

A Framework for the Integration of the Emotiv EEG System into the NeuCube Spiking Neural Network Environment for Robotics Control

Jianfei Wang

A thesis submitted to Auckland University of Technology in partial fulfillment of the requirements for the degree of Master of Computer and Information Sciences (MCIS)

2014

Faculty of Design and Creative Technologies
School of computing and Mathematical science

Supervisors:

Professor Nikola Kasabov

Mr. Nathan Scott

Abstract

As BCI has been widely studied in recent years, lots of BCI applications have been developed. In this thesis, to implement BCI into individual devices is simulated by controlling a robot. Before discussing the experiment, a literature review is outlined. As the experiment is based on the Emotiv system, a framework of implementing the Emotiv system on NeuCube is stated. The objective is to develop a new feature on NeuCube, which can record EEG data in real-time through the Emotiv headset. The limitations and potential future research areas are highlighted at the end.

Contents

Abstract	1
List of figures	4
List of tables	6
List of listings	7
List of Abbreviations	8
Attestation of Authorship	9
Acknowledgments.....	10
Chapter 1. Introduction.....	11
1.1. Motivation	11
1.2. Research Focus	12
1.3. Study Question	12
1.4. Contribution of the Study	12
1.5. Study Structure	12
Chapter 2. Literature review	14
2.1. Review of BCI.....	14
2.1.1. Invasive BCI and Part-invasive BCI.....	14
2.1.2. Non-invasive BCI	15
2.2. BCI system using EEG	17
Chapter 3. Methodology	20
3.1. EEG Detector	21
3.2. Digitization Module	21
3.3. EEG Data.....	21
3.4. Classification Algorithms.....	22

3.4.1. Linear Discriminant Analysis (LDA) Algorithm.....	22
3.4.2. Support Vector Machine (SVM) Algorithm.....	23
3.5. Analyzing Module.....	24
3.6. Controlling Software	24
Chapter 4. An experiment by using the Emotiv built-in system	26
Chapter 5. A Framework to Implement the Emotiv system on NeuCube	29
5.1. The NeuCube System.....	29
5.1.1. Overall Architecture and Main Principles	29
5.1.2. Input Encoding Module	30
5.1.3. The NeuCube Module.....	31
5.1.4. Output Function Module.....	32
5.2. Framework of implementing Emotiv in NeuCube	32
5.2.1. Emotiv input module	33
5.2.2. NeuCube input module	36
Chapter 6. Conclusion and Future Works	38
6.1. Conclusion and contribution	38
6.2. Limitations and Future Works.....	38
References.....	40
Appendices.....	43
A. WITH robot console pseudocode (robot.py) (Stefan Schliebs, 2012).....	43
B. robot console HUD pseudocodes (Key_events.py) (Stefan Schliebs, 2012).....	43
C. EEG Logger pseudocodes (EEG_Logger.cpp) (Emotiv system, 2014, Edited by Jianfei Wang)	44

List of figures

Figure 2.1. Jens Naumman with invasive BCI

Figure 2.2. EEG detector and EEG spike graph

Figure 2.3. four-blade helicopter controlled by EEG-BCI

Figure 2.4. A general structure of BCI system

Figure 3.1. Structure of an EEG-BCI system

Figure 3.2. Emotiv headset, EEG detector

Figure 3.3. A hyperplane created by the LDA algorithm to split the data into two classes

Figure 3.4. Overview of linear SVM algorithm

Figure 3.5. The WITH robot produced in Kyushu Institute of Technology

Figure 4.1. Emotiv EEG headset

Figure 4.2. Emotiv software environment

Figure 4.3. Framework of Emotiv experiment.

Figure 4.4. Controlling WITH robot with Emotiv system at KEDRI

Figure 5.1. A diagram of the general NeuCube architecture, consisting of: input data encoding module; NeuCube module; output function module (e.g., for classification or prediction); and gene regulatory networks (GRN) module (optional).

Figure 5.2. (a) Population rank order coding of input information; (b) Address Event Representations (AER) of continuous value time series input data into spike trains and recovery of the signals from the spike train.

Figure 5.3. Illustrative visualization of connectivity and spiking activity of a SNNr: (a) spiking activity-active neurons are represented in a large area of red; (b) connectivity before training – small world connections – positive connections are represented in blue and negative- in red; (c) connectivity after training; (d) connectivity of a NeuCube with 2.4M neurons; (e) zoomed-in visualization of connectivity of a small internal section from the NeuCube.

Figure 5.4. Setting input mapping manually in NeuCube.

List of tables

Table 4.1. Robot controlling commands set

List of listings

Listing 5.1. Connect to EmoEngine

Listing 5.2. Buffer creation and management.

Listing 5.3. Access to EEG data

Listing 5.4. Start Acquiring Data.

Listing 5.5. Acquiring Data

Listing 5.6. Emotiv data channels

List of Abbreviations

Address Event Representation (AER)

Amyotrophic Lateral Sclerosis (ALS)

Ben's Spike Algorithm (BSA)

Brain Computer Interface (BCI)

Common Spatial Pattern (CSP)

Electroencephalography (EEG)

Functional Magnetic Resonance Imaging (fMRI)

Gene Regulatory Networks (GRN)

Human Computer Interaction (HCI)

Knowledge Engineering and Discovery Research Institute (KEDRI)

Leaky-Integrate and Fire Model (LIFM)

Linear Discriminant Analysis (LDA)

Magnetoencephalography (MEG)

Multilayer Back Propagation neural network (MBP)

Optimum Spatio-Spectral Filtering Network (OSSFN)

Personal Computer (PC)

Regularized Tensor Discriminant Analysis (RTDA)

SNN Reservoir (SNNr)

Sparse Common Spatial Pattern (SCSP)

Spatio- and Spectro-Temporal Brain Data (STBD)

Support Vector Machine (SVM)

Wavelet Packet Transform (WPT)

Attestation of Authorship

“I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial degree has been submitted for the award of any other degree or diploma of a university or other institution of higher learning”.

Jianfei Wang

September, 2014

Acknowledgments

At the end of my master study, I wish this acknowledgment to show my appreciation to the people who have helped me a lot. I'm grateful for 14 months' study, which has been a memorable part of my life.

First of all, I am truly grateful to my primary supervisor, Professor Nikola Kasabov. The comments he has given to me have been valuable, and his inspiration has kept me carrying on until I finished this thesis.

Many thanks on my secondary supervisor, Mr. Nathan Scott. He has helped me on Emotiv and NeuCube and given me the chance to understand my study field in depth.

Also, lots of thanks to KEDRI; I am proud to be a member and to have studied there. I wish to thank Joyce D'Mello, the manager of KEDRI, for the selfless support. And many thanks to other members of KEDRI; it was great to study with them.

I wish to express my sincere appreciation to Dr. William Liu, who encouraged me a lot and gave me lots of valuable advice.

Grateful acknowledgments to my best friends: Alex, Roger, Yichen, Elven, Seriena, and Yang. There were great times when I stayed with them, and they helped me relax.

At last, I wish to express my appreciation to my father and my mother. They encouraged me to keep stepping forward, whatever difficulties I was facing.

Chapter 1. Introduction

Human-Computer Interaction (HCI) is the communication between humans and computers. HCI has been developed extensively for decades, from cards and records in early days to mouse and touchpads in recent times. Novel interfaces have been studied and developed for humans to have a better user experience on controlling computers.

Stephan and Khudayer (2010) showed a design of a gesture recognition HCI prototype based on shape analysis for PC applications. Rogers (2012) gave an overview on HCI, including gesture recognition, speech recognition, and Brain-Computer Interface (BCI). As the gesture recognition and speech recognition have been widely implemented on terminals such like PC, pads, and smartphones, BCI is still not satisfactory for industrial application. However, BCI is a rich field for studying. In 2010, Tan and Nijholt defined BCI as the real time interaction system which builds a direct communication panel between the human brain and computers. The first appearance of BCI in scientific literature was in the 1970s by Vidal (1973). After 40 years of studying, researchers have developed several systems for BCI application, including the NeuCube system.

NeuCube is a system used for mapping, learning and analyzing spatio-temporal brain data. It was introduced by Nikola Kasabov and developed by Knowledge Engineering and Discovery Research Institute (KEDRI). There is an introduction to NeuCube in Chapter 5, and the development of a new feature for NeuCube is the main focus of this thesis.

In the rest of this chapter, the motivation of this study is presented, as well as the research focus. The research questions are discussed following that. Then the contribution of this study is pointed out. Finally, the structure of this thesis is shown.

1.1. Motivation

BCI is a novel communication method that connects humans and computers. People can control computers using BCI without verbal or body language, including with the original input methods, such as mouse, keyboard or touch screen. With this feature, disabled people are able to operate computers by themselves. Over 2 million patients suffer from Amyotrophic Lateral Sclerosis (ALS) in America, and there are 5000 more cases diagnosed annually (Felbecker, 2010). The patients with ALS cannot move their bodies, or

even make any sounds. The BCI is one of the solution that helping patients express themselves.

1.2. Research Focus

As NeuCube is designed for spatio- and spectro-temporal brain data (STBD), it could be helpful to fully develop the system, making it work as a BCI system. To build a feature which reveal EEG data on the NeuCube system is developing a potential study area. To control a robot by using a NeuCube based BCI system is what this study is focusing on.

1.3. Study Question

As mentioned previously, the main objective of this study is to control a robot, which simulates a wheel chair, by using the NeuCube system, and to describe the framework. Consequently, the study questions are as following:

1. What is BCI? How does it work?
2. How can a robot be controlled by using BCI?
3. How can a robot be controlled by using NeuCube with EEG recording devices?

1.4. Contribution of the Study

The contributions of this study are generalized below:

A critical literature review is given, which gives an overview of the existing BCI applications and discusses the technologies.

An experiment by which to control a robot using the Emotiv system is discussed.

A framework implementing the Emotiv headset (EEG data detector) to the NeuCube system is shown.

1.5. Study Structure

In this part, the structure of this thesis is stated below:

In Chapter 2, an overview of BCI technology is provided. The definition, principle, and the latest developments are described. In addition, several examples of existing BCI research are introduced.

Chapter 3 discusses a BCI system prototype. The methodology of design of a BCI system is shown.

Chapter 4 summarizes an experiment to control a robot by using the Emotiv BCI system.

In Chapter 5, the NeuCube system is described particularly, and a framework of implementing NeuCube with an EEG data recording feature is stated.

Finally, the conclusion is given in Chapter 6. Both contributions and limitations of this study are highlighted. In addition the potential areas of future research are also pointed out.

Chapter 2. Literature review

BCI is a communication method which creates a direct connection path between a human (or animal) brain and external devices. In this definition, the “brain” refers to an organic form of a real brain or nervous system, not the abstract “mental mind”. The “computer” means any processing or computing device, which can be implemented as a simple circuit to a computer system. In this chapter, a brief review of BCI studies will be shown.

2.1. Review of BCI

In general, BCI can be assorted into three types: invasive, part-invasive, and non-invasive.

2.1.1. Invasive BCI and Part-invasive BCI

Invasive BCI is mainly used for the reconstruction of special senses, vision for example, or motor function in paralyzed patients. As the invasive BCIs are typically implanted into the ectocinerea directly, the quality of acquitted neural signals is relatively high. Part-invasive BCI is generally implanted into the skull, but located outside the ectocinerea.

In 2002, Jens Naumann, who was blinded in adulthood, implanted an invasive BCI device to help him rehabilitate his vision. The device was developed by William Dobelle, one of the first scientists in the BCI area. The device was a single-array BCI containing 68 electrodes that was implanted onto Naumann’s brain to produce phosphenes (sense of light) a short time after the device was implanted.

Naumann was able to drive an automobile slowly at the research institute. (Naumann, 2012)

However, the invasive BCI needs users to have surgery to implant the chips into the body. Also, some patients’ bodies may reject the electrodes, which means the invasive BCI does not work. The most important reason why invasive BCI is not widely used is that the invasive BCI devices and the surgery are too expensive, while the non-invasive

BCIs are cheaper and portable. This is a main reason why this study focuses on EEG based BCI.



Figure 2.1. Jens Naumman with invasive BCI (Wikipedia, 2014)

2.1.2. Non-invasive BCI

Non-invasive BCI devices are not implanted into the human body. They use non-invasive neuroimaging technologies, such like Electroencephalography (EEG), Magnetoencephalography (MEG) and functional magnetic resonance imaging (fMRI). In this way, the BCI devices are easy to wear and do not hurt the user's body generally. However, the signal can be reduced by skull and skin disturbance.

BCI based on EEG has been studied extensively as a great potential solution to reduce the cost because of the good temporal resolution, ease of use, low cost and portability. But EEG data would be disturbed easily by any user's motion or lost attention. This requires the user practicing hard for operating the BCI system proficiently.

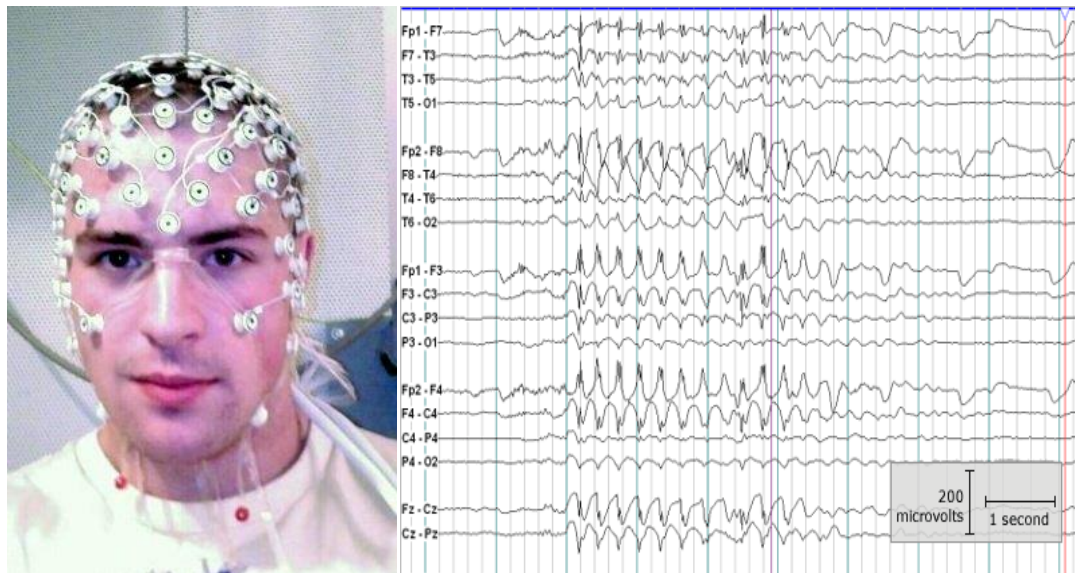


Figure 2.2. EEG detector and EEG spike graph (Wikipedia, 2014).

In 2000, a study based on the P300 signal showed that users without specialized training can use the BCI system. The P300 signal is generated autonomously when people see familiar objects. Bayliss(2000) showed that subjects can control some objects in virtual reality though the P300 signal; for example, switching a light on or off, or manipulating a virtual car.

By applying an artificial neural network, a computer can share the patient's burden of learning BCI. The Fraunhofer Society (2004) reduced required training time of BCI systems significantly by using artificial neural networks.

Doud et al. (2011) designed a BCI system which can control a virtual helicopter on a computer simulation. And in June 2013, the research team finished the final built of the system to control a four-blade helicopter to go through mission path. The system only

used motor imagination to control the helicopter with high accuracy.



Figure 2.3. four-blade helicopter controlled by EEG-BCI (Doud, et al., 2013)

Magnetoencephalography (MEG) and functional magnetic resonance imaging (fMRI) have both proved that they are good carriers to achieve a functioning BCI system. However, the drawbacks are easy to find. The MEG detector is too huge to relocate and the time delay of fMRI is too long.

2.2. BCI system using EEG

Due to the advantages, EEG-BCI is the richest study area of BCI. In this section, more information of EEG-BCI will be reviewed.

Early in 2004, Schalk and his team created a model for general-purpose BCI system. He called this platform BCI2000.

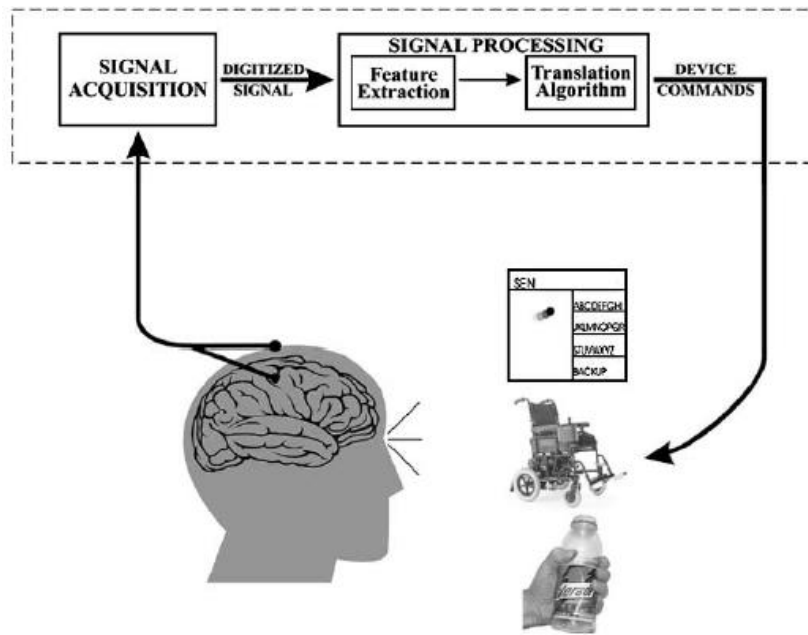


Figure 2.4. A general structure of BCI system (Schalk et al., 2004).

Figure 2.4 shows the basic structure of a BCI system. Researchers use a device to detect the human brain signal, then, the digitized signal will be processed. In the processing, the feature of an EEG graph will be extracted, and this allows the computer to classify brain activity. With a contrast about the feature and datasets, the “idea” of the human brain will be translated into device commands, to order the devices to finish this interaction.

Schalk et al. (2004) also noted that the main goal of BCI development is to offer a solution to people with severe motor disabilities to improve communication and control capabilities. The whole system including classification of brain signals, recording methods, processing algorithms, output formats, and operating protocols should be studied and improved. And the model could be uniform. Schalk et al. (2004) developed the BCI2000 system as a general-purpose BCI research and development platform. Several successful BCI systems based on the BCI2000 platform were implemented for a variety of brain signals, processing algorithms and applications. The six essential features of the BCI2000 system are: (1) Common model. BCI2000 is designed to be a standard prototype which can be implemented by different methods. (2) Interchangeability and independence. The modules of BCI2000 can be exchanged with other systems. (3) Scalability. BCI2000 can be updated to improve the performance. (4) Real-time capability. BCI2000 can read and process the brain signals in real time and with high accuracy. (5)

Support and facilitation of offline analyses. The data needs to be stored and can be analyzed without converting the storage format. (6) Practicality. BCI2000 is very easy to implement with different approaches.

However, the Signal-to-Noise ratio of EEG graph is low; it not easy to extract the features with high accuracy using some simple methods. Researchers are focusing on improving the accuracy of EEG graph classification. Arvaneh (2011) noticed that optimizing the channel selection with a novel sparse common spatial pattern (SCSP) algorithm could increase the accuracy of the EEG classification by about 10%. Another improvement was provided by Li and Zhang (2010). They discovered that using a regularized tensor discriminant analysis (RTDA) algorithm to extract tensor-represented EEG data was highly accurate.

In Li and Zhang's study, the EEG signals were converted into third order tensors in the spatial-spectral-temporal domain using wavelet transform. After this, a multi-way discriminative subspace extraction from EEG data was proposed by using a RTDA algorithm. Li and Zhang also maintained the Superiority of their scheme more than the Common Spatial Pattern (CSP) algorithm: First, the optimal multi-way discriminative subspace could be extracted; second, it could identify discriminative characteristics robustly and do well without prior neurophysiologic knowledge; third, it was able to find the most significant channels from EEG data for classification. As a result, the proposed scheme provided a great increase on performance.

The accuracy can be improved both in feature extraction and feature classification. For years, researchers have been testing different algorithms to find better algorithms for different requirements. Vijay (2010) compared three classifiers' performances, which were Multilayer Back Propagation Neural Network (MBP), Support Vector Machine (SVM) and Radial Basis Function Neural Network (RBF). The feature extraction was based on Wavelet Packet Transform (WPT). The RBF neural work had the best performance in his research with 100% accuracy.

Chapter 3. Methodology

In Chapter 3, the framework of the BCI controlling system will be introduced. The main structure is shown below:

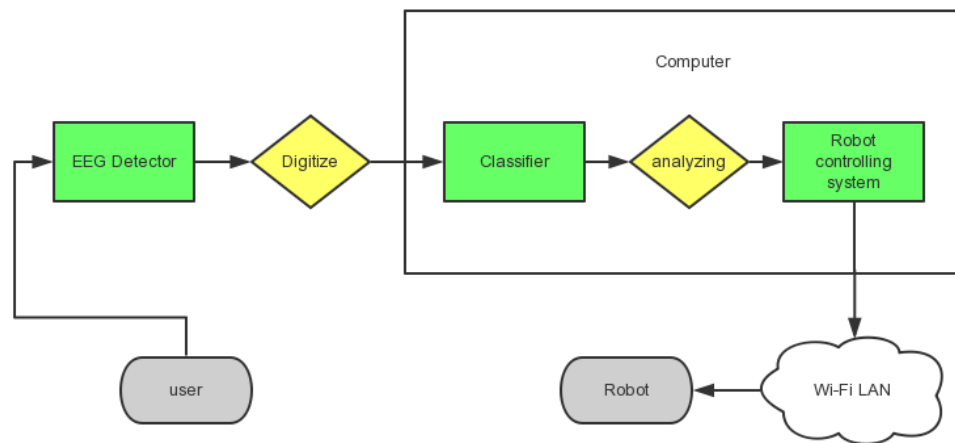


Figure 3.1. Structure of an EEG-BCI system

The process of the whole system is as follows:

3.1. EEG Detector



Figure 3.2. Emotiv headset, EEG detector.

First step is the EEG detector acquires EEG signal from user and digitizes the signal into digital data. The EEG is a record of electrical activity along the scalp. The EEG detecting device can measure voltage changes of human brain neurons, resulting in curve graphs. The detector of EEG data uses electrodes to record brain spikes. The spikes will be converted into digital data at the terminal, then the data will be stored in the system buffer or form files.

3.2. Digitization Module

As digitization, real-number signals of continuous value are converted into spike trains. The EEG data is normalized into the range of $[0\ 1]$ firstly. Then the normalized data is transformed into spike trains with Ben's Spike Algorithm (BSA) encoding method. (Hu et al., 2014)

3.3. EEG Data

Second step, the pre-processed digital EEG data is sent to the classifier to identify the class of it. The captured EEG data will be pre-processed first. Feature extraction algorithms will be used in this step to increase the value of the data, including decreasing the computation time, the detector preparation time and the noise of data. For example, the

Optimum Spatio-Spectral Filtering Network (OSSFN) could increase the accuracy by 10-36% with minimized costs (Haihong et al., 2011).

3.4. Classification Algorithms

Third step, the data will be classified. The classification algorithms can be Linear Discriminant Analysis (LDA) algorithm, SVM algorithm, or NeuCube. The LDA algorithm and SVM algorithm are well-known mining algorithms, and they are widely used in BCI applications. These two algorithms are introduced in detail below.

3.4.1. Linear Discriminant Analysis (LDA) Algorithm

LDA is a well-known mining algorithm. It has been widely applied in numerous different classification problems. It is simple to use and requires a very low computational time.

According to Yoon, Roberts, Dyson and Gan (2011), the LDA classifier aims to separate the dataset into two classes using hyper planes. The class of a sample is determined by the side of the hyper plane in which a sample, or a feature vector, is placed on, as shown in Figure 3.3.

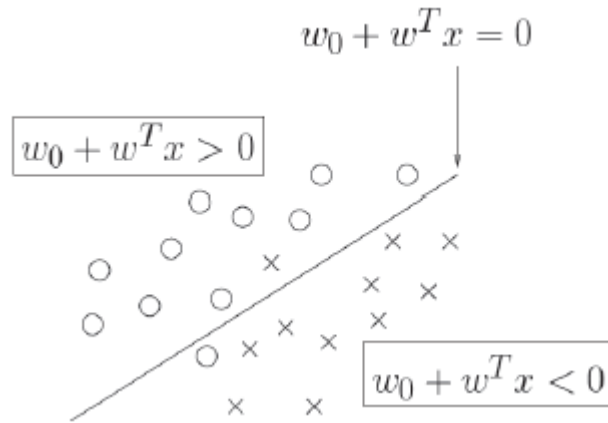


Figure 3.3. A hyperplane created by the LDA algorithm to split the data into two classes. (Yonn et al., 2011)

The LDA algorithm assumes normal distribution of the conditional probability density function $P(\vec{X}|y = 0)$, and mean of $P(\vec{X}|y = 1)$. Based on that, samples are associated with the second class if the log-likelihoods ratio s is below the scale T .

$$(\vec{X} - \vec{M}_0)^T \sum_{y=0}^{-1} (\vec{X} - \vec{M}_0) + \ln \left| \sum_{y=0} \right| - (\vec{X} - \vec{M}_1)^T \sum_{y=1}^{-1} (\vec{X} - \vec{M}_1) - \ln \left| \sum_{y=1} \right| < T$$

3.4.2. Support Vector Machine (SVM) Algorithm

Linear SVM is a powerful classification scheme. It was originally introduced by Vapnik (1998). This algorithm has been successfully applied in different BCI applications and general classifying problems.

To classify a set of binary labeled data, this algorithm also uses a hyper plane to separate the data into two classes. After training the algorithm on a given dataset, the discriminate hyper plane is optimized and selected based on the maximum margins between the hyper plane and the data. This is done through transforming the data from the input space into feature space in which linear classification is achievable. This can be achieved through accommodating outliers and allowing errors during the training stage (Bashashati, Fatourehchi, Ward, and Birch, 2007); see Figure 3.4.

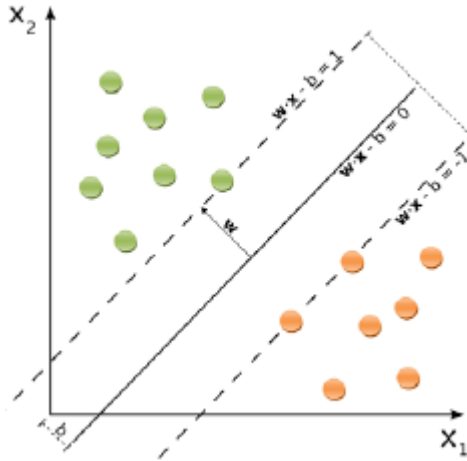


Figure 3.4. Overview of linear SVM algorithm (Liang, 2009).

SVM is mathematically formulated to the following formula to classify the data set D :

$$D = \{((x_1, y_1), \dots, (x_i, y_i)) | x \in R^n, y \in \{-1, 1\}\}_{i=1}^m$$

where x is the n -dimensional vector. The class' name that x belongs to is represented by y , and the hyper plane is defined by the following equation:

$$W^T(w_1, \dots, w_n) \times x + b = 0$$

in which w is the weight vector: $w(A) = \min C(w, b, A)$ and b is the scalar.

3.5. Analyzing Module

After classification, the digitalized EEG data will be analyzed and compared with an existed EEG mapping dataset, which includes classified EEG trigger data. The incoming data will be checked with flag events, as the events triggered, commands sent to the controlling application.

3.6. Controlling Software

Finally, the classified results will be sent to the robot-controlling software. After the data is classified, flags will be set as action triggers. The EEG readings when user blinks, smiles, or any other activity, could be used to make the robot work. The robot, a WITH robot produced by Kyushu Institute of Technology for example, will receive the commands sent from the console system. With different commands received, the robot can react, like move or turn.

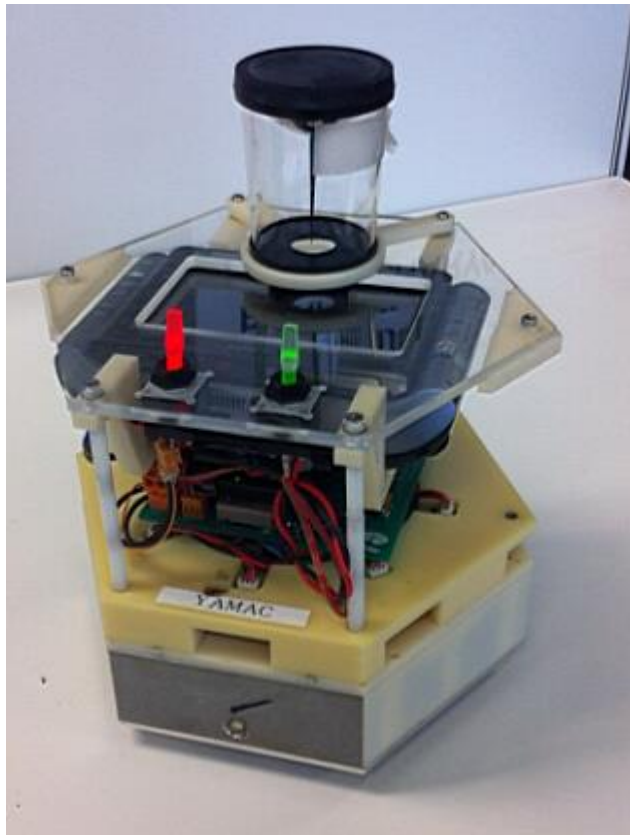


Figure 3.5. The WITH robot produced by the Kyushu Institute of Technology.

In the console system, the trained model is used as the commands list. When the EEG headset records the data, the console system analyzes this data to communicate with the robot. When one trigger happens, the console sends a combination command to the robot, and the robot can act following the command. More details of the WITH robot will be described in Chapter 4.

Chapter 4. An experiment by using the Emotiv built-in system

The first experiment was implemented by using the Emotiv system.



Figure 4.1. Emotiv EEG headset (Emotiv Systems, 2013)

Emotiv Systems (2013) developed this EEG system as an individual BCI solution. It is very easy to setup and is portable. The headset has 14 saline sensors with optimal positioning to capture the spatial EEG signal. The headset is wireless and connect to the computer with an USB dongle.

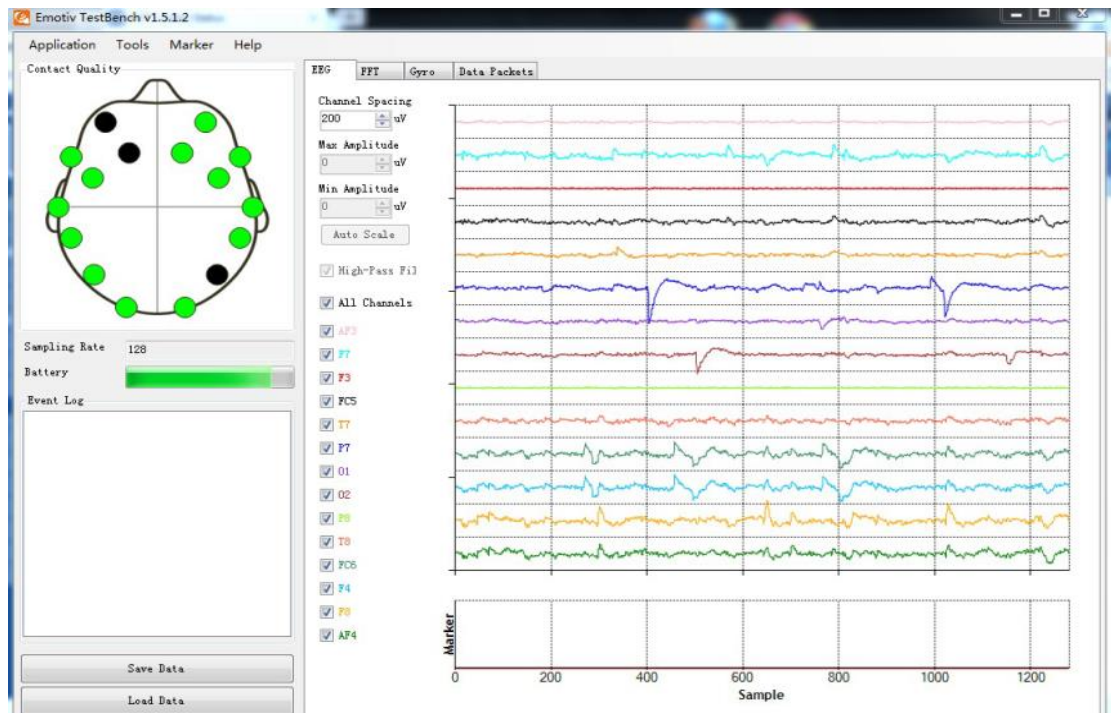


Figure 4.2. Emotiv software environment (Emotiv Systems, 2013)

The classifier of this system is EmoKey, a classification program provided by Emotiv Systems. It includes a common expression classification database for users, which allows users to use the system without any particular training.

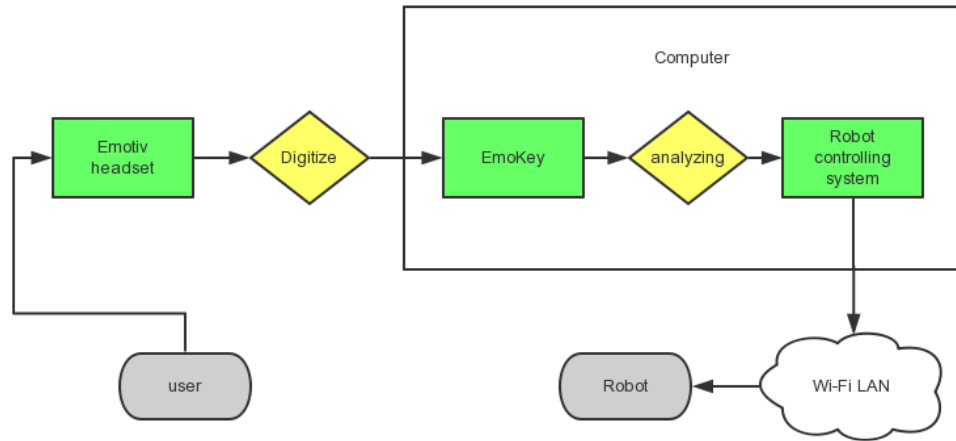


Figure 4.3. Framework of Emotiv experiment.

As the EmoKey database mainly works on expressions, the triggers to robot controlling commands are shown below.

Table 4.1. Robot controlling Commands set

Command	Trigger
Move forward	Frown
Stop	Blink once
Turn left	Looking left
Turn right	Looking right



Figure 4.4. Controlling WITH robot with Emotiv system at KEDRI.

As shown in Figure 4.4, the experiment succeeded. The WITH robot was controlled by the Emotiv system to move and turn. However the result was not satisfactory. By using the common expression classification database, the controlling accuracy was far from satisfactory. Generally, the system worked well on moving forward and stopping, but it was confused on turning. In addition, the emotion of the user had an impact on the result. In a perfect state, the user should focus his mind on controlling the robot and not be distracted.

Chapter 5. A Framework to Implement the Emotiv system on NeuCube

Human brains work like a spatio-temporal information processing machine. Generally, spatio- and spectro-temporal brain data (STBD) are collected to measure brain activities. There are various STBD in human brain studies, such as EEG, MEG, fMRI, DTI, genetic and NIRS. According to Kasabov (2014), the spiking neural networks (SNN) are considered suitable to create a computational framework for learning and understanding STBD. The NeuCube system is one of them. In this chapter, the NeuCube is introduced in detail.

5.1. The NeuCube System

The main objective of creating NeuCube system is integrating different modules in one system to recognize brain signals, while the system can consist of different neuronal types and genetic parameters from different parts of the human brain and different senses.

5.1.1. Overall Architecture and Main Principles

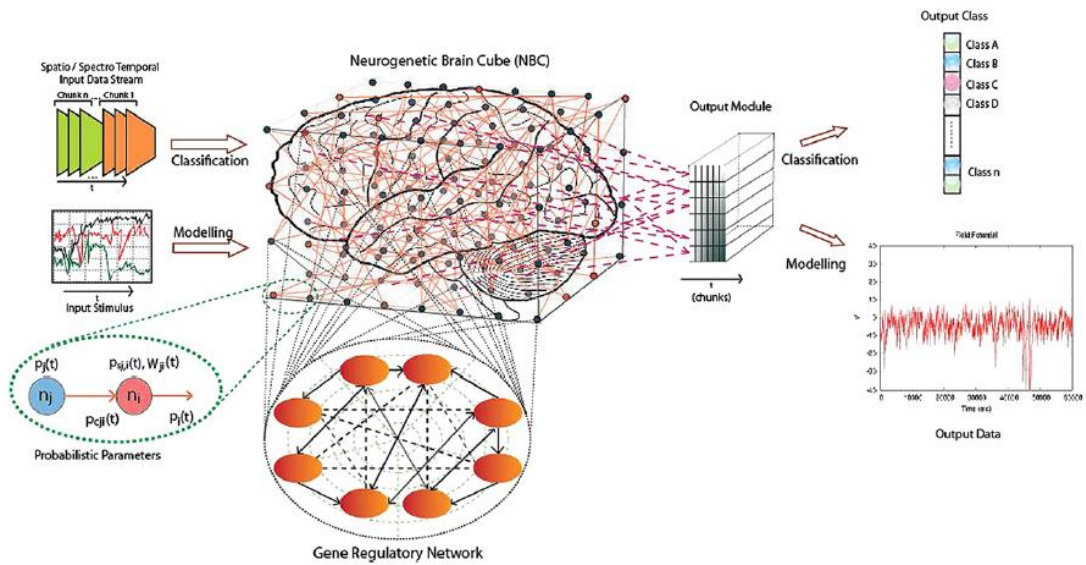


Figure 5.1. A diagram of the general NeuCube architecture, consisting of: input data encoding module; NeuCube module; output function module (e.g., for classification or prediction); and gene regulatory networks (GRN) module (optional) (Kasabov, 2014).

As shown in Figure 5.1, a NeuCube system includes four modules: input data encoding module; 3D SNN reservoir module (SNNr); output function (classification) module; and gene regulatory network (GRN) module (optional).

The principle of generating a NeuCube model for a given STBD is as follows:

- a. Encode the STBD into spike sequences: Continuous value input information is encoded into trains of spikes;
- b. Construct and train in an unsupervised mode a recurrent 3D SNN reservoir (SNNr), to learn the spike sequences that represent individual input patterns;
- c. Construct and train in a supervised mode an evolving SNN classifier to learn to classify different dynamic patterns of the SNNr activities that represent different input patterns from SSTD that belong to different classes;
- d. Optimize the model through several iterations of steps (a)–(c) above for different parameter values until maximum accuracy is achieved;
- e. Recall the model on new data (Kasabov, 2014).

The detailed description of each module is given below.

5.1.2. Input Encoding Module

The main work of the input encoding module is converting input data into trains of spikes, and the converted spike trains will be sent to the corresponding areas of the NeuCube system. There are two methods for transferring from STBD to spike trains: population rank coding and Address Event Representation (AER).

In the population rank coding method, continuous value input data is transferred into spikes. The value of different input variables, such as pixel, EEG channel, and voxel, is joined a population of spiking neurons emitting spikes by their receptive fields (Figure 5.2 a). The higher the membership degree, the earlier a spike is generated (Bohte, 2004).

Population rank coding is suitable when the input data is presented as a sequence of frames.

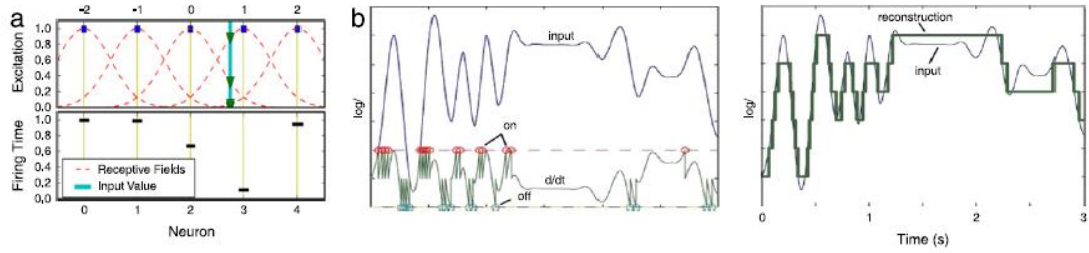


Figure 5.2. (a) Population rank order coding of input information; (b) Address Event Representations (AER) of continuous value time series input data into spike trains and recovery of the signals from the spike train (Delbruck, 2007).

The AER is based on thresholding the difference between two consecutive values of the same input variable over time (Delbruck, 2007). AER is suitable for input data stream, as it is only processing the changes in consecutive values.

5.1.3. The NeuCube Module

The structure of the NeuCube is a 3D approximate map of brain areas of the senses. The connections of neurons are initialized by small-world connections. The structure can be expanded by creating new neurons and new connections based on the STBD with the ECOS principles (Kasabov, 2014). The latest version of the NeuCube system can create a 3D structure connecting Leaky-Integrate and Fire Model (LIFM) spiking neurons with recurrent connections. Figure 5.3 shows how NeuCube visualizes converting spiking activity into neurons and connectivity.

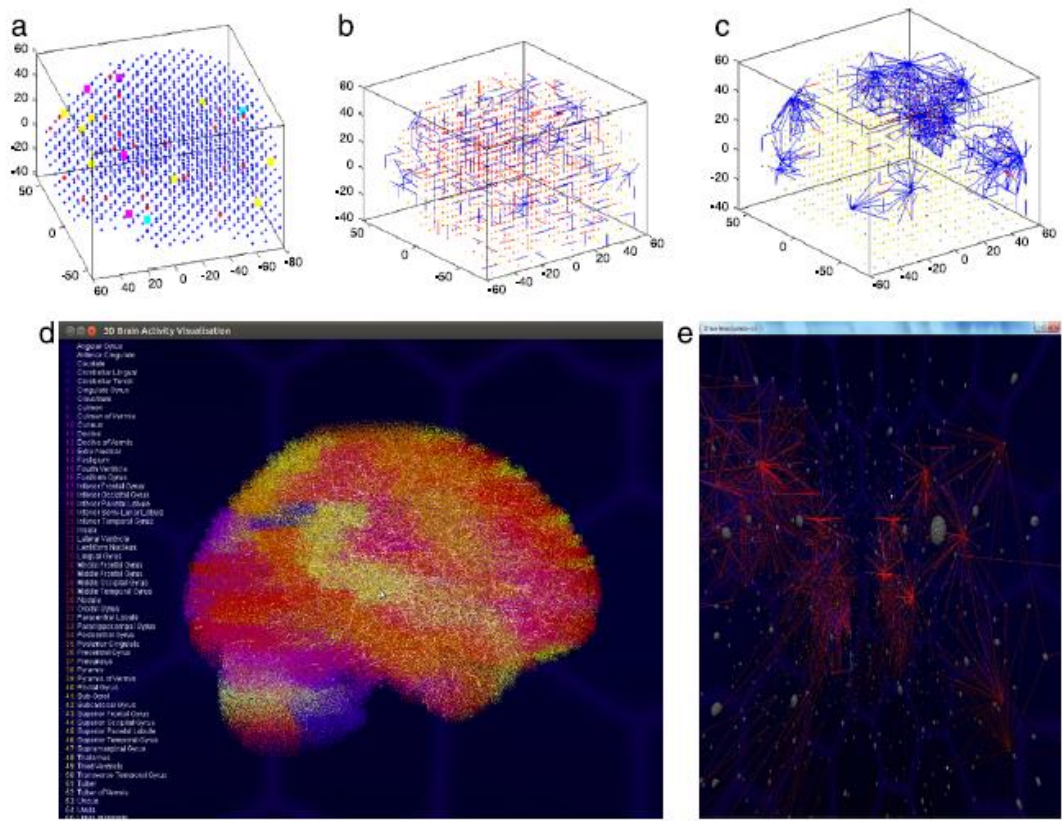


Figure 5.3. Illustrative visualization of connectivity and spiking activity of a SNNr: (a) spiking activity-active neurons are represented in a large area of red; (b) connectivity before training – small world connections – positive connections are represented in blue and negative- in red; (c) connectivity after training; (d) connectivity of a NeuCube with 2.4M neurons; (e) zoomed-in visualization of connectivity of a small internal section from the NeuCube (Kasabov, 2014).

5.1.4. Output Function Module

After the mapping is completed in the NeuCube module, the input STBD is learned. These NeuCube patterns will be associated in a classifier of the output function module, or communicated to other terminals.

5.2. Framework of implementing Emotiv in NeuCube

In this section, a framework to connect the Emotiv input headset with the NeuCube input module is stated.

5.2.1. Emotiv input module

The Emotiv system is easy to develop. The Emotiv headset has 14 channels to collect EEG data. An EEG logger will be designed to record EEG data by C++.

```
// ... print some instructions...
std::string input;
std::getline(std::cin, input, '\n');
option = atoi(input.c_str());
switch (option) {
case 1: {
if (EE_EngineConnect() != EDK_OK) {
throw exception("Emotiv Engine start up failed.");
}
break;
}
case 2: {
std::cout << "Target IP of EmoComposer? [127.0.0.1] ";
std::getline(std::cin, input, '\n');
if (input.empty()) {
input = std::string("127.0.0.1");
}
if (EE_EngineRemoteConnect(input.c_str(), 1726) != EDK_OK){
throw exception("Cannot connect to EmoComposer!");
}
break;
}
default:
throw exception("Invalid option...");
break;
}
```

Listing 5.1. connect to EmoEngine (Emotiv system, 2014).

The logger needs to initialize the connection with the Emotiv EmoEngine by calling EE_EngineConnect(), via EE_EngineRemoteConnect() together with the target IP address (127.0.0.1) and the fixed port 1726. The remote connection can be established by verifying the return value of the EE_EngineRemoteConnect() function.

```

EmoEngineEventHandle eEvent = EE_EmoEngineEventCreate();
EmoStateHandle eState = EE_EmoStateCreate();
unsigned int userID = 0;
while (...) {
int state = EE_EngineGetNextEvent(eEvent);
// New event needs to be handled
if (state == EDK_OK) {
EE_Event_t eventType = EE_EmoEngineEventGetType(eEvent);
EE_EmoEngineEventGetUserId(eEvent, &userID);
// Log the EmoState if it has been updated
if (eventType == EE_EmoStateUpdated) {
// New EmoState from user
EE_EmoEngineEventGetEmoState(eEvent, eState);
// Log the new EmoState
logEmoState(ofs, userID, eState, writeHeader);
writeHeader = false;
}}}
```

Listing 5.2. Buffer creation and management (Emotiv system, 2014).

A buffer is created to restore the events and states of the EmoEngine.

```

float secs = 1;
...
DataHandle hData = EE_DataCreate();
DataSetBufferSizeInSec(secs);
std::cout << "Buffer size in secs:" << secs << std::endl;
...==> Setting Cognitiv active actions for user 0...
```

Listing 5.3. Access to EEG data (Emotiv system, 2014).

A DataHandle is created by the access requirement. The DataHandle is used to provide access to the underlying data. The EE_DataCreate() function can be called on to initialize this DataHandle. During the measurement process, EmoEngine maintains a data buffer of sampled data, measured in seconds. The DataSetBufferSizeInSec() can be called to initialize the data buffer.

```
while (...)
state = EE_EngineGetNextEvent(eEvent);
if (state == EDK_OK) {
EE_Event_t eventType =
EE_EmoEngineEventGetType(eEvent);
EE_EmoEngineEventGetUserId(eEvent, &userID);
// Log the EmoState if it has been updated
if (eventType == EE_UserAdded) {
EE_DataAcquisitionEnable(userID,true);
readytocollect = true;
}
}
```

Listing 5.4. Start Acquiring Data (Emotiv system, 2014).

The system needs to register a valid user when the first connection to EmoEngine is made via EE_EngineConnect(). The EE_UserAdded event is the trigger for the registration. Once the user is registered, it is possible to enable data acquisition via a call to DataAcquisitionEnable. With this enabled, EmoEngine will start collecting EEG for the user, storing it in the internal EmoEngine sample buffer.

```
if (readytocollect)
...
DataUpdateHandle (0, hData);
unsigned int nSamplesTaken=0;
DataGetNumberOfSample(hData,&nSamplesTaken);
if (nSamplesTaken != 0)
...
double* data = new double[nSamplesTaken];
EE_DataGet(hData, targetChannelList[i], data,
```

```
nSamplesTaken);  
delete[] data;
```

Listing 5.5. Acquiring Data (Emotiv system, 2014).

To initiate retrieval of the latest EEG buffered data, a call is made to `DataUpdateHandle()`. When this function is processed, the `EmoEngine` will ready the latest buffered data for access via the `hData` handle. All data captured since the last call to `DataUpdateHandle` will be retrieved. Then a call is placed to `DataGetNumberOfSample()` to establish how much buffered data is currently available. The number of samples can be used to set up a buffer for retrieval into your application.

The EEG data received can be transferred into the `NeuCube` input buffer by calling the `EE_DataGet()` function. The available data channels are shown below:

```
ED_COUNTER,ED_AF3, ED_F7, ED_F3, ED_FC5, ED_T7,  
ED_P7, ED_O1, ED_O2, ED_P8, ED_T8, ED_FC6, ED_F4,  
ED_F8, ED_AF4, ED_GYROX, ED_GYROY, ED_TIMESTAMP,  
ED_FUNC_ID, ED_FUNC_VALUE, ED_MARKER, ED_SYNC_SIGNAL
```

Listing 5.6. Emotiv data channels (Emotiv system, 2014).

For example, to retrieve the first sample of data held in the sensor `FC5`, place a call to `EE_DataGet` as follows:

```
EE_DataGet(hData, ED_FC5, databuffer, 1).
```

5.2.2. NeuCube input module

To load EEG data collected by the Emotiv headset, the `NeuCube` system needs to set the input mapping first. Then calling the `EmoEngine` functions to load samples from the Emotiv headset.

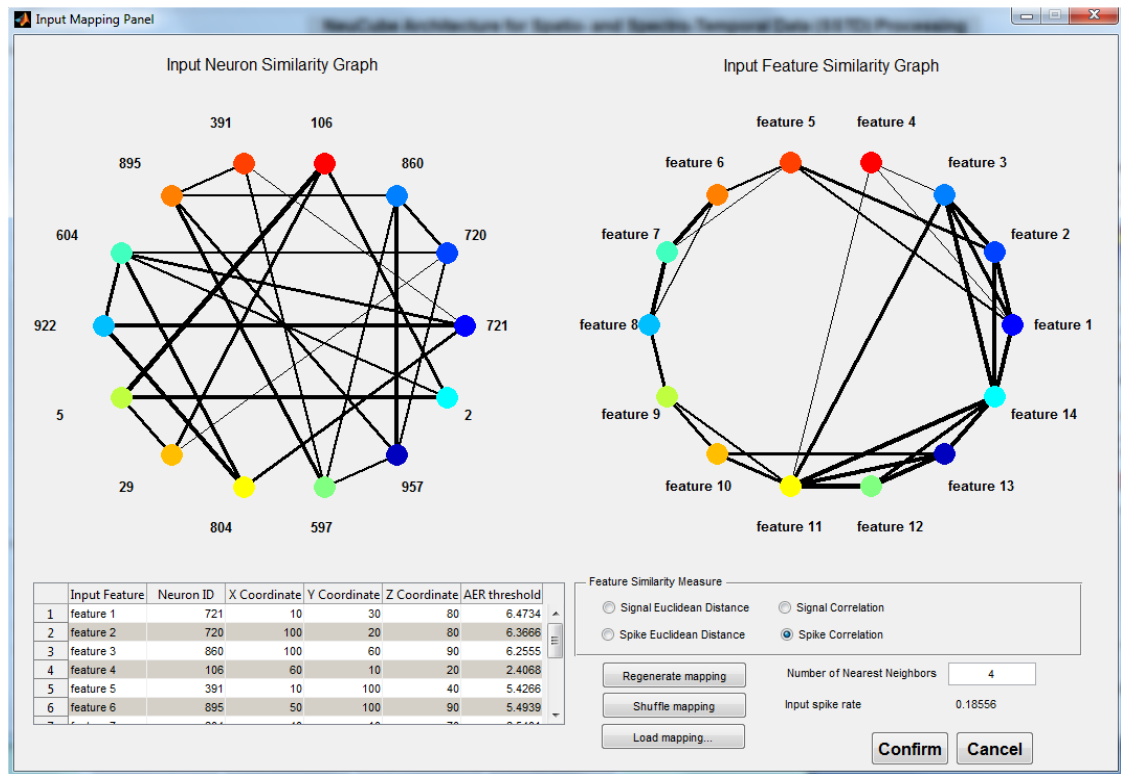


Figure 5.4. Setting input mapping manually in NeuCube.

As shown in Figure 5.4, the 14 features represented the 14 channels from the Emotiv headset. The space position of each sensor is set. Also, the connections of the sensors have been trained. As a trigger set, the system can record the EEG data for 128 samples per second by each sensor.

Chapter 6. Conclusion and Future Works

In this chapter, the whole study is summarized, and the main outcome is outlined. Furthermore, the limitations are stated and the potential future areas of research are indicated.

6.1. Conclusion and contribution

BCI is a very potential study field as many BCI applications have been developed in recent years. The EEG based BCI is suitable to implement on individual devices, is cheap, has a rapid response time, and is portable.

A literature review was given to introduce recent BCI developments. The EEG based BCI studies are further described as the advantages shown above.

An experiment which controls a robot by using the Emotiv system was illustrated. This prototype pointed out the principle of an EEG based BCI system working. In addition, the framework to implement EmoEngine on the NeuCube input module was accomplished.

The three research questions were answered. The BCI was introduced in detail in Chapter 2, and the experiment in Chapter 4 showed the principle of the BCI system working. The early stage of controlling a robot by using the NeuCube system was illustrated in Chapter 5. This outcome provided a solution that showed the NeuCube system acquiring EEG data in real-time through the Emotiv system is possible. These were the main outcomes of this thesis.

6.2. Limitations and Future Works

The third research question was about controlling the robot by using NeuCube. However, in this study, a framework was provided instead of an application. Furthermore, the Emotiv system was only equipped with 14 channels. In theory, the more sensors there are, the better classifying the result is. Another potential problem is that the processing speed of NeuCube is much slower than the sensors acquiring speed. This may lead to a problem of achieving the visualization of EEG data input in real-time.

In future studies, the first aim could be to accomplish the program, making the Emotiv system work effectively on the NeuCube system. In addition, a recognition module should be designed and coded for outputting tasks. With this recognition module, the system could classify EEG data in real-time, and output commands, making the NeuCube system as an EEG inputting system.

References

- Arjona, C., Pentácolo, J., Gareis, I., Atum, Y., Gentiletti, G., Acevedo, R., & Rufiner, L. (2011). Evaluation of LDA ensembles classifiers for brain computer interface. *Journal of Physics: Conference Series*, 332, 012025. doi:10.1088/1742-6596/332/1/012025
- Arvaneh, M., Guan, C., Ang, K. K., & Quek, C. (2011). Optimizing the channel selection and classification accuracy in EEG-based BCI. *IEEE Transactions on Bio-Medical Engineering*, 58(6), 1865.
- Bashashati, A., Fatourehchi, M., Ward, R.K., & Birch, G.E. (2007). A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals. *Journal of Neural Engineering*, 4 (2), 32. doi:10.1088/1741-2560/4/2/R03
- Bayliss, J. (2000). *Brain-computer interface and virtual reality*. Retrieved from <http://www.rochester.edu/pr/releases/cs/bayliss.html>
- Bohte, S. M. (2004). The evidence for neural information processing with precise spike-times: A survey. *Natural Computing*, 3(2), 195-206.
- Delbruck, T. (2007). *jaER open source project*. Retrieved from <http://jaer.wiki.sourceforge.net>.
- Doud, A. J., Lucas, J. P., Pisansky, M. T., & He, B. (2011). Continuous three-dimensional control of a virtual helicopter using a motor imagery based brain-computer interface. *PloS One*, 6(10), e26322. doi:10.1371/journal.pone.0026322
- Emotiv Systems (2013). *Emotiv EEG system electroencephalography*. Retrieved from <http://emotiv.com/>
- Fabiani, G. E., McFarland, D. J., Wolpaw, J. R., & Pfurtscheller, G. (2004). Conversion of EEG activity into cursor movement by a brain-computer interface (BCI). *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 12(3), 331-338.
- Fatourehchi, M., Bashashati, A., Ward, R. K., & Birch, G. E. (2007). EMG and EOG artifacts in brain computer interface systems: A survey. *Clinical Neurophysiology*, 118(3), 480-494. doi:10.1016/j.clinph.2006.10.019
- Felbecker, A. (2010). Four familial ALS pedigrees discordant for two SOD1 mutations: Are all SOD1 mutations pathogenic? *Journal of Neurology, Neurosurgery and Psychiatry*, 81(5), 572-589.
- Haihong, Z., Yang, C., Kai Keng, A., Cuntai, G., & Chuanchu, W. (2011). Optimum spatio-spectral filtering network for brain-computer interface. *Neural Networks*, 22(1), 52-63. doi: 10.1109/TNN.2010.2084099
- Hu, J., Hou, Z. G., Chen, Y. X., Kasabov, N., & Scott, N. (2014, August). EEG-based classification of upper-limb ADL using SNN for active robotic rehabilitation. In *Bio-medical Robotics and Biomechatronics (2014 5th IEEE RAS & EMBS International Conference on)* (pp. 409-414). IEEE.
- Huang, D., Qian, K., Fei, D., Jia, W., Chen, X., & Bai, O. (2012). Electroencephalography (EEG)-based brain-computer interface (BCI): A 2-D virtual wheelchair control

- based on event-related desynchronization/synchronization and state control. *IEEE Transactions on Neural Systems and Rehabilitation Engineering: A Publication of the IEEE Engineering in Medicine and Biology Society*, 20(3), 379.
- Jerbi, K., Vidal, J. R., Mattout, J., Maby, E., Lecaigard, F., Ossandon, T., Hamamé, C. M., Dalal, S. S., Bouet, R., Lachaux, J. P., Leahy, R. M., Baillet, S., Garnero, L., Del-puech, C., & Bertrand, O. (2011). Inferring hand movement kinematics from MEG, EEG and intracranial EEG: From brain-machine interfaces to motor rehabilitation. *Journal of IRBM*, 32(1). doi: 10.1016/j.irbm.2010.12.004
- Kasabov, N. K. (2014). NeuCube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data. *Neural Networks*, 52, 62-74.
- Khare, V., Santhosh, J., Anand, S., & Bhatia, M. (2010). Performance comparison of three artificial neural network methods for classification of electroencephalograph signals of five mental tasks. *Journal of Biomedical Science and Engineering*, 3(2), 200-205.
- Leeb, R., Friedman, D., Müller-Putz, G. R., Scherer, R., Slater, M., & Pfurtscheller, G. (2007). Self-paced (asynchronous) BCI control of a wheelchair in virtual environments: A case study with a tetraplegic. *Computational Intelligence and Neuroscience*, 2007, 79642-79650. doi:10.1155/2007/79642
- Levine, S. P., Huggins, J. E., BeMent, S. L., Kushwaha, R. K., Schuh, L. A., Rohde, M. M. & Smith, B. J. (2000). A direct brain interface based on event-related potentials. *IEEE Transactions on Rehabilitation Engineering*, 8(2), 180-185.
- Li, J., & Zhang, L. (2010). Regularized tensor discriminant analysis for single trial EEG classification in BCI. *Pattern Recognition Letters*, 31(7), 619-628. doi: 10.1016/j.patrec.2009.11.012
- Liang, W. (2009). *Integrated features, neighborhood, and model optimization for personalised modeling and knowledge discovery* (Unpublished master's thesis). Auckland University of Technology, New Zealand.
- Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F., & Arnaldi, B. (2007). A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of Neural Engineering*, 2(4), 1.
- McFarland, D. J., & Wolpaw, J. R. (2005). Sensorimotor rhythm-based brain-computer interface (BCI): Feature selection by regression improves performance. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13(3), 372-379. doi:10.1109/TNSRE.2005.848627
- Naumann, J. (2012). Search for Paradise: A Patient's Account of the Artificial Vision Experiment. United States of America: Xlibris Corporation.
- Pires, G., Castelo-Branco, M., & Nunes, U. (2008). Visual P300-based BCI to steer a wheelchair: A bayesian approach. *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 658-661. Vancouver, BC: IEEE.
- Rogers, P. (24/06/2011). Small gesture, major memory: WWII veteran, family will always remember encounter with schilling. *Chicago Tribune*, p. 4.

- Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., & Wolpaw, J. R. (2004). BCI2000: A general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Bio-medical Engineering*, 51(6), 1034-1043. doi: 10.1109/TBME.2004.827072
- Stephan, J. J. & Khudayer, S. (2010). Gesture recognition for human-computer interaction (HCI). *International Journal of Advancements in Computing Technology*, 2(4), 30-35. doi:10.4156/ijact.vol2.issue4.3
- Tan, D. & Nijholt, A. (2010). *Brain-computer interfaces and human-computer interaction*. In D. Tan & A. Nijholt (Eds.), *Brain-computer interfaces: Applying our minds to human-computer interaction* (pp. 3-19). London, England: Springer-Verlag.
- Vapnik, V. (1998). *Statistical learning theory*. New York, US: Wiley-Interscience.
- Vidal, J. J. (1977). Real-time detection of brain events in EEG. *Proceedings of the IEEE*, 65(5), 633-641. doi: 10.1109/PROC.1977.10542
- Yoon, J.W., Roberts, S.J., Dyson, M., & Gan, J.Q. (2011). Bayesian inference for an adaptive ordered probit model: An application to brain computer interfacing. *Neural Networks*, 24(7), 726-734. Retrieved from <http://www.sciencedirect.com/science/article/pii/S0893608011001031>

Appendices

A. WITH robot console pseudocode (robot.py) (Stefan Schliebs, 2012)

```
Imports

//Global constants

set SOCKET_HOST;

set SOCKET_PORT;

set WITH_CONTROL_COMMANDS{LIGHTS,FORWARD,BACK-
WARD,LEFT,RIGHT,STOP};

class RobotController

{      //Implements basic robot controls for the WITH robot provided by KIT via a
TCP/IP interface.

    def initialize();

    def connect(){check HOST&PORT if OK, connect to HOST; else return error;}

    def disconnect(){disconnect from HOST; else return error;}

    def STOP(){send stop command; else return error;}

    def FORWARD(){send move_forward command; else return error;}

    def BACKWARD(){send move_backward command; else return error;}

    def LEFT(){send turn_left command; else return error;}

    def RIGHT(){send turn_right command; else return error;}

    def LIGHT_ON(){turn on light on robot; else return error;}

    def LIGHT_OFF(){turn off light on robot; else return error;}

}
```

B. robot console HUD pseudocodes (Key_events.py) (Stefan Schliebs, 2012)

```
Imports

class Application()

{ //Application window capturing keyboard events. If the arrow keys are pressed, move-
ment commands are sent to the WITH robots.

    def app_initialize(){
```

```

        build application window;
        create buttons on this window;
        define arrow key_mappings with buttons;}
def create_widgets(){
    setup buttons on the application window;}
def keys(){
    capture key events sent to the application window;
    set key_events{STOP,FORWARD,BACK-
WARD,LEFT,RIGHT,LIGHT_ON,LIGHT_OFF}, else return "Error, Key pressed is
not set."
}
def button_check(){ if button pressed, set button_background "green";}

create controller instance;
create Tkinter object;
loop Application(robot);
disconnect(robot);
}

```

C. EEG Logger pseudocodes (EEG_Logger.cpp) (Emotiv system, 2014, Edited by Jianfei Wang)

Imports

```

class EEGLogger(){
//reading EEG data from Emotiv device, and check flag events, then send key press
command to the key_event application.

    set Emotiv headset chan-
nels{ AF3,F7,F3,FC5,T7,P7,O1,O2,P8,T8,FC6,F4,F8,AF4,GYROX,GY-
ROY,TIMESTAMP,FUNC_ID,FUNC_VALUE,MARKER,SYNC_SIGNAL.}

    def logger_initialize(){
        check device connection, return CONNECTED; else return ERROR;
        read Channels;
        output buff = Channels;}

    def Events_trigger(){

```

```
        load events_dataset;}  
def controller(){  
    classify buff;  
    if buff == events_dataset, return key_event(events_dataset);}  
loop controller(Application);  
EmoEngineDisconnect();  
return 0;  
}
```