Brain Machine Interface

Student Name: Md. Ochiuddin Miah
Student Id: 011131145
Student Name: Md. Mahfuzur Rahman
Student Id: 011131107
Student Name: Md. Rashed Khan Menon
Student Id: 011131153

A thesis in the Department of Computer Science and Engineering presented in partial fulfillment of the requirements for the Degree of

Bachelor of Science in Computer Science and Engineering

United International University

Dhaka, Bangladesh

February, 2018.

Declaration

We, Md. Ochiuddin Miah, Md. Mahfuzur Rahman, Md. Rashed Khan Menon, certify that this thesis titled “Brain Machine Interface” and the work presented in it are ours own. We assure that we have done this undertaking based on our personal examination and research under the direction of Dr. Khondaker Abdullah-Al-Mamun, Director, AIMS Lab, and Associate Professor, Department of Computer Science and Engineering, United International University, Bangladesh. We have done this thesis mostly at the time of candidature for a BSc degree at United International University. We declare that we have acknowledged all the main sources that we have utilized in the preparation of our thesis work. We have also admitted all the sources as well as substances that we have utilized in our work, whether they are books, articles, journals as well as any other type of document. We also ascertain that we have not submitted this thesis formerly to any other university.

_____

Md. Ochiuddin Miah
011131145
Dept. of Computer Science & Engineering

_____

Md. Mahfuzur Rahman
011131107
Dept. of Computer Science & Engineering

_____

Md. Rashed Khan Menon
Student Id: 011131153
Dept. of Computer Science & Engineering

In my capacity as supervisor of candidate’s thesis, I certify that the above statements are true to the best of my knowledge.

_____

Dr. Khondaker Abdullah-Al-Mamun
Director, Advanced Intelligent Multidisciplinary Systems (AIMS) Lab
Associate Professor, Department of Computer Science and Engineering
United International University
Dhaka -1209, Bangladesh.

Certificate

I do hereby declare that the research works embodied in this thesis/project entitled "**Brain Machine Interface**" is the outcome of an original work carried out by Md. Ochiuddin Miah, Md. Mahfuzur Rahman, Md. Rashed Khan Menon under my supervision.

I further certify that the dissertation meets the requirements and the standard for the degree of BSc in Computer Science and Engineering.

_____

Dr. Khondaker Abdullah-Al-Mamun
Director, Advanced Intelligent Multidisciplinary Systems (AIMS) Lab
Associate Professor, Department of Computer Science and Engineering
United International University
Dhaka -1209, Bangladesh.

Abstract

In last few decades fields like- Intelligence Systems for biosignals processing and modeling has developed extensively and this advancement has opened up many new windows of opportunities. Brain Machine Interface (BMI) is one of such opportunities. It is a technology which connects brain and machines directly in order to command and control the machines. This technology is about to revolutionize the health and rehabilitation industry. BMI system acquires signals generated from the brain as input. It then processes this signals to understand the user thought, intension, consequently which is used to generate command to operate machine. In this thesis, we explored the BMI technology and aimed to develop a system that can able to distinguish different human thoughts and use it as actions or commands for playing computer games.

At the first part of this thesis, we obtained brain signals and extracted features from this signals. This features were analyzed to find out the informative patterns. We collected training and test data for designing two class classifier as well as for three class classifier. Two-class classifier classifies from right-hand movement and steady states. Whereas, three-class classifier classifies from right-hand movement, left-hand movement and steady states. At the second part of the thesis, we have worked with different classification methods including - OneR, Naïve Bayesian, C 4.5, and CART. For two-class classifier CART gave best performance, 91.3353% accuracy. But, C 4.5 did give very close performance to it, 90.8917% accuracy. For three-class classifier C 4.5 gave best performance, 65.6612% accuracy. At the last part of the thesis, we have considered overall performance and used C4.5 model to make the targeted application system. We made a virtual-ball movement controlling game, in which we can control the direction of movement of a virtual-ball using brain signals of voluntary movements.

We used Emotiv SDK and Java language to develop a program to record the brain signals. We have used R-programming and weka tool for data visualization and model construction. We made an application system that can be used for rehabilitation as well as improvement of user well-being. User can exercise his mind to recover from attention deficiency. User can also increase his attention span via playing the game. It can be also used for gaming and entertainment purposes.

Acknowledgement

At first, we thank our thesis supervisor Dr. Khondaker Abdullah-Al-Mamun, Director, AIMS Lab, and Associate Professor, Department of Computer Science and Engineering, United International University, Bangladesh. We are very grateful to our honorable supervisor. We express our deep gratitude to him. Artificial Intelligence as well as acute interest of our prudent supervisor in the field of Brain Machine Interface encouraged us to carry out our thesis. We also thank AIMS Lab, United International University for its supports and facilities. We express our deepest respect to our beloved United International University for making our graduation life amusing, productive and rich. We show our wholehearted appreciation to Department of Computer Science and Engineering for making us to think more skeptically and sincerely. Our department not only aided us with our graduation program but also guided us to be diligent, heedful, earnest and veracious human beings.

Moreover, we would love to exhibit our ingenuous approbation to the various faculty members, who so lovingly mentored and counseled us on different stages of our graduation program. Furthermore, thanks to our classmates for their support, friendship and merriment. Also, thanks to the various administration and staffs for their assiduous and painstaking services. At last, we find us thanking our precious parents and for their supports, guidance and unconditional love. Love of our family gave us encouragement and was the most pressing force that has pushed us to complete the arduous journey of graduation program and propelled us to achieve good results.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
ARM	Auto Regressive Method
BMI	Brain Machine Interface
BCI	Brain Computer Interface
CMS	Common Mode Sense
D	Training Data
DNI	Direct Neural Interface
DM	Data Mining
DT	A Decision Tree
DRL	Driven Right Leg
EEG	Electroencephalogram
EM	Eigenvector Methods
EMG	Electromyogram
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FIRDA	Frontal Intermittent Rhythmic Delta
FN	False Negatives
FP	False Positives
GUI	Graphics User Interface
HMI	Human Machine Interface
ICA	Independent Component Analysis
ID3	Iterative Dichotomiser 3
IIR	Infinite Impulse Response

OIRDA	Occipital Intermittent Rhythmic Delta
OS	Operating System
SMA	Supplementary Motor Area
TFD	Time Frequency Distributions
TN	True Negatives
TP	True Positives
WHO	World Health Organization
WT	Wavelet Transform
X	A Subset of Instances

Chapter 1: Introduction

1.1: Background

Bioengineering is an emerging field of science and technology of the 21st century. It bases its application on the knowledge of biology, physics, chemistry, as well as mathematics and computer science. It integrates concepts and methods all over from engineering, biological sciences as well as clinical medicine. It aims at solving practical issues of the life sciences.

Disability is one of the most severe problems in human life today. It disarranges human day to day life. Different discoveries from past bioengineering researches have made some notable impact to improve health and enhance living. Such discoveries include battery powered cardiac pacemaker, neural prosthetic device of cochlear implant or bionic ear, etc. The natural cardiac pacemaker can be replaced by battery powered cardiac pacemaker in order to help patients who are suffering from dysfunctionality of heart to improve their heart's electrical activity. People who are suffering from deafness can recover their hearing sensation via the assistance of bionic ear [1].

A huge range of bioengineering research is being conducted to enhance the knowledge of the field to get outstanding results. Facebook is developing brain-typing and skin-hearing technology using brain-computer interfaces. In near future, augmented reality as well as virtual reality could be controlled and experienced via human mind due to the advancement in brain-computer interfaces.

1.2: Motivation

A huge and diverse amount of research is going on to develop tools and technology to assist people with physical impairments and deficiencies, and also to enhance human functionalities. Conventionally, people have been trying different approaches to help disable people to do things but that seems not good enough. Brain machine interface is part of bioengineering technology that aims at developing products and services that medicate and assist disabled people to do tasks that they cannot do otherwise. It is a research field which works on automating the interfacing system between human and machine. And, this change in interfacing technology is going to make huge change in fields like gaming, education, medication, and communication, etc [2]. The basic idea is to get input information directly from human mind through some sensors and use that information to run machines to get things done. Lately, there has been lots of paper that studied to come up with novel gadgets. The goal of these undertakings has been the improvement of the life standard of impaired people by clinical and physical intervention. We have studied some of those papers and got inspired to work on this field.

1.3: Machine Learning

Machine learning is one of the most important parts of artificial intelligence. It is closely related to data mining and statistics. It is focused on writing software that can learn from past experience. It is practiced to set computers in a mode where they can learn on their own. Thus, machine learning can make explicit programming redundant for the learning purpose of a computer. Computer programs that have utilized Machine Learning tools are capable of learning and growing all by themselves while presented with novel data. Thus, machine learning enables computer program to be changed as well as developed only by themselves, without the necessity of involvement from any outside agent.

Machine learning has provided us with many advance products and services. Self-driving cars, practical speech recognition, as well as effective web search, etc. are some examples of the sophisticated feat that can be performed by machine learning. It has also progressed the understanding of the human genome vastly. It has also wide-ranging applications in field such as: - computational finance, image processing, computer vision, energy production, medical diagnosis, automotive and manufacturing.

1.4: Big Data

Developing applications using brain machine interface in most of the time requires manipulation of 'Big Data'. Big data is not only about volume but also about complexity. Volume is a very important property of big data, but the big data properties also incorporate data variety as well as velocity. Hence, the three properties of big data are volume, velocity and variety. Together these three properties are called three V's of big data.

Three V's of big data: -

1) Volume,
2) Velocity,
3) Variety

The tracking of data or information from the real world is referred to as volume. Velocity is the parameter that indicates the fastness of the data availability for analysis. Data can be both structured and unstructured. Some examples of the data variation could be log files, audio data, text data, video data, transaction level data, etc. Variety expresses the variation of the data.

The dataset we used was collect from a real time dataset. It had a large volume and there were different types of data types like we already mentioned. So, we can say that our dataset was a big data.

1.5: Data Mining

Data Mining is one of the most popular tools used nowadays for crucial decision making. It utilizes large databases to find out intelligible information that has been previously not known. The owner of the data sets the resulted information into action; which leads to huge advantages for him. Data mining makes use of knowledge and methods of various fields like- statistics, data analysis, and machine learning, etc. in order to manipulate large data sets. Data mining tools are now being increasingly used for future prediction. It allows its practitioner to make knowledge-driven resolutions. Data mining tools are very time-efficient. The problems that consumes tremendous amount of time while using conventional methods and tools, can be solved very quickly by it.

Anything that can be processed by a computer can be called data. It can be numbers, images, audio, text, etc. Until now, the common ways in which data mining is used involves- sales forecasting, database marketing, basket analysis, and merchandise planning, etc. Popular data mining methods involves: - classification, clustering, neural network, association, estimation, visualization, etc.

1.6: Human Brain

The human nervous system has the human brain as its central organ. The central nervous system is made of the brain as well as the spinal cord. It could be called the master of the body, because most of the body activities are governed by it. It is responsible for wide range of tasks that are vital for the human body. It receives from the sense organs. It does the processing of input information provided by the sense organs. It also integrates and coordinates the information that it receives. It does all these tasks so that it can generate decisions as well as send instructions to the rest of the body.

Different part of the brain is accountable for different tasks. The part that generates movement and also is in charge of controlling the movements is called motor system [3]. The movements generated by the motor system needs to be passed from the brain to motor neurons in the body in order to control and command the action of muscles. The passing is done by the nerves. The corticospinal tract transfers movements from the brain to the torso and limbs. It does so via the spinal cord. [4].The movements related to the eyes, mouth and face are carried by the cranial nerve.

The movement of arms and legs is generated in the motor cortex. The motor cortex consists of three parts. One of them is primary motor cortex, which is in the frontal lobe of the brain. The primary motor cortex is one of the principal brain areas involved in motor function. The role of the primary motor cortex is to generate neural impulses that control the execution of movement.

1.7: Thesis Contribution

The overall research in this thesis are the collecting data from brain signals, feature extraction, feature selection, classification of EEG signals, develop a system that can distinguish different human thoughts and take different actions based on these identifications.

Collecting data from brain signals is difficult, we used Emotiv SDK and Java technology to develop a program to record the brain signals. We collected training and test data for two class classifier as well as for three class classifier.

We have applied several classification models on our two class and three class dataset to find out the best classification model who gives maximum accuracy.

We made an application system that can be used for rehabilitation. User can exercise his mind to recover from attention deficiency and increase his attention span via playing the game using brain signals of voluntary movements.

1.8: Outline of the Thesis

Chapter 1 introduces the background, motivation and focus of this thesis.

Chapter 2 gives a review of Human Machine Interface, Brain Machine Interface and description of neuro-headset device available in the market.

Chapter 3give overview of EEG signals, steps and technique of processing EEG signals.

Chapter 4 describes the information and structure of our collected dataset. Here, we've also discussed about data visualization and showed some graphical representation of the data.

Chapter 5 provides a brief introduction of classification models and describes different types of classification algorithms that have been used in machine learning to mine information. Especially the chapter focuses on those classification models which we have applied in our research and can be useful in the related field.

Chapter 6 aims to describe the experimental results that we've found by using our dataset. The results that we've found demonstrated through some tables to understand the result comparison.

Chapter 7gives knowledge of emotiv epoc+ headset configuration, recording of brain signals using emotiv epoc+ and emotive SDKs, and how to developing a brain controlled game.

Chapter 8 gives conclusion, limitation and projection of our future works.

Chapter 2: Review of Brain Machine Interface

2.1: Human Machine Interface

Human Machine Interface (HMI) is a communication technology that makes it possible to connect human with machines without the necessity of any conventional input devices. It is a thrilling and promising research discipline of bioengineering which is going to reshape and transform our way of interacting with machines. This is also going to improve human healthcare system and human functionalities.

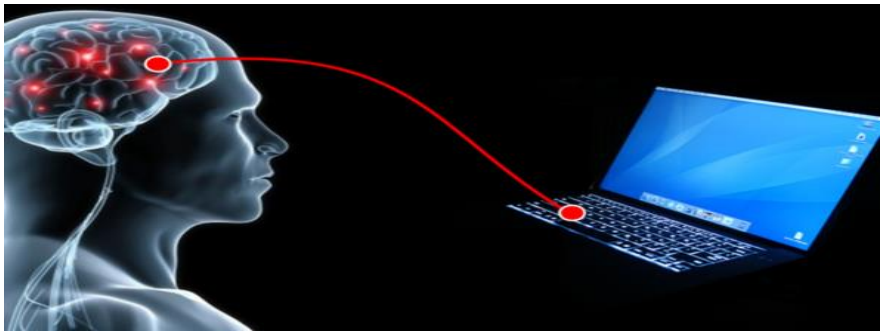


Figure 1: An abstract model of Human Machine Interface.

2.2: Importance of Human Machine Interface

According to World Health Organization (WHO) and World Bank Group, in our current world, there are about 200 million people suffering from disability of various kinds. This disability leads people to go through considerable challenges in functioning. And, in the coming years, disability situation is going to exacerbate. It is because the number of ageing population is on the rise. In addition with that, the risk of disability in ageing people is also increasing.

There are various causes of these disabilities. Some people get born with it. And, again some people get it as a result of an accident or aging. No matter how the disability is caused, it maims the person and makes him dependent on other people for care and wellbeing. But, if HMI technology is used the person can take care of himself on his own. He can do the entire necessary task by himself via the machine. Thought controlled wheelchair, thought controlled robotic arm, thought controlled robotic leg, etc. are some examples of this kind of technology.

Figure 2: An example of Human Machine Interface- a robotic leg.

HMI based robotic leg is used as a compensation of the original leg. This robotic leg is used autonomously like the original leg using HMI. HMI uses biological signal generated by the human body to accomplish it.

People may also lose their functionalities without losing any body parts. For example, severe depression causes people immense loss of their functionalities. People may lose sleep, attention, composure, memory, performance and peace of mind, etc. As a result of scathing depression; which is an ever increasing phenomena among people in industrialized and urbanized society. There are BMI devices which help and train people to meditate and tackle stress; and thereby to get rid of depression.

2.3: HMI Technology

Human machine interface can connect between human and machines two ways: direct way and mediate way. While connecting direct way, user intentions such as: - gesture, speech, vision or physiological and neurophysiological signals are used as input. This mode of connection is used to generate commands and services which compensate for physical impairment of the disabled people so that they can do their day to day tasks by themselves. This kind of HMI is called assistive HMI. And, it is focused on restoring lost functionalities of disabled people [5].

Figure 3: An overview of assistive HMI technology [1].

Person connects with machines via HMI technology. A disabled person can connect with a particular machine based on the nature of his disability and perform desired tasks. Here, user intentions such as: - gesture, speech, vision or physiological and neurophysiologic signals are used as input.

HMI technology can also make connection between human and machine using virtual environments, graphical user interfaces, and collaborative software agents. This is the mediated way of communicating.

2.4: Brain Machine Interface

Brain Machine Interface (BMI) is a type of HMI that establishes direct communication between human or animal brain and machines [6]. This technology is also known as brain computer interface (BCI) or direct neural interface (DNI). BMI technology use brain generated complex neurophysiologic activity as input signal, which is later interpreted into control commands to perform target tasks and actions. This field of research is going to revolutionize the medical treatment of disabled individuals who cannot otherwise physically communicate with their environment and perform essential day to day chores.

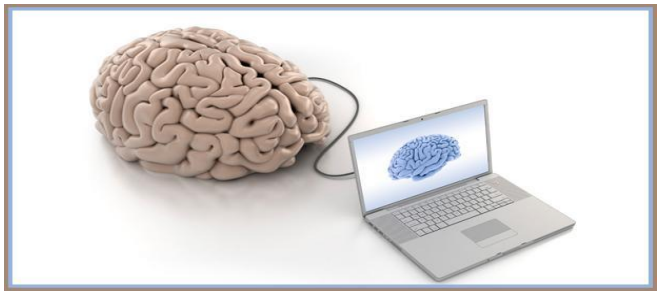


Figure 4: An abstract model of BMI

2.5: Type of BMI

There have been three types of BMI [7]:

1) Invasive BMI
2) Non Invasive BMI
3) Partially Invasive BMI

While using invasive BMI technology wires are placed inside the grey matter of the brain. This technology measures from a single or a few neurons. While using non-invasive BMI technology electrodes are placed on the surface of the scalp so that activities could be measured from a huge group of neurons. In partially invasive BMI technology wires are placed inside the brain but above the grey matter of it.

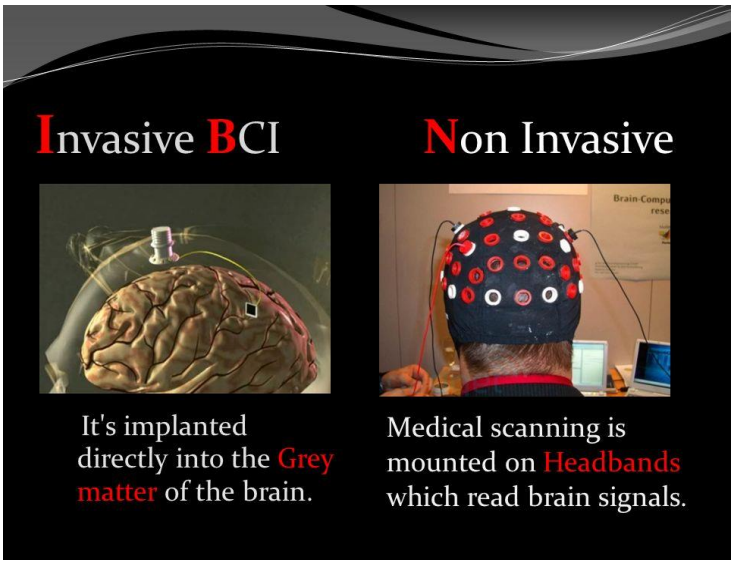


Figure 5: A picture showing both invasive and non-invasive BMI.

Invasive technology implements wire directly into the Grey matter of the brain. Noninvasive BMI technology put a set of electrodes on the surface of the scalp.
Invasive BMI technology generates better input signal. But, this technology is sophisticated and sensitive which requires highly specialized people to manipulate with. On the contrary, noninvasive BMI technology does not generate input signal as good as invasive BMI technology, but this technology is less sophisticated hence easy to manipulate with.

2.6: BMI Technology

When we use conventional interface to connect and communicate with machines, we use input device like keyboard, mouse, pen, etc. BMI system does not require this kind of input devices, as it directly connects machines with thoughts. To read the thoughts the neurosignals are monitored via sensors, and then these neurosignals are fed to

machine learning algorithm. Biosignals patterns are identified through decoding by the algorithm to generate desired commands and services [8].

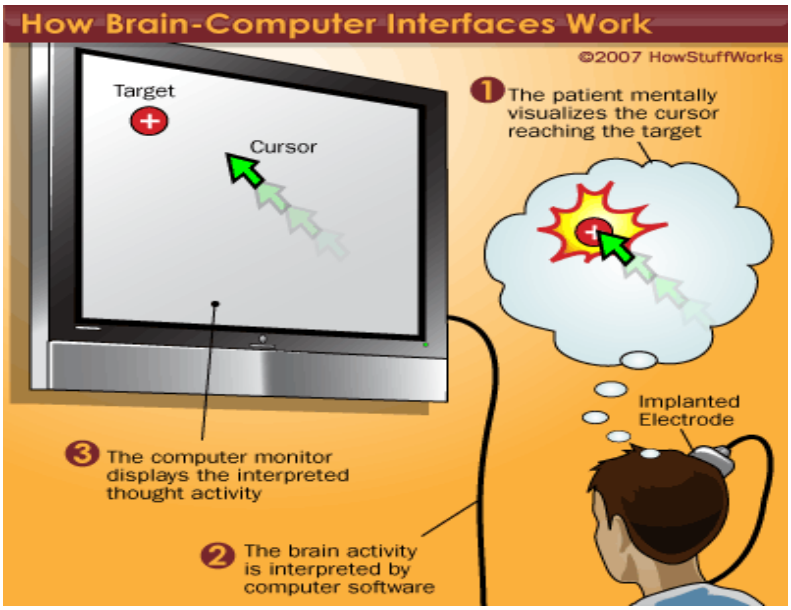


Figure 6: A user using BMI

A user, who is incapable of using a mouse, uses cursor via BMI. User thinks of moving the cursor, it generates brain signals, implanted electrode read the signals, and then signals are fed to the machine learning algorithm in the computer, the algorithm decodes and interpret the signals and generates desired actions. Thus, user thoughts are executed directly using brain machine interface.

2.7: BMI Application

Applications using brain machine interface are being developed in wide range of fields, such as: - medical applications, education, self-regulation, gaming, entertainment, etc.

2.7.1: Thought Controlled Robot

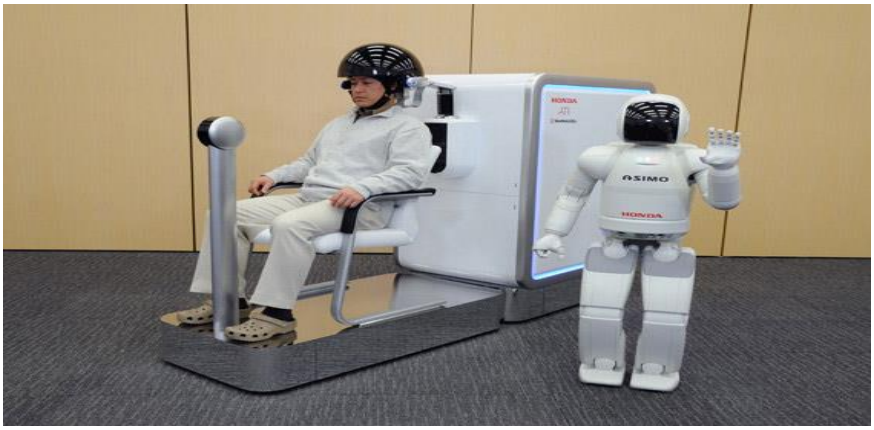


Figure 7: A thought controlled robot

A thought controlled robot is a robotic agent which can be controlled and commended by the thought of a human. Individual can control the movement of the robot his thought. He can also make the robot to pick, move, bring or sort objects.

2.7.2: Thought Controlled Wheelchair

Figure 8: A thought controlled wheelchair

A thought controlled wheelchair is a wheelchair which can be moved by the thought of user. This is a revolutionary technology for the people who cannot walk and move.

2.7.3: Neuroprosthetics

Figure 9: A robotic hand

A robotic hand can be controlled via user thought. People who has lost hand or who does not have a functioning hand can use this and does day to day chores like picking, holding, moving, placing and sorting objects without help from anybody else.

2.7.4: Gaming

Figure 10: BMI Gaming

BMI technology is going to revolutionize the gaming industry. Here, in this picture, two people are playing a game using noninvasive BMI. A player can control his gaming agent in the gaming environment using only his thought without any need of keyboard, mouse or joystick. This is going to make gaming more fun and lively.

2.7.5: Sleeping and Dreaming

Figure 11: A user wearing an Aurora dream headband

Aurora dream headband is a BMI technology which is developed to enable sleeping, to enhance and track the quality of sleep and to control dreams.

2.8: BMI Devices

There are so many brain machine interface device in the market, which have been developed to perform so many different tasks and to fulfill so many different purposes. Here are some of them: -

2.8.1: Star Wars Science - Force Trainer Game

Star Wars science - force trainer, is a gaming device which is very fun to play. User wears the headset; the headset has multiple electrodes placed onto it which collects signals from the brain as input. This input is used to control a physical ball in the tube. The user can move the ball up or down via his thinking. It was developed by Uncle Milton Industries.

Figure 12: Star Wars Science - Force Trainer Game

Table 1: Star Wars Science - Force Trainer Game Description.

Name	Star Wars Science - Force Trainer Toy
Maker	Uncle Milton Industries
Features	• Bluetooth headset that reads and interprets your brainwaves • Holograms featuring different Jedi Challenges from the Star Wars galaxy • 10 levels of Jedi Training • Music, sound effects, and personal instruction from Master Yoda • The Force™ Trainer II App with Episode 7 Update • Science learning poster included
Required	Works with most popular tablet devices including iPad (iPad 2 - 2011 and newer) and Android devices (Samsung Galaxy Tablet 10.1 recommended).

2.8.2: Aurora Dream Headband

Aurora Dream Headband is a neuroheadset device which is made by iWinks; iWinks makes products based on neuroscience and machine learning. The company is founded by Daniel Schoonover and Andrew Smile, they made it mostly to study and manipulate sleep. Aurora Dream Headband was first released on July, 2015. The device uses both bio-sensors and motion sensors. It is a wireless device. It can be used for sleep tracking purposes; it can identify different sleeping stages; it can keep track of the duration of each stages; it can also record the data. Aurora Dream Headband can be also used for sleep enhancement; it is a sleep enabler device. It can enhance and manipulate lucid dreaming. It also supports alarm clock to wake user up. It is compatible with various mobile phone applications and also supports internet of things.

Aurora Dream Headband technology uses AI to learn about user's sleeping pattern. It stores user's sleep statistics. Sleep research is always going on at iwinks. An individual can learn about his sleep health by submitting his sleep stats to iWinks.

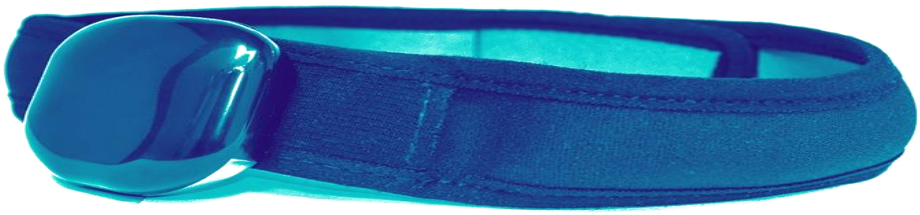


Figure 13: Aurora Dream Headband

Table 2: Description of Aurora Dream Headband.

Name	Aurora Dream Headband
Maker	iwinks
Release Date	July 2015
Sensor	Bio Sensor: • Brainwaves (EEG) • Eye Movements (EOG) • Muscle Tension (EMG) • Heart Rate (ECG) Motion Sensor: • 3-Axis Accelerometer • 3-Axis Gyroscope • Orientation Detection
Power / Connectivity	• USB rechargeable/updateable • Connects with modern smart devices • Built-in protection circuit
Free SDK	• Bio Sensor Data • Motion Sensor Data • Real-time sleep stages • Ambient Temperature • LED control
Uses	• Consumer Sleep Device

2.8.3: Muse Headband

Muse Headband is made by InteraXon. It was first released on April, 2014. It uses 7 sensors; it is a Bluetooth device. User can use Muse Headset as personal medication assistant. Our mind is sometime calm and again active sometimes. Muse can lead our mind into calm states. While meditating, muse can evaluate if our mind is active or calm; based on this evaluation Muse creates feedback. When user's mind is at calm state, user hears peaceful weather sounds. When user's mind is at wandering state, the weather will intensify, guiding him back to a calm state.

Figure 14: Muse Headband

Table 3: Description of Muse Headband.

Name	MUSE
Maker	InteraXon
Release Date	April 2014
Sensor	• 7 sensors; 5 front (2 active, 2 DRL, 1 reference), 2 active behind ears • Bluetooth
Uses	• reduce stress and improve focus
Applications	1. control your iPhone or Android device with your mind power 2. MuseIO, MuseLab, MusePlayer, LibMuse

2.8.4: MindWave Headset

MindWave Headset is produced by NeuroSky. It was first made available to the public on November 22, 2012. MindWave offers EEG technology for home use. MindWave is a very lightweight device; it is a wireless headset technology which uses safe and passive biosensors. To detect states of attention and meditation, MindWave Headset is very suitable. It can safely measure brainwave data; it can also transfer and communicate brainwave data to computer, android and ios devices to see user's brainwaves change in real time. Using this technology, user can even monitor his level of attention and relaxation in real time.

Figure 15: MindWave Headset

Table 4: Description of MindWave Headset.

Name	MindWave Headset
Maker	NeuroSky
Release Date	November 22, 2012
Measures	• Raw-Brainwaves • Processing and output of EEG power spectrums • Processing and output of NeuroSky proprietary eSense meter for Attention, Meditation, and other future meters • EEG/ECG signal quality analysis
Physical	• Weight: 90g • Sensor arm up: height: 225mm x width: 155mm x depth: 92mm • Sensor Arm down: height: 225mm x width: 155mm x depth: 165mmHeadset Diagram
Bluetooth	Version 3.0
Others	• BT Minimum Voltage: 1.0V • BT Power Consumption: 80mA • Low Battery Indicator 1.1V
applications	1. The Arduino Prosthesis Using the NeuroSky MindWave 2. Orbit Helicopter 3. Neuro Turntable Mobile 4. NeuroSky MindWave Controlled NXT - 2

2.8.5: Emotiv Insight 5 Channel Mobile EEG

Emotiv Insight 5-channel is a popular mobile EEG headset. It is made by Emotiv Systems; it was first released on July 21, 2015. It can record user brainwaves; it can also extract

useful knowledge utilizing these brainwaves. It is mostly used in research fields. It can be used to make BCI applications. Emotiv Insight capitalizes on advanced electronics. It gives very transparent and robust signals. Emotiv Insight device has 5 EEG sensors as well as 2 reference sensors. It is capable of gathering thorough information of user brain activity using its high spatial resolution. It is a very lightweight device; it also designed to be user-friendly.

Figure 16: Emotiv Insight 5 Channel Mobile EEG

Table 5: Emotiv Insight 5 Channel Mobile EEG Device Description.

Name	EMOTIV Insight 5 Channel Mobile EEG
Maker	EMOTIV Systems
Release date	Jul 21, 2015
Signals	• 5 channels: AF3, AF4, T7, T8, Pz • 2 references: In the CMS/DRL noise cancellation configuration
Signal resolution	• Data transmission rate: 128 samples per second per channel • Minimum voltage resolution: 0.51µV least significant bit
Frequency response	• 1-43Hz
Connectivity	• Wireless: Bluetooth 4.0 LE (May require EMOTIV Universal USB Receiver for certain devices) • Proprietary wireless: 2.4GHz band
Power	• Battery: Internal Lithium Polymer battery 480mAh • Battery life: 4 hours minimum run time
Applications	1. Mind Workstation 2. EMOTIV EmoBot 3. Cortex Arcade

2.8.6: Emotiv Epoc + 14 channel mobile EEG

The Emotiv EPOC+ 14 channel is a research grade neuroheadset. It has high resolution. It can do Neuro-signal acquisition and processing. It is a wireless device. It is built by Emotiv Systems. It was first released on 21 December, 2009. A set of sensors is used by it to tune into electric signals generated from the user brain. These signals are utilized to identify user's mental processes like- thoughts, feelings, expressions, etc. This device connects wirelessly to most PCs. It has 14 saline sensors which in turns offer optimal positioning and accuracy. It is very good for research purposes.

Figure 17: Emotiv Epoc + 14 channel

Table 6: Emotiv Epoc + 14 channel Description.

Name	EMOTIV EPOC+ 14
Maker	EMOTIV Systems
Release Date	21 December 2009
Signals	• 14 channels: AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4 • 2 references: In the CMS/DRL noise cancellation configuration P3/P4 locations
Signal resolution	• Sampling method: Sequential sampling. Single ADC • Sampling rate: 128 SPS or 256 SPS* (2048 Hz internal) • Resolution: 14 bits 1 LSB = 0.51µV • Bandwidth: 0.2 – 43Hz,digital notch filters at 50Hz and 60Hz • Filtering: Built in digital 5th order Sinc filter • Dynamic range (input referred): 8400µV(pp)
Connectivity	• Wireless: Bluetooth® Smart • Proprietary wireless: 2.4GHz band
Power	• Battery: Internal Lithium Polymer battery 480mAh
Applications	1. Brain Driver 2. Brain Controlled Wheelchair 3. Emotiv Epoc Brain Activity Map

2.9: Conclusion

BMI allows a user to connect with machines directly, so that he can communicate with the outside world and get things done controlling the machines. It makes the use of conventional input devices redundant. It uses signal generated from the brain activity as input. It then feeds these signals to machine learning algorithm; which generates command to be done by the machines. BMI could be very vital for rehabilitation purposes. There are already several uses of BMI for rehabilitation purposes, such as- thought controlled wheelchair, thought controlled robot, bionic arm, etc. Aurora, MindWave, Muse, Emotiv Epoc, Emotiv Insight, etc. are some of the neuroheadset devices used in, while working with BMI technology. Among these Emotiv is the most suitable for research grade BMI activities.

Chapter 3: Processing of EEG Signals

3.1: EEG Signals Processing Steps

There are several steps from taking row signals to the meaningful command. First we record brain signals, process is called Electroencephalography is an electrophysiological observing method to record the brain electrical activity. Brain is complex organ composed of numerous glial cells and neurons which transport information using electrical and chemical signals. Brain interface technology using the electrical brain signals for analyzing and making experimental applications. Electrode for taking brain signals is the most important part of recording neural movements or in stimulating neural cells. It is mainly non-invasive, where electrodes placed along the scalp. Invasive electrodes are also sometimes used in experiment of neural signals. EEG measures voltage variations as the result of ionic current within the brain's neurons. In, chapter 7 we will discuss details about our brain signal recording. After recording, we processing the signals, in this step we de-noise the signals, remove artifacts and simplify signals. Then we extract features. A feature represents a characteristic property, an accredited measurement, and a functional component acquired from a division of a pattern. Extracted features are should be like that purpose to minimize the loss of important information attached in the signal. In pattern recognition, a feature is a characteristic or an individual measurable property of a phenomenon being performed. It is necessary to keep the complexity minimum of implementation, to information processing cost minimum, and compress the information easily which need less potential to compress. Most recently, a various number of methods have been used widely to extract the features from EEG signals, such as Fast Fourier Transform (FFT, Eigenvector Methods (EM), Time Frequency Distributions (TFD), Wavelet Transform (WT) and Auto Regressive Method (ARM) and so on. After extracting features, we minimize features and select those features which are best suited for our experiment. Then by using different kinds of classification algorithms like dichotomiser 3, gini index, one R, bayesian, naive bayesian and so on, we classify the signal and get meaningful command. In chapter 5, we will discuss about classification methods we used to classify brain signals and in chapter 6, we will discuss about our result. In figure 18 showing the steps of signal processing.

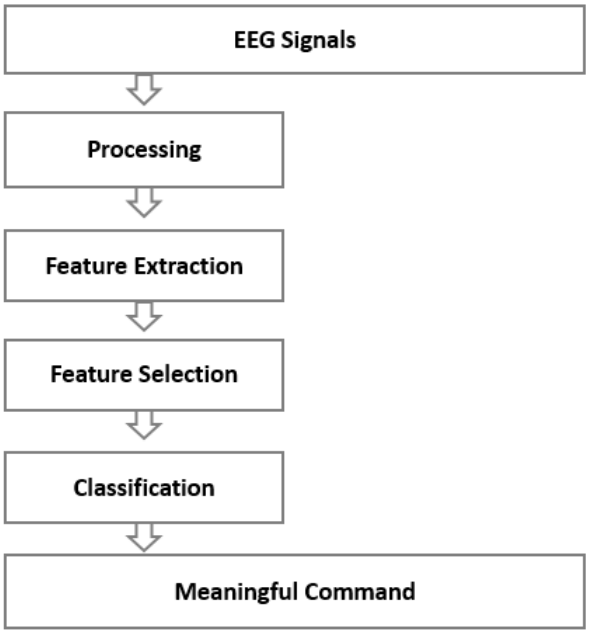


Figure 18: EEG Signal Processing Steps

3.2: Recording of EEG Signals

In the EEG recording, recorded waveforms reflect the cortical electrical activity [11][17]. EEG activity is measured in microvolts (μV). The main frequencies of the human EEG waves are:

- **Delta**: has a frequency of0.1-4 Hz. It has the highest in amplitude and the slowest waves. It occurs include deep sleep, pathologies and comatose state.
- **Theta**: has a frequency of 4 - 8 Hz. It occurs include sleeping, abnormal in awake adults.
- **Alpha**: has a frequency between 8 - 12 Hz. It appears when closing the eyes, thinking, calculating and awake but relaxed.
- **Mu**: has a frequency between 8-12 Hz. It is sensorimotor cortex activity.
- **Beta**: has a frequency of 12- 30 Hz. It occurs include brain processes, anxiety and arousal.
- **Gamma:** has a frequency of 30 – 90 Hz. It occurs include high mental activity, burst of physical activity and anxiety.

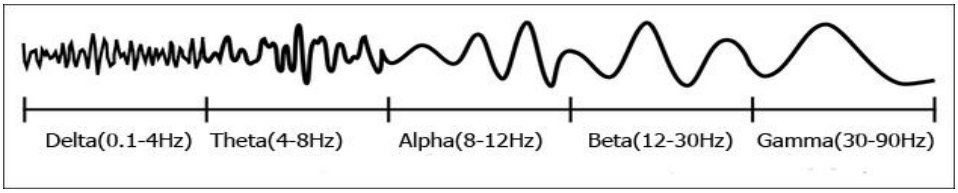


Figure 19: Types of Brain Signals

3.3: Artifacts

In signal processing and natural science, an artifact is any error in the observation or representation of any information [16].We can classify artifacts in two major class- Biological Artifacts and Environmental Artifacts.

- **Biological Artifacts:** Electrical signals detected from scalp by an EEG but which produce from non-cerebral origin are called biological artifacts. Some of biological artifacts include- Eye Induced Artifacts, ECG (cardiac) Artifacts, EMG (muscle activation) Induced Artifacts, Gloss Kinetic Artifacts[21] [23].
- **Environmental Artifacts:** Electrical signals detected from scalp by an EEG but which produce from outside the body by movement of patients or just settling of the electrodes may cause *electrode pops*, spikes is called environmental artifacts [21] [23].

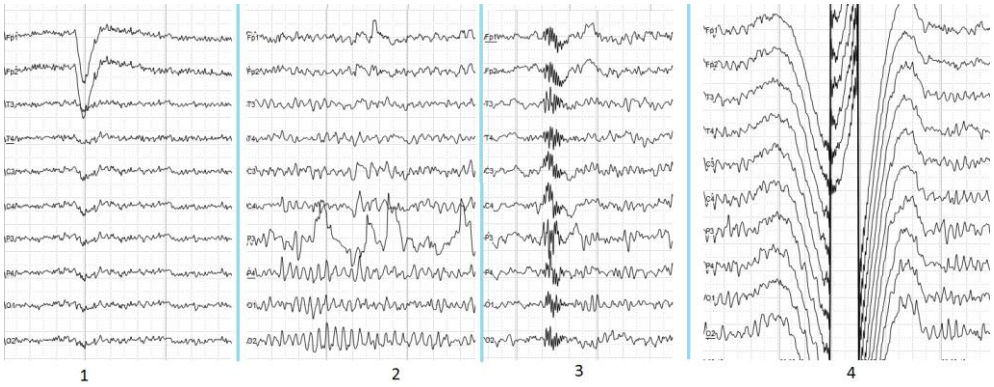


Figure 20: Brain Signals Artifacts

3.4: Artifact Correction

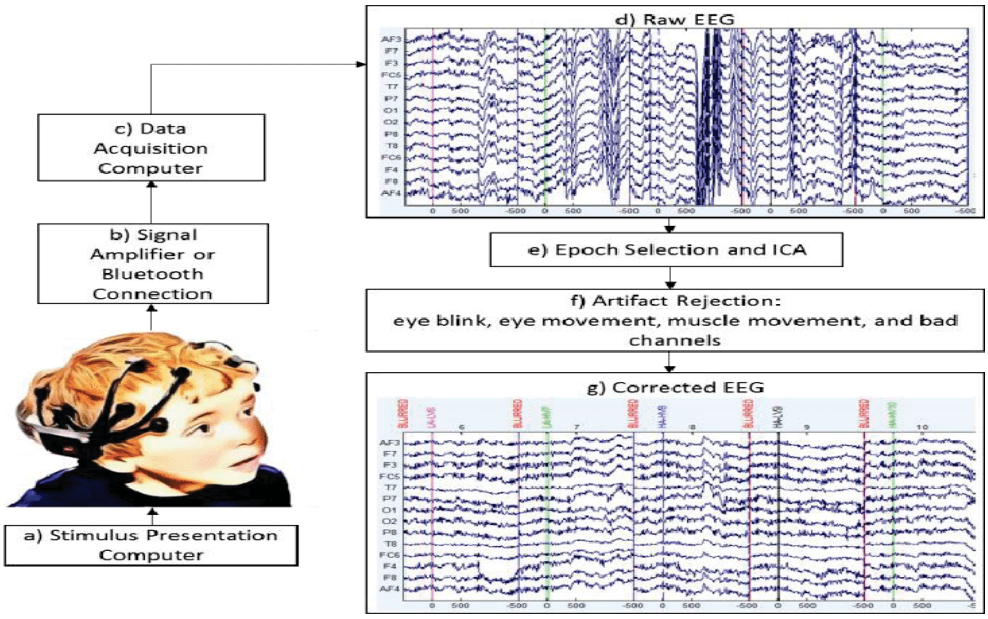


Figure 21: Correction of Artifacts [9]

Recently, ICA (independent component analysis) is the techniques have been used to correct or remove EEG artifacts. There are a large number of source separation algorithms, often assuming various behaviors or natures of EEG. The principle behind of using any particular method usually allow remixing only those components that would result in clean EEG by setting null weight of unwanted components However, Surface Laplacian has found as an effective in eliminating muscle artefact, especially for central electrodes. The combination of automated techniques and Surface Laplacian for removing muscle artifacts using ICA found more effective in the neural signal study [9][18][19][20].

3.5: Filtering

In signal processing, a filter is a process which removes some unwanted features or components from a signal. Filtering is a step of signal processing which complete or partial suppression some aspect of the signal [20][21][22][23][24][25]. This means removing some frequencies or frequency bands from signals. Correlations may be removed for specific frequency components. There are many filtering methods and these overlap in many different ways. Among these signal filtering methods include: Linear or Non-Linear, Time-Variant or Time-Invariant, Causal or not-Causal, Digital or Analog, Continuous Time or Discrete Time, Passive Type or Active Type (type of continuous-time filter), Finite Impulse Response (FIR) or Infinite Impulse Response (IIR) (type of discrete-time)

The frequency response can be classified into a number of different number of band forms:

- Low-pass filter (only low frequencies are passed)
- High-pass filter (only high frequencies are passed
- Band-pass filter (only frequencies are passed)
- Band-stop filter or Band-reject filter (only frequencies attenuated)
- Notch filter (rejects just one specific frequency)
- Comb filter (giving the band form the appearance of multiple regularly spaced narrow pass)
- All-pass filter (all frequencies are passed but the phase modified)

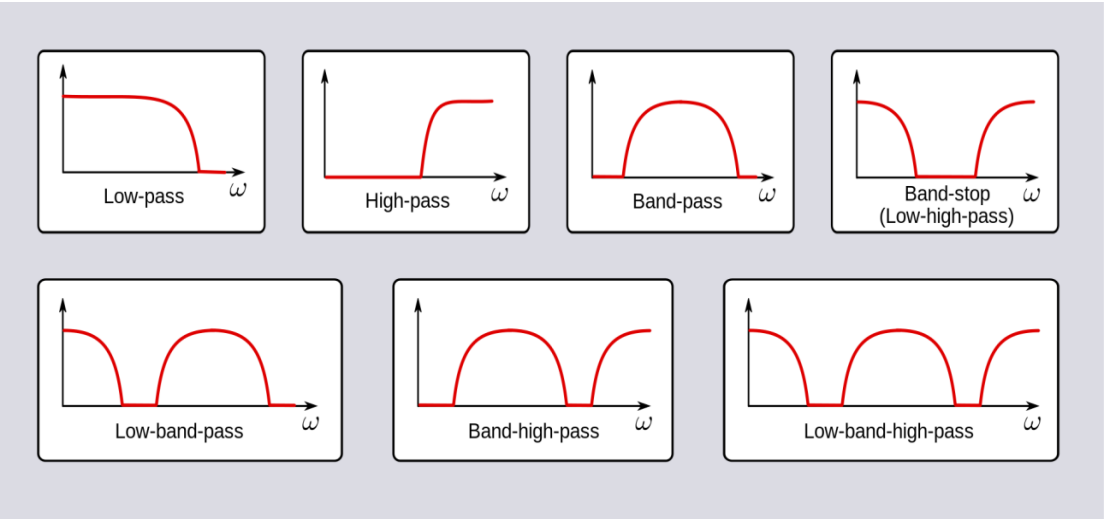


Figure 22: Filtering Brain Signals

3.6: Conclusion

Electroencephalography is an electrophysiological observing method to record the brain electrical activity. Brain is complex organ composed of numerous glial cells and neurons which transport information using electrical and chemical signals. Brain interface technology using the electrical brain signals for analyzing and making experimental applications. EEG signals have noises and artifacts. In order to reduce the noise of brain signals, brain signals experiments gather data from hundreds of trials and collected from a few individuals. Electrical signals detected along the scalp by an EEG, but which produce from non-cerebral origin or outside of body are called artifacts. In addition, artifacts mainly generated by the body, but many artifacts originate from outside the body. Some of these artifacts can be useful in various applications. So, we were very conscious about any necessary information was losing or not. After recording our brain signals for our experiment we de-noise the brain signals using various de-noising method and remove unnecessary artifacts to simply brain signals. Use different signal filtering method to make brain signal more simplify which will make feature extraction from brain signal easier. A filter is a process which removes some unwanted features or components from a signal. After that decrease features and select those features which are best suited for our experiment to remove multi-dimensional complexity of classification and get best result. Classification is the problem of identifying on which set a new observation belongs. In a training set of data containing observations (or instances) whose category subjects is known. By using different classification method we classify the brain signals and get result of brain commands.

Chapter 4: Data Set

4.1: Hand Movement Data

The Dataset that we used was collected from our brain signals. The dataset was collected using emotiv epoc+ EEG headset. We collect dataset for two classes and for three classes. Our two classes are 'righthand' and 'steady'. Our three classes are 'lefthand', 'righthand' and 'steady'. We take brain signals using emotiv epoc+ headset when we are steady, when we move our left hands and when we move our right hands.

For three classes dataset we take two trials for training data and one trial for test data. For trial-1, we take data for 30 seconds steady, 30 seconds righthand and move our right hand 23 times within 30 seconds, 30 seconds lefthand and move our left hand 25 times within 30 seconds. For trial-2, we also take data for 30 seconds steady, move our right hand 25 times within 30 seconds and move our left hand 24 times within 30 seconds. We also take test dataset for 15 seconds steady, move our right hand 14 times within 15 seconds and move our left hand 13 times within 15 seconds.

Table 7: Hand Movement data info for three classes.

Name	Default Task	Attribute Type	Training Data Points	Test Data Points	Attributes	Class Values
Hand movement	Classification	Real	473676	124492	5	Lefthand, Righthand, Steady

For two classes dataset we take two trials for training data and one trial for test data. For trial-1, we take data for 30 seconds steady, 30 seconds righthand and move our right hand 23 times within 30 seconds. For trial-2, we also take data for 30 seconds steady, move our right hand 25 times within 30 seconds. We also take test dataset for 15 seconds steady and move our right hand 14 times within 15 seconds.

Table 8: Hand Movement data info for two classes

Name	Default Task	Attribute Type	Training Data Points	Test Data Points	Attributes	Class Values
Hand movement	Classification	Real	320488	52304	5	Righthand, Steady

The table below describes the features of our hand movement dataset:

Table 9: Feature Description of Hand Movement Dataset.

No.	Name	Type	Description
1	Theta	Numeric	4-8 Hz frequency range
2	Alpha	Numeric	8-12 Hz frequency range
3	Low_beta	Numeric	12-18 Hz frequency range
4	High_beta	Numeric	18-30 Hz frequency range
5	Gamma	Numeric	30-90 Hz frequency range

4.2: Data Visualization

Presenting Data in a pictorial or graphical format is known as data visualization. We have followed several methods to visualize our data. We have plotted histogram, density graphs for our attributes or features. We have plotted bar plot, pie charts for class attribute or feature [26].

Histogram: Histograms of the three class data's features are shown below [26]:

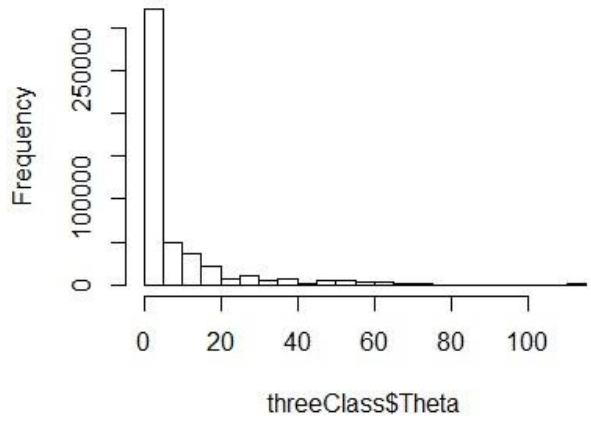


Figure 24:Histogram of Theta. Vertical line represents frequency and horizontal line represents Theta values.

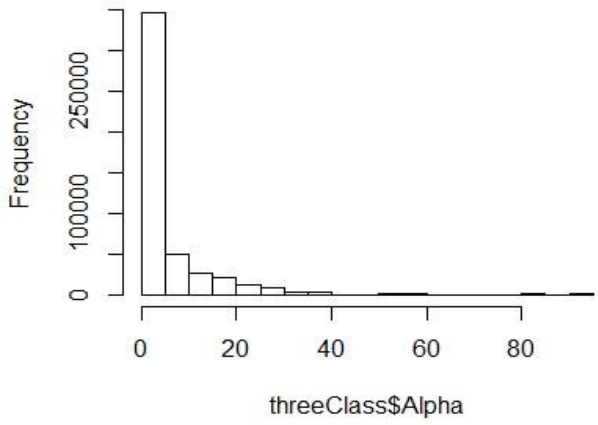


Figure 23: Histogram of Alpha. Vertical line represents frequency and horizontal line represents Alpha values.

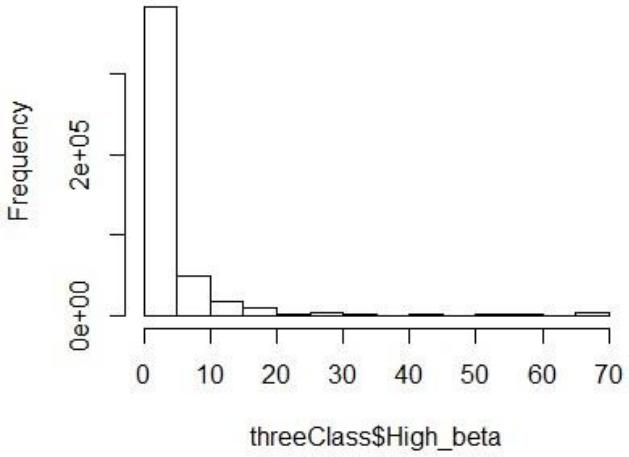


Figure 26: Histogram of High_beta. Vertical line represents frequency and horizontal line represents High_beta values.

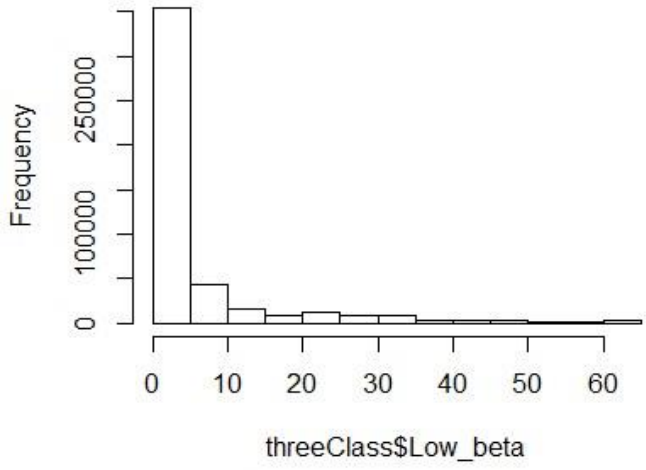


Figure 25: Histogram of Low_beta. Vertical line represents frequency and horizontal line represents Low_beta values.

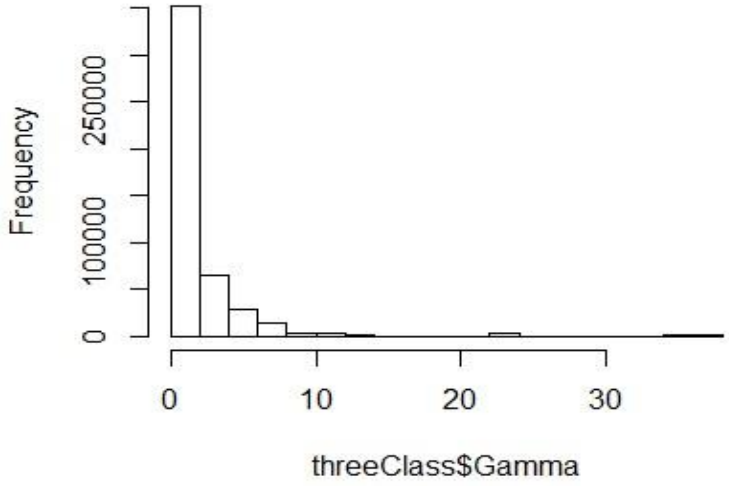


Figure 27: Histogram of Gamma. Vertical line represents frequency and horizontal line represents Gamma values.

Density Graph:

Density graphs for the three class data's features are shown below [26]:

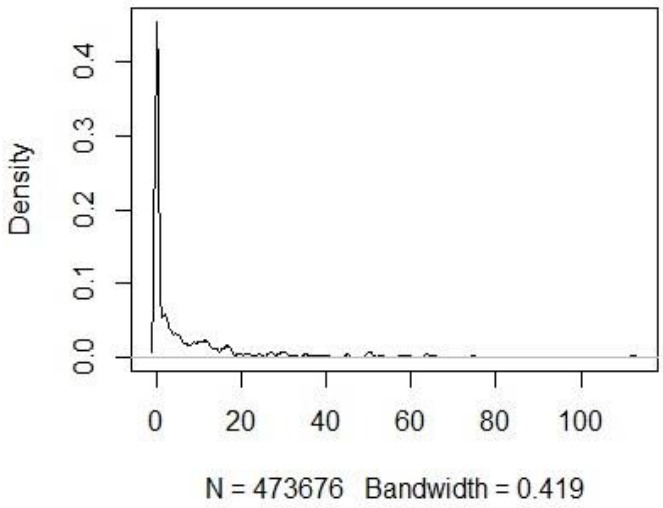


Figure 30: Density Graph of Theta. Vertical line represents density and horizontal line represents Theta values.

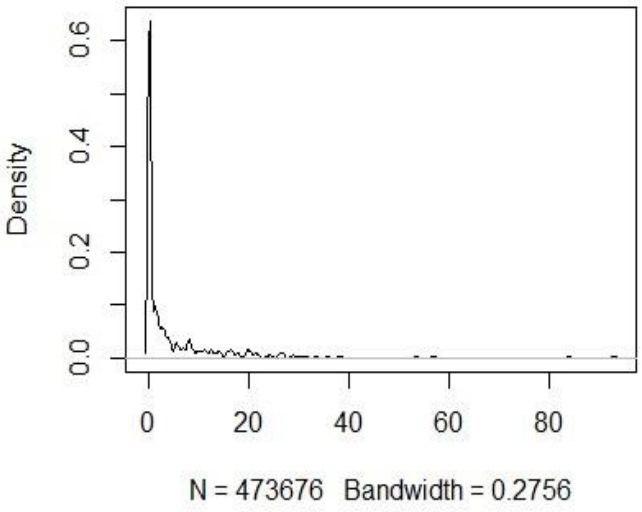


Figure 31: Density Graph of Alpha. Vertical line represents density and horizontal line represents Alpha values.

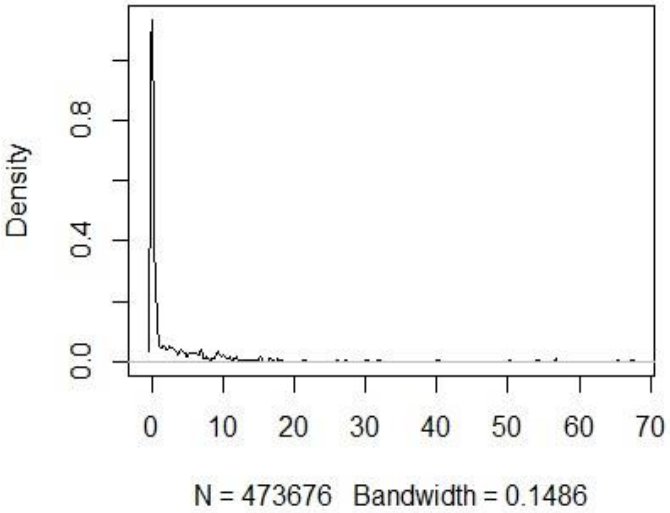


Figure 29: Density Graph of High_beta. Vertical line represents density and horizontal line represents High_beta values.

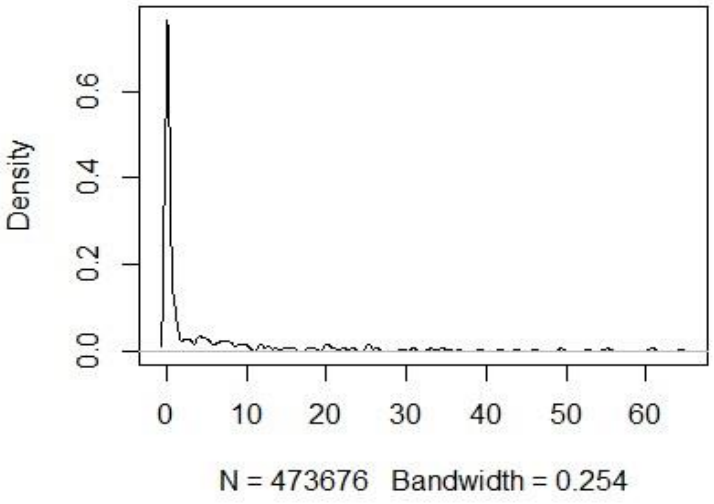


Figure 28: Density Graph of Low_beta. Vertical line represents density and horizontal line represents Low_beta values.

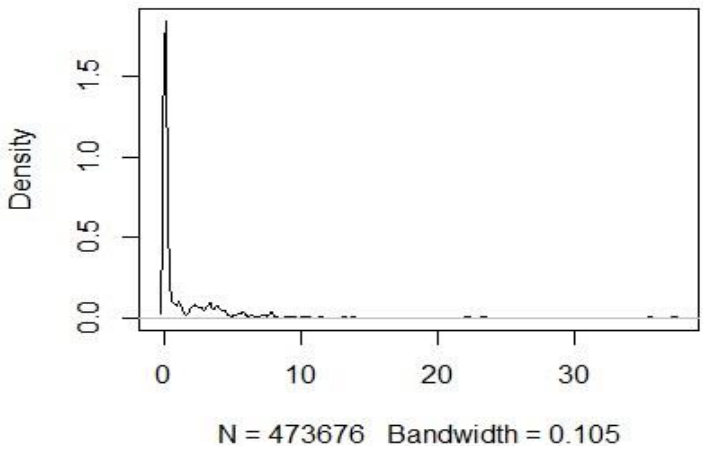


Figure 32: Density Graph of Gamma. Vertical line represents density and horizontal line represents Gamma values.

Histogram: Histograms of the two class data's features are shown below [26]

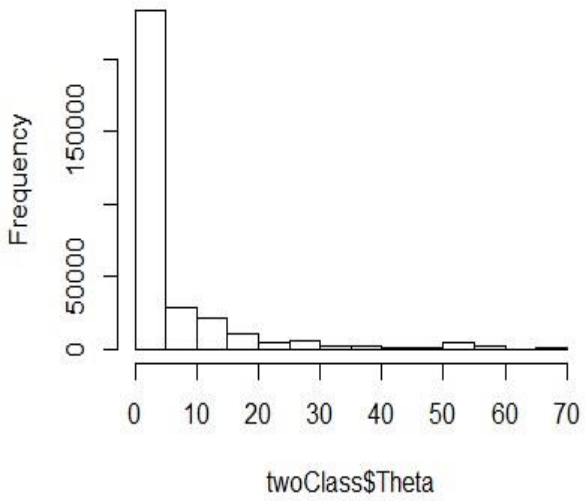


Figure 36: Histogram of Theta. Vertical line represents frequency and horizontal line represents Theta values.

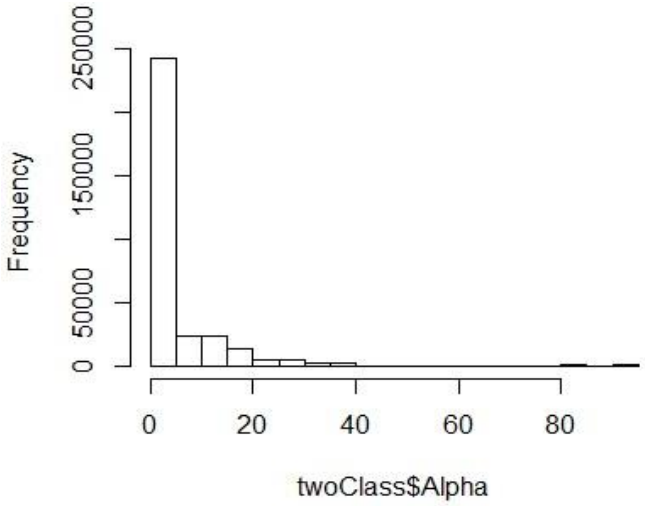


Figure 35: Histogram of Alpha. Vertical line represents frequency and horizontal line represents Alpha values.

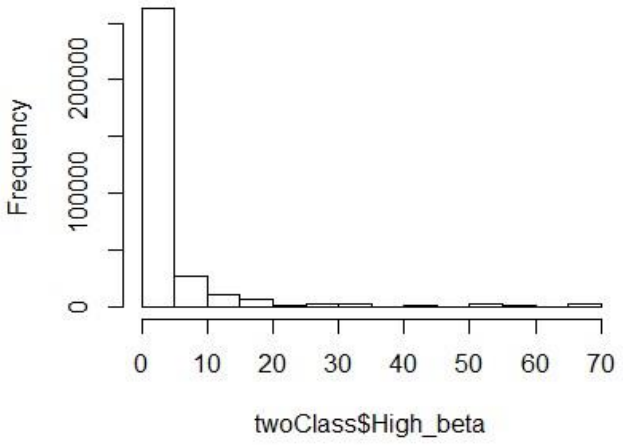


Figure 34: Histogram of High_beta. Vertical line represents frequency and horizontal line represents High_beta values.

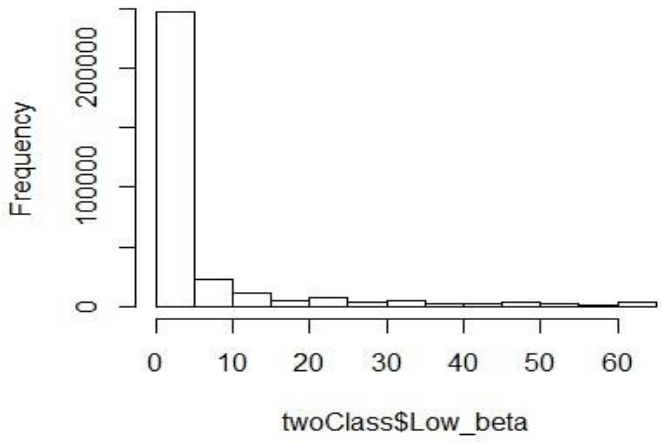


Figure 33: Histogram of Low_beta. Vertical line represents frequency and horizontal line represents Low_beta values.

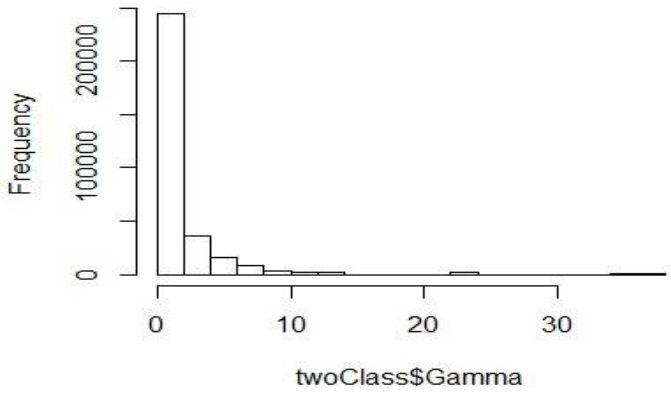


Figure 37: Histogram of Gamma. Vertical line represents frequency and horizontal line represents Gamma values.

Density Graph: Density graphs for the two class data's features are shown below [26]:

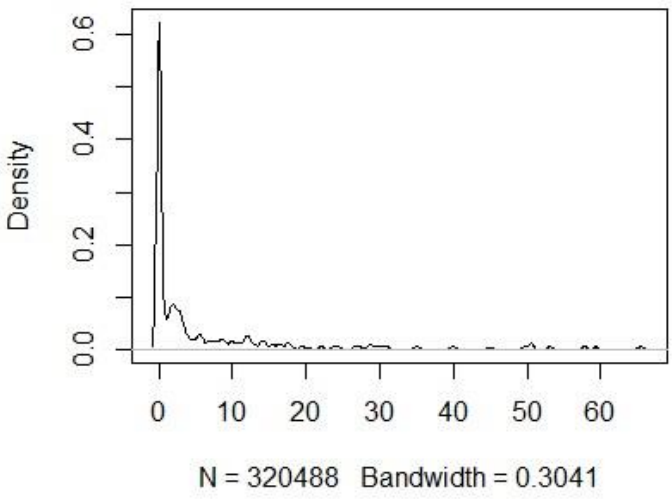


Figure 41: Density Graph of Theta. Vertical line represents density and horizontal line represents Theta values.

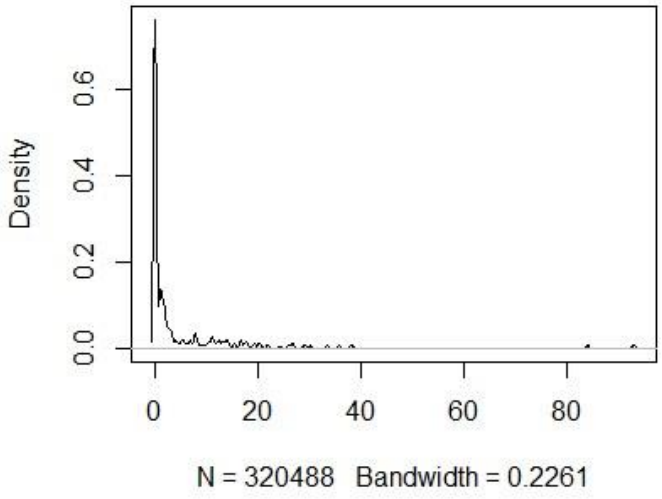


Figure 40: Density Graph of Alpha. Vertical line represents density and horizontal line represents Alpha values.

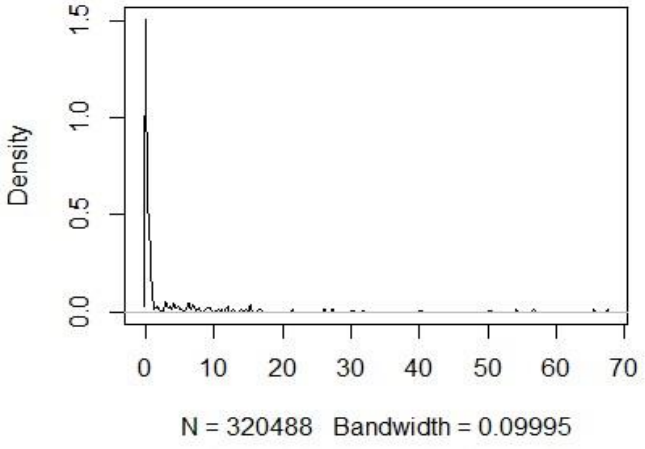


Figure 38: Density Graph of High_beta. Vertical line represents density and horizontal line represents High_beta values.

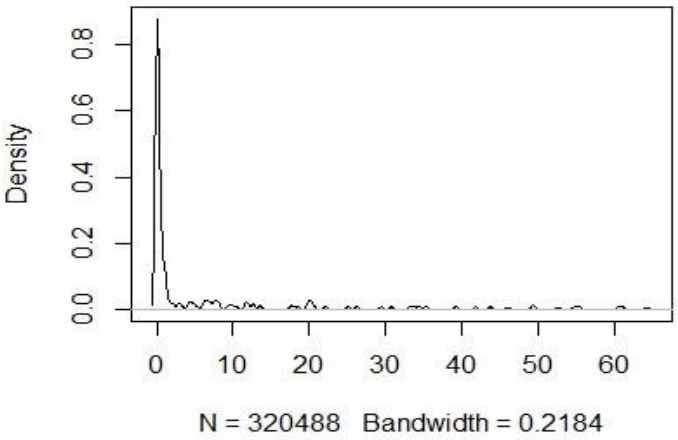


Figure 39: Density Graph of Low_beta. Vertical line represents density and horizontal line represents Low_beta values.

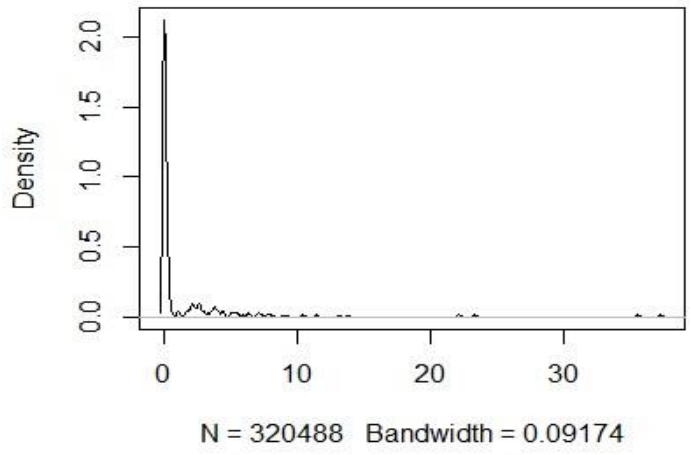


Figure 42: Density Graph of Gamma. Vertical line represents density and horizontal line represents Gamma values.

Barplot and Piechart:

Barplot and piechart for the class feature are shown below [26]:

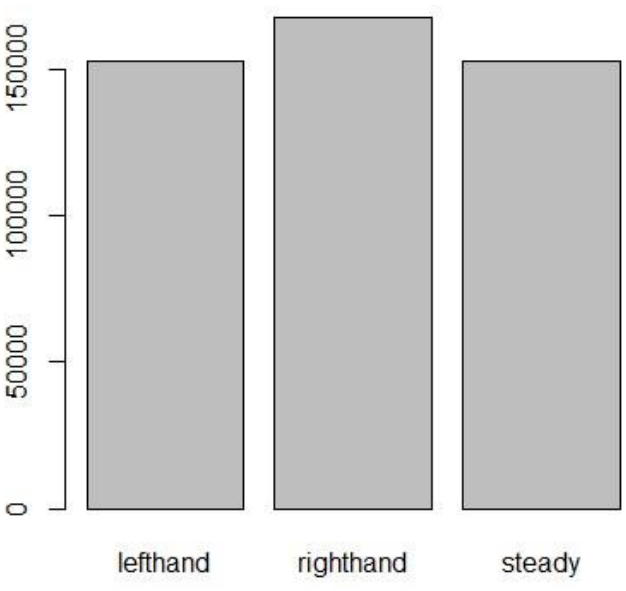


Figure 44: Barplot of three class.Vertical line represents frequency and horizontal line represents class labels.

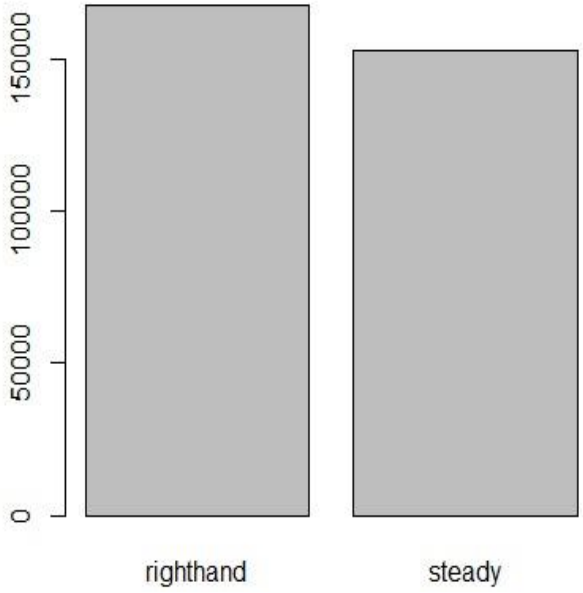


Figure 43: Barplot of two class. Vertical line represents frequency and horizontal line represents class labels.

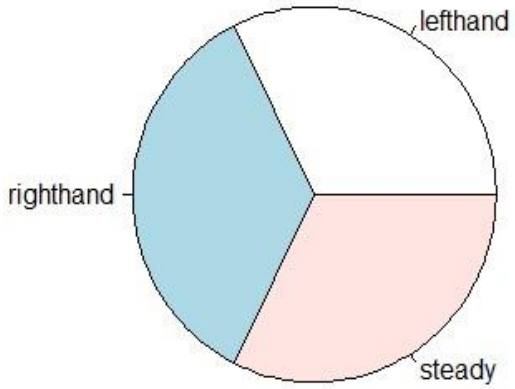


Figure 45: Piechart of three class. Class labels are righthand, lefthand and steady.

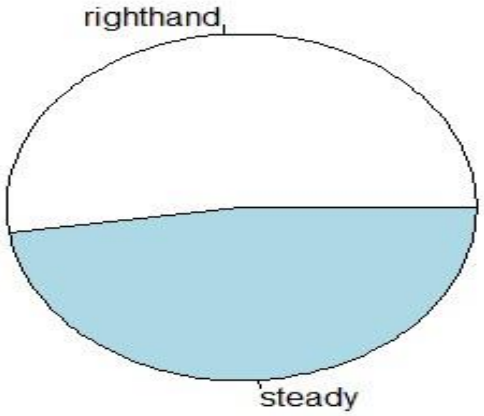


Figure 46: Piechart of two class. Class labels are righthand and steady.

4.3: Data Preprocessing

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format [26].

Feature selection:

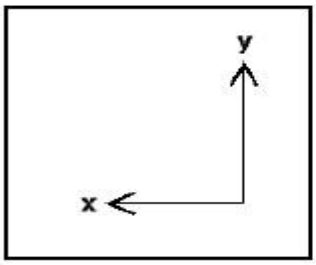
Feature selection is also known as subset selection or dimensionality reduction. It is a process commonly used in machine learning that select an optimal subset of features, which contains the least number of dimensions and have the most contribution to accuracy. Here we use five features and the features are Theta, Alpha, Low-beta, High-beta and Gamma [26].

Distance Measures:

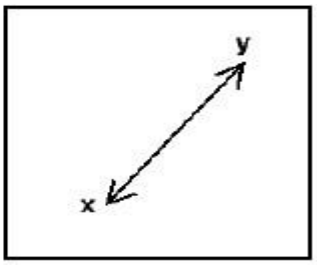
The distance between the two points in the plane with coordinate (x, y) and (a, b) is given by:

Euclidean Distance, (x, y) (a, b) =$\sqrt{(x-a)^2+(y-b)^2}$

Manhattan Distance, (x, y) (a, b) =|x-a|+|y-b|



Manhattan



Euclidean

Handle missing values:

Here I mention three technique to handle missing values.

1) When the values are numeric then average value replaced as missing value.
2) When the values are nominal then most frequent value replaced as missing value.
3) Nearest instance value replaced as missing value. Find the nearest instance you can use Manhattan or Euclidean distance.

For handle missing values we use first technique because all the data's of our brain signal are nominal.

Undersampling and oversampling:

Undersampling and oversampling are used for class imbalance problem. Undersampling and oversampling in data analysis are techniques used to adjust the class distribution of a data set.

Undersampling is cut or remove some data points. Here our total data points are 473676 and within these data points, righthand data points are maximum. We randomly remove some data points which class label is righthand.

Oversampling is added some data points artificially. Within our total data points, steady data points are minimum. We added some data points which attributes values are close to the attributes mean values and class value is steady.

Chapter 5: Classification

Classification is a data mining function that describes and distinguishes data classes or concepts. The goal of classification is to accurately predict class labels of instances whose attribute values are known, but class values are unknown [31].

5.1: Decision Tree

Decision tree (DT) induction is the learning of decision trees from class-labeled training instances, which is a top-down recursive divide and conquer algorithm. The goal of DT is to create a model (classifier) that predicts the value of a target class for an unseen test instance based on several input instances [27] [31].

5.1.1: Iterative Dichotomiser 3

The goal of DT is to iteratively partition the data into smaller subsets until all the subsets belong to a single class or the stopping criteria of DT building process are met [31].

$\text{Info (D)} = -\sum_{i=1}^{N} P_i \log_2(P_i)$

Here P_i is the probability that x_i belongs to the class, $C_l$and is estimated by $|C_l$, D|/|D|.

$\text{Info}_A\text{ (D)} = \sum_{j=1}^{n} \frac{|D_j|}{|D|} \times \text{Info } (D_j)$

Info_A (D) is the expected information required to correctly classify $x_i \in$ D based on the partitioning by A_j.

Gain (A) = Info (D) − Info_A (D)

Information gain is defined as the difference between Info (D) and Info_A (D).

5.1.2: C4.5

Quinlan later presented C4.5 (a successor of ID3 algorithm) that became a benchmark in supervised learning algorithms. C4.5 uses an extension of Information Gain, which is known as Gain Ratio [27] [31].

$\text{SplitInfo}_A\text{ (D)} = -\sum_{j=1}^{n} \frac{|D_j|}{|D|} \times \log_2(\frac{|D_j|}{|D|})$

It applies a kind of normalization of Information Gain using Split Information defined analogously to Info (D).

$\text{GainRatio (A)} = \frac{\text{Gain(A)}}{\text{SplitInfo(A)}}$

The A_j with the maximum Gain Ratio, is selected as the splitting feature.

5.1.3: Gini Index

The Gini Index is used in Classification and Regression Trees (CART) algorithm, which generates the binary classification tree for decision making. It measures the impurity of D, a data partition or X, as shown in Eq. 6, where, pi is the probability that $x_i \in$ D belongs to class, $C_l$and is estimated by $|C_l$,D$|/|$D$|$. The sum is computed over M classes [31].

Gini (D) = 1- $\sum_{i=1}^{N} P_i^2$

$\mathrm{Gini_A}$ (D) = $\sum_{\mathrm{j}=1}^{\mathrm{n}} \frac{|\mathrm{D_j}|}{|\mathrm{D}|}$ × Gini ($\mathrm{D_j}$)

$\mathrm{Gini_A}$ (D) is the expected information required to correctly classify $x_i \in$ D based on the partitioning by A_j.

ΔGini (A) = Gini (D) − $\mathrm{Gini_A}$ (D)

For each A_j, each of the possible binary splits is considered. The A_j that maximizes the reduction in impurity is selected as the splitting feature.

Decision Tree Induction Algorithm [31]

Input: D = $\{x_1, x_2, \ldots\ldots, x_N\}$

Output: A decision tree, DT

Method:

1: DT = ∅;

2: find the root node with best splitting, $A_j \in$ D;

3: DT = create the root node;

4: DT = add arc to root node for each split predicate and label;

5: **for** each arc **do**

6: $\mathrm{D_j}$created by applying splitting predicate to D;

7: **if** stopping point reached for this path **then**

8: DT' = create a leaf node and label it withC_l;

9: **else**

10: DT' = DTBuild (D_j);

11: **end if**

12: DT = add DT' to arc;

13: **end for**

5.2: OneR Classifier

OneR, short for "One Rule", is a simple, yet accurate, classification algorithm that generates one rule for each predictor in the data, then selects the rule with the smallest total error as its "one rule" [27] [31].

OneR Classifier Algorithm

Input: D = $\{x_1, x_2, \ldots\ldots, x_N\}$ // Training data.

Output: OneR Model.

Method:

1: **for** each attribute, $A_i \in$ D, **do**

2: **for** each attribute value, $A_{ij} \in A_i$, **do**

3: **Make a classification rule**:

4: count how often each class appears

5: find the most frequent class

6: make the rule assign that class to this A_{ij};

7: **end for**

8: Calculate the error rate of this attribute's A_i rule.

9: **end for**

10: Choose the attribute $A_i \in$ D with the smallest error rate.

5.3: Naïve Bayes Classifier

A naïve Bayes (NB) classifier is an important mining classifier for pattern classification and applied in many real world classification problems because of its high classification performance and can handle missing attributes values. It is a simple probabilistic classifier based on [31]:

a) Bayes theorem

b) Strong (naïve) independence assumptions

c) Independent feature models

Priori Probabilities

We will initially focus on the two-class case. Let C_1 and C_2 be the two classes in which our patterns belong. First, we need to estimate the priori probabilities, P (C_1) and P (C_2) from the available training feature vectors [31].

If N is the total number of available training patterns, and N_1, $N_2$of them belong to C_1 andC_2, respectively, the P (C_1) ≈ $\frac{N_1}{N}$ and P (C_2) ≈$\frac{N_2}{N}$.

Class-Conditional Probabilities

Second, we need to know the class-conditional probability density functions, P (x|C_i, i= 1, 2), describing the distribution of the feature vectors in each of the classes.

Now we have all the ingredients to compute the conditional probabilities using the Bayes rule [31]:

$$P(C_i|x) = \frac{P(x|C_i)P(C_i)}{P(x)}$$

Naïve Bayes Classifier Algorithm [31]

Input: D = $\{x_1, x_2, \ldots\ldots, x_N\}$ // Training data.

Output: A naïve Bayes Model.

Method:

1: **for** each class, $C_i \in$ D, **do**

2: Find the prior probabilities, P (C_i).

3: **end for**

4: **for** each attribute, $A_i \in$ D, **do**

5: **for** each attribute value, $A_{ij} \in A_i$, **do**

6: Find the class conditional probabilities, P ($A_{ij} \mid C_i$).

7: **end for**

8: **end for**

9: **for** each instance, $x_i \in$ D, **do**

10: Find the posterior probability, P ($C_i \mid x_i$);

11: **end for**

5.4: Conclusion

In chapter 4, we mentioned about our data set that our class values are Lefthand, Righthand and Steady. It is supervised learning and we can apply classification algorithms. We need some models that can classify unknown instances. For creating models, we have applied several classification methods including - OneR, Naïve Bayesian, C 4.5, and CART.

Chapter 6: Experimental Results

To check the performance of classification models over our hand movement dataset we have applied four classification models. First we visualize our features Statistics to begin the experiment. Then we applied OneR classifier, NB classifier, C4.5 decision tree, and classification and regression tree (CART) [31].

6.1: Attribute Statistics

The table below describes the maximum, the minimum, the mean and the standard deviation values for each of the attributes in the dataset.

Table 10: 3 class hand movement data's attribute statistics.

Attribute Name	Min Value	Max Value	Mean Value	STD Value
Theta	0.009	112.322	6.839	13.141
Alpha	0.006	92.899	4.999	10.214
Low_beta	0.006	64.354	5.529	11.349
High_beta	0.01	67.406	3.317	8.316
Gamma	0.01	37.254	1.637	3.833

Table 11: 2 class hand movement data's attribute statistics.

Attribute Name	Min Value	Max Value	Mean Value	STD Value
Theta	0.009	65.455	5.806	11.453
Alpha	0.006	92.899	4.696	10.451
Low_beta	0.006	64.354	5.578	12.481
High_beta	0.01	67.406	3.646	9.744
Gamma	0.01	37.258	1.681	4.416

6.2: Evaluating Classifiers Performance

For the experiment, we have used WEKA data mining tool to implement the classification models. The result that we get after implementing each classification models is then used to test our hand movement test data. Basically we used 2 classes and 3 classes in our experiments and the results of the classifiers are [31]:

Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

Error Rate = $\frac{FP+FN}{TP+TN+FP+FN}$

Precision = $\frac{TP}{TP+FP}$

Recall = $\frac{TP}{TP+FN}$

F-Score = $\frac{2*\text{Precision}*\text{Recall}}{\text{Precision} + \text{Recall}}$

Table 12: 2 class hand movement data's performance.

Name of Classifier	Accuracy %	Error Rate %	Sensitivity/Recall	Precision	F-score
OneR	73.6961	26.3039	0.737	0.765	0.734
NB Classifier	73.3194	26.6806	0.733	0.830	0.718
C 4.5	90.8917	9.1083	0.909	0.912	0.908
CART	91.3353	8.6647	0.913	0.914	0.913

Here, Classification and Regression Trees (CART) algorithm performs best with 91.34% accuracy and C4.5 decision tree also performs well with 90.89% accuracy for our 2 classes hand movement data.

Table 13: 3 class hand movement data's performance.

Name of Classifier	Accuracy %	Error Rate %	Sensitivity/Recall	Precision	F-score
OneR	51.2973	48.7027	0.513	0.524	0.515
NB Classifier	59.9476	40.0524	0.599	0.738	0.587
C 4.5	65.6612	34.3388	0.657	0.668	0.660
CART	52.5688	47.4312	0.526	0.537	0.530

Here, C4.5 decision tree performs well with 65.66% accuracy for our 3 classes hand movement data.

6.3: Build Decision Tree

Here, C4.5 decision tree performs well for both data sets. We build decision tree (model) using C4.5 algorithm and 2 class hand movement data. We use this model to develop a game and this game is played by our brain thought.

6.4: Rule-base Classifier

Rule-based classifier is easy to deal with complex classification problems. It has various advantages:

1: Highly expressive as decision tree (DT)

2: Easy to interpret

3: Easy to generate

4: Can classify new instances rapidly

5: Performance comparable to DT

6: New rules can be added to existing rules without disturbing ones already in there

7: Rules can be executed in any order

We create rules using our decision tree model so that we can easily use it in our game development.

6.5: Conclusion

In this chapter, we have find out attributes statistics for 2 class data's as well as for 3 class data's. We have also evaluated OneR, NB Classifier, C 4.5, CART classifiers performances. For two-class classifier CART gave best performance, 91.3353% accuracy. But, C 4.5 did give very close performance to it, 90.8917% accuracy. For three-class classifier C 4.5 gave best performance, 65.6612% accuracy. We have considered overall performance and used C4.5 model to create classification rules.

Chapter 7: Developing Game Using Emotiv Epoc+

7.1: Emotiv Epoc+ 14 Channel Headset

Emotiv Epoc+ is a brain signal taker device [10]. We use it for our research**.** First we put headset on head using 10-20 system. Then turn on the device and connect dongle with pc and open Emotive Xavier control panel. After successful connection all the point will be green as this picture.

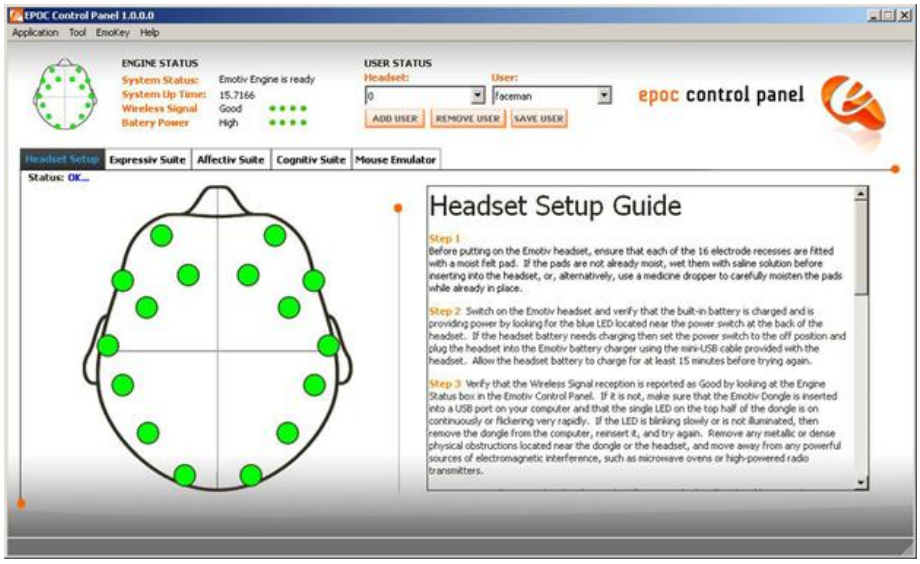


Figure 47: Emotive Control Panel [10]

7.2: Electrode Placement Using 10-20 System

Electrode placement using 10-20 system for a classical EEG recording has become common now a days. In this system, the distance in percentages between fixed points and Nasion-Inion is 10/20.The actual distance between two adjacent electrodes be can either 10% or 20% of the total left-right or front back distance of scalp. These points are called as Occipital (O), Frontal Pole (FP), Central(C), Parietal (P) and Temporal (T).

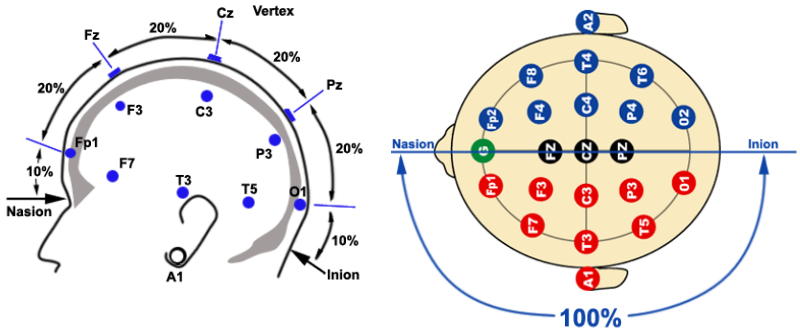


Figure 48: 10/20 System Electrode Positioning

7.3: Functional Areas of Brain

Motor Cortex Area: [12][13][14][15]

One of the region of the cerebral cortex is Motor Cortex involved in the controlling, plan and execution of voluntary movements. Frontal lobe's motor cortex is an area located in the posterior pre-central gyrus, in the immediately anterior to the central sulcus. It can be divided into three areas- Primary Motor Cortex, Premotor Cortex and Supplementary Motor Area (or SMA).

- The Primary Motor Cortex is the main contributor which generate neural impulses which pass down via spinal cord and control the execution of movement.
- The Premotor Cortex is responsible for motor control partially. Possibly, including, in the preparation for movement, reaching of the spatial guidance, some movements with an emphasis on trunk muscles and control of proximal of the body.
- The Supplementary Motor Area has many proposed functions. Including, the internally generated planning of movement, planning of sequences movement, coordination in bi-manual coordination.

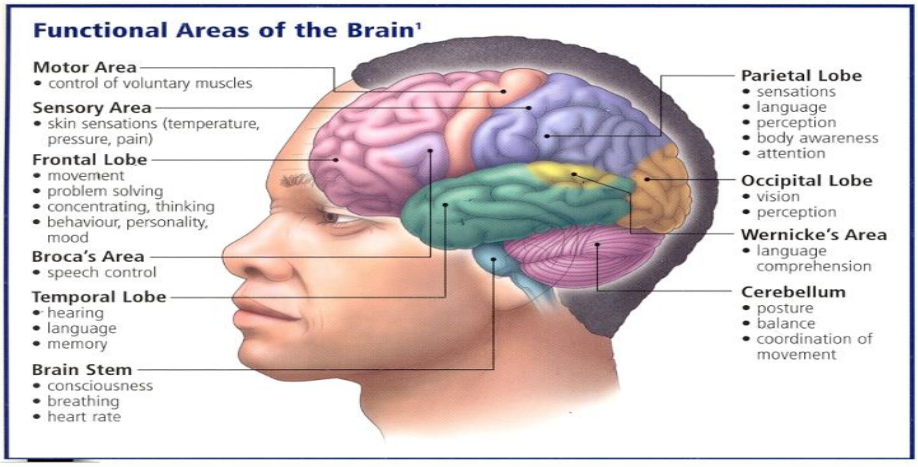


Figure 49: Functional Areas of Brain

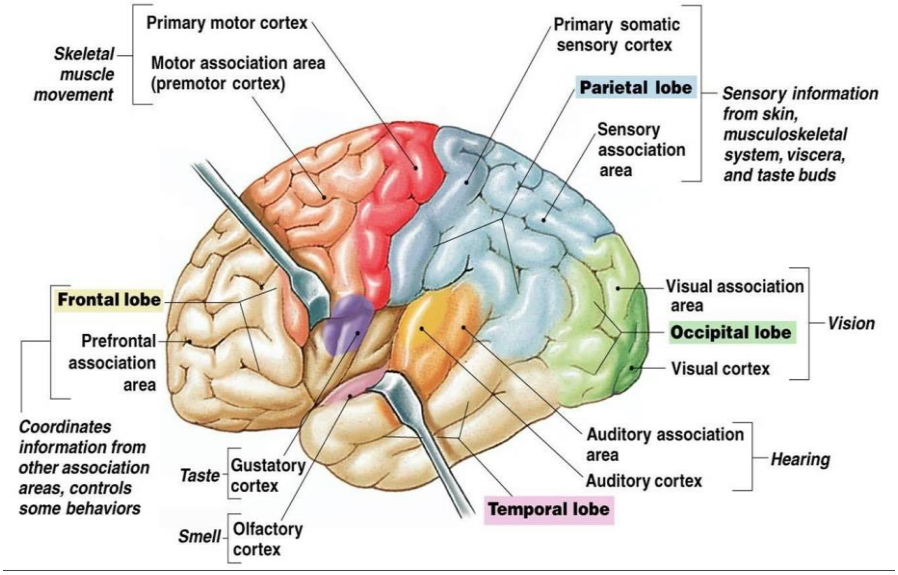


Figure 50: Functional Areas of Brain

7.4: Emotiv Epoc+ Headset Placement

Slide the headset down from the top of your head by using both hands. Put the hands approximately as depicted, place the sensors carefully, insert on the bone just behind each ear lobe with the black rubber. The rubber sensor correct placement is critical for correct operation.

Notice, the 2 front sensors should be place approximately about the width of 3 fingers above your eyebrows or at the hairline [10].

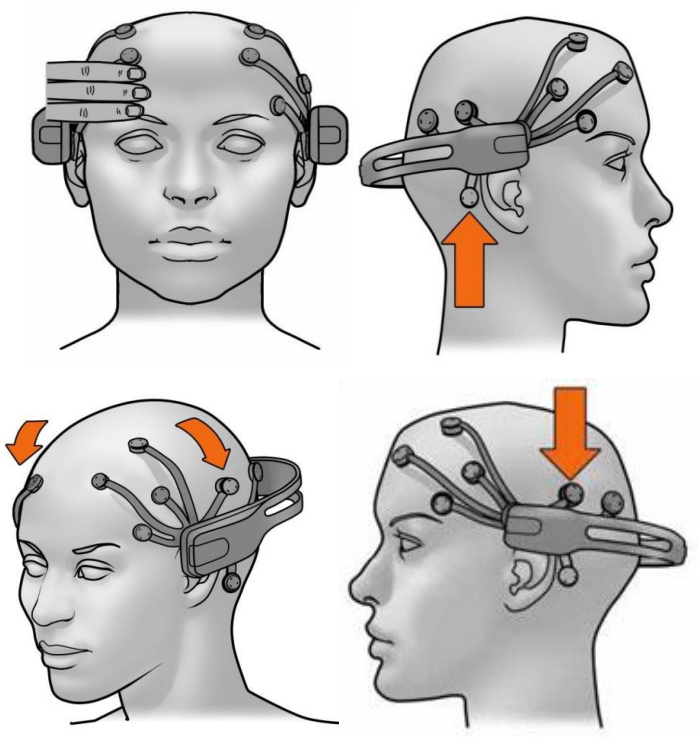


Figure 51: Emotiv Epoc+ Headset Placement [10]

7.5: Emotiv Epoc+ Headset Referential Electrodes

The EPOC has 4 'slots' for reference electrodes. In order to perform a recording, you must place electrodes in at least 2 of them. This means you can either use the bottom ones marked in red in the picture, or the top ones marked in green [10].

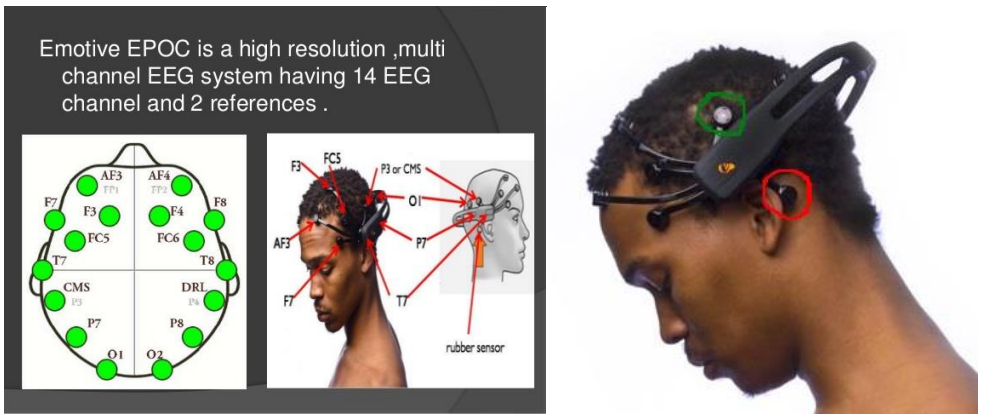


Figure 52: Emotiv Epoc+ Headset Referential Electrodes

Whether you use the 'top' or the 'bottom' positions for the reference electrodes makes no difference in the sensor map, as both positions are mapped to the same indicators in the sensor map.

Electric potential (voltage) is always measured as a difference between two points, in this case as the difference between each of the 14 electrodes and the average of the two reference electrodes. This means 14 data channels are available, even though 17 electrodes are attached.

- 14 channels are AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC7, F4, F8, AF4
- 2 references are in the CMS (Common Mode Sense) active electrodes / DRL (Driven Right Lag) passive electrodes and noise cancellation configuration P3-P4 locations.

7.6: Emotiv Epoc+ Headset SDKs

Emotiv give us three types of SDKs- Basic, Advance and Prime. Here is comparison of these SDKs [10].

	BASIC	ADVANCED	PRIME
Description	for BCI, wellness and performance training, games	for EEG research, education, neurofeedback	for market and consumer research, usability testing
API access			
Mental Commands 9 Axis Inertial Sensors Facial Expressions Frequency Bands Performance Metrics - Low Res	✓	✓	✓
Raw EEG		✓	✓
Performance Metrics - High Res			✓

Table 14: Comparison between these SDKs

7.7: Record Signals from Emotiv Epoc+

We use NetBeans IDE and Emotiv Community SDK as modification of java code and take signal from Emotiv Headset Device [10]. Hare is the function marked as red which is used for signal taker function.

```
        // Log the EmoState if it has been updated
        if (eventType == Edk.IEE_Event_t.IEE_UserAdded.ToInt()) {
            if (userID != null) {
                System.out.println("User added");
                ready = true;
            }
        }
    } else if (state != EdkErrorCode.EDK_NO_EVENT.ToInt()) {
        System.out.println("Internal error in Emotiv Engine!");
        break;
    }
        if (ready) {

        DoubleByReference alpha = new DoubleByReference(0);
        DoubleByReference low_beta = new DoubleByReference(0);
        DoubleByReference high_beta = new DoubleByReference(0);
        DoubleByReference gamma = new DoubleByReference(0);
        DoubleByReference theta = new DoubleByReference(0);

        for (int i = 3; i < 17; i++) {
            int result = Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), 4, theta, alpha, low_beta, high_beta, gamma);
```

We use Average Band Power Community SDK samples and record theta, alpha, low-beta, high beta, gamma filtered signals.
While, complete Emotiv Epoc+ channel labels are: [IED_AF3, IED_F7, IED_F3, IED_FC5, IED_T7, IED_P7, IED_Pz, IED_O1, IED_O2, IED_P8, IED_T8, IED_FC7, IED_F4, IED_F8, IED_AF4]
But if we try to map above labels with indexes, as is the case below:
IED_AF3 = 3, IED_F7 = 4, IED_F3 = 5, IED_FC5 = 7, IED_T7 = 7, IED_P7 = 8, IED_Pz = 9, IED_O1 = 10, IED_O2 = 11, IED_P8 = 12, IED_T8 = 13, IED_FC7 = 14, IED_F4 = 15, IED_F8 = 17, IED_AF4 = 17
The indexes run from 3 to 17. The background buffer contains additional columns (such as sample counter, contact quality, motion sensors etc.) which explains the EEG indexing commencing at 3.
We take channels 5, 7, 14, 15 (F3, PC5, PC7, F4) which are more related with motor cortex.

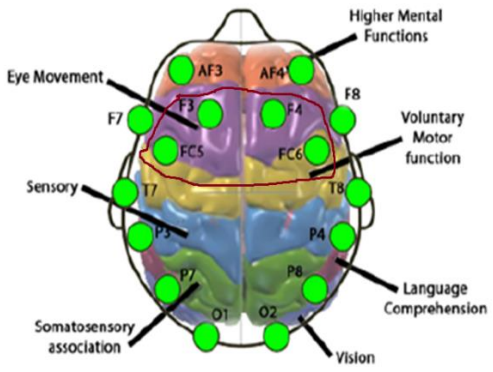


Figure 53: Brain Signal Recorded Channels

7.8: Live Detection of Brain Signals Using Models of Data

We Build Our Data Model using R-programming and “Weka” Software by supplying test and train dataset. By using this data model we compare the possibility in which class any

data should be classified. For this we made a GUI using JavaFX and show some action and percentage of different classified class. In this game, we can control ball movement by using brain signals of voluntary movements. There are two versions of this game. One is for two class classifications - actions are steady and right hand move. Another is for three class classifications - actions are left hand move, steady and right hand move. If the player remain steady and concentrate about steady the Graphics User Interface will be show like the first picture. If the player move his/her right hand and concentrate about right hand movement the Graphics User Interface will be show like the second picture. We give some animation of these balls to make more user comfort visualization. [Appendix A]

Figure 54: 2 Class GUI (steady and right hand move)

In the three class version game, if the player move his/her left hand and concentrate about left hand movement the Graphics User Interface will be show like the first picture. If the player remain steady and concentrate about steady the Graphics User Interface will be show like second picture. If the player move his/her right hand and concentrate about right hand movement the Graphics User Interface will be show like the third picture.

Figure 55: 3 Class GUI (left hand move, steady and right hand move)

7.9: Conclusion

After studying about several brain signal taker devices, we choose Emotiv Epoc+ headset because it is best in our budget. We studying about this device connectivity, signal type, different type of SDK Emotiv provides, overall how to use it to make a live brain signal classification game development. We have studied about motor cortex area of brain because motor cortex area is responsible for hand movement. We only take 4 electrode channels (F3, PC5, PC7, F4) which more related with motor cortex area. We place Emotive Epoc+ headset electrodes using 10-20 system. In the context of an EEG test, the 10–20 system is an internationally recognized method to describe and apply the location of scalp electrodes. We modify Emotiv SDK codes to record brain in our desired format in .arff type file to build our data model which will be used to classify new live brain signals. We build our data model using weka library integrated with our Java project and use that classification method to build model which give us less % of error in 10-fold validation of result. In our experiment, we want to classify left hand movement, right hand movement and steady mode brain signals. We make two versions of game one is classify two class (right hand, steady) and another is classify thee class (left hand, steady, right hand).

Chapter 8: Conclusion and Future Work

8.1: Conclusion

In our thesis, first we have taken a sufficient understanding of both assistive Human Machine Interface and Brain Machine Interface field. We have gained a sound apprehension of the opportunities and advantages that are about to be brought about by this technology in health and rehabilitation sectors. We have studied about brain signals and signal processing techniques. We have analyzed several BMI devices that can be found in the market like- Aurora Dream Headband, Muse Headband, MindWave Headset, Emotiv Insight 5 Channel Mobile EEG and Emotiv Epoc+ 14 channel mobile EEG. We have selected Emotiv Epoc+ to use in our research as it is more suitable for research grade activities than the other devices. We have worked hard to gain sufficient understanding of Emotiv Epoc+'s working mechanism. After completing these foundational labors we have decided to build an application program that can differentiate unalike human thoughts and take disparate actions on the basis of these identifications.

We have utilized Emotiv SDK and Java technology to make a program in order to acquire the brain signals. We have used R-programming and weka tool for data visualization and model construction. To make our desired program, at the first phase of our work, we gathered brain signals and extracted data from this signals. This extracted data was analyzed to obtain informative patterns. We have collected data to develop a two class classifier program. Two-class classifier program can make distinction between right-hand movement and steady states. We have also gathered data to develop a three class classifier program. Three-class classifier program identifies from right-hand movement, left-hand movement and steady states. At the next phase of the thesis, we have applied different classification methods on both two-class classifier problem and three-class classifier problem. These methods are: - OneR, Naïve Bayesian, C 4.5, and CART. For two-class classifier problem their performances are consequently 73.6961%, 73.3194%, 90.8917% and 913353% accuracy. For three-class classifier problem their performances are consequently 51.2973%, 59.9476%, 65.6612% and 52.5688% accuracy. We can see, for two-class classifier program CART gave best performance, 91.3353% accuracy. But, C 4.5 did give very close performance to it, 90.8917% accuracy. On the other hand, for three-class classifier program C 4.5 gave best performance, 65.6612% accuracy. After evaluating overall result, we have used C4.5 model to make the targeted application system. We have classified live brain signals using this model for both two-class and three-class classifier programs. We have developed a virtual-ball movement controlling game. In this game we can control the movement of a virtual-ball using brain signals of voluntary movements without any need of conventional input devices like- mouse, keyboard, joystick, etc.

Our game has two versions. In one version, the virtual-ball can be in two states: - steady state and right hand state. When the user mind thinks of right and moves right hand, the ball gets toward all the sides of the console screen following the user's mental processes. In the same way, when the user mind thinks of being steady, the ball gets positioned on the center of the console screen. In the other version of our game, the virtual-ball can be in three states: - steady, left side and right side states. Here, when the user brain thinks of left and moves left hand, the ball gets positioned on the left side of the console. When the user brain thinks of right and moves right hand, the ball gets positioned on the right side of the console. When the user brain is at steady, the ball gets positioned on the center of the console screen. Playing this game user can exercise and enhance attention of his mind. Thus, it helps the user to increase his attention time-span; allows him to perform his actions more intensely, improves the quality of his works; makes his life more productive. Moreover, playing this game also provides with amusement to the user.

8.2: Limitation

While making the game we have collected data from few people's brains. If we could obtain data from many people's brains our program could give more accurate performances. The game that we have developed works only on Windows OS. However, it does not support Android OS, iOS and other operating systems. It is a single player game; it cannot be played by two players simultaneously. In addition with that, our program can make distinction from only two or three states; it cannot make distinction from states more than that.

8.3: Future Work

We have created single player game. In future, it can be upgraded into a multiplayer game in order to make it more challenging and entertaining to play. It would be also possible to make android versions of both single player and multiplayer games. Correspondingly, it is also attainable to make the game compatible with iOS platform. The number of actions in the game can be increased in order to make the gaming experience more diverse and captivating. The finding of our thesis can be enhanced and implemented to control and manipulate movements of robotic agents. The finding of our thesis may open up new doors of opportunities in the health and well-being of our society.

References

[1] Khondaker Abdullah-Al-Mamun, "Pattern identification of movement related states in biosignal," Thesis for the degree of Doctor of Philosophy, June 2012.
[2] Sarah N. Abdulkader, Ayman Atia, Mostafa-Sami M.Mostafa, "Brain computer interfacing: Applications and challenges," Egyptian Informatics Journal, July 2015.
[3] Hall, John (2011). Guyton and Hall textbook of medical physiology (12th ed.), p 685
[4] Hall, John (2011). Guyton and Hall textbook of medical physiology (12th ed.), p 687
[5] Khondaker Abdullah-Al-Mamun, "Pattern identification of movement related states in biosignal,"Thesis for the degree of Doctor of Philosophy, June 2012.
[6] Khondaker Abdullah-Al-Mamun, "Pattern identification of movement related states in biosignal,"Thesis for the degree of Doctor of Philosophy, June 2012.
[7] Rabie A.Ramadan, S. Refat, Marwa A. Elshahed and Rasha A. Ali, "Basics of Brain Compuer Interface", Textbook (2015), Chapter 2.
[8] Khondaker Abdullah-Al-Mamun, "Pattern identification of movement related states in biosignal,"Thesis for the degree of Doctor of Philosophy, June 2012.
[9] M. Rajya Lakshmi, Dr. T. V. Prasad, Dr. V. Chandra Prakash, "Survey on EEG Signal Processing Methods", International Journal of Advanced Research in Computer Science and Software Engineering, January 2014.
[10] Emotiv user menual 2014 And Documentation on Eemotiv Epoc+ and SDKs.
[11] McGill University Physiology Virtual Lab documentation on Biomedical Signals Acquisition.
[12]Campbell, A. W, "Histological Studies on the Localization of CerebralFunction", Cambridge, MA: Cambridge University Press, 1905.
[13] Roland, P.E., Larsen, B., Lassen, N.A. and Skinhoj, "Supplementary motor area and other cortical areas in organization of voluntary movements in man", at 1980J. Neurophysiol. 43(1):p 118–136.
[14] Muhammad, R., Wallis, J.D. and Miller, "A comparison of abstract rules in the prefrontal cortex, premotor cortex, inferior temporal cortex, and striatum". J. Cogn. Neurosci. 18 (6): 974–989.
[15] Fulton "A note on the definition of the "motor" and "premotor" areas". Year 1935 Brain. 58(2): p 311–316.
[16]Raja Majid Mehmood, Hyo Jong Lee, "Towards human brain signal preprocessing and artifact rejection methods", Conference: Int'l Conf. Biomedical Engineering and Sciences (BIOENG'16), At Lasvegas, USA July 2016.
[17] Niedermeyer E.; da Silva F.L."Electroencephalography: Basic Principles, Clinical Applications, and Related Fields". Lippincott Williams & Wilkins. Year 2004.
[18] Jung, Tzyy-Ping; Makeig, Scott; Humphries, Colin; Lee, Te-Won; McKeown, Martin J.; Iragui, Vicente; Sejnowski, Terrence J., "Removing electroencephalographic artifacts by blind source separation". Year 2000, Psychophysiology. 37 (2): p 163–78.
[19] Nolan, H.; Whelan, R.; Reilly, R.B.,"FASTER: Fully Automated Statistical Thresholdfor EEG artifact Rejection". Year 2010, Journal of Neuroscience Methods. 192 (1): p 152–162.

[20] Richard Markell. "Better then Bessel Linear Phase Filters for Data Communications" Year 1994. p. 3.
[21] Miroslav D. Lutovac, Dejan V. Tosic, "Brian Lawrence Evans, Filter Design for Signal Processing Using MATLAB and Mathematica, Miroslav Lutovac", Year 2001.
[22] B. A. Shenoi, John Wiley & Sons, "Introduction to Digital Signal Processing and Filter Design",Year 2005.
[23] L. D. Paarmann, Springer, "Design and Analysis of Analog Filters: A Signal Processing Perspective", Year 2001.
[24] J.S.Chitode, "Digital Signal Processing", Technical Publications, Year 2009.
[25] Leland B. Jackson, Springe, "Digital Filters and Signal Processing", Year 1996.
[26] Ishteaque Alam, Dewan Md. Farid, Swakkhar Shatabda, “Pattern Mining from Historical Traffic Big Data”, IEEE Technologies for Smart Cities, 14 – 16 July, 2017
[27] Dewan Md. Farid, Ann Nowe, Bernard Manderic, “A Feature Grouping Method for Ensemble Clustering of High-Dimensional Genomic Big Data”, Future Technologies Conference 2016 6-7 December 2016 | San Francisco, United States.
[28] Dewan Md. Farid, Mohammad Abdullah Al-Mamun, Bernard Manderick, Ann Nowe, “An adaptive rule-based classifier for mining big biological data”, Department of Population Medicine & Diagnostic Sciences, College of Veterinary Medicine, Cornell University, Ithaca, NY 14850, USA.
[29] AlZoubi, O., Koprinska, I., & Calvo, R. A. (2008). Classification of Brain-Computer Interface Data. In the 7th Australasian Data Mining Conference (AusDM 08), Adelaide, Australasian, 123-131.
[30] Andersen, R. A., Musallam, S., & Pesaran, B. (2004). Selecting the signals for a brain-machine interface. Current Opinion in Neurobiology, 14(6), 720-726.
[31] Deyan Md. Farid, Li Zhang, Chowdhury Mofizur Rahman, “Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks”

Appendix A

Emotiv Epoc+ data saving Java code:

```
public class AverageBandPowers {

private static String className;

 public static void main(String[] args) throws FileNotFoundException {

      PrintWriter pw = new PrintWriter(new File("omirighthandtest.arff"));//added this

      //className = "steady";
      //className = "lefthand";
      className = "righthand";
      //className = "movement";

      Pointer eEvent = Edk.INSTANCE.IEE_EmoEngineEventCreate();
      Pointer eState = Edk.INSTANCE.IEE_EmoStateCreate();

      IntByReference userID = null;
      boolean ready = false;
      int state = 0;

      Edk.IEE_DataChannels_t dataChannel;

      userID = new IntByReference(0);

      if(Edk.INSTANCE.IEE_EngineConnect("Emotiv Systems-5…") !=
EdkErrorCode.EDK_OK
              .ToInt()) {

         return;
      }

      //added this
          pw.write("@relation relationName.attribute.NumericToNominal-
          weka.filters.unsupervised.attribute.NumericToNominal\n\n");
      pw.write("@attribute Theta numeric\n");
      pw.write("@attribute ' Alpha' numeric\n");
      pw.write("@attribute ' Low_beta' numeric\n");
      pw.write("@attribute ' High_beta' numeric\n");
      pw.write("@attribute ' Gamma' numeric\n");
      pw.write("@attribute Class {lefthand,righthand,steady}\n\n");
      pw.write("@data\n");
```

```
        while (true) {
            state = Edk.INSTANCE.IEE_EngineGetNextEvent(eEvent);

            // New event needs to be handled
            if (state == EdkErrorCode.EDK_OK.ToInt()) {
                int eventType = Edk.INSTANCE.IEE_EmoEngineEventGetType(eEvent);
                Edk.INSTANCE.IEE_EmoEngineEventGetUserId(eEvent, userID);

                // Log the EmoState if it has been updated
                if (eventType == Edk.IEE_Event_t.IEE_UserAdded.ToInt()) {
                    if (userID != null) {
                        ready = true;
                    }
                }
            } else if ( EdkErrorCode.EDK_NO_EVENT.ToInt() != state) {
                break;
            }

            //if (true) {
                if (ready) {

                DoubleByReference alpha = new DoubleByReference(0);
                DoubleByReference low_beta = new DoubleByReference(0);
                DoubleByReference high_beta = new DoubleByReference(0);
                DoubleByReference gamma = new DoubleByReference(0);
                DoubleByReference theta = new DoubleByReference(0);

                for (int i = 3; i < 17; i++) {
                    int result = Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), 4,
theta, alpha, low_beta, high_beta, gamma);
                    //if (true) {//edited
                        if(result == EdkErrorCode.EDK_OK.ToInt()){


                        //added this
                        pw.write(String.valueOf(theta.getValue()));
                        pw.write(",");
                        pw.write(String.valueOf(alpha.getValue()));
                        pw.write(",");
                        pw.write(String.valueOf(low_beta.getValue()));
                        pw.write(",");
                        pw.write(String.valueOf(high_beta.getValue()));
                        pw.write(",");
                        pw.write(String.valueOf(gamma.getValue()));
                        pw.write(",");
                        pw.write(className);
```

```
                }

                //added this
                if(theta.getValue()!=0)
                   {
                // if (true) {
                   pw.write("\n");
                }//edited
                //*************
                 result = Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), 5,
theta, alpha, low_beta, high_beta, gamma);
                //if (true) {//edited
                    if(result == EdkErrorCode.EDK_OK.ToInt()){


                   //added this
                   pw.write(String.valueOf(theta.getValue()));
                   pw.write(",");
                   pw.write(String.valueOf(alpha.getValue()));
                   pw.write(",");
                   pw.write(String.valueOf(low_beta.getValue()));
                   pw.write(",");
                   pw.write(String.valueOf(high_beta.getValue()));
                   pw.write(",");
                   pw.write(String.valueOf(gamma.getValue()));
                   pw.write(",");
                   pw.write(className);

                }

                //added this
                if(theta.getValue()!=0)
                   {
                // if (true) {
                   pw.write("\n");
                }//edited
                //****************
                 result = Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), 14,
theta, alpha, low_beta, high_beta, gamma);
                //if (true) {//edited
                    if(result == EdkErrorCode.EDK_OK.ToInt()){

                   //added this
                   pw.write(String.valueOf(theta.getValue()));
                   pw.write(",");
                   pw.write(String.valueOf(alpha.getValue()));
```

```
                pw.write(",");
                pw.write(String.valueOf(low_beta.getValue()));
                pw.write(",");
                pw.write(String.valueOf(high_beta.getValue()));
                pw.write(",");
                pw.write(String.valueOf(gamma.getValue()));
                pw.write(",");
                pw.write(className);

            }

            //added this
            if(theta.getValue()!=0)
               {
            // if (true) {
               pw.write("\n");
            }//edited
            //*******************
             result = Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), 15,
theta, alpha, low_beta, high_beta, gamma);
            //if (true) {//edited
                if(result == EdkErrorCode.EDK_OK.ToInt()){


                //added this
                pw.write(String.valueOf(theta.getValue()));
                pw.write(",");
                pw.write(String.valueOf(alpha.getValue()));
                pw.write(",");
                pw.write(String.valueOf(low_beta.getValue()));
                pw.write(",");
                pw.write(String.valueOf(high_beta.getValue()));
                pw.write(",");
                pw.write(String.valueOf(gamma.getValue()));
                pw.write(",");
                pw.write(className);

            }

            //added this
            if(theta.getValue()!=0)
               {
            // if (true) {
               pw.write("\n");
            }//edited
         }
      }
```

```
        }

        Edk.INSTANCE.IEE_EngineDisconnect();
        Edk.INSTANCE.IEE_EmoStateFree(eState);
        Edk.INSTANCE.IEE_EmoEngineEventFree(eEvent);

        //added this
        pw.close();
    }
}
```

2 class Java controller code:

```
public class EmotivDataClassificationFXMLController implements Initializable {

    static int totalTrials;
    static int movementTotalCount;
    static int steadyTotalCount;
    static String previousState = "Steady";
    static String currentState;
    private int dataRowCount = 1;

    TranslateTransition translateTransition1, translateTransition2, translateTransition3,
translateTransition4, translateTransition5;
    ScaleTransition scaleTransition1, scaleTransition2, scaleTransition3, scaleTransition4,
scaleTransition5, scaleTransition6;
    RotateTransition rotateTransition1, rotateTransition2, rotateTransition3,
rotateTransition4, rotateTransition5;
    ParallelTransition parallelTransition;
    static double animationDiration;
    Boolean animationEnd = true;

    @FXML
    private Label totalTrialsLabelTxt;
    @FXML
    private Label movementTotalCountLabelTxt;
    @FXML
    private Label steadyTotalCountLabelTxt;
    @FXML
    private Label currentStateLabelTxt;
    @FXML
    private TextField numberOfDataRowCount;
    @FXML
    private Button circleBtn1;
    @FXML
```

```
    private Button circleBtn2;
    @FXML
    private Button circleBtn3;
    @FXML
    private Button circleBtn4;
    @FXML
    private Button circleBtn5;
    @FXML
    private Button circleBtnCenter;

    @FXML
    private void handleButtonAction(ActionEvent event) {

        if (tryParseInt(numberOfDataRowCount.getText())) {
            dataRowCount = Integer.parseInt(numberOfDataRowCount.getText());
            if (dataRowCount > 10000) {
                animationDiration = 2;
            } else {
                animationDiration = 1;
            }
        } else {
            dataRowCount = 5000;
            animationDiration = 1;
        }
        totalTrials = 0;
        movementTotalCount = 0;
        steadyTotalCount = 0;

        translateTransition1 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn1);
        scaleTransition1 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn1);
        rotateTransition1 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn1);
        translateTransition2 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn2);
        scaleTransition2 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn2);
        rotateTransition2 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn2);
        translateTransition3 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn3);
        scaleTransition3 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn3);
        rotateTransition3 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn3);
```

```
        translateTransition4 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn4);
        scaleTransition4 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn4);
        rotateTransition4 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn4);
        translateTransition5 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn5);
        scaleTransition5 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn5);
        rotateTransition5 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn5);
        scaleTransition6 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtnCenter);
        final EmotiveDataSaveAndClassify classifyObj = new EmotiveDataSaveAndClassify();
        classifyObj.start();
    }

    boolean tryParseInt(String value) {
        try {
            Integer.parseInt(value);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    @Override
    public void initialize(URL url, ResourceBundle rb) {

        String image = EmotivDataClassification.class.getResource("Tennis-
Ball2.png").toExternalForm();
        circleBtn1.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
                + "-fx-background-repeat: no-repeat;");
        circleBtn2.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
                + "-fx-background-repeat: no-repeat;");
        circleBtn3.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
                + "-fx-background-repeat: no-repeat;");
        circleBtn4.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
```

```
            + "-fx-background-repeat: no-repeat;");
    circleBtn5.setStyle("-fx-background-image: url('" + image + "'); "
            + "-fx-background-position: center center; "
            + "-fx-background-size:cover;"
            + "-fx-background-repeat: no-repeat;");
    final Circle clip1 = new Circle(25, 25, 23);
    final Circle clip2 = new Circle(25, 25, 23);
    final Circle clip3 = new Circle(25, 25, 23);
    final Circle clip4 = new Circle(25, 25, 23);
    final Circle clip5 = new Circle(25, 25, 23);
    circleBtn1.setClip(clip1);
    circleBtn2.setClip(clip2);
    circleBtn3.setClip(clip3);
    circleBtn4.setClip(clip4);
    circleBtn5.setClip(clip5);

}

private void AnimationBall() {
    animationEnd = false;
    if (previousState == "HandMovement") {
        if (currentState == "Steady") {
            translateTransition1.setToX(0);
            translateTransition1.setToY(0);
            scaleTransition1.setToX(1);
            scaleTransition1.setToY(1);
            rotateTransition1.setFromAngle(180);
            rotateTransition1.setToAngle(315);

            translateTransition2.setToX(0);
            translateTransition2.setToY(0);
            scaleTransition2.setToX(1);
            scaleTransition2.setToY(1);
            rotateTransition2.setFromAngle(180);
            rotateTransition2.setToAngle(315);

            translateTransition3.setToX(0);
            translateTransition3.setToY(0);
            scaleTransition3.setToX(1);
            scaleTransition3.setToY(1);
            rotateTransition3.setFromAngle(180);
            rotateTransition3.setToAngle(315);

            translateTransition4.setToX(0);
            translateTransition4.setToY(0);
            scaleTransition4.setToX(1);
            scaleTransition4.setToY(1);
```

```
        rotateTransition4.setFromAngle(180);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(0);
        translateTransition5.setToY(0);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(180);
        rotateTransition5.setToAngle(315);

        scaleTransition6.setToX(1);
        scaleTransition6.setToY(1);

    } else {
        translateTransition1.setToX(144.626);
        translateTransition1.setToY(-83.5);
        scaleTransition1.setToX(1.5);
        scaleTransition1.setToY(1.5);
        rotateTransition1.setFromAngle(45);
        rotateTransition1.setToAngle(315);

        translateTransition2.setToX(-34.721);
        translateTransition2.setToY(-163.35);
        scaleTransition2.setToX(1.5);
        scaleTransition2.setToY(1.5);
        rotateTransition2.setFromAngle(45);
        rotateTransition2.setToAngle(315);

        translateTransition3.setToX(-166.085);
        translateTransition3.setToY(17.456);
        scaleTransition3.setToX(1.5);
        scaleTransition3.setToY(1.5);
        rotateTransition3.setFromAngle(45);
        rotateTransition3.setToAngle(315);

        translateTransition4.setToX(-67.925);
        translateTransition4.setToY(152.562);
        scaleTransition4.setToX(1.5);
        scaleTransition4.setToY(1.5);
        rotateTransition4.setFromAngle(45);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(124.105);
        translateTransition5.setToY(111.745);
        scaleTransition5.setToX(1.5);
        scaleTransition5.setToY(1.5);
        rotateTransition5.setFromAngle(45);
```

```
            rotateTransition5.setToAngle(315);

            scaleTransition6.setToX(5.5);
            scaleTransition6.setToY(5.5);
        }

    } else {
        if (currentState == "HandMovement") {
            translateTransition1.setToX(144.626);
            translateTransition1.setToY(-83.5);
            scaleTransition1.setToX(1.5);
            scaleTransition1.setToY(1.5);
            rotateTransition1.setFromAngle(180);
            rotateTransition1.setToAngle(315);

            translateTransition2.setToX(-34.721);
            translateTransition2.setToY(-163.35);
            scaleTransition2.setToX(1.5);
            scaleTransition2.setToY(1.5);
            rotateTransition2.setFromAngle(180);
            rotateTransition2.setToAngle(315);

            translateTransition3.setToX(-166.085);
            translateTransition3.setToY(17.456);
            scaleTransition3.setToX(1.5);
            scaleTransition3.setToY(1.5);
            rotateTransition3.setFromAngle(180);
            rotateTransition3.setToAngle(315);

            translateTransition4.setToX(-67.925);
            translateTransition4.setToY(152.562);
            scaleTransition4.setToX(1.5);
            scaleTransition4.setToY(1.5);
            rotateTransition4.setFromAngle(180);
            rotateTransition4.setToAngle(315);

            translateTransition5.setToX(124.105);
            translateTransition5.setToY(111.745);
            scaleTransition5.setToX(1.5);
            scaleTransition5.setToY(1.5);
            rotateTransition5.setFromAngle(180);
            rotateTransition5.setToAngle(315);

            scaleTransition6.setToX(5.5);
            scaleTransition6.setToY(5.5);

        } else {
```

```
            translateTransition1.setToX(0);
            translateTransition1.setToY(0);
            scaleTransition1.setToX(1);
            scaleTransition1.setToY(1);
            rotateTransition1.setFromAngle(-45);
            rotateTransition1.setToAngle(-270);

            translateTransition2.setToX(0);
            translateTransition2.setToY(0);
            scaleTransition2.setToX(1);
            scaleTransition2.setToY(1);
            rotateTransition2.setFromAngle(-45);
            rotateTransition2.setToAngle(-270);

            translateTransition3.setToX(0);
            translateTransition3.setToY(0);
            scaleTransition3.setToX(1);
            scaleTransition3.setToY(1);
            rotateTransition3.setFromAngle(-45);
            rotateTransition3.setToAngle(-270);

            translateTransition4.setToX(0);
            translateTransition4.setToY(0);
            scaleTransition4.setToX(1);
            scaleTransition4.setToY(1);
            rotateTransition4.setFromAngle(-45);
            rotateTransition4.setToAngle(-270);

            translateTransition5.setToX(0);
            translateTransition5.setToY(0);
            scaleTransition5.setToX(1);
            scaleTransition5.setToY(1);
            rotateTransition5.setFromAngle(-45);
            rotateTransition5.setToAngle(-270);

            scaleTransition6.setToX(1);
            scaleTransition6.setToY(1);
        }
    }
    parallelTransition = new ParallelTransition(translateTransition1,
translateTransition2, translateTransition3, translateTransition4, translateTransition5,
scaleTransition1, scaleTransition2, scaleTransition3, scaleTransition4, scaleTransition5,
scaleTransition6, rotateTransition1, rotateTransition2, rotateTransition3,
rotateTransition4, rotateTransition5);
    parallelTransition.play();
    parallelTransition.setOnFinished((e) -> {
        previousState = currentState;
```

```
        animationEnd = true;
      });
    }

    public class EmotiveDataSaveAndClassify extends Thread {

      public EmotiveDataSaveAndClassify() {
        setDaemon(true);
      }

      @Override
      public void run() {
        while (true && animationEnd == true) {
          Pointer eEvent = Edk.INSTANCE.IEE_EmoEngineEventCreate();
          Pointer eState = Edk.INSTANCE.IEE_EmoStateCreate();
          int counter = 0;
          int rightHandCurrentCount = 0;
          int steadyCurrentCount = 0;

          IntByReference userID = null;
          boolean ready = false;
          int state = 0;
          Edk.IEE_DataChannels_t dataChannel;

          userID = new IntByReference(0);

          if (Edk.INSTANCE.IEE_EngineConnect("Emotiv Systems-5") !=
EdkErrorCode.EDK_OK
                .ToInt()) {

            return;
          }


          while (true) {
            state = Edk.INSTANCE.IEE_EngineGetNextEvent(eEvent);

            // New event needs to be handled
            if (state == EdkErrorCode.EDK_OK.ToInt()) {
              int eventType = Edk.INSTANCE.IEE_EmoEngineEventGetType(eEvent);
              Edk.INSTANCE.IEE_EmoEngineEventGetUserId(eEvent, userID);

              // Log the EmoState if it has been updated
              if (eventType == Edk.IEE_Event_t.IEE_UserAdded.ToInt()) {
                if (userID != null) {

                  ready = true;
```

```
            }
          }
        } else if (state != EdkErrorCode.EDK_NO_EVENT.ToInt()) {

          break;
        }

        //if (true) {
        if (ready) {

          DoubleByReference alpha = new DoubleByReference(0);
          DoubleByReference low_beta = new DoubleByReference(0);
          DoubleByReference high_beta = new DoubleByReference(0);
          DoubleByReference gamma = new DoubleByReference(0);
          DoubleByReference theta = new DoubleByReference(0);

          for (int i = 4; i < 16; i++) {

            int result =
Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), i, theta, alpha, low_beta,
high_beta, gamma);
            // if (true) {//edited
            if (result == EdkErrorCode.EDK_OK.ToInt()) {


            }
            double Low_beta = low_beta.getValue();
            double High_beta = high_beta.getValue();
            double Theta = theta.getValue();
            double Alpha = alpha.getValue();
            double Gamma = gamma.getValue();

            //if (animationEnd == true)
            if (Alpha != 0) {
              //if (Alpha != 0 && animationEnd == true) {
              /*int randomNumber = totalTrials % 3 + totalTrials % 2;
              if (randomNumber == 1) {
                movementCurrentCount++;
              } else {
                steadyCurrentCount++;
              }*/

              if (Gamma <= 0.604081) {
                if (Gamma <= 0.066597) {
                  if (Gamma <= 0.023499) {
                    if (Alpha <= 0.037432) {
                      if (Alpha <= 0.037013) {
```

```
        if (Theta <= 0.05345) {
            if (Theta <= 0.044339) {
                if (Gamma <= 0.019915) {
                    if (Gamma <= 0.0194) {
                        if (Theta <= 0.026088) {
                            rightHandCurrentCount++;
                        } else if (Theta > 0.026088) {
                            if (Gamma <= 0.015285) {
                                steadyCurrentCount++;
                            } else if (Gamma > 0.015285) {
                                if (Theta <= 0.042352) {
                                    rightHandCurrentCount++;
                                } else if (Theta > 0.042352) {
                                    steadyCurrentCount++;
                                }
                            }
                        } else if (Gamma > 0.0194) {
                            steadyCurrentCount++;
                        }
                    } else if (Gamma > 0.019915) {
                        rightHandCurrentCount++;
                    }
                } else if (Theta > 0.044339) {
                    rightHandCurrentCount++;
                }
            } else if (Theta > 0.05345) {
                steadyCurrentCount++;
            }
        } else if (Alpha > 0.037013) {
            steadyCurrentCount++;
        }
    } else if (Alpha > 0.037432) {
        if (Theta <= 0.012762) {
            if (Theta <= 0.009156) {
                rightHandCurrentCount++;
            } else if (Theta > 0.009156) {
                steadyCurrentCount++;
            }
        } else if (Theta > 0.012762) {
            rightHandCurrentCount++;
        }
    }
} else if (Gamma > 0.023499) {
    if (Alpha <= 0.051453) {
        if (Low_beta <= 0.060763) {
            if (Theta <= 0.088441) {
                if (Alpha <= 0.026033) {
```

```
                    if (Alpha <= 0.026) {
                        if (Alpha <= 0.011678) {
                            if (Alpha <= 0.011437) {
                                steadyCurrentCount++;
                            } else if (Alpha > 0.011437) {
                                rightHandCurrentCount++;
                            }
                        } else if (Alpha > 0.011678) {
                            steadyCurrentCount++;
                        }
                    } else if (Alpha > 0.026) {
                        rightHandCurrentCount++;
                    }
                } else if (Alpha > 0.026033) {
                    steadyCurrentCount++;
                }
            } else if (Theta > 0.088441) {
                rightHandCurrentCount++;
            }
        } else if (Low_beta > 0.060763) {
            rightHandCurrentCount++;
        }
    } else if (Alpha > 0.051453) {
        if (Theta <= 0.038796) {
            steadyCurrentCount++;
        } else if (Theta > 0.038796) {
            rightHandCurrentCount++;
        }

    }
  }
} else if (Gamma > 0.066597) {
    steadyCurrentCount++;
  }
} else if (Gamma > 0.604081) {
  rightHandCurrentCount++;
}

counter++;


if (counter == dataRowCount) {
  //System.out.printf("\n\n");
  if (steadyCurrentCount >= rightHandCurrentCount) {
    movementTotalCount++;
    currentState = "HandMovement";
  } else {
```

```
                    steadyTotalCount++;
                    currentState = "Steady";
                }

                totalTrials++;

                counter = 0;
                rightHandCurrentCount = 0;
                steadyCurrentCount = 0;

            if (movementTotalCount > 0) {
            } else {
            }
            if (steadyTotalCount > 0) {
            } else {

            }
/*
            *** update ui
                */
                Platform.runLater(new Runnable() {
                    @Override
                    public void run() {

                        currentStateLabelTxt.setText("" + currentState);
                        if (movementTotalCount > 0) {
                            movementTotalCountLabelTxt.setText("" +
Math.round(((float) movementTotalCount / totalTrials) * 10000) / 100.0 + " %");
                        } else {
                            movementTotalCountLabelTxt.setText("0 %");
                        }
                        if (steadyTotalCount > 0) {
                            steadyTotalCountLabelTxt.setText("" + Math.round(((float)
steadyTotalCount / totalTrials) * 10000) / 100.0 + " %");
                        } else {
                            steadyTotalCountLabelTxt.setText("0 %");
                        }
                        totalTrialsLabelTxt.setText("" + totalTrials);

                        //animation method call
                        AnimationBall();

                    }
                });
            }
```

```
                    if (i == 5) {
                        i = 13;
                    }
                }
            }
        }
    }
            Edk.INSTANCE.IEE_EngineDisconnect();
            Edk.INSTANCE.IEE_EmoStateFree(eState);
            Edk.INSTANCE.IEE_EmoEngineEventFree(eEvent);

        }
    }

}
}
```

3 class Java controller code:

```
public class EmotivDataClassificationFXMLController implements Initializable {

    static int totalTrials;
    static int leftHandTotalCount;
    static int rightHandTotalCount;
    static int steadyTotalCount;
    static String previousState = "Steady";
    static String currentState;
    private int dataRowCount = 1;

    TranslateTransition translateTransition, translateTransition2, translateTransition3,
translateTransition4, translateTransition5;
    ScaleTransition scaleTransition, scaleTransition2, scaleTransition3, scaleTransition4,
scaleTransition5;
    RotateTransition rotateTransition, rotateTransition2, rotateTransition3,
rotateTransition4, rotateTransition5;
    ParallelTransition parallelTransition;
    static double animationDiration;
    Boolean animationEnd = true;

    @FXML
    private Label totalTrialsLabelTxt;
    @FXML
    private Label leftHandTotalCountLabelTxt;
    @FXML
    private Label rightHandTotalCountLabelTxt;
    @FXML
```

```
    private Label steadyTotalCountLabelTxt;
    @FXML
    private Label currentStateLabelTxt;
    @FXML
    private TextField numberOfDataRowCount;
    @FXML
    private Button circleBtn;
    @FXML
    private Button circleBtn2;
    @FXML
    private Button circleBtn3;
    @FXML
    private Button circleBtn4;
    @FXML
    private Button circleBtn5;

    @FXML
    private void handleButtonAction(ActionEvent event) {

        if (tryParseInt(numberOfDataRowCount.getText())) {
            dataRowCount = Integer.parseInt(numberOfDataRowCount.getText());
            if (dataRowCount > 10000) {
                animationDiration = .2;
            } else {
                animationDiration = .2;
            }
        } else {
            dataRowCount = 5000;
            animationDiration = .2;
        }
        totalTrials = 0;
        leftHandTotalCount = 0;
        rightHandTotalCount = 0;
        steadyTotalCount = 0;

        translateTransition = new TranslateTransition(Duration.seconds(animationDiration),
circleBtn);
        scaleTransition = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn);
        rotateTransition = new RotateTransition(Duration.seconds(animationDiration),
circleBtn);
        translateTransition2 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn2);
        scaleTransition2 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn2);
        rotateTransition2 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn2);
```

```
        translateTransition3 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn3);
        scaleTransition3 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn3);
        rotateTransition3 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn3);
        translateTransition4 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn4);
        scaleTransition4 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn4);
        rotateTransition4 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn4);
        translateTransition5 = new
TranslateTransition(Duration.seconds(animationDiration), circleBtn5);
        scaleTransition5 = new ScaleTransition(Duration.seconds(animationDiration),
circleBtn5);
        rotateTransition5 = new RotateTransition(Duration.seconds(animationDiration),
circleBtn5);
        final EmotiveDataSaveAndClassify classifyObj = new EmotiveDataSaveAndClassify();
        classifyObj.start();
    }

    boolean tryParseInt(String value) {
        try {
            Integer.parseInt(value);
            return true;
        } catch (NumberFormatException e) {
            return false;
        }
    }

    @Override
    public void initialize(URL url, ResourceBundle rb) {

        String image = EmotivDataClassification.class.getResource("Tennis-
Ball2.png").toExternalForm();
        circleBtn.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
                + "-fx-background-repeat: no-repeat;");
        circleBtn2.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
                + "-fx-background-repeat: no-repeat;");
        circleBtn3.setStyle("-fx-background-image: url('" + image + "'); "
                + "-fx-background-position: center center; "
                + "-fx-background-size:cover;"
```

```
            + "-fx-background-repeat: no-repeat;");
    circleBtn4.setStyle("-fx-background-image: url('" + image + "'); "
            + "-fx-background-position: center center; "
            + "-fx-background-size:cover;"
            + "-fx-background-repeat: no-repeat;");
    circleBtn5.setStyle("-fx-background-image: url('" + image + "'); "
            + "-fx-background-position: center center; "
            + "-fx-background-size:cover;"
            + "-fx-background-repeat: no-repeat;");
    final Circle clip = new Circle(25, 25, 23);
    final Circle clip2 = new Circle(25, 25, 23);
    final Circle clip3 = new Circle(25, 25, 23);
    final Circle clip4 = new Circle(25, 25, 23);
    final Circle clip5 = new Circle(25, 25, 23);
    circleBtn.setClip(clip);
    circleBtn2.setClip(clip2);
    circleBtn3.setClip(clip3);
    circleBtn4.setClip(clip4);
    circleBtn5.setClip(clip5);

}

private void AnimationBall() {
    animationEnd = false;
    if (previousState == "Left Hand") {
        if (currentState == "Right Hand") {
            translateTransition.setToX(100);
            scaleTransition.setToX(1);
            scaleTransition.setToY(1);
            rotateTransition.setFromAngle(180);
            rotateTransition.setToAngle(315);

            translateTransition2.setToX(120);
            translateTransition2.setToY(-35);
            scaleTransition2.setToX(1);
            scaleTransition2.setToY(1);
            rotateTransition2.setFromAngle(180);
            rotateTransition2.setToAngle(315);

            translateTransition3.setToX(163);
            translateTransition3.setToY(-70);
            scaleTransition3.setToX(1);
            scaleTransition3.setToY(1);
            rotateTransition3.setFromAngle(180);
            rotateTransition3.setToAngle(315);

            translateTransition4.setToX(120);
```

```
        translateTransition4.setToY(35);
        scaleTransition4.setToX(1);
        scaleTransition4.setToY(1);
        rotateTransition4.setFromAngle(180);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(163);
        translateTransition5.setToY(70);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(180);
        rotateTransition5.setToAngle(315);

    } else if (currentState == "Left Hand") {
        translateTransition.setToX(-100);
        scaleTransition.setToX(1.5);
        scaleTransition.setToY(1.5);
        rotateTransition.setFromAngle(-45);
        rotateTransition.setToAngle(-270);

        translateTransition2.setToX(-120);
        translateTransition2.setToY(-35);
        scaleTransition2.setToX(1.5);
        scaleTransition2.setToY(1.5);
        rotateTransition2.setFromAngle(-45);
        rotateTransition2.setToAngle(-270);

        translateTransition3.setToX(-163);
        translateTransition3.setToY(-70);
        scaleTransition3.setToX(1.5);
        scaleTransition3.setToY(1.5);
        rotateTransition3.setFromAngle(-45);
        rotateTransition3.setToAngle(-270);

        translateTransition4.setToX(-120);
        translateTransition4.setToY(35);
        scaleTransition4.setToX(1.5);
        scaleTransition4.setToY(1.5);
        rotateTransition4.setFromAngle(-45);
        rotateTransition4.setToAngle(-270);

        translateTransition5.setToX(-163);
        translateTransition5.setToY(70);
        scaleTransition5.setToX(1.5);
        scaleTransition5.setToY(1.5);
        rotateTransition5.setFromAngle(-45);
        rotateTransition5.setToAngle(-270);
```

```
    } else {
        translateTransition.setToX(0);
        translateTransition3.setToY(0);
        scaleTransition.setToX(1);
        scaleTransition.setToY(1);
        rotateTransition.setFromAngle(45);
        rotateTransition.setToAngle(315);

        translateTransition2.setToX(0);
        translateTransition2.setToY(0);
        scaleTransition2.setToX(1);
        scaleTransition2.setToY(1);
        rotateTransition2.setFromAngle(45);
        rotateTransition2.setToAngle(315);

        translateTransition3.setToX(0);
        scaleTransition3.setToX(1);
        scaleTransition3.setToY(1);
        rotateTransition3.setFromAngle(45);
        rotateTransition3.setToAngle(315);

        translateTransition4.setToX(0);
        translateTransition4.setToY(0);
        scaleTransition4.setToX(1);
        scaleTransition4.setToY(1);
        rotateTransition4.setFromAngle(45);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(0);
        translateTransition5.setToY(0);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(45);
        rotateTransition5.setToAngle(315);
    }

} else if (previousState == "Right Hand") {
    if (currentState == "Left Hand") {
        translateTransition.setToX(-100);
        scaleTransition.setToX(1);
        scaleTransition.setToY(1);
        rotateTransition.setFromAngle(180);
        rotateTransition.setToAngle(315);

        translateTransition2.setToX(-120);
        translateTransition2.setToY(-35);
```

```
    scaleTransition2.setToX(1);
    scaleTransition2.setToY(1);
    rotateTransition2.setFromAngle(180);
    rotateTransition2.setToAngle(315);

    translateTransition3.setToX(-163);
    translateTransition3.setToY(-70);
    scaleTransition3.setToX(1);
    scaleTransition3.setToY(1);
    rotateTransition3.setFromAngle(180);
    rotateTransition3.setToAngle(315);

    translateTransition4.setToX(-120);
    translateTransition4.setToY(35);
    scaleTransition4.setToX(1);
    scaleTransition4.setToY(1);
    rotateTransition4.setFromAngle(180);
    rotateTransition4.setToAngle(315);

    translateTransition5.setToX(-163);
    translateTransition5.setToY(70);
    scaleTransition5.setToX(1);
    scaleTransition5.setToY(1);
    rotateTransition5.setFromAngle(180);
    rotateTransition5.setToAngle(315);

} else if (currentState == "Steady") {
    translateTransition.setToX(0);
    scaleTransition.setToX(1);
    scaleTransition.setToY(1);
    rotateTransition.setFromAngle(45);
    rotateTransition.setToAngle(315);

    translateTransition2.setToX(0);
    translateTransition2.setToY(0);
    scaleTransition2.setToX(1);
    scaleTransition2.setToY(1);
    rotateTransition2.setFromAngle(45);
    rotateTransition2.setToAngle(315);

    translateTransition3.setToX(0);
    translateTransition3.setToY(0);
    scaleTransition3.setToX(1);
    scaleTransition3.setToY(1);
    rotateTransition3.setFromAngle(45);
    rotateTransition3.setToAngle(315);
```

```
        translateTransition4.setToX(0);
        translateTransition4.setToY(0);
        scaleTransition4.setToX(1);
        scaleTransition4.setToY(1);
        rotateTransition4.setFromAngle(45);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(0);
        translateTransition5.setToY(0);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(45);
        rotateTransition5.setToAngle(315);
    } else {
        translateTransition.setToX(100);
        scaleTransition.setToX(1.5);
        scaleTransition.setToY(1.5);
        rotateTransition.setFromAngle(-45);
        rotateTransition.setToAngle(-270);

        translateTransition2.setToX(120);
        translateTransition2.setToY(-35);
        scaleTransition2.setToX(1.5);
        scaleTransition2.setToY(1.5);
        rotateTransition2.setFromAngle(-45);
        rotateTransition2.setToAngle(-270);

        translateTransition3.setToX(163);
        translateTransition3.setToY(-70);
        scaleTransition3.setToX(1.5);
        scaleTransition3.setToY(1.5);
        rotateTransition3.setFromAngle(-45);
        rotateTransition3.setToAngle(-270);

        translateTransition4.setToX(120);
        translateTransition4.setToY(35);
        scaleTransition4.setToX(1.5);
        scaleTransition4.setToY(1.5);
        rotateTransition4.setFromAngle(-45);
        rotateTransition4.setToAngle(-270);

        translateTransition5.setToX(163);
        translateTransition5.setToY(70);
        scaleTransition5.setToX(1.5);
        scaleTransition5.setToY(1.5);
        rotateTransition5.setFromAngle(-45);
        rotateTransition5.setToAngle(-270);
```

```
    }
} else {

    if (currentState == "Left Hand") {
        translateTransition.setToX(-100);
        scaleTransition.setToX(1);
        scaleTransition.setToY(1);
        rotateTransition.setFromAngle(180);
        rotateTransition.setToAngle(315);

        translateTransition2.setToX(-120);
        translateTransition2.setToY(-35);
        scaleTransition2.setToX(1);
        scaleTransition2.setToY(1);
        rotateTransition2.setFromAngle(180);
        rotateTransition2.setToAngle(315);

        translateTransition3.setToX(-163);
        translateTransition3.setToY(-70);
        scaleTransition3.setToX(1);
        scaleTransition3.setToY(1);
        rotateTransition3.setFromAngle(180);
        rotateTransition3.setToAngle(315);

        translateTransition4.setToX(-120);
        translateTransition4.setToY(35);
        scaleTransition4.setToX(1);
        scaleTransition4.setToY(1);
        rotateTransition4.setFromAngle(180);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(-163);
        translateTransition5.setToY(70);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(180);
        rotateTransition5.setToAngle(315);

    } else if (currentState == "Right Hand") {
        translateTransition.setToX(100);
        scaleTransition.setToX(1);
        scaleTransition.setToY(1);
        rotateTransition.setFromAngle(45);
        rotateTransition.setToAngle(315);

        translateTransition2.setToX(120);
        translateTransition2.setToY(-35);
```

```
        scaleTransition2.setToX(1);
        scaleTransition2.setToY(1);
        rotateTransition2.setFromAngle(45);
        rotateTransition2.setToAngle(315);

        translateTransition3.setToX(163);
        translateTransition3.setToY(-70);
        scaleTransition3.setToX(1);
        scaleTransition3.setToY(1);
        rotateTransition3.setFromAngle(45);
        rotateTransition3.setToAngle(315);

        translateTransition4.setToX(120);
        translateTransition4.setToY(35);
        scaleTransition4.setToX(1);
        scaleTransition4.setToY(1);
        rotateTransition4.setFromAngle(45);
        rotateTransition4.setToAngle(315);

        translateTransition5.setToX(163);
        translateTransition5.setToY(70);
        scaleTransition5.setToX(1);
        scaleTransition5.setToY(1);
        rotateTransition5.setFromAngle(45);
        rotateTransition5.setToAngle(315);

    } else {
        translateTransition.setToX(0);
        scaleTransition.setToX(1);
        scaleTransition.setToY(1);
        rotateTransition.setFromAngle(-45);
        rotateTransition.setToAngle(-270);

        translateTransition2.setToX(0);
        translateTransition2.setToY(0);
        scaleTransition2.setToX(1);
        scaleTransition2.setToY(1);
        rotateTransition2.setFromAngle(-45);
        rotateTransition2.setToAngle(-270);

        translateTransition3.setToX(0);
        translateTransition3.setToY(0);
        scaleTransition3.setToX(1);
        scaleTransition3.setToY(1);
        rotateTransition3.setFromAngle(-45);
        rotateTransition3.setToAngle(-270);
```

```
            translateTransition4.setToX(0);
            translateTransition4.setToY(0);
            scaleTransition4.setToX(1);
            scaleTransition4.setToY(1);
            rotateTransition4.setFromAngle(-45);
            rotateTransition4.setToAngle(-270);

            translateTransition5.setToX(0);
            translateTransition5.setToY(0);
            scaleTransition5.setToX(1);
            scaleTransition5.setToY(1);
            rotateTransition5.setFromAngle(-45);
            rotateTransition5.setToAngle(-270);

        }

    }
    parallelTransition = new ParallelTransition(translateTransition, translateTransition2,
translateTransition3, translateTransition4, translateTransition5, scaleTransition,
scaleTransition2, scaleTransition3, scaleTransition4, scaleTransition5, rotateTransition,
rotateTransition2, rotateTransition3, rotateTransition4, rotateTransition5);
    parallelTransition.play();
    parallelTransition.setOnFinished((e) -> {
      previousState = currentState;
      animationEnd = true;
    });
  }

  public class EmotiveDataSaveAndClassify extends Thread {

    public EmotiveDataSaveAndClassify() {
      setDaemon(true);
    }

    @Override
    public void run() {
      while (true && animationEnd == true) {
        Pointer eEvent = Edk.INSTANCE.IEE_EmoEngineEventCreate();
        Pointer eState = Edk.INSTANCE.IEE_EmoStateCreate();
        int counter = 0;
        int leftHandCurrentCount = 0;
        int rightHandCurrentCount = 0;
        int leftFootCurrentCount = 0;
        int rightFootCurrentCount = 0;
        int steadyCurrentCount = 0;
        int[] counterArray = new int[4];
```

```
        IntByReference userID = null;
        boolean ready = false;
        int state = 0;

        Edk.IEE_DataChannels_t dataChannel;

        userID = new IntByReference(0);

        if (Edk.INSTANCE.IEE_EngineConnect("Emotiv Systems-5…") !=
EdkErrorCode.EDK_OK
              .ToInt()) {
           return;
        }


        while (true) {
           state = Edk.INSTANCE.IEE_EngineGetNextEvent(eEvent);

           // New event needs to be handled
           if (state == EdkErrorCode.EDK_OK.ToInt()) {
              int eventType = Edk.INSTANCE.IEE_EmoEngineEventGetType(eEvent);
              Edk.INSTANCE.IEE_EmoEngineEventGetUserId(eEvent, userID);

              // Log the EmoState if it has been updated
              if (eventType == Edk.IEE_Event_t.IEE_UserAdded.ToInt()) {
                 if (userID != null) {
                    ready = true;
                 }
              }
           } else if (state != EdkErrorCode.EDK_NO_EVENT.ToInt()) {
              break;
           }

           if (true) {
           // if (ready) {

              DoubleByReference alpha = new DoubleByReference(0);
              DoubleByReference low_beta = new DoubleByReference(0);
              DoubleByReference high_beta = new DoubleByReference(0);
              DoubleByReference gamma = new DoubleByReference(0);
              DoubleByReference theta = new DoubleByReference(0);

              for (int i = 4; i < 16; i++) {

                 int result =
Edk.INSTANCE.IEE_GetAverageBandPowers(userID.getValue(), i, theta, alpha, low_beta,
high_beta, gamma);
```

```
 if (true) {//edited
// if (result == EdkErrorCode.EDK_OK.ToInt()) {

 }
 double Low_beta = low_beta.getValue();
 double High_beta = high_beta.getValue();
 double Theta = theta.getValue();
 double Alpha = alpha.getValue();
 double gammaValue = gamma.getValue();
// if (gammaValue != 0 && animationEnd == true)
  if (animationEnd == true)
 {
    int randomNumber = totalTrials % 3 + totalTrials %2;
    if (randomNumber == 1) {
       leftHandCurrentCount++;
    } else if (randomNumber == 2) {
       rightHandCurrentCount++;
    } else if (randomNumber == 3) {
       steadyCurrentCount++;
    }

   /* if (gammaValue < 0.005072252186533857) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.006802435776292941) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.007344306109092959) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.0078041735816908875) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.008526992578365651) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.009021998091728513) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.009520290673810087) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.010670782884125951) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.010812544879293156) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.011040449076516688) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.011147504315911534) {
       rightHandCurrentCount++;
    } else if (gammaValue < 0.011155805801253376) {
       leftHandCurrentCount++;
    } else if (gammaValue < 0.01165783798335064) {
       rightHandCurrentCount++;
```

```
} else if (gammaValue < 0.011750274765603055) {
   steadyCurrentCount++;
} else if (gammaValue < 0.011991878588902521) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.012150917392372484) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.012739205530378113) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.013320286448680468) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.01386473876277632) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.014286295047601006) {
   steadyCurrentCount++;
} else if (gammaValue < 0.014958639705651696) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.015153917681799963) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.015402596002847931) {
   steadyCurrentCount++;
} else if (gammaValue < 0.015936899460652493) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.01620770094399511) {
   steadyCurrentCount++;
} else if (gammaValue < 0.016523001660879298) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.01696044086602634) {
   steadyCurrentCount++;
} else if (gammaValue < 0.01699304982145982) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.017933169092696676) {
   steadyCurrentCount++;
} else if (gammaValue < 0.01817769707796915) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.01885952205380844) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.018944417556311974) {
   steadyCurrentCount++;
} else if (gammaValue < 0.019039589546881154) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.019612633620094598) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.02071079037965929) {
   steadyCurrentCount++;
} else if (gammaValue < 0.020823398700207045) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.021234442102609233) {
```

```
    rightHandCurrentCount++;
} else if (gammaValue < 0.021268840442077885) {
    steadyCurrentCount++;
} else if (gammaValue < 0.021285075906355123) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.021340882637984337) {
    steadyCurrentCount++;
} else if (gammaValue < 0.02162444956634521) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.021786308247955313) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.021954436427623562) {
    steadyCurrentCount++;
} else if (gammaValue < 0.02208239852896844) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.022310976612749826) {
    steadyCurrentCount++;
} else if (gammaValue < 0.022685303382698682) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.022900693067599596) {
    steadyCurrentCount++;
} else if (gammaValue < 0.023348659262292498) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.024125217910867837) {
    steadyCurrentCount++;
} else if (gammaValue < 0.024768287788538616) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.025351807385968067) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.025655822762809935) {
    steadyCurrentCount++;
} else if (gammaValue < 0.026191463318499253) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.026791605072725294) {
    steadyCurrentCount++;
} else if (gammaValue < 0.027179074962124697) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.027476623615907324) {
    steadyCurrentCount++;
} else if (gammaValue < 0.028000412347286008) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.029005093655770604) {
    steadyCurrentCount++;
} else if (gammaValue < 0.029479309205240563) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.029737780599615918) {
    steadyCurrentCount++;
```

```
} else if (gammaValue < 0.030113670801496743) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.030970698297651773) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.0316069039976392) {
    steadyCurrentCount++;
} else if (gammaValue < 0.0328136452998908) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.035544500950707866) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.03579102175854096) {
    steadyCurrentCount++;
} else if (gammaValue < 0.036049090396196456) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.03638206594674854) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.03765521692708187) {
    steadyCurrentCount++;
} else if (gammaValue < 0.04029280602510216) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.04309561184197798) {
    steadyCurrentCount++;
} else if (gammaValue < 0.05004540527398328) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.059068932996450045) {
    steadyCurrentCount++;
} else if (gammaValue < 0.06320789165269472) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.06474124007617726) {
    steadyCurrentCount++;
} else if (gammaValue < 0.0676555998869634) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.07116963177846206) {
    steadyCurrentCount++;
} else if (gammaValue < 0.07292500095788053) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.07622300590880797) {
    steadyCurrentCount++;
} else if (gammaValue < 0.07710282602956617) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.07795957019168208) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.07908837506278416) {
    steadyCurrentCount++;
} else if (gammaValue < 0.07995702312978004) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.0814282847114639) {
```

```
    steadyCurrentCount++;
} else if (gammaValue < 0.08195978129664062) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.08969383346951286) {
    steadyCurrentCount++;
} else if (gammaValue < 0.09145665957918239) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.09262763018036556) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.09348325773255754) {
    steadyCurrentCount++;
} else if (gammaValue < 0.09632577804379139) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.09786847059706164) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.0997939467162787) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.10182148510482092) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.1050544544422882) {
    steadyCurrentCount++;
} else if (gammaValue < 0.10896508531765131) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.11056277856268248) {
    steadyCurrentCount++;
} else if (gammaValue < 0.11203891008740897) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.11403904673348325) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.12031270333032784) {
    steadyCurrentCount++;
} else if (gammaValue < 0.12183631929758607) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.12354339779135315) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.13248972576382606) {
    steadyCurrentCount++;
} else if (gammaValue < 0.13439838778141003) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.13842375525552952) {
    steadyCurrentCount++;
} else if (gammaValue < 0.1411569328995308) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.1428119238468402) {
    steadyCurrentCount++;
} else if (gammaValue < 0.1476058286936982) {
    leftHandCurrentCount++;
```

```
} else if (gammaValue < 0.15245517258583813) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.153688301400205) {
   steadyCurrentCount++;
} else if (gammaValue < 0.15487761980017684) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.15526332341424184) {
   steadyCurrentCount++;
} else if (gammaValue < 0.15554346705312533) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.15619691463725266) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.16037518224627173) {
   steadyCurrentCount++;
} else if (gammaValue < 0.16492719889781604) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.16531235358556554) {
   steadyCurrentCount++;
} else if (gammaValue < 0.16585942265961107) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.16658901979243687) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.17242839172570126) {
   steadyCurrentCount++;
} else if (gammaValue < 0.17342922204078579) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.1926432165961567) {
   steadyCurrentCount++;
} else if (gammaValue < 0.19314127554962923) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.19333704698737247) {
   steadyCurrentCount++;
} else if (gammaValue < 0.1935910242224641) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.1979057260933137) {
   steadyCurrentCount++;
} else if (gammaValue < 0.20056120857864018) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.20255083779270244) {
   steadyCurrentCount++;
} else if (gammaValue < 0.20348271768520726) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.20434669723516816) {
   steadyCurrentCount++;
} else if (gammaValue < 0.20537698849564384) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.20942762004717644) {
```

```
    steadyCurrentCount++;
} else if (gammaValue < 0.2102683237822714) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.2149302693766609) {
    steadyCurrentCount++;
} else if (gammaValue < 0.2188419625538675) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.21980934268743327) {
    steadyCurrentCount++;
} else if (gammaValue < 0.2279683430285896) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.23020702607641552) {
    steadyCurrentCount++;
} else if (gammaValue < 0.23107947686262398) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.23133867226986451) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.23357237264134856) {
    steadyCurrentCount++;
} else if (gammaValue < 0.23389070456348046) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.23627723200483147) {
    steadyCurrentCount++;
} else if (gammaValue < 0.239736613207093) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.24197057020228402) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.25079545217572297) {
    steadyCurrentCount++;
} else if (gammaValue < 0.2515672216302177) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.25310032450990533) {
    steadyCurrentCount++;
} else if (gammaValue < 0.25424634699584786) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.25478202781879566) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.26193253615737766) {
    steadyCurrentCount++;
} else if (gammaValue < 0.26313080236562686) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.26564125930257265) {
    steadyCurrentCount++;
} else if (gammaValue < 0.26740400928327) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.2803646016663129) {
    steadyCurrentCount++;
```

```
} else if (gammaValue < 0.2816060411107062) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.2862086044085886) {
   steadyCurrentCount++;
} else if (gammaValue < 0.29014434714664317) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.3206160835344246) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3325371507418124) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.33734599857778214) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3380892693783284) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.34234136783155383) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.34783889021798964) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3529184230094095) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.3542617162991796) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.37656662588952816) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3813789202247845) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.38220150143095466) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.3868068370993887) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3880642630301411) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.3919765516852619) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.3932029396358304) {
   steadyCurrentCount++;
} else if (gammaValue < 0.3956345236914768) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.39893310760491263) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.40051173716669725) {
   steadyCurrentCount++;
} else if (gammaValue < 0.4022035296197242) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.40301452909545205) {
   steadyCurrentCount++;
} else if (gammaValue < 0.40660943731096155) {
```

```
    rightHandCurrentCount++;
} else if (gammaValue < 0.4128803705629316) {
    steadyCurrentCount++;
} else if (gammaValue < 0.4295088384686942) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.43235685485623576) {
    steadyCurrentCount++;
} else if (gammaValue < 0.4340517429495843) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.4352361859246362) {
    steadyCurrentCount++;
} else if (gammaValue < 0.43580942289326197) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.4362233338493132) {
    steadyCurrentCount++;
} else if (gammaValue < 0.43718455585445326) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.43892798958965085) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.44029878184360727) {
    steadyCurrentCount++;
} else if (gammaValue < 0.44094882052974493) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.44439768299793553) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.4614349145834983) {
    steadyCurrentCount++;
} else if (gammaValue < 0.4630126734242713) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.46394317616528985) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.4659514756601044) {
    steadyCurrentCount++;
} else if (gammaValue < 0.4690003843918087) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.47027362922076316) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.4765875330524011) {
    steadyCurrentCount++;
} else if (gammaValue < 0.47857964308470613) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.4831127726949642) {
    steadyCurrentCount++;
} else if (gammaValue < 0.4858953451879233) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.4969484461234161) {
    rightHandCurrentCount++;
```

```
} else if (gammaValue < 0.4999692708707191) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5008163339537937) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.5154167941258672) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.5201470607725998) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5232520999501722) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.524993451557382) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5262421306694747) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.5328627559125625) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.534986188724327) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5397372209509468) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.5430614238726027) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.5470157661782813) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5523586454498397) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.5605223142527574) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.5700247409908956) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.5776031896681442) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.5810737527766048) {
   steadyCurrentCount++;
} else if (gammaValue < 0.5900862744014992) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.600077422726397) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.6067623568259258) {
   steadyCurrentCount++;
} else if (gammaValue < 0.6090318773951866) {
   rightHandCurrentCount++;
} else if (gammaValue < 0.6216809311899626) {
   steadyCurrentCount++;
} else if (gammaValue < 0.6311837170194461) {
   leftHandCurrentCount++;
} else if (gammaValue < 0.6421437104310829) {
```

```
    steadyCurrentCount++;
} else if (gammaValue < 0.6514153323805898) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.6549115974640733) {
    steadyCurrentCount++;
} else if (gammaValue < 0.6663751319310511) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.6724174722979607) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.6778124111821147) {
    steadyCurrentCount++;
} else if (gammaValue < 0.6821902619805593) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.6847688836588286) {
    steadyCurrentCount++;
} else if (gammaValue < 0.7116152570023504) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.7236463568858219) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.7281569583816038) {
    steadyCurrentCount++;
} else if (gammaValue < 0.779961252359161) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.7818680402145304) {
    steadyCurrentCount++;
} else if (gammaValue < 0.7899225733346353) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.7971831782046896) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.8070272849893303) {
    steadyCurrentCount++;
} else if (gammaValue < 0.8400382135348339) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.8475190665103266) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.8803820539571087) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.8876175354726429) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.8937337746685313) {
    steadyCurrentCount++;
} else if (gammaValue < 0.9106365491972208) {
    rightHandCurrentCount++;
} else if (gammaValue < 0.9278030114571767) {
    leftHandCurrentCount++;
} else if (gammaValue < 0.9503016835473239) {
    rightHandCurrentCount++;
```

```
} else if (gammaValue < 1.0284064837544564) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.0360760886862246) {
   steadyCurrentCount++;
} else if (gammaValue < 1.0383979069954625) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.046048168081987) {
   steadyCurrentCount++;
} else if (gammaValue < 1.0505738182651299) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.0519715424919116) {
   steadyCurrentCount++;
} else if (gammaValue < 1.0558507928156722) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.062144739564013) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.0658363372987623) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.0678881034596777) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.0692079969780053) {
   steadyCurrentCount++;
} else if (gammaValue < 1.0795968788264139) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.0846926424458856) {
   steadyCurrentCount++;
} else if (gammaValue < 1.1007608683493753) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.1071168948938621) {
   steadyCurrentCount++;
} else if (gammaValue < 1.1130322085211835) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.1204281844249349) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.1390595427267607) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.1547152653287238) {
   steadyCurrentCount++;
} else if (gammaValue < 1.1576114006828218) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.161296509532367) {
   steadyCurrentCount++;
} else if (gammaValue < 1.1623661527304425) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.1675927849350396) {
   steadyCurrentCount++;
} else if (gammaValue < 1.1731163822431006) {
```

```
    leftHandCurrentCount++;
} else if (gammaValue < 1.18449215775244) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.200744961637419) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2013199537854193) {
    steadyCurrentCount++;
} else if (gammaValue < 1.2029240996255117) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.204693026682032) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2083453970113645) {
    steadyCurrentCount++;
} else if (gammaValue < 1.2137996040635781) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.221623886081066) {
    steadyCurrentCount++;
} else if (gammaValue < 1.227601742673396) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.2329670265145234) {
    steadyCurrentCount++;
} else if (gammaValue < 1.236588290927459) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2371313895176437) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.2385479941902178) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2407156017141259) {
    steadyCurrentCount++;
} else if (gammaValue < 1.2421467206661294) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.2435694742067391) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2464232924924552) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.2513639383081956) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2564772807171485) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.2609192821528517) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.2728709933185645) {
    steadyCurrentCount++;
} else if (gammaValue < 1.2886317594580186) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.297829557283559) {
    rightHandCurrentCount++;
```

```
} else if (gammaValue < 1.3045735010669484) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.3069269989027545) {
   steadyCurrentCount++;
} else if (gammaValue < 1.3119969256902722) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.3145642425428643) {
   steadyCurrentCount++;
} else if (gammaValue < 1.3266780960712503) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.3377991655230916) {
   steadyCurrentCount++;
} else if (gammaValue < 1.3408638079267385) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.3524682414634759) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.362259956338645) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.3796663876317277) {
   steadyCurrentCount++;
} else if (gammaValue < 1.4012392235778242) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.4136951266241895) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.4184684231133087) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.4198843566442236) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.4238171317650958) {
   steadyCurrentCount++;
} else if (gammaValue < 1.4339883568530558) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.4550421669821247) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.4709071824640771) {
   steadyCurrentCount++;
} else if (gammaValue < 1.4791114626173414) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.4845591206084994) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.493285939327673) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.4990160216773338) {
   steadyCurrentCount++;
} else if (gammaValue < 1.5005766895537296) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.5359606520378097) {
```

```
    leftHandCurrentCount++;
} else if (gammaValue < 1.5394225213873811) {
    steadyCurrentCount++;
} else if (gammaValue < 1.5515105568990983) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.5641899711604128) {
    steadyCurrentCount++;
} else if (gammaValue < 1.572204847499504) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.5906090509792925) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.5983027745652731) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.6050214408751633) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.6146257813625664) {
    steadyCurrentCount++;
} else if (gammaValue < 1.622211155330242) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.6390029440627631) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.6512020680013235) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.6629621528683884) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.6781861424840656) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.6841910267876945) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.70857396082278) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.7579662284564637) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.7814669769396385) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.7948975754859333) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.8129449658071457) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.8292378180834432) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.8352173988016087) {
    leftHandCurrentCount++;
} else if (gammaValue < 1.8384496374373174) {
    rightHandCurrentCount++;
} else if (gammaValue < 1.849226662578412) {
    leftHandCurrentCount++;
```

```
} else if (gammaValue < 1.8577898970561806) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.8771379626831408) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.921183250260209) {
   rightHandCurrentCount++;
} else if (gammaValue < 1.9507893506248652) {
   leftHandCurrentCount++;
} else if (gammaValue < 1.9570393055768123) {
   steadyCurrentCount++;
} else if (gammaValue < 1.981145455050519) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.002895663580559) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.015227419116271) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.024128548594833) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.083508240416153) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.1200981204487177) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.1265640220435924) {
   steadyCurrentCount++;
} else if (gammaValue < 2.1516695748807324) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.1687369410061725) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.2230260965459627) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.2902409540148505) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.406263957983401) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.4294700466392865) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.531988237302973) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.608912434711395) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.6442516544495653) {
   steadyCurrentCount++;
} else if (gammaValue < 2.6526302859593764) {
   rightHandCurrentCount++;
} else if (gammaValue < 2.6802308183354473) {
   leftHandCurrentCount++;
} else if (gammaValue < 2.7334423663038727) {
```

```
    rightHandCurrentCount++;
} else if (gammaValue < 2.7880437799360944) {
    leftHandCurrentCount++;
} else if (gammaValue < 2.887803499515223) {
    rightHandCurrentCount++;
} else if (gammaValue < 2.909292771304707) {
    leftHandCurrentCount++;
} else if (gammaValue < 2.9488946895530876) {
    rightHandCurrentCount++;
} else if (gammaValue < 2.9932128599931302) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.0262578169168144) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.0537970756868957) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.0949419657826707) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.1500692372513983) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.386840710320314) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.4038682337470023) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.413071671472947) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.4615248166811847) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.6087808853025827) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.6442932505440147) {
    leftHandCurrentCount++;
} else if (gammaValue < 3.746732878263457) {
    rightHandCurrentCount++;
} else if (gammaValue < 3.7593530760893086) {
    leftHandCurrentCount++;
} else if (gammaValue < 4.249234417484567) {
    rightHandCurrentCount++;
} else if (gammaValue < 4.259044375636631) {
    leftHandCurrentCount++;
} else if (gammaValue < 4.283790772024654) {
    rightHandCurrentCount++;
} else if (gammaValue < 4.326160058258664) {
    leftHandCurrentCount++;
} else if (gammaValue < 6.929004408505042) {
    rightHandCurrentCount++;
} else if (gammaValue < 7.394212053007484) {
    leftHandCurrentCount++;
```

```
} else if (gammaValue < 8.71342459835991) {
   rightHandCurrentCount++;
} else if (gammaValue < 9.075782301814105) {
   steadyCurrentCount++;
} else if (gammaValue < 9.29803679775884) {
   leftHandCurrentCount++;
} else if (gammaValue < 9.729355964981744) {
   rightHandCurrentCount++;
} else if (gammaValue < 10.30347008950128) {
   steadyCurrentCount++;
} else if (gammaValue < 16.961935216267822) {
   rightHandCurrentCount++;
} else if (gammaValue < 18.166027467870766) {
   steadyCurrentCount++;
} else if (gammaValue < 19.171423570518286) {
   rightHandCurrentCount++;
} else if (gammaValue >= 19.171423570518286) {
   steadyCurrentCount++;
}
*/
counter++;

if (counter == dataRowCount) {

   counterArray[0] = leftHandCurrentCount;
   counterArray[1] = rightHandCurrentCount;
   int maxCount = steadyCurrentCount;
   int maxCountIndex = 2;
   for (int j = 0; j < 2; j++) {
      if (maxCount < counterArray[j]) {
         maxCountIndex = j;
      }
   }
   if (maxCountIndex == 0) {
      leftHandTotalCount++;
      currentState = "Left Hand";
   } else if (maxCountIndex == 1) {
      rightHandTotalCount++;
      currentState = "Right Hand";
   } else {
      steadyTotalCount++;
      currentState = "Steady";
   }
   totalTrials++;

   counter = 0;
   leftHandCurrentCount = 0;
```

```
                    rightHandCurrentCount = 0;
                    steadyCurrentCount = 0;

                    if (leftHandTotalCount > 0) {
                       leftHandTotalCount / totalTrials) * 100);
                    } else {

                    }
                    if (rightHandTotalCount > 0) {
                       rightHandTotalCount / totalTrials) * 100);
                    } else {

                    }
                    if (steadyTotalCount > 0) {
                    } else {
                                              }
                      */
                    /*
                    *** update ui
                     */
                    Platform.runLater(new Runnable() {
                       @Override
                       public void run() {

                          currentStateLabelTxt.setText("" + currentState);
                           if (leftHandTotalCount > 0) {
                              leftHandTotalCountLabelTxt.setText("" + Math.round(((float)
leftHandTotalCount / totalTrials) * 10000) / 100.0 + " %");
                           } else {
                              leftHandTotalCountLabelTxt.setText("0 %");
                           }
                           if (rightHandTotalCount > 0) {
                              rightHandTotalCountLabelTxt.setText("" +
Math.round(((float) rightHandTotalCount / totalTrials) * 10000) / 100.0 + " %");
                           } else {
                              rightHandTotalCountLabelTxt.setText("0 %");
                           }
                           if (steadyTotalCount > 0) {
                              steadyTotalCountLabelTxt.setText("" + Math.round(((float)
steadyTotalCount / totalTrials) * 10000) / 100.0 + " %");
                           } else {
                              steadyTotalCountLabelTxt.setText("0 %");
                           }
                           totalTrialsLabelTxt.setText("" + totalTrials);

                           //animation method call
                          AnimationBall();
```

```
                        }
                    });
                }

                }
                if (i == 5) {
                    i = 13;
                }
            }
        }

        Edk.INSTANCE.IEE_EngineDisconnect();
        Edk.INSTANCE.IEE_EmoStateFree(eState);
        Edk.INSTANCE.IEE_EmoEngineEventFree(eEvent);

    }
  }

  }
}
```