinger. "Sleep classification in infants by decision tree-based neural networks". *Artificial Intelligence in Medicine*, Vol. 8, pp. 387–401, 1996.

[Krep 04] R. Krepki and B. Blankertz. "The Berlin Brain-Computer Interface (BBCI): towards a new communication channel for online control in gaming applications". *Journal of Multimedia Tools and Applications*, 2004.

[Lalo 05] E. Lalor, S. P. Kelly, C. Finucane, R. Burke, R. Smith, R. B. Reilly, and G. McDarby. "Steady-state VEP-based Brain Computer Interface Control in an Immersive 3-D Gaming Environment". Tech. Rep., Mindgame Group, Media Lab Europe, 2005.

[Lee 06] J. C. Lee and D. S. Tan. "Using a low-cost electroencephalograph for task classification in HCI research". In: *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pp. 81–90, ACM Press, New York, NY, USA, 2006.

[Lian 05] S.-F. Liang, C.-T. Lin, R.-C. Wu, T.-Y. Huang, and W.-H. Chao. "Classification of Driver's Cognitive Responses From EEG Analysis". In: *IEEE International Symposium on Circuits and Systems, 2005*, 2005.

[Maso 05] S. G. Mason, M. M. Jackson, and G. E. Birch. "A general framework for characterizing studies of brain interface technology". *Ann Biomed Eng*, Vol. 33, No. 11, pp. 1653–70, 2005.

[Mull 00] T. Muller, T. Ball, R. Kristeva-Feige, T. Mergner, and J. Timmer. "Selecting relevant electrode positions for classification tasks based on the electroencephalogram". *Med Biol Eng Comput*, Vol. 38, No. 1, pp. 62–7, 2000.

[Nijh 07] A. Nijholt and D. Tan. "Playing with Your Brain: Brain-Computer Interfaces and Games". In: *BrainPlay 2007: Playing with Your Brain - Workshop of the International Conference on Advances in Computer Entertainment Technology*, 2007.

[Onto 06] J. Onton and S. Makeig. "Information-based modeling of event-related brain dynamics". *Elsevier Progress in Brain Research*, 2006.

[Open 07] OpenEEG. “OpenEEG Project website”. available online through: `http://openeeg.sourceforge.net`, 2007.

[Palk 04] A. Palke. *Brainathlon: Enhanching Brainwave Control Through Brain-Controlled Game Play.* Master’s thesis, Department of Mathematics and Computer Science, Mills College, 2004.

[Rani 05] P. Rani, N. Sarkar, and C. Lin. “Maintaining Optimal Challenge in Computer Games through Real-Time Physiological Feedback”. In: *HCI International 2005*, 2005.

[Robe 00] S. J. Roberts and W. D. Penny. “Real-time Brain-Computer Interfacing: a preliminary study using Bayesian learning”. *Medical & Biological Engineering and Computing*, Vol. 38, pp. 56–61, 2000.

[Rome 03] S. Romero, M. Mailanas, S. G. S. Clos, and M. Barbanoj. “Reduction of EEG Artifacts by ICA in Different Sleep Stages”. In: *Proceedings of the 25” Annual International Conference of the IEEE EMBS*, 2003.

[Vour 00] M. Vourkas, G. Papadourakis, and S. Micheloyannis. “Use of ANN and Hjorth Parameters In Mental-Task Discrimination”. In: *First International Conference on Advances in Medical Signal and Information Processing, 2000*, pp. 327–332, 2000.

[Wess 06] M. Wessel. *Pioneering Research into Brain Computer Interfaces.* Master’s thesis, Technische Universiteit Delft, 2006.

[Witt 05] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann, 2005.

[Wolp 00] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan. “Brain-computer interface technology: a review of the first international meeting”. *IEEE Trans Rehabil Eng*, Vol. 8, No. 2, pp. 164–73, 2000.

[Wolp 94] J. R. Wolpaw and D. J. McFarland. “Multichannel EEG-based braincomputer communication”. *Electroenceph. Clin. Neurophysiol.*, Vol. 90, pp. 444 – 449, 1994.

Appendix A

Source Code

These listings below are mainly used for feature extraction purpose and for data conversion.

A.1 frequency_powers.m

```
function outData = frequency_powers( timeData )

len = length(timeData);

fSample = 128; fftWin = hamming(len);

fftData = fft(fftWin' .* timeData); fStep = (fSample) / len;

fLows = [0 4 8 13 30]; fHighs = [4 8 13 30 50];

outData = zeros(5,1); for band = 1:5

nLow = max(1,round(fLows(band) / fStep));

nHigh = min(len/2,round(fHighs(band) / fStep));

fftpower = zeros(nHigh,1);

fftPower = 0;

if (nLow < nHigh)
    for i = nLow:nHigh
        fftPower = fftPower + abs(fftData(i));
```

```
        fftpower(i)=fftPower;
    end
end

bandData = fftPower / len;

outData(band) = bandData;
```

A.2 read_files.m

```
function files = read_files( directory )

files = dir(directory); ix=[files.isdir]; files = files(~ix);
```

A.3 hjorth_activity.m

```
function outData = hjorth_activity( timeData )

outData = var(timeData);
```

A.4 hjorth_mobility.m

```
function outData = hjorth_mobility( timeData )

d1 = diff(timeData);

m0 = var(timeData); m2 = var(d1);

if (m0 == 0)
    outData = 0;
else
    outData = sqrt(m2/m0);
end
```

A.5 hjorth_complexity.m

```
function outData = hjorth_complexity( timeData )

d1 = diff(timeData); d2 = diff(d1);

m0 = var(timeData); m2 = var(d1); m4 = var(d2);
```

```
if ((m2 == 0) || (m0 == 0))
    outData = 0;
else
    outData = sqrt((m4/m2)/(m2/m0));
end
```

A.6 calculate_fft.m

```
function outData = calculate_fft( timeData )

len = length(timeData);

fftWin = hamming(len);

fftData = fft(fftWin' .* timeData);
```

A.7 write_arff.m

```
function write_arff( directory, state, experiment )

files = read_files( directory ); filename = files(1).name; dots =
strfind(filename,'.'); if (~isempty(dots))
    dots = dots(length(dots));
    if (strcmp(filename(dots:length(filename)),'.arff') == 0)
        filename = [filename '.arff'];
    end
else
    filename = [filename '.arff'];
end

relation = 'Dataset';

time = clock(); file = fopen(filename,'w');

fprintf(file,'%%ARFF File  \n');
fprintf(file,'%%Export time : \n');
fprintf(file,'%%    date (dd/mm/yy) : %d/%d/%d\n',time(3), time(2), time(1));
fprintf(file,'%%    time (hh:mm:ss) : %d:%d:%d\n',time(4), time(5), time(6));

fprintf(file,'@RELATION %s\n',relation);
fprintf(file,'\n\n%% Attributes\n');

fprintf(file,'@ATTRIBUTE MaximumPositiveAmplitude  NUMERIC\n');
fprintf(file,'@ATTRIBUTE MaximumNegativeAmplitude  NUMERIC\n');
fprintf(file,'@ATTRIBUTE Mean  NUMERIC\n'); fprintf(file,'@ATTRIBUTE
Variance  NUMERIC\n'); fprintf(file,'@ATTRIBUTE DFT_band_delta
NUMERIC\n'); fprintf(file,'@ATTRIBUTE DFT_band_theta NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha NUMERIC\n');
```

```
fprintf(file,'@ATTRIBUTE DFT_band_beta  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_gamma NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2theta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2alpha_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_delta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2alpha_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_theta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha2beta_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_alpha2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_band_beta2gamma_ratio  NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_mean NUMERIC\n');
fprintf(file,'@ATTRIBUTE DFT_variance NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Activity NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Mobility NUMERIC\n');
fprintf(file,'@ATTRIBUTE HJorth_Complexity NUMERIC\n');

if (strcmp(experiment,'experiment1') == 1)
    fprintf(file,'@ATTRIBUTE State {relax, bored, excited}\n');
else
    fprintf(file,'@ATTRIBUTE Difficulty {good, medium, worse}\n');
end

fprintf(file,'\n');
fprintf(file,'%%Begining of data\n');
fprintf(file,'@DATA\n'); fprintf(file,'\n');

str = classify_all( directory, files, state);

fprintf(file,'%s\n',str);

fclose(file);
```

A.8 classify_all.m

```
function strng = classify_all( directory, files, state )

strng = []; for i=1:length(files)
    str = [directory '\' files(i).name];

    disp(['Read file' str]);
    data = dlmread(str);
    timeData = data';
    %%write_files(files(i).name, data);

    %% MaximumPositiveAmplitude
    outData = max(timeData);

    strng = [strng num2str(outData) sprintf('\t')];

    %% MaximumNegativeAmplitude
```

```
outData = -max(-timeData);

strng = [strng num2str(outData) sprintf('\t')];

%% Mean
outData = mean(timeData);

strng = [strng num2str(outData) sprintf('\t')];

%% Variance
outData = var(timeData);

strng = [strng num2str(outData) sprintf('\t')];

%% DFT

outData = frequency_powers(timeData);

%% DFT band delta
delta = outData(1);

strng = [strng num2str(delta) sprintf('\t')];

%% DFT band theta
theta = outData(2);

strng = [strng num2str(theta) sprintf('\t')];

%% DFT band alpha
alpha = outData(3);

strng = [strng num2str(alpha) sprintf('\t')];

%% DFT band beta
beta = outData(4);

strng = [strng num2str(beta) sprintf('\t')];

%% DFT band gamma
gamma = outData(5);

strng = [strng num2str(gamma) sprintf('\t')];

%% DFT band delta2theta ratio
delta2theta = delta / theta;

strng = [strng num2str(delta2theta) sprintf('\t')];

%% DFT band delta2alpha ratio
delta2alpha = delta / alpha;

strng = [strng num2str(delta2alpha) sprintf('\t')];

%% DFT band delta2beta ratio
delta2beta = delta / beta;
```

```
strng = [strng num2str(delta2beta) sprintf('\t')];

%% DFT band delta2gamma ratio
delta2gamma = delta / gamma;

strng = [strng num2str(delta2gamma) sprintf('\t')];

%% DFT band theta2alpha ratio
theta2alpha = theta / alpha;

strng = [strng num2str(theta2alpha) sprintf('\t')];

%% DFT band theta2beta ratio
theta2beta = theta / beta;

strng = [strng num2str(theta2beta) sprintf('\t')];

%% DFT band theta2gamma ratio
theta2gamma = theta / gamma;

strng = [strng num2str(theta2gamma) sprintf('\t')];

%% DFT band alpha2beta ratio
alpha2beta = alpha / beta;

strng = [strng num2str(alpha2beta) sprintf('\t')];

%% DFT band alpha2gamma ratio
alpha2gamma = alpha / gamma;

strng = [strng num2str(alpha2gamma) sprintf('\t')];

%% DFT band beta2gamma ratio
beta2gamma = beta / gamma;

strng = [strng num2str(beta2gamma) sprintf('\t')];

%% DFT Mean
outData = calculate_fft(timeData);

x = mean(outData);

strng = [strng num2str(x) sprintf('\t')];

%% DFT Variance
outData = calculate_fft(timeData);

x = var(outData);

strng = [strng num2str(x) sprintf('\t')];

%% Hjorth Activity
x = hjorth_activity(timeData);
```

```
    strng = [strng num2str(x) sprintf('\t')];

    %% Hjorth Mobility
    x = hjorth_mobility(timeData);

    strng = [strng num2str(x) sprintf('\t')];

    %% Hjorth Complexity
    x = hjorth_complexity(timeData);

    strng = [strng num2str(x) sprintf('\t')];

    strng = [strng state];

    strng = [strng sprintf('\n')];
end
```

A.9 Separator.java

This last listing of source code is in Java. It was used for separating the data into segments. Initially the whole source code aforementioned was also in Java. Due to performance reason, we converted to MATLAB.

```
package com.tudelft.thesis;

import java.io.*;

public class Separator {

    public static void main(String[] args)  {
        if (args.length > 0)    {
            String contents = "";
            int i = 0;
            int index = 1;

            do  {
                contents =
                    Separator.getContents(args[0].toString(),
                        1+i*3840,3840+i*3840);

                Separator.setContents(args[1].toString() +
                    "-" + index + ".txt", contents);
                i++;
                index++;
            }   while(!contents.isEmpty());
        }
    }

    public static String getContents(String fileName,
        int startPosition, int stopPosition)  {
```

```
        StringBuffer contents = new StringBuffer();
        int i;

        File file = new File(fileName);
        try {
            RandomAccessFile raf = new RandomAccessFile(file, "r");

            String line = null;

            for (i = 1; i < startPosition; i++) {
                line = raf.readLine();
            }

            while((line = raf.readLine())!=null &&
                (i < stopPosition))  {
                contents.append(line);
                contents.append(System.getProperty("line.separator"));
                i++;
            }

        } catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        return contents.toString();
    }

    public static String getContents(String fileName)   {

        StringBuffer contents = new StringBuffer();

        BufferedReader input = null;

        try {
            input = new BufferedReader( new FileReader(fileName));

            String line = null;

            while((line = input.readLine())!=null)  {
                contents.append(line);
                contents.append(System.getProperty("line.separator"));
            }

        } catch(Exception e) {
            e.printStackTrace();
        } finally {
            try {
                input.close();
            }   catch(Exception e){
                e.printStackTrace();
```

```
            }
        }
        return contents.toString();
    }

    public static void setContents(String fileName,
        String contents){
        FileOutputStream output;
        PrintStream ps;

        try {
            output = new FileOutputStream(fileName);
            ps = new PrintStream(output);
            ps.print(contents);
        } catch(Exception e)    {
            e.printStackTrace();
        }
    }

    public static void setContents(String fileName,
        String contents, int noIteration){
        FileOutputStream output;
        PrintStream ps;

        try {
            output = new FileOutputStream(fileName);
            ps = new PrintStream(output);
            ps.print(contents);
        } catch(Exception e)    {
            e.printStackTrace();
        }
    }
}
```

Appendix B

XML File for Tracks Definition

This appendix explains the XML file which we used for defining the track for Experiment 1 - Playing with Racing Car Game. The following is the XML file for long track that is used in session 2.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE params SYSTEM"../../../src/libs/tgf/params.dtd" [ <!--
general definitions for tracks  --> <!ENTITY default-surfaces SYSTEM
"../../../data/tracks/surfaces.xml"> ]>

<params name="test" type="param" mode="mw">
  <section name="Surfaces">&default-surfaces;</section>
  <section name="Header">
    <attstr name="name" val="Long" />
    <attstr name="category" val="oval" />
    <attnum name="version" val="4" />
    <attstr name="author" val="Z.Dharmawan" />
    <attstr name="description" val="Long EEG recording" />
  </section>
  <section name="Graphic">
    <attstr name="3d description" val="Long.ac" />
    <section name="Terrain Generation">
      <attnum name="track step" unit="m" val="20" />
      <attnum name="border margin" unit="m" val="50" />
      <attnum name="border step" unit="m" val="30" />
      <attnum name="border height" unit="m" val="15" />
```

```
      <attstr name="orientation" val="clockwise" />
    </section>
  </section>
  <section name="Main Track">
    <attnum name="width" unit="m" val="10.0" />
    <attnum name="profil steps length" unit="m" val="4.0" />
    <attstr name="surface" val="asphalt2-lines" />
    <!--Left part of track-->
    <section name="Left Side">
      <attnum name="start width" unit="m" val="4.0" />
      <attnum name="end width" unit="m" val="4.0" />
      <attstr name="surface" val="grass" />
    </section>
    <section name="Left Border">
      <attnum name="width" unit="m" val="0.5" />
      <attnum name="height" unit="m" val="0.05" />
      <attstr name="surface" val="curb-5cm-r" />
      <attstr name="style" val="plan" />
    </section>
    <section name="Left Barrier">
      <attnum name="width" unit="m" val="0.1" />
      <attnum name="height" unit="m" val="1.0" />
      <attstr name="surface" val="barrier" />
      <attstr name="style" val="curb" />
    </section>
    <!--End of left part-->
    <!--Right part of track-->
    <section name="Right Side">
      <attnum name="start width" unit="m" val="4.0" />
      <attnum name="end width" unit="m" val="4.0" />
      <attstr name="surface" val="grass" />
    </section>
    <section name="Right Border">
      <attnum name="width" unit="m" val="0.5" />
      <attnum name="height" unit="m" val="0.05" />
      <attstr name="surface" val="curb-5cm-r" />
      <attstr name="style" val="plan" />
    </section>
    <section name="Right Barrier">
      <attnum name="width" unit="m" val="0.1" />
      <attnum name="height" unit="m" val="1.0" />
      <attstr name="surface" val="barrier" />
      <attstr name="style" val="curb" />
    </section>
    <!--End of right part-->
    <section name="Pits">
      <attstr name="side" val="right" />
      <attstr name="entry" val="pit entry" />
      <attstr name="start" val="pit start" />
      <attstr name="end" val="pit end" />
      <attstr name="exit" val="pit exit" />
      <attnum name="length" unit="m" val="10.0" />
      <attnum name="width" unit="m" val="4.0" />
    </section>
    <section name="Track Segments">
```

```
<!--********************************-->
<!--     Segment 1                -->
<!--********************************-->
<section name="1">
  <attstr name="type" val="str" />
  <attnum name="lg" unit="m" val="10000.0" />
  <attnum name="z start" unit="m" val="0.0" />
  <attnum name="z end" unit="m" val="0.0" />
  <attstr name="surface" val="asphalt2-lines" />
  <!--Left part of segment-->
  <section name="Left Side">
    <attnum name="start width" unit="m" val="4.0" />
    <attnum name="end width" unit="m" val="4.0" />
    <attstr name="surface" val="grass" />
  </section>
  <section name="Left Border">
    <attnum name="width" unit="m" val="0.5" />
    <attnum name="height" unit="m" val="0.05" />
    <attstr name="surface" val="curb-5cm-r" />
    <attstr name="style" val="plan" />
  </section>
  <section name="Left Barrier">
    <attnum name="width" unit="m" val="0.1" />
    <attnum name="height" unit="m" val="1.0" />
    <attstr name="surface" val="barrier" />
    <attstr name="style" val="curb" />
  </section>
  <!--End of left part-->
  <!--Right part of segment-->
  <section name="Right Side">
    <attnum name="start width" unit="m" val="4.0" />
    <attnum name="end width" unit="m" val="4.0" />
    <attstr name="surface" val="grass" />
  </section>
  <section name="Right Border">
    <attnum name="width" unit="m" val="0.5" />
    <attnum name="height" unit="m" val="0.05" />
    <attstr name="surface" val="curb-5cm-r" />
    <attstr name="style" val="plan" />
  </section>
  <section name="Right Barrier">
    <attnum name="width" unit="m" val="0.1" />
    <attnum name="height" unit="m" val="1.0" />
    <attstr name="surface" val="barrier" />
    <attstr name="style" val="curb" />
  </section>
  <!--End of right part-->
</section>
<!--********************************-->
<!--     Segment 2                -->
<!--********************************-->
<section name="2">
  <attstr name="type" val="rgt" />
  <attnum name="arc" unit="deg" val="180.0" />
  <attnum name="radius" unit="m" val="100.0" />
```

```
    <attnum name="z start" unit="m" val="0.0" />
    <attnum name="z end" unit="m" val="0.0" />
    <attstr name="surface" val="asphalt2-lines" />
    <!--Left part of segment-->
    <section name="Left Side">
      <attnum name="start width" unit="m" val="4.0" />
      <attnum name="end width" unit="m" val="4.0" />
      <attstr name="surface" val="grass" />
    </section>
    <section name="Left Border">
      <attnum name="width" unit="m" val="0.5" />
      <attnum name="height" unit="m" val="0.05" />
      <attstr name="surface" val="curb-5cm-r" />
      <attstr name="style" val="plan" />
    </section>
    <section name="Left Barrier">
      <attnum name="width" unit="m" val="0.1" />
      <attnum name="height" unit="m" val="1.0" />
      <attstr name="surface" val="barrier" />
      <attstr name="style" val="curb" />
    </section>
    <!--End of left part-->
    <!--Right part of segment-->
    <section name="Right Side">
      <attnum name="start width" unit="m" val="4.0" />
      <attnum name="end width" unit="m" val="4.0" />
      <attstr name="surface" val="grass" />
    </section>
    <section name="Right Border">
      <attnum name="width" unit="m" val="0.5" />
      <attnum name="height" unit="m" val="0.05" />
      <attstr name="surface" val="curb-5cm-r" />
      <attstr name="style" val="plan" />
    </section>
    <section name="Right Barrier">
      <attnum name="width" unit="m" val="0.1" />
      <attnum name="height" unit="m" val="1.0" />
      <attstr name="surface" val="barrier" />
      <attstr name="style" val="curb" />
    </section>
    <!--End of right part-->
  </section>
  <!--******************************-->
  <!--     Segment 3                -->
  <!--******************************-->
  <section name="3">
    <attstr name="type" val="str" />
    <attnum name="lg" unit="m" val="10000.0" />
    <attnum name="z start" unit="m" val="0.0" />
    <attnum name="z end" unit="m" val="0.0" />
    <attstr name="surface" val="asphalt2-lines" />
    <!--Left part of segment-->
    <section name="Left Side">
      <attnum name="start width" unit="m" val="4.0" />
      <attnum name="end width" unit="m" val="4.0" />
```

```
    <attstr name="surface" val="grass" />
  </section>
  <section name="Left Border">
    <attnum name="width" unit="m" val="0.5" />
    <attnum name="height" unit="m" val="0.05" />
    <attstr name="surface" val="curb-5cm-r" />
    <attstr name="style" val="plan" />
  </section>
  <section name="Left Barrier">
    <attnum name="width" unit="m" val="0.1" />
    <attnum name="height" unit="m" val="1.0" />
    <attstr name="surface" val="barrier" />
    <attstr name="style" val="curb" />
  </section>
  <!--End of left part-->
  <!--Right part of segment-->
  <section name="Right Side">
    <attnum name="start width" unit="m" val="4.0" />
    <attnum name="end width" unit="m" val="4.0" />
    <attstr name="surface" val="grass" />
  </section>
  <section name="Right Border">
    <attnum name="width" unit="m" val="0.5" />
    <attnum name="height" unit="m" val="0.05" />
    <attstr name="surface" val="curb-5cm-r" />
    <attstr name="style" val="plan" />
  </section>
  <section name="Right Barrier">
    <attnum name="width" unit="m" val="0.1" />
    <attnum name="height" unit="m" val="1.0" />
    <attstr name="surface" val="barrier" />
    <attstr name="style" val="curb" />
  </section>
  <!--End of right part-->
</section>
<!--*******************************-->
<!--     Segment 4                -->
<!--*******************************-->
<section name="4">
  <attstr name="type" val="rgt" />
  <attnum name="arc" unit="deg" val="180.0" />
  <attnum name="radius" unit="m" val="100.0" />
  <attnum name="z start" unit="m" val="0.0" />
  <attnum name="z end" unit="m" val="0.0" />
  <attstr name="surface" val="asphalt2-lines" />
  <!--Left part of segment-->
  <section name="Left Side">
    <attnum name="start width" unit="m" val="4.0" />
    <attnum name="end width" unit="m" val="4.0" />
    <attstr name="surface" val="grass" />
  </section>
  <section name="Left Border">
    <attnum name="width" unit="m" val="0.5" />
    <attnum name="height" unit="m" val="0.05" />
    <attstr name="surface" val="curb-5cm-r" />
```

```
            <attstr name="style" val="plan" />
          </section>
          <section name="Left Barrier">
            <attnum name="width" unit="m" val="0.1" />
            <attnum name="height" unit="m" val="1.0" />
            <attstr name="surface" val="barrier" />
            <attstr name="style" val="curb" />
          </section>
          <!--End of left part-->
          <!--Right part of segment-->
          <section name="Right Side">
            <attnum name="start width" unit="m" val="4.0" />
            <attnum name="end width" unit="m" val="4.0" />
            <attstr name="surface" val="grass" />
          </section>
          <section name="Right Border">
            <attnum name="width" unit="m" val="0.5" />
            <attnum name="height" unit="m" val="0.05" />
            <attstr name="surface" val="curb-5cm-r" />
            <attstr name="style" val="plan" />
          </section>
          <section name="Right Barrier">
            <attnum name="width" unit="m" val="0.1" />
            <attnum name="height" unit="m" val="1.0" />
            <attstr name="surface" val="barrier" />
            <attstr name="style" val="curb" />
          </section>
          <!--End of right part-->
        </section>
      </section>
    </section>
  </params>
```

The outcome of this XML file is a long straight track with length of 20 km plus additional length at the corner, see Figure B.1. Supposedly, the player will have developed a bored feeling toward the game.

The XML file for track that is used in session 3 will not be shown due to its length. It has the same structure as the aforementioned XML. Since there are more curves than the first track, it has more segments compared to the first XML file. These segments represent certain part of the track. The first has less segments, because the track consist almost the same condition for all course of the track. The outcome of this XML file can be seen in Figure B.2.

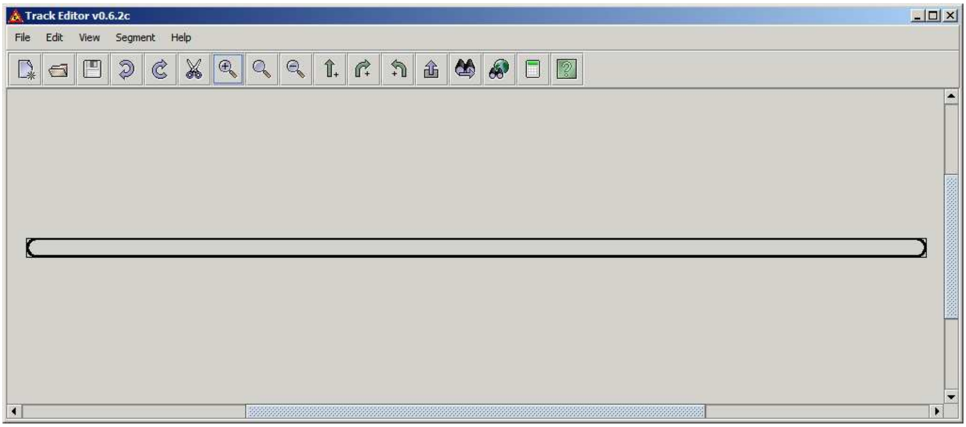


Figure B.1: Long Track for Session 2 of Experiment 1

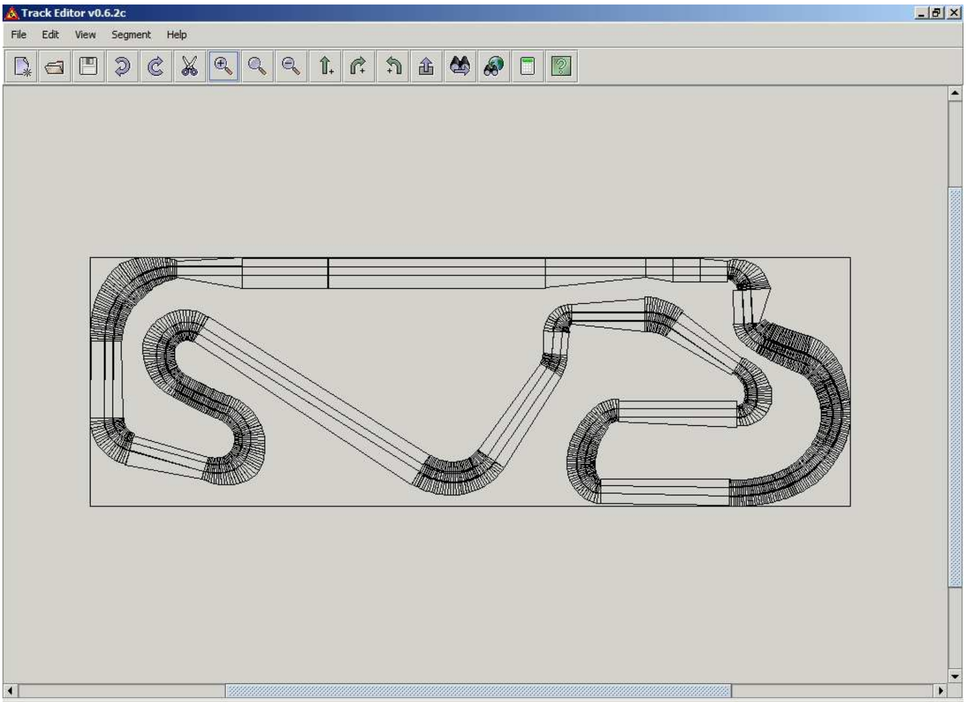


Figure B.2: More fast and exciting track for Session 3 of Experiment 1

Appendix C

Questionnaire

Below is questionnaire that we gave to the participants. The aim of this questionnaire is to know the demographic background of the subjects and to know their experience after conducted the experiment.

Questionnaire
EEG - Computer Games Session
Man-Machine Interaction Group - TU Delft

Name: ....
Age: ....
Gender: M/F

Date: ....
Type of Game: ....

For Racing Games

1. Can you drive? Y/N

2. Have you ever played a racing game before? Y/N

3. What do you think of the 1st Session?
   ....

4. What do you think of the 2nd Session?
   ....

5. What do you think of the 3rd Session?
   ....

   item Any difficulties during the game?
   ....

For Chess

1. Have you ever play a chess? Y/N

2. Was it difficult?
   ....

3. Why did you win?
   ....

4. Why did you loose?
   ....

For First-Person Shooters (FPS)

1. Have you ever play an FPS? Y/N

2. Was it difficult?
   ....

3. Do you see any changes of difficulties?
   ....

Appendix D

Attribute-Relation File Format

This appendix explains the .ARFF file that we use to classify the brain signal for Experiment 1, playing the racing computer game. ARFF file can be broken down into three parts, @$RELATION$, @$ATTRIBUTE$, and the @$DATA$ part. @$RELATION$ tells the title of the file, usually here we put features that is classified. @$ATTRIBUTE$ is the list of features. @$DATA$ is the value from to be computed features. At the last line of the @$ATTRIBUTE$, we define the class or the result to be.

```
% Data  :   C3, C4, Cz, F3, F4, F7, F8, Fp1, Fp2, Fz,
%           O1, O2, P3, P4, Pz, T3, T4, T5, T6
@RELATION "FFTTheWholeRecordings"

% Attributes
@ATTRIBUTE      FFT_band_delta  NUMERIC

@ATTRIBUTE      FFT_band_theta NUMERIC

@ATTRIBUTE      FFT_band_alpha  NUMERIC

@ATTRIBUTE      FFT_band_beta   NUMERIC

@ATTRIBUTE      FFT_band_gamma NUMERIC

@ATTRIBUTE      FFT_band_delta2theta_ratio  NUMERIC
```

```
@ATTRIBUTE      FFT_band_delta2alpha_ratio  NUMERIC

@ATTRIBUTE      FFT_band_delta2beta_ratio  NUMERIC

@ATTRIBUTE      FFT_band_delta2gamma_ratio  NUMERIC

@ATTRIBUTE      FFT_band_theta2alpha_ratio  NUMERIC

@ATTRIBUTE      FFT_band_theta2beta_ratio  NUMERIC

@ATTRIBUTE      FFT_band_theta2gamma_ratio  NUMERIC

@ATTRIBUTE      FFT_band_alpha2beta_ratio  NUMERIC

@ATTRIBUTE      FFT_band_alpha2gamma_ratio  NUMERIC

@ATTRIBUTE      FFT_band_beta2gamma_ratio  NUMERIC

@ATTRIBUTE      ASSIGNED_CLASSES {"Relax","Not Excited","Excited"}

@DATA

54.478  30.272  51.084  59.245  28.783  1.7996  1.0664  0.91953
1.8927  0.59259 0.51096 1.0517  0.86224 1.7748  2.0584  "Relax"

64.91   35.388  56.025  70.37   31.601  1.8342  1.1586  0.92241
2.0541  0.63165 0.50289 1.1199  0.79615 1.7729  2.2268  "Relax"

60.901  36.103  54.602  63.931  30.658  1.6869  1.1154  0.95261
1.9865  0.6612  0.56472 1.1776  0.85408 1.781   2.0853  "Relax"

64.946  32.301  43.969  55.796  28.324  2.0107  1.4771  1.164 2.293
0.73463 0.57891 1.1404  0.78802 1.5524  1.97    "Relax"

75.918 38.551  46.553  64.894  31.646  1.9693  1.6308  1.1699 2.399
0.82812 0.59407 1.2182  0.71737 1.4711  2.0506  "Relax"

66.667 27.542 29.638 42.625  24.586  2.4205  2.2493  1.564 2.7115
0.92927 0.64615 1.1202 0.69533 1.2055  1.7337  "Relax"

57.35 31.684 35.268 56.442 27.682 1.8101  1.6261  1.0161 2.0718
0.89837 0.56135 1.1446 0.62485 1.2741 2.039   "Relax"

56.915 31.061 32.034 48.142 25.36 1.8323  1.7767 1.1822 2.2443
0.96963 0.6452 1.2248 0.66541 1.2632 1.8984  "Relax"

53.565 29.705 32.632 47.936 23.83 1.8032 1.6415 1.1174 2.2478 0.9103
0.61968 1.2465 0.68074 1.3694 2.0116 "Relax"

55.656 30.592 35.01 49.146 24.903 1.8193 1.5897 1.1325 2.2349
0.87381 0.62248 1.2285 0.71237 1.4059 1.9735 "Relax"

74.405 46.778 74.48   96.493 65.427 1.5906 0.99899 0.77109 1.1372
0.62807 0.48478 0.71497 0.77186 1.1384 1.4748 "Relax"
```

```
60.444 33.593 80.624 94.935 63.007 1.7993  0.7497 0.63669 0.95932
0.41666 0.35385 0.53316 0.84926 1.2796  1.5067 "Relax"

71.414 40.64   81.838 80.014 36.813  1.7572 0.87262 0.89251 1.9399
0.49659 0.50791 1.104 1.0228 2.2231 2.1735 "Relax"

65.1 36.578 76.105  77.945  35.608 1.7797 0.8554 0.8352 1.8282
0.48063 0.46928 1.0272  0.97639 2.1373 2.1889 "Relax"

73.602 39.553 83.624  80.387 37.203  1.8609  0.88016 0.9156 1.9784
0.47298 0.49202 1.0631  1.0403 2.2478  2.1608 "Relax"

51.791 25.543 41.697 87.956  65.269  2.0276 1.2421  0.58882 0.79349
0.61257 0.2904 0.39134 0.47407 0.63885 1.3476  "Relax"

61.064 32.695 42.396 61.589 29.013  1.8677 1.4403 0.99148 2.1047
0.77118 0.53086 1.1269 0.68837 1.4613  2.1228 "Relax"

55.466 31.734 69.941 75.929 43.387 1.7478  0.79304 0.73049 1.2784
0.45373 0.41795 0.73143 0.92113 1.612 1.7501 "Relax"

72.256 33.219 63.051 72.561 34.721 2.1752  1.146 0.9958 2.081
0.52686 0.4578 0.95673 0.86893 1.8159  2.0898  "Relax"

209.88 67.063 68.48 143.74 99.635  3.1296 3.0649  1.4602  2.1065
0.9793 0.46657 0.67309 0.47643 0.68731 1.4426  "Not Excited"

242.95 85.052 80.704 164.35 108.43  2.8565 3.0104  1.4782  2.2407
1.0539 0.51749 0.78442 0.49103 0.74431 1.5158 "Not Excited"

231.01 81.514 73.817 151.59 101.59 2.8339  3.1294 1.5239  2.274
1.1043 0.53775 0.80241 0.48697 0.72664 1.4922  "Not Excited"

236.31 76.804 66.129 144.32 101.19 3.0768 3.5735 1.6375 2.3353
1.1614 0.5322 0.75901 0.45823 0.65351 1.4262 "Not Excited"

254.56 92.131 76.643 156.96 103.19 2.763 3.3213 1.6218  2.4669
1.2021 0.58696 0.89283 0.48829 0.74274 1.5211 "Not Excited"

213.64 52.013 45.968 154.71 153.82 4.1073 4.6474 1.3809 1.3888
1.1315 0.3362 0.33814 0.29713 0.29884 1.0058 "Not Excited"

258.47 90.989 71.99 159.67 123.07 2.8406 3.5903 1.6188 2.1002 1.2639
0.56986 0.73933 0.45087 0.58496 1.2974 "Not Excited"

297.04 106.02 64.401 130.09 86.658 2.8017 4.6124 2.2833 3.4277
1.6463 0.81498 1.2234 0.49505 0.74316 1.5012 "Not Excited"

304.53 114.46 68.245 128.17 85.269 2.6605 4.4623 2.376 3.5714 1.6772
0.89306 1.3424 0.53246 0.80035 1.5031 "Not Excited"

218.75 79.126 62.931 131.69 88.021 2.7646 3.4761 1.6612 2.4852
1.2573 0.60087 0.89895 0.47788 0.71496 1.4961 "Not Excited"

214.35 80.722 99.824 251.23 204.42 2.6554 2.1472 0.8532 1.0486
```

```
0.80864 0.32131 0.39488 0.39735 0.48833 1.229 "Not Excited"

212.32 76.194 94.641 237.48 189.42  2.7866 2.2434 0.89406 1.1209
0.80508 0.32084 0.40225 0.39853 0.49964 1.2537 "Not Excited"

224.83 85.803 88.745 175.07 116.08 2.6203  2.5334 1.2842 1.9368
0.96685 0.49012 0.73915 0.50692 0.76449 1.5081  "Not Excited"

223.1 78.731 87.025 175.73 116.78  2.8337 2.5636  1.2696 1.9104
0.90469 0.44802 0.67416 0.49522 0.74519 1.5048 "Not Excited"

231.99 88.91 90.949 177.4 115.72 2.6093  2.5508 1.3077 2.0048
0.97759 0.50117 0.76832 0.51266 0.78594 1.533   "Not Excited"

209.36 58.136 64.178 204.98 174.14 3.6012 3.2622 1.0214 1.2022
0.90586 0.28362 0.33384 0.31309 0.36854 1.1771 "Not Excited"

246.09 76.09 74.49 165.8 126.91  3.2342 3.3037 1.4842 1.9392 1.0215
0.45892 0.59959 0.44927 0.58697 1.3065 "Not Excited"

210.05 75.201 86.595 233.09 204.9   2.7932  2.4257  0.90115 1.0252
0.86842 0.32262 0.36702 0.3715  0.42262 1.1376  "Not Excited"

234.01 71.856 83.493 175.75 123.32  3.2567  2.8028  1.3315  1.8977
0.86062 0.40884 0.5827 0.47505 0.67707 1.4252  "Not Excited"

204.23 45.528 50.908 109.8 79.062  4.4857  4.0117  1.8599  2.5831
0.89433 0.41463 0.57586 0.46362 0.6439  1.3889  "Excited"

214.07 55.229 59.46 124.62 89.716 3.8761  3.6002  1.7177  2.3861
0.92883 0.44316 0.6156 0.47712 0.66276 1.3891  "Excited"

212.73 53.035 54.224 115.91 78.835 4.0111 3.9231  1.8353  2.6984
0.97807 0.45757 0.67274 0.46783 0.68782 1.4702  "Excited"

219.89 56.52 49.269 110.58 77.873 3.8905 4.463 1.9886  2.8237 1.1472
0.51114 0.72579 0.44556 0.63268 1.42 "Excited"

214.7 56.892 53.635 114.44 78.904 3.7738 4.0029  1.8761 2.721 1.0607
0.49714 0.72102 0.46868 0.67975 1.4503 "Excited"

201.7 35.334 33.091 108.28 93.988 5.7082 6.0952 1.8628 2.146 1.0678
0.32633 0.37595 0.30561 0.35208 1.152 "Excited"

206.63 52.749 53.798 127.09 99.17 3.9173 3.841 1.626 2.0836 0.98051
0.41507 0.53191 0.42332 0.54248 1.2815 "Excited"

214.99 51.725  46.054 99.558 67.808  4.1564  4.6683 2.1595  3.1706
1.1232 0.51955 0.76282 0.46258 0.67917 1.4682 "Excited"

209.69 51.046 45.79 98.205 66.021 4.108   4.5795  2.1353 3.1762
1.1148 0.51978 0.77317 0.46627 0.69356 1.4875  "Excited"

199.68 52.876 47.963 102.57 69.896 3.7764 4.1633  1.9468  2.8568
1.1024  0.51551 0.7565 0.4676  0.6862 1.4675 "Excited"
```

```
208.57 62.568  87.24 227.15 187.98 3.3335  2.3907 0.9182 1.1095
0.7172 0.27545 0.33284 0.38407 0.46409 1.2084 "Excited"

206.09 54.526 77.19 200.73 165.58  3.7797 2.6699 1.0267 1.2446
0.70638 0.27164 0.3293 0.38455 0.46617 1.2122 "Excited"

210.52 53.302 59.097 128.69 92.858  3.9495  3.5623 1.6358 2.2671
0.90195 0.41418 0.57402 0.4592  0.63642 1.3859 "Excited"

210.95 51.614 62.894 136.55 98.341  4.0871  3.3541 1.5449 2.1451
0.82065 0.378 0.52484 0.46061 0.63955 1.3885 "Excited"

223.77 60.136 62.971 133.42 93.387  3.721 3.5535  1.6771 2.3961
0.95498 0.45073 0.64395 0.47197 0.6743 1.4287  "Excited"

201.85 38.555 41.803 141.76 123.49 5.2352 4.8285  1.4238  1.6345
0.9223 0.27197 0.31222 0.29488 0.33852 1.148 "Excited"

212.62 51.544 57.982 137.14 111.47 4.125 3.6671 1.5504  1.9075
0.88898 0.37585 0.46241 0.42279 0.52016 1.2303 "Excited"

203.84 51.447 64.44 218.11 196.71 3.9622 3.1633 0.93458 1.0363
0.79836 0.23587 0.26154 0.29544 0.3276  1.1088 "Excited"

209.35 51.455 64.246 150.95 112.72 4.0686 3.2586  1.3868 1.8572
0.80092 0.34087 0.45649 0.4256 0.56995 1.3392 "Excited"
```