Brain Controlled Device
An application of BCI

A Project Report

submitted by

Rachith P (12EC76)

Sanjay R (12EC84)

Sourav Sudhir (12EC91)

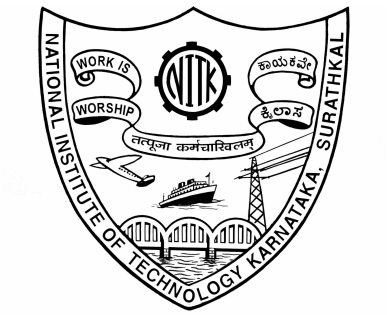
under the guidance of

Dr.Deepu Vijayasenan

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

April 18, 2016

ABSTRACT

The purpose of this project is to obtain control signals from the electroencephalograph (EEG), and to use these signals to control a bot or any other electronic device. EEG is the recorded electrical activity generated by the brain. A headset is used to record this signal. The first phase of the project involves obtaining a good accuracy for classification from a set of existing mental imagery datasets. For this, the essential features such as average signal, sub-band power and common spatial pattern are analyzed. Pre-processing techniques such as laplacian filtering is done on the raw data in order to remove artifacts contributing redundant information. The next phase is to collect the data using an EEG headset and to analyse it. In this project, data is analysed from two headsets: Neurosky Mindwave (single channel) and Emotiv Epoc(mutli-channel). For multi-channel headset, data is recorded from the different key regions of the brain. Using arduino, processing and python, real-time analysis is done in order to obtain the control signals from the EEG devices. Here power spectral densities across moving(overlapping)time windows are considered as features. These features are then used to classify the different mental tasks. Suitable classifiers have to be chosen to obtain better accuracy. Data for visual, auditory of around 10 subjects and motor imagery tasks of around 50 were analysed and suitable classifiers are used to get good accuracy. Following this, real-time integration of the multi-channel headset is completed using arduino and python and various tasks are tested for accuracy with suitable demonstrations.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
Introduction

1.1 Problem definition

Brain-computer interface (BCI) has recently emerged as an important and interesting topic of research with widespread applications. It has evolved as a novel technique in assisting people to activate certain devices by brain signals. Currently there are two main methods for accomplishing BCI: electroencephalography (EEG) signals, or Electrocorticography (ECoG), which measures the electrical activity of the brain taken from beneath the skull. But, ECoG method is accompanied by a host of risks and complications; therefore most BCI research is currently focused on EEG signals. In this method, electrodes are placed on the scalp, and the brain's electrical activity is monitored and recorded. By using the EEG signals, classification algorithms can be developed to detect distinct brain functions. The acquisition of EEG signals can be done using a variety of hardware. Once brain functions are decodable, various problems pertaining to real life situations can be solved. The control signals can be used by disabled people to control a prosthetic arm or to move his wheelchair or any other applications which bring about a positive change in their lifestyle. Additionally signals obtained can be used in medical field for epilepsy monitoring, predicting seizures etc. Hence, our focus is on the methods to extract these control signals from raw EEG data.

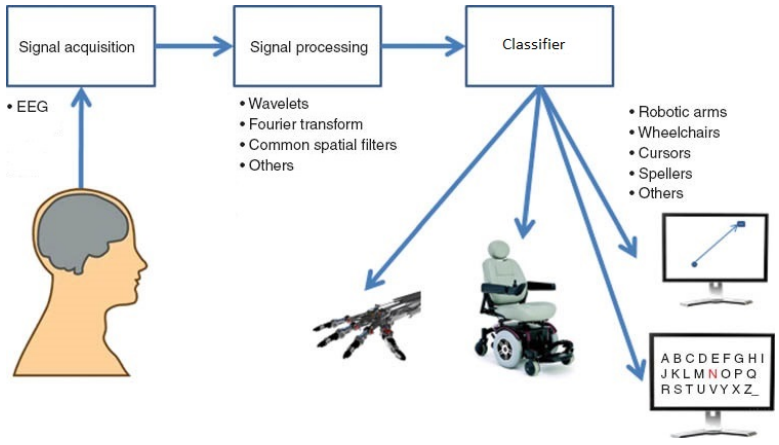


Figure 1.1: Problem definition

1.2 Previous work

Most of the previous work (like [7] and [1])had focused on processing signals corresponding to actual motor movement or actual movement of the hand. In this project, signals corresponding to intended motor movement or intended motion of the hand is processed to obtain the control signals.
The previous works such as [5] and [8] have worked on parts of the brain such as cortical and parietal respectively. In this project we are trying to obtain control signals by analyzing EEG signals for different tasks pertaining to different parts of the brain such as parietal, occipital and temporal.

1.3 Motivation

It has always been a mystery on how the brain works. There has been a significant amount of improvement in understanding the brain over the past decade. There has been an increase in research in the field of bio-medical where understanding the brain has led to finding cure for many diseases. Various methods are in existence to study brain functions including Electroencephalography, functional magnetic resonance imaging, magnetoencephalography, Nuclear Magnetic Resonance Spectroscopy, etc. Among those, EEG is powerful tool for tracking brain changes during different phases of life. Also it is considered to involve less cost and safer compared to ECoG(Electrocorticography). In order to move an object, brain sends control signals to one's hand/leg. During the process, neurons are fired up at different parts of the brain producing low voltage signals. If it would be possible to classify these brain signals and try to obtain a decision/thought being thought by a person, it would revolutionise the paradigm of interfacing brain with electronic devices. This would greatly enhance the lifestyle of people suffering from ailments like epilepsy, etc. Nevertheless, this is quite challenging as one cannot obtain a perfect signal from the brain, owing to debasing from various artefacts.

1.4 Overview

The project can be divided into multiple stages, with the first stage being identification of feature set that when is input to an appropriate classifier outputs an high accuracy classification result. The major task required was to get a set of suitable and efficient feature set which would be bankable to produce results with high accuracy. Then, feature extraction is carried out on a suitable existing dataset which had a similar experiment of motor imagery which we intend to study. Two suitable datasets with two classes(left,right) are taken into consideration .

Different set of features were used as inputs for the classifiers: averaging over common window length per class, Discrete wavelet transform analysis or Fourier transform in order to obtain respective frequency band(alpha,beta,gamma,etc.) components, the power spectrum of these rhythmic waves, Spatial filtering approach using Common Spatial Pattern filter to get a feature set which has notable difference in variance of the data from different classes respectively. After analysis, power spectral density of sub-bands were considered to be the main feature set as they gave better results.The next step is to choose a suitable classifier and test the accuracy of the filtered data.

The next stage of the project is to record the EEG signal for a diverse set of experiments, using both single-channel and multi-channel headsets and to extract PSD(features) to obtain control signals. Multi-channel headset gave better results than single-channel as expected. Control signals are used for suitable application with demonstration.

CHAPTER 2
Description

2.1 The Datasets

The first task is to obtain an appropriate dataset which contained EEG signals corresponding to imagined motor movements. The requirement is that the dataset should be as close as possible to the data which would be obtained from the headset.

2.1.1 BCI Competition III - Dataset V

The dataset selected is on multiple class mental imagery. It was taken from BCI competition 3, dataset 5. It contained data from 3 normal subjects during 4 non-feedback sessions. Subjects performed 2 imagined motor movement tasks as EEG activity was recorded on 32 channels placed on their scalp as per the 10-20 standard. The subjects were given 2 tasks:

1. Imagination of repetitive self-paced left hand movements
2. Imagination of repetitive self-paced right hand movements

All 4 sessions of a given subject were acquired on the same day, each lasting 4 minutes. The subject performed a given task for about 15 seconds and then switched randomly to another task at the operator's request. The sampling rate of the raw EEG signal was 512 Hz.

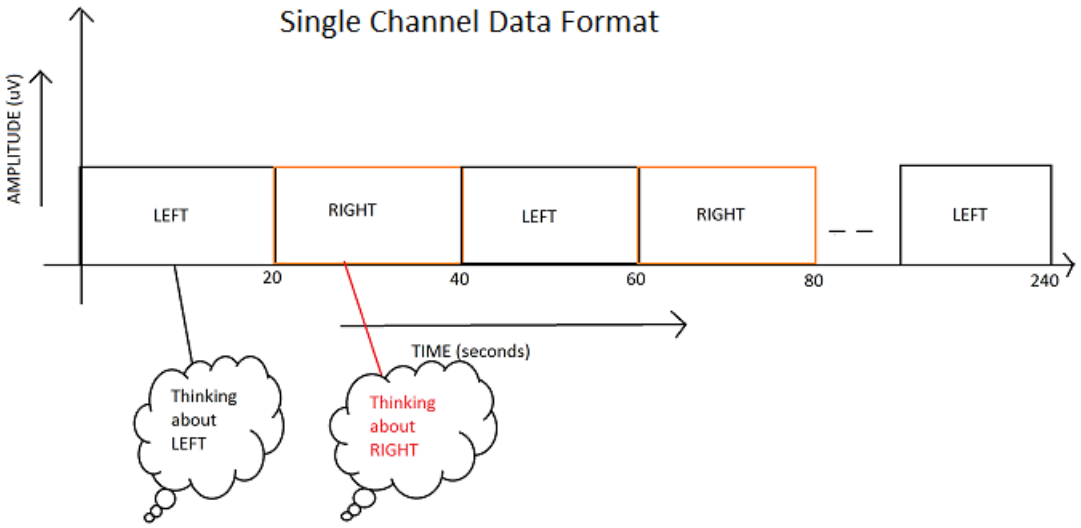


Figure 2.1: Description of dataset

2.1.2 BCI Competition IV - Dataset I

Most demonstrations of algorithms on BCI data are evaluating classification of EEG trials, i.e., windowed EEG signals for fixed length, where each trial corresponds to a specific mental state. But in BCI applications with asynchronous feedback, one is faced with the problem that the classifier has to be applied continuously to the incoming EEG without having cues of when the subject is switching her/his intention. This data set poses the challenge of applying a classifier to continuous EEG for which no cue information is given. These data sets were recorded from healthy subjects. In the whole session motor imagery was performed without feedback. For each subject two classes of motor imagery were selected from the three classes left hand, right hand, and foot (side chosen by the subject; optionally also both feet).
In the first two runs, arrows pointing left, right, or down were presented as visual cues on a computer screen. Cues were displayed for a period of 4s during which the subject was instructed to perform the cued motor imagery task. These periods were interleaved with 2s of blank screen and 2s with a fixation cross shown in the center of the screen. The fixation cross was superimposed on the cues, i.e. it was shown for 6s. These data sets are provided with complete marker information.
The recording was made using BrainAmp MR plus amplifiers and a Ag/AgCl electrode cap. Signals from 59 EEG positions were measured that were most densely distributed over sensorimotor areas. Signals were band-pass filtered between 0.05 and 200 Hz and then digitized at 1000 Hz with 16 bit (0.1 uV) accuracy. THe resulting signal was downsampled to 100 Hz.

Format of the Data

Data includes continuous signals of 59 EEG channels and, for the calibration data, markers that indicate the time points of cue presentation and the corresponding target classes. Data are provided in Matlab format (*.mat) containing variables:

- cnt: the continuous EEG signals, size [time × channels]. The array is stored in datatype INT16. To convert it to uV values, cnt= 0.1*double(cnt) was done.
- mrk: structure of target cue information with fields (the file of evaluation data does not contain this variable)
  - pos: vector of positions of the cue in the EEG signals given in unit sample
  - y: vector of target classes (-1 for class one or 1 for class two)
- nfo: structure providing additional information with fields
  - Fs, sampling rate,
  - clab: cell array of channel labels,
  - classes: cell array of the names of the motor imagery classes,
  - xpos: x-position of electrodes in a 2d-projection,
  - ypos: y-position of electrodes in a 2d-projection,

2.1.3 Headsets

Once analysis had been carried out on existing BCI datasets, data of people performing similar tasks were recorded using a headset.

Brainsense

Figure 2.2: Single electrode headset

This headset(Brainsense) contains one EEG electrode placed at the position corresponding to Fp1 as shown in fig:2.4 on the frontal lobe of the brain. The device consists of a headset, an ear-clip, and a dry sensor. The headset's reference and ground electrodes are on the ear clip and the EEG electrode is on the sensor arm, resting on the forehead above the eye (FP1 position). The headset communicates using Bluetooth v2.1. The data output from the headset is the rate of 512 Hz.

Emotiv Epoc

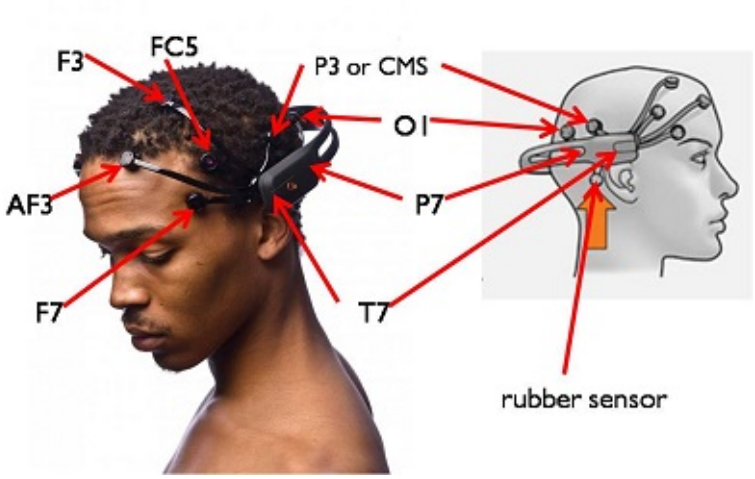


Figure 2.3: Single electrode headset

The single channel electrode was sufficient to determine the state of the brain and blinking. But in order to record and analyse complex actions like motor imagery, response to

visual and auditory stimulus, multi-channel is required. The emotiv EPOC has 14 electrodes distributed according to the 10-20 International standard and records data from frontal, parietal, temporal and occipital lobe. The sampling rate of the device is 128Hz and communication occurs through a RF module.

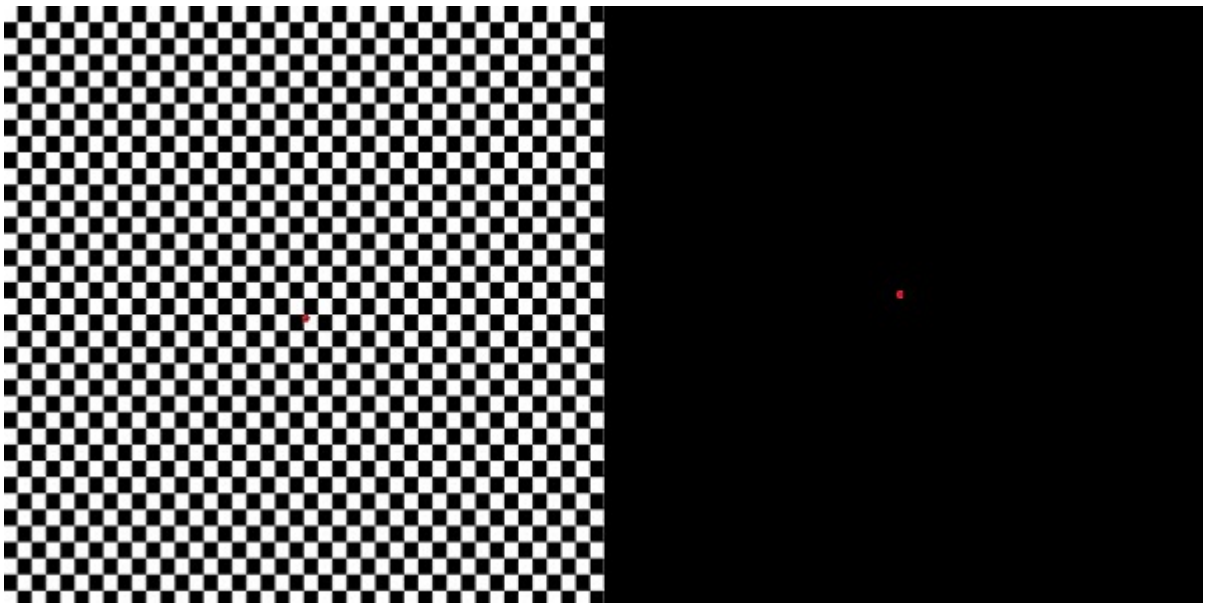
Data recording for single channel headset

Once the headset is connected to the system with a Bluetooth device, continuous stream of data is obtained. Once a connection is established between the headset and the laptop, software called Neuroview is used to obtain raw data and Power Spectral Density values from the headset. These values can be logged into a csv file. Data corresponding to a person thinking right and left was recorded, each for a duration of 20s. 3-4 instances of the recording was done at a time per day. Similar recordings were repeated 2-3 times with 1 day interval. Time logged data was obtained, hence, each raw data was class labelled(to either right or left).

Data recording for multi-channel headset

Data is collected from 10 subjects for visual and auditory stimulus.

Visual Experiment - I
For the visual stimulus, the cues used were an alternating checkerboard indicating the active state and a plain dark black background indicating neutral state.



Visual Experiment - II
Another visual task which was worked on was SSVEP. Steady State Visually Evoked Potentials (SSVEP) are signals that are natural responses to visual stimulation at specific frequencies. When the retina is excited by a visual stimulus ranging from 3.5 Hz to 75 Hz, the brain generates electrical activity at the same (or multiples of) frequency of the visual stimulus. An LED blinking at a rate of 12.8Hz and 15.6Hz was used as the stimulus.

Auditory Experiment
For the auditory stimulus,two different sounds were played(via earphones) where the subject listens to the sounds with eyes closed, sitting on a chair comfortably. The two sounds used were bird chirp indicating the active state and white noise indicating neutral state. The duration of each sound 8s. Two trials were recorded for each of the tasks and in each trial, the subject was asked to listen to the two sounds twice alternatively.

Motor Imagery Experiment
Data from 50 subjects was collected for motor imagery. For motor imagery, the positions of the electrode were changed slightly such that data from parietal lobe can be recorded. Each subject is given three motor imagery tasks: push-pull-neutral, left-right-neutral and up-down-neutral. Each activity is recorded for 15 seconds. An audible beep is used as a cue for the subject to change the task. Two trials were recorded for each of the motor imagery tasks and in each trial, the subject was asked to repeat the task two times. Each user had to perform the given task for a duration of 15s. The data so obtained was used to train a model which is user-independent.

2.2 Channel Selection

2.2.1 BCI Dataset

After obtaining the dataset, the next task was to identify the channels(location of the electrode on the scalp) in the data that had to be selected. As shown in fig:2.4 channels namely C3, CZ, C4, P3, PZ, P4 are selected for analysis. Since parietal section of the brain is associated with imagination as said in [5] and [8], the above channels were considered.
Since data from one channel might interfere with the data recorded from the other channel, namely artifacts may be present. These artifacts may decrease the accuracy of the system. Hence, Laplacian smoothing is applied in order to remove the redundant information.

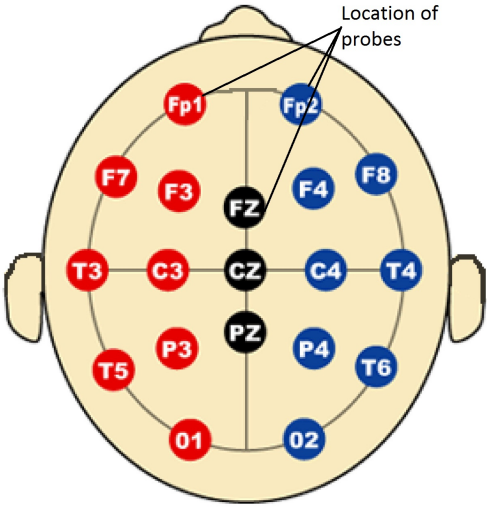


Figure 2.4: 10-20 diagram with location of probes

2.2.2 Multi-Channel Headset

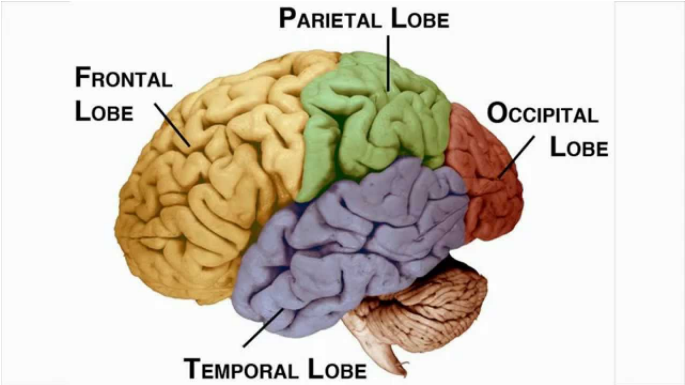


Figure 2.5: Various lobes in the brain

Since the occipital lobe (shown in fig:2.5) is activated during visual activity, the channels corresponding to electrodes from the occipital lobe are considered for actions involving visual stimulus. The channels O1, O2, P7 and P8 are therefore selected.
Any auditory stimulus results is a activation of the temporal lobe (shown in fig:2.5) of the brain and therefore the channels T7 and T8 are selected while processing the data recorded for auditory response.
As the motor imagery tasks results in excitation of the parietal lobe region (shown in fig:2.5), the data from the channels corresponding to the electrodes in this region are considered for processing.

2.3 Feature Extraction

Feature extraction involves reducing the amount of resources required to describe a large set of data. The extracted features should be able to represent a huge data withholding the important information without much loss, thus minimizing the complexity. Therefore, selecting the appropriate feature was important. The features studied were:

1. Average value

2. Sub-band power

3. Common spatial pattern (CSP)

2.3.1 Average Value

As the brain does not think of a single thought at all times, it is better to consider the average value of all the observations.
For e.g., If we have 240 seconds of data of an individual thinking left and 240 seconds of data of the same individual thinking right, such that for a duration of 20 seconds, the person thinks only of left or right. Then, the average value is the average of all such 20 second windows.

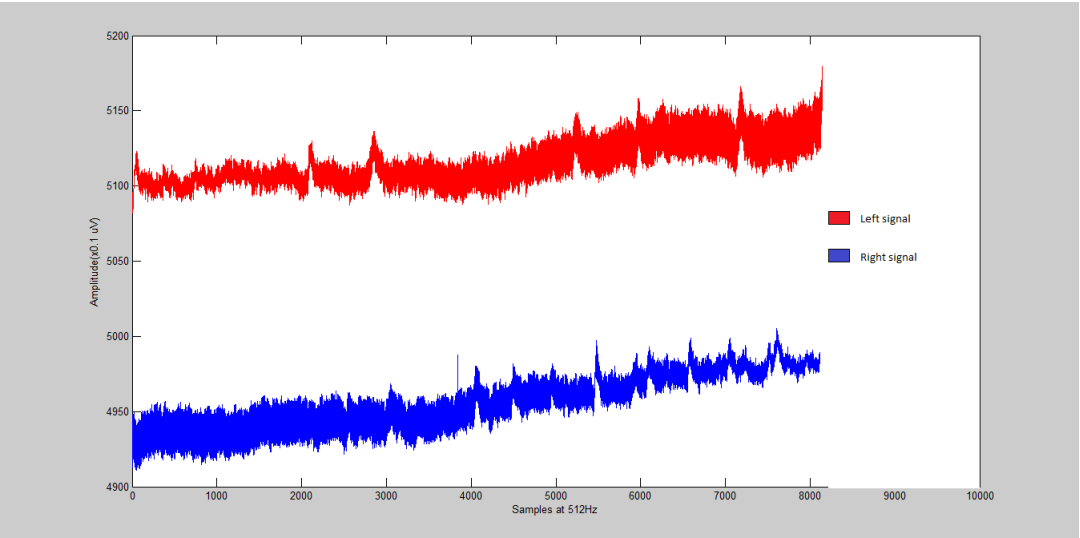


Figure 2.6: Average signal

2.3.2 Sub-band Power

The next feature which was considered is the sub-band power. Fourier transform was applied on the data and the Power Spectral Density of the sub-bands ranging from 8Hz to 50Hz were computed. These sub-bands are usually called alpha(8-12Hz),beta(12-40Hz),gamma(>40Hz),theta(4-8Hz) and delta(0-4Hz) bands. For using feature vectors for the classification, power spectral densities with frequency resolution of 0.25-2 Hz were considered and tested. This means that for an EEG signal of sampling frequency 512 Hz(say) the fft of 2048-256 point is taken depending on 0.25-2 Hz resolution. Then power spectral density is calculated using the following formula,

$$X_{PSD}(k) = |X(k)|^2$$

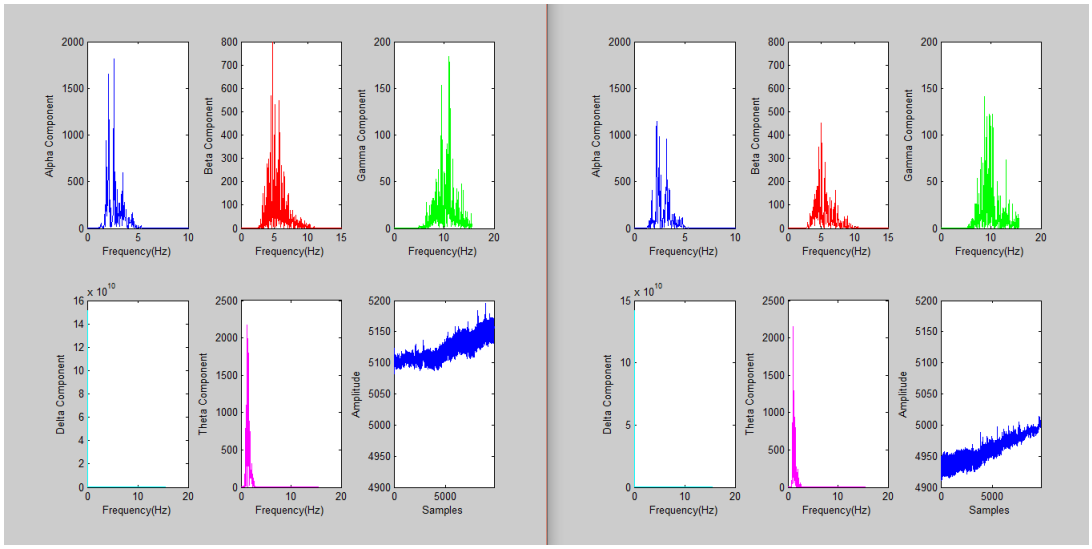


Figure 2.7: Sub-band power

2.3.3 Common Spatial Pattern

Common spatial pattern (CSP) is a mathematical procedure used in signal processing for separating a multivariate signal into additive subcomponents which have maximum differences in variance between two windows. In other words, the method of Common Spatial Pattern aims at designing spatial filters which could discriminate two data sets based on its variance. The algorithm is explained in detail in [9] and [10].

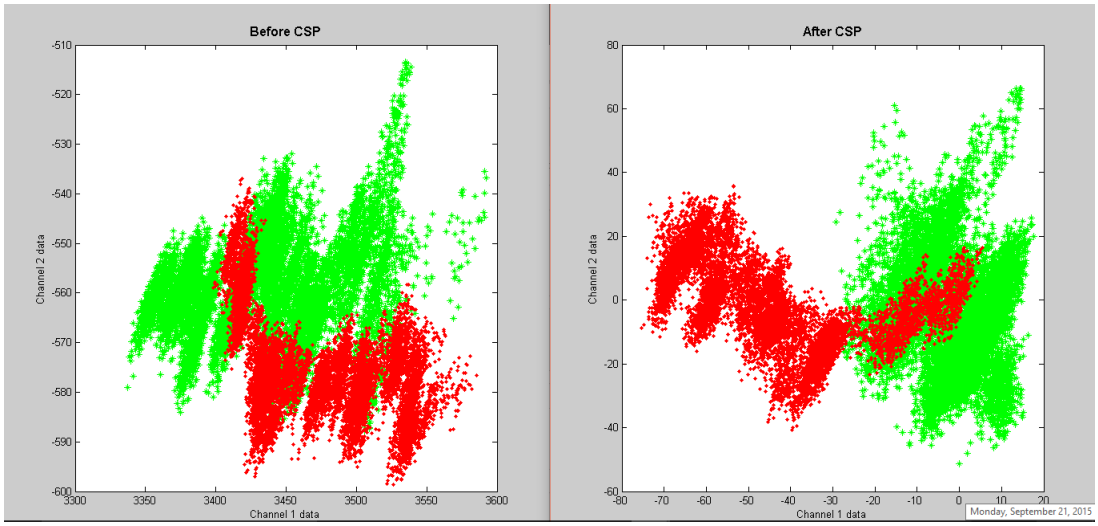


Figure 2.8: Before and after CSP

CHAPTER 3
Analysis

3.1 Feature Selection

Based on the given dataset (BCI3, Data V), attempts have been made to study some of the different methods used to extract the various features helpful in classification of EEG data. Based on the study the following conclusions were drawn:

1. The CSP analysis requires the presence of multiple channels, which were not available in the initial headset used.

2. The CSP analysis was able to successfully inject disparity between data points of the two classes based on variance. However, in most cases there was a negative effect of using CSP which blurred the difference in variance making it more difficult to classify. The fig:2.8 delineates the effect of CSP on data taken from average of trail-1 of channels Fp1, Fp2 and Fz.

Therefore, it was concluded that CSP was not a useful method for obtaining features for the required real-time application.
Also, it was concluded that the average signal was not a useful feature because it reduced the number of datapoints considerably and meaningful results could not be obtained.
The sub-band power feature was initially obtained using Discrete Wavelet Transform(DWT). Using DWT gave very few bins, namely alpha, beta, gamma etc., and the resolution was low. Lower number of frequency bins meant lesser number of features, so the classification accuracy would be low. In order to improve the resolution and to obtain more number of frequency bins, FFT was performed on the data and the corresponding Power Spectral Density was calculated. This was used as the input feature set.

3.1.1 Preprocessing

Surface Laplacian was used to process the signals and remove the artifacts. From the 10-20 standard, a distance matrix was obtained which comprised of distances of each of the channels with respect to another. The resulting signal obtained after laplacian smoothing for the j^{th} channel was calculated by:

$$resultant\_signal_j = \frac{\sum_{i=1, i\neq j}^{n} signal_i \times e^{-\lambda_i \times d_{ij}}}{\sum_{i=1}^{n} e^{-\lambda_i \times d_{ij}}}$$

where,
λ_i is the reciprocal average distance between the i^{th} channel and other channels
d_{ij} is the distance between i^{th} and j^{th} channel
n is the total number of channels

3.1.2 Power Spectral Density

Power Spectral Density was obtained by first obtaining the moving window fft of the raw data. Also, the sampling rate of the data had to be reduced in order to reduce the amount of data. The length of the fft was determined by the required frequency resolution for the dataset. The length of fft was chosen to be 256 for dataset 1(giving a frequency resolution of 2Hz) and 2048 for dataset 2 and headset data(giving a resolution of 0.25Hz).The size of the window determined the the reduced sampling rate of the data. For dataset 1 512 Hz data was downsampled to 16 Hz data. The window size was chosen to be 32 for dataset 1(giving a sampling rate of 16Hz) and 64 for dataset 2 and headset data(giving a sampling rate of 8Hz). After the fft is obtained, the Power Spectral Density value was computed.

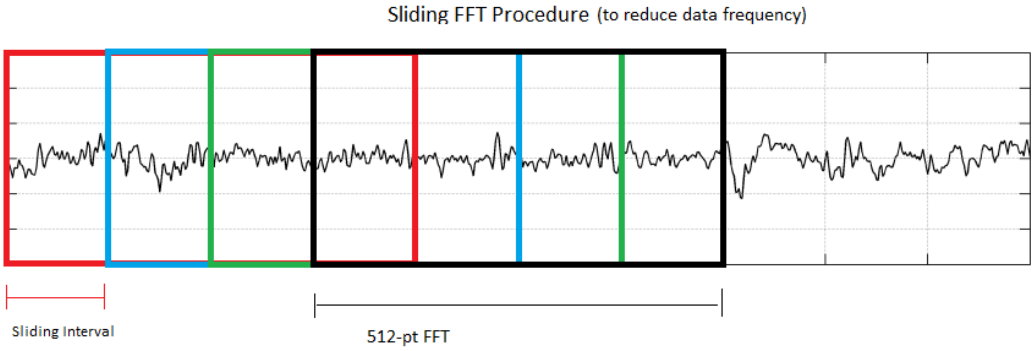


Figure 3.1: Moving window fft

3.2 Classifier Selection

Due to the nature of the data, the following classifiers were tested:

- Logistic Regression
- libSVM with linear kernel
- libSVM with RBF kernel
- Random Forest
- J48 Tree

3.3 Tools used for analysis

3.3.1 Weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. The

Figure 3.2: Weka

Weka tool takes the input file in csv format and gives accuracy and confusion matrix.
In addition to a large number of classifiers, weka also offers functionalities such as the option to split the input data into training and testing sets, cross-validation of the input data and using different files for training and testing.
Since the raw data in the dataset and that obtained from the headset are stored in csv format, weka is an ideal tool for training and classification.

3.3.2 Scikit-learn

Scikit-learn is an open source machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Figure 3.3: Scikit-learn

The software development kit(sdk) provided by the headset which was used to record the data gave the resulting file in csv format. These files were then imported to Matlab or Weka and processing was done. This procedure could not be extended to real-time analysis as the amount of time consumed from storing the data in csv and then accessing it in Weka or Matlab was not feasible.
This dependency was removed by writing scripts in python to do the data recording

and classification for real-time applications. Scikit-learn could be easily integrated with python, therefore it was the best choice for use in real-time applications.

3.4 Post classification processing

The real time data from Emotiv epoc headset, after classification will be output at 8 Hz frequency, i.e., every one second gives 8 predictions. To make the system more accurate and also less jerky, the output is smoothed to 1 Hz. For this reduction, the maximum occurrence of the class in a window of 8 predictions is taken. This technique yielded slightly better classification accuracy compared to direct output from the classifier.

3.5 Real-time classification with the single channel headset

Connection using Arduino and HC-05

For obtaining the real-time signal, the bluetooth module in the headset was first paired with the HC 05 module. The external bluetooth module was connected to the arduino UNO microcontroller. The HC 05 module was configured to automatically pair with the headset using Attention(AT) commands, making it the master device and the headset to be the slave device. This facilitated easy coding to send commands to the ASIC on the headset. The flow chart of the connections made is depicted in fig:3.4

3.5.1 Communication protocol

Thinkgear is the ASIC inside the headset which enables easy interfacing with the wearers' brainwave. It includes the sensor that touches the forehead, the contact and reference points located on the ear pad, and the onboard chip that processes all of the data and provides this data to software and applications in digital form. Both the raw brainwaves and the Attention and Meditation values are calculated on the ThinkGear chip. ThinkGear components deliver their digital data as an asynchronous serial stream of bytes. The serial stream must be parsed and interpreted as ThinkGear Packets in order to properly extract and interpret the data values.
A ThinkGear Packet is a packet format consisting of 3 parts:

1. Packet Header
2. Packet Payload
3. Packet Checksum

ThinkGear Packets are used to deliver Data Values from a ThinkGear module to an arbitrary receiver (a PC, another microprocessor, or any other device that can receive a serial stream of bytes).

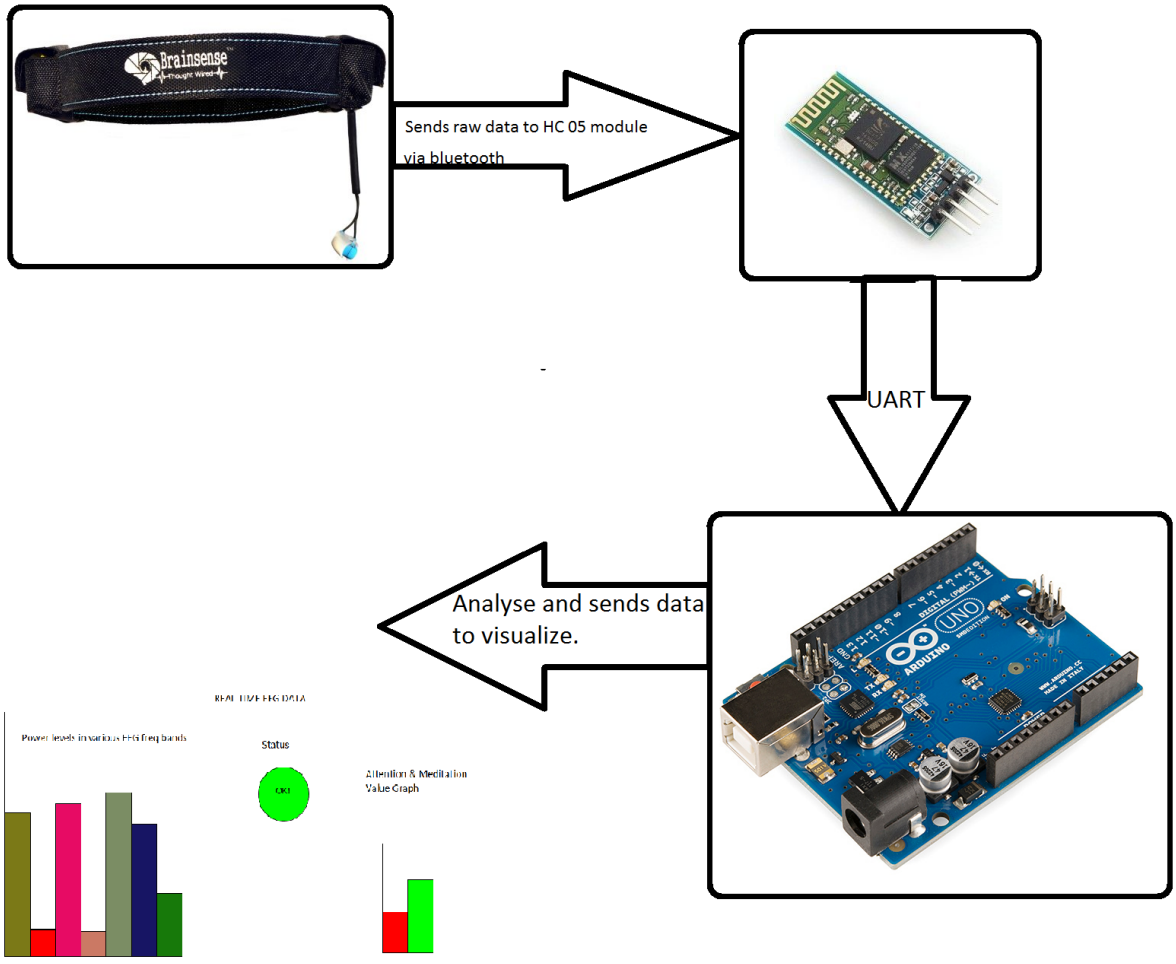


Figure 3.4: Connections made between headset and PC

Packet Structure

Packets are sent as an asynchronous serial stream of bytes. The transport medium may be UART, serial COM, USB, bluetooth, file, or any other mechanism which can stream bytes. Each Packet begins with its Header, followed by its Data Payload, and ends with the Payload's Checksum Byte, as follows:

```
[SYNC] [SYNC] [PLENGTH]  [PAYLOAD...]    [CHKSUM]
_____________________    _____________   ____________
^^^^^^^^(Header)^^^^^^^  ^^(Payload)^^   ^(Checksum)^
```

Figure 3.5: Packet Structure

The [PAYLOAD] section is allowed to be up to 169 bytes long, while each of [SYNC], [PLENGTH], and [CHKSUM] are a single byte each. This means that a complete, valid Packet is a minimum of 4 bytes long (possible if the Data Payload is zero bytes long, i.e. empty) and a maximum of 173 bytes long (possible if the Data Payload is the maximum 169 bytes long).

Packet Header
The Header of a Packet consists of 3 bytes: two synchronization [SYNC] bytes (0xAA 0xAA), followed by a [PLENGTH] (Payload length) byte:

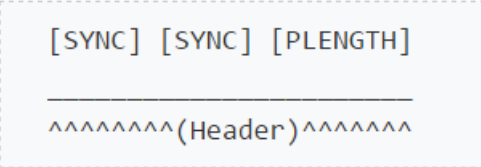


Figure 3.6: Packet Header

The two [SYNC] bytes are used to signal the beginning of a new arriving Packet and are bytes with the value 0xAA (decimal 170). Synchronization is two bytes long, instead of only one, to reduce the chance that [SYNC] (0xAA) bytes occurring within the Packet could be mistaken for the beginning of a Packet. Although it is still possible for two consecutive [SYNC] bytes to appear within a Packet (leading to a parser attempting to begin parsing the middle of a Packet as the beginning of a Packet) the [PLENGTH] and [CHKSUM] combined ensure that such a "mis-sync'd Packet" will never be accidentally interpreted as a valid packet.
The [PLENGTH] byte indicates the length, in bytes, of the Packet's Data Payload [PAYLOAD] section, and may be any value from 0 up to 169. Any higher value indicates an error (PLENGTH TOO LARGE). Note that [PLENGTH] is the length of the Packet's Data Payload, not of the entire Packet. The Packet's complete length will always be [PLENGTH] + 4.

Data Payload
The Data Payload of a Packet is simply a series of bytes. The number of Data Payload bytes in the Packet is given by the [PLENGTH] byte from the Packet Header. Note that parsing of the Data Payload typically should not even be attempted until after the Payload Checksum Byte [CHKSUM] is verified as described in the following section.

Payload Checksum
The [CHKSUM] Byte must be used to verify the integrity of the Packet's Data Payload. The Payload's Checksum is defined as:

- summing all the bytes of the Packet's Data Payload
- taking the lowest 8 bits of the sum
- performing the bit inverse (one's compliment inverse) on those lowest 8 bits

A receiver receiving a Packet must use those 3 steps to calculate the checksum for the Data Payload they received, and then compare it to the [CHKSUM] Checksum Byte received with the Packet. If the calculated payload checksum and received [CHKSUM] values do not match, the entire Packet should be discarded as invalid. If they do match, then the receiver may proceed to parse the Data Payload.

Data Payload Structure
Once the Checksum of a Packet has been verified, the bytes of the Data Payload can be parsed. The Data Payload itself consists of a continuous series of Data Values, each contained in a series of bytes called a DataRow. Each DataRow contains information about what the Data Value represents, the length of the Data Value, and the bytes of the Data Value itself. Therefore, to parse a Data Payload, one must parse each DataRow from it until all bytes of the Data Payload have been parsed.
A DataRow consists of bytes in the following format:

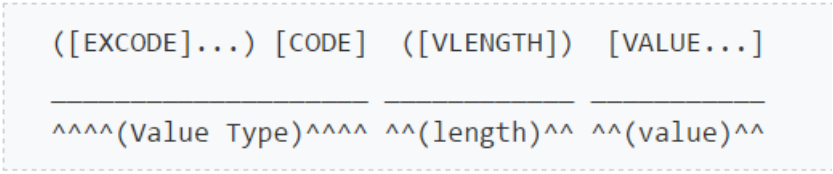


Figure 3.7: DataRow format

Bytes in parentheses are conditional, meaning that they only appear in some DataRows, and not in others. The DataRow may begin with zero or more [EXCODE] (Extended Code) bytes, which are bytes with the value 0x55. The number of [EXCODE] bytes indicates the Extended Code Level. The Extended Code Level, in turn, is used in conjuction with the [CODE] byte to determine what type of Data Value this DataRow contains. Parsers should therefore always begin parsing a DataRow by counting the number of [EXCODE] (0x55) bytes that appear to determine the Extended Code Level of the DataRow's [CODE].

The [CODE] byte, in conjunction with the Extended Code Level, indicates the type of Data Value encoded in the DataRow. For example, at Extended Code Level 0, a [CODE] of 0x04 indicates that the DataRow contains an eSense Attention value. For a list of defined [CODE] meanings, see the CODE Definitions Table below. Note that the [EXCODE] byte of 0x55 will never be used as a [CODE] (incidentally, the [SYNC] byte of 0xAA will never be used as a [CODE] either).

If the [CODE] byte is between 0x00 and 0x7F, then the the [VALUE] is implied to be 1 byte long (referred to as a Single-Byte Value). In this case, there is no [VLENGTH] byte, so the single [VALUE] byte will appear immediately after the [CODE] byte. If, however, the [CODE] is greater than 0x7F, then a [VLENGTH] (Value Length) byte immediately follows the [CODE] byte, and this is the number of bytes in [VALUE] (referred to as a Multi-Byte Value). These higher CODEs are useful for transmitting arrays of values, or values that cannot be fit into a single byte.

The DataRow format is defined in this way so that any properly implemented parser will not break in the future if new CODEs representing arbitrarily long DATA values are added (they simply ignore unrecognized CODEs, but do not break in parsing), the order of CODEs is rearranged in the Packet, or if some CODEs are not always transmitted in every Packet.

Single-Byte CODEs

```
Extended              (Byte)
Code Level  [CODE]  [LENGTH]  Data Value Meaning
----------  ------  --------  ------------------
```

```
         0    0x02          -  POOR_SIGNAL Quality (0-255)
         0    0x03          -  HEART_RATE (0-255)
                                 Once/s on EGO.
         0    0x04          -  ATTENTION eSense (0 to 100)
         0    0x05          -  MEDITATION eSense (0 to 100)
         0    0x06          -  8BIT_RAW Wave Value (0-255)
         0    0x07          -  RAW_MARKER Section Start (0)
```

Figure 3.8: Single-Byte codes

Multi-Byte CODEs

```
Extended            (Byte)
Code Level  [CODE]  [LENGTH]  Data Value Meaning
----------  ------  --------  ------------------
         0    0x80         2  RAW Wave Value: a single big-endian
                                 16-bit two's-compliment signed value
                                 (high-order byte followed by
                                 low-order byte) (-32768 to 32767)
         0    0x81        32  EEG_POWER: eight big-endian 4-byte
                                 IEEE 754 floating point values
                                 representing delta, theta, low-alpha
                                 high-alpha, low-beta, high-beta,
                                 low-gamma, and mid-gamma EEG band
                                 power values
```

```
         0    0x83        24  ASIC_EEG_POWER: eight big-endian
                                 3-byte unsigned integer values
                                 representing delta, theta, low-alpha
                                 high-alpha, low-beta, high-beta,
                                 low-gamma, and mid-gamma EEG band
                                 power values
         0    0x86         2  RRINTERVAL: two byte big-endian unsigned
                                 integer representing the milliseconds
                                 between two R-peaks
```

```
       Any    0x55          -  NEVER USED (reserved for [EXCODE])
       Any    0xAA          -  NEVER USED (reserved for [SYNC])
```

Figure 3.9: Multiple-Byte codes

3.5.2 Processing the real-time data

Once a connection is established between the HC-05 and the headset programmed by the arduino, stream of data is received from the headset. The data sent is 32 byte long. With the first two bytes being the sync bytes, third byte being the poor-signal quality indicator, fourth byte indicated the type of data sent by the headset - in this headset, 0x83 is sent indicating 'ASIC_EEG_POWER' as shown in fig:3.9. The next byte indicated the length of the data corresponding to the EEG_POWER(byte value 24) wherein 3 consecutive bytes corresponds to a the EEG_POWER of the frequency bands namely: delta (0.5 - 2.75Hz), theta (3.5 - 6.75Hz), low-alpha (7.5 - 9.25Hz), high-alpha (10 - 11.75Hz), low-beta (13 - 16.75Hz), high-beta (18 - 29.75Hz), low-gamma (31 - 39.75Hz), and mid-gamma (41 - 49.75Hz). These values have no units and therefore are only meaningful compared to each other and to themselves, to consider relative quantity and temporal fluctuations. Next byte(0x04) indicates that the byte after that contains the attention value and the byte 0x05 indicates that the byte after that contains meditation value.
Once these are decoded, processing software is used to visualize the EEG_POWER spectrum of the frequency bands, the attention and meditation values and connection status as shown in fig:3.10 and fig:3.11.

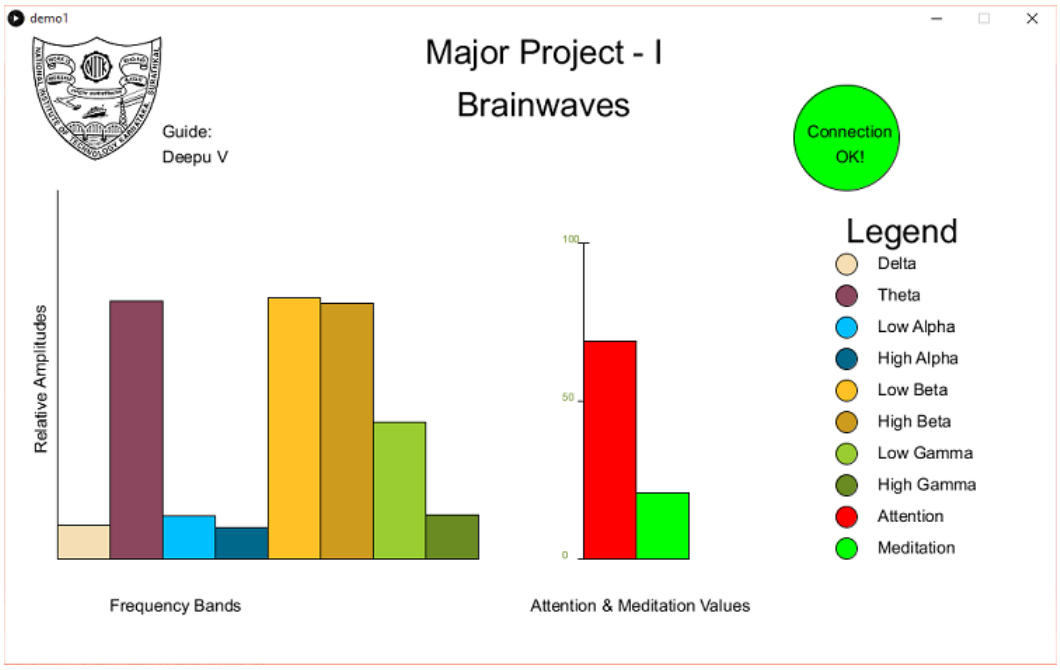


Figure 3.10: Simulation of real-time data

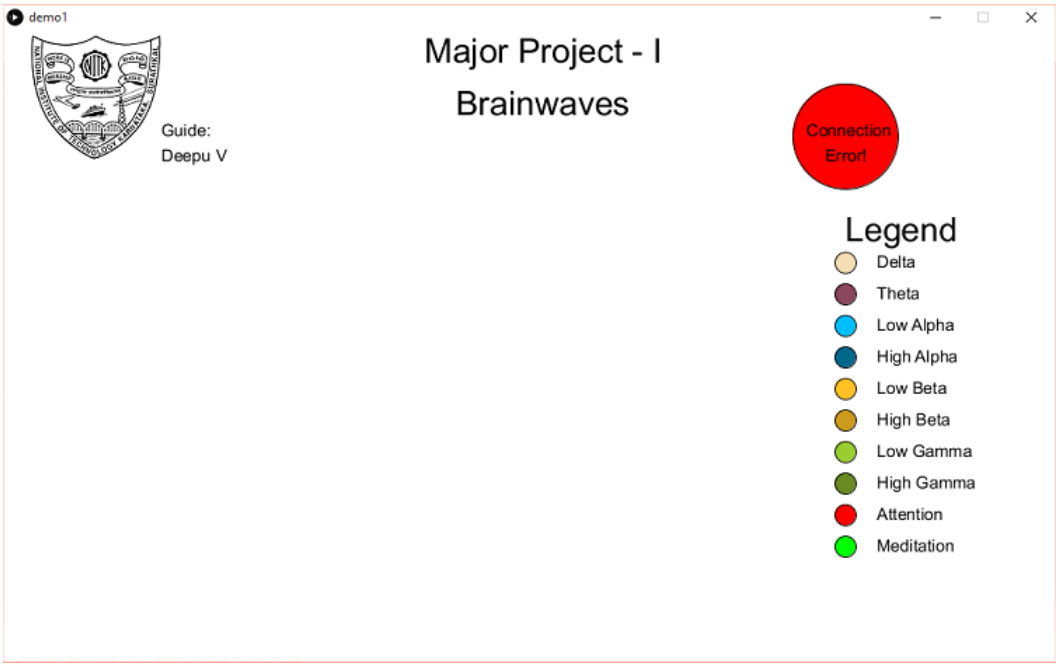


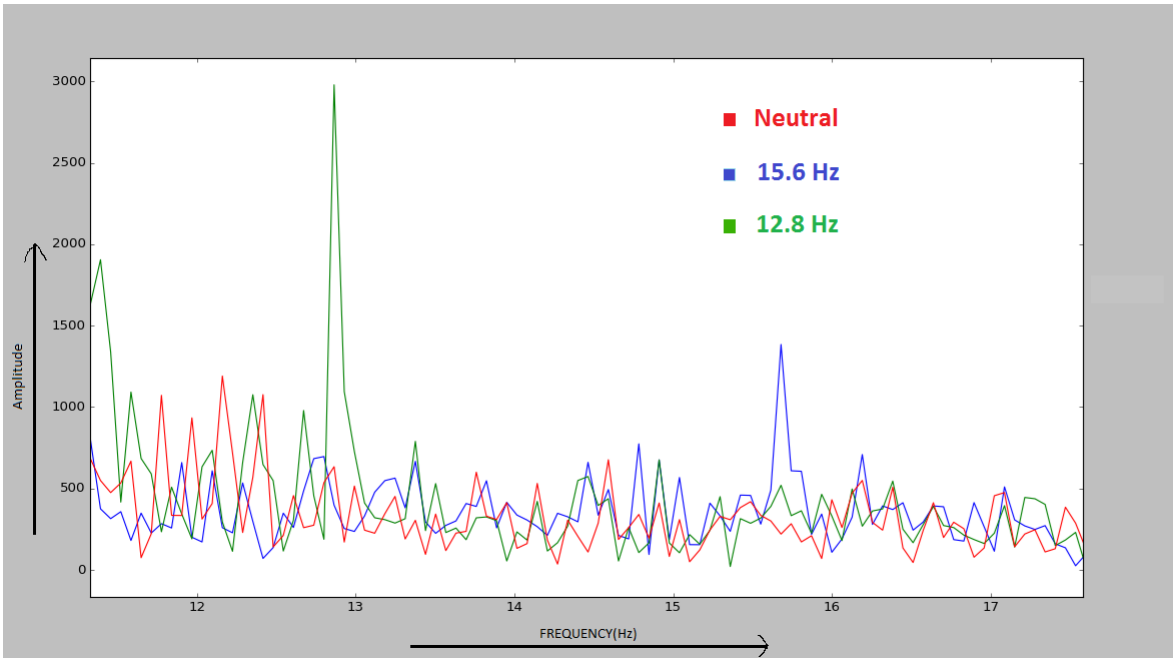
Figure 3.11: Bad Connection

3.6 Real-time classification with multi-channel headset

3.6.1 SSVEP - Steady State Visually Evoked Potentials

SSVEP are signals that are natural responses to visual stimulation at specific frequencies. When the retina is excited by a visual stimulus ranging from 3.5 Hz to 75 Hz, the brain generates electrical activity at the same (or multiples of) frequency of the visual stimulus.

This technique is used widely with electroencephalographic research regarding vision. SSVEP's are useful in research because of the excellent signal-to-noise ratio and relative immunity to artifacts. SSVEP's also provide a means to characterize preferred frequencies of neocortical dynamic processes. SSVEP is generated by stationary localized sources and distributed sources that exhibit characteristics of wave phenomena.
In our experiment, we are using a LED blinking at two frequencies: 12.8Hz and 15.6Hz.



3.6.2 Controlling the time period of a remote triggered pendulum

The objective of this remote triggered experiment is to set the time period and the waveform of an oscillating pendulum. The length of the pendulum can be adjusted by the user.
A graphic user interface(GUI) is developed using which the user can set the length of the pendulum using push motor movement. The GUI consists of a scale with a moving pointer. When the pointer reaches the value of the required length, the user imagines a push movement. The scale is then zoomed around this value. This is repeated thrice so that the user can select the exact value. The user gives another push signal to actuate the system and the results of the experiment are obtained.

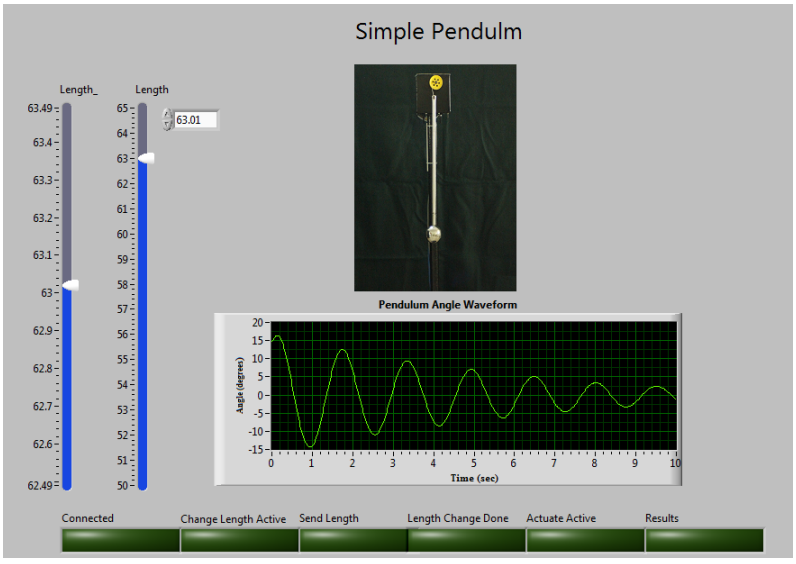


Figure 3.12: GUI for pendulum experiment

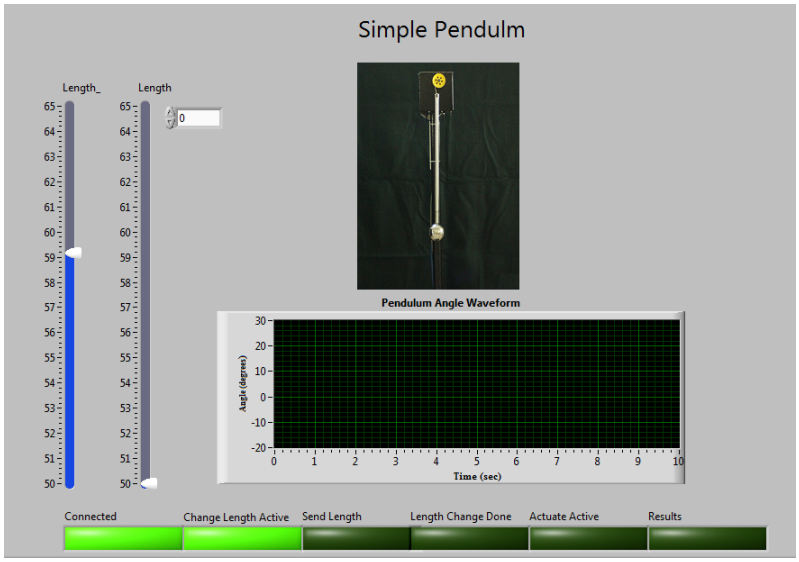


Figure 3.13: Setting the length of the pendulum

3.6.3 Controlling a robotic arm

This experiment consists of a robotic arm made using polypropylene, multiple servo motors and a micro-controller. It performs the action of rotating about its axis, picking up an item placed on its left or right and giving it to the user present in front of it.
This system can be used by a physically challenged person with arm disability to eat by controlling it using his thoughts. Two windows of 8 second duration each (corresponding to left and right direction) are used and during each window, the system expects the user to either think the intended direction (think left in the window allocated for left and right in the other window) or neutral. If the person thinks of a particular state continuously for 4 of the 8 seconds, then the action is performed by the robotic arm.

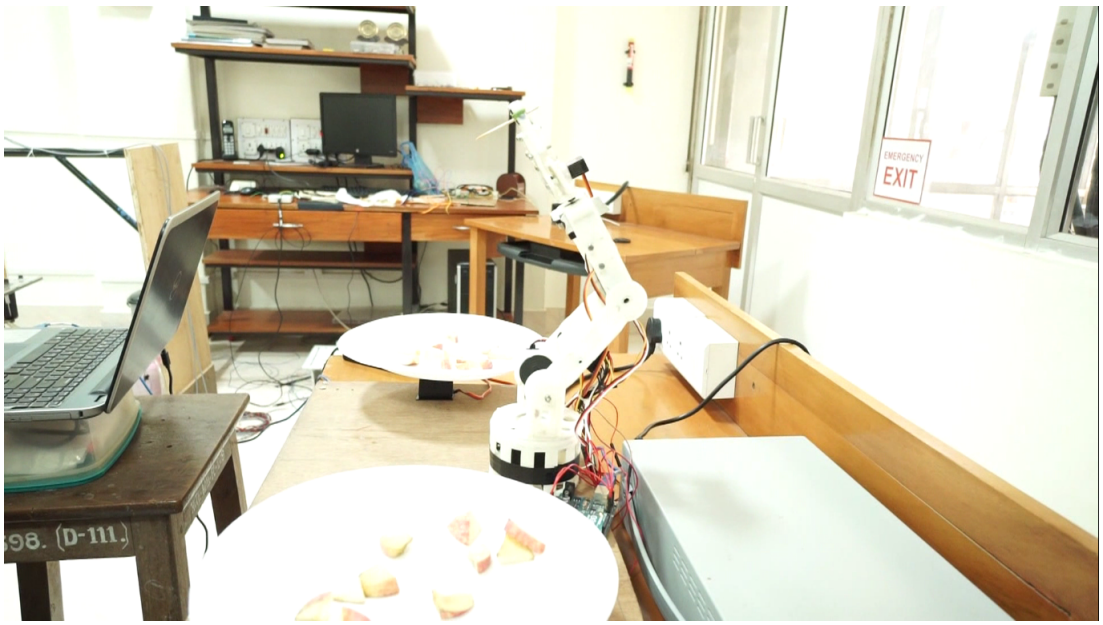


Figure 3.14: Robotic arm

Figure 3.15: Robotic arm picking a fruit

3.6.4 Typing

The experiment consists of a GUI comprising of a 6 x 5 matrix, consisting of alphabets and special characters like space, full stop, backspace and question mark. The aim of the experiment is for the user to type any desired word or sentence using the motor imagery tasks of push and neutral. The alphabets are chosen as follow: First, the selection window shifts from one column to another. If the desired alphabet is in a particular column, the user has to give an indication through push signal. A vertical bar with markings from 0 to 100 which depicts the strength of thought signal is present. As the person thinks continuously about push, the bar rises by 20 values per second. Once the indicator crosses 90, the particular column is selected. If the person has selected the column by mistake, the person can start thinking of neutral thought and this causes the value of the indicator bar to decrease by 10 values per second. If the value goes below 10, the column under consideration is deselected. Once the column is selected, the selection window navigates through each row of that column. The user has to give a push signal to select the alphabet. This process is carried on until the desired sentence is typed.

Figure 3.16: Selecting column

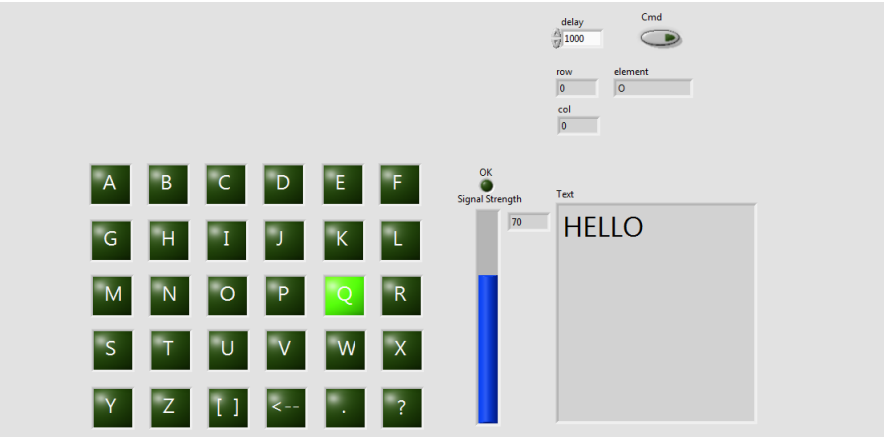


Figure 3.17: Selecting row

3.6.5 Home Automation

The experiment consists of a GUI indicating different devices and its state(on or off). Devices such as fan, table lamp, etc. were controlled. The aim of the experiment is for the user to switch on or off any of the four devices using the motor imagery tasks of push and neutral. The selection window shifts from one device to another giving an interval of 5 seconds in each window. If the user wants to switch on or off a particular device, he has to give an indication through the push signal within the time frame of 5 seconds allotted for that device. A vertical bar with markings from 0 to 100 which depicts the strength of thought signal is present. As the person thinks continuously about push, the bar rises by 20 values per second. Once the indicator crosses 90, that particular device is selected. If the person has started to push in the wrong time-frame, he has to make sure the indicator does not cross 90 and also for the selection window to shift, the indicator value should go below 10. This can be done by thinking of neutral which causes the value in the bar to fall by 20 values per second. Using this procedure, the on or off state of any of the device is set.

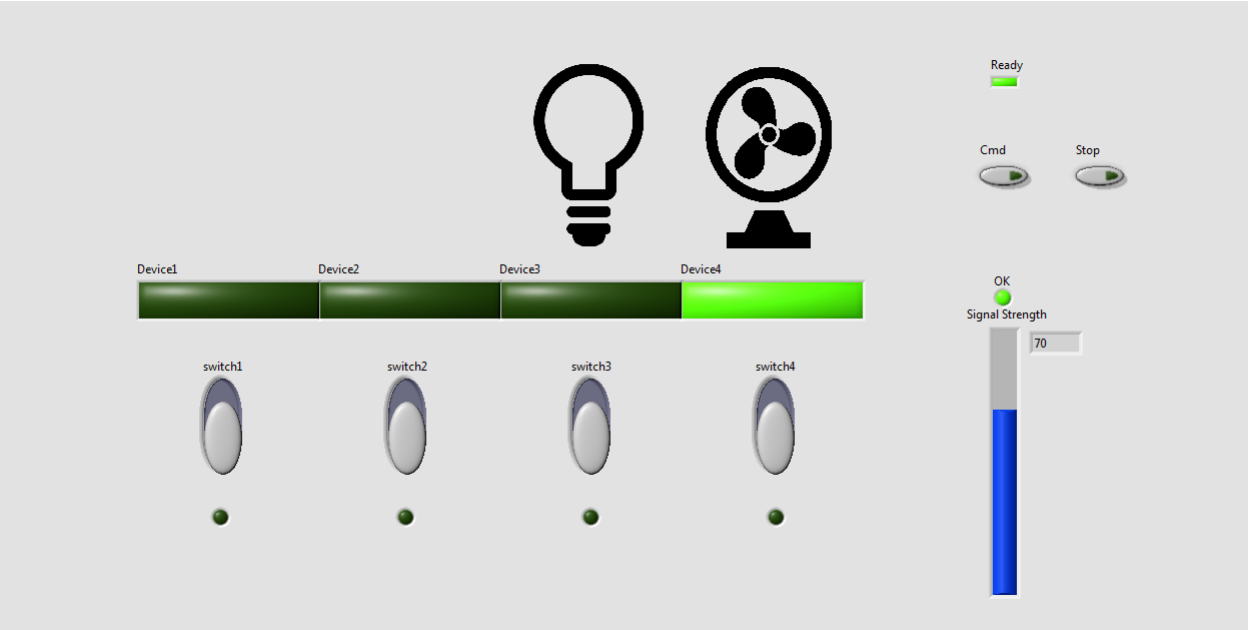


Figure 3.18: Selecting table fan

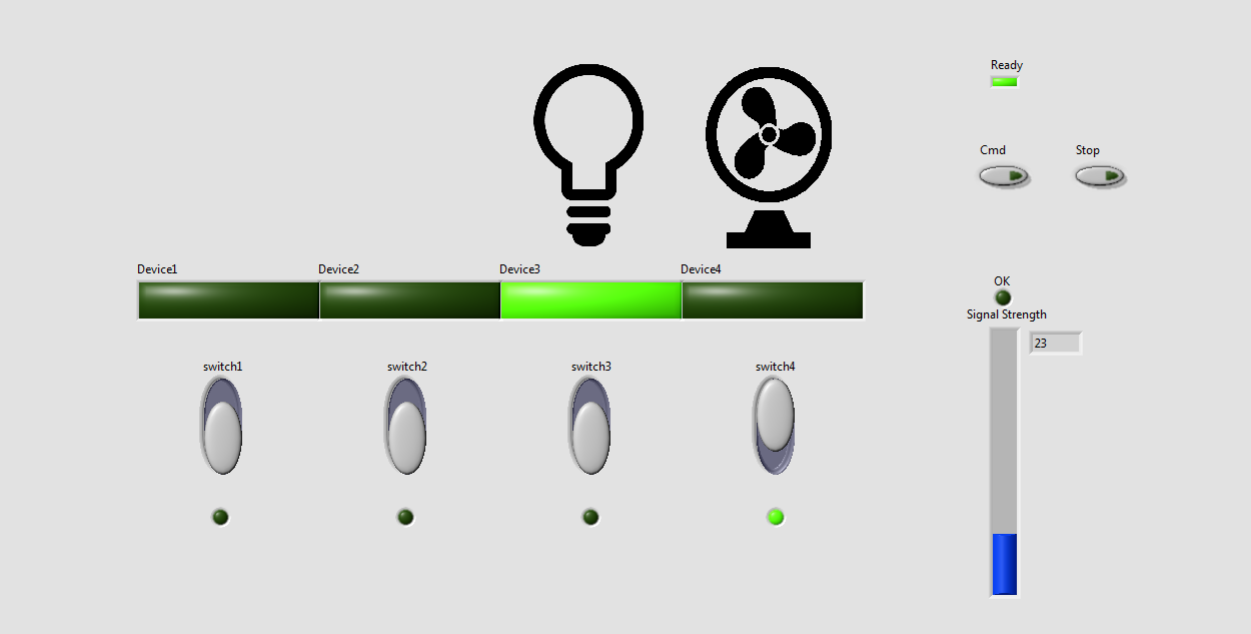


Figure 3.19: Selecting table lamp

CHAPTER 4
Results

4.1 Dataset and Single-Channel Headset

Percentage accuracy for 66-33 split of data

Filters	Dataset 1	Dataset 2	Headset Data
Logistic	76	60	75
SVM- linear kernel	60	56	80
SVM- RBF kernel	55	57	50
Random Forest	90	59	85
J48 tree	84	55	74

The following conclusions can be drawn:

- No particular classifier is suited for all the 3 datasets. The best result in different datasets is given by different classifiers.
- The accuracy is atleast 55%.

4.2 Multi-Channel Headset

- Motor imagery task: Imagine left and right hand movement
  - Within Subject
    - 3 classes:Left-Right-Neutral, 12.5s each

Classifier = Random Forest
Accuracy = 0.66
Confusion Matrix

L	**N**	**R**
105	60	10
36	211	20
16	64	88

    - 3 classes:Push-Pull-Neutral, 12.5s each

Classifier = Random Forest
Accuracy = 0.707
Confusion Matrix

P	**N**	**Pl**
100	32	27
40	181	25
29	12	119

    - 3 classes:Up-Down-Neutral, 12.5s each

Classifier = Random Forest
Accuracy = 0.619
Confusion Matrix

U	**N**	**D**
71	51	37
39	184	23
17	49	96

- Across subject
  - 3 classes:Left-Right-Neutral, 12.5s each

Classifier = Random Forest
Accuracy = 0.402
Confusion Matrix

L	**N**	**R**
57	63	38
71	129	45
56	64	40

  - 3 classes:Push-Pull-Neutral, 12.5s each

Classifier = Random Forest
Accuracy = 0.4219
Confusion Matrix

P	**N**	**Pl**
50	40	48
89	78	48
43	17	80

  - 3 classes:Up-Down-Neutral 12.5s each

Classifier = Random Forest
Accuracy = 0.401
Confusion Matrix

U	**N**	**D**
40	74	44
49	154	40
40	89	31

Visual imagery task: 3 subjects, 3 trial each

- Within subject
  - 2 classes: Active-Blank, 10s each

Classifier = random forest, Confusion Matrix, Accuracy = 0.86(615/715)

A	**B**
306	49
51	309

- Across subject
  - 2 classes: Active-Blank, 10s each

Classifier = logistic regression
Accuracy = 0.785(565/720)
Confusion Matrix

A	**B**
272	85
70	293

CHAPTER 5
Conclusion and Future Work

At the end of this project, EEG data analysis of existing datasets and recorded data, for various experiments such as motor imagery and visual stimulation, has been completed. Using the single channel headset, real-time attention state of the brain has been captured. Based on the results obtained from multi-channel headset, the classification is much better for an already trained person than for a new person. This clearly indicates that EEG signals for motor imagery are person dependant. Three different motor imagery tasks were done; Left-Neutral-Right, Push-Neutral-Pull, Up-Neutral-Down. Of the above tasks Push-Neutral-Pull gave better results compared to others. It can also be noted that most of the mis-classifications of non-Neutral classes are into Neutral, which would be better than mis-classifying into the complimentary action(e.g., 'Push' classified as 'Pull'). Visual stimulus response is good as it is an involuntary response. However auditory response failed to classify the two sounds and was almost random(50%), hence those results have been omitted. Many suitable BCI applications for disabled/paralysed patients has been implemented.

These are few improvements which can be considered in the future :

- Auto regressive (AR) methods can be used to estimate PSD of EEG using a parametric approach. AR methods don't have the problem of spectral leakage and thus yield better frequency resolution.
- Time domain processing can be studied to obtain and test on other features.

REFERENCES

[1] Mohammad H. Alomari, Aya Samaha, and Khaled AlKamha. *Automated Classification of L/R Hand Movement EEG Signals using Advanced Feature Extraction and Machine Learning*. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 4, No. 6, 2013

[2] A. Vallabhaneni, T. Wang, and B. He. *BrainComputer Interface*. Neural Engieering B. He, Ed.: Springer US, 2005, pp. 85-121.

[3] A. E. Selim, M. A. Wahed, and Y. M. Kadah. *Machine Learning Methodologies in Brain-Computer Interface Systems*. Biomedical Engineering Conference, 2008, CIBEC 2008. Cairo, 2008, pp. 1-5.

[4] G. Pfurtscheller, C. Neuper, D. Flotzinger, and M. Pregenzer. *EEG-based discrimination between imagination of right and left hand movement*. Electroencephalography and Clinical Neurophysiology, vol. 103, pp. 642-651, 1997

[5] C. Neuper and G. Pfurtscheller. *Evidence for distinct beta resonance frequencies in human EEG related to specific sensorimotor cortical areas*. Clinical Neurophysiology, vol. 112, pp. 2084-2097, 2001.

[6] Jason Sleight, Preeti Pillai, Shiwali Mohan. *Classification of Executed and Imagined Motor Movement EEG Signals*. Computer Science and Engineering, University of Michigan, Ann Arbor December 17, 2009

[7] V. Morash, O. Bai, S. Furlani, P. Lin; M. Hallett. *Classifying EEG signals preceding right hand, left hand, tongue, and right foot movements and motor imageries*. Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology, 119(11):2570-8, 2008

[8] Yijun Wang and Scott Makeig. *Predicting Intended Movement Direction Using EEG from Human Posterior Parietal Cortex*. HCI International, San Diego CA, July 2009

[9] Zoltan J. Koles, Michael S. Lazaret and Steven Z. Zhou. *Spatial patterns underlying population differences in the background EEG*. Brain topography, Vol. 2 (4) pp. 275-284, 1990.

[10] G. Pfurtscheller, C. Gugeret and H. Ramoser. *EEG-based brain-computer interface using subject-specific spatial filters*. Engineering applications of bio-inspired artificial neural networks, Lecture Notes in Computer Science, 1999, Vol. 1607/1999, pp. 248-254

[11] Dataset
`http://www.bbci.de/competition/iii/desc_V.html`

[12] Implementing CSP in MATLAB
`http://www.mathworks.com/matlabcentral/fileexchange/22915-common-spatial-patterns`

[13] The communication protocol used by neurosky thinkgear
http://developer.neurosky.com/docs/doku.php?id=thinkgear_communications_protocol#thinkgear_data_values

[14] The communication between the headset and arduino.
http://pantechsolutions.net/brain-computer-interface.html

[15] http://www.cs.waikato.ac.nz/ml/weka/

[16] http://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

[17] http://scikit-learn.org/stable/modules/ensemble.html#random-forests