

**ADVANCED COMPUTATIONAL
INTELLIGENCE STRATEGIES FOR
MENTAL TASK CLASSIFICATION
USING ELECTROENCEPHALOGRAPHY
SIGNALS**

By

Rifai Chai

Submitted to Faculty of Engineering and Information Technology
in fulfilment of the requirement for the degree of
Doctor of Philosophy
at the University of Technology, Sydney



Sydney, July 2014

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any assistance that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

Rifai Chai

Date:

Acknowledgement

Foremost, I would like to express my sincere gratitude to my Principal Supervisor, Professor Hung Tan Nguyen, for providing an immeasurable amount of unwavering support, constant guidance, valuable time and inspiring discussions throughout my PhD candidature in the Faculty of Engineering and Information Technology, University of Technology, Sydney.

I would also like to thank my Co-Supervisors, Dr. Greg Hunter and Dr. Steve Ling for providing ongoing technical support, advice and review.

My gratitude is also to Dr. Yvonne Tran for providing scientific advice on the research's experiment and helping to find participants especially patients with tetraplegia.

I would like to extend my appreciation to my colleagues at Centre for Health Technologies (CHT:UTS) and to many technical and administrative staffs in the Faculty of Engineering and Information Technology for their useful help, advice and encouragement during this research.

I am very grateful to my family in Indonesia and Australia for their strong support and encouragement.

Contents

List of Figures	viii
List of Tables	xii
Abbreviations	xiv
Abstract	xix
Chapter 1 Introduction	1
1.1 Problem Statement	1
1.2 Thesis Objective	6
1.3 Thesis Contribution	7
1.4 Thesis Outline	9
1.5 Thesis Publications	11
Chapter 2 Literature Review	13
2.1 Introduction: Brain Neurophysiology	13
2.2 Brain-Computer Interface (BCI) Components	18
2.3 State of Art BCI Technology	20
2.3.1 Invasive BCI	21
2.3.1.1 Intra-cortical Recording (Microelectrodes)	21
2.3.1.2 Electrocorticography (ECoG)	22
2.3.2 Non-invasive BCI	23
2.3.2.1 Magneto-encephalography (MEG)	23
2.3.2.2 Functional Near Infrared Spectroscopy (fNIRS)	24
2.3.2.3 Functional Magnetic Response Imaging (fMRI)	25
2.3.2.4 Positron Emission Tomography (PET)	26
2.3.2.5 Single Photon Emission Computed Tomography (SPECT)	27
2.3.2.6 Electroencephalography (EEG)	28
2.4 EEG-based BCI	29

2.4.1	EEG Rhythm and Measurement	29
2.4.2	Existing EEG-based BCIs	31
2.4.2.1	P300	32
2.4.2.2	Steady State Visual Evoked Potential (SSVEP)	33
2.4.2.3	Slow Cortical Potential (SCP)	35
2.4.2.4	Sensorimotor Rhythms (SMR)/ Motor Imagery Tasks.....	37
2.4.2.5	Mental Tasks (Non-motor Imagery Mental Tasks)	39
2.5	Review on Computational Intelligence for EEG-based BCI.....	41
2.5.1	Features Extraction	42
2.5.1.1	Time (Temporal) Methods	43
2.5.1.2	Frequency (Spectral) Methods	44
2.5.1.3	Time-Frequency Methods	45
2.5.2	Feature Translation or Classification.....	46
2.5.2.1	Linear Classification Methods	47
2.5.2.2	Non-linear Classification Methods	48
2.6	Discussion	49
2.7	Conclusion.....	55
Chapter 3	Wireless Embedded EEG System for Mental Task Classification.....	57
3.1	Introduction	57
3.2	Components of Microcontroller-based BCI for Mental Task Classifications....	63
3.2.1	Wireless EEG and Wireless Receiver Development.....	64
3.2.1.1	Wireless EEG	64
3.2.1.2	Wireless Receiver	86
3.2.2	Software on PC for Wireless EEG Recording.....	87
3.2.3	Computational Intelligence.....	90
3.2.3.1	Signal Pre-processing.....	90
3.2.3.2	Feature Extraction Algorithm based on Fast Fourier Transform (FFT)	91
3.2.3.3	Classification Algorithm based on Artificial Neural Network (ANN)	95
3.2.3.3.1	Properties of ANN	96
3.2.3.3.2	Multilayer of ANN.....	97
3.2.3.3.3	Learning of Multilayer ANN	98

3.2.3.3.4	The Back-propagation Algorithm of ANN	100
3.2.3.3.5	The Levenberg-Marquardt Algorithm of ANN	102
3.2.3.3.6	Early Stopping for Generalization Performance	105
3.2.4	Main Embedded System Controller	105
3.3	Performance Measurement	111
3.3.1	Classification Accuracy	111
3.3.2	Bit Rate and Information Transfer Rate (ITR)	112
3.4	Data Collection and Experiment	112
3.5	Experimental Results and Discussions	114
3.5.1	Measurement for the Specifications of Wireless EEG	114
3.5.2	Testing Wireless EEG	119
3.5.3	Complete Hardware of Microcontroller-based BCI system for Mental Task Classifications	123
3.5.4	Results of ANN Classifications	123
3.5.5	Results of Real-time Online Classifications	128
3.6	Conclusion	133
Chapter 4 Optimized BCI using Hilbert-Huang Transform Feature Extractor and Genetic Algorithm-Artificial Neural Network Classifier		134
4.1	Introduction	134
4.2	Data Collection	137
4.3	Signal Pre-processing	139
4.4	Feature Extraction Algorithms	141
4.4.1	Fast Fourier Transform (FFT)	141
4.4.2	Hilbert-Huang Transform (HHT)	142
4.5	Classification Algorithms using the Genetic Algorithm Optimization of Artificial Neural Network (GA-ANN)	151
4.6	Results and Discussion	162
4.6.1	Result Classification with FFT (Feature Extractor) and GA-ANN (Classifier)	162
4.6.2	Testing of the HHT Algorithm for Feature Extractor	167

4.6.3	Result Classification with HHT (Feature Extractor) and GA-ANN (Classifier).....	171
4.7	Conclusion.....	175
Chapter 5 Advanced BCI System using Fuzzy Particle Swarm Optimization of Artificial Neural Network		
	176	176
5.1	Introduction	176
5.2	Data Collection.....	180
5.3	Signal Pre-Processing and Feature Extraction	183
5.3.1	Signal Pre-processing	183
5.3.2	Feature Extractor using Hilbert-Huang Transform (HHT).....	184
5.3.3	Other Feature Extractors for Comparison Purpose.....	186
5.3.3.1	Fast Fourier Transform (FFT).....	186
5.3.3.2	Wavelet	187
5.4	Classification Algorithms.....	189
5.4.1	Fuzzy Particle Swarm Optimization with Cross-Mutated Operation of Artificial Neural Network (FPSOCM-ANN).....	189
5.4.2	Other Classifiers for Comparison Purposes.....	197
5.4.2.1	Linear Perceptron.....	197
5.4.2.2	Linear Discriminant Analysis (LDA)	198
5.4.2.3	Support Vector Machine (SVM).....	200
5.4.2.4	Genetic Algorithm of Artificial Neural Network (GA-ANN)	201
5.5	Results	202
5.5.1	Testing of the HHT as the Features Extraction	202
5.5.2	Classification using FPSOCM-ANN and Comparison with GA-ANN at Different Time-Windows.....	203
5.5.3	Further Comparison Classifiers and Features Extractors	208
5.6	Discussion	212
5.6.1	Comparison between the Globally Trained Classifier and the Subject's Specifically Trained Classifier.....	212
5.6.2	Computational time of Classifiers	213
5.6.3	Two Channels Classification for Practical Application	215

5.7	Conclusion.....	218
Chapter 6	Conclusion and Future Work.....	220
6.1	Conclusion.....	220
6.2	Future Work	227
Appendix A	Research Ethics Clearance	229
A.1	Ethics Clearance Letter	230
A.2	Ethics Amendment Clearance Letter.....	231
A.3	Information Sheet.....	232
A.4	Consent Form	233
Appendix B	Firmware and Software Code	234
B.1	Firmware of Wireless EEG	234
B.2	Firmware of Wireless Receiver.....	248
B.3	Firmware of Embedded System	250
B.4	Software of ANN	271
B.5	Software of GA-ANN	278
B.6	Software of FPSOCM-ANN	286
Bibliography	292

List of Figures

Figure 1.1: Overall BCI works in this thesis.....	10
Figure 2.1: The structure of neuron	14
Figure 2.2: Different areas of the cerebral cortex	17
Figure 2.3: Block diagram components of BCI	18
Figure 2.4: BCI classifications from recording methods	20
Figure 2.5: The international ten-twenty (10-20) system for electrode placement.....	29
Figure 2.6: Existing BCI-EEG methods	31
Figure 2.7: Existing BCI-EEG computational intelligence.....	41
Figure 2.8: BCI technologies electrophysiological view	51
Figure 2.9: Resolutions of BCI technologies	51
Figure 3.1: Example BCI-based products on the current market.....	59
Figure 3.2: Instrumentation of BCI for wheelchair control	61
Figure 3.3: Block diagram of microcontroller-based BCI for mental task classifications	63
Figure 3.4: The shape of ECG and EEG signal of measurements	66
Figure 3.5: General block diagram of EEG measurement	66
Figure 3.6: Human body and electromagnetic interference	67
Figure 3.7: Op-Amp in inverting voltage amplifier	69
Figure 3.8: Op-Amp in non-inverting voltage amplifier.....	71
Figure 3.9: Op- Op-Amp in a unity-gain buffer.....	71
Figure 3.10: Differential amplifier analogy with differential and common-mode voltage applied to the input.....	73
Figure 3.11: Differential amplifiers with an op-amp	73
Figure 3.12: AC coupled In-Amp	74
Figure 3.13: Prototype wireless EEG V1	75
Figure 3.14: Schematic of wireless EEG V1	76
Figure 3.15: DC coupled In-Amp	77

Figure 3.16: Virtual ground for single supply amplifier	78
Figure 3.17: Prototype wireless EEG V2.....	79
Figure 3.18: Schematic wirelesses EEG V2	80
Figure 3.19: Flowchart of firmware wireless EEG	82
Figure 3.20: Internal functions/routines of firmware for Wireless EEG	83
Figure 3.21: Interrupt timer at 256 Hz (~ 4 ms).....	83
Figure 3.22: ADC conversion measurements on the wireless EEG	83
Figure 3.23: Wireless receiver	86
Figure 3.24: Flowchart of firmware wireless receiver	87
Figure 3.25: Design configurations on the recording software.....	88
Figure 3.26: Software on PC for wireless EEG recording.....	89
Figure 3.27: Eight point decimation in time of Radix-2 FFT algorithm.....	94
Figure 3.28: Two-point FFT butterfly illustration	94
Figure 3.29: Neuron of ANN	98
Figure 3.30: The three layers ANN.....	98
Figure 3.31: Main embedded system controller of the BCI.....	106
Figure 3.32: Block diagram firmware of main embedded system.....	107
Figure 3.33: Real-time data receiving at 256 Hz	107
Figure 3.34: Flowchart FFT Radix-2 of 256 point in assembly routine	108
Figure 3.35: Real-time data receiving at 256 Hz	109
Figure 3.36: Wireless EEG with two channels electrodes placement.....	113
Figure 3.37: Wireless EEG gain testing with simulator.....	115
Figure 3.38: CMRR measurement of Wireless EEG amplifier.....	115
Figure 3.39: Testing Wireless EEG with its Software on PC connected to EEG simulator	120
Figure 3.40: Testing Procomp2 (Thought Technology) with its software on PC connected to EEG simulator	121
Figure 3.41: Electrode placements for ECG measurement.....	121
Figure 3.42: Wireless EEG V2 testing for ECG measurement on body.....	122
Figure 3.43: Compumedics-Siesta system with ECG measurement on body.....	122
Figure 3.44: Complete system of microcontroller	123

Figure 3.45: Display of the recording software on PC with the wireless EEG V2 for eyes closed and open.....	124
Figure 3.46: Wireless EEG V2 for alpha wave detection during eyes closed action..	125
Figure 3.47: Training and validation error cycle of three mental task classifications (math, letter and count)	127
Figure 3.48: Block diagram real-time online microcontroller-based BCI.....	129
Figure 3.49: Real-time online operation of microcontroller-based BCI for eyes open classification	130
Figure 3.50: Real-time online operation of microcontroller-based BCI for eyes closed classification	130
Figure 3.51: Real-time online operation of microcontroller-based BCI for mental task detection.....	131
Figure 3.52: Real-time online BCI classifications; map into wheelchair commands and display to the LCD	131
Figure 4.1: Components of BCI using mental tasks	135
Figure 4.2: EEG system set-ups for data collection.....	138
Figure 4.3: Data segmentation	140
Figure 4.4: DSP filters on the data.....	140
Figure 4.5: EMD process with random test signal.....	144
Figure 4.6: Flowchart of EMD.....	147
Figure 4.7: Example of the 3D plot of the HHT spectrum.....	150
Figure 4.8: Architecture of ANN	152
Figure 4.9: Procedure of GA.....	153
Figure 4.10: Plot of accuracy of five subjects with FFT (feature extractor) and GA-ANN (classifier).....	163
Figure 4.11: Plot fitness value of five subjects with FFT (feature extractor) and GA-ANN (classifier).....	166
Figure 4.12: Number of hidden neuron vs. accuracy of five subjects with FFT (feature extractor) and GA-ANN (classifier)	167
Figure 4.13: EMD process for test signal $x(t)$	168
Figure 4.14: HHT process (HT spectrum) for test signal $x(t)$	169

Figure 4.15: EMD process for eyes closed signal at 1s time-window	170
Figure 4.16: HHT process (HT spectrum) of eyes closed signal at a 1s time-window	170
Figure 5.1: Components of mental tasks-based BCI for wheelchair commands	178
Figure 5.2: Experiment setup of EEG-BCI with patients with able-bodied subjects and patients with tetraplegia	181
Figure 5.3: Impedance checking using the recording software.....	182
Figure 5.4: Pictures from the video guidance for the participants	183
Figure 5.5: Wavelet decomposition of EEG signal with sampling frequency at 256Hz	188
Figure 5.6: The ANN structure for mental task-based BCI	190
Figure 5.7: The FPSOCM procedure	191
Figure 5.8: The Fuzzy rules for inertia weight, $\tilde{\omega}(t)$	193
Figure 5.9: The Fuzzy rules for control parameter, $\beta(t)$	196
Figure 5.10: HHT spectrum for eyes closed 1s time-window	202
Figure 5.11: Accuracy of eyes closed-open action at 1s time-window.....	204
Figure 5.12: Plot of overall accuracy FPSOCM-ANN vs. GA-ANN	207
Figure 5.13: Plot of accuracy for able bodied subjects vs. patients with tetraplegia, time-windows at 1-10s; and classification based on FPSOCM-ANN.	207
Figure 5.14: Plot of overall comparison between different classifiers and features extractors at 7s time-window of data.	211
Figure 5.15: Comparison accuracy between globally trained classifier and subject's specifically trained classifier.....	213

List of Tables

Table 2.1: Overview BCI technologies	54
Table 2.2: Comparison BCI-EEG methods.....	55
Table 3.1: Electrical potential and frequency ranges of various Bio-potential signals.	65
Table 3.2: Protocol packet of the wireless EEG.....	85
Table 3.3: EEG rhythms selection each channel.....	94
Table 3.4: Sigmoid function of Taylor series approximation	111
Table 3.5: Comparison sigmoid function and approximation.....	111
Table 3.6: Wireless EEG V1 gain testing	115
Table 3.7: CMRR measurement for wireless EEG V1	116
Table 3.8: CMRR measurement without DC filter for wireless EEG V1	116
Table 3.9: Wireless EEG V2 gain testing	117
Table 3.10: CMRR measurement for wireless EEG V2	118
Table 3.11: Measurement of wireless EEG V2 specifications.....	118
Table 3.12: Result of classification eyes closed-open action.....	126
Table 3.13: Result of mental tasks classifications.....	126
Table 3.14: Result of real-time online testing	132
Table 4.1: Parameters GA and FPSOCM for ANN training.....	162
Table 4.2: Accuracy of with FFT (feature extractor) and GA-ANN (classifier)	164
Table 4.3: Chosen mental task of each subject with FFT (feature extractor) and GA-ANN (classifier).....	165
Table 4.4: Classification accuracy of chosen mental task of 2 channels EEG combinations and comparison feature extractors (FFT and HHT) with GA-ANN as classifier	173
Table 4.5: Comparisons between classifiers ANN and GA-ANN; between feature extractor FFT and HHT	174
Table 5.1: Details of participants: able-bodied subjects and patients with tetraplegia	181
Table 5.2: EEG Frequencies Corresponding to Wavelet Decomposition	189

Table 5.3: Parameters GA and FPSOCM for ANN training.....	203
Table 5.4: Comparison accuracy between GA-ANN and FPSOCM-ANN of 6 channels for 3 mental task classifications in different time-windows with 5 able-bodied subjects and 5 patients with tetraplegia	206
Table 5.5: Comparison between different classifiers and feature extractors using 7s time-window of data	210
Table 5.6: Statistical significant using overall means classification accuracies at 7s of time-window	211
Table 5.7: Comparison of training times and classification times	214
Table 5.8: Accuracies of two channels combinations of 3 mental tasks classification in 7s time-window with 5 able-bodied subjects (S1-S5) and 5 patients with tetraplegia (T1-T5) using FPSOCM-ANN	216
Table 5.9: Accuracies and correctly classified rates of each mental task in 7s time-window with 5 able-bodied subjects (S1-S5) and 5 patients with tetraplegia (T1-T5) using FPSOCM-ANN for 6 channels (C3-C4-P3-P4-O1-O2) and the best 2 channels (O1-O4) EEG	217

Abbreviations

ABS	: Australian Bureau of Statistic
AC	: Alternating current
ADC	: Analog-to-digital converter
ALN	: Adaptive logic network
ALS	: Amyotrophic lateral sclerosis
ANN	: Artificial neural network
AR	: Autoregressive
ASSEP	: Auditory steady state evoked potential
AT	: Assistive technology
BCGA	: Binary code genetic algorithm
BCI	: Brain-computer interface
BLRNN	: Bayesian logistic regression neural network
BLX- α	: Blend- α crossover
BMI	: Brain-machine interface
BOLD	: Blood-oxygen-level-dependent
BSVM	: Biased support vector machine
CM	: Cross-mutated
CMRR	: Common mode rejection
CNS	: Central nervous system
CNV	: Contingent negative variation

CRC	. Cyclic redundancy check
DBI	. Direct brain interface
DC	. Direct current
deoxy-Hb	. Deoxy-haemoglobin
DFT	. Discrete Fourier transform
DNA	. Deoxyribonucleic acid
DSP	. Digital signal processing
DWT	. Discrete wavelet transform
EA	. Evolutionary Algorithm
ECG	. Electrocardiography
ECoG	. Electrocorticography
EEG	. Electroencephalography
EIX	. Extended intermediate crossover
EEMD	. Ensemble empirical mode decomposition
EMD	. Empirical mode decomposition
EMG	. Electromyography
EOG	. Electrooculography
ERD	. Event-related desynchronization
ERD/ERS	. Event-related desynchronization/synchronization
ERP	. Event-related potential
ERS	. Event-related synchronization
FES	Functional electrical stimulation
FFT	. Fast Fourier transform
FIFO	. First-in-first-out
FIS	. Fuzzy inference system

fMRI	. Functional magnetic resonance imaging
fNIRS	. Functional near-infrared spectroscopy
FPSOCM	. Fuzzy particle swarm optimization with the cross-mutated operation
FPSOCM-ANN	. Fuzzy particle swarm optimization using cross-mutated operation of . artificial neural network
GA	. Genetic algorithm
GA-ANN	. Genetic algorithm optimization of artificial neural network
GD	. Gradient-descent
HHT-EMD	. Hilbert-Huang transform using empirical mode decomposition
HHT-EEMD	. Hilbert-Huang transform using ensemble empirical mode . decomposition
HHT	. Hilbert-Huang transform
HT	. Hilbert transform
In-Amp	. Instrumentation amplifier
IC	. Integrated circuit
iEEG	. Intracranial electroencephalography
IMF	. Intrinsic mode function
IRQ	. Interrupt request
ISM	. Industrial, scientific and medical
ITR	. Information transfer rate
LED	. Light-emitting diode
LDA	. Linear discriminant analysis
LFP	. Local field potentials
LVQ	. Learning vector quantization
MAC	. Multiplier and accumulator

MEG	. Magneto-encephalography
MLP	. Multi-layer perceptron
MRI	. Magnetic resonance imaging
MSE	. Mean square error
NADA	. Noise-assisted data analysis
NFB	. Neurofeedback
OAA	. One-against-all
OAo	. One-against-one
oxy-Hb	. Oxy-haemoglobin
PCB	. Printed circuit board
PC	. Personal computer
PET	. Positron emission tomography
PSD	. Power spectral density
PSO	. Particle swarm optimization
RBF	. Radial basis function
RCGA	. Real-coded genetic algorithm
RF	. Radio frequency
RFI	. Radio frequency interference
RMP	. Resting membrane potential
RTC	. Real time clock
RTOS	. Real-time operating system
SCP	. Slow cortical potential
SCI	. Spinal cord injury
SDAC	. Survey of disability, ageing and carers
SNR	. Signal-to-noise ratio

SMR	: Sensorimotor rhythm
SPECT	: Single photon emission computed tomography
SPI	: Serial peripheral interface
SQUID	: Superconducting quantum interference device
SSVEP	: Steady state visual evoked potential
SVM	: Support vector machine
TTD	: Thought translation device
UART	: Universal asynchronous receiver-transmitter
UIEA	: Utah intracranial electrode array
UNDX	: Unimodal normal distribution crossover
VEP	: Visual evoked potential
WT	: Wavelet transform

Abstract

Brain-computer interface (BCI) has been known as a cutting-edge technology in the current research. It is able to measure the brain activity directly instead of using the natural peripheral nerves and muscles and translates the user's intent brain activity into useful control signals. There is still a need for a technology for severely disabled individuals who suffer from locked-in syndromes, such as amyotrophic lateral sclerosis (ALS), cervical spinal cord injury (SCI) or tetraplegia and brain stem stroke. A brain-computer interface (BCI) could be used here as an alternative solution for control and communication. The main aim of this research is to develop a BCI system to assist mobility as hand-free technology for people with severe disability, with improved accuracy, which provides effective classification accuracy for wheelchair control.

Electroencephalography (EEG) is the chosen BCI technology because it is non-invasive, portable and inexpensive. Currently, BCI using EEG can be divided into two strategies; selective attention and spontaneous mental signal. For the selective attention strategy, BCI relies on external stimuli which might be uncomfortable for severely disabled individuals who need to focus on external stimuli and the environment simultaneously. This is not the case for BCIs which rely on spontaneous mental signals initiated by the users themselves. BCI that uses sensorimotor rhythm (SMR) is one of the examples of the spontaneous mental strategy. There have been many reports in research using SMR-based BCI; however, there are still some people who are unable to use this. As a result, in this thesis, mental task-based EEG is used as an alternative.

This thesis presents the embedded EEG system for mental task classification. A prototype wireless embedded EEG system for mental task BCI classification is developed. The prototype includes a wireless EEG as head gear and an embedded system with a wireless receiver. The developed wireless EEG provides a good common mode rejection ratio (CMRR) performance and a compact size with a low current consumption coin cell battery for power. Mental tasks data are collected using the prototype system from six healthy participants which include arithmetic, figure rotation,

letter composing and counting task with additional eyes closed task. The developed prototype BCI system is able to detect the dominant alpha wave between 8-13Hz during eyes closed. Using the FFT as the features extractor and artificial neural network (ANN) as the classifier, the developed prototype EEG system provides high accuracy for the eyes closed and eyes open tasks. The classification of the three mental task combinations achieve an overall accuracy of around 70%.

Also, an optimized BCI system for mental task classification using the Hilbert-Huang transform (HHT) feature extractor and the genetic algorithm optimization of the artificial neural network (GA-ANN) classifier is presented. Non motor imagery mental tasks are employed, including: arithmetic, letter composing, Rubik's cube rolling, visual counting, ringtone, spatial navigation and eyes closed task. When more mental tasks are used, users are able to choose the most effective of tasks suitable for their circumstance. The result of classification for the three user chosen mental tasks achieves accuracy between 76% and 85% using eight EEG channels with GA-ANN (classifier) and FFT (feature extractor). In a two EEG channels classification using FFT as the features extractor, the accuracy is reduced between 65% and 79%. However, the HHT features extractor provides improved accuracy between 70% and 84%.

Further, an advanced BCI system using the ANN with fuzzy particle swarm optimization using cross-mutated operation (FPSOCM-ANN) for mental task classification is presented. This experiment involves five able-bodied subjects and also five patients with tetraplegia as the target group of the BCI system. The three relevant mental tasks used for the BCI concentrates on mental letter composing, mental arithmetic and mental Rubik's cube rolling forward. Although the patients group has lower classification accuracy, this is improved by increasing the time-window of data with the best at 7s. The results classification for 7s time-window show the best classifier is using the FPSOCM-ANN (84.4% using FPSOCM-ANN, 77.4% using GA-ANN, 77.0% using SVM, 72.1% using LDA, and 71.0% using linear perceptron). For practical use of a BCI, the two channels EEG is also presented using this advanced BCI classification method (FPSOCM-ANN). For overall, O1 and C4 are the best two channels at 80.5% of accuracy, followed by the second best at P3 and O2 at 76.4% of accuracy, and the third best at C3 and O2 channels at 75.4% of accuracy.

Chapter 1

Introduction

1.1 Problem Statement

As reported by the Australian Bureau of Statistic (ABS), for Survey of Disability, Ageing and Carers (SDAC), in 2009, 18.5% of the population had a disability. An additional 21% of the population had a long term health condition that did not restrict their daily activities. The remaining 60% of the population had neither a disability nor a long term health condition. Of those with a reported disability, around 87% had a specific limitation and restriction in one or more core activities, including mobility, communication, schooling and employment. The rate of disability increased with age. For example, disability affected 3.4% of people aged four years and under, 40% of people aged between 65 and 69; and 88% of those aged 90 years and over (Australian Bureau of Statistics 2009). In the case of spinal cord injury (SCI), the incidence in

industrial countries is around 40 new cases every year per million population with an increasing percentage of non-traumatic origin (van den Berg et al. 2010).

With increasing need for mobility by individuals with disability, the powered wheelchair is one example of an assistive device. Some alternative hands-free technologies can be utilized to replace the joystick control, including sip-and-puff, chin controller (Guo et al. 2002), voice recognition (Peixoto, Nik & Charkhkar 2013; Simpson & Levine 2002), tongue controller (Huo & Ghovanloo 2010) and muscle-based system (Zhang et al. 2012a). Another technology that has been the focus at the Centre for Health Technologies, UTS, is the real-time telemetric head movement, which uses an accelerometer sensor (Joseph & Nguyen 1998; Nguyen, King & Knight 2004; Taylor & Nguyen 2003). All of these technologies have their own benefits and drawbacks. In practical situations, the user may feel uncomfortable with the operation of a sip-and-puff or a chin or tongue controller. Noisy environments can be problematic for voice recognition systems. Muscle and head movement technology are targeted for disabled individuals who are still able to provide the relevant head and body movement for real-time control.

Furthermore, there is still a need for other technologies for individuals with a severe disability or who suffer from locked-in syndromes such as amyotrophic lateral sclerosis (ALS), cervical spinal cord injury (SCI), brain stem stroke and other diseases. ALS, sometimes called Lou Gehrig's disease, is a progressive neuronal degeneration. An ALS patient has global brain atrophy with regional grey matter density being highest in the right hemispheric of primary motor-cortex and left hemispheric medial frontal gyrus. As a result, the patient progressively becomes severely physically impaired. In the later stage, it could affect speech, swallowing and even breathing. The spinal cord is the thick cord of nerves connecting the brain to the peripheral nervous system. Cervical (neck) SCI refers to tetraplegia or quadriplegia, which is the injury of the spinal cord between the level of the first cervical vertebrae (C1) and the seventh vertebrae (C7). A patient with tetraplegia injury C1-C4 has limited head and neck movement, depending on muscle strength, and complete paralysis of arm, body and legs. A patient with

tetraplegia at level C5 and above has started gaining limited arm movement but still has complete paralysis of body and legs. A stroke of the brain stem could also cause locked-in syndrome with total immobility except for eye movement and blinking.

These locked-in syndrome patients are unable to move while the brain is still properly functioning, trapping them inside a paralysed body. A brain-computer interface (BCI) could be used here as an alternative solution for control and communication. In the case of mobility, BCI could be used for individuals with disabilities by converting brain activities into commands to control a powered wheelchair. As a result, BCI is used in order to improve the quality of life for people with severe disabilities (Kubler et al. 2006; Wolpaw & Wolpaw 2012).

In terms of definition, brain-computer interface (BCI) is a system that measures the intention of the user's brain activity and bypasses the normal neuromuscular pathway. Thus, BCI offers an option as the natural way for communication and control. BCI is able to measure the brain activity directly instead of using the natural peripheral nerves and muscles and translates the user's intent brain activity into useful control signals for BCI applications (He et al. 2013; Wolpaw et al. 2002). Some other BCI terminologies have been used so far in the literature such as direct brain interface (DBI) (Graumann et al. 2004; Levine et al. 1999), brain-machine interface (BMI) (Donoghue 2002; Serruya et al. 2002), and thought controller/classifier/recognition (Chai et al. 2012d; Craig & Nguyen 2007; Craig, Nguyen & Burchey 2006; Pfurtscheller et al. 2003b).

BCI research is one of the cutting-edge technologies in the current research and an interdisciplinary area of research involving engineering, neuroscience, psychology, computer science, clinical rehabilitation and others (Vaughan et al. 2003). Engineering and computer science fields can contribute to the BCI such as developing suitable instrumentation with improved signal processing, and machine learning algorithms for practical BCI operation. In the Neurophysiology field, BCI can be used to understand specific brain functionality. Psychologists are able to apply BCI for neurofeedback (NFB) of their patients in order to understand human behaviour.

The basic components of BCI consist of several elements, including signal acquisition, signal pre-processing, features extraction and a classification or translation algorithm (He et al. 2013; Mason & Birch 2003; Wolpaw et al. 2002). The processes start by measuring the intent of the user's brain signal with the instrumentation to provide data acquisition. The output of the data acquisition process is in a digital form, which is further enhanced by the process of signal pre-processing to filter the noise. This is followed by the features extraction, which transforms the signals into useful features that correspond to the mental strategy of the BCI. The features will be fed into the classification or translation method which will give the output of commands or logical control signals to operate application devices that replace, restore, enhance, and supplement the natural way of central nervous system functions.

A BCI system that uses invasive methods, such as electrocorticography (ECoG) and intra-cortical (microelectrodes) recording, provides a better signal resolution, frequency range, and better quality of signal. However, the disadvantages include risk of infection, post-operative scarring and other unknown long term effects. On the other hand, non-invasive BCI technologies, including functional magnetic resonance imaging (fMRI), Magneto-encephalography (MEG) and Positron Emission Tomography (PET), are not mobile and are expensive. BCI using electroencephalography (EEG) and functional near infrared spectroscopy (fNIRS) can be more portable and inexpensive, but fNIRS provides lower throughput and poor temporal resolution. This research will use EEG as a preferable tool in BCI at the current state. A drawback of EEG is a higher sensitivity to noise, including ocular-muscular artefacts and other noises generated in the system. A novel signal processing algorithm will be needed to address the noise issue (Brunner et al. 2011; Hochberg & Donoghue 2006; Kubler & Muller 2007; Wolpaw et al. 2006).

So far, there have been no products released on the market that use BCI particularly in application on wheelchair controls for mobility of individuals with disabilities. Most of the thought controller products on the market are targeted for gaming applications. The operation of the system is simply based on mental concentration or relaxation with the electrodes placed on the scalp (Emotiv ; Neurosky ; OCZ Technology).

Regarding current BCI-EEG research, the popular methods focus on selective attention and spontaneous mental signal. P300 and steady state visual evoked potential (SSVEP) are examples of the selective attention method in which the user needs to pay attention to external stimuli whilst controlling the wheelchair. A commercial P300-based BCI as a spelling system is also available currently on the market (g.tech). Concentrating on the control of the wheelchair and the stimuli at the same time may prove difficult. However, this is not the case for a BCI system based on spontaneous mental signals given by the user without any external cues. A BCI-based motor imagery task is an example of the spontaneous mental signal method, which focuses on the motor imagery area by imagining hand, feet and tongue movement. Furthermore, there is a possibility that individuals who are amputees or have been paralysed for years may not be able to perform motor imagery mental tasks competently, so as an alternative solution, other non-motor imagery mental tasks could be used.

Further potential applications of the BCI system are actually only limited by the imagination (Allison et al. 2012). For patients with other disabilities, BCI might be used to control other technologies, such as neuroprosthesis, to restore hand motor function of patients with tetraplegia (Hochberg et al. 2012; Müller-Putz et al. 2005). Also, BCI might possibly be used for neurorehabilitation of stroke survivors (Buch et al. 2008; Mrachacz-Kersting et al. 2013). For the general population which includes able-bodied users, BCI can be applied to the transportation field. It can be used to detect a driver's mental workload and provide an alert to the driver when it detects the driver is fatigued (Craig et al. 2012; Kohlmorgen et al. 2007; Lal et al. 2003; Thuraisingham et al. 2009). Another potential application for BCI is as a new tool in the entertainment industry especially for gaming and virtual reality applications (Bonnet, Lotte & Lécuyer 2013; Nijholt, Bos & Reuderink 2009; Scherer et al. 2008).

In terms of ethical issues of BCI research, due to the involvement of people with or without a disability, a proper approval from an ethics committee is needed. BCI research requires that the subjects provide informed consent. This is to ensure a clear understanding of the BCI research and the definition of beneficence of the BCI

research. For example, BCI can be used to help people with disabilities to regain normal function in communication, mobility and so on. A concern about the risk of privacy invasion can be alleviated, as BCI requires active engagement of the user intent, and is not a mind reading device. Thus, BCI proves more beneficial for the user (Kubler et al. 2006; Vlek et al. 2012).

1.2 Thesis Objective

The aim of the research is to develop a prototype of BCI as an assistive mobility device for people with disabilities, with improved accuracy to provide effective classification of non-motor imagery mental tasks-based BCI to be used in application of a powered wheelchair control. Three objectives will be explored in this thesis.

The first objective of the research is the development of instrumentation of BCI that focuses on the mental tasks classification method. The BCI system comprises a wireless EEG (as head gear) and an embedded system with a wireless receiver as the main controller module. The hardware of the EEG has an amplifier which requires a high common mode rejection ratio (CMRR) to tolerate the interfering noise entering in the system, which should be able to detect an EEG signal of a few micro volts. The firmware will provide real-time acquisition. The system would also need to satisfy real-time aspects, including portability, power consumption and affordability.

The second objective is to develop a computational intelligence method to classify mental tasks as the mental strategy, into commands to control a wheelchair. An improved and advanced computational intelligence method will be addressed further to improve the accuracy of the system including features extraction and classification algorithms.

The third objective is the validation study, thus experiments are conducted with patients with tetraplegia as a representative of the BCI target group of individuals with severe disability. A comparison of able-bodied subjects and patients with tetraplegia can be investigated further for the validation of the study.

1.3 Thesis Contribution

This thesis presents a design of BCI as an assistive device to help in the mobility of people with disabilities, especially in the application of powered wheelchair control. The improved algorithms (features extraction and classification methods) are presented to provide effective output commands based on non-motor imagery mental tasks, which are mapped for wheelchair movements. The overall BCI works in this thesis are illustrated in Fig. 1.1. The substantial contributions of this thesis are summarised as follows:

- A prototype wireless embedded EEG system for a practical mental task BCI classification is developed. The prototype includes a wireless EEG as head gear and an embedded system with a wireless receiver. The developed wireless EEG provides a good CMRR performance and a compact size powered by a low current consumption coin cell battery. Mental tasks data is collected using the prototype system from six healthy participants; these tasks include arithmetic, figure rotation, letter composing and counting task with additional eyes closed task. The developed prototype BCI system is able to detect the dominant alpha wave between 8-13Hz during eyes closed. Using the FFT as the features extractor and artificial neural network (ANN) as the classifier, the developed prototype EEG system provides high accuracy for the eyes closed and eyes open tasks. The classification of the three mental task combinations achieve an overall accuracy of around 70%.
- An optimized BCI system for mental task classification is presented using the Hilbert-Huang transform (HHT) feature extractor and genetic algorithm optimization of artificial neural network (GA-ANN) classifier. In this study, the medical EEG system, Compumedic-Siesta, is used with eight channels that are attached to the scalp at locations *C3*, *C4*, *P3*, *P4*, *O1*, *O2*, *T3*, and *T4*. Non motor imagery mental tasks are used, including: arithmetic, letter composing, Rubik's cube rolling, visual counting, ringtone, spatial navigation and eyes closed. When

more mental tasks are used, users are able to choose the most effective of tasks suitable for their circumstance. The result of classification for the three user chosen mental tasks achieves accuracy between 76% and 85% using eight EEG channels with GA-ANN (classifier) and FFT (feature extractor). In a two EEG channels classification using FFT as the features extractor, the accuracy is reduced between 65% and 79%. However, the HHT features extractor provides improved accuracy between 70% and 84%.

- An advanced BCI system is presented using the ANN with fuzzy particle swarm optimization using cross-mutated operation (FPSOCM-ANN) for mental task classification. This experiment involves five able-bodied subjects and also five patients with tetraplegia as the target group of the BCI system. The three relevant mental tasks used for the BCI concentrates on mental letter composing, mental arithmetic and mental Rubik's cube rolling forward, and these are associated with three wheelchair commands: left, right and forward, respectively. This study uses Compumedics-Siesta system with six channels positioned at locations C3, C4, P3, P4, O1 and O2. HHT is used for the feature extractor with additional FFT and wavelet method for the comparison. For the classifier comparison, other classifiers, including GA-ANN, multi-class linear perceptron, multi-class LDA, and multi-class BSVM are also included.

The results show the HHT testing for the eyes closed EEG signal has a dominant alpha (8-13Hz) wave with high classification accuracy for able bodied subjects and patients with tetraplegia. As a result for the three mental tasks classification, HHT as the feature extractor provides an improved accuracy compared to FFT and wavelet methods. Although the patients group has lower classification accuracy, this is improved by increasing the time-window of data with the best at 7s. The resulting classification accuracy of three mental tasks for 7s time-window show the best classifier is using the FPSOCM-ANN (84.4% using FPSOCM-ANN, 77.4% using GA-ANN, 77.0% using SVM, 72.1% using LDA and 71.0% using linear perceptron). For practical use of a BCI, the

two channels EEG is also presented using this advanced BCI classification method (FPSOCM-ANN). Overall, O1 and C4 are the best two channels at 80.5% of accuracy, followed by the second best at P3 and O2 at 76.4% of accuracy, and the third best at C3 and O2 channels at 75.4% of accuracy.

1.4 Thesis Outline

The thesis consists of six chapters that are organised as follows:

- **Chapter 1** – Introduction
- **Chapter 2** – Literature Review
- **Chapter 3** – Wireless Embedded EEG System for Mental Task Classification
- **Chapter 4** – Optimized BCI using Hilbert-Huang Transform Feature Extractor and Genetic Algorithm-Artificial Neural Network Classifier
- **Chapter 5** – Advanced BCI System using Fuzzy Particle Swarm Optimization of Artificial Neural Network
- **Chapter 6** – Conclusion and future work

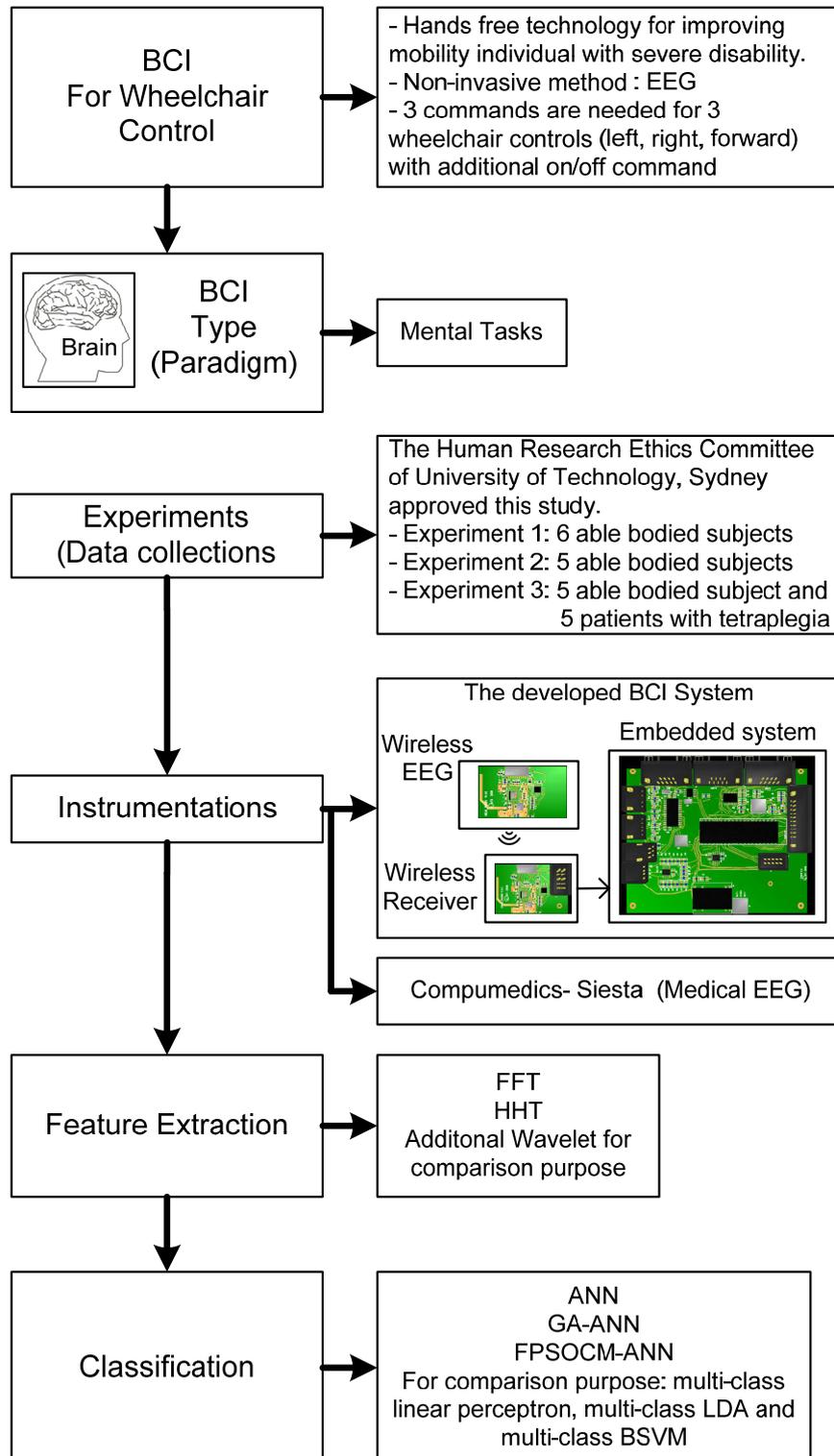


Figure 1.1: Overall BCI works in this thesis

1.5 Thesis Publications

Published Journal paper:

- [1] **R. Chai**, S. H. Ling, G. P. Hunter, Y. Tran, and H. T. Nguyen, “Brain Computer Interface Classifier for Wheelchair Commands using Neural Network with Fuzzy Particle Swarm Optimization”, IEEE Journal of Biomedical and Health Informatics (Chai et al. 2013a) (In Press/Accepted on December 2013).

Published Conference papers:

- [2] **R. Chai**, S. H. Ling, G. P. Hunter, Y. Tran, and H. T. Nguyen, “Classification of Wheelchair Commands using Brain Computer Interface: Comparison between Able-Bodied Persons and Patients with Tetraplegia “ in Proc. of the 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Osaka, 2013, pp. 989-992(Chai et al. 2013b).
- [3] **R. Chai**, G. P. Hunter, S. H. Ling, and H. T. Nguyen, “Real-Time Microcontroller based Brain Computer Interface for Mental Task Classifications using Wireless EEG Signals from Two Channels “ in Proceedings of the 9th IASTED International Conference on Biomedical Engineering (BioMed 2012), Innsbruck, 2012, pp. 336-342 (Chai et al. 2012a).
- [4] **R. Chai**, S. H. Ling, G. P. Hunter, and H. T. Nguyen, “Mental non-motor imagery tasks classifications of brain computer interface for wheelchair commands using genetic algorithm-based neural network,” in Proc. of the 2012 International Joint Conference on the Neural Networks (IJCNN), Brisbane, 2012, pp. 1-7 (Chai et al. 2012b).

- [5] **R. Chai**, S. H. Ling, G. P. Hunter, and H. T. Nguyen, “Mental task classifications using prefrontal cortex electroencephalograph signals,” in Proc. of the 34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), San Diego, 2012, pp. 1831-1834 (Chai et al. 2012c).
- [6] **R. Chai**, S. H. Ling, G. P. Hunter, and H. T. Nguyen, “Toward fewer EEG channels and better feature extractor of non-motor imagery mental tasks classification for a wheelchair thought controller,” in Proc. of the 34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), San Diego, 2012, pp. 5266-5269 (Chai et al. 2012d).

Submitted Journal paper:

- [7] **R. Chai**, S. H. Ling, G. P. Hunter, and H. T. Nguyen, “A Hybrid Brain Computer Interface for Biomedical Cyber-Physical System Application using Wireless Embedded EEG Systems,” IEEE Transactions on Industrial Informatics (Revised).

Chapter 2

Literature Review

2.1 Introduction: Brain Neurophysiology

The brain, particularly in humans, is known to have a very complex architecture. It is the main part of the central nervous system (CNS), including all physiological and cognitive functions. The brain is responsible for important functions, including body movement, memory, thought, emotions and sensing. The brain controls unconscious activities, such as breathing, pulse and digestion. It also controls conscious activities, such as thinking, planning, reasoning and feeling.

The central nervous system consists of the brain and the spinal cord, which has billions of neurons making up the neurological connections that work together to carry out complex functions. A single neuron, as illustrated in Fig. 2.1 (Bear, Connors & Michael 2007) is the basic unit of information processing and which actually has a large complex network consisting of:

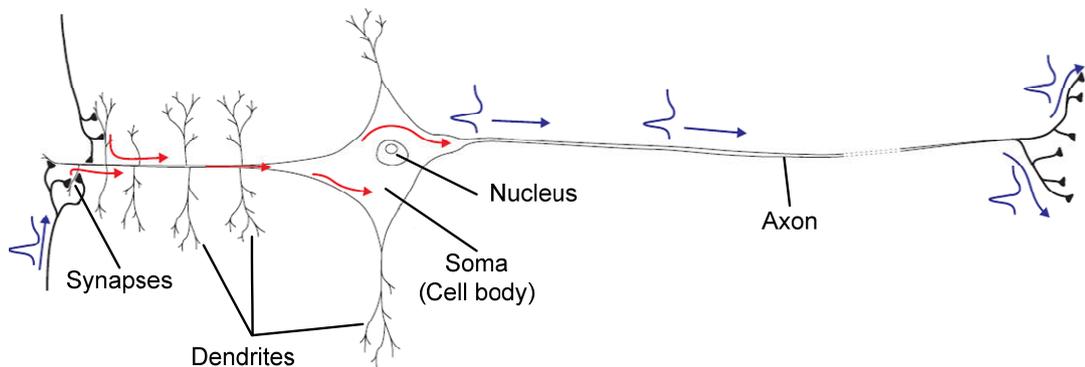


Figure 2.1: The structure of neuron

- Soma, known as the cell body and metabolic centre of the neuron. At the centre, the nucleus has chromosomes which contain deoxyribonucleic acid (DNA). This DNA was passed from the parent and provides the genetic material, the blueprint for the entire body.
- Axon, which is a single long fibre for conducting neural signals from the cell body (soma) to other parts of the nervous system. It is specialized for the information transfer over distances in the nervous system. It begins with the region called axon hillock, where the axon meets the cell body and aggregates all the received action potentials. The axon ends at the presynaptic axon terminal, where the contact of the axon, with other neurons or other cells, passes information.
- Synapse, which is the point of contact of the axon with other neurons and has two sides: the presynaptic side, which has its axon terminal, and the postsynaptic side, which can be the dendrites or soma of another neuron. The space between presynaptic and postsynaptic membranes is the synaptic cleft. At synapses, the information, in the form of electrical impulses, travels to the axon and is converted in the terminal into a chemical signal, known as a neurotransmitter, which crosses the synaptic cleft. On the postsynaptic side, the chemical signal is converted back into an electrical signal.

- Dendrites, which are like the branches of a tree that extend from the soma. There are a variety of shapes and sizes of the dendrite trees for different groups of neuron classification. As the function of dendrites refers to the antenna of the neuron, they are covered with a synapse (postsynaptic membrane).

In general, a neuron accepts inputs from other cells via its dendrites and sums up all of them in some way. If the result of the addition satisfies some neuro-dependent condition, which is determined by the trigger zone near the axon hillock, the cell sends out a new signal via axon output. The neurons are composed mainly of fluids contained in very thin semi-permeable membranes, whereby the interchange of molecules and ions between inside and outside is restricted, but is not entirely shut off. The fluid inside a neuron has a high concentration of potassium K^+ and a low concentration of sodium Na^+ and chloride Cl^- ions. On the other hand the concentration of Na^+ and Cl^- is high and the number of K^+ ions is low outside the cell. The different ionic concentration gives rise to an electrical voltage, which can be described by the Goldman equation as follows:

$$V_m = \frac{RT}{F} \ln \left(\frac{P_K [K^+]_o + P_{Na} [Na^+]_o + P_{Cl} [Cl^-]_o}{P_K [K^+]_i + P_{Na} [Na^+]_i + P_{Cl} [Cl^-]_i} \right) \quad (2.1)$$

where V_m denotes the membrane potential; R is the gas constant; T is the absolute temperature; F is the Faraday's constant, and; P_K , P_{Na} , and P_{Cl} are the relative permeability of the membrane to three ions (potassium, sodium and chloride). For a typical neuron at rest, the membrane potentials are around -70 mV (Bear, Connors & Paradiso 2007; Biswal et al. 2010).

The forces of nature tend to move free particles from higher concentrations to a lower one, therefore, if it is not controlled, the ion tends to move via the cell membrane until the concentration inside the cell equals the concentration outside, known as chemical equilibrium. Since ions surrounding the cell membranes are charged electrically, they can also be affected by the electrical forces that are trying to move

positively charged ions to areas of negative charge, and negatively charged ions to a positive area in order to achieve electrical equilibrium by eliminating the voltage over the membrane. The total force on the system due to the chemical and electrical imbalances is known as the electrochemical gradient. This total gradient is maintained by a sodium-potassium pump that continuously pumps K^+ ions into the cell and Na^+ ions out through the membrane (Bear, Connors & Paradiso 2007; Biswal et al. 2010).

The axons transmit information of a series of identical electrical impulses, known as action potentials. This action potential is initiated as a result of processing inputs collected by the dendrites. It further causes the membrane potential of the axon hillock to become less negative than the resting state. If the potential across the membrane achieves a less negative value than a certain value, there is a sudden change in the cell membrane permeability at that point. It becomes highly permeable to sodium (Na^+) ions and is much more permeable than it was to potassium (K^+). The sodium ions enter the neuron down their concentration of electrochemical gradient and cause the inside to become more positive compared to the outside, effectively reversing the potential across the membrane. After a few milliseconds, the permeability to Na^+ drops back to normal condition and the membrane reverts back to its original state with the exit of K^+ ions. The key process of changing the membrane permeability to Na^+ lies with a specialized protein in the membrane called the voltage sensitive Na^+ channel. At resting membrane potential (RMP) of about -70 mV, the channel is closed, and when depolarized beyond threshold potential value, the Na^+ channel opens transiently and closes again (Bear, Connors & Paradiso 2007; Biswal et al. 2010).

In the bigger perspective, the human brain has three distinct parts which are: the large convoluted cerebrum, the rippled cerebellum and the brain stem. The cerebrum has two hemispheres: the left and the right. The left hemisphere senses and controls the right side of the body, and the right hemisphere senses the left side of the body. The layer for holding the networks in the brain together is known as the cerebral cortex. From the aspect of cerebral functioning, the cortex can be divided into four lobes:

frontal, parietal, occipital and temporal, as shown in Fig 2.3 (Bear, Connors & Michael 2007; Biswal et al. 2010).

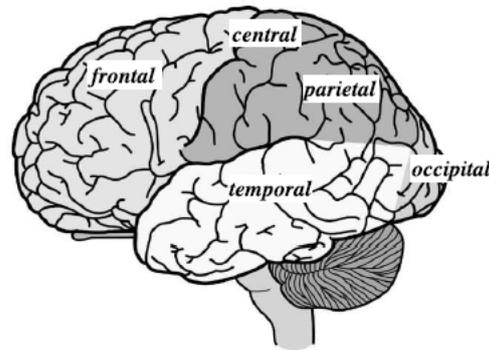


Figure 2.2: Different areas of the cerebral cortex

This cerebral cortex seems to be the centre for higher-order functions of the brain, which are: vision, hearing, movement, sensing and planning. The frontal lobe includes the prefrontal area responsible for higher-order executive functions, such as complex cognitive behaviours, personality and decision-making. The parietal lobe is a region of the multimodal association cortex that receives input from the somatosensory, visual and auditory sensory area that surrounds it. The parietal lobe is responsible for tactile discrimination and motivation, reading, writing and arithmetic calculation. The occipital lobe located at the posterior pole of the brain is responsible for visual area. The temporal lobes are located ventrally along the sides of the brain and are responsible for auditory signal processing, higher-level visual processing and memory. The primary motor cortex is located in the frontal lobe and is closely related to movement control of the particular body areas. This area is organized into motor homunculus with a shaped body that is drawn along the central sulcus, which represents legs, feet, trunk, arm, hand, face and so on (Nolte 2002).

So far, the cerebral cortex is the area most widely used in brain computer interface (BCI) research due to the high involvement for the executive for cognitive function including motor cognitive, communication behaviours and other mental cognitive tasks.

These tasks can be used directly for BCI classifications. Moreover, the cerebral cortex is the most accessible area for the BCI sensors for both invasive and non-invasive types.

2.2 Brain-Computer Interface (BCI) Components

The functional basic components of the BCI and the interaction between each part is illustrated in Fig 2.3 and consists of several elements, including: 1) signal measurement and data acquisition, 2) computational intelligence or algorithms covering signal pre-processing, features extraction and classification or translation algorithm, 3) output of the classification as outputs for BCI application and feedback to the user (He et al. 2013; Mason & Birch 2003; Wolpaw et al. 2002).

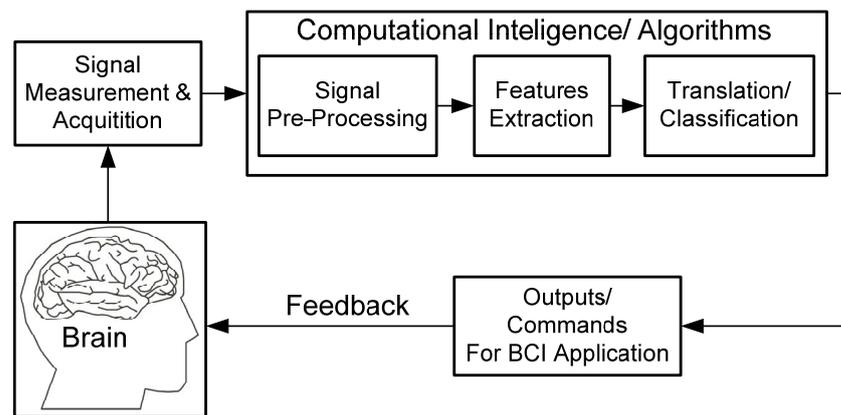


Figure 2.3: Block diagram components of BCI

The processes start by measuring brain signals with the BCI instrumentation to provide data acquisition. The output of the data acquisition process is in a digital form which will be further enhanced by the process of signal pre-processing to filter the noise. This is continued with the features extraction, which transforms the signals into useful features, which correspond to the mental strategy of the BCI. The features will be fed into the classification or translation method, which will give the output of command or logical control signals for different BCI applications. The specific function is elaborated in the following:

-
- **Signal measurement/Acquisition:** this is the first step of the BCI system of using instrumentation and sensors to record the brain activity. The instrumentation can be based on an invasive BCI that records the signal inside the brain, or non-invasive BCI by using electrodes mounted on the surface of the scalp. For the instrumentation, the brain signal being measured is amplified, digitized and transmitted for further signal processing using computational intelligence.
 - **Signal pre-processing:** This step is applied for de-noising the raw brain signal in order to enhance the relevant information embedded in the signals. After the digital raw data is acquired, some further digital filtering and noise reduction algorithms can be used to enhance the signal-to-noise ratio (SNR).
 - **Features extraction:** It would be difficult to classify directly from the raw brain signal, as the brain signal is non-stationary, high dimensional and non-linear. Therefore, this step is used by applying algorithms to extract the important features of the brain signal. Different features algorithms can be viewed from time, frequency, and spatial domain perspectives.
 - **Translation/Classification:** This step is to use an algorithm to recognize particular important patterns of the brain signal. The algorithm can utilize linear or non-linear classification algorithms and can be based on a self-learning algorithm that is trained to recognize the features of the brain signal.
 - **Output:** the result of the classification step is to identify outputs which can be translated as commands associated with a particular brain signal to control different BCI applications, such as wheelchair control, motor prosthesis, alerts as to drowsiness during driving, rehabilitation of stroke patients, games and virtual reality.
 - **Feedback:** This is as an additional step to provide BCI as a closed loop system and confirm to the user the result of the classified output.

2.3 State of Art BCI Technology

From the recording method, BCI technology basically can be classified into invasive and non-invasive methods of recordings, as shown in Fig 2.4. For the invasive method, prior to the measurement, a surgical operation is needed to insert the sensor into the brain to acquire the brain signal. Two examples that can be found for invasive BCI methods: intra-cortical recording or local field potential using microelectrodes and electrocorticography (ECoG). Both examples are based on the measurement of electrical potential.

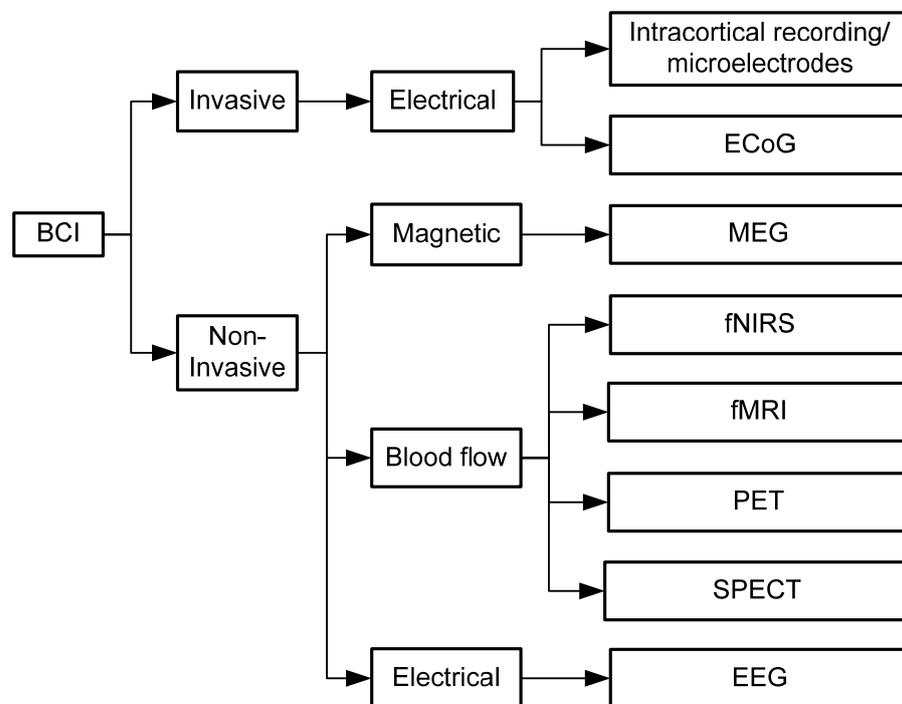


Figure 2.4: BCI classifications from recording methods

On the other hand, for the non-invasive methods, there are several ways to acquire the brain activities without performing surgery, including: 1) from electrical potential measurements using electroencephalography (EEG), 2) from magnetic measurements using magneto-encephalography (MEG), 3) blood flowing measurements using functional near infrared spectroscopy (fNIRS), functional magnetic resonance imaging

(fMRI), positron emission tomography (PET) and single photon emission computed tomography (SPECT) (He et al. 2013; Hochberg et al. 2006; Nicolas-Alonso & Gomez-Gil 2012; Wolpaw et al. 2006).

2.3.1 Invasive BCI

2.3.1.1 Intra-cortical Recording (Microelectrodes)

Intra-cortical recording uses implanted microelectrodes to record electrophysiological activity or action potential from single neurons, such as the local field potentials (LFP) that reflect the combined activity of surrounding neurons and synapses. This can provide the highest resolution of signals including temporal and spatial resolutions. There are some microelectrode types which have been used in BCI systems such as the Utah intracranial electrode array (UIEA) that contains 100 penetrating silicon electrodes in a 10×10 array that are placed on the surface of the cortex with the needles penetrating into the brain.

Due to the invasive nature of this method, including risks of inserting multiple electrode arrays within brain tissue, unresolved issues relating to acute, chronic tissue damage and long term stability of microelectrodes (Grill & Mortimer 2000; Polikov, Tresco & Reichert 2005; Prasad & Sanchez 2012; Szarowski et al. 2003; Vetter et al. 2004), testing has occurred mostly in animals. For example, the number of studies with humans has reduced, monkeys have been used instead. During these experiments, by presenting the monkey with suitable feedback about the firing rate of single neurons, the monkeys were able to learn to control the feedback (Musallam et al. 2004; Nicolelis et al. 2003; Shenoy et al. 2003). The recording activity of neurons, especially in the brain motor area, allows for an application, such as the robot arm 3D-control. In a further application, microelectrodes have been implanted in the motor cortex area of a human, a patient with tetraplegia with the aim to replace and restore lost motor function (Donoghue 2002; Hochberg et al. 2012; Hochberg et al. 2006; Serruya et al. 2002).

2.3.1.2 Electrocorticography (ECoG)

ECoG is the invasive BCI method that records the electrical signals from the location underneath the skull, but not within the brain itself. The grid of electrodes is placed directly on the surface of the cortex without penetrating the brain. The ECoG involves surgery, but the electrodes are placed on the surface of dura matter (epidural) using screws, which serve as the electrodes by penetrating the skull. Alternatively, electrical signals can be recorded from beneath the dura matter (subdural) using electrodes placed directly on the surface of the brain. This means that ECoG provides less invasive and lower signal quality compared to intra-cortical recording, but better comparison parameters to EEG, including amplitude, topographical resolution, frequency range, and resistance to artefacts (Henle et al. 2013; Leuthardt et al. 2004).

So far, the experiment of ECoG-based BCI were not performed with healthy subjects, but mainly with patients suffering from epilepsy, with the ECoG arrays implanted over a period of one or two weeks for localization of epileptic foci or for pre-surgical monitoring reasons. It is therefore sometimes the location of ECoG electrodes that are not under the control of the BCI researcher, but mostly related to the epileptic symptoms of particular patients, that need to undergo the epilepsy surgery. ECoG refers to intracranial electroencephalography (iEEG). Compared to EEG, ECoG actually has several advantages, including: 1) higher spatial resolution (1.25 mm for subdural recording (Freeman et al. 2000; Henle et al. 2013) and 1.4 mm for epidural recording (Slutzky et al. 2010)) compared to several centimetres for EEG, 2) higher amplitude of EEG signal; less sensitivity to ocular and muscular artefacts (Ball et al. 2009), 3) broader bandwidth (0-500 Hz for ECoG compared to 0-40 Hz EEG) (Gaona et al. 2011; Staba et al. 2002). ECoG is able to archive low impedance, and the surrounding skull bone also provides shielding against muscle artefacts. The experiments have shown the ECoG can be applied to BCI through motor imagery (Grazimann et al. 2004; Schalk et al. 2007), speech imagery (Leuthardt et al. 2006), mental calculation (Ramsey et al. 2006), and auditory imagery (Wilson et al. 2006).

There are some limitations using ECoG, especially as an invasive method, and the placement of the electrodes at the moment requires a surgical operation to implant. Furthermore, current ECoG-based BCI that apply to humans is limited to people who have been temporarily implanted, such as for a week, for example, the placement of ECoG electrodes array for localization of seizure and essential cortical functions for curing of epilepsy.

2.3.2 Non-invasive BCI

2.3.2.1 Magneto-encephalography (MEG)

Beside electrical measurement, brain-current sources can also generate an external magnetic field. This field can be detected with magneto-encephalography (MEG) as a non-invasive method. The sensors of MEG are in the form of induction coils statically arranged in a helmet, and the head position of the user relative to the helmet is controlled by using three localization coils attached to the head. This MEG method records the small magnetic field generated by the brain using a superconducting quantum interference device (SQUID) magnetometer. Individual magnetometers can be arranged in different configurations for different purposes. An array of 100-200 magnetometers provides coverage of the whole head, and each magnetometer detects only the radial component of the magnetic field of the brain sources (Burmistrov et al. 2013; Mellinger et al. 2007; Sato 1990).

MEG has a similar high temporal resolution and low spatial resolution to EEG. The advantage of MEG compared to EEG is that MEG provides a true field measure at a specific point, while EEG measures the potential difference between two points on the head. Therefore, MEG measurement does not require the choice of a reference sensor. The main difference between MEG and EEG is that they are sensitive to different sources preferentially. The chosen method depends on the location, orientation, and size of the source region. The main advantage of MEG for source localization is that the model of the spread of the magnetic field is much simpler and much more understood than models of volume conduction of electric current. For the limitations, the largest

artefact for MEG is generated from the geomagnetic field of 8-9 orders of magnitude greater than the magnetic field produced by the brain; therefore, to get a good signal quality, it requires a room with a magnetic shield. This shield could also be used as a shield against artefacts from electrical devices. Subjects who are using the MEG equipment must remove all magnetic materials they have with them, as high sensitivity of MEG to this moving magnetic object could interfere. The same as EEG, eyes and muscle artefacts also could provide interference to the EMG measurement. The main disadvantage of MEG is cost and the huge complexity of the instrumentation.

2.3.2.2 Functional Near Infrared Spectroscopy (fNIRS)

Functional near infrared spectroscopy (fNIRS) is a non-invasive method that is based on a blood flow measurement by tracking changes in different forms of haemoglobin, which is used to measure the hemodynamic activity of the cortex. It reveals brain activity for only the top few millimetres of cortex. It is basically blood-oxygen-level-dependent (BOLD) response technology that measures the changes in the relative levels of oxy-haemoglobin (oxy-Hb) and deoxy-haemoglobin (deoxy-Hb) during brain activity. When a subject performs a particular task, brain activity increases in those particular brain areas relevant to the task and the amounts of oxy- and deoxy-Hb change in the immediate vicinity of those areas. The changes in the oxy- and deoxy ratio can be detected because of the absorbance spectra in the near infrared wavelength zone differ from oxy- and deoxy-Hb. In fNIRS, light with a wavelength of 690 and 830 nm is shined on the skull. This causes some of the light to be absorbed by the brain tissue. Some of the light is scattered and reflected back out of the head. Less reflection means the light is absorbed. Since both deoxy-Hb and Oxy-Hb have different patterns of absorbance at 690 and 830 nm, the absorbance at a particular wavelength is proportional to the concentration of the absorbing molecules. The measurement of the reflected light reveals the concentration of the absorbing molecules deoxy-Hb and oxy-Hb (Fazli et al. 2012; Villringer et al. 1993).

The instrumentation of fNIRS consists of light source-detector pairs positioned on the head. The best signal is obtained from the forehead as hair could block the passage of light and the hair movement could disturb the signal. Other positions are possible, but they would require a careful repositioning of the hair that blocks the sources and the detectors.

The fNIRS is still a new technology for BCI, with advantages including; being relatively easy to use, non-invasive, portable and relatively inexpensive compared to other blood-flow measurement technologies. The disadvantages of the fNIRS systems include a lower temporal resolution compared to the EEG due to the neurovascular coupling, relatively low for spatial resolution of the order of centimetres. Several experiments have been done with fNIRS-based BCI for healthy subjects with observation during motor imagery (Coyle et al. 2004; Coyle, Ward & Markham 2007; Ranganatha, Hoshi & Guan 2005; Zimmermann et al. 2013). Moreover, investigation of fNIR for people with severe disabilities also has been implemented (Naito et al. 2007). However, this method has limited temporal resolution since it is based on BOLD response, which is relatively slow.

2.3.2.3 Functional Magnetic Response Imaging (fMRI)

Functional magnetic response imaging (fMRI) is similar to as fNIRS as a non-invasive method, and is basically a BOLD response method. It measures the relative amount of oxy- and deoxy-Hb in the brain tissue. It is one of the most widely used techniques for imaging brain function and provides excellent spatial detail by producing maps of the hemodynamic response to brain activity based on the effect that local deoxy-Hb has on the magnetic field. A magnetic resonance imaging (MRI) scanner consists of six key components, including a large magnet, radio frequency transmitter, receiver antenna, gradient coils, front-end computer and reconstruction computer. (Ogawa et al. 1990; Sorger et al. 2012).

The use of fMRI-based BCI is not very practical at the present time due to the high expense, bulky equipment, need for liquid helium to cool the superconducting coils and

the need for a radio frequency (RF) shielded chamber. However, since fMRI activity has a strong relation with electrical activity in the cortex, the fMRI imaging can be used to identify the most promising brain region for BCI application (Hermes et al. 2012; Ramsey et al. 2006). This can be used for supporting the invasive BCI method, such as the microelectrode arrays that can not be implanted for the entire area of the cortex. By using the fMRI, it could identify the best location for implanting arrays. Moreover, because of the non-invasive method, fMRI also can be utilized in large control studies with the aim to identify brain locations and functions for developing a training protocol of BCIs for people with disabilities.

2.3.2.4 Positron Emission Tomography (PET)

Positron emission tomography (PET) is a nuclear medical imaging method that produces three-dimensional maps of the brain function and metabolic processes by detecting changes in blood flow. It requires the injection of radioactive materials prepared in a cyclotron. Note that the radioactive isotopes used have short half-lives and are in small doses to avoid harmful biological effects.

The process is started by inserting radioactive oxygen into water molecules to create a radioactive water compound, which is injected into the brain. Soon after the injection of the marker, the PET scanner starts to collect images by measuring radiation with ring detectors in the scanner. When a radioactive oxygen atom releases a positron that collides with an electron in the immediate surroundings, the two particles annihilate each other and emit two gamma photons which travel in opposite directions. These gamma photons hit a scintillator, a block made of crystals that surround the subject's head in the scanner. A flash of light occurs when a photon hits the crystal blocks, and this flash is detected by an optical camera arranged directly outside the crystals. The image from the PET scanner displays the density distribution of the radioactive tracers in the brain. When this particular brain region is active, blood flow in the region increases, more radioactive water flows to the brain tissue and PET detects the

radioactivity to create an image of the brain (Cherry & Phelps 2002; Decety et al. 1994; Juweid & Hoekstra 2011).

The drawbacks of PET are for temporal resolution. The PET needs at least 40s to collect information to construct an image with the spatial resolution of about 5 mm. This scanner needs a large area as well, and as it is an expensive system, these are issues to consider when using PET as a practical BCI system. So far, PET scanning is widely used for the diagnosis of brain disease such as tumours, strokes and dementia, where great changes in brain metabolism are apparent.

2.3.2.5 Single Photon Emission Computed Tomography (SPECT)

Single photon emission computed tomography (SPECT) is another nuclear imaging technology and works in a similar way to the PET scanner to measure blood flow. However, SPECT uses a photomultiplier tube to measure the photon generated by gamma rays, while PET emits positrons. The test of the SPECT differs from the PET scanner, in that the tracer stays in the blood stream rather than being absorbed by surrounding tissue. As a result, it limits the images to areas where blood flows. The key idea of SPECT is that a radiopharmaceutical emitting gamma rays is injected in the blood vessel prior to the experiment. These radioactive tracers are transported via the cardiovascular system in a way that their concentration is high in the regions with high blood flow so the gamma activity from a certain brain region is able to serve as an indicator for activity in this region.

The SPECT scanner is cheaper and more readily available than the high resolution PET scanner, but still needs a bulky machine to operate. This becomes impractical and a drawback for BCI application when coupled with the constant need to inject radioactive substances for measurement. Currently this method is mainly used as the scanner for patients with stroke and tumour (Holly et al. 2010; Van Heertum, Tikofsky & Ichise 2000).

2.3.2.6 Electroencephalography (EEG)

EEG is the most widely used non-invasive method in BCI research to record electrical potential of brain activity by using electrodes placed on the scalp (McCane et al. 2014; Wolpaw et al. 2002). These scalp electrodes are sensitive for picking up the mixed activity of a large population of neurons, rather than the activity of a single neuron. These electrodes are only sensitive to macroscopic coordinated firing of large groups of neurons and only when they are directed along a perpendicular vector relative to the scalp. Moreover, as the fluid, bone, and skin separate the electrode from the actual electrical potential, the small signals are scattered and attenuated before reaching the electrodes. The EEG signal on the scalp is within the range 5-300 μV therefore for instrumentation, the amplification module is necessary at certain gain. To make sure there is proper contact with the scalp during EEG measurement, contact impedance requires less than 10 $\text{k}\Omega$ or even less than 5 $\text{k}\Omega$. To make this achievable, a scalp preparation is needed. The scalp is abraded at the electrode site using a conductive gel or paste between the electrode and the scalp. This conductive-gel based electrode is often referred to as a wet electrode. The EEG electrodes are usually made from tin (Sn), silver/silver-chloride (Ag/AgCl), gold (Au), or platinum (Pt). The tin electrodes are the least costly, but provide noise in low frequency below 1 Hz. Most commercial EEG electrode systems use silver/silver-chloride or gold electrodes. The gold electrodes also minimize drift and show less noise in high frequency compared to silver/silver-chloride (Niedermeyer & da Silva 2005).

The advantage of EEG-based BCI instrumentation is that it can be designed to be low cost and portable compared to other BCI methods, even though it has the drawback of weak signals and higher sensitivity to ocular and muscular noises. The EEG also provides high results in terms of temporal resolution, but has low spatial resolution. Non-brain physiological sources could affect the EEG recording, including muscle activity measured by electromyography (EMG), eye movements measured by electrooculography (EOG) and heart muscle activity measured by electrocardiography (ECG). Muscle artefacts are broadband artefacts found above 10 Hz. Eye movements,

eye blinks and heart muscle activity have a stereotypical wave form that can be easily identified in the EEG signal. A power line noise (50/60 Hz) can also contaminate the EEG signal; therefore many commercial EEG systems have a notch filter to accommodate the power line signal. More details of state of the art BCI-EEG will be discussed further in this chapter.

2.4 EEG-based BCI

2.4.1 EEG Rhythm and Measurement

In terms of the frequency ranges, EEG signals can be divided into rhythms, which are: delta (δ), theta (θ), alpha (α), beta (β) and gamma (γ). Delta rhythm has a frequency range of 1-4 Hz which can be found in adults during deep sleep. Theta emerges at a frequency range of 4-7 Hz and can be found normally during drowsiness. Alpha range of frequency of 8-13 Hz can be found clearly on the occipital lobe area during eyes closed and relaxation. Mu (μ) rhythm is in the same alpha range of the signal which affects the sensorimotor area. Beta rhythm has a frequency range of 14-30 Hz which can be observed in the awake, active and conscious state of mind. Above 30 Hz is gamma rhythm's frequency range, which is associated with cognitive and motor functions (Niedermeyer & da Silva 2005).

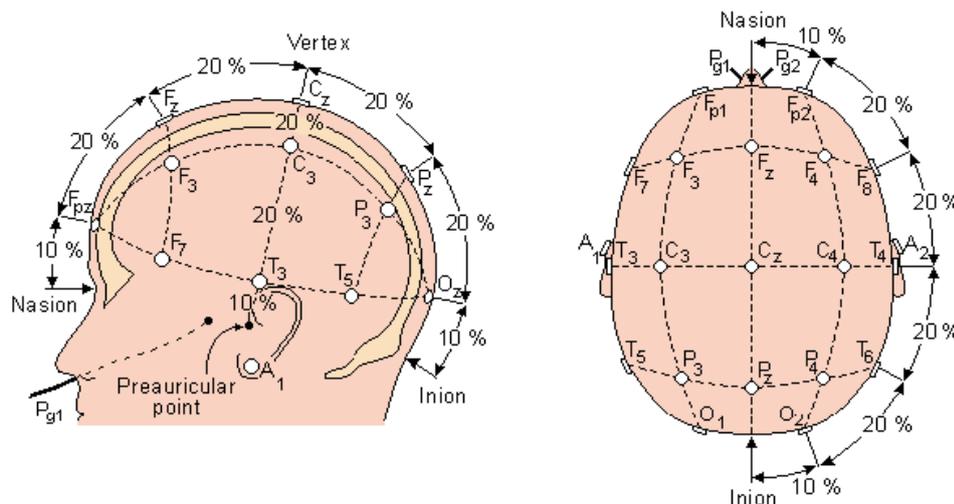


Figure 2.5: The international ten-twenty (10-20) system for electrode placement

To conduct measurements with the EEG system, electrodes will be placed on the scalp. The international 10-20 system (Jasper 1958; Klem et al. 1999) is the standard which provides electrode placement on the entire scalp. Even numbered electrodes are used for placement on the right side of the head and odd number electrodes are placed on the left side. The placement is along the bisecting line starting from the nose (nasion) to the back of the head (inion). The notations are *F* (frontal), *C* (central), *P* (parietal), *O* (occipital), *T* (temporal) and *A* refers to ear lobe. The 10-20 refer to the 10% and 20% of the inter-electrode area on the scalp as shown in Fig 2.5 (Klem et al. 1999).

For electrodes, the most commonly used EEG electrodes are made of gold, tin or silver/silver-chloride (Ag/AgCl). The impedance of the skin-electrode junction is called the electrode impedance, and is one of the important factors in determining the quality and stability of EEG recordings. The conducting electrode gel is placed between the electrode and the scalp, decreasing the impedance level. As a result, it allows current to travel more easily from the scalp to the sensor. The impedance should be below 5 k Ω for good EEG recording (Fisch & Spehlmann 1999).

The basic instrumentation of EEG basically comprises amplifiers block, analog-to-digital converter (ADC) and filters. Instrumentation of the EEG amplifier requires a high common mode rejection ratio (CMRR) above 80 dB to tolerate the interfering noise into the system (MettingVanRijn, Peper & Grimbergen 1994). The amplifier should also be able to detect EEG signal within the range of 5-300 μ V (Webster 2010). The recorded brain signals are basically analog signals. Therefore, further an ADC is needed to convert the analog signal into the digital signal. An ADC digitized signal from each electrode many times per second is called a sampling rate. To accurately acquire and reconstruct the information present in the signal, the sampling frequency must meet the requirement of the Nyquist criterion, which must be at least two times larger than the highest frequency occurring in the signal. Otherwise, the digital signal will be distorted by aliasing. As the biological signal typically contains a different range of frequencies, as a result, a low-pass filter, high-pass filter, and notch filter can be added. A low-pass filter allows low frequencies to pass and blocks higher frequencies.

A high-pass filter allows high frequencies to pass and blocks lower frequencies. The combination of low-pass and high-pass filters can be used; these are called band-pass filters. The notch filters are used to suppress interference from power lines of 50 Hz or 60 Hz (Wilson, Guger & Schalk 2012).

2.4.2 Existing EEG-based BCIs

In Fig. 2.6, the BCI-EEG viewed from mental strategies can be divided into selective attention or spontaneous mental signal methods. The P300 and the steady state visual evoked potential (SSVEP) is the example for the selective attention method. They are based on event-related potentials (ERP) in the EEG, which are triggered by a specific event. For these, the user needs to concentrate on external stimuli that flash in succession (P300) or continuously in a certain frequency (SSVEP). BCI systems relying on spontaneous mental signals generated voluntarily by the user may include self-regulation of the slow cortical potential (SCP), sensorimotor rhythm (SMR) or motor imagery task, which is based on event-related desynchronization/synchronization (ERD/ERS), and other mental tasks or non-motor imagery mental tasks.

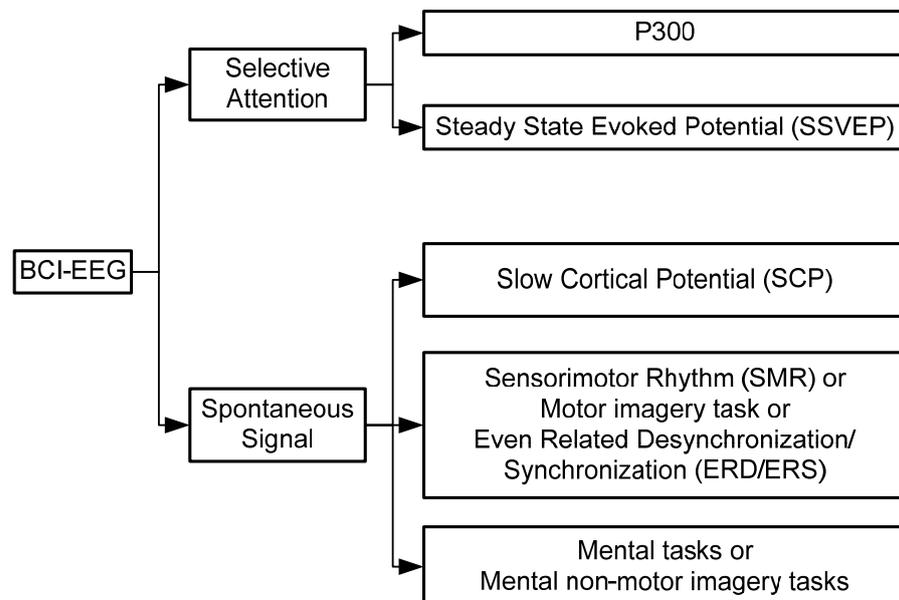


Figure 2.6: Existing BCI-EEG methods

2.4.2.1 P300

The P300 is a positive deflection that happens in the recorded EEG after about 300ms to deliver a stimulus under specific circumstances, with the largest over the central parietal scalp and attenuates gradually by increasing of the distance of this area. Although it often happens at about 300 ms relative to the eliciting stimulus, the latency may vary between 250 ms to 740 ms. This is because P300 is elicited by a decision, which is not necessarily a conscious one of the occurred event, and the decision latency which can also vary with the nature of the decision (Comerchero & Polich 1999; Käthner et al. 2013; McCane et al. 2014; McCarthy & Donchin 1981; Polich 2007). P300, in 1988, was first used for the BCI system and has been growing steadily since then for the BCI application (Farwell & Donchin 1988). BCI using P300 on the current state allows the user to select items displayed on the computer screen, therefore providing P300-based BCI selection, which is essentially equivalent to a computer keyboard of selection. The context-updating model is the most comprehensive account model of the functional role of P300. It proposes that P300 reflects context-updating operations. This model stimuli was presented and evaluated to the degree to which the events are consistent with the current model of the context being assessed. A P300 is elicited when an event violates the expectations dictated by the model and when the violation needs the revision of the model. The model accounts for some of the salient characteristics of the P300 and is supported by a variety of behavioural and psychophysiological studies (Barcelo, Perianez & Nyhus 2007; Dien, Spencer & Donchin 2003).

P300 in the BCI-EEG application has advantages, including that it is non-invasive, can be parameterized to a new user in a few minutes, requires minimal user training and can provide basic communication and control functions. There are three attributes of the paradigm for P300-based BCI to be served for communication and control, including: stimuli as the representation BCI outputs are presented in a random order, the stimulus representing each possible output is presented rarely and the BCI users are asked to

attend to the stimulus that represent the output of users' desires. P300-based speller is one of the P300-based BCI. Here, the subject views a 6×6 matrix of letters and commands with the stimulus events that flashes an entire row or column of the matrix, and the user needs to attend to given letters to keep a running mental count of the number of times that letter flashes.

There is an issue in BCI application, as P300 amplitude is positively correlated with the time interval between events. The amplitude of P300 in a standard oddball experiment is around 10-20 μV , whereas for BCI application, P300 amplitude is around 4-10 μV . This is caused by the rapid stimulus presentation rates used by P300-based BCI and the resulting overlap of the ERP to successive stimuli (Martens et al. 2009; Woldorff 1993). P300 amplitude is affected by moment-to-moment changes in probability of the stimulus, and is also affected by the sum total of the subject's concurrent activities. It is therefore P300 amplitude which decreases when a subject is performing a P300 task and is asked to perform a second task at the same time. The major disadvantage of P300-based BCI, is the inconvenience to the users who need to concentrate on the environment for control and the stimuli at the same time, e.g. wheelchair control. P300-based BCI performance may depend on the user's ability to look directly at the desired item (McCane et al. 2014).

2.4.2.2 Steady State Visual Evoked Potential (SSVEP)

SSVEP-based BCI is part of visual evoked potential (VEP) by using external sudden visual stimuli including a flash of light, the appearance of an image, and change in colour or pattern. They are generated near the primary visual cortex, therefore, they can be found mostly in the occipital lobe. The steady-state VEP is based on the stable oscillation in voltage, elicited by repetitive stimulation, for example a strobe light, light-emitting diode (LED), or a pattern checker box presented on a display monitor. SSVEP is analysed by frequency analysis, which reveals a peak at the frequency stimulation, as well as peaks at higher harmonic frequencies. In the SSVEP paradigm, the user is presented with repetitive stimuli at a fixed frequency that differs from the other stimuli.

Each stimulus refers to a specific BCI output. The user then makes a selection by paying attention to the stimulus of interest that represents the desired BCI output. The BCI system analyses the frequency spectrum to find the peak that matches the rate of stimulus of which the user is paying attention. As a result, BCI is able to provide output represented by a particular stimulus (Allison et al. 2008; Zhang et al. 2012b).

A variety of studies has applied SSVEP in different BCI applications. SSVEP-based BCI was used in the study to control a functional electrical stimulator (FES) for knee flexion initialization (Middendorf et al. 2000). SSVEP has also been explored in aeronautical and navigation applications, such as in games where the players use SSVEP-based BCI to help an avatar walk across a tightrope. The user, as the player, needs to stabilize by focusing on one of two oscillating checker boxes (Lalor et al. 2005). Other studies in virtual car navigation used SSVEP-based BCI, in which users move a map and a car in one of four directions by focusing on one of four oscillating checker boxes as the stimuli. In such a design, the resultant performance might be improved by simultaneously focusing on two different stimuli to produce diagonal movement (Martinez, Bakardjian & Cichocki 2007; Müller et al. 2003; Trejo, Rosipal & Matthews 2006; Zhang et al. 2012b).

The advantage of BCI using SSVEP involves a straightforward user task by focusing on the stimulus that represents the desired BCI output. Therefore, it does not require significant training and works for most prospective users. Also, it has long been appreciated for its high information transfer rate (ITR) (Allison et al. 2010a; Allison et al. 2014; Allison et al. 2010b; Wang, Wang & Jung 2010).

SSVEP-based BCI can be used as a dependent BCI, which depends on the user's muscle of gaze direction. This might be an issue to users who lack of reliable gaze control (Gao et al. 2003; Wolpaw et al. 2002). In recent work, SSVEP-based BCI can be applied without the user's gaze fixation being directly at it. By attending to the stimulus, even when the user is not gazing, it increases the amplitude in the fundamental and harmonic spectral peaks associated with a particular stimulus (Allison et al. 2008;

Kelly et al. 2005). This effect can be analysed by using different methods, including spectral analysis, band power and correlation analysis (Kelly et al. 2005).

Gaze dependence still remains an issue for SSVEP so far. Moreover, the SSVEP-based BCI has the same issue as P300-based BCI as an external stimulus may distract the user, such as in the application for wheelchair control, as the user needs to concentrate on the stimulus and the environment at the same time. People with severely compromised vision could be unable to use this type of BCI competently. In this case, BCI based on an auditory steady state evoked potential (ASSEP) can be an option (Kim et al. 2011). In recent years, the hybrid BCI has drawn considerable attention. A Hybrid BCI was able to combine different types of BCIs together with a conventional muscle-based control device to improve the system. One hybrid system combines SSVEP-based BCI with sensorimotor-rhythm (SMR)-based BCI, in which the motor imagery controls the grasp function and SSVEP controls the elbow function (Horki et al. 2011; Millán et al. 2010; Pfurtscheller et al. 2010a; Pfurtscheller et al. 2010b). Another study used SSVEP activity to ascertain whether the user was paying attention to BCI using P300 (Allison et al. 2014). Here, if there is no SSVEP activity, P300-based BCI will simply indicate a ‘no-control’ state. Instead, P300-based BCI provides an output character, thus the SSVEP signal is used to turn the P300 BCI into an asynchronous BCI system (Panicker, Puthusserypady & Sun 2011).

2.4.2.3 Slow Cortical Potential (SCP)

SCP relates with the event-related potential of time-locked and phase-locked to specific sensorimotor events. SCP signals typically consist of negative potential shifts that precede actual or imagined movement, or even other cognitive tasks to represent cortical activation in preparation for activation. Thus, SCP is normally followed by a biphasic wave as movement-related potential (Beuchat et al. 2013; Birbaumer & Cohen 2007; Birbaumer et al. 1999). The Bereitschaft’s potential or readiness potential is a negative SCP that normally starts at 500-100 ms before a self-initiated movement. This has several components that differ in onset time and topographical distribution that

would probably reflect activity in the supplementary motor area and primary motor and sensory cortices. The amplitude and topography of this type of signal are affected by movement type, muscle involved and psychological variables (Shibasaki & Hallett 2006). The SCP uses brain potential changes below 1 Hz that can be up to several seconds and are generated in the upper cortical layers. The negative potential shift represents increased excitability of neurons, and a positive shift is recorded during the consumption of cognitive resources or during testing. Both able-bodied subjects and locked-in patients are able to learn to produce positive and negative SCP shift when they are provided with visual feedback and when potential changes in the desire direction are reinforced.

The thought translation device (TTD) uses SCP-based BCI, which requires training supported through feedback and positive reinforcement (Birbaumer et al. 2003). A visual feedback is normally used with the TTD. With visual feedback modality, for example, users view the course of their SCP as the vertical movement of a cursor on the screen with the movement corresponding to the amplitude of the SCP signal. The aim of the task is to move the cursor toward the polarity, as indicated by a red rectangle at the top or bottom half of the screen. Audio feedback can also be used as an option for patients who have visual difficulty, such as those who are in an advanced stage of ALS. A study was carried out to compare visual and auditory feedback with the result that visual feedback is better in learning SCP signals compared to auditory feedback, although there is a possibility that successful SCP regulation can be obtained with auditory feedback, especially for the users who have the visual version of TTD (Hinterberger et al. 2004; Pham et al. 2005).

One of the disadvantages for SCP-based BCI is that the system requires repeated sessions of training over weeks or months for the user to be able to regulate the SCP signal. Therefore, the system is quite a slow system to be used. Moreover, SCP-based BCI has not allowed for a good multi-dimensional control application. SCP systems have been less successful in getting more than one dimension of control (Kübler et al. 1999). However, the design of SCP-based BCI might be used to enhance other types of

BCIs as a hybrid system, as discussed in the previous section. Especially, it has shown the changes of contingent negative variation (CNV) as the negative SCP associated with anticipation (Gangadhar, Chavarriaga & del R Millán 2009). This could be used, for example, in a wheelchair-control system to use anticipation measurement of the SCP-based BCI to determine whether the user is willing to enter the next room or to continue moving forward. Together, with the sensorimotor rhythm-based BCI, it can improve the performance of the system application (Bai et al. 2011; Friedrich et al. 2009).

2.4.2.4 Sensorimotor Rhythms (SMR)/ Motor Imagery Tasks

BCIs based on sensorimotor rhythm (SMR) have been popular and very much explored in research for many years. It uses motor imagery task associated with the oscillatory EEG signals from the sensorimotor cortex. The SMR uses three major frequency bands: mu/ μ (8-13 Hz), beta/ β (15-30 Hz) and gamma (30-200+ Hz). At the time of the movement onset, it can be found that SMR decreases, known as event-related desynchronization (ERD). When the movement is complete, it can be found that SMR increases, known as event-related synchronization (ERS). ERD and ERS patterns are basically associated with actual or imagined movement, such as foot, leg, hand or tongue movements (Boord et al. 2010; Grosse-Wentrup & Schölkopf 2013; Pfurtscheller et al. 2006a).

The mu rhythm has lower frequency at 8-10 Hz and higher frequency at 10-13Hz as the two distinct ERD patterns. During the lower frequency of mu rhythm, the ERD event occurs during almost all of the motor behaviour, and it is widespread over the entire sensorimotor cortex, so mu ERD appears as non-specific. On the other hand, the higher frequency of mu rhythm ERD event is restricted to task-specific aspects of the performances. In the same way as the mu rhythm, the beta rhythm exhibits ERD in association with somatosensory stimulation and motor behaviours. Furthermore, the beta rhythm also provides ERS following movement (Blankertz et al. 2010; Müller-Putz et al. 2013; Pfurtscheller, Stancak Jr & Neuper 1996). To localize mu rhythm of ERD, which is related to a specific event, it is normally accompanied by simultaneous ERS. It

is also accompanied by a surrounding cortical area, which may reflect a mechanism that emphasizes attention to a specific sensorimotor subsystem by inhibiting other cortical areas that are not directly involved in the specific behaviour.

In terms of the frequency analysis, the standard ERD/ERS calculation composes band pass filtering for each trial, squaring of the samples and averaging of multiple trials. The results are used to define the proportional power decrease (ERD) and power increase (ERS) relative to a specific reference interval of several seconds before the onset of the event. In this case, it is the real or imagery movement. As a result, a time-frequency map can be produced (Grimm et al. 2002). For the spatial analysis of SMR, spatial filtering methods can emphasise the localised signal features over a specific cortical region and can improve recognition (McFarland et al. 1997). Since it is an EEG-based system, SMR can also be interfered by artefacts, including: electromagnetic signal of power lines, electromyographic activity (EMG) from neck, cranial, and facial muscles and electrooculographic activity (EOG) from eye movements and eye blinks. For the EMG activity, it produces a broadband signal over wide areas of the scalp. Thus, it can overlap with the SMR signal in frequency and location (Goncharova et al. 2003). The EOG artefacts are mostly at lower frequency (1-4 Hz), lower than the SMR signal (μ and beta signals). As a result, it is usually not a major issue for BCI using the SMR. A variety of methods for removing the EOG artefacts have been discussed further (Croft et al. 2005; Fatourehchi et al. 2007).

One of the examples of SMR-based BCI is in the application of cursor movement control, during which the users need to move a cursor to hit a target located on the top or bottom edge of the video screen. They do this by using the SMR signal, especially the μ rhythm (8-13 Hz), in the sensorimotor area as a one dimension cursor control allows the selection of more than two targets (Pfurtscheller et al. 2006b; Wolpaw et al. 1991). The SMR signal moves the cursor in two dimensions at a comparable speed and accuracy (Cincotti et al. 2008; Wolpaw 2010; Wolpaw & McFarland 2004). Also, a third dimension has been explored recently (Royer et al. 2010). SMR-based BCI also has been applied in the communication application such as a spelling system to allow

users or people with disabilities to communicate by selecting a letter, or other items forming words and sentences (Friedrich et al. 2009; Kubler et al. 2005; Millan & Mourino 2003; Muller & Blankertz 2006; Neuper et al. 2003; Wolpaw et al. 2003). Other examples of SMR-based BCI were used in the neuroprosthesis application to restore hand grasp in a paralysed individual based on combination functional electrical stimulation (FES) with SMR signal. As a result, using the BCI-FES system, the disabled user was able to perform foot motor imagery for hand grasping (Müller-Putz et al. 2005; Pfurtscheller et al. 2003a). Further application with tetraplegia in virtual reality also has been applied with SMR-based BCI to provide simulation of wheelchair control (Leeb et al. 2007).

It is possible in an SMR-based system where people with severe disabilities are the primary user of assistive BCI device that a pathological process of severe motor disability may also impair the SMR signal. Therefore, in this particular case, they would not be able to use the SMR-based BCI effectively. Thus, an alternative to motor imagery would be beneficial for those who might have difficulty with motor imagery (Conson et al. 2008).

2.4.2.5 Mental Tasks (Non-motor Imagery Mental Tasks)

Although SMR is a popular method and many reports can be found in research using motor imagery-based BCI, there are still people who are unable to use this, as in the BCI illiteracy phenomenon (Allison & Neuper 2010; Blankertz et al. 2010). There might be a case of inter subject variability issue to use the SMR-based BCI, which resulted in high variability of accuracy for each subject. Other research also reported selective motor imagery task defects in severely disabled patients (Conson et al. 2008). In a scenario for people who have certain area impairment, such as stroke patients, it would be problematic to use the SMR-based BCI as it is mostly a concentration on the motor cortex area. The choice of different mental tasks, other than motor imagery task, would be used here and would be a more suitable strategy for this particular case (Curran & Stokes 2003). Furthermore, individuals who amputees or have been

paralysed for a number of years may be unable to perform motor imagery mental tasks effectively (Birch, Bozorgzadeh & Mason 2002; Curran et al. 2004).

Several other mental tasks also have been explored in the research as the spontaneous based mental strategy. These tasks are mental cognitive tasks and they are not motor imagery task. Therefore, it is referred to non-motor imagery mental task-based BCI or simply as a mental task-based BCI. The tasks that have been explored include five tasks (one baseline task and four mental tasks): relaxation (baseline), mental multiplication, mental geometric figure rotation, mental letter composing and mental visual counting, with a total of 6 EEG electrodes being placed at C3, C4, P3, P4, O1 and O2. In the baseline task, there was no mental task involved, and the participant was asked to relax and try to think nothing; in mental multiplication, the participant was asked to mentally perform multiplication calculation; in mental geometric figure rotation, the participant was asked to visualize a 3D-object being rotated around an axis; in mental letter composing, the participant was asked to mentally compose a letter in mind without vocalizing through their mouth; in mental visual counting, participants mentally counted numbers by visualising the number appearing and disappearing on a blackboard in their mind (Anderson & Bratman 2008; Anderson, Devulapalli & Stolz 1995a; Anderson et al. 2006; Craig & Nguyen 2007; Faradji, Ward & Birch 2009; Friedrich, Scherer & Neuper 2012; Keirn & Aunon 1990; Lei et al. 2011; Nakayama & Inagaki 2006; Palaniappan 2005; Palaniappan et al. 2002; Teli & Anderson 2009).

With more electrodes attached, other non-motor imagery tasks are also explored further, including mental audio imagery, where the participants were asked to imagine a familiar tune in their head without moving their mouth; mental spatial navigation, where participants were asked to imagine moving around and scanning the surroundings in a familiar environment. The mental spatial navigation task is not using motor imagery, because the imagination is involved in examining the surroundings, rather than the foot walking as in motor imagery mental task. Mental imagining a familiar face with participants being asked to imagine a face of a familiar person has been used (Başar et al. 2007; Cabrera & Dremstrup 2008; Curran et al. 2004; de Kruif, Schaefer & Desain

2007; Friedrich, Scherer & Neuper 2012; Özgören, Başar-Eroğlu & Başar 2005). For a practical system, some researchers tried to study with different combinations of two and three EEG channels (Craig, Nguyen & Burchey 2006; Faradji, Ward & Birch 2009). With fewer electrodes used, the BCI system will be much better for a practical system as it will reduce the preparation time and it will be a low cost system. Moreover some channels can also provide redundant features and the same performance can be done with the reduced channels.

2.5 Review on Computational Intelligence for EEG-based BCI

Computational intelligence is the fundamental component for operation of BCI, as previously shown in Fig 2.7. After the electrical brain signal is measured, this is continued with the BCI computational intelligence, including a feature extraction block to extract the important feature of the raw EEG signal and features translation or classification block to classify the feature vector into BCI output control (Al-Fahoum & Al-Fraihat 2014; Cheolsoo et al. 2013; Chun-Hsiang et al. 2014; He et al. 2013; Llera, Gómez & Kappen 2014; Millán 2013; Mühl et al. 2014; Nicolas-Alonso & Gomez-Gil 2012; Ortiz-Rosario & Adeli 2013; Sanei et al. 2013).

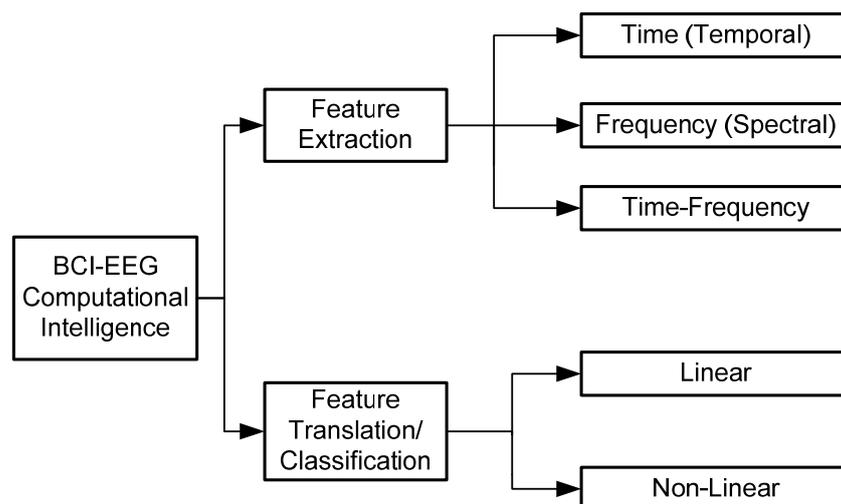


Figure 2.7: Existing BCI-EEG computational intelligence

Prior features extraction and an additional signal pre-processing block can be added to reduce the noise and to improve the signal-to-noise ratio (SNR). A digital band pass filter is commonly used for signal pre-processing to eliminate frequencies that lie outside the frequency range of the brain activity relevant to the BCI application. For example, in the SMR-based BCI band pass filter, 8-30 Hz can be applied to isolate the important ERD/ERS feature, which is located in the alpha and beta bands. In P300-based BCI, some consideration needs to be carefully taken as the features evoked potential can be located in the low frequency band, and the time domain feature analysis is preferable. Furthermore, for the feature extraction, it can be viewed from the time, frequency and time-frequency domain analysis. For feature translation /classification block, a linear and non-linear algorithm can be used. The surveys of the BCI computational intelligence (feature extraction and classification algorithms) are discussed (Bashashati et al. 2007; Georgieva et al. 2014; Krusienski et al. 2011; Lotte et al. 2007; Makeig et al. 2012; McFarland et al. 2006; Muller, Anderson & Birch 2003) .

2.5.1 Features Extraction

The features extraction block in BCI is the operation to transform the raw EEG signal into a feature vector in order to obtain the best classification performance for the BCI. If the extracted features do not describe well the neuro-physiological signal, there would be an issue in identifying the class for this feature. As a result, a low rate of classification accuracy may happen. Therefore, a suitable feature extraction algorithm is needed to extract good features from the raw EEG signal to maximize the performance classification accuracy of the BCI.

Numerous features extraction methods have been explored for BCI. In Fig 2.7, the feature extraction algorithm can be divided into three main groups including time (temporal) domain analysis, frequency domain analysis and combination time-frequency domain analysis. Different methods of BCI-EEG, as described in the previous section, may also dictate the choice of the feature extraction algorithm. For example, the SMR-based BCI uses the sensorimotor rhythms, which are amplitude

modulated at specific frequencies (alpha and beta bands) over the sensorimotor cortex; therefore the frequency domain analysis algorithm can be used. In other cases, the particular neuro-physiological feature is unknown, so both time and frequency domain analysis can be used.

2.5.1.1 Time (Temporal) Methods

Time (temporal) method is used to describe the BCI-EEG as a neurophysiological signal with a precise and specific time signature. The method based on signal amplitude is the simplest temporal information. Further enhancement can be done by applying the peak-picking and integration methods to determine the feature of BCI. The peak-picking method uses the feature, which determines the minimum and maximum value of the signal samples in a particular time block. Alternatively, the integration method can be applied, in which the signal can be averaged and integrated over all or part of the time block to provide the feature. P300 is an example of the straightforward method of features extraction (Farwell & Donchin 1988; Hoffmann et al. 2005; Kaper et al. 2004).

Alternatively, the autoregressive (AR) method can be used in the BCI-EEG (Anderson, Stolz & Shamsunder 1998; Burke et al. 2005; Pfurtscheller et al. 1998). The autoregressive method with a signal $X(t)$, measured at time t , can be modelled as a weighted sum of the value of this signal at previous time steps. The primary disadvantage of the feature extractor using AR modelling is that it is highly dependent on the selected model order. An insufficient order of model will affect low classification accuracy. Note that frequency information can be possible if derived from the AR coefficients (McFarland & Wolpaw 2005). AR model order selection for BCI applications have been discussed in the research (Georgieva et al. 2014; McFarland & Wolpaw 2008).

2.5.1.2 Frequency (Spectral) Methods

EEG signals are manifested by continuous amplitude and frequency modulated oscillations known as EEG rhythms (delta, theta, alpha, beta and gamma). As a result, a frequency domain feature extraction method can be used.

One of the most basic methods for tracking the amplitude modulation at a particular frequency is using the band power feature extraction method. It is done to isolate the frequency of interest by filtering the signal with a band-pass filter, which produces a large sinusoidal signal. This is continued by producing purely positive values. The signal is rectified by squaring the signal or by computing the absolute value of the signal. Finally, the adjacent peaks are smoothed together using the low-pass filtering. Fast Fourier transform (FFT) is a method to implement the discrete Fourier transform in an efficient implementation, which is the discrete-time equivalent of the continuous Fourier transform.

The FFT represents the frequency spectrum of the digital EEG signal with a frequency resolution of the sampling rate. FFT is often used as the baseline method for comparison with another spectral analysis method because of its simplicity and effectiveness. A power spectral density (PSD) or power spectrum can also be used, which informs the distribution of the power of the signal between the different frequencies. A simple calculation of the power spectrum can be obtained by squaring the FFT magnitude with each bin of the FFT magnitude spectrum which tracks the sinusoidal amplitude of the signal at the certain frequency. As an efficient implementation, features extraction based on FFT has been used widely in the EEG-based BCI (Barreto, Frota & de Medeiros 2004; Besserve, Garnero & Martinerie 2007; Craig & Nguyen 2007; Craig, Nguyen & Burchey 2006; Keirn & Aunon 1990; Kelly et al. 2005; Lalor et al. 2005; Millan & Mouriño 2003; Millán et al. 2002; Park et al. 2013; Pregenzer & Pfurtscheller 1999; Wolpaw & McFarland 1994).

2.5.1.3 Time-Frequency Methods

Instead of using features extraction on the time domain or frequency domain alone, the combination of time-frequency features extractor method has also been used to design BCI. Wavelet analysis is an example of a time-frequency feature extraction method. It is found that results based on temporal and spectral resolution are highly dependent on the selected segment length, model order and other parameter. This can be an issue when the signal has a wide range of relevant frequency components with each possessing temporally distinct amplitude modulation characteristics. For example, in the sample block length, the amplitude of a particular high frequency section has the potential to fluctuate over each cycle within the sample block. The wavelet method tries to solve this issue by introducing the time-frequency representation of the signal.

However, there is also a trade-off using the method in that it is impossible to determine the instantaneous frequency and time of occurrence event precisely. This means in a longer time windows it will produce spectral estimation of higher frequency resolution and in a shorter window will have lower frequency resolution. In wavelet analysis, a mother wavelet as a time-limited pulse shape is used to construct a template for each temporal band-pass filter of the filter bank. There is a variety of mother wavelets, and each has specific time-frequency characteristics and mathematical properties. By using wavelet, it is possible to analyse at different scales simultaneously, whilst the resolution depends on the scale. Just as the FFT provides an efficient computation of the Fourier Transform for the digital signal, the discrete wavelet transform (DWT) also provides an efficient computation of the wavelet transform using specific scale and factor, which minimize redundancy in the time-frequency analysis. As a result, this provides a wavelet which is an interesting tool for analysing BCI-EEG signal (Bostanov 2004; Graimann et al. 2004; Hammon & de Sa 2007; Krusienski et al. 2011; Lemm, Schafer & Curio 2004; Mason & Birch 2000; Qin & He 2005; Samar et al. 1999; Varsta, Heikkonen & Mourino 2000; Yong, Hurley & Silvestre 2005).

2.5.2 Feature Translation or Classification

In BCI, features translation or classification algorithms convert signal features taken from features extraction into output for device control commands. The output commands may be continuous or discrete and the success of a BCI is determined by the appropriateness of the selected signal features, and by how effectively it translates this control into device commands. The core of a classification algorithm is a mathematical model and procedure, which is comprised of a mathematical equation and/or mapping mechanism, such as a lookup table. The model uses the input of a feature vector and processes the feature vector into output with a set of commands to be recognized in the application device.

The model parameters are selected by using a training set, in which each unit of training data consists of a feature vector and its intended output. The parameters are repeatedly adjusted until the model translates the feature vector into output commands as accurately as possible through an iterative procedure called supervised learning. For the result performance, the accuracy of the model is evaluated with a fitness function as an objective function. For example, the mean square error (MSE) between a model of the output and the correct output can be used as the fitness function. The smaller the error of the MSE, the more accurate the model will be. It is important that it has the ability to generalise by using an independent set of observation data called a testing set. However, some classification models and training procedures are prone to overfitting or overtraining, in which the parameterised model is tuned so precisely to the training data that subtle differences between the training set and the test set prevent it from modelling the testing set accurately. Early stopping and cross-validation methods can be applied for the generalisation issue.

Basically, the classification algorithm can be divided into two groups: linear classifier and non-linear classifier (Georgieva et al. 2014; Krusienski et al. 2011). Linear classifier is simpler than non-linear classifier, as it uses a linear mathematical model only for the classifier training, but from the signal processing perspective. As

little is known about the physiological factors generating the EEG, a non-linear classifier could be used.

2.5.2.1 Linear Classification Methods

Linear discriminant analysis (LDA) is the most popular linear method that has been used for the BCI-EEG classification method. It is a machine learning method used to find the linear combination of features, which best separates two or more classes of the outputs by using the hyperplanes. For example, in a two-class classification problem, the class of a feature vector depends on which side of the hyperplane the vector is. LDA assumes a normal distribution of the data with equal covariance matrices for both classes. The separating hyperplane is obtained by seeking the projection that maximizes the distance between the two classes' means and minimizes the interclass variance. For solving an N-class classification problem, several hyperplanes need to be used. The LDA method has a low computational requirement, which makes it suitable even for an online BCI system. LDA has been applied in a great number of BCI systems (Blankertz, Curio & Müller 2002; Bostanov 2004; Garrett et al. 2003; Long et al. 2012; Müller et al. 2004; Onishi & Natsume 2013; Pfurtscheller & Lopes da Silva 1999; Scherer et al. 2004).

Support vector machine (SVM) is another linear classifier that also uses a discriminant hyperplane to identify classes. However, in SVM, the selected hyperplane is the one that maximized the margin which is known to increase the generalisation capability. On the training set, SVM uses a regularisation parameter C that enables accommodation to outliers and allows error on the training set. The SVM minimise an objective function using a series of iteration that includes two components. The first component consists of the Euclidian distance between the observations that are on the correct side of the margin and do not contribute to the component. The second component is the Euclidian distance between margins. SVM must select observations from the data set that simultaneously minimise these two components of the objective function. Linear SVM have been used to tackle the BCI problem (Blankertz, Curio &

Muller 2002; De Massari et al. 2013; Garrett et al. 2003; Rakotomamonjy & Guigue 2008; Rakotomamonjy et al. 2005).

2.5.2.2 Non-linear Classification Methods

Linear methods for BCI classification may not always be an effective method (Muller, Anderson & Birch 2003). Therefore, a non-linear method of classification can be used as an option.

It is possible for SVM to create a non-linear decision boundary with only a low increase of the classifier's complexity. The SVM algorithm makes extensive use of projection of the data into higher-dimensional feature space, a method of linearizing a non-linear problem by using the kernel methods. It consists in implicitly mapping the data to another higher dimensionality space using the kernel function. The popular SVM kernel for BCI classifier is the Gaussian or radial basis function (RBF). SVM also provide fairly good result for BCI classification as SVM is known to have good generalisation properties (Garcia, Ebrahimi & Vesin 2003; Garrett et al. 2003; Georgieva et al. 2014; Kaper et al. 2004; Schlogl et al. 2005).

Artificial neural network (ANN) is a popular non-linear classifier algorithm used in BCI research. ANNs are simplified models of biological neural networks with the expectations they will allow the powerful decision-making capabilities of biological networks to be applied to classification problems. For BCI purposes, the network inputs consist of the feature vectors and its output consists of the commands for BCI application. Generally, ANNs have some important properties, which include learning, generalisation, non-linearity and fault tolerance.

The learning properties of ANN are its ability to learn from the examples and produce a certain output when presented with a certain input. This learning process involves modification of the ANN parameters and the connection weights to make the overall behaviour correspond to a desired behaviour defined by a set of training data sets. The generalisation properties of the ANN enable it to produce outputs even for inputs not encountered during training. For the non-linearity properties, ANN defines a

mapping from an input space to an output space. This mapping can be described as a vector-valued function that transforms the input vector to the output vector where both can be in the form of any dimension. The mapping function itself is a combination of mappings performed in parallel by simpler units, the neurons. The information processing in each neuron is non-linear, resulting in non-linearity of mapping. The fault tolerance properties of the ability of ANN try to mimic the robust structure from the brain network.

There are some ANN known architectures. The most widely used ANN for BCI is the multi-layer perceptron (MLP), which is composed of several layers of neurons: input layer, hidden layers and output layer. For BCI classification, MLP of ANN has been applied widely to binary and multi-class BCI problems (Anderson & Sijercic 1996; Asvestas et al. 2014; Balakrishnan & Puthusserypady 2005; Chiappa et al. 2004; Craig & Nguyen 2007; Craig, Nguyen & Burchey 2006; Georgieva et al. 2014; Haselsteiner & Pfurtscheller 2000; Huan & Palaniappan 2005; Palaniappan 2005; Sussillo et al. 2012). Other types of ANN are also used in BCI, such as the Gaussian classifier of neural network (Millan et al. 2004), learning vector quantization (LVQ) neural network (Pfurtscheller, Flotzinger & Kalcher 1993), fuzzy ARTMAP neural network (Palaniappan et al. 2002), dynamic neural network (Barreto, Taberner & Vicente 1996), radial basis function (RBF) neural network (Hoya et al. 2003), Bayesian logistic regression neural network (BLRNN) (Penny et al. 2000), adaptive logic network (ALN) (Kostov & Polak 2000), and probability estimating guarded neural classifier (PeGNC) (Felzer & Freisieben 2003).

2.6 Discussion

Comparison of BCI technologies from the electrophysiological signal point of view is illustrated in Fig. 2.8, which includes intra-cortical recording (microelectrodes), ECoG and EEG. Both microelectrodes and ECoG are invasive BCI methods, therefore requiring surgery. The microelectrodes method is the most invasive method among BCI technologies to measure the cortex. The ECoG technology is less invasive compared to

the microelectrodes method with the sensor placed on the dura area, the top of the cortex. The EEG is the only non-invasive method among these electrophysiological BCI technologies.

General comparison of BCI technologies in terms of the spatial and temporal resolution is shown in Fig 2.9 and Table 2.1. In the invasive method, microelectrodes and ECoG have the better spatial and temporal resolutions compared to non-invasive methods (EEG, MEG, fNIRS, fMRI, PET and SPECT). But the invasive methods have the disadvantages of risk of infection, scarring from post-surgery and possible unknown long term effects. Moreover, the implanted electrodes might have a limited lifetime, so there might be a need for regular surgical operations to replace the electrodes which would raise numerous ethical issues. It is therefore the non-invasive method which is chosen for this research. For non-invasive methods, compared to EEG, MEG provides the better spatial and temporal resolutions, although it is expensive and needs a huge space for the complex instrumentation. The fMRI, SPECT and PET as the imaging systems also have the same disadvantages of high cost and require a large space for the hardware. Therefore, they are not the chosen method in this research. The hardware fNIRS system can be portable, but it is still in its infancy and has lower temporal resolution compared to the EEG. For the general view, EEG technology is widely used because it is fast, inexpensive and less cumbersome than non-invasive technologies. As a result of the advantageous comparison of EEG with other technologies, this thesis focuses on EEG-based BCI.

The comparison among available BCI-EEG methods is shown in Table 2.2. The methods include: P300-, SSVEP-, SCP-, SMR- and mental task-based BCI. The P300-based BCI is the BCI method to use the EEG feature as a positive shift about 300 ms after the external stimulus occurred.

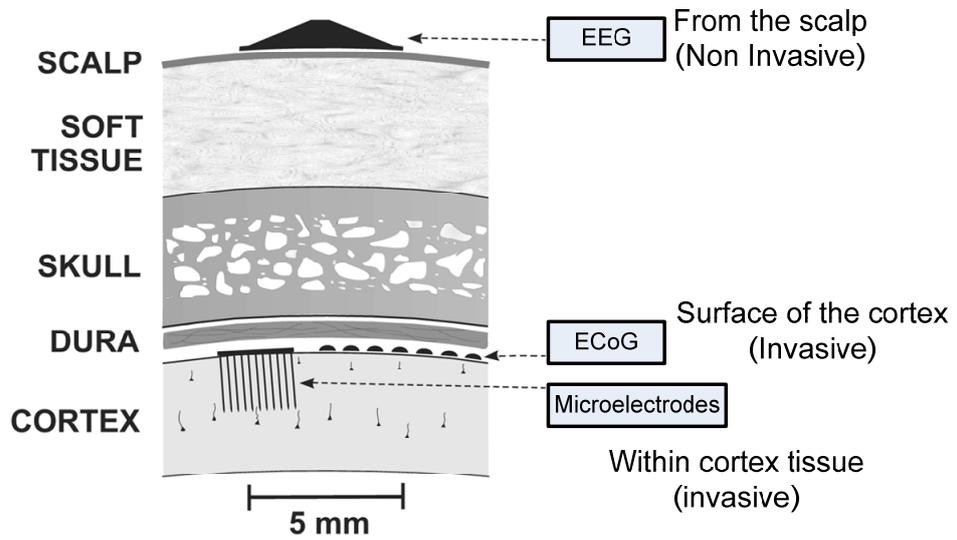


Figure 2.8: BCI technologies electrophysiological view

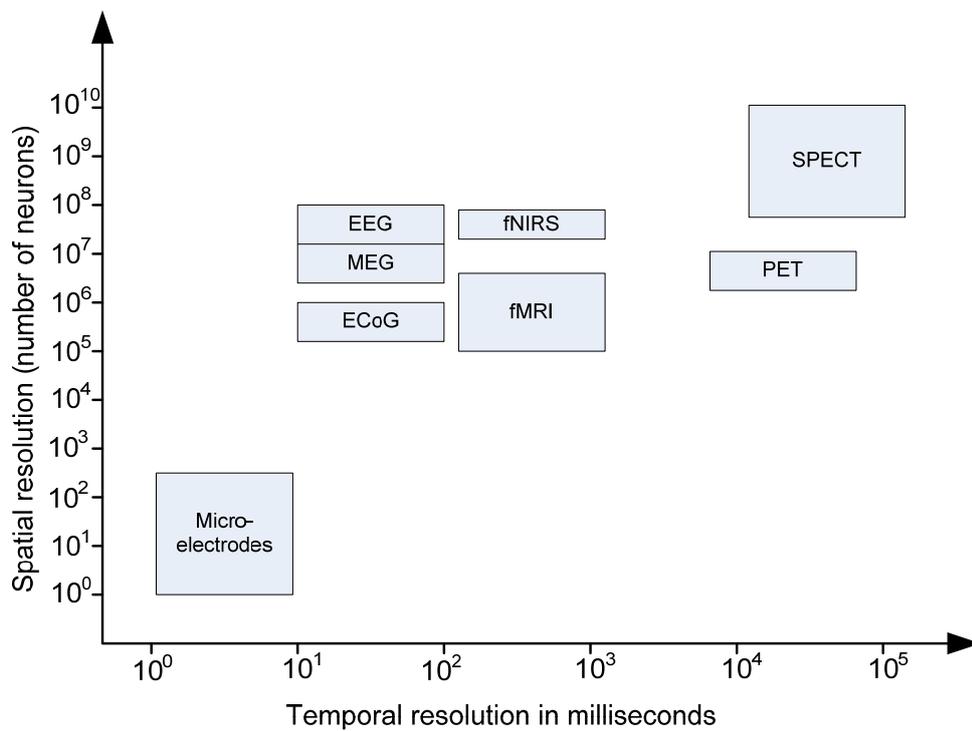


Figure 2.9: Resolutions of BCI technologies

The SSVEP-based BCI uses features in a specific frequency in response to a modulated visual stimulus. The P300 and SSVEP method both have the advantage of minimal user training for BCI to be used, but as they are a selective method group, they need external stimuli for operation. This might cause discomfort to the user, such as for wheelchair control, as the user needs to control the wheelchair, which needs environment concentration and external stimuli concentration at the same time.

For the spontaneous signal method, the SCP-based BCI uses negative or positive shifts in the EEG signal. The disadvantage of SCP is that it requires long repeated times of training over weeks or months for the user to be able to regulate the SCP signal; therefore, the system is quite a slow system to be used. The SMR-based BCI is a spontaneous mental signal based on the event-related desynchronization/synchronization (ERD/ERS) which is produced by imagining hand, foot or tongue movements with corresponding EEG electrodes positioned primarily over the motor cortex regions. It is a popular method in the current state of research. There are some disadvantages for SMR-based BCI, including; inability by some people who are unable to use this, as in BCI illiteracy phenomenon, selective motor imagery task defects in severely disabled patients were also reported, individuals who have been paralysed or are amputees for a number years may be unable to use SMR-based BCI and difficulties experienced by individuals with an impairment of a certain motor cortex area (stroke patients). With the disadvantages of SMR-based BCI, alternatively, mental task-based BCI can be used. Mental task-based BCI uses mental non-motor imagery tasks, and the user needs to be familiar with these to imagine the particular mental task to operate the BCI. This thesis further explores the use of mental task-based BCI as an option in the BCI methods.

For the instrumentation point of view, portability, convenience and cost effectiveness are important aspects for a practical BCI system. An embedded system that uses a low power microcontroller system and a wireless interface would maximise aspects of creating a portable, convenient and low cost system. Moreover a lower number of EEG channels used would also improve portability and convenience aspects

for practical BCI. This thesis explores the development of the instrumentation of wireless EEG for mental task-based BCI using the low powered microcontroller-based system. Employing a lower number of EEG channels is also discussed further.

From a computational intelligence point of view, the features extraction algorithm can be divided into three main groups, including time (temporal) domain analysis, frequency domain analysis and combination time-frequency domain analysis. Different BCI-EEG methods use different feature extraction algorithms. For example, in P300, time domain analysis is preferable, as the feature locates about 300 ms after the stimuli occurred. In SMR-based BCI, it uses the sensorimotor rhythms, which are amplitude modulated at specific frequencies (alpha and beta bands) over the sensorimotor cortex. Therefore, the frequency domain analysis algorithm can be used. In other cases, the particular neuro-physiological feature is unknown; time-frequency domain analysis, such wavelet analysis and other methods can be used.

This thesis also investigates another time-frequency analysis, the Hilbert-Huang Transform used as the features extractor for mental task-based BCI application. The translation/ classification algorithms for BCI basically are divided into linear and non-linear algorithms. The EEG signal is a non-linear and non-stationary signal. This thesis discusses the classifier based on the Artificial Neural Network (ANN) as a non-linear classification algorithm, together with improved optimization techniques. Furthermore, most BCI experiments have been conducted with able-bodied subjects rather than with severely disabled subjects for whom this new technology is primarily intended. One of the chapters of this thesis includes the experiment on able-bodied subjects and patients with tetraplegia.

Table 2.1: Overview BCI technologies

BCI Techniques	Measurement property	Advantages	Disadvantages
Intra-cortical recording/ Microelectrodes (Donoghue 2002; Hochberg et al. 2012)	Electrical within cortex	- Highest temporal resolution - Good spatial resolution - Equipment inexpensive	- Most invasive - Require surgery - Procedurally expensive
ECoG (Freeman et al. 2000; Henle et al. 2013)	Electrical on surface of cortex	- High temporal resolution - Good spatial resolution - Equipment inexpensive	- Less invasive - Require surgery - Procedurally expensive
EEG (Farwell & Donchin 1988 ; McCane et al. 2014)	Electrical of the scalp	- Non-invasive - Good temporal resolution - Portable and wearable - Low cost	- Low spatial resolution - Artefacts interference
MEG (Sato 1990; (Burmistrov et al. 2013)	Magnetic	- Non-invasive - Good temporal resolution	- Bulky and expensive - Require magnetic isolation room
fNIRS (Coyle et al. 2004; Zimmermann et al. 2013)	Blood flow\ Oxygenated & deoxygenated haemoglobin	Non-invasive Good spatial resolution Portable and wearable Low cost	- Low temporal resolution - Still new method
fMRI (Ogawa et al. 1990; Sorger et al. 2012)	Blood flow\ Oxygenated & deoxygenated haemoglobin	Non-invasive High spatial resolution	- Bulky and expensive - Low temporal resolution
PET (Decety et al. 1994; Juweid & Hoekstra 2011)	Blood flow/ Tracking of radioactive tracer in blood stream	Non-invasive Good spatial resolution	- Bulky and expensive - Need to inject radioactive substances - Low temporal resolution
SPECT (Van Heertum, Tikofsky & Ichise 2000; Holly et al. 2010)	Blood flow/ Tracking of radioactive tracer in blood stream	Non-invasive	- Bulky and expensive - Need to inject radioactive substances - Lower spatial and temporal resolution than PET

Table 2.2: Comparison BCI-EEG methods

BCI-EEG Methods	Advantages	Disadvantages
P300 (Farwell & Donchin 1988; McCane et al. 2014)	- Feature as a positive shift in the EEG about 300ms after the stimulus - No requirement of significant training	Need external stimulus
SSVEP (Steady state visual evoked potential) (Middendorf et al. 2000; Allison et al. 2014)	- Feature in specific frequency in response to a visual stimulus modulated - No requirement of significant training.	Need external stimulus
SCP (Slow cortical potential) (Birbaumer et al. 1999; Beuchat et al. 2013)	Negative and positive shift	Require a long time for training
SMR (Sensorimotor rhythm) (Pfurtscheller et al. 1996; Müller-Putz et al. 2013)	- Motor imagery task - Without external stimulus - Popular method in BCI research	-There still is BCI illiteracy phenomenon found - Selective motor imagery task defects in severely disabled patients were also reported -Individuals who have been paralysed or are amputees for number years or with an impairment of certain motor cortex area (stroke patient) may be unable to use SMR
Mental task (Keirn & Aunon 1990; Friedrich, Scherer & Neuper 2012)	- Non-motor imagery tasks - Without external stimulus - Alternative option of disadvantages of SMR	- User needs to be familiar with the mental task

2.7 Conclusion

Although the invasive methods have the better spatial and temporal resolutions compared to non-invasive methods, the invasive methods have disadvantages, including risk of infection, scarring from post-surgery and possible unknown long term effects.

Moreover, the implanted electrodes might have a limited lifetime; as a result, there might be a need for regular surgical operations to replace the electrodes, which would raise numerous ethical issues. The non-invasive method is the chosen method of this research if compared to the invasive method. For the non-invasive method, EEG technology is widely used because it is fast, inexpensive and less cumbersome than other non-invasive technologies. BCI using selective attention strategy such as P300 and SSVEP relies on external stimuli, which might be uncomfortable for severely disabled individuals who need to focus on external stimuli and the environment simultaneously. This is not the case for BCI that rely on spontaneous mental signals (SMR and mental tasks) initiated by the users themselves. There have been many reports in the research using SMR-based BCI; however, there are still some people who are unable to use this. Thus, non-motor imagery mental tasks or simply BCI using mental tasks is the chosen strategy. In terms of the instrumentation portability, convenience and cost effectiveness are important aspects in BCI.

A wireless EEG and embedded system would maximize the aspect of creating a portable, convenient and low cost system. Moreover, a lower number of EEG channels used would also improve portability and convenience aspects for practical BCI. The feature extractor using time-frequency analysis can be used instead of using the conventional FFT as the frequency analysis. As the EEG signal is a non-linear and non-stationary signal, a non linear classifier such as ANN would be suitable. This thesis discusses the classifier based on ANN together with improved and advanced techniques. Furthermore, most BCI experiments have been conducted on able-bodied subjects only. It is an important experiment to include severely disabled subjects for whom this new technology is primarily intended. Therefore, in the next chapter, this thesis focuses on discussions including EEG as a non-invasive system, BCI using mental tasks paradigm, portable and low cost instrumentation, time-frequency feature extractor, improved and advanced classifiers, and BCI experiments to include both able-bodied groups and severely disabled groups.

Chapter 3

Wireless Embedded EEG System for Mental Task Classification

3.1 Introduction

An assistive technology (AT) is a device that bridges the gap between the physical (cognitive) abilities of individuals with disabilities and a function that they want to perform. For example, a wheelchair provides mobility and improves function for people who are otherwise unable to move around by themselves. In the application of a wheelchair control for people with disabilities, various hands-free technologies have been used to replace the joystick, including sip-and-puff, chin controller (Guo et al. 2002), muscle-based system (Zhang et al. 2012a), voice recognition (Simpson & Levine 2002), tongue controller (Huo & Ghovanloo 2010), and head movement systems (Nguyen, King & Knight 2004; Taylor & Nguyen 2003). These technologies have their own benefits and drawbacks. In practical situations, the user may feel uncomfortable with the operation of a sip-and-puff or a chin or tongue controller. Noisy environments can be problematic for voice recognition systems. Muscle and head movement

technologies are targeted for disabled individuals who are still able to provide the relevant body movement for real-time control. Certain neurological conditions, such as amyotrophic lateral sclerosis (ALS), spinal cord injury and brain stem stroke, may lead to severe motor paralysis. These conditions could cause individuals to suffer mobility restrictions, or even worse, to suffer locked-in syndrome.

A brain computer interface (BCI) system could be used as an alternative control and communication method for such people by converting their brain activities into commands to be used to improve mobility purpose, for example, to control a wheelchair as an AT. A BCI using electroencephalography (EEG) to measure brain activities could provide severely disabled people with alternative means of control and communication. A BCI can be integrated with an application so that together they constitute a standalone BCI-AT system. Mobility is a key factor in independence, so it is a function that is highly important and valued by many potential BCI users.

In a survey of people with a severe disability, ALS, interest in BCI-based operation was high for all kinds of AT applications, and for powered wheelchair operation, which tended to generate more interest than other non-communications tasks (Huggins, Wren & Gruis 2011). It is because the mobility covers several categories of abilities, which includes the ability to adjust body position while sitting or lying down, the ability to move within the living quarters and the ability to leave the living quarters and travel in the community. The AT devices for mobility, such as the motorized wheelchair, can be operated by a joystick, switch, or other controls. The same control can be operated by BCI, as well as an alternative hands-free technology.



Figure 3.1: Example BCI-based products on the current market

On the current state in the market, there are still no available products yet released targeted especially for BCI in the application of wheelchair control for mobility of people with disabilities. Most of the BCI products on the market, as shown in Fig. 3.1, are used for gaming applications, such as the NIA game controller (BCInet 2012), Mindset (Neurosky 2010) and Emotiv Epoc (Emotiv 2010). So far, most of the operation of the system is based on a simple form that is using the mental concentration or relaxations with the electrodes placed on the scalp. Another BCI product, Intendix (g.tech 2011), is an EEG-based spelling system, in which the user needs to focus attention on the external stimulus on the screen. It might be distracting to the user for the application of wheelchair control. As a result, BCI as the cutting edge technology would need to be addressed further, especially to using other BCI-EEG methods in the application for individuals with severe disabilities to control a wheelchair.

Furthermore, the electroencephalography (EEG)-based BCI is non-invasive, safe and relatively inexpensive compared to invasive methods or other non-invasive methods. EEG measurement is recorded with electrodes placed on the scalp and is the summation of the electrical activity of millions of synapses, neurons and axons in the

cortex of the brain to produce scalp voltages. As discussed in the previous chapter, this thesis concentrates on using mental tasks-based BCI as an option among other BCI-EEG methods as illustrated in chapter 2, Fig.2.6, which includes P300-, SSVEP-, SCP-, and SMR-based BCIs. P300 and SSVEP are examples of the selective attention method in which the user needs to concentrate on external stimuli that flash in succession (P300) or continuously at a certain frequency (SSVEP). The use of external stimuli can be cumbersome in wheelchair control, as the user needs to pay attention to the external stimuli and control the wheelchair at the same time. Therefore, the spontaneous mental signal methods are preferred. SMR-based BCI is one of the examples of spontaneous mental signal that focuses on the motor imagery area, such as by imagining the hand, foot and tongue movement. The electrode placement for SMR-based BCI is concentrated on the motor cortex or the motor homunculus; a figurative representation of the body map encoded in the primary cortex.

Some issues still can be found for SMR-based BCI, including: there are still people who are unable to use this method (BCI illiteracy phenomenon) (Allison & Neuper 2010; Blankertz et al. 2010); selective motor imagery task defects in severely disabled patients were also reported (Conson et al. 2008); individuals who have been paralysed or are amputees for numerous years may be unable to use SMR-based BCI (Birch, Bozorgzadeh & Mason 2002; Curran et al. 2004); it also can happen to an individual with an impairment of certain motor cortex, such as in a stroke patient. As a result, mental non-motor imagery tasks, or simply mental task-based BCI, can be used as an option.

Hardware and software are critically important in implementing mental tasks-based BCI, and in ensuring that they function effectively in real-time. BCI hardware covers the physical aspect, during which brain signals are acquired/ measured, digitised, stored, and analysed. The hardware generally includes the sensors that detect the brain activity in this case, an EEG amplifier, analog to digital converter (ADC), computer to process the digitised signals, computational intelligence (signal pre-processing, features extraction and classification), and the output conversion that achieve the user's intent.

BCI typically makes use of a personal computer (PC) for the sophisticated signal processing and computational intelligence required for the classifier.

However, with the rapid development of the embedded system (microcontroller-based system) and wireless technology, a microcontroller-based BCI system with a separate head-mounted battery-operated wireless EEG sensor provides maximum portability, convenience and cost effectiveness.

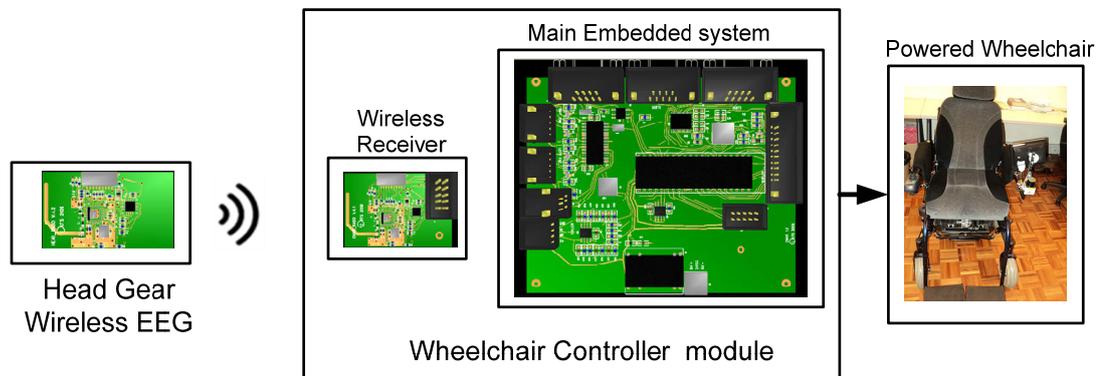


Figure 3.2: Instrumentation of BCI for wheelchair control

The general block diagram of microcontroller-based BCI instrumentation for wheelchair control is illustrated in Fig 3.2. The head gear has wireless EEG, which comprises analog and digital components. A combined microcontroller and wireless transceiver in the digital section transfers data wirelessly. The same wireless microcontroller on the receiver captures and sends the data to a second module with the main microcontroller for further computational intelligence. These processes include signal pre-processing, features extraction, and classification to identify the brain signal, which translate into outputs for wheelchair control. This is shown above in the block diagram, which covers data acquisition, signal analysis or computational intelligence (signal pre-processing, features extraction, classification) and output. In data acquisition, brain signals are recorded, stored and made available for analysis, in this case using wireless EEG in the headgear. The main embedded system controller handles the computational intelligence part. The output component of the computational

intelligence controls the output device (wheelchair) and can also provide appropriate information about the output to the user as feedback. The wireless interface technology and the design of the low power instrumentation enhance the portability factor of the system.

Several researchers have used other non-motor imagery task BCI based on mental task imaginations. They used six EEG channels with electrodes placed on the scalp positions central (C3, C4), parietal (P3, P4) and occipital (O1, O2). The chosen mental tasks included baseline, multiplication, letter composing, figure 3-D rotation and counting (Anderson, Devulapalli & Stolz 1995b; Craig & Nguyen 2007; Friedrich, Scherer & Neuper 2012; Keirn & Aunon 1990; Lei et al. 2011; Palaniappan et al. 2002). For a practical system, the number of channels needs to be reduced. For example, some researchers have tried to use different combinations of two and three channels (Craig, Nguyen & Burchey 2006; Faradji, Ward & Birch 2009). Looking from the brain function point of view, it has been found that the parietal lobe shows significant activity during mental arithmetic calculation, language and writing skills especially on the left hemisphere (Menon & Desmond 2001; Osaka 1984; Rivera et al. 2005). Another study showed mental figure rotation created activity in both left and right parietal lobes (Milivojevic, Hamm & Corballis 2009). A visual task has been shown to produce activity more on the right occipital cortex (Vanni et al. 1996). Therefore, parietal and occipital areas provide significant features for BCI purposes. Consequently, this chapter focuses on the development of a system using two channel only wireless EEG placed on the left parietal (P3) and the right occipital (O2) lobes, which cover the mental tasks above.

For the computational intelligence methods, a Fast Fourier Transform (FFT) algorithm is used as a popular feature extractor for EEG-BCI. In the classification algorithm, the EEG signal is a multi-dimensional signal, a non-linear method named the artificial neural network (ANN) is investigated. The ANN has been widely used in biomedical applications or other engineering application, particularly for classification algorithms.

This chapter is organized as follows: section 3.2 covers the details of prototype design of the microcontroller-based BCI for mental task classification, which include wireless EEG, wireless receiver, computational intelligence and main embedded system controller; section 3.3 describes the performance measurement, section 3.4 describes the experimental data collection, section 3.5 covers the experimental results, section 3.6 covers the discussions and section 3.7 is the conclusion.

3.2 Components of Microcontroller-based BCI for Mental Task Classifications

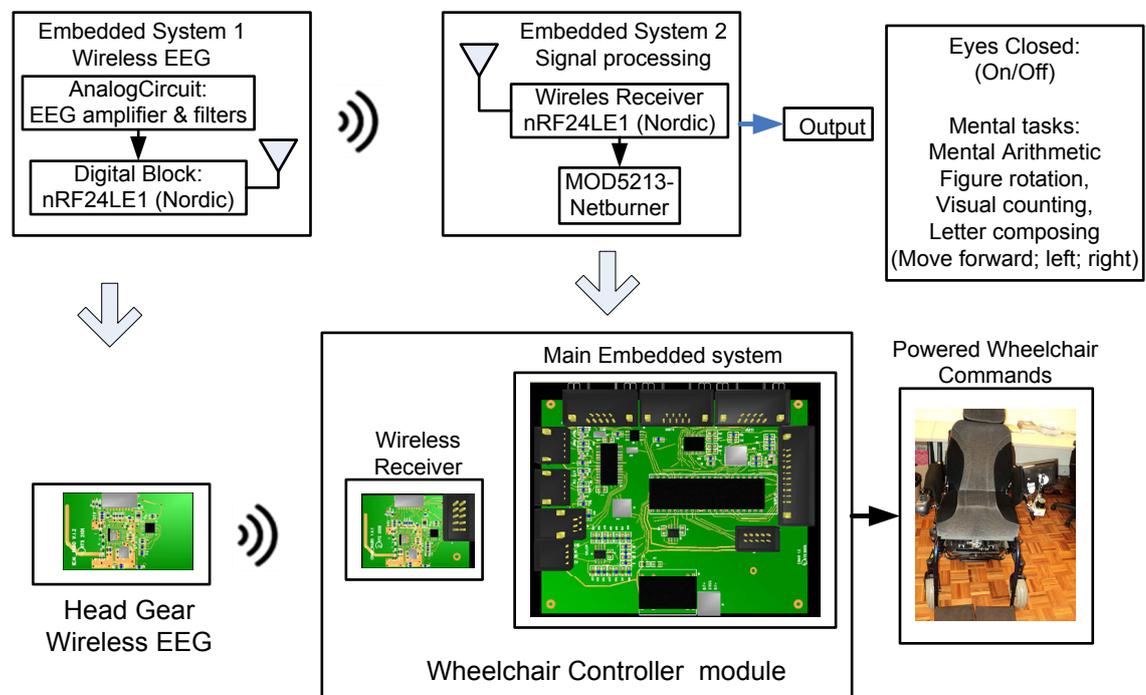


Figure 3.3: Block diagram of microcontroller-based BCI for mental task classifications

A complete block diagram of the BCI system for the purpose of controlling the wheelchair is shown in Fig. 3.3. The system uses two embedded system controllers. The first block is basically a wireless EEG module, which is divided into analog and digital sections. The natural EEG signal has a voltage level of a few micro volts, and therefore, the analog section has amplifiers and filters to boost up the signal. The

remaining digital section is handled by a wireless microcontroller, nRF24LE1 from Nordic semiconductor. The digital data will be transferred wirelessly to the receiver, which has the same controller, nRF24LE1. The receiver will transfer the data further to another embedded system using a MOD5213 (Netburner) controller which internally has a 32-bit ColdFire-MCF5213 (Freescale) microcontroller for signal processing and classification algorithm. The output of the classifier is mapped into output of the wheelchair commands. To make sure the system provides maximum portability, convenience and cost effectiveness, the design of the hardware wireless EEG will use a minimum components.

3.2.1 Wireless EEG and Wireless Receiver Development

3.2.1.1 Wireless EEG

Biomedical instrumentation as the sensor converts energy or information from the measurand to another form, such as electric form covers: measurand, sensor, signal conditioning and output. The measurand is the physical quantity or condition for the system to be measured for example: bio-potential of electromyography (EMG) for the muscle, electrocardiography (ECG) for the heart, electrooculography (EOG) for eyes movement and electroencephalogram (EEG) for the brain. The sensor or transducer is a device to convert one form of energy to another. It converts a physical measurand to an output of electrical signal. The signal conditioner is used as the signal from the sensor that can not be directly coupled to the output device. The signal conditioning process can cover amplification, filter and converting to digital form. The output can be used for the display or for further processing, which depends on the applications.

The similarity of EMG, EOG, ECG and EEG instrumentation is that they all measure the bio-electrical potential. The physiological parameters of these sensors, electrical potential and frequency ranges (bandwidth) are shown in Table 3.1 (Prutchi & Norris 2005; Webster 2010). It can be seen that for electrical potential ranges, EEG signal has the lowest range of measurement in term of microvolt, 5 – 300 μV . The

frequency ranges of the biopotential measurement are varied: EMG (up to 10 kHz); EOG (up to 50 Hz); Clinical ECG (0.67- 40 Hz); EEG (0-100 Hz). This means the instrumentation of EEG can be used to measure the other bio-potential measurement (EMG, EOG, and ECG) with additional adjustment on the frequency ranges. Furthermore, the ECG for the heart signal measurement has a unique signal (PQRST waves) compared to EEG or other measurement as non-linear and non-stationary spikes, as shown in Fig. 3.4. The bandwidth of the clinical ECG is inside the bandwidth of EEG measurement. As a result, the design of the EEG prototype could be tested for the ECG measurement as well.

Table 3.1: Electrical potential and frequency ranges of various Bio-potential signals

Various Bio-potential measurement	Electrical potential ranges	Frequency ranges (Bandwidth)
Electromyography (EMG)	0.1 – 5 mV	DC – 10 kHz
Electrooculography (EOG)	50 – 3.5mV	DC – 50Hz
Electrocardiography (ECG)	0.5 – 4 mV	R-R interval: 4 Hz – 7 Hz R-R variability due thermoregulation: 0.01Hz – 0.04 Hz R-R variability due baroreflex dynamics: 0.04 – 0.15 Hz R-R variability due to respiration PQRST complex: 0.05 Hz – 100 Hz Ventricular late potentials : 40Hz– 200Hz Clinical ECG: 0.67 Hz – 40 Hz
Electroencephalography (EEG)	5 - 300 μ V	Delta: 0 – 3 Hz Theta: 4 – 7 Hz Alpha: 8 Hz – 13 Hz Beta: 13 Hz – 30 Hz Gamma: 30 Hz – 100 Hz

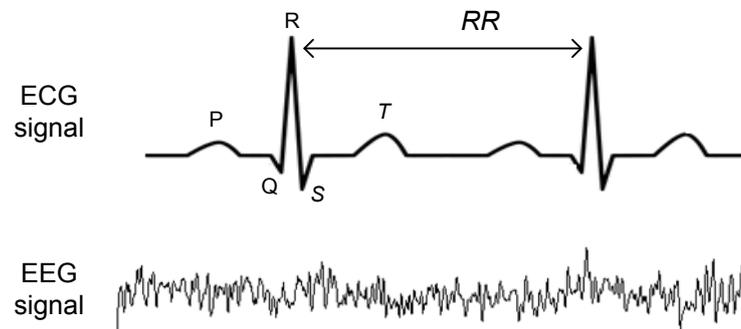


Figure 3.4: The shape of ECG and EEG signal of measurements

The general block diagram of an EEG measurement system is shown in Fig. 3.5. EEG electrodes are placed on the scalp to measure the brain electrical signals as the analog signal. The EEG comprises amplifier, filter and analog to digital (ADC) blocks.

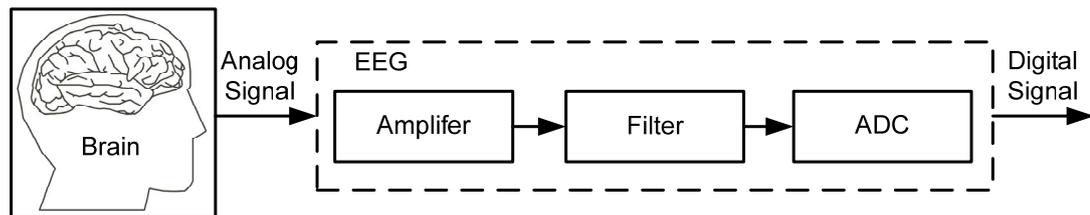


Figure 3.5: General block diagram of EEG measurement

The brain signals, generally resulting from physiological activity, have very small amplitude, as mentioned above at $5 - 300 \mu\text{V}$. Therefore, after acquiring the signal by EEG electrodes, they need to be amplified to the certain magnitude with minimum distortion before further processing. The important characterizing parameters of EEG instrumentation include gain, frequency response (bandwidth), common-mode rejection, input impedance and noise. As the EEG signal results from electrophysiological activity in order of a few microvolts, the EEG amplifier and most of the bio-potential amplifiers need a certain gain. The gain of the amplifier can be measured in decibel (dB). The linear gain can be calculated into decibel form by using the equation as follows:

$$Gain(dB) = 20 \log(\text{linear gain}) \quad (3.1)$$

The body of the human itself is a good conductor and can act as an antenna to pick up electromagnetic interference present in the surrounding environment, as illustrated in Fig. 3.6. Normally, the electromagnetic interference is the 50/60 Hz wave and its harmonics coming from the power line or radiated by power cord, fluorescent lighting, electrical machinery, computers and so on. As a result, the interference, especially on the single-ended bio-electrode, is very significant and often obscures the underlying electrophysiological signals. Therefore, the EEG instrumentation needs a high common-mode rejection ratio (CMRR) which is the measurement of its capability to reject common-mode signals.

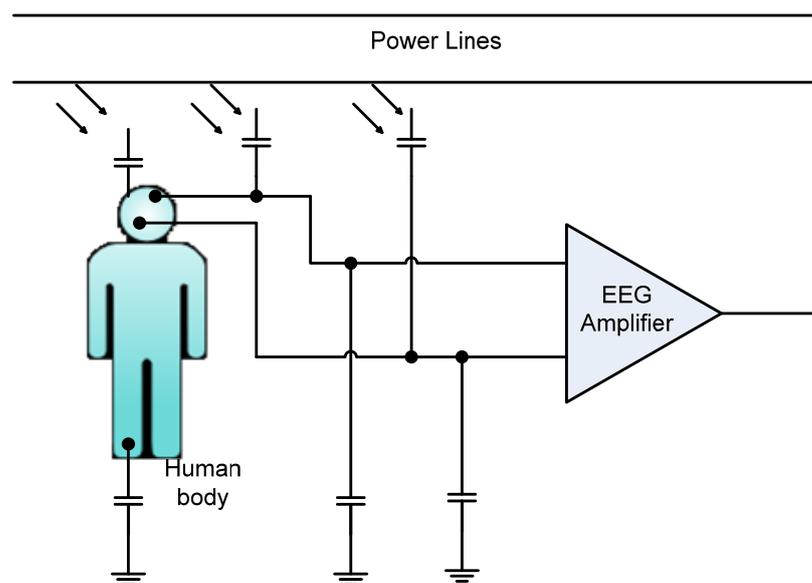


Figure 3.6: Human body and electromagnetic interference

CMRR is defined as the ratio between amplitude of the common-mode signal to amplitude of an equivalent differential signal that would produce the same output from the amplifier. CMRR is usually expressed in dB with the following:

$$A = V_{out} / V_{diff} \quad (3.2)$$

$$A_{cm} = V_{out} / V_{cm} \quad (3.3)$$

$$CMRR = V_{cm} / V_{diff} \quad (3.4)$$

$$CMRR (dB) = 20 \log CMRR \quad (3.5)$$

$$CMRR (dB) = 20 \log (A / A_{cm}) \quad (3.6)$$

$$CMRR (dB) = 20 \log ((V_{cm} / V_{out}) \times A) \quad (3.7)$$

where A denotes differential-mode gain, A_{cm} denotes common-mode gain, V_{out} denotes output voltage, V_{diff} denotes differential input voltage, V_{cm} denotes common-mode input voltage, $CMRR$ denotes common-mode rejection ratio and $CMRR (db)$ denotes common-mode rejection ratio in dB. Instrumentation of EEG amplifier requires a high CMRR above 80 dB to tolerate the interfering noise into the system (MettingVanRijn, Peper & Grimbergen 1994). The input impedance of the bio-potential amplifier needs to be sufficiently high so as not to attenuate significantly the original electrophysiological signal of measurement so as to maintain a high signal to noise ratio against the unwanted signals that contaminate an original bio-potential signal under measurement. Noise is generated within the amplifier circuit and is usually measured in microvolts peak to peak (μV_{pp}) or microvolts root mean square (μV_{RMS}). It applies as if it were a differential input voltage.

Most bio-potential amplifiers include EEG amplifier and are operational amplifier (op-amp)-based circuits. The voltage present at the output on the op-amp is proportional to the differential voltage across its inputs. As a result, the non-inverting input produces an in-phase output signal and the inverting input produces an output signal with phase of 180° out of phase of the input. The two basic rules that need to be followed are: 1) The two input terminals of the op-amp are at the same voltage when the output op-amp is in its linear range; 2) There are no current flows into either input terminal of the op-amp. For the first rule, if the two input terminals are not at the same voltage, the

differential input voltage would be multiplied by the infinity gain, which provides infinite output voltage. Most op-amps use a certain level of supply of power. As a result, the output voltage is restricted to this range. For the second rule is true as assuming the infinity for the input impedance and there is no current flows into an infinite impedance. Additionally, the first rule also stated no voltage drop for the input; therefore, there are no current flows.

Thus, for ideal op-amps to have infinity input impedance implies that the input current is zero. The inverting input will neither sink nor source any current. The circuit of an op-amp is shown in Fig 3.7. The V_i as input signals is presented through resistor R_i to the inverting input of an ideal op-amp. Resistor R_f provides feedback to amplifier input from amplifier output. The non-inverting input is tied to the ground. With the fact in an ideal op-amp, the setting conditions at one input will have the same condition on the other input, thus point X can be treated as grounded.

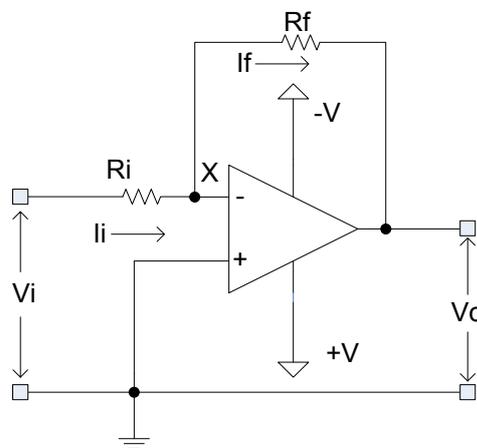


Figure 3.7: Op-Amp in inverting voltage amplifier

According to the Kirchhoff's law, the total of currents at junction X should be sum to zero, thus the following are the equations for inverting voltage amplifier:

$$I_i = -I_f \quad (3.8)$$

$$I_i = \frac{V_i}{R_i} \quad (3.9)$$

$$I_f = -\frac{V_i}{R_i} \quad (3.10)$$

$$V_o = -\frac{R_f V_i}{R_i} \quad (3.11)$$

$$V_o = -G V_{in} \quad (3.12)$$

$$G = -\frac{R_f}{R_i} \quad (3.13)$$

where G denotes the voltage gain constant, V_+ and V_- are the positive and negative supply power. The non-inverting voltage amplifier, as known as a non-inverting voltage follower, is presented in Fig. 3.8. In non-inverting amplifier $I_i = I_o$. By using equation 8 and 9 above, I_f , V_o and G are as follows:

$$I_f = \frac{V_o - V_i}{R_f} \quad (3.14)$$

$$V_o = \left(1 + \frac{R_f}{R_i}\right) V_i \quad (3.15)$$

$$G = 1 + \frac{R_f}{R_i} \quad (3.16)$$

When R_i and R_f equal zero, it leads to a resistance ratio of $R_f/R_i = 0$, which in turn results in unity gain which is known as unity gain buffer or voltage follower as shown in Fig. 3.9. This circuit is often used to couple a high impedance signal to a low impedance processing circuit connected to the very low impedance output of the op-amp in the biomedical instrumentation.

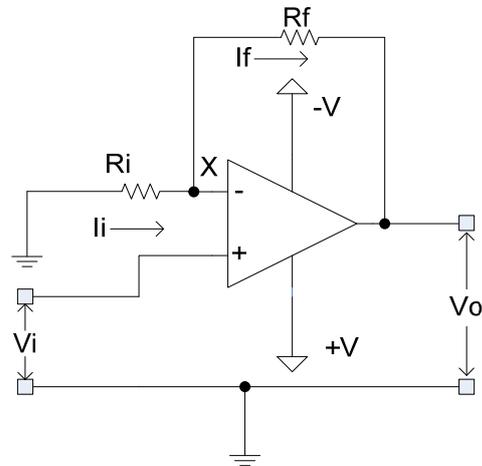


Figure 3.8: Op-Amp in non-inverting voltage amplifier

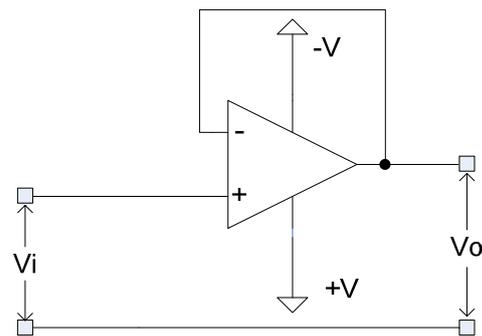


Figure 3.9: Op- Op-Amp in a unity-gain buffer

The need of an amplifier is an important part in modern instrumentation of bio-potentials measurement due to such measurements involving low level voltages which is known as bio-bio-potential amplifier. All bio-potential amplifiers need to meet certain basic requirements. They need to have high input impedance to provide minimal loading of the signal measurement. The bio-potential amplifier should also provide protection to the user. As a result, isolation and protection circuitry are needed so the current through the electrode circuit can be kept safe and any artefact generated by such current can be minimized. As mentioned previously, the bio-potential signals have a certain bandwidth, therefore the bio-potential amplifiers would need to operate in that

portion of the frequency spectrum in which the signal they amplify exists. Because of the low level of such a signal, it is important to limit the bandwidth of the amplifier with just enough to process the signal adequately. This is to obtain optimal signal-to-noise ratio (SNR). Bio-potential signals are frequently measured from bipolar electrodes with electrically respect to ground and the most appropriate one is a differential amplifier. With the bipolar electrodes, a common mode voltage can occur with respect to ground that is much larger than the signal amplitude and the symmetry with one electrode at ground potential. As a result, the bio-amplifier needs a high CMRR to minimize interference from the result of the common mode signal.

Fig.3.10 illustrates the bio-potential measurement based on the differential and common-mode voltage applied to the input of an amplifier. The desired bio-potential appears as a voltage between the two input terminals of the differential amplifier, thus it amplifies the difference between two voltages of the two input terminals. Furthermore, the differential amplifier does not amplify on particular voltages.

When a differential voltage is applied to the input terminals of an op-amp, the transfer function of the inverting follower circuit and non-inverting follower circuit can be derived as follows:

$$V_{out1} = -\frac{R_f}{R_i}(V_1 - V_2) \quad (3.17)$$

$$V_{out2} = \left(1 + \frac{R_f}{R_i}\right)(V_1 - V_2) \quad (3.18)$$

where V_{out1} and V_{out2} denote the output voltage of the inverting follower and non-inverting follower circuits; $V_1 - V_2$ is basically the differential voltage (V_d); V_{cm} is the common mode voltage. The balance of a differential amplifier is important to preserve the property of an ideal amplifier which is its infinity of CMRR value. An implementation of a differential amplifier with an op-amp is shown in Fig. 3.11. If $V_1=V_2$, an output voltage should be equal to zero, disregarding any common-mode

voltage V_{cm} . The common mode rejection deteriorates if the resistor equalities are not preserved ($R1 = R2$ and $R3 = R4$).

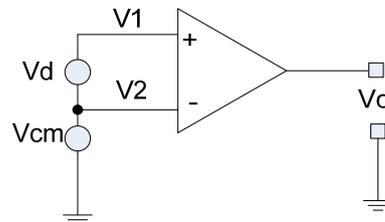


Figure 3.10: Differential amplifier analogy with differential and common-mode voltage applied to the input

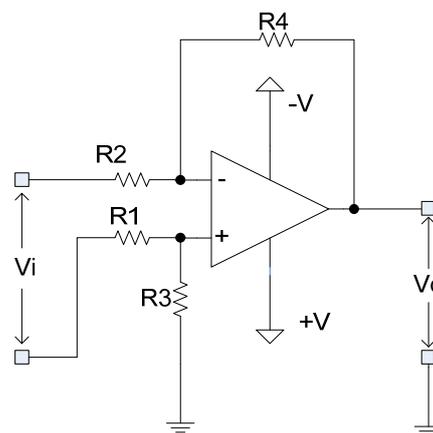


Figure 3.11: Differential amplifiers with an op-amp

The main issue of using a simple differential amplifier as a bio-potential amplifier is its low input impedance. As an option to the simple differential amplifier, the multiple op-amps in differential configuration can be used. This configuration is known as instrumentation amplifier (In-Amp) with the advantage of preserving the high input impedance of the non-inverting follower and offering gain. In fact, bio-potential amplifiers are seldom built in the current state using individual op-amps. The In-Amp typically requires external resistors to set their gain. These resistors do not affect the high CMRR and high input impedance achieved in In-Amp due to its precise matching

of the internal components. There are two ways to implement the bio-potential amplifier using In-Amp: alternating current (AC)-coupled In-Amp and direct current (DC) coupled In-Amp design. An AC-coupled In-Amp as shown in Fig.3.12 can be used as the nature of bio-potential signal is an analog differential signal. The AC coupling can accurately extract the AC signal. The AC coupling can also suppress the electrode DC offset potential. Moreover if the amplification in a single stage is set to maximal, it will provide maximum CMRR.

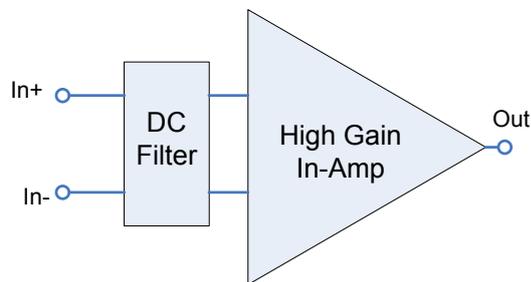


Figure 3.12: AC coupled In-Amp

High gain at the first stage is difficult to achieve for the EEG measurement due to the electrode offset voltages. As a result, a DC filter circuit is placed at the front of the AC-coupled In-Amp. This can be achieved with a passive element to form the RC high pass filter. The first prototype of two channels of wireless EEG instrumentation is shown in Fig. 3.13. The general EEG measurement uses the bipolar configuration. This initial hardware design was based on an AC coupled In-Amp, which has a DC filter at the front end amplifier to make sure no saturation issue of electrodes offset occurs. This is followed by an amplifier which has In-Amp AD8553 with the high gain of 1000 times or more. The In-Amp contains voltage to current amplifier, current to voltage amplifier and a high precision auto-zero amplifier. Before connecting to the wireless microcontroller, an anti-aliasing filter is used to prevent the aliasing interference. The microcontroller handles the analog to digital conversion (ADC) and the wireless transmission. An additional block also can be found as a module to generate the virtual GND as the reference of the EEG measurement. The schematic of wireless EEG V1

shown in Fig.3.14 is comprised of the RC filter as DC filter, In-Amp with high gain, RC filter as anti-aliasing filter and virtual GND circuit.

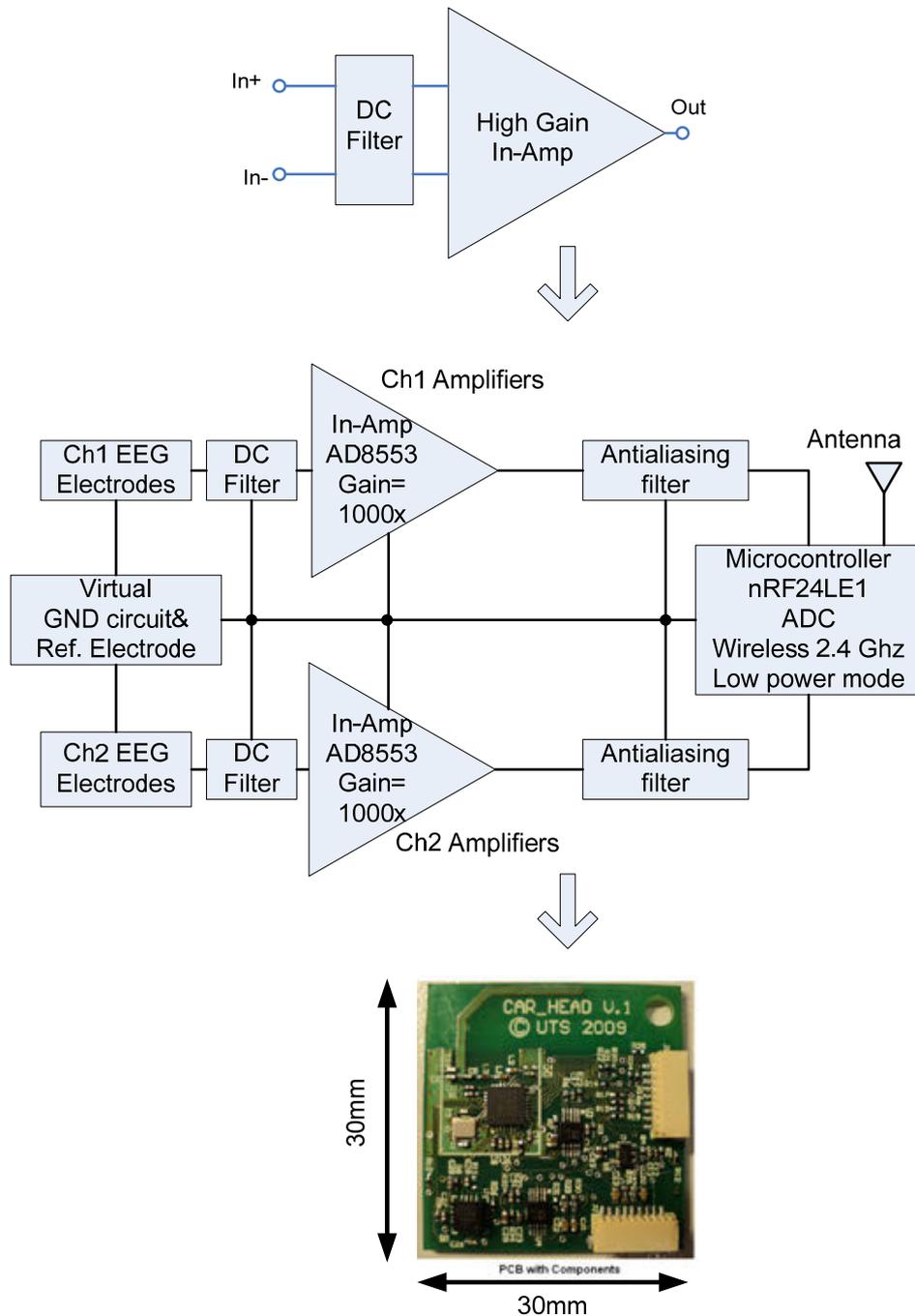


Figure 3.13: Prototype wireless EEG V1

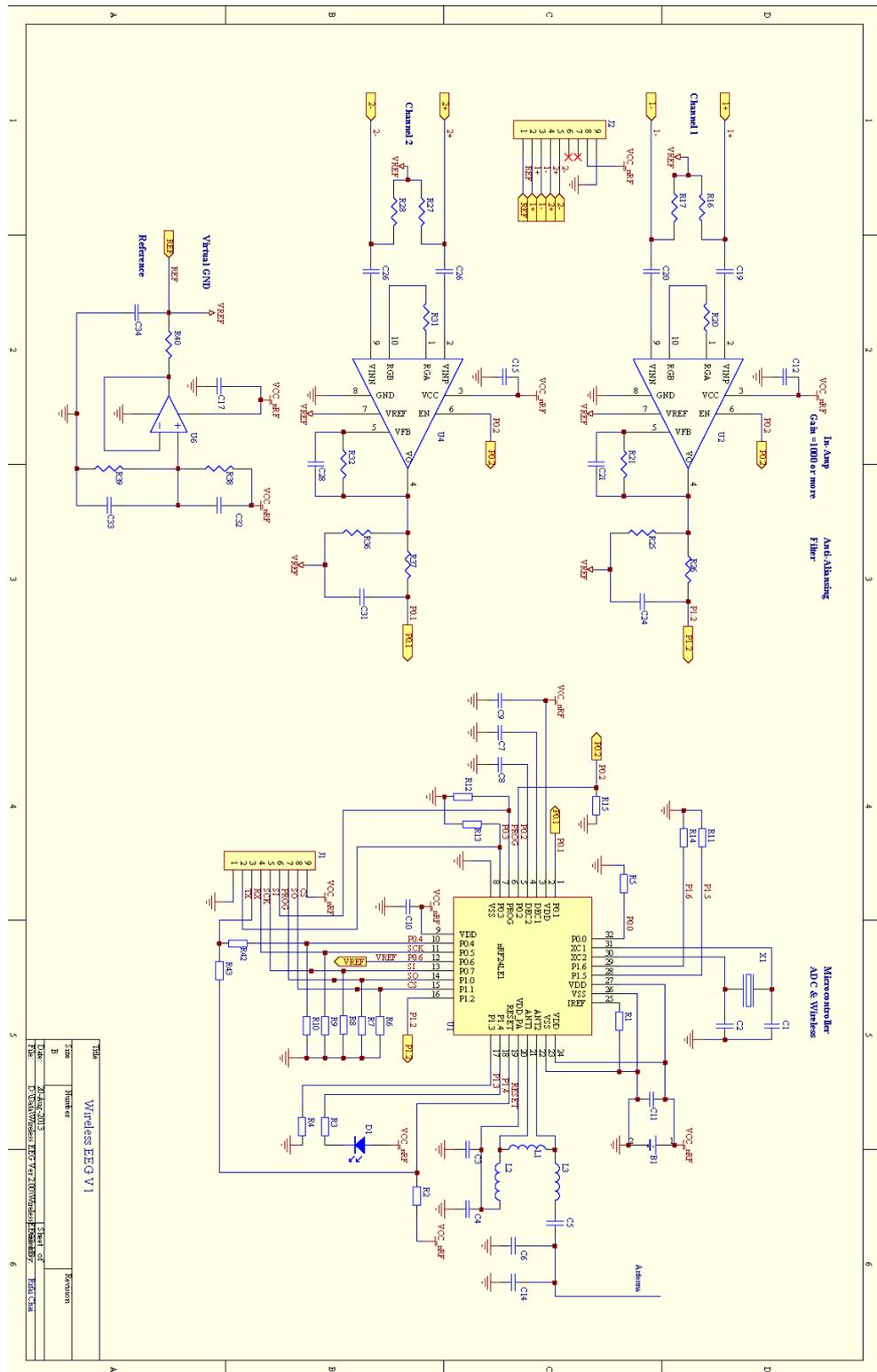


Figure 3.14: Schematic of wireless EEG V1

The second version of the wireless EEG is based on the DC coupled In-Amp design. This is done by using In-Amp with a small gain at the first stage and continuing by removing the DC offset voltage before going into other stages of amplifier, as shown in Fig.3.15.

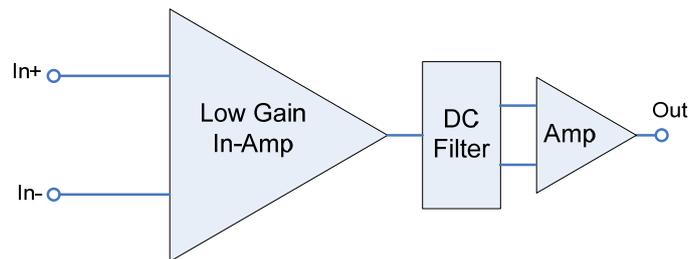


Figure 3.15: DC coupled In-Amp

A block diagram and printed circuit board (PCB) for wireless EEG V2 can be seen in Fig.3.17. The wireless EEG V2 is a two channels bipolar wireless EEG where each channel has a non-inverting and inverting input channel. A reference electrode is also added to give a total of five electrodes. The schematic of Wireless EEG V1 is shown in Fig.3.18. The circuit contains EEG gold electrodes, filters and amplifier circuits in an analog section and an nRF24LE1 wireless microcontroller for the digital processing and wireless data transmission.

The wireless EEG is a coin cell battery powered system. There are two stages of amplification for the electrode signals. A precision current mode instrumentation amplifier (In-Amp) AD8553 is used for the first stage. This internally contains a voltage to current differential amplifier with auto correction for DC offset and flicker noise and a current to voltage amplifier on the output. To accommodate the DC offset from the electrodes, which could saturate the amplifier, the gain of the In-Amp is set at a low value of around 10, followed by a passive RC 0.15 Hz high-pass filter to remove the DC offset. The second stage amplifier uses an OPA333 (Texas Instruments) op-amp in a non-inverting amplifier circuit with a gain adjustable up to 1000 using a potentiometer.

The circuit uses the single supply amplifier configuration. As a result, the choice of the In-Amp and op-amps follows the unipolar amplifier configuration. To cope with the issue of not being able to generate negative voltage, a virtual ground is used. A centre point virtual ground is provided for the single supply system, using a resistive voltage divider, followed by a voltage follower also using an OPA333 op-amp. This virtual ground is simply a voltage reference that is typically halfway between supply power ($+V$) and ground, as shown in Fig.3.16. Two resistors form a voltage divider where $+V/2$ appears at the non-inverting input of the op-amp. $+V/2$ also appears at the output.

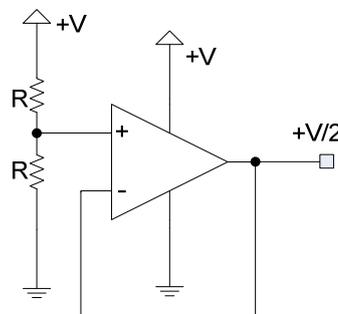


Figure 3.16: Virtual ground for single supply amplifier

A differential 1.5 KHz low-pass filter is added to the input of the first amplifier stage to minimize radio frequency interference (RFI). A 100 Hz double-pole active low-pass filter is added to the second stage amplifier to restrict bandwidth for further noise reduction. At the end of the analog block, before connecting to an analog digital converter (ADC), a 1.5 KHz anti-aliasing filter is added.

The ADC has 12 bits resolution and is configured in differential mode for an improved common mode rejection. For further noise rejection, four ADC samples are taken and averaged for each reading. This is particularly effective at removing noise that is internally generated by the microcontroller. The sampling rate of the ADC is 256Hz, which means the ADC acquires 256 samples per second (about one every 4 ms). The minimum requirement of the sampling rate is needed to satisfy the requirement of the Nyquist criterion as it should be at least two times larger than the highest frequency

occurring in the EEG signal (delta, theta, alpha, beta, and gamma rhythms). If the Nyquist criterion is not satisfied, then the digital signal will be distorted by aliasing. Furthermore, when aliasing occurs, the digitized signals with frequencies that are higher than half the sampling rate masquerade as a signal of lower frequency. The ADC is part of the wireless microcontroller nRF24LE1.

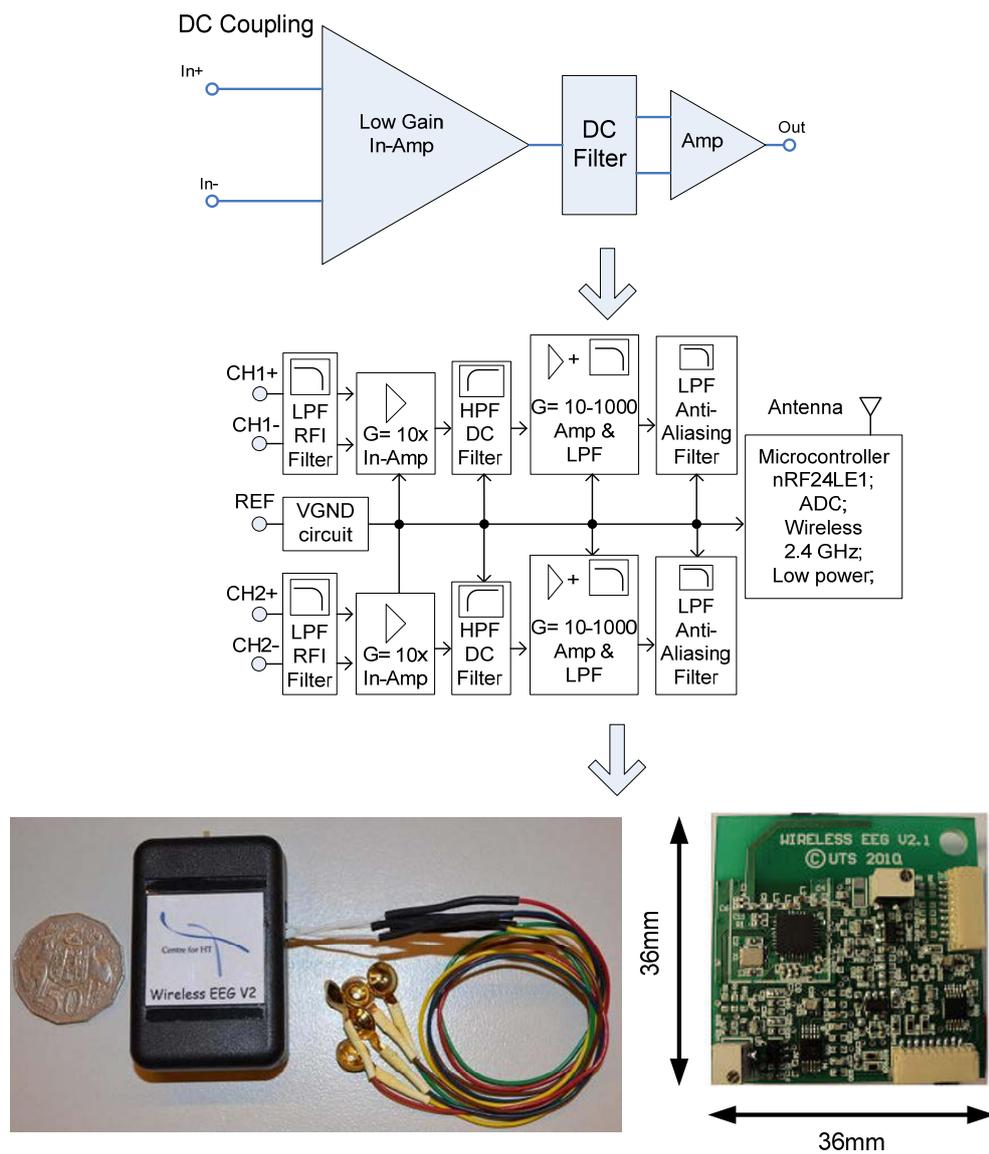


Figure 3.17: Prototype wireless EEG V2

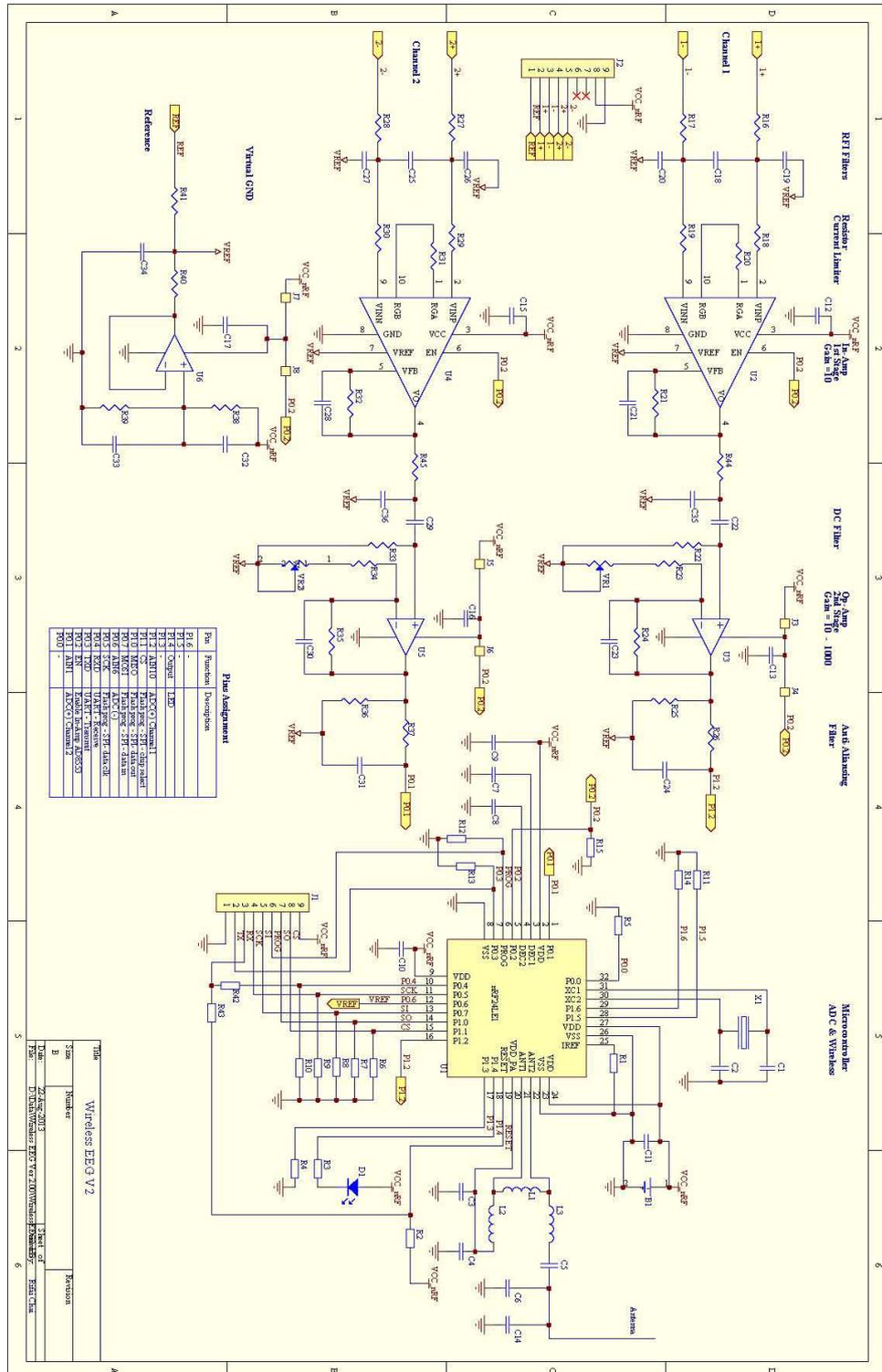


Figure 3.18: Schematic wireless EEG V2

The flowchart diagram of the firmware for the wireless EEG is shown in Fig.3.19. The important functions of the firmware for wireless EEG are illustrated in Fig.3.20. The system creates a sampling rate at 256 Hz by using the interrupt timer of the microcontroller nRF24LE1. Fig.3.21 shows the result of testing of the interrupt timer at 256 Hz by toggling the output port. Each state transition has 256 Hz or around 4 ms.

Once the analog signal is amplified and filtered, the next process is digitization using the internal analog to digital converter (ADC) of the microcontroller nRF24LE1. The ADC is configured in differential mode for improved common mode. The averaging of multiple sequences of ADC readings is used to provide more stable ADC acquisition. The following are the equation for the least-significant bit of the ADC and the digital conversion equation:

$$LSB = \frac{FS}{2^n} \quad (3.19)$$

$$DIG = \frac{2^n (IN^+ - IN^-)}{V_{ref}} + \frac{2^n}{2} \quad (3.20)$$

where LSB is least- significant bit, FS is full scale analog input, n is the number of bit resolution of ADC, DIG is the digital conversion value, IN^+ is the voltage of non-inverting input of ADC and IN^- is inverting input voltage. V_{ref} refers to voltage reference. The ADC setting has configurations which include: internal reference at 1.2V ($V_{ref}=1.2V$); 12-bit resolution ($n=12$); inverting input (IN^-) is connected to virtual GND; a midway of the power 3.3V, which is around 1.65V; the non-inverting input (IN^+) is tied to the output of the amplifier module; in differential mode of ADC, the input range swings from $-V_{ref}/2$ to $+V_{ref}/2$ which is equal to -0.6 to +0.6 V; the total full scale (FS) will be equal to 1.2 volts. LSB can be calculated by using equation (3.1) which gives a result around 293 μ Volts. Total gain of the proposed wireless EEG will be adjusted to around 2000.

Fig.3.22 shows the experimental measurement of the output conversion on different voltages. The ADC conversion range is between 0 and 4095 for 12 bit ADC. The middle point (2047) will be reached when both inputs have zero difference.

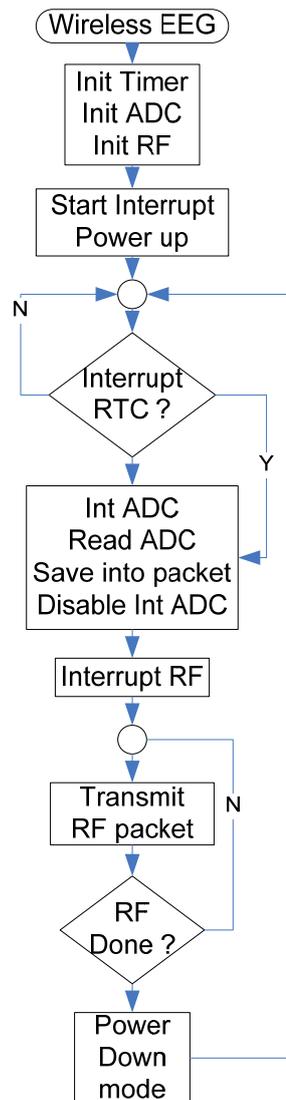


Figure 3.19: Flowchart of firmware wireless EEG

Wireless EEG functions:	
Sampling rate 256Hz	→ Interrupt Timer
Signal acquisition	→ Interrupt ADC
Wireless Transmission 2.4 GHz	→ Interrupt RF
Debugging	→ Interrupt UART
Power saving	→ Interrupt RTC (swap with the Interrupt Timer)

Figure 3.20: Internal functions/routines of firmware for Wireless EEG

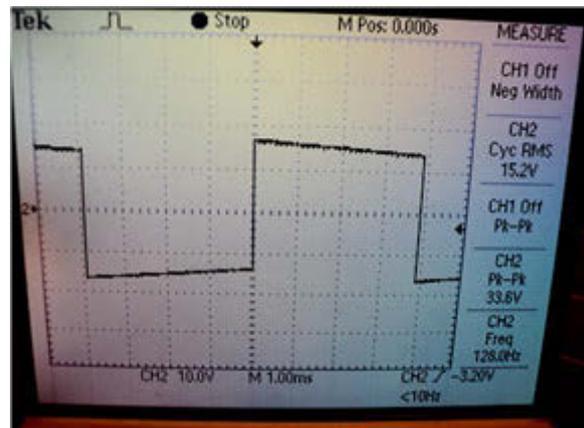


Figure 3.21: Interrupt timer at 256 Hz (~ 4 ms)

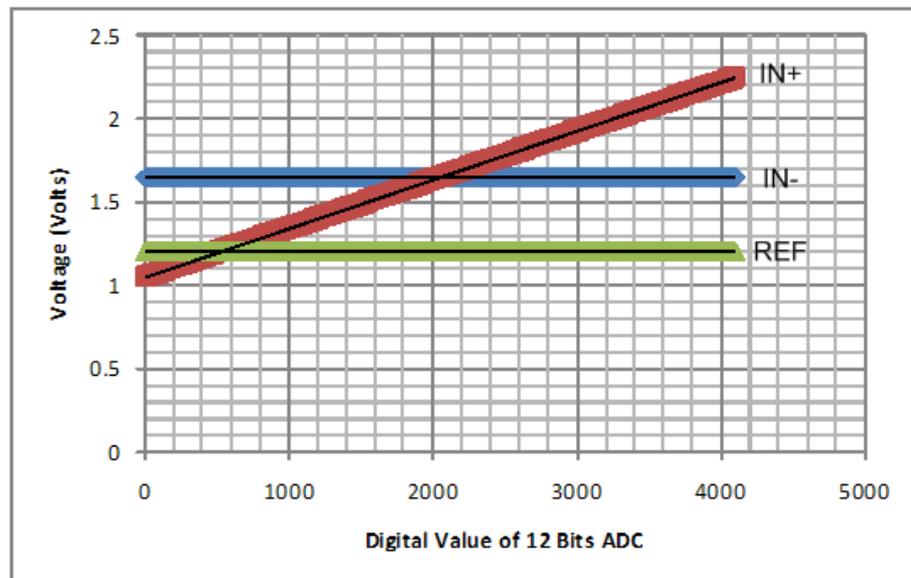


Figure 3.22: ADC conversion measurements on the wireless EEG

Power consumption is a crucial factor in creating a portable wireless EEG. A special timer register real time clock (RTC) is substituted with the original timer to create the 256 Hz sampling rate of the system. RTC provides an additional wake up function for power saving mode. The microcontroller nRF24LE1 has the nRF24L01+ which is the ultra low power 2 Mbps RF transceiver integrated circuit (IC) for the 2.4 GHz industrial, scientific and medical (ISM) band. For the low power consumption of the EEG system, the RF section runs in power down mode, which turns off every time the transmission is finished and waits for the next RTC wakeup cycle.

In power down mode, the RF transceiver is disabled with minimal current consumption. All the register values are maintained and can be activated. The air data rate of the RF transmission is the modulated signalling rate the RF transceiver uses when transmitting and receiving data. It can be selected from 250kbps, 1 Mbps or 2 Mbps. Using a lower air data rate gives better receiver sensitivity than a higher air data rate, but a high air data rate would provide lower current consumption and reduce the probability of on-air collisions. The RF channel frequency decides the centre of the channels used by the RF transceiver. The channels occupy a bandwidth of less than 1 MHz at 250 kbps and 1Mbps with a bandwidth of less than 2 Mhz at 2Mbps. The RF transceiver is able to operate on frequencies from 2.4 Ghz up to 2.525GHz with the programming resolution of the RF channel frequency setting at 1 MHz. At 2 Mbps, the channel occupies a bandwidth wider than the resolution of the RF channel frequency setting. The channel spacing must be 2 MHz or more to ensure non-overlapping channel in 2 Mbps mode. At 1 Mbps and 250kbps the channel bandwidth is the same or lower than the resolution of the RF frequency.

For the transmission, the automatic packet assembly and disassembly are used. The automatic packet assembly puts together the preamble, address, packet control field; payload and cyclic redundancy check (CRC) to make a complete packet before it is transmitted. After the packet is validated, the automatic packet disassembly takes apart the packet and loads the payload into the receiver buffer and asserts an interrupt request (IRQ) as a data ready interrupt in the buffer.

Table 3.2: Protocol packet of the wireless EEG

No	Description	Value	Number of byte
1	1 st synchronization	A5 (hexadecimal)	1 byte
2	2 nd synchronization	5A (hexadecimal)	1 byte
3	Counter	Not used	1 byte
4	Version	Not used	1 byte
5	High byte channel 1	Value of high byte Ch. 1	1 byte
6	Low byte channel 1	Value of low byte Ch.1	1 byte
7	High byte channel 2	Value of high byte Ch. 2	1 byte
8	Low byte channel 2	Value of low byte Ch. 2	1 byte
9	High byte channel 3	Not used	1 byte
10	Low byte channel 3	Not used	1 byte
11	High byte channel 4	Not used	1 byte
12	Low byte channel 4	Not used	1 byte
13	High byte channel 5	Not used	1 byte
14	Low byte channel 5	Not used	1 byte
15	High byte channel 6	Not used	1 byte
16	Low byte channel 6	Not used	1 byte
17	Switches	Not used	1 byte

The packet data transmission to cope with the two channels EEG itself uses the OpenEEG (OpenEEG 2008) protocol with the total packet per transmission at 17 bytes. The reason to use the OpenEEG protocol is that the same software on the PC can be used for recording purpose. Table 3.2 shows detail of the packet protocol. The first two bytes are a constant to acknowledge the start of the packet. The third and fourth bytes are constants only and they are not used. Since the wireless EEG has two channels, only the relevant channel is updated. Data bytes for channel one are placed on the fifth and sixth bytes continued with channel two in the seventh and eight bytes. The rest of the data bytes are not used in this design.

3.2.1.2 Wireless Receiver

Wireless receiver module uses the same microcontroller as the wireless EEG, which is nRF24LE1. Data from the wireless EEG is transferred to the embedded system via a serial peripheral interface (SPI) port and at the same time to the personal computer (PC) via universal asynchronous receiver-transmitter (UART) or RS-232 for the data collection and further process. Fig.3.23 shows the wireless receiver module, which contains a wireless microcontroller module and connector to the main embedded system.

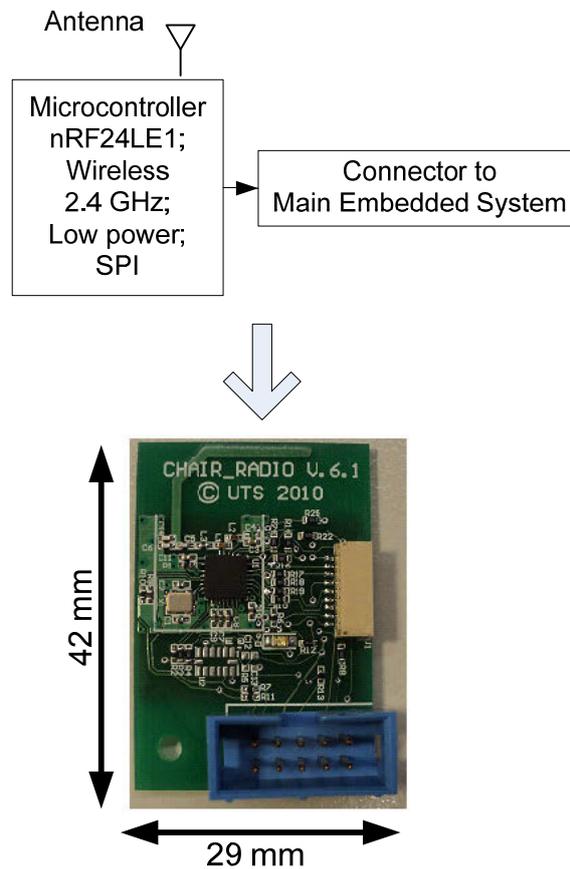


Figure 3.23: Wireless receiver

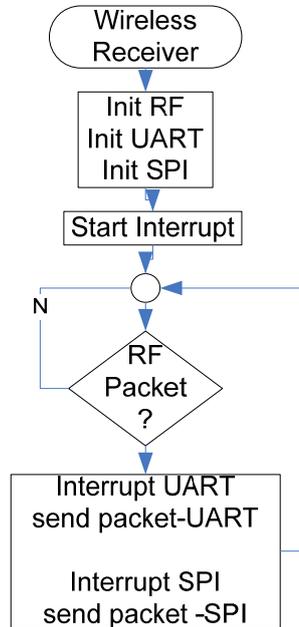


Figure 3.24: Flowchart of firmware wireless receiver

The flowchart of the firmware for the wireless receiver is shown in Fig.3.24. This is started by initialising for the radio frequency (RF), UART, and serial peripheral interface (SPI). This is continued with the interrupt RF activation. As soon as the receiver detects any incoming data from the interrupt RF routine and protocol is matched, data is forwarded to UART and SPI ports simultaneously. A PC captures the data via UART/RS-232 and the embedded system gets the data via SPI bus.

3.2.2 Software on PC for Wireless EEG Recording

The reason to use the packet protocol from the OpenEEG system is some available open source recording software can be used together if the wireless EEG uses the same protocol. One of the open source software is BrainBay (Veigl). The personal computer (PC) is connected to the main embedded system via the RS-232 port for the recording purpose only.

In this software design configuration, EEG signals are connected to a processing element for visual, data saving collection and so on. The following Fig.3.25 shows a design configuration that connects the 2 channels of the wireless EEG to the oscilloscope display elements for the real-time data display, Fast Fourier transform (FFT) display, EEG rhythms (delta, theta, alpha and beta) display and recording to an archive file. Fig. 3.26 shows the software when connected to the wireless EEG.

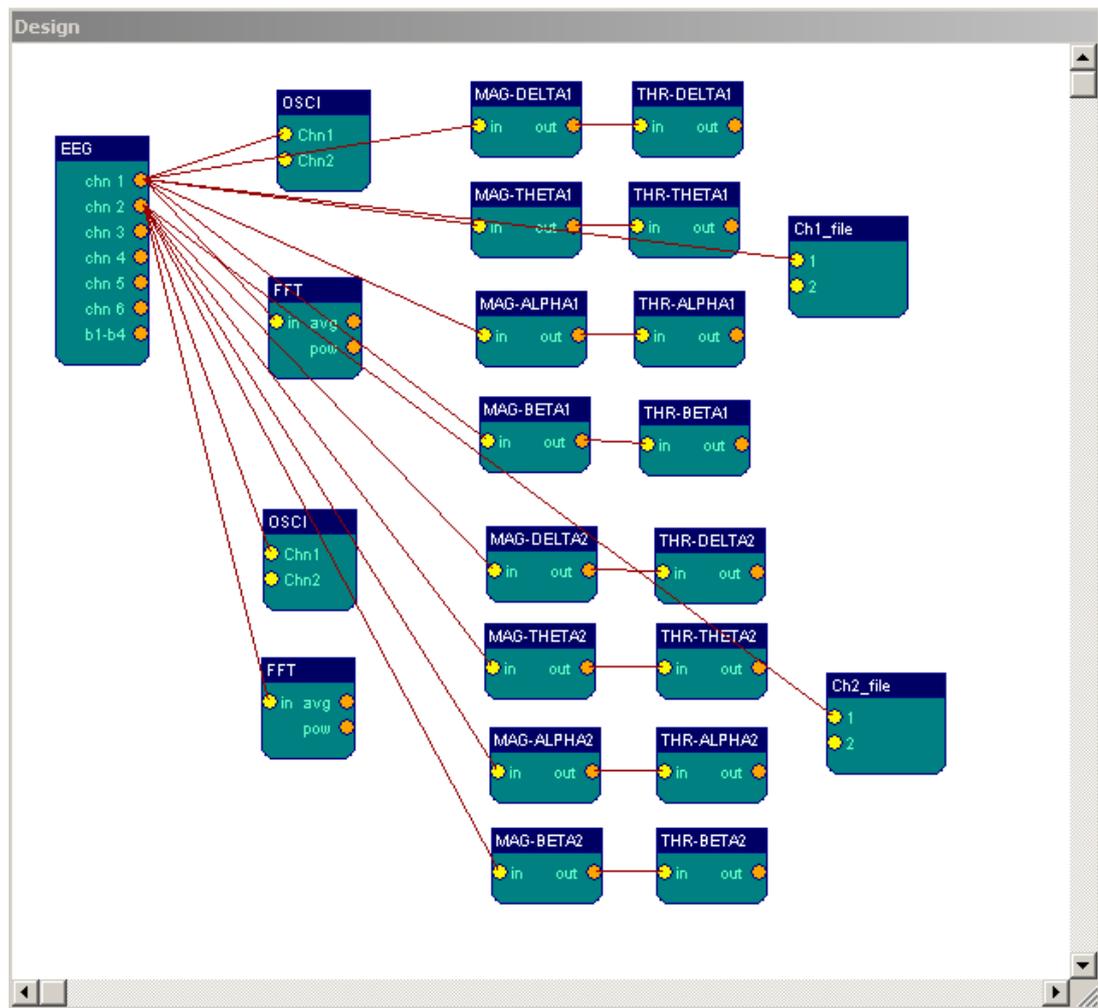


Figure 3.25: Design configurations on the recording software

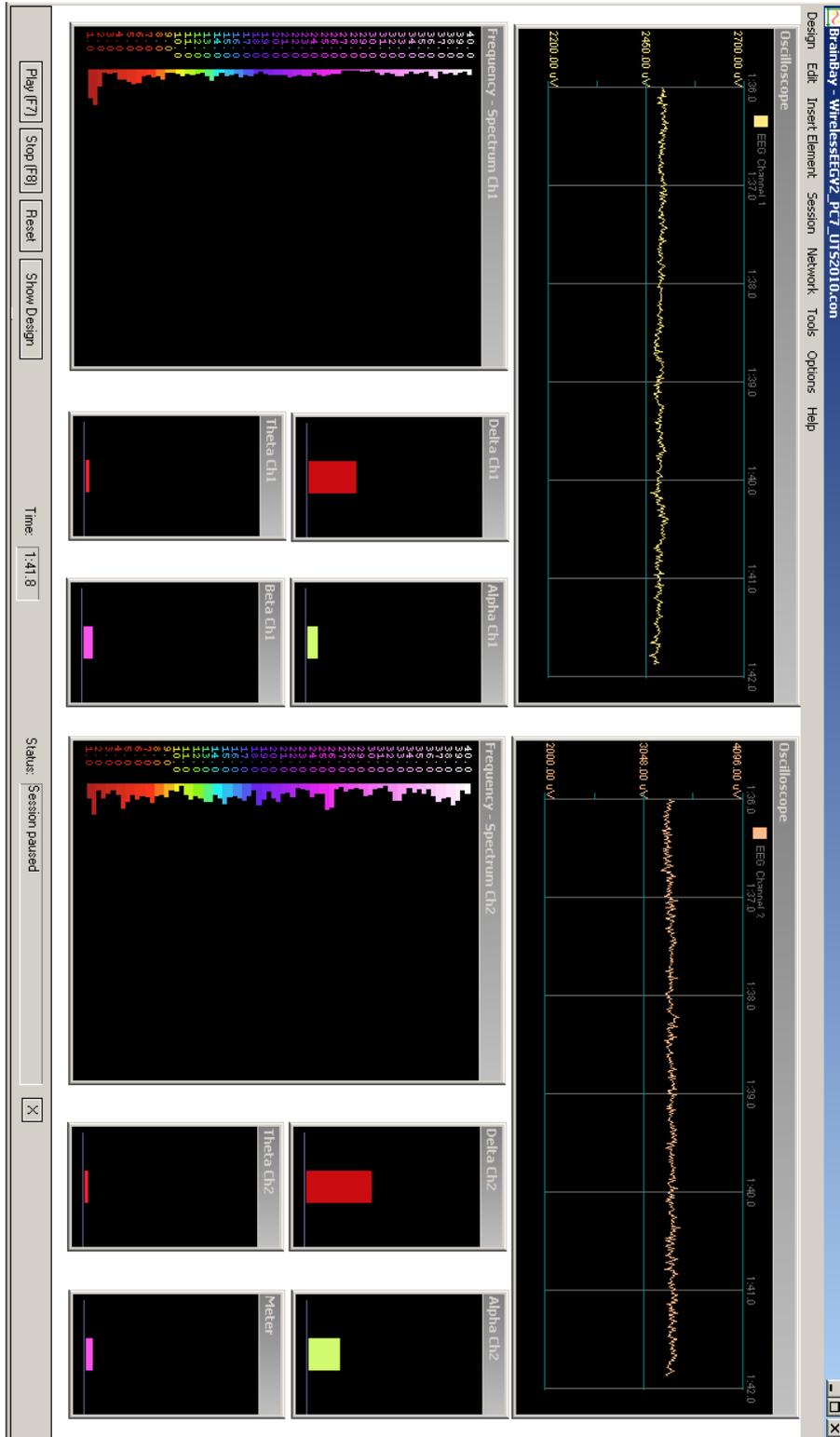


Figure 3.26: Software on PC for wireless EEG recording

3.2.3 Computational Intelligence

Inside the computational intelligence the EEG signal pre-processing take place to reduce noise and to enhance the relevant aspect of the EEG signals. This is continued by using the features extraction algorithm to extract the useful EEG features. Finally, the classification method is applied to classify the feature into outputs.

3.2.3.1 Signal Pre-processing

Digital signal processing (DSP) filters are used for the signal pre-processing step. Given a digital signal $x[n]$, each input sample of $x[n]$ enters the filter sequentially. The digital filter's output $y[n]$ at any given time point is simply a weighted sum of the present and past input samples:

$$y[n] = \sum_{l=0}^{M-1} b_l x[n-l] - \sum_{l=1}^N a_l y[n-l] \quad (3.21)$$

where $x[n-l]$ denotes the l^{th} past input sample, $y[n-l]$ denotes the l^{th} past output sample, b_l and a_l are scalar weights for each input and output sample, M and N denote the number of input and output sample weights employed by the filter. The resulting filter output $y[n]$ is a digital signal that has the same number of samples as $x[n]$, where the first sample of $y[n]$ corresponds to the first sample of $x[n]$.

The moving average filter is the most common filter in DSP due to its simplicity and ease of use. It is effective particularly for removing high frequency random noise while smoothing a signal. The moving average filter is so named because it takes the arithmetic mean of a number of past input samples in order to produce each output sample. This may be represented by following:

$$y(i) = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j] \quad (3.22)$$

where $x[n]$ is the input signal, $y[n]$ is the output signal, and M is the number of points used in the moving average. The moving average filter is good for reducing white noise while keeping the sharpest step response.

An additional band-pass filter is used to clean out further the original EEG noise. A band-pass filter will pass band frequencies while attenuating frequencies above or below that band. Here the pass-band exists between the lower pass-band edge frequency and upper pass-band edge frequency. A band-pass filter has two stop-bands. The lower stop-band extends from zero to first stop band frequency, while the upper stop-band extends from the second stop-band frequency to infinity. A Butterworth band-pass filter with the bandwidth 0.5Hz to 100Hz is applied in this experiment. The Butterworth filter is a classic filter approximation that has a smooth response in both pass band and stop band that provide the most linear phase response compared to other approximation techniques.

3.2.3.2 Feature Extraction Algorithm based on Fast Fourier Transform (FFT)

The purpose of the BCI is the characteristic of brain detection and quantification that indicate what the intention of the user is and translates their measurement in real-time into the device control. The brain characteristics used in this case are the signal features. The BCI features extraction is the process of distinguishing the brain signal characteristic from extraneous content and representing them in a compact and meaningful form. An accurate and robust feature extraction simplifies the subsequent classification stage and produces more accurate and reliable actions.

In this chapter, Fast Fourier transform (FFT) frequency domain analysis is the most used feature extraction algorithm in the BCI applied. Much of the signal processing theory is rooted in Fourier analysis, which transforms a time-domain signal into its equivalent frequency domain. The primary utility of Fourier analysis is to decompose a signal into individual sinusoidal components that can be isolated and evaluated independently. In Fourier analysis, the signal can be represented as a sum of a number

of amplitude-scaled and time-shifted sinusoids at specific frequencies. To model continuous signals, it is necessary to properly adjust the phase and magnitude of each sinusoid. For an arbitrary signal $x(t)$, the magnitude (scale) and the phase (shift) of the sinusoid at each frequency ω (radian) = $2\pi f$ (Hz) required to represent the arbitrary signal can be determined from the Fourier transform:

$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt = \int_{-\infty}^{\infty} x(t)[\cos \omega t + j \sin \omega t] dt \\ &= \int_{-\infty}^{\infty} x(t) \cos \omega t dt + j \int_{-\infty}^{\infty} x(t) \sin \omega t dt \\ &= a(\omega) + jb(\omega) \end{aligned} \quad (3.23)$$

The magnitude and phase for each sinusoidal component are expressed as:

$$\text{Magnitude : } |X(\omega)| = \sqrt{a^2(\omega) + b^2(\omega)} \quad (3.24)$$

$$\text{Phase : } \theta = \arg(X(\omega)) = \tan^{-1} \left(\frac{b(\omega)}{a(\omega)} \right) \quad (3.25)$$

The Fourier transform represents a conversion from the time domain to the frequency domain. The magnitude and phase representations produce real numbers. These values can be plotted with respect to frequency in order to visualize the frequency content of a signal. The inverse Fourier transform can be computed from the magnitude and phase to reconstruct the signal back in the time domain. As a result, the original time-domain signal is reconstructed from the scaled and shifted sinusoids at each frequency, as follows:

$$x(t) = \int_{-\infty}^{\infty} |X(\omega)| \cos(\omega t + \theta(\omega)) d\omega \quad (3.26)$$

The discrete Fourier transform (DFT) for digital signal can be efficiently implemented using a computer via the Fast Fourier transform (FFT) algorithm, which is the discrete-time equivalent of the continuous Fourier transform. The FFT represent the frequency spectrum of a digital signal with a frequency resolution of sampling rate, where the FFT point is a selectable scalar that must be greater than or equal to the length of the signal. The FFT takes an N sample digital signal and produces N frequency samples uniformly spaced over a frequency range of \pm sampling rate/2, thus making it a one-to-one transformation that incurs no loss of information. These frequency domain samples are often called a frequency bin and are the digital equivalent of the result of a continuous Fourier transform with the frequency resolution specified. The FFT will produce complex values that can be converted to magnitude and phase in equation (3.24) and equation (3.25). The FFT spectrum of a real signal has symmetry such that only half of the bins are unique which is from zero to +sampling rate/2. The bins from zero to $-\text{sampling rate}/2$ are the mirror image of the positive bins. Thus, for an N sample real signal, there are $N/2$ unique frequency bins from zero to sampling rate/2. One way to decompose the FFT is based on radix-2 as shown in Fig. 3.27 method with the following procedures:

- The sequence $x(n)$ would need be rearranged in bit-reversed procedure.
- An N -point DFT, where $N = 2^m$, is computed through m stages of computation, with each having its own input and output. The output from particular stages is the input to the next stage. The input to the first stage is $x(n)$ but the order of the indices have been bit-reversed and the output from the last stage is the DFT of $x(n)$.
- In the p -th stage computation, where $1 \leq p \leq m$, the elements of the input sequence are arranged into $N/2^p$ groups, each consists of 2^p elements and 2^{p-1} butterfly computations. The index of the first elements participating in the butterfly computation in the p -th stage is spaced in index 2^{p-1} .
- The R -th butterfly computation of the L -th group in the p -th stage is illustrated in Fig. 3.28.

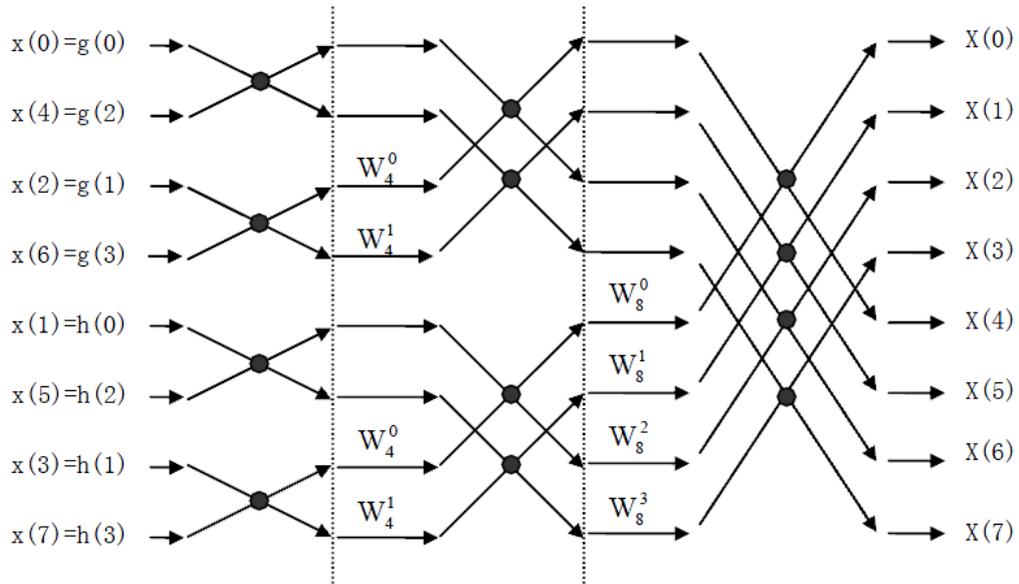


Figure 3.27: Eight point decimation in time of Radix-2 FFT algorithm

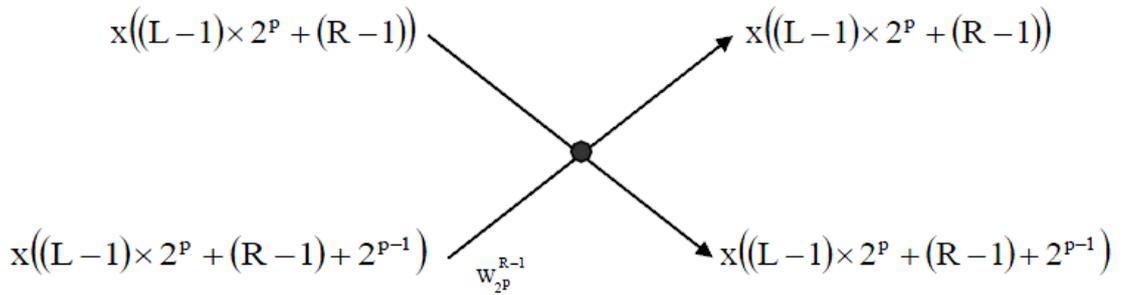


Figure 3.28: Two-point FFT butterfly illustration

Table 3.3: EEG rhythms selection each channel

Description	Value Frequency Range	Index
Delta (δ)	0-3 Hz	Discarded
Theta (θ)	4-7 Hz	1-4
Alpha (α)	8-13 Hz	5-10
Beta (β)	14-30Hz	11-27
Gamma (γ)	Above 30 Hz	Discarded
Total (θ, α, β)	4-30 Hz	27

The FFT is not an approximation of the DFT, but it provides the same result as the DFT with fewer computations required. FFT is applied to the EEG signal in the pre-processing process. This is done by applying a 256 point FFT to every one second width data set and converting into the frequency bands of EEG rhythms as shown in Table 3.3: δ (0-3Hz), θ (4-7Hz), α (8-13Hz) and β (14-30Hz). The δ rhythm is not used due to the low frequency noise such as noise generated from ocular artefact. The γ rhythm is also discarded. As the result, only 27 inputs (θ , α , β) are formed in each channel and 54 inputs for two channels.

3.2.3.3 Classification Algorithm based on Artificial Neural Network (ANN)

The core of the classification algorithm is a model, which typically comprises sets of mathematical equation and/or mapping mechanisms, such as a lookup table with the model to accept the features vector at a given time instant as its input and processes the feature vector to output a set of commands or controls that the application device, such as a wheelchair, can recognize. The model parameters are commonly selected or trained by using a set of training data. Each unit of training data consists of a feature vector and its correct output. Through an iterative procedure, which is called supervised learning, the parameters are adjusted through repeated adjustments until the model translates the features vector into output commands that are as accurate as possible. The accuracy of the model is evaluated with an objective function called a fitness function, the mean-squared error (MSE) between the model output and the correct output. The smaller this error, the more accurate the model.

During the supervised learning process, the features vectors are processed by the model with some initial parameters with known of desired output and the model parameters are then updated based on the fitness function. The process is repeated until stopping criteria are satisfied. Its ability to generalize is tested using an independent set of data called testing data with each unit of testing data consisting of a feature vector and its correct output.

In this chapter, the non-linear artificial neural network (ANN) is discussed. An ANN is a simplified model of a biological neural network where the purpose is not to simulate or replicate actual biological neural network (brain activity), but rather to develop a mental task-based BCI application in the expectation that this will allow the powerful decision-making capabilities of biological neural network to be applied in this case. As a result for BCI classification purpose, the network input consists of the feature vectors and its output consists of the commands sent to the application

3.2.3.3.1 Properties of ANN

Some important properties of ANN include learning, generalization, non-linearity and fault tolerance. ANN has the ability to learn from the examples and learn to produce a particular output when presented with a particular input. This learning process involves modification of the internal parameter of the net; the connection weights to produce overall behaviour correspond to desired behaviour defined by a set of training data. Each data training set consist of the input pattern and a desired output pattern. For the ANN training, the training set is fed to the network and the output it produces is checked. If the output is not as expected, internal weights are modified according to some training algorithm, so as to minimize the difference between the desired and the actual output. The training is then continued with the next training data set and so on; until the network has reached a steady state where no more significant changes to the weights are made, with the aim that the system produces correct outputs for all data in the training set.

The ANN has the capability to produce good outputs even for inputs not encountered during training. The mathematics of ANN defines a mapping from the input space to the output space, which can be described as a vector valued function $y = f(x)$ that transforms the input vector x to an output vector y . Both x and y can have any dimension. The mapping f itself is a combination of mappings performed in parallel by the neurons. The information processing in each neuron is non-linear. As a result, the resulting mapping is non-linear. This non-linear property of ANN has proven to be

important, especially if the physical mechanism that generates the input signal as in EEG is a non-linear signal.

In robust and fault tolerant systems, the systems are kept in function even if parts of them stop working. The good property of fault tolerance in the brain is much due to its massively parallel, distributed structure. ANN try to mimic the natural brain structure, which inherits the robustness the brain comprises.

3.2.3.3.2 Multilayer of ANN

ANN comprises individual units, called neurons, as shown in Fig. 3.29. These neurons consist of a summing node, which sums all the inputs to the neuron and a subsequent activation function to produce the output. The activation function transforms the product of the summing node and can take any form. An ANN is formed by interconnecting individual neurons in a parallel and cascaded fashion that combines the output of the individual neurons to form complex decision boundaries. Each neuron uses a decision hyperplane to convert a weighted sum of its inputs to an output.

Typically ANN has a hidden layer of neurons that form as three layers; input, hidden and output layers. Through training, the hyperplane produced by each neuron in the network partitions the feature space to define the decision boundaries to represent each possible output. Fig. 3.30 illustrates a feed-forward network that has four inputs and two outputs, which represent, for example, four input features and two possible BCI commands. During ANN training, the network processes the training set. After each observation is processed, the output of the network is compared to the correct output label to define the error. Then, the summing nodes of each neuron are updated to reduce the error. This updating process proceeds backward through the network from the output layer to the input layer as the back-propagation method.

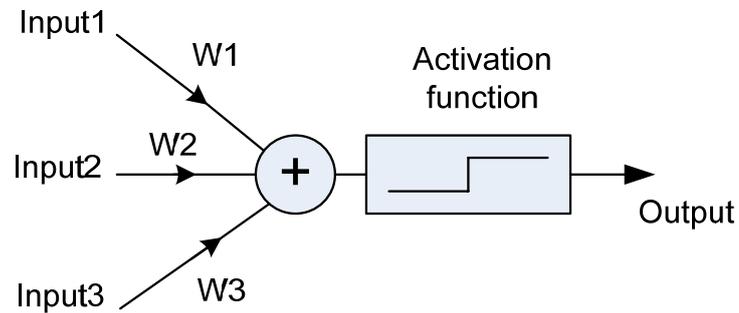


Figure 3.29: Neuron of ANN

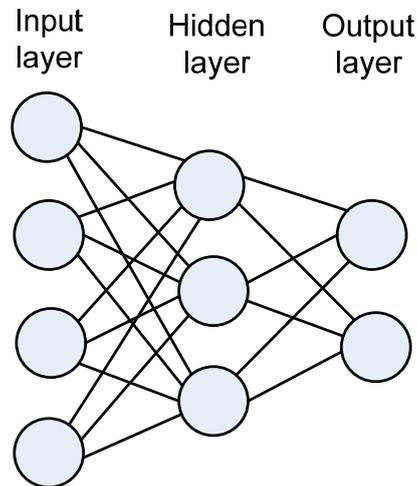


Figure 3.30: The three layers ANN

3.2.3.3.3 Learning of Multilayer ANN

A perfectly trained network will always give the desired output, or will produce zero for the error, no matter what the input pattern is. The error function (E) is a real valued function that, for a fixed input, measures the distance between the actual output (y) and desired output (d) with the following equation:

$$E = \frac{1}{2} \sum_{i=1}^N (y_i - d_i)^2 \quad (3.27)$$

E always has a positive value. The actual outputs and the desired outputs can be arranged as N -dimensional vector \mathbf{y} and \mathbf{d} and the error function can be defined as:

$$E = \frac{1}{2} (\mathbf{y} - \mathbf{d})^T (\mathbf{y} - \mathbf{d}) \quad (3.28)$$

As a result, the problem of training the network is a multi-dimensional, non-linear and unconstrained optimization problem. Standard procedure for solving the problem is based on an iterative searching algorithm to start in some initial point in the landscape, and then repeatedly work downward by taking small steps in well-chosen directions. These directions can be found by making a first order Taylor expansion of E around some initial point w_0 .

$$E(w) = E(w_0) + \nabla E(w_0)^T (w - w_0) \quad (3.29)$$

where the first term in the approximation is a constant. The second term is in fact a dot product which has its minimum when the two vectors are parallel, but with opposite sign.

$$w = w_0 - \eta \nabla E(w_0) \quad (3.30)$$

where η is a small constant. Consequently, the value of the error function can be decreased by adjusting the weight according to equation (3.29), provided the gradient is non-zero, and the step-length η is small enough. It is simply done by recalculating equation (3.30) from the new point until the solution is good enough. This method is known as the steepest decent algorithm.

3.2.3.3.4 The Back-propagation Algorithm of ANN

As the vector ∇E is a collection of partial derivatives arranged as a column vector, focusing on a single weight in the system corresponds to extracting one row from the weight update formula.

$$w_{ij} \leftarrow w_{ij} - \eta \frac{\partial E}{\partial w_{ij}} \quad (3.31)$$

This is the update rule for the weight that connects node i and node j . The output from node i , is as follows:

$$y_i = f\left(\sum_{k \rightarrow i} w_{ki} y_k\right) = f(z_i) \quad (3.32)$$

where the sum is taken over all nodes k that connect to node i . The partial derivative can be rewritten by the chain rule and this develops:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial z_i} \frac{\partial z_i}{\partial w_{ij}} = \frac{\partial E}{\partial z_j} y_i = y_i (y_j - d_j) \frac{\partial f(z_j)}{\partial z_j} = y_i \delta_j \quad (3.33)$$

Note that E is differentiable if and only if the activation function f is differentiable. If the transfer function is a logistic sigmoid function, the last derivative yields:

$$\frac{\partial f}{\partial z_i} = \frac{\partial}{\partial z_i} \left(\frac{1}{1 + e^{-\beta z_i}} \right) = \frac{\beta}{(1 + e^{-\beta z_i})^2} e^{-\beta z_i} = f(1 - f) \quad (3.34)$$

Combining these results give the final update rule for the nodes in the output layer.

$$w_{ij} = w_{ij} - \eta\beta y_j^2(1 - y_j)(y_j - d_j) \quad (3.35)$$

This formula is only applicable to nodes for which the desired output value is d . For the nodes in the hidden layers use:

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial z_k} \frac{\partial z_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial z_k} \frac{\partial}{\partial y_j} \sum_i w_{jk} y_i = \sum_k \frac{\partial E}{\partial z_k} w_{jk} = \sum_k \delta_k w_{jk} \quad (3.36)$$

Therefore,

$$\delta_j = \frac{\partial f(z_j)}{\partial z_j} \sum_k \delta_k w_{jk} \quad (3.37)$$

The update rule for a hidden node can be obtained as follows:

$$w_{ij} \leftarrow w_{ij} - \eta\beta y_j^2(1 - y_j) \sum_k \delta_k w_{jk} \quad (3.38)$$

The last sum is taken over all nodes k that are connected to the output of the node j . This means the update of the weights has to be performed backwards through the network. Before updating the weights in the node j , the error term δ_k needs to be calculated for all its successors as the back-propagation rule.

The steps for the back-propagation ANN can be summarized as follows:

- (1) Initialize \mathbf{w} by setting all weight to small random numbers
- (2) Present a training pattern to the input layer and calculate the outputs from all nodes in the network in a feed-forward way with the following:

$$y = \frac{1}{1 + e^{-\beta \sum_{i=0}^N w_i x_i}} \quad (3.39)$$

(3) Adjust the weights in the output layer:

$$w_{ij}^{(p+1)} = w_{ij}^{(p)} - \eta\beta y_j^2(1 - y_j)(y_j - d_j) + \alpha(w_{ij}^{(p)} - w_{ij}^{(p-1)}) \quad (3.40)$$

where η denotes the learning rate, β denotes the spread of the threshold function and α is the momentum term. The actual output from node j is denoted y_j and d_j is the desired output.

(4) Apply the process backwards through the network and update the rest of the weights according to the back-propagation rule as follows:

$$w_{ij}^{(p+1)} = w_{ij}^{(p)} - \eta\beta y_j^2(1 - y_j) \sum_k \delta_k w_{jk} + \alpha(w_{ij}^{(p)} - w_{ij}^{(p-1)}) \quad (3.41)$$

The error term δ_k is the error of the successor node k for output nodes and is given by the following:

$$\delta_k = \beta y_k(1 - y_k)(y_k - d_k) \quad (3.42)$$

For the nodes in the hidden layers the error term is given by:

$$\delta_k = \beta y_k(1 - y_k) \sum_m \delta_m w_{jm} \quad (3.43)$$

(5) The process is repeated from step 2 until the result is good enough.

3.2.3.3.5 The Levenberg-Marquardt Algorithm of ANN

While the back-propagation is a steepest descent algorithm, the Levenberg-Marquardt algorithm is an approximation to the Newton's method (Marquardt 1963). If function $V(x)$ is used to minimize with respect to the parameter vector x :

$$\Delta x = -[\nabla^2 V(x)]^{-1} \nabla V(x) \quad (3.44)$$

where $\nabla^2 V(x)$ denotes the Hessian matrix and $\nabla V(x)$ denotes the gradient. Assume that the $V(x)$ is a sum of square function,

$$V(x) = \sum_{i=1}^N e_i^2(x) \quad (3.45)$$

as a result, it can be shown as below:

$$\nabla V(x) = J^T(x)e(x) \quad (3.46)$$

$$\Delta^2 V(x) = J^T(x)J(x) + S(x) \quad (3.47)$$

where $J(x)$ is the Jacobian matrix:

$$J(x) = \begin{bmatrix} \frac{\partial e_1(x)}{\partial x_1} & \frac{\partial e_1(x)}{\partial x_2} & \dots & \frac{\partial e_1(x)}{\partial x_n} \\ \frac{\partial e_2(x)}{\partial x_1} & \frac{\partial e_2(x)}{\partial x_2} & \dots & \frac{\partial e_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_N(x)}{\partial x_1} & \frac{\partial e_N(x)}{\partial x_2} & \dots & \frac{\partial e_N(x)}{\partial x_n} \end{bmatrix} \quad (3.48)$$

and,

$$S(x) = \sum_{i=1}^N e_i(x)\nabla^2 e_i(x) \quad (3.49)$$

It is assumed that $S(x) \approx 0$ for the Gauss-Newton method and the update of equation (3.44) is as follows:

$$\Delta x = [J^T(x)J(x)]^{-1} J^T(x)e(x) \quad (3.50)$$

The Marquardt-Levenberg modification to the Gauss-Newton method is shown below:

$$\Delta x = [J^T(x)J(x) + \mu I]^{-1} J^T(x)e(x) \quad (3.51)$$

The parameter μ is multiplied by some factor β whenever a step would result in an increased $V(x)$. When a step reduces $V(x)$, μ is divided by β . When μ has a large value, the algorithm reaches its steepest descent and becomes Gauss-Newton when μ has small value. The Levenberg-Marquardt algorithm is a modification to the Gauss-Newton method. The key to this algorithm is the computation of the Jacobian matrix. In the ANN mapping problem, the term in the Jacobian matrix can be computed using the modification to the back-propagation algorithm. The element of the Jacobian matrix that is needed for the Levenberg-Marquardt algorithm is needed to calculate the terms below:

$$\frac{\partial e_q(m)}{\partial w^k(i, j)} \quad (3.52)$$

These terms can be actually computed using the standard back-propagation algorithm with modification at the final layer as below:

$$\Delta^M = -F^M(n^M) \quad (3.53)$$

Each column of the matrix in equation (3.53) is a sensitivity vector which must be back-propagated through the network to produce one row of the Jacobian.

The Levenberg-Marquardt algorithm was compared with the conjugate gradient algorithm and with the back-propagation method as discussed in (Hagan & Menhaj 1994). The result showed that the Levenberg-Marquardt algorithm is very efficient when training the network, and in some cases, the Levenberg-Marquardt algorithm is easily converged compared to the original back-propagation method. As a result, the Levenberg-Marquardt algorithm is used for the ANN training in this chapter.

3.2.3.3.6 Early Stopping for Generalization Performance

As the aim of the ANN training is to minimize the training error, it is prone to over-fitting or over-training. This means that during training, the network will reach a state where the generalisation does not improve anymore. If the ANN training proceeds past this point, the network will begin to overfit with the result of losing the ability to generalise.

One of the methods of dealing with over-fitting is the early stopping method. It is to stop the training when the generalisation error has reached its minimum. To do this, the data set is divided into a training set, validation set and testing set. The training set and validation set is fed to the ANN training. The error on the training of the training set and validation set are monitored. This error will normally decrease during the initial phase of the training for both training cycle error and validation cycle error. However, when the ANN begins to overfit the data, the cycle error on the validation set will typically begin to rise. The training of ANN is stopped when the cycle error of the validation set increases for a pre-specified number of iterations. The testing set is used to measure the performance of the ANN.

3.2.4 Main Embedded System Controller

An embedded system will be used to create a real-time BCI system, as shown in Fig. 3.31. The signal processing algorithm, which includes pre-processing, feature

extraction and classification algorithms at the final end will be implemented in the embedded platform. The embedded system module has MOD5213, which is ColdFire-MCF5213 microcontroller (Freescale Semiconductor), which operates at 66MHz. The system has a flash capacity 256 Kbytes and RAM of 32 Kbytes. The system also has capability of the real-time operating system μ C/OS. An additional special register multiplier and accumulator (MAC) is used to speed up the calculation.

The firmware of the real-time BCI system is illustrated in Fig. 3.32 that provides signal processing and real-time mental task classification. Real-time operating system μ C/OS is configured on several tasks simultaneously. The system captures packets via the SPI bus from the wireless receiver and saves the data into first-in-first-out (FIFO) input buffer. Fig. 3.33 shows the real-time data transfer from the wireless receiver. Other μ C/OS task is also created for signal processing process, which includes pre-processing, features extraction and classification methods. A moving window of one second of data from the FIFO buffer is applied for response adjustment of the system. A variable is created to adjust the gap of the moving window. This is followed by signal pre-processing, moving average filter to smooth the signal by averaging a number of points of input to produce the output signal.

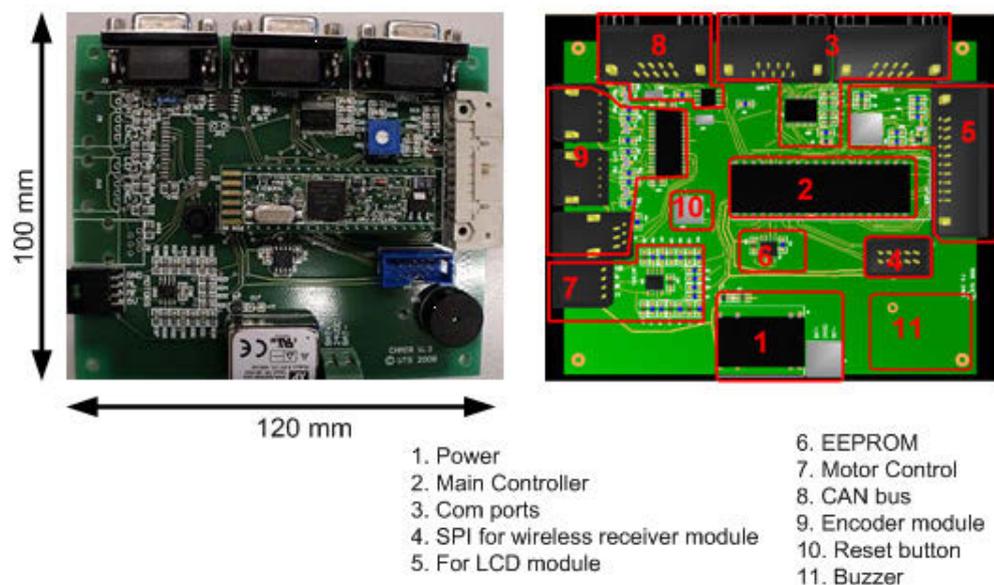


Figure 3.31: Main embedded system controller of the BCI

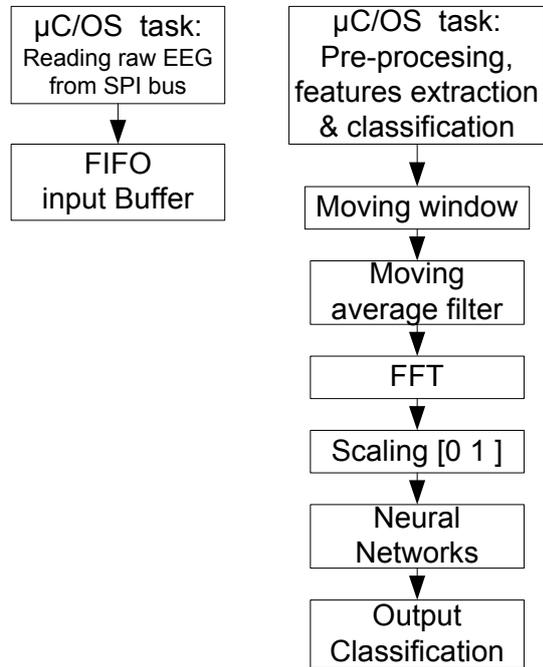


Figure 3.32: Block diagram firmware of main embedded system

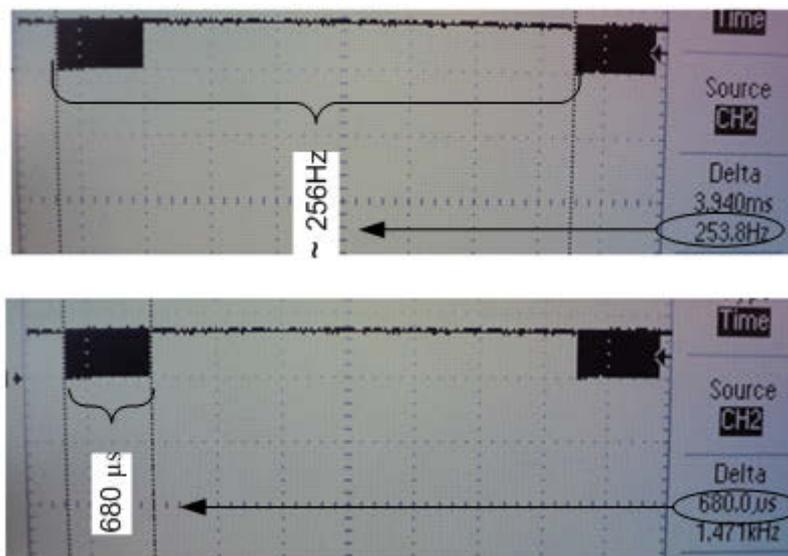


Figure 3.33: Real-time data receiving at 256 Hz

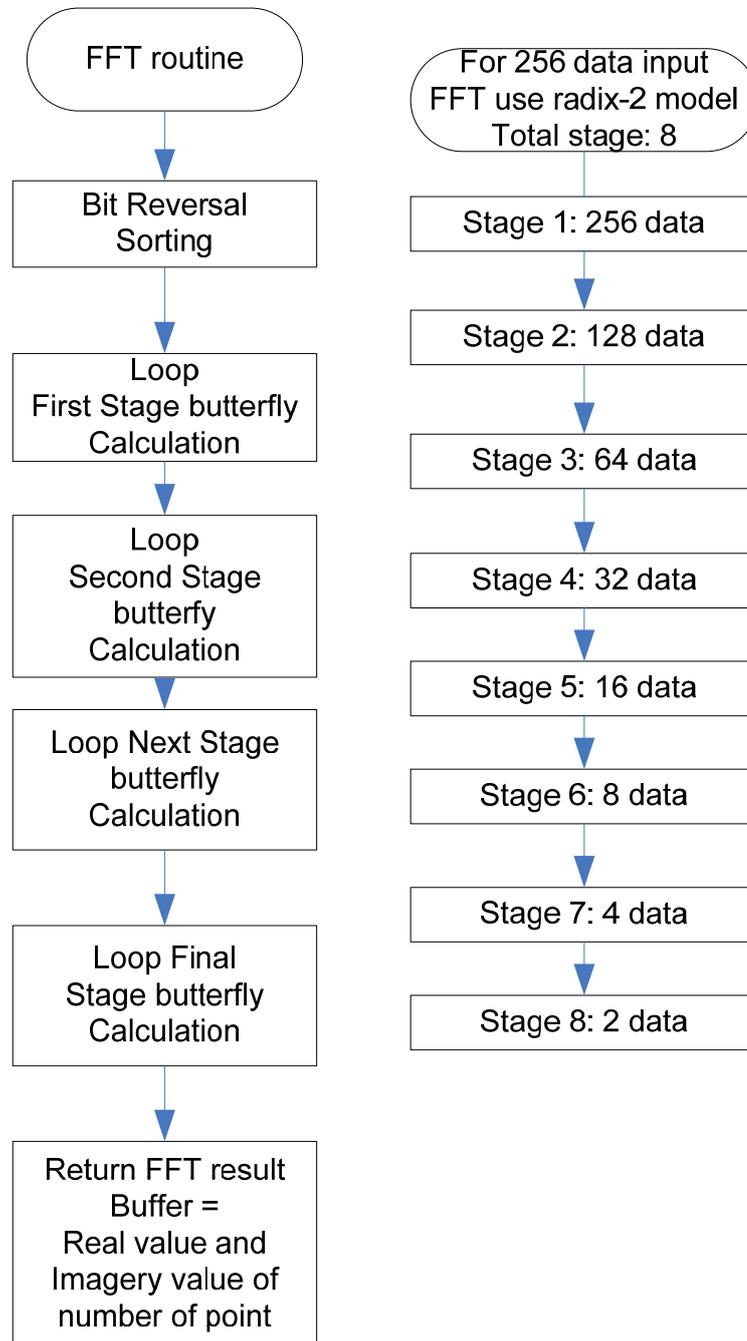


Figure 3.34: Flowchart FFT Radix-2 of 256 point in assembly routine

Fast Fourier Transform (FFT) as a features extraction algorithm is an efficient algorithm for computing the discrete Fourier transform (DFT) which converts the EEG data into the frequency bands and is implemented in the microcontroller of the main embedded system. FFT radix-2 of 256 point is implemented in an assembler routine with a multiplication accumulator unit (MAC) as a special register used to speed up the multiplication operation, as shown in Fig. 3.34. This is started by the bit reversal sorting routine and followed by the looping routine of the first stage of the FFT butterfly calculation. As the 256 point of data input is used, there will be a total of 8 stages. Finally, after all stages are calculated, the final FFT result is placed in the buffer, which contains real and imagery values. Comparison of the FFT routine on the embedded system and MATLAB function on PC are presented in Fig. 3.35, which shows a comparable result. The time delay for FFT based on embedded system is only 520 μ s.

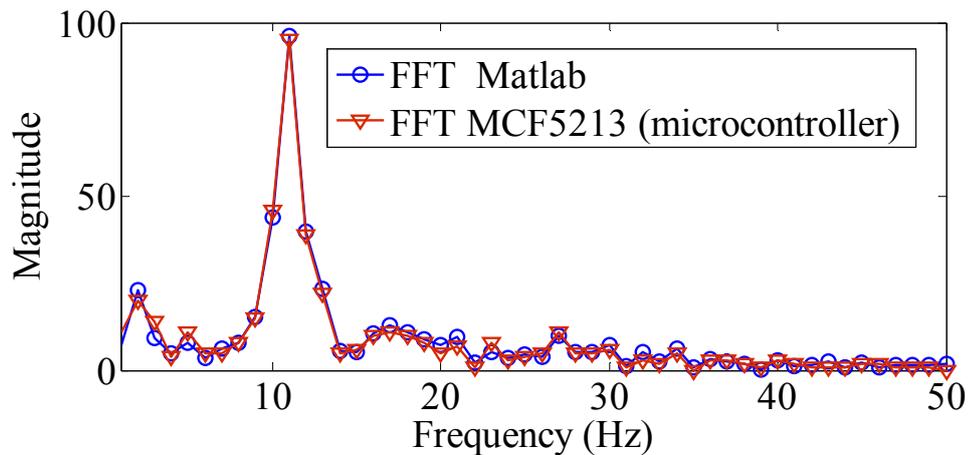


Figure 3.35: Real-time data receiving at 256 Hz

An ANN is applied on the embedded system for the real-time classification. Prior to the ANN, features need to be scaled in value between one and zero with the function as follows:

$$X^* = \frac{(X - X_{\min})}{(X_{\max} - X_{\min})} \quad (3.54)$$

where X^* is the result of features after scaling, X is the input features, X_{\min} is the minimum value of input features and X_{\max} is the maximum value of the features.

ANN with feed forward algorithm is applied for the classification algorithm with constant of best weight value as a result of the ANN training process. The non-linear activation function is based on sigmoid function with the formula as follows:

$$\text{Sig}(x) = \frac{1}{1 + e^{-x}} \quad (3.55)$$

where $\text{Sig}(x)$ is the output of sigmoid function, x is the input, e is the exponential function. In the embedded system to formulate the sigmoid function (3.55), a standard math library needs to be used. The delay time to execute sigmoid function based on standard math library is measured around 360 μs . To speed the calculation by using the approximation of sigmoid function, which is based on Taylor 7th series function, calculate e^x as below:

$$\text{Exp_Taylor} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!} + \frac{x^6}{6!} + \frac{x^7}{7!} \quad (3.56)$$

where Exp_Taylor is the approximation of the exponential function based on the Taylor series, x is the input value.

The approximation of sigmoid function consists of two functions which apply for inputs less than zero and inputs above zero, as shown in Table 3.4. The delay time of each execution is reduced at around 1.84 μs with the tolerance around 0.1%. The original sigmoid function on using Matlab on PC is around 360 μs . Comparison delay time and tolerance of original and approximation of sigmoid function is presented in Table 3.5.

Table 3.4: Sigmoid function of Taylor series approximation

Interval	Taylor series approximation of sigmoid function
Input < 0	$output = (0.5 - ((1.0/(1 + (1/ExpTaylor(input \times (-1)))))) - 0.5)$ <i>ExpTaylor</i> is exponential function from Taylor series approximation from equation (3.56)
Input ≥ 0	$output = 1.0 / (1 + (1 / ExpTaylor(input)))$

Table 3.5: Comparison sigmoid function and approximation

Function	Tolerance	Delay
Original sigmoid function	0	360 μs
Taylor series approximation	0.1%	1.84 μs

3.3 Performance Measurement

3.3.1 Classification Accuracy

The classification accuracy is the most widely used evaluation criteria in BCI research especially for the classifier with more than two outputs or multi-class classification. Classification accuracy refers to the probability of performing a correct classification. It can be calculated by dividing the number of correct classifications by the total number of trials, as follows:

$$ACC = \frac{\sum C_{i,i}}{N} \quad (3.57)$$

where ACC is the classification accuracy, $C_{i,i}$ is the i th diagonal element of the confusion matrix, and N is the total number of trials. The error rate or misclassification rate, $ERR = 1 - ACC$ is the probability of making an incorrect classification.

3.3.2 Bit Rate and Information Transfer Rate (ITR)

Bit rate refers to the amount of reliable information received or the amount of information communicated per unit time (Wolpaw et al. 1998), which can be calculated as follows:

$$B = \log_2 N + P \log_2 P + (1 - P) \log_2 \left[\frac{1 - P}{N - 1} \right] \quad (3.58)$$

$$ITR = V B \quad (3.59)$$

where B refers to the bit rate (bits/trial), N refers to the number of mental task and P refers to the classification accuracy. To attain the ITR (bit/minutes), multiply B by the number of selections per unit time or the application speed (V).

3.4 Data Collection and Experiment

This study was approved by the Human Research Ethics Committee of University of Technology, Sydney. Mental tasks data were collected from six healthy participants with ages between 25 and 35 using the developed BCI prototype. The relevant mental tasks used are as follows:

- Mental arithmetic task: participants imagined mentally solving non-trivial multiplication problem such as 54 times 47.
- Mental figure rotation task: participants were asked to imagine a cube being rolled forward.
- Mental letter composing: participants composed a simple letter in their mind.
- Mental counting task: participants mentally visualized appearing and disappearing counting numbers on a blackboard.

- Eyes closed and open task: This task is used to measure alpha wave production. Participants were asked to relax while performing open and closed eyes action.

Gold electrodes were positioned using a bipolar montage at P3-T4 for channel 1 (CH1) and O2-T3 for channel 2 (CH2) as shown in Fig. 7. Location CZ was used for the reference electrode. This placement is based on the international 10-20 system. EEG gel was applied to keep the impedance level low and a better electrical contact. Unnecessary movements and eye blinks were kept as minimal as possible during data collection in each session. Each mental task was measured for a total of 6 sessions, with each session lasting 13 seconds. The first 3 seconds of data were discarded for preparation time, and the remaining 10 seconds were used for the further signal processing.

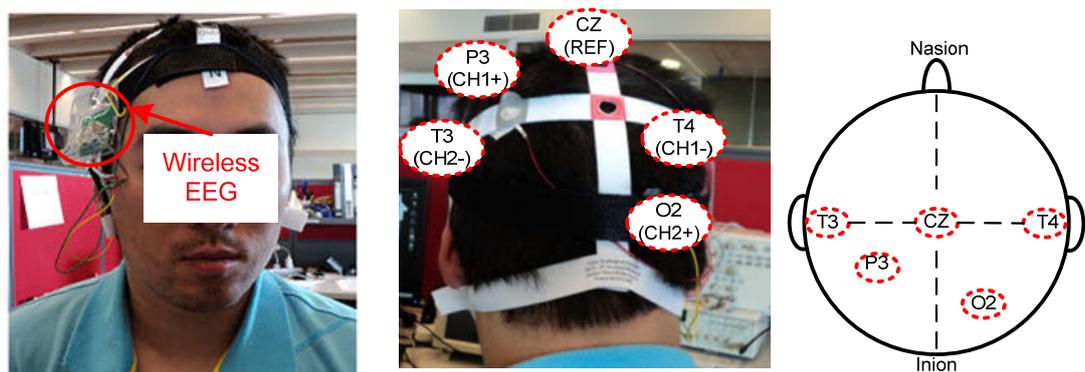


Figure 3.36: Wireless EEG with two channels electrodes placement

A data pre-processor extracts one second window of EEG channel readings every quarter second for processing by the features extraction algorithm. As the result, a one second window data set with some overlap between data sets is obtained every 0.25 sec. A total of 10 seconds data window resulted in 37 overlapping segments. To further improve signal quality, the data sets are then passed through two digital signal processing (DSP) filters: first, a moving average filter of 3 data samples width to

smooth the signal then a Butterworth band-pass filter with the bandwidth 0.5Hz to 100Hz. A Fast Fourier Transform (FFT) is used for features extraction method. This is done by applying a 256 point FFT to every one second width data set and converting into the frequency bands of EEG rhythms as shown in Table 1: δ (0-3Hz), θ (4-7Hz), α (8-13Hz) and β (14-30Hz), The δ rhythm is not used due to the low frequency noise such as noise generated from ocular artifact. The γ rhythm is also discarded. As the result, only 27 inputs (θ , α , β) are formed in each channel and 54 inputs for two channels.

3.5 Experimental Results and Discussions

3.5.1 Measurement for the Specifications of Wireless EEG

An initial testing used the EEG simulator (Netech-Minisim 300) to inject a sinusoidal signal with different adjustable frequency and amplitude. This is started by checking the gain measurement as shown in Fig.3.37, with steps as follows:

- Connect the EEG simulator to the wireless EEG amplifier. Input voltage (V_{in}) and frequency of the signal can be changed from the EEG simulator.
- Connect the output signal of the Wireless EEG amplifier to the oscilloscope for the output voltage (V_{out}).
- Determine the gain on the measurement ($gain\ measured = V_{out} / V_{in}$)

The CMRR is measured, as in Fig. 3.38, with the following steps:

- Connect the equipment, as shown in the Fig. 3.38, by connecting two inputs wireless EEG amplifier together with the signal generator and output to the oscilloscope.
- Adjust the signal general to produce generator peak to peak amplitude to be set to full common mode voltage at 50 Hz.
- Measure the corresponding output voltage (V_{out})
- Calculate the CMMR using equations: (3.2), (3.4) and (3.7).

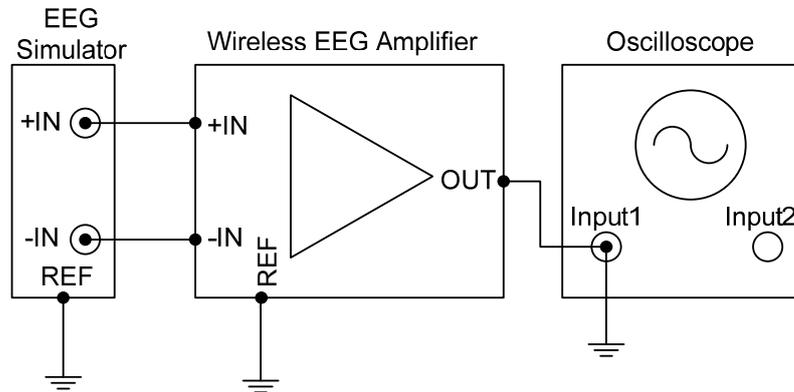


Figure 3.37: Wireless EEG gain testing with simulator

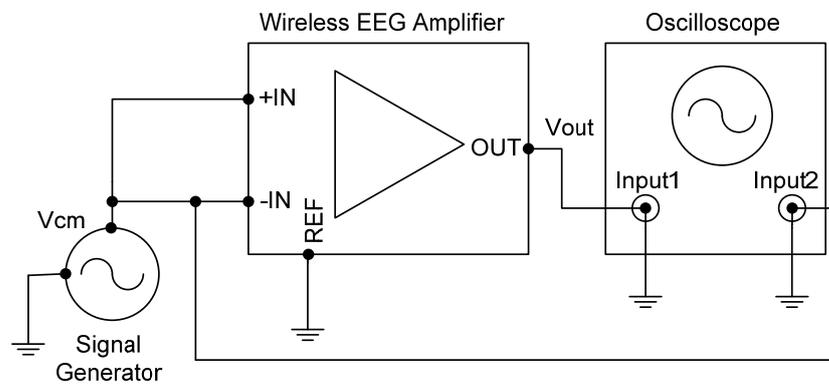


Figure 3.38: CMRR measurement of Wireless EEG amplifier

Table 3.6: Wireless EEG V1 gain testing

Input Voltage V_{in} (μV)	Input Frequency (Hz)	Gain	Output Voltage V_{out} (μV)	Output Frequency (Hz)	Gain Measured V_{out}/V_{in}
500	5	1000	5.13×10^5	4.902	1026
100	5	1000	9.41×10^4	4.902	941
50	5	1000	5.0×10^4	4.902	1000
30	5	1000	2.94×10^4	4.902	980

Table 3.7: CMRR measurement for wireless EEG V1

V_{cm} (mV)	V_{out} (mV)	A	CMRR	CMRR(dB)
153	92	1000	1663.04	64.42
25.9	16.7	1000	1550.90	63.81
32.3	22	1000	1468.18	63.34
21.4	13.6	1000	1573.53	63.94
153	88	1000	1738.64	64.80
Average				64.06

Table 3.8: CMRR measurement without DC filter for wireless EEG V1

V_{cm} (mV)	V_{out} (mV)	A	CMRR	CMRR(dB)
368	4.6	1000	80000.00	98.06
476	13.4	1000	35522.39	91.01
399	4.30	1000	92790.70	99.35
427	6.90	1000	61884.06	95.83
592	12.20	1000	48524.59	93.72
Average				95.59

Wireless EEG V1 is based on the AC-Coupled In-Amp, which composes the DC filter in front of the In-Amp, and an additional anti-aliasing filter is connected after the In-Amp module, before connecting to the microcontroller. Table 3.6, shows the results for wireless EEG V1 testing that the gain measurements are proportional to the applied gain on the different injected input from the simulator. The result of output measurement is shown in Table 3.6, which has a value consistent to the gain amplification setup. The CMRR measured at around 64 dB as shown in Table 3.7, which is lower than the requirement (> 80 dB). By removing the DC filter, which is an RC circuit, the CMRR improved to around 95 dB as shown in Table 3.8. The reason for the low CMRR is the mismatched impedance of the RC filter on the two inputs of the

In-Amp. The value of the capacitors and resistors need to be exactly balanced to each other. A small difference would degrade the CMRR. In reality, it is an issue to find such components with the exact same value. On the other hand, the DC filter is important to include with the amplifier, otherwise, the amplifier will be saturated due the direct amplification of the DC offset of the EEG electrode.

The Wireless EEG V2 is based on the DC-coupled In-Amp, which composes two stages of amplification. The first stage uses the In-Amp at low gain at $10\times$ in such a way that the electrode dc offset does not saturate the In-Amp. The DC filter is connected to the output of the In-Amp. Once the dc component is removed, the signal is gained up again with another amplifier. An additional anti-aliasing filter is added to the output of the second stage amplifier before being connected to the microcontroller. The result of the testing of the overall gain is shown in Table 3.9, in which the gain measured is relevant to the original calculated gain. The measurement of the CMRR for Wireless EEG V2 is shown in Table 3.10 with the average value at 95.14 dB. This CMRR has satisfied the requirement of the EEG design (>80 dB). As a result, the Wireless EEG V2, with size of the core module of the wireless EEG being small, $36 \times 36 \text{ mm}^2$ is the choicen design for this study.

Table 3.9: Wireless EEG V2 gain testing

Input Voltage V_{in} (μV)	Input Frequency (Hz)	Gain	Output Voltage V_{out} (μV)	Output Frequency (Hz)	Gain Measured V_{out}/V_{in}
1000	5	1000	1.02×10^6	5.038	1020
100	5	1000	9.34×10^4	5.038	934
30	5	1000	3.88×10^4	5.038	1293.33

Table 3.10: CMRR measurement for wireless EEG V2

V_{cm} (mV)	V_{out} (mV)	A	CMRR	CMRR(dB)
2227	37.3	1000	59705.09	95.52
1314	24.2	1000	54297.52	94.70
775	15.2	1000	50986.84	94.15
2209	34.2	1000	64590.64	96.20
Average				95.14

Table 3.11: Measurement of wireless EEG V2 specifications

Specifications	Value
Dimension	36 mm x 36 mm
Number of channels	Two
Number of electrodes	Five
Sampling rate	256 Hz
ADC bits	12 bits
CMRR	>95 dB
Noise	3.5 μV_{RMS}
Gain	Adjustable up to 10000 times
Current consumption	5.03 mA
Battery	CR2032 (190 -225mAh)
Battery life	38 -45 hours for continuous operation

The gain can be set up to 10^4 times. Current consumption of the wireless EEG V2 with the power saving mode programming was around 5 mA if the system is operated continuously. The noise measurement with the input shored was around $3.5\mu V_{RMS}$. The detailed measurement of the specifications is shown in Table 3.11.

3.5.2 Testing Wireless EEG

The developed wireless EEG V2 was also compared to measure the EEG and ECG signals with other EEG systems that are available in the market. Fig. 3.39 shows the measurement of the wireless EEG V2 with its supported software on PC to measure the given signal from the EEG simulator, Minisim 330-Netech (Netech). The input EEG simulator has a sinusoid signal with a frequency at 2 Hz with the amplitude in the EEG range. The result shows the system recognises the adequate sinusoid signal and its frequency. The compared EEG system that is available on the market uses the Procomp2 from Thought Technology. Fig. 3.40 shows the software for Procomp2 to detect the same test signal.

The developed Wireless EEG is also tested to measure another bio-potential signal; ECG for the heart signals measurement that has a unique signal (PQRST waves). As understood, the range of the ECG signal is in terms of millivolt, while the EEG signal is in the microvolt level. Moreover, the bandwidth of the ECG signal is inside of the EEG signal. At a result, the EEG instrument should be able to measure the normal EEG signal on the human body. For this reason, the ECG system, Compumedics-Siesta that is available on the market is used for the comparison. Fig. 3.42 shows the ECG setup on the human body for the ECG measurement.

The ECG (Lead II) measurement on the human body using Wireless EEG V2 is shown in Fig. 3.42. This is compared with the same position measurement using the ECG machine from the market, Compumedic-Siesta as shown in Fig. 3.43. The result shows a clear, comparable ECG signal, PQRST waves between the Wireless EEG system and the Compumedics-Siesta system.

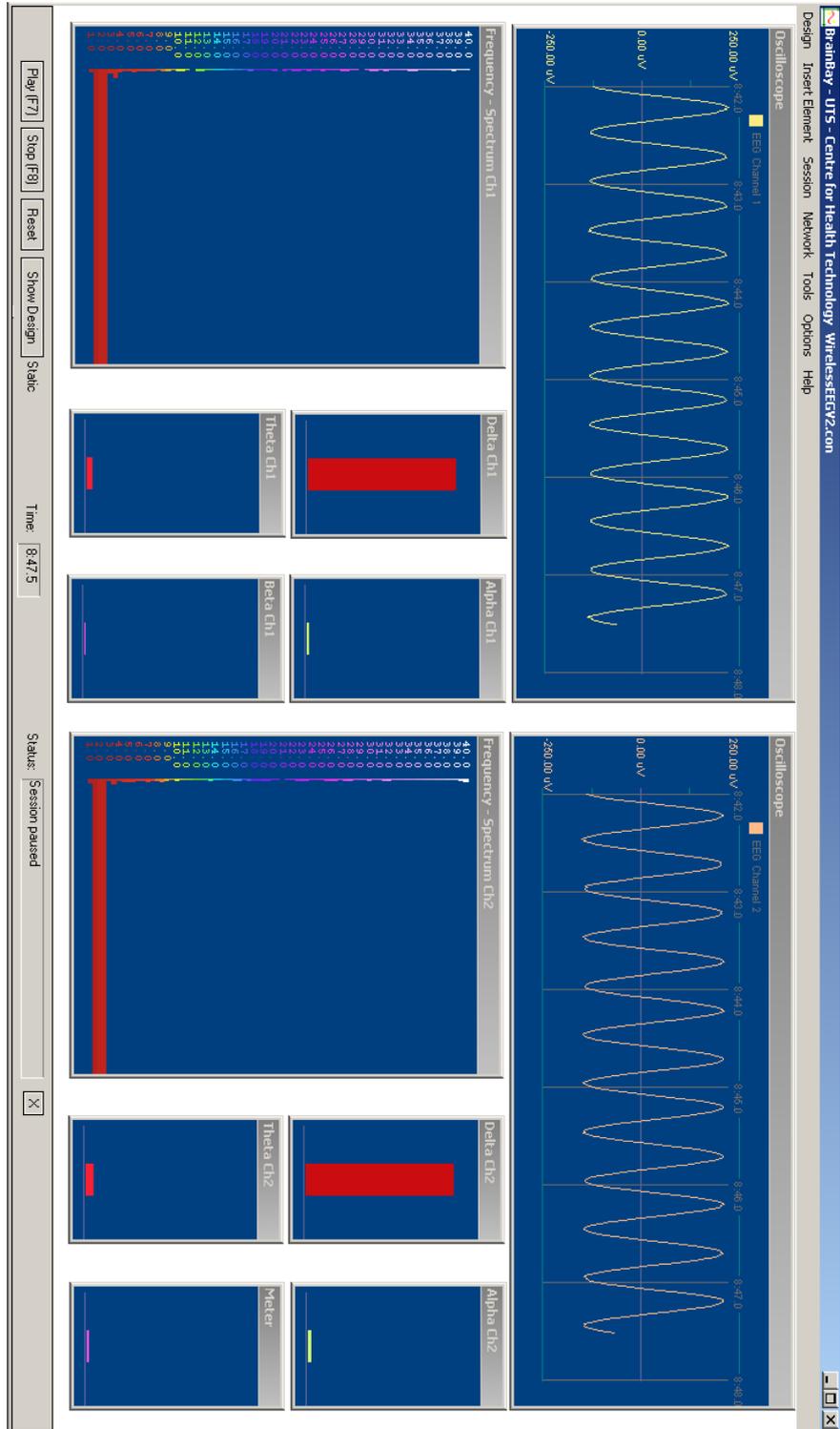


Figure 3.39: Testing Wireless EEG with its Software on PC connected to EEG simulator

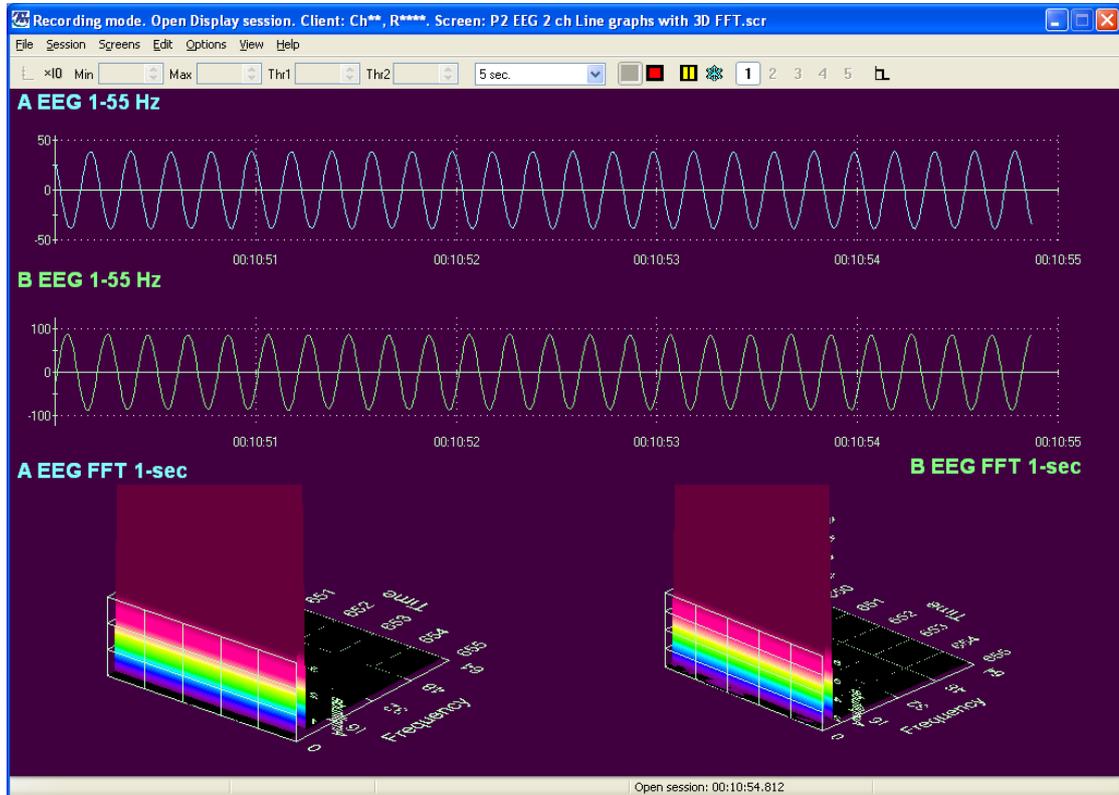


Figure 3.40: Testing Procomp2 (Thought Technology) with its software on PC connected to EEG simulator

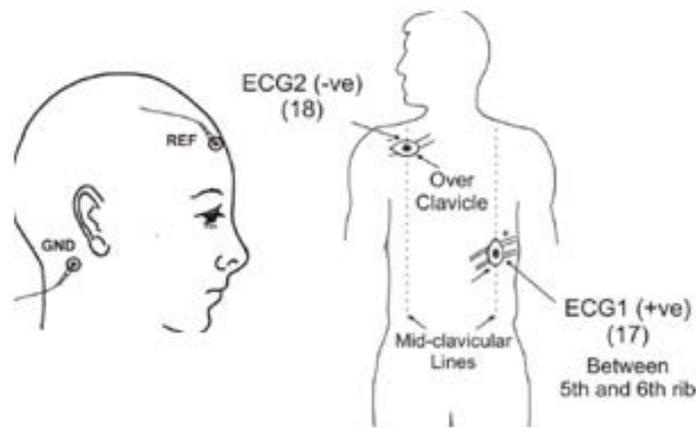


Figure 3.41: Electrode placements for ECG measurement

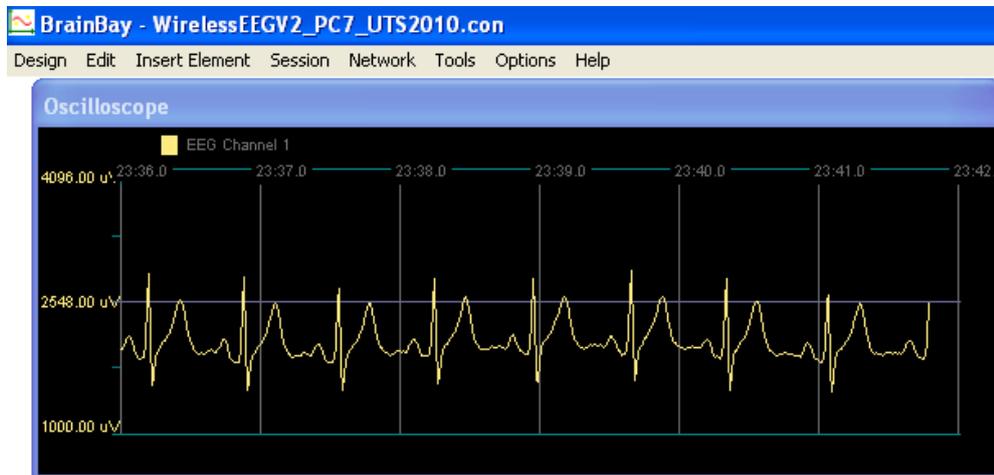


Figure 3.42: Wireless EEG V2 testing for ECG measurement on body



Figure 3.43: Compumedics-Siesta system with ECG measurement on body

3.5.3 Complete Hardware of Microcontroller-based BCI system for Mental Task Classifications

The complete instrumentation for the microcontroller-based BCI system that uses the development Wireless EEG V2 is shown in Fig. 3.44, which is composed of a wireless EEG V2 with the 2 channels configuration and gold electrodes, wireless receiver and main embedded system controller for the real-time computational intelligence.

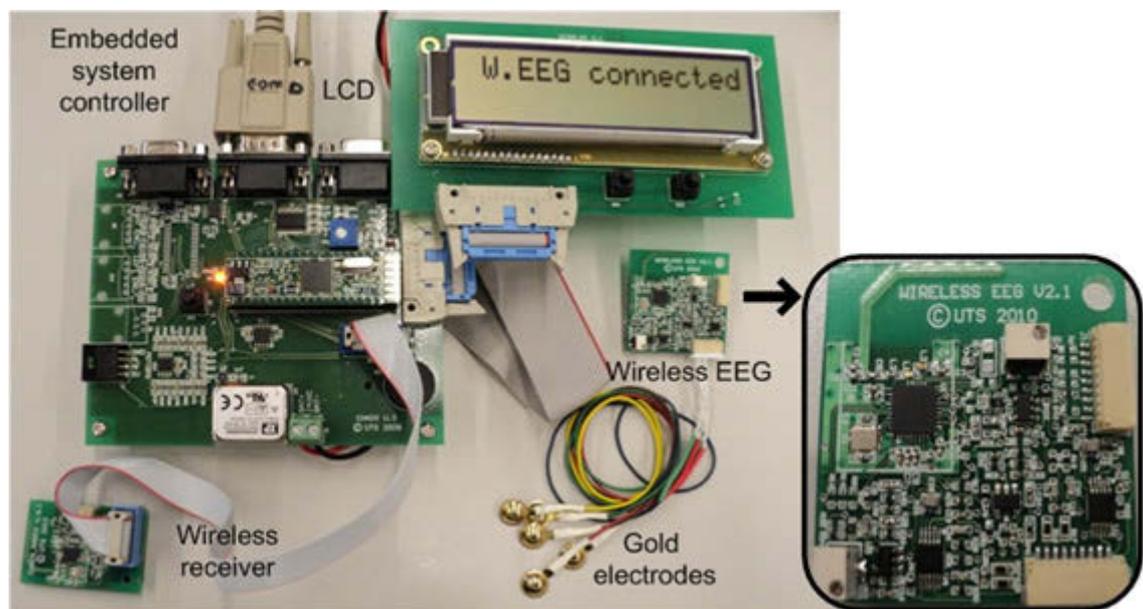


Figure 3.44: Complete system of microcontroller

3.5.4 Results of ANN Classifications

It is well known that there is an increase of alpha waves on the visual cortex, which is in the occipital lobe for most people during closed eyes action. Individuals with a disability issue could use the eyes closed action as the mind switch to provide integrated control with other BCI technologies (Craig et al. 2002). Therefore, it is crucial for the developed BCI system to detect the dominant alpha wave (8-13 Hz) during eyes closed.

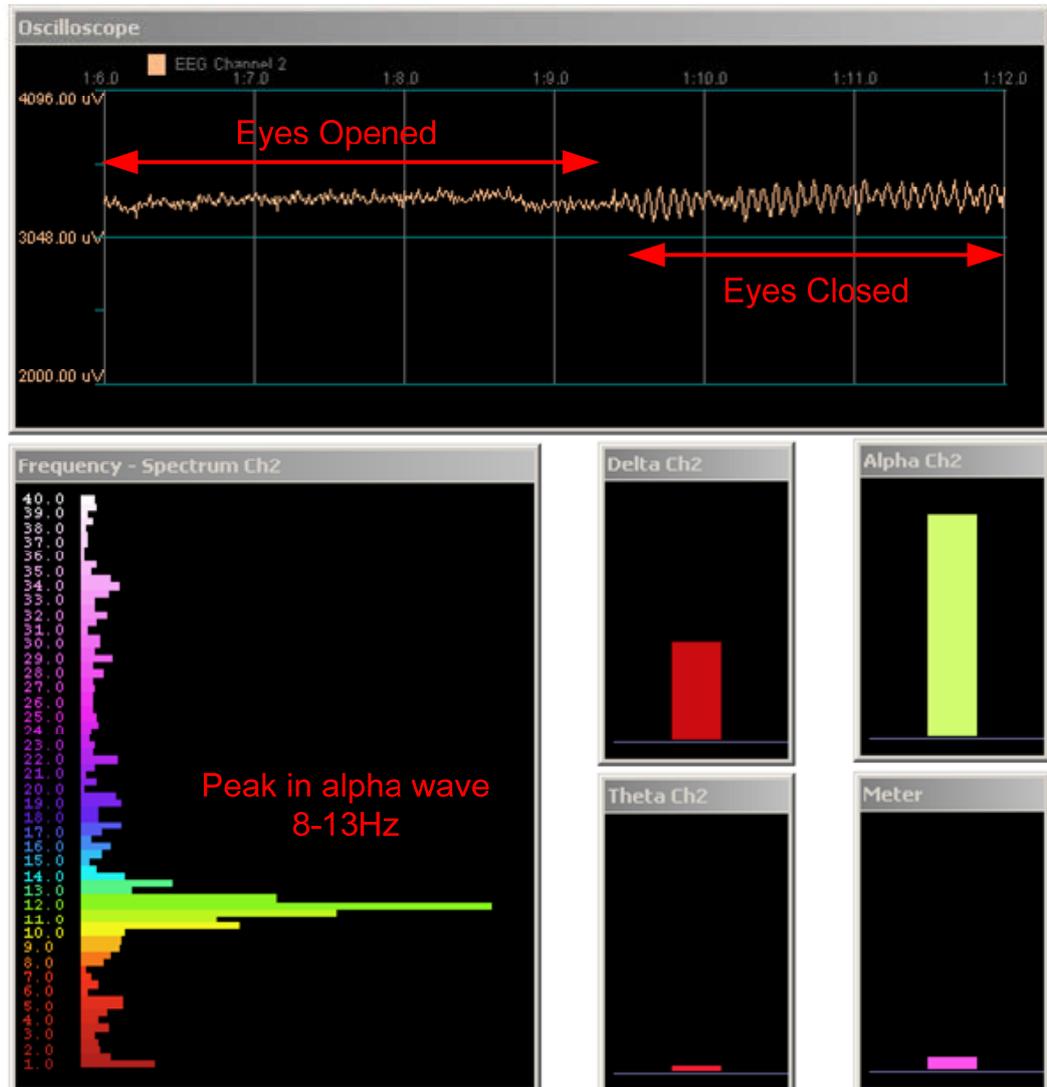


Figure 3.45: Display of the recording software on PC with the wireless EEG V2 for eyes closed and open

Fig. 3.45 shows the display using the recording software on PC for eyes open and closed action. Fig. 3.46 shows the data plot of the raw data by using MATLAB. The time plots in Fig. 3.45 and Fig. 3.46 show a larger amplitude signal from the eyes closed action compared to the eyes open one. The dominant alpha frequency between 8-13Hz during eyes closed on the occipital lobe (O2) is also clearly shown in the frequency plots.

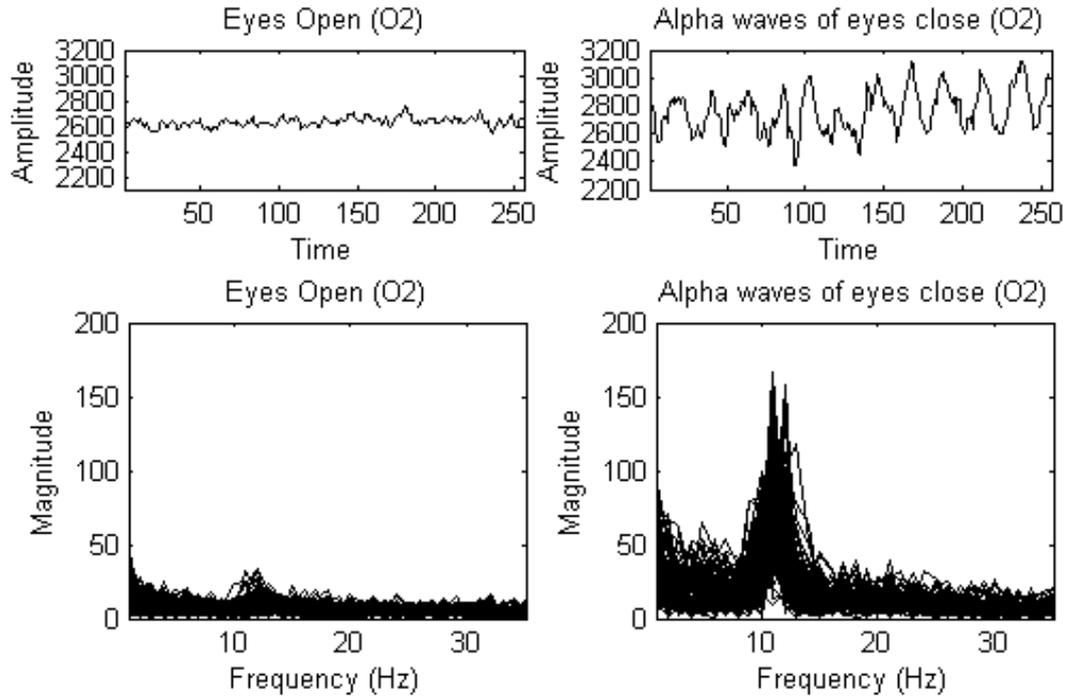


Figure 3.46: Wireless EEG V2 for alpha wave detection during eyes closed action

For the performance measurement, there are two important ways that may be used in the BCI system: classification accuracy and bit rate/ information transfer rate (ITR).

The total dataset for each mental task per subject is 222 units (37 overlapping segments x 6 sessions), which is divided into training, validation and testing sets of 74 units each. This EEG-based BCI is an inter-subject variability system. Therefore, the training of the ANN was done on each subject with initial random weight with the number of hidden neurons varying between 4 to 20 neurons until the best classification was reached with the minimum mean square error (MSE). A validation set is used to determine a network early stop.

Table 3.12: Result of classification eyes closed-open action

Tasks	% Correctly classified per subject						Mean
	S1	S2	S3	S4	S5	S6	
Eyes Closed, Open	99.7	92.3	97.9	98.8	99.6	98.0	97.7

Table 3.13: Result of mental tasks classifications

Mental Imagery Tasks	Accuracy (Acc, %); Information Transfer Rate (ITR, bits/trial)													
	Subject1		Subject2		Subject3		Subject4		Subject5		Subject6		Mean	Mean
	Acc	ITR	Acc	ITR	Acc	ITR	Acc	ITR	Acc	ITR	Acc	ITR	Acc	ITR
Arithmetic, Rotation	84.8	0.4	82.9	0.3	79.1	0.3	76.6	0.2	82.6	0.3	85.1	0.4	81.8	0.3
Arithmetic, Letter	84.8	0.4	83.0	0.3	86.3	0.4	77.4	0.2	85.0	0.4	75.0	0.2	81.9	0.3
Arithmetic, Count	77.3	0.2	82.8	0.3	89.9	0.5	89.3	0.5	75.7	0.2	90.0	0.5	84.2	0.4
Rotation, Letter	85.6	0.4	91.3	0.6	83.7	0.4	77.4	0.2	80.9	0.3	77.7	0.2	82.8	0.4
Rotation, Count	80.9	0.3	82.2	0.3	81.6	0.3	91.8	0.6	78.9	0.3	75.4	0.2	81.8	0.3
Letter, Count	77.1	0.2	91.3	0.6	89.6	0.5	80.6	0.3	80.7	0.3	85.8	0.4	84.2	0.4
Average of combination of two mental tasks classification													82.8	0.4
Arithmetic, Rotation, Letter	73.6	0.5	75.4	0.5	70.4	0.4	65.9	0.3	73.7	0.5	69.0	0.4	71.4	0.4
Arithmetic, Rotation, Count	70.2	0.4	70.8	0.4	72.7	0.5	74.8	0.5	66.8	0.3	71.9	0.5	71.2	0.4
Arithmetic, Letter, Count	67.5	0.4	75.6	0.5	78.5	0.6	70.1	0.4	66.9	0.3	71.4	0.4	71.7	0.5
Rotation, Letter, Count	68.9	0.4	76.7	0.6	72.1	0.5	70.8	0.4	68.4	0.3	66.8	0.3	70.6	0.4
Average of combination of three mental tasks classification													71.2	0.4
Arithmetic, Rotation, Letter, Count	63.7	0.3	66.6	0.3	67.5	0.4	61.7	0.2	61.4	0.4	61.6	0.2	63.8	0.3

The results of classification are shown in Table 3.12 and Table 3.13 with a different classification result for each subject. The overall classification for the closed and open eyes action has reached an average of more than 97%, shown in Table 3.12. Although this task is not mental imagery, the dominant alpha wave during eyes closed action is a good enough option for paralysed individuals who still have the proper eye function,

such as to be used as a command to switch on and off the BCI system or an emergency stop.

A threshold tolerance was added to the output layer with a value of at least 60%. Otherwise, it would be treated as noise. Table 3.13 shows each combination of two mental tasks yielding accuracy above 80% with the best combination of mental arithmetic vs. counting and mental letter composing vs. counting with accuracy around 84%. Moreover, the combination of three mental tasks for most of the subjects has accuracy above 70%, although subject 5 has lower accuracy above 65%. By using three mental tasks classification, the system is satisfactory for providing wheelchair commands forward, left and right. Additional four mental tasks classification is also shown with the result around 63%.

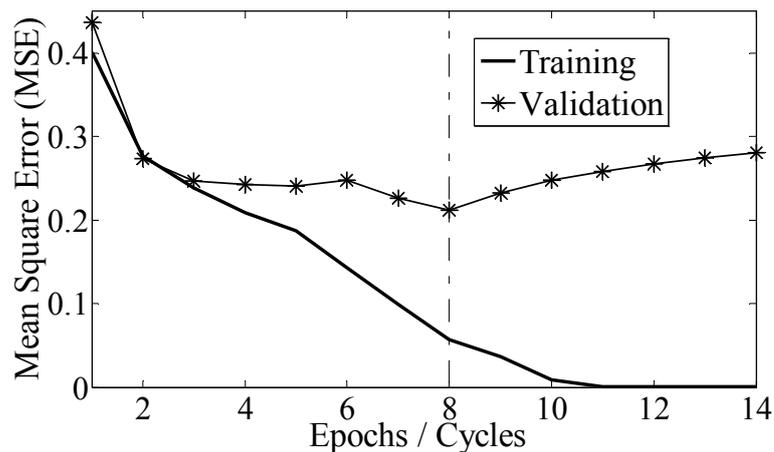


Figure 3.47: Training and validation error cycle of three mental task classifications (math, letter and count)

One of the error cycles performance is shown in Fig. 3.47. Note that the error (MSE) of the training set decreased smoothly. The validation set dropped from the beginning up to 8 epochs, then started to increase continuously. The training of the network was stopped at 8 epochs because the validation performance started to increase continuously. This is used to prevent over-training of the network.

The bit rate values, as shown in Table 3.13, are computed based on equation (3.58). Average bit rate value varies between 0.3 to 0.5 bit/trial. It shows that the accuracy significantly affects to the bit rate value. For example, in four mental tasks classification, as the number of mental tasks is increased and the accuracy is dropped, the speed of the bit rate is reduced. In three mental tasks, classification provides higher average bit rate at an accuracy of around 70%. These three mental tasks would correspond to three wheelchair commands (forward, left and right) as one to one mapping. Otherwise, if using two mental tasks classification, tasks combinations are necessary for three wheelchair commands, which might reduce the bit rate. For wheelchair control that needs at least three commands (left, right, forward), the bit rate of three mental task classification is between 0.4 to 0.5 bit/trial. The ITR values previously reported for BCI systems have reached at least 25 bits/min (Wolpaw 2007). Using the (3.59) and system is configured to provide a classification every second ($V = 60$ trials/min). The ITR can be then calculated by multiplying the bit rate (B) and the speed of the system (V), with the values being between 24 bit/mins and 30 bit/mins.

3.5.5 Results of Real-time Online Classifications

The final weights giving the best classification of the mental tasks are transferred as parameters to the embedded system to provide real-time online classification. Fig. 3.48 illustrates the real-time whole operation of the microcontroller-based BCI operation, and the embedded system uses an MCF5213 microcontroller is applied. The wireless EEG system transmits the real-time raw data via RF 2.4 GHz. The wireless receiver captures and forwards the packets to the main microcontroller. The main embedded system internally has a 32-bit ColdFire (Freescale) core.

This system implements the μ C/OS real-time operating system (RTOS), which runs several application tasks simultaneously. First, the system captures the packets via the SPI bus from the wireless receiver and saves them in a first-in-first out (FIFO) buffer. This is followed by the moving window of one second of data, the band-pass Butterworth filter, the moving average filter, the FFT and the scaling algorithm. The

FFT radix-2 of 256 point was written in an assembler due to the need to use a special accumulator register to speed up the transform. Next, the ANN provides real-time classification with the input parameters set to the best weights taken from the training process. Fig. 3.49 and Fig. 3.50 show the real-time detection of the eyes open and closed action and displays the detection on the LCD. The eyes closed action can be used with the start-stop command. Fig.3.51 shows the real-time detection for mental tasks and provides output detection on the LCD.

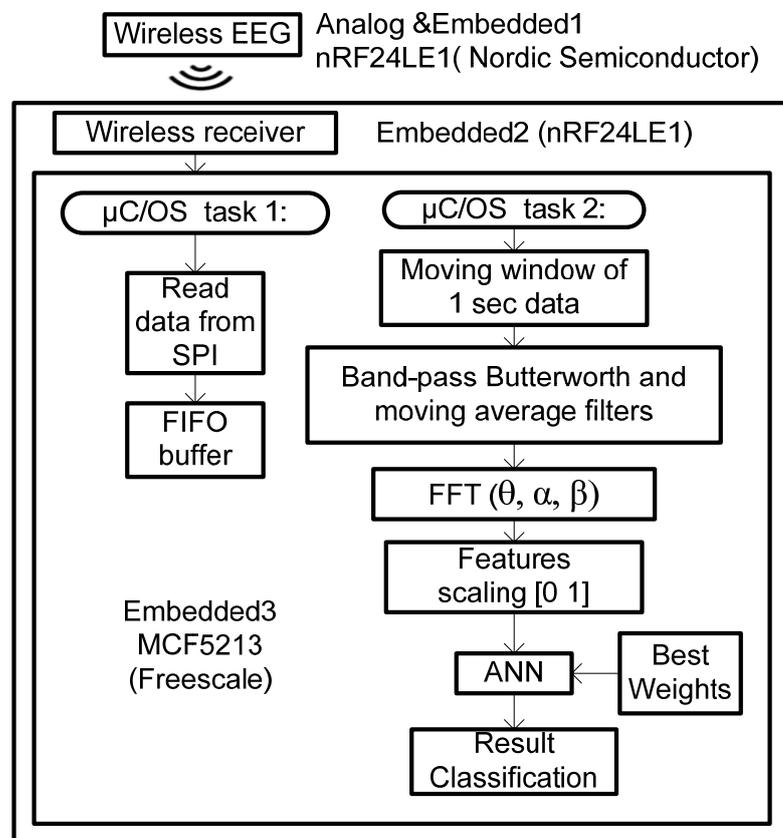


Figure 3.48: Block diagram real-time online microcontroller-based BCI

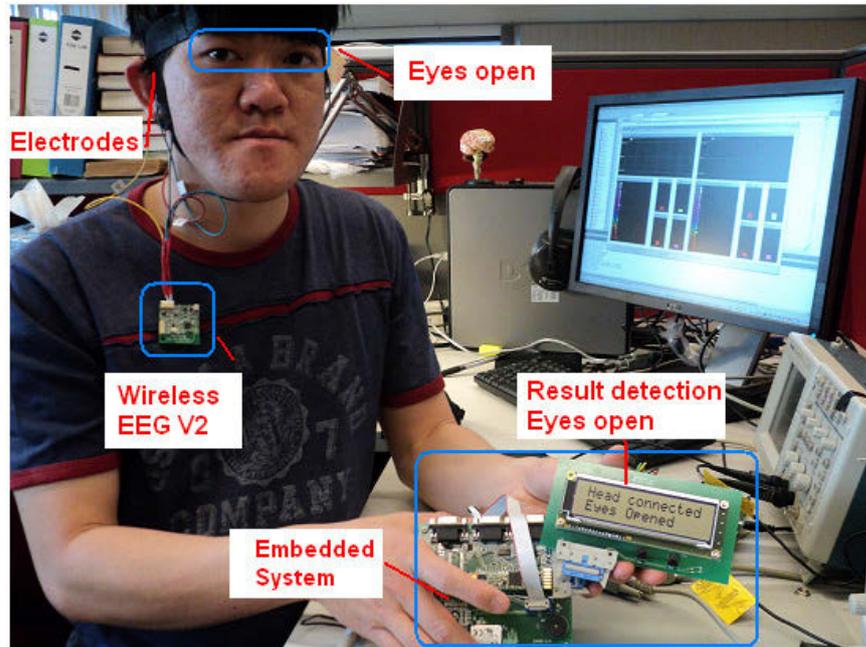


Figure 3.49: Real-time online operation of microcontroller-based BCI for eyes open classification

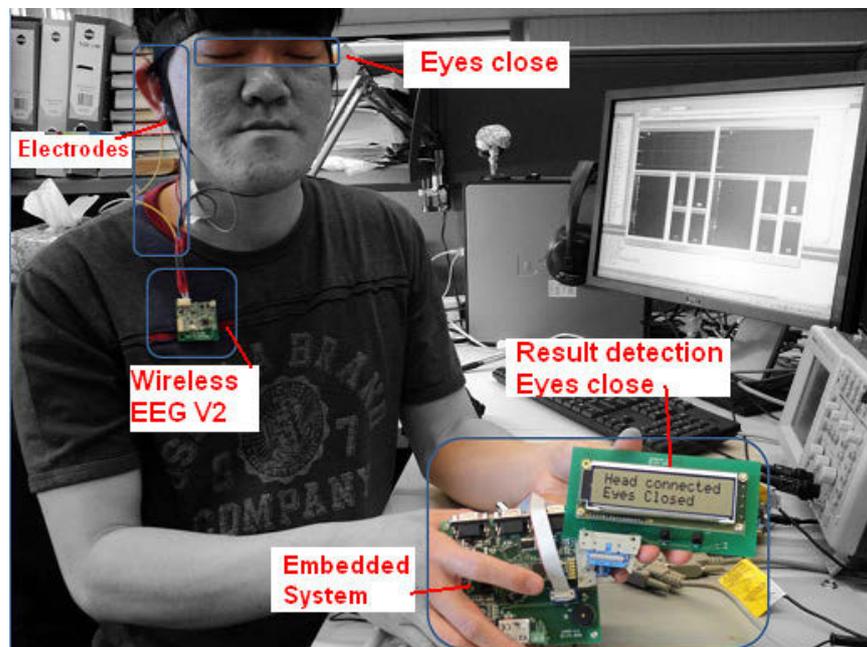


Figure 3.50: Real-time online operation of microcontroller-based BCI for eyes closed classification

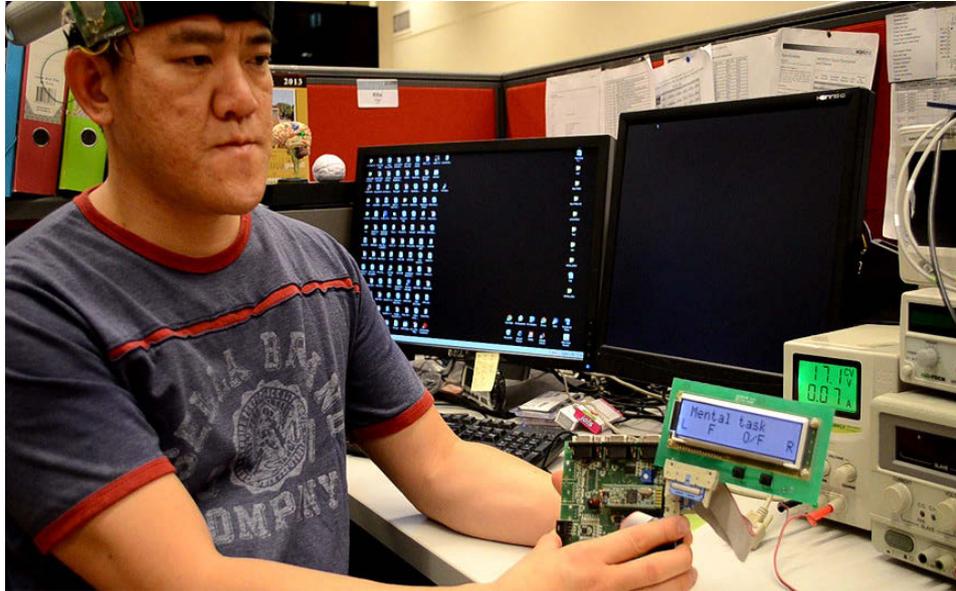


Figure 3.51: Real-time online operation of microcontroller-based BCI for mental task detection

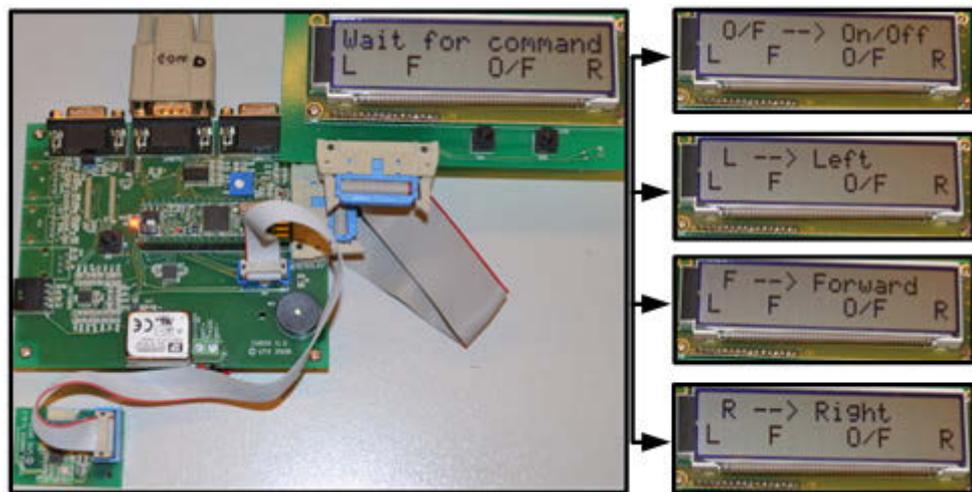


Figure 3.52: Real-time online BCI classifications; map into wheelchair commands and display to the LCD

Table 3.14: Result of real-time online testing

Subjects (S1-S5)	Testing Wheelchair Commands eyes closed for on/off command and user chosen 3-task for left, forward, right commands (each runs at 25 times)								Mean of Success Rate (%)	Speed of execution (seconds)
	On/Off		Left		Forward		Right			
	Correctly identify1	Success rate1(%)	Correctly identify2	Success rate2(%)	Correctly identify3	Success rate3(%)	Correctly identify4	Success rate4(%)		
S1	24	96	18	72	16	64	17	68	75	2 to 3
S2	20	80	16	64	15	60	15	60	66	2 to 4
S3	20	80	17	68	16	64	16	64	69	2 to 3
S4	21	84	18	72	16	64	17	68	72	2 to 4
S5	22	88	15	60	17	68	16	64	70	2 to 4
Mean	21	86	17	67	16	64	16	65	70	2 to 4

The LCD displays the output of the commands recognition, as shown in Fig. 3.52, which is related to the three mental tasks and eyes closed task, which are mapped into wheelchair commands (left, right, forward, and on/off). The embedded system turns into a waiting state for any command activation. As soon as a command is activated, the LCD displays the result of the classification on the first line of the LCD. If the on/off command is activated, the LCD displays ‘O/F → On/Off’. If the turn left command is activated, the LCD displays ‘L → Left’. For a moving forward command, the LCD displays ‘F → Forward’ and for turn right command, the LCD displays ‘R → Right’. The second line of the LCD displays the same information: ‘L F O/F R’ to represent the four intentional signals of wheelchair control. In the real-time testing as shown in Table 3.14, each particular task related to each command is repeated 25 times for the testing experiment. The correctly detected commands and speed of execution are identified. Subjects are achieving, with high classification accuracy on eyes closed task at between 80% and 96% for the on/off command. Testing for the ‘turn left’ command indicates success rates at between 60% and 72%. The success rates for detecting the ‘move

forward' and turn right commands are between 60% and 68%. The average success rate of detection for all commands is 70%, and the speed of executing each task is varied from 2 to 4 seconds.

3.6 Conclusion

The developed microcontroller-based BCI has maximized the aspects of portability, low power, real-time operation and low cost. This wireless EEG provides a good CMRR performance, a compact size and low current consumption with a battery life of 38 to 45 hours for the CR2032 coin cell battery.

The results of classification show a clear detection of alpha waves during closed eyes action with a high classification rate. The overall classification for the closed and open eyes action has reached an average of more than 97%. The system has been able to classify two, three and four mental tasks using only two wireless EEG channels, with the main electrodes positioned at P3 and O2. The combination of the two mental tasks yields accuracy above 80%. Moreover, the combination of three mental tasks for most of the subjects has accuracy above 70%. The four mental tasks classification is also shown with the result around 63%. The three mental tasks classification is sufficient for the wheelchair control, which when mapped into commands moves a wheelchair forward, left and right. The real-time classification validates the result of the classification, with accuracy for eyes closed at between 80% and 96% for the on/off command and the mental tasks for the other commands with accuracy between 60% and 72%. The average success rate of the detection for all commands is 70%, and the speed of executing each task is varied from 2 to 4 seconds. The accuracy of the real-time online classification verified the result accuracy of the offline classification with a close approximation of overall accuracy at 70%.

More study on the feature extraction method and classification algorithm are investigated and will be discussed in the next chapter to find a better feature extractor and classifier in the application of BCI using the mental task paradigm.

Chapter 4

Optimized BCI using Hilbert-Huang Transform Feature Extractor and Genetic Algorithm of Artificial Neural Network Classifier

4.1 Introduction

This thesis concentrates further for BCI optimization using the mental tasks feature extraction and classification. The reasons to choose the BCI using mental tasks are as follows. First, mental task imagery is the same as the sensorimotor rhythm (SMR), which relies on the spontaneous mental signals initiated by the users themselves. As a result, both share the same advantage of not using the external stimuli if compared to

the other types of BCI (SSVEP and P300) as discussed in chapter 2. In mental task-based BCI, the user just needs to imagine the non-motor imagery task, such as letter composing, arithmetic and figure rotation. These mental tasks are not complicated, and most people are familiar with them. Second, although many reports in SMR-based BCI research can be found, there are still some people who are unable to use SMR, known as the BCI illiteracy phenomenon (Allison & Neuper 2010). Third, the selective SMR or motor imagery task defects in severely disabled patients were also reported (Conson et al. 2008). Fourth, individuals who are amputees or have been paralysed for years may be unable to perform SMR tasks effectively (Birch, Bozorgzadeh & Mason 2002; Curran et al. 2004). The SMR task would be problematic for people with impairments in the particular area, such as stroke patients who mostly experience impairments on the motor cortex area.

Fig.4.1 illustrates the basic components for BCI using mental tasks. This starts from data collection of mental tasks by using EEG, followed by a signal pre-processing module, then window segmentation and digital filters. Next, a features extraction module transforms the signals into useful features associated with the related mental task. The features are processed into a classification algorithm using ANN, which includes optimization, training and classification tasks. The outputs classification can be mapped to the three wheelchair steering commands (left, right and forward).

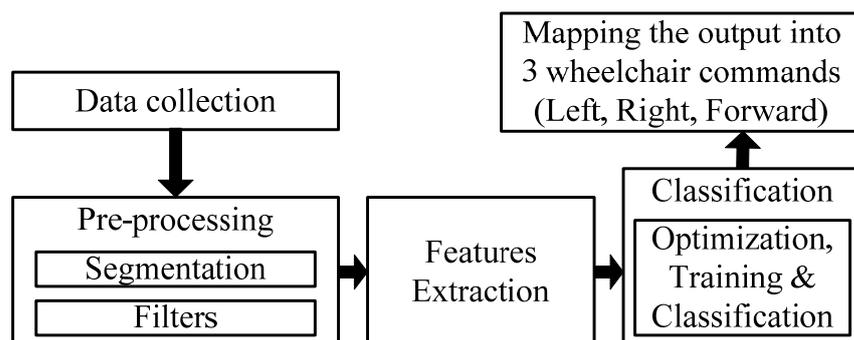


Figure 4.1: Components of BCI using mental tasks

The previous chapter showed a prototype of a microcontroller-based BCI system for mental tasks classification. Computational intelligence was applied to the prototype to form a real-time BCI application. These include the features extraction based on conventional FFT and the classifier based on the ANN without an optimization of the classifier training.

For the FFT features extractor, it is a spectral-based analysis which requires the system to be linear and periodic (stationary); otherwise, the spectrum of the FFT will make little physical sense. The FFT also requires linearity in its analysis. Many natural phenomena can be approximated by using linear analysis but they also have the tendency to be nonlinear whenever their variations become finite in amplitude. EEG is known as a non-linear and non-stationary signal, which is composed of many different frequency components (Andrzejak et al. 2001; Sannelli et al. 2008).

To overcome the drawback using the FFT feature extractor, a time-frequency feature extractor analysis can be used. The Hilbert-Huang transform (HHT) is a time-frequency analysis to analyse nonlinear and non-stationary signals (Huang et al. 1998). The HHT has two steps. The first step is pre-treatment of the original data, which transforms it into orders of intrinsic mode function (IMF) by using the empirical mode decomposition (EMD) method. This is continued in the second step; The Hilbert transform (HT) is used on each of the IMF to calculate the instantaneous frequency to create an integrated time-frequency representation. The HHT is different from FFT, which is based on cosine functions. HHT is self-adaptive to analyse both stationary and non-stationary signals. This can avoid the fake components, which are produced during the FFT. The IMF is deduced based on the time characteristic scale of series data. Different series data can produce different IMF. Each IMF can be regarded as an inherent model of the signal. The instantaneous frequency obtained from the HHT has a clear physical meaning and can express the local characters of the signal. Moreover, the EMD method can extract the average of the data series effectively. The FFT and wavelet methods vibrate surrounding the horizontal axis. As a result, this could limit their performance.

One of the important issues on the neural network is the learning or training of the network. The learning process aims to find a set of optimal network parameters. The conventional ANN used the error correction (Widrow & Lehr 1990) and gradient-descent (GD) technique (Ham & Kostanic 2000). The drawback of the gradient method is that the derivative information is needed so that the error function can be minimized. It has to be continuous and differentiable. Furthermore, the learning process is easily trapped in a local optimum, especially when the problems are multimodal and the learning rules are network structure dependent. To solve this problem, some global search evolutionary algorithms (EAs), such as the genetic algorithm (GA) are employed for searching in a large, complex, non-differentiable and multimodal domain. The GA optimization can be used here to train the ANN classifier to form a GA-ANN for BCI using the mental task data to improve classification accuracy.

This chapter is organized as follows: section 4.2 describes the experimental data collection, section 4.2 covers data pre-processing methods, section 4.3 covers methods of the feature extractors, section 4.4 covers the classification algorithm, section 4.5 shows the results and finally, section 4.6 covers the discussions and conclusion.

4.2 Data Collection

This study was approved by the university human research ethics committee. A total of five able-bodied subjects (three males and two females) aged between 22 and 40 years participated in the experiment. A mono-polar 32 channels EEG system from Compumedic with the sampling rate set to 256 Hz was used, but only eight channels were attached on the scalp for the measurement as shown in Fig.4.2 with the electrodes positioned at locations C3, C4, P3, P4, O1, O2, T3, and T4. A reference electrode was placed at location A2 and location A1 as GND electrode. The electrode placement is referred to the standard of international 10-20 system.

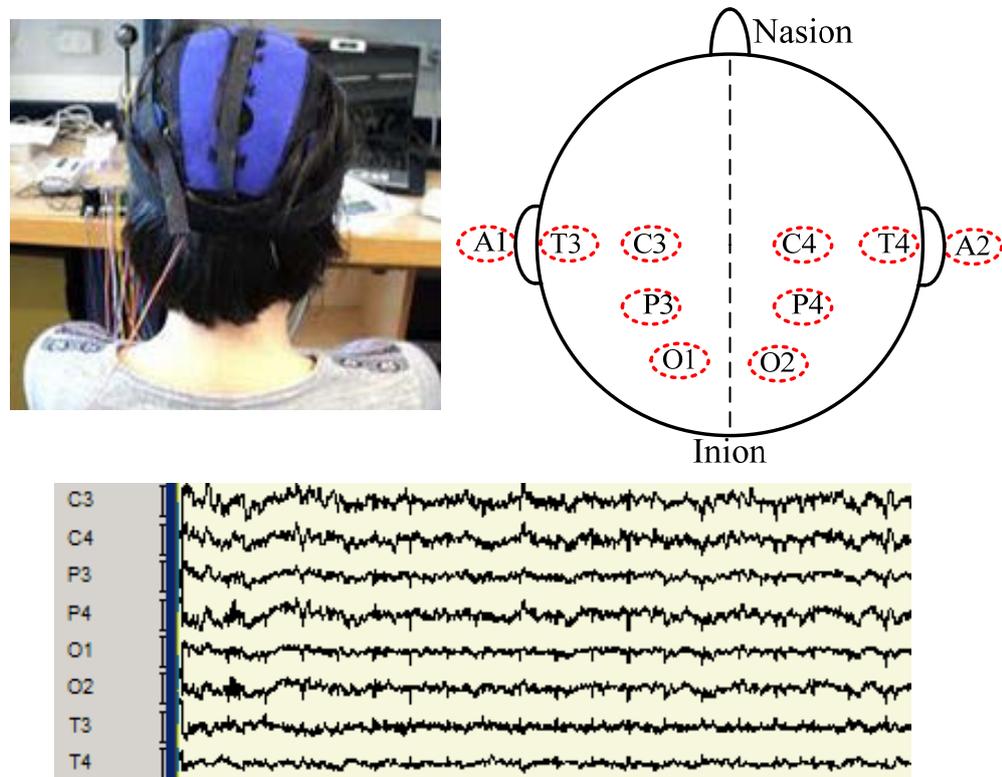


Figure 4.2: EEG system set-ups for data collection

To keep the impedance level low and good electrical contact, prepping and EEG gels were applied on the scalp. The impedance was measured and maintained below 5 k Ω . Unnecessary movements and eye blinks were kept as minimal as possible during data collection in each session, with the total of six mental non-motor imagery tasks used in the study as follows:

- 1) Arithmetic calculation (math): Participants were instructed to imagine solving a series of one by one digit multiplications.
- 2) Letter composing (letter): Participants were asked to mentally compose a simple letter without vocalizing or mouthing the letter.
- 3) Rubik's cube rolling (cube): Participants were asked to imagine a figure of Rubik's cube being rolled forward.

- 4) Visual counting (count): Participants performed mentally counting numbers from one to nine repeatedly by visualizing the number appearing and disappearing on a blackboard in their mind.
- 5) Ringtone (tone): Participants were asked to imagine a familiar mobile ringtone in their head without moving their mouth.
- 6) Spatial navigation (navigate): Participants were asked to imagine moving around and scanning the surroundings in a familiar environment. This task is not using motor imagery because the imagination involved examining the surroundings rather than foot walking as in a motor imagery mental task.
- 7) Additional eyes closed and open tasks are recording for the feature extraction algorithm testing.

4.3 Signal Pre-processing

Each subject performs a recording session of ten sub-sessions on each particular mental task with the duration of 15 seconds on each sub-session as illustrated in Fig.4.3.

The first three seconds is discarded as preparation time. The remaining 12 seconds is processed for further signal pre-processing techniques. First, a moving window segmentation of one second is used with overlapping every quarter second segments to give a result in 45 overlapping segments for the remaining 12 seconds of data. Therefore each subject provides data around 45×10 or 450 units. Next, digital signal processing (DSP) filters are employed to improve raw signal quality. These consist of a Butterworth band-pass filter with a bandwidth of 0.1 Hz to 100 Hz followed by a Butterworth notch filter at 50 Hz. Fig.4.4 shows the alpha wave raw data during eyes closed action and results after the filters were applied.

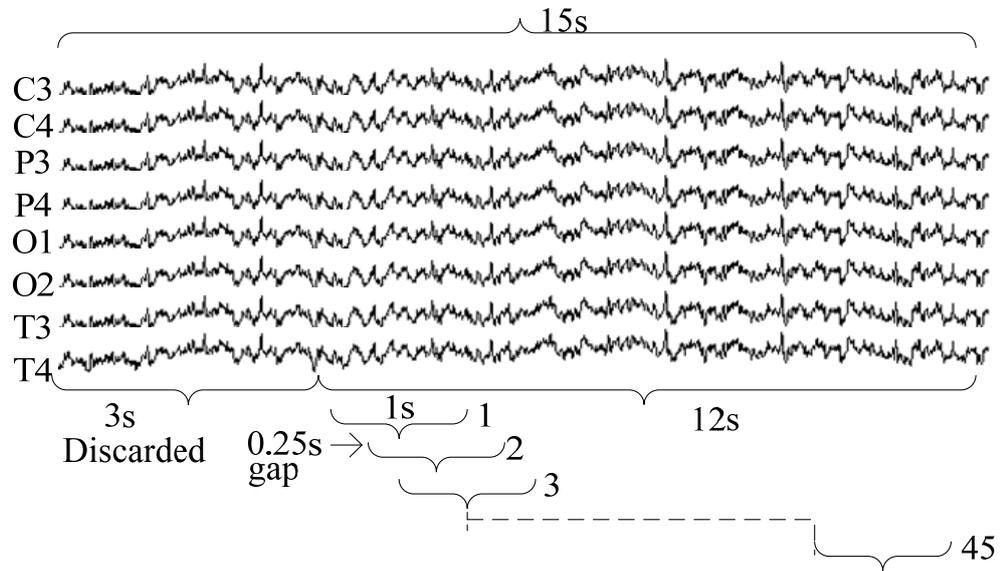


Figure 4.3: Data segmentation

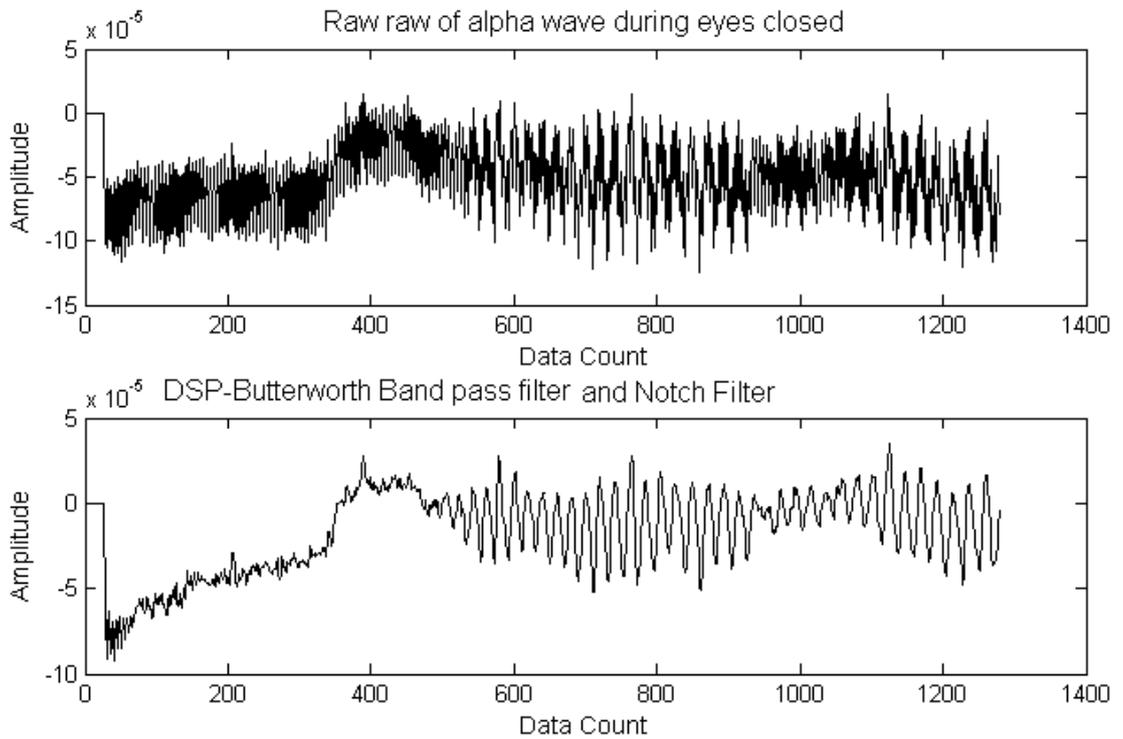


Figure 4.4: DSP filters on the data

4.4 Feature Extraction Algorithms

4.4.1 Fast Fourier Transform (FFT)

The FFT has been discussed in chapter 3. Its spectrum defines uniform harmonic components globally, thus it requires many additional harmonic components for non-stationary data simulation that are non-uniform globally. It spreads the energy over a wide frequency range. The spurious harmonics and the wide frequency spectrum cannot faithfully represent the true energy density in the frequency space. The FFT representation needs the existence of negative light intensity for the components to cancel out one another to give the final delta function. As a result, the FFT components might provide mathematical sense, but do not make physical sense.

Moreover, FFT as the spectral analysis uses linear superposition of trigonometric functions; thus, it requires additional harmonic components to simulate the deformed wave of the profiles. If the form of the data deviates from a pure sine or cosine function, the FFT spectrum has harmonics, which could cause misleading energy-frequency distribution for nonlinear and non-stationary data.

In this chapter, FFT is used again for the features extraction method together with the time-frequency feature extractor which will be elaborated on the next section. For the features extraction process, the power spectral density (PSD) is first computed by squaring the Fast Fourier transform (FFT) of each one-second segment of signal to convert the time base data into the following frequency bands of EEG rhythms: δ (0-3Hz), θ (4-7Hz), α (8-13Hz) and β (14-30Hz). Next, each total energy frequency band is calculated by numerical integration of the PSD over that band using the trapezoidal rule method. With the energy over four bands calculated for each of the 8 channels (C3, C4, P3, P4, O1, O2, T3 and T4), 32 total powers are made available. Finally, power difference as the asymmetry ratio in each spectral band was also calculated with the equation as follows:

$$P_{dif} = (P_R - P_L) / (P_R + P_L) \quad (4.1)$$

where P_{dif} is the power difference on each band, P_R is the power of a particular band on the right channel and P_L is the power of a particular band on the left channel. The total of 64 spectral power differences (4 pairs of channels \times 4 combinations of channels \times 4 bands) is calculated. As a result, an overall 96 units of features are extracted on each one second segment.

4.4.2 Hilbert-Huang Transform (HHT)

Traditional feature extraction methods are based on linear and stationary assumptions. Fourier analysis has been used widely for the traditional feature extractor in BCI. There are some important restrictions of the Fourier spectral analysis: the system must be linear; and the data must be strictly periodic or stationary. If these conditions are not met, the resulting spectrum will make little physical sense. Other than stationary, Fourier spectral analysis also requires linearity. Although many natural phenomena can be approximated by linear system, they also have the tendency to be nonlinear. EEG signal is one of the examples of nonlinear and non-stationary signal (Huang et al. 1998; Huang & Wu 2008).

Alternative to Fourier analysis, other non-stationary feature extraction methods can be used including spectrogram and wavelet analysis. These methods are time-frequency features extraction methods. The spectrogram is a limited time window-width Fourier spectral analysis by successively sliding the window along the time axis to get a time-frequency distribution. Since it is used on the traditional Fourier spectral analysis, the data should be piecewise stationary. This assumption is not always justified in non-stationary data. The wavelet approach is an adjustable window Fourier spectral analysis. The limitation of wavelet analysis is its leakage generated by the limited length of the wavelet function, which makes the quantitative definition of the energy-frequency-time distribution difficult. Another time-frequency analysis is based on HHT as an empirically based data-analysis method. Its basis is adaptive, thus it can produce

physically meaningful representations of data from non-linear and non-stationary signals. (Huang et al. 1998; Huang & Wu 2008).

Multiple superimposed oscillatory components can be found from an EEG signal, which possibly presents different underlying physical components of brain activity. If the components from the signal are properly extracted, it would become much easier to interpret the signal. As a state of the art technology, HHT has been applied in many scientific fields of research, including geophysics (Huang, Shen & Long 1999), image processing (Tan et al. 2006), mechanical fault diagnosis (Li, Zheng & Tang 2005), financial applications (Huang et al. 2003).

There are two basic processes in HHT. The empirical mode decomposition (EMD) is used to decompose the time series of data into sets of intrinsic mode function (IMF), and applying the Hilbert transform (HT) to obtain a spectrum of HHT (Huang et al. 1998; Huang & Wu 2008), as follows:

$$x(t) = \sum_{i=1}^n c_i(t) + r_n(t) \quad (4.2)$$

where $x(t)$ denotes an original signal such as the segment of EEG data, $c_i(t)$ is the i -th extracted IMF, $r_n(t)$ is the residual, and n denotes the number of data. The oscillatory components $c_i(t)$ of the signal are separated with EMD, which is the nonlinear method that adapts to the signal. Components produced by this method are the IMFs. As a result, the instantaneous frequency of these IMFs can be calculated in a meaningful way. The underlying idea is that a signal $x(t)$ is high frequency oscillations superimposed on low oscillations. This leads to the possibility to separate the details $c_i(t)$ from the trend $r_n(t)$. The signal $x(t) = c(t) + r(t)$ consists of intrinsic mode function and residue signal. The same decomposition can be used again on the residual to extract low frequency details, and so on. The EMD algorithm consists of the following steps:

- 1) Find extrema (maxima and minima) of $x(t)$;
- 2) Interpolate between maxima and minima using cubic spline to create upper envelope ($E_u(t)$) and lower envelope ($E_l(t)$);

- 3) Compute mean envelope $m(t) = (E_u(t) + E_l(t)) / 2$;
- 4) Go to step 2 until $c(t)$ is considered IMF
- 5) Iterate from the start with the residue signal $r(t) = x(t) - c(t)$ until the stopping criterion is satisfied.

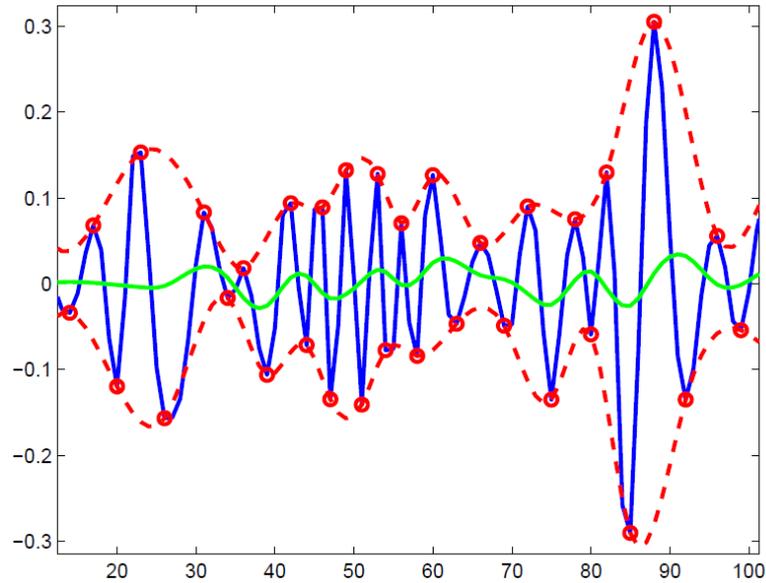


Figure 4.5: EMD process with random test signal

Each IMF from the EMD process should satisfy two conditions. Firstly, the number of extrema and the number of zero crossings must differ at most by one. Secondly, the mean of the upper and the lower envelopes must be equal to zero. These restrictions are necessary to meet the strict conditions for calculating instantaneous frequency. The EMD is used to obtain monotonic IMF from a signal. Sifting is the iterative process of separating one oscillation from the signal. EMD features multiple iterations, and the goal is to decompose the whole signal. The local maxima are connected with a cubic spline to form an envelope. The same is done to the local minima as well. The use of a cubic spline is justified empirically as linear or polynomial interpolation which spread their components to other modes, as shown in Fig 4.5, with the following information: the blue solid line is the random test signal and the red dotted lines are the cubic splines

that go through extrema points. The mean envelope is shown in a green solid signal and it tells that this IMF is not yet finalized as more sifting rounds will be needed to bring the mean envelope to a zero level.

The local mean of the signal, m_l , of these envelopes is calculated. The first component for this round of the sifting h_l is obtained by subtracting the mean from the original signal as follows:

$$h_1 = x(t) - m_1 \quad (4.3)$$

This subtraction of the mean envelope from the component in question is repeated k times as shown in equation (4.3) until a predefined stopping criterion is met. It should be noted that k is individual for each IMF.

$$h_{1k} = h_{1(k-1)} - m_{1k} \quad (4.4)$$

After the last subtraction, such as after k rounds, the first IMF c_l is obtained, as shown in equation (4.5). This IMF is suitable for further analysis for calculating instantaneous frequency.

$$c_1 = h_{1k} \quad (4.5)$$

This concludes the sifting process since an IMF, here c_l is sifted from the original signal with the first sifting extracting the highest frequency oscillation. The IMF is subtracted from the original signal to gain the first residue in equation (4.6). Then, another round of sifting is started using this residue as the input signal. The first sifting process has extracted the highest frequency oscillation. The next sifting will produce the

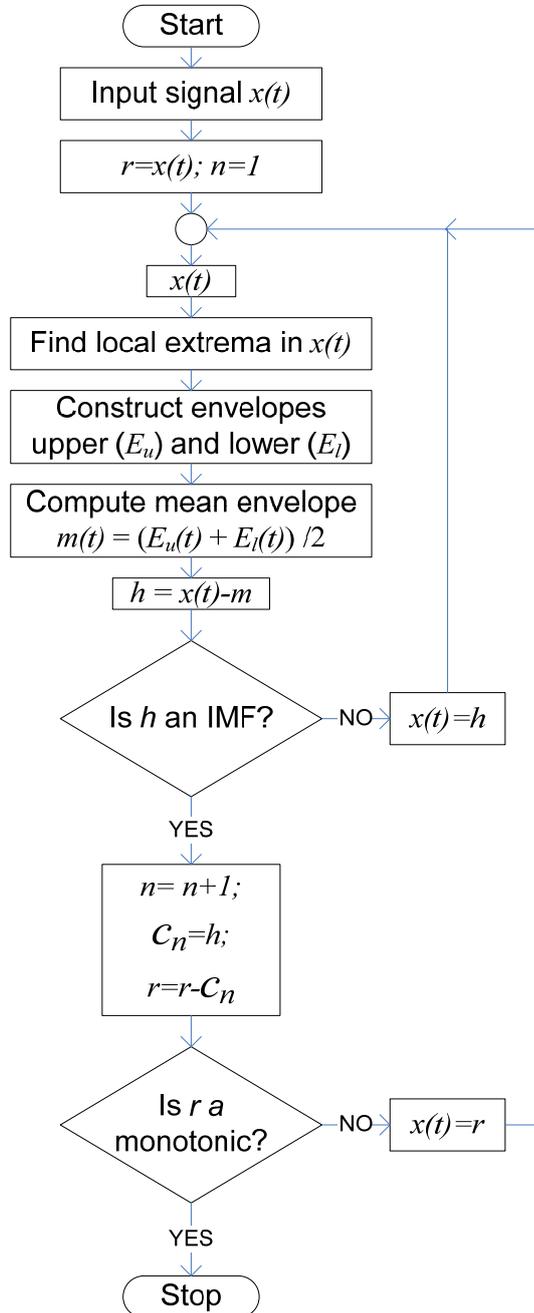
second highest oscillation. Similarly, the number of extrema decreases as more IMFs are extracted.

$$r_1 = x(t) - c_1 \quad (4.6)$$

The process is finished when r_n reaches monotonic or c_n or r_n have too small an effect. The number of the resulting IMFs is not predefined thus the EMD method automatically determines how many oscillation modes are to be found. After n rounds of sifting a residue is left, which is constant, which represent the trend as shown in the equation below.

$$r_n = r_{n-1} - c_n \quad (4.7)$$

Since EMD relies on subtraction, the original signal can be composed by summing up in the IMFs and the residue as in the original equation (4.2). As each IMF contains lower frequencies than the earlier ones, it is possible to exclude high or low frequencies, thus detrending the signal. The sifting process is the core of the EMD method; it forms the IMFs from the input signal. Sifting removes the riding wave from the signal and makes it more symmetric. Maxima with a negative value are moved up and minima with a positive value are moved down. It is noteworthy that this method requires only the local extrema as parameters. As a result, no zero reference is required, since the EMD method generates it for each IMF. The flowchart of the EMD process is shown in Fig.4.6.

**Figure 4.6:** Flowchart of EMD

Next, the HT is applied to the IMF to obtain instantaneous amplitude and frequency as a function of time, as represented by the spectrum of HHT. In order to produce a time-frequency domain presentation of the signal, the Hilbert spectrum needs to be calculated. It is a complex conjugate transform that takes into account the local changes of a signal. The Cauchy principal value is an integration method that is used in the HT. By surrounding a point with small interval ε , a mathematically difficult point may be integrated along a positive semi-circle of radius ε . The integration is completed when ε approaches 0. The principal value of a finite integral between $a \leq b$ of function f at point c , where $a \leq c \leq b$ is defined as follows:

$$\text{P} \int_a^b f(x)dx \equiv \lim_{\varepsilon \rightarrow 0^+} \left[\int_a^{c-\varepsilon} f(x)dx + \int_{c+\varepsilon}^b f(x)dx \right] \quad (4.8)$$

For a doubly infinite integral, the Cauchy principal value is defined in the following equation which is used with the HT. The Cauchy principal value is denoted by P.

$$\text{P} \int_{-\infty}^{\infty} f(x)dx \equiv \lim_{R \rightarrow \infty} \int_{-R}^R f(x)dx \quad (4.9)$$

The HT is defined in the following equation:

$$\tilde{x}(t) = \frac{1}{\pi} \text{P} \int_{-\infty}^{\infty} \frac{x(u)}{t-u} du \quad (4.10)$$

where P denotes the Cauchy principal value, which solves the singularity point problem when $u=t$. Point t is delimited with a small interval $[t- \varepsilon, t+ \varepsilon]$ and then this point is approached from each ends of the interval.

$$z(t) = x(t) + j\tilde{x}(t) = a(t)e^{j\theta(t)} \quad (4.11)$$

where imaginary unit $j = \sqrt{-1}$, the instantaneous amplitude $a(t)$ and phase $\theta(t)$ are gained from the alternative presentation. Imaginary $\tilde{x}(t)$ is the Hilbert transform of $x(t)$ as in equation (4.10).

The instantaneous features of the analytic signal $z(t)$ are simple to compute with the instantaneous amplitude $a(t)$ which can be interpreted as the distance of the values of the analytic signal from the time axis as in equation (4.12). Instantaneous phase $\theta(t)$ is gained from the angle in the complex plane in equation (4.13). The instantaneous frequency $\omega(t)$ can be calculated as the derivative of the phase if the processed signal is an IMF.

$$a(t) = \sqrt{x^2(t) + \tilde{x}^2(t)} \quad (4.12)$$

$$\theta(t) = \arctan \left[\frac{\tilde{x}(t)}{x(t)} \right] \quad (4.13)$$

$$x(t) = \cos(\omega t) \quad (4.14)$$

$$\tilde{x}(t) = \sin(\omega t) \quad (4.15)$$

Then the analytic signal is as follows:

$$z(t) = x(t) + j\tilde{x}(t) = \cos(\omega t) + j \sin(\omega t) \quad (4.16)$$

To calculate phase, simply determine the angle in the complex plane by using the following:

$$\theta(t) = \arctan \left[\frac{\sin(\omega t)}{\cos(\omega t)} \right] = \arctan(\tan(\omega t)) = \omega t \quad (4.17)$$

Then by differentiating, it will provide the instantaneous frequency ω as follows:

$$\frac{d\theta(t)}{dt} = \frac{d(\omega t)}{dt} = \omega \quad (4.18)$$

After having acquired the n HT of the IMFs, it is possible to reconstruct the signal. The original signal can be expressed in equation (4.19) as the real part:

$$x(t) = \Re \left[\sum_{i=1}^n a_i(t) \exp \left(j \int \omega_i(t) dt \right) \right] \quad (4.19)$$

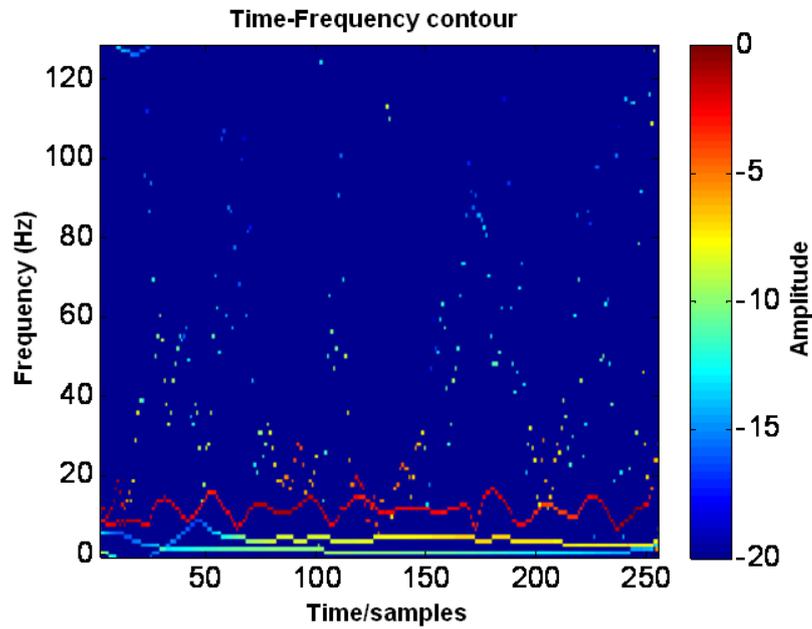


Figure 4.7: Example of the 3D plot of the HHT spectrum

An amplitude distribution on the time-frequency plane is called the Hilbert spectrum that can be constructed from the Hilbert analytical signal, which produces a three-dimensional plot, where amplitude and frequency act as a function of time. The HT spectrum is presented as a colour map, with time and frequency axes and the colour indicating the amplitude as shown in Fig.4.7.

In practical terms, the HT spectrum presents the time-frequency distribution as a matrix of time by frequency rows and columns. The value of a selected cell then represents the amplitude at that time and frequency. Each of the IMFs is presented in this matrix, having its frequency and amplitude changes plotted through the time-frequency field.

For this study, as in the FFT method in the previous section, the power spectrum of the HHT calculated by squaring the HHT was used. Next, the total energy of each frequency band is calculated by numerical integration of the spectra over that band using the trapezoidal rule method. With the energy over four bands calculated for each of the 8 channels (C3, C4, P3, P4, O1, O2, T3 and T4), 32 total power levels are made available for 8 channels and 8 total power levels if two channels are used. The same is applied for the FFT. The power difference of the asymmetry ratio is also calculated in each spectral band using equation (4.2). The total of 64 spectral power differences (4 pairs of channels \times 4 combinations on channels \times 4 bands) is calculated for an 8 channel EEG. As a result, a total of 96 units of features are extracted on each one second segment.

4.5 Classification Algorithms using the Genetic Algorithm Optimization of Artificial Neural Network (GA-ANN)

The artificial neural network (ANN), which has been discussed in chapter 3, is a popular tool used in biomedical and brain computer interface applications, especially for the pattern recognition and classification algorithm (Nguyen 2008).

This study utilizes a three-layer feed forward ANN with one hidden layer network and is used to classify 3 output classes of the mental tasks as shown in Fig. 4.8. The output vector z and the k -th component z_k are computed as follows:

$$z = f(\mathbf{W}\mathbf{x} + b) \tag{4.20}$$

$$z_k(x, w) = f_1\left(b_k + \sum_{j=1}^m w_{kj} f_2\left(b_j + \sum_{i=1}^n w_{ji} x_n\right)\right) \tag{4.21}$$

where f, f_1, f_2 is the activation function, x represents the input vector, W is the weight matrix vector, b is the scalar bias, n is the number of input nodes, m is the number of output nodes, w_{ji} is the weight to the hidden unit y_j from the input unit x_i , w_{kj} represents the weights to output unit z_k from hidden unit y_j . The biases are represented by b_j and b_k .

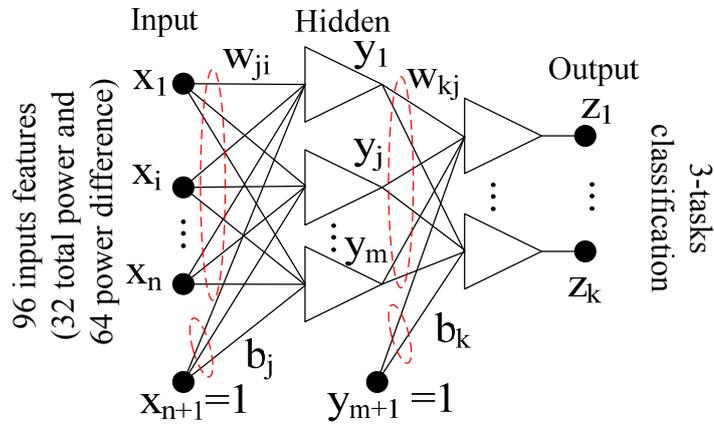


Figure 4.8: Architecture of ANN

The log-sigmoid function is used as the activation function, which provides data values between one and zero. As a result, prior to ANN the feature data value needs to be scaled to within the range of zero to one, as follows:

$$X^* = (X - X_{min}) / (X_{max} - X_{min}) \quad (4.21)$$

where X is the input features value, X^* is the value after scaling, X_{min} is the minimum and X_{max} is the maximum of the input feature values.

One of the problems of the ANN is its training. The training process aims to achieve a set of optimal network parameters. In a conventional way, two major classes of learning method, the error correction (Widrow & Lehr 1990) and gradient methods (Ham & Kostanic 2000) were used. One major disadvantage of the gradient method is that the derivative information is necessary to continue and the differentiable is such that the error function can be minimized. Also, the training process is easily trapped in a local optimum, especially when the problems are multimodal and the learning rules are dependent on network structure. To solve this problem, some global search using evolutionary algorithms (EA) (Bäck & Schwefel 1993) such as genetic algorithm (GA), is used for searching in a large, complex, non-differentiable and multimodal domain.

```
begin
     $\tau \rightarrow 0$            //  $\tau$ : iteration number
    initialize  $P(\tau)$     //  $P(\tau)$ : population for iteration  $\tau$ 
    evaluate  $f(P(\tau))$  //  $f(P(\tau))$ : fitness function
    while (not termination condition) do
        begin
             $\tau \rightarrow \tau + 1$ 
            select 2 parent  $p_1$  and  $p_2$  from  $P(\tau - 1)$ 
            perform crossover and mutation operations
            reproduce a new  $P(\tau)$ 
            evaluate  $f(P(\tau))$ 
        end
    end
end
```

Figure 4.9: Procedure of GA

In this chapter, the genetic algorithm is used to optimize the ANN training. This combination forms a GA-ANN classification algorithm. The GA is one of the

evolutionary computation techniques (Michalewicz 1994; Srinivas & Patnaik 1994) that can tackle complex optimization problems and can help to find a globally optimal solution over a domain. The GA process for optimization of ANN is shown in Fig.4.9.

A population of chromosomes P is first initialised, where $P = [\mathbf{p}_1 \ \mathbf{p}_2 \ \cdots \ \mathbf{p}_{pop_size}]$, pop_size is the number of chromosomes in the population. Each chromosome \mathbf{p}_i contains some genes (variables) p_{i_j} .

$$\mathbf{p}_i = [p_{i_1} \ p_{i_2} \ \cdots \ p_{i_j} \ \cdots \ p_{i_{no_vars}}], \quad i=1, 2, \dots, no_vars, \quad (4.22)$$

$$para_{min}^j \leq p_{i_j} \leq para_{max}^j, \quad (4.23)$$

$$\mathbf{p}_{max} = [para_{max}^1 \ para_{max}^2 \ \cdots \ para_{max}^{no_vars}] \quad (4.24)$$

$$\mathbf{p}_{min} = [para_{min}^1 \ para_{min}^2 \ \cdots \ para_{min}^{no_vars}] \quad (4.25)$$

where no_vars denotes the number of variables (genes); $para_{min}^j$ and $para_{max}^j$ are the minimum and maximum values of p_{i_j} respectively for all j .

Repeating procedures were done for the purpose of the population evolving from generation τ to $\tau+1$, as follows:

- 1) Two parents (\mathbf{p}_1 and \mathbf{p}_2) are selected from P based on the selection operation in such a way that probability of selection is proportional to their fitness values.
- 2) A new offspring is generated from these parents after undergoing the crossover and mutation operations, which are governed by the probabilities of the crossover (μ_c) and mutation (μ_m).
- 3) Thus the population is generated to replace the current population.
- 4) The procedures (1-3) are repeated until certain termination conditions are satisfied, such as when a predefined number of iterations (generations) have been reached.

Traditional binary code genetic algorithm (BCGA) (Michalewicz 1994) has some disadvantages when applied to multidimensional, high-precision numerical problems. For example, if 100 variables in the range $[-500, 500]$ are involved, and a precision of 6 digits after the decimal point is required, the length of the binary solution vector is 3000. This, in turn, generates a search space of about 2^{3000} points. The performance of the BCGA will then be poor. The situation can be improved if GA uses the real (floating point) numbers. Each chromosome is coded as a vector of floating point numbers of the same length as the solution vector. This kind of GA in real numbers is called a real-coded genetic algorithm (RCGA) (Eshelman & Schaffer 1993; Herrera, Lozano & Verdegay 1998).

RCGA uses three operations: selection, crossover and mutation. The selection is used to select the chromosomes from the population with respect to some probability distribution based on some fitness values. The crossover operation is used to combine the information of the selected chromosomes (parents) and generate the offspring. The mutation operation is used to change the offspring variables.

For the selection operations, the roulette wheel selection (Michalewicz 1994), normalised geometric ranking selection (Goldberg & Deb 1991; Joines & Houck 1994), and tournament selection (Goldberg & Deb 1991) can be used. Crossover operations, which include single-point crossover, arithmetic crossover (Michalewicz 1994), heuristic crossover (Grefenstette et al. 1985), extended intermediate crossover (EIX) (Mühlenbein & Schlierkamp-Voosen 1993), blend- α crossover (BLX- α) (Eshelman & Schaffer 1993) and unimodal normal distribution crossover (UNDX) (Kita, Ono & Kobayashi 1998; Ono, Satoh & Kobayashi 1999) have been proposed. Mutation operations, including uniform mutation (random mutation), boundary mutation (Michalewicz 1994), and non-uniform mutation can be found (Michalewicz 1994; Neubauer 1997).

In roulette wheel selection, for reproduction, two chromosomes in the population will be selected to undergo genetic operations. In this way, it is believed that high potential parents will produce better offspring (survival of the best ones). The

chromosome that has a higher fitness value should therefore have a higher chance to be selected. The roulette wheel selection can be done by assigning a probability q_i to the chromosome \mathbf{p}_i , as follows:

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{j=1}^{pop_size} f(\mathbf{p}_j)}, i=1, 2, \dots, pop_size \quad (4.26)$$

The cumulative probability \hat{q}_i for the chromosome \mathbf{p}_i is defined as,

$$\hat{q}_i = \sum_{j=1}^i q_j, i=1, 2, \dots, pop_size \quad (4.27)$$

The roulette wheel selection process starts by randomly generating a nonzero floating point number, $d \in [0, 1]$. Then, the chromosome \mathbf{p}_i is chosen if $\hat{q}_{i-1} < d \leq \hat{q}_i$, $i = 1, 2, \dots, pop_size$ ($\hat{q}_0 = 0$). It can be observed from this selection process that a chromosome having a larger $f(\mathbf{p}_i)$ will have a higher chance to be selected, where $f(\mathbf{p}_i)$ is the fitness value of the selected parent. Consequently, the best chromosomes will produce more offspring, the average will stay and the worst will die off.

Normalized geometric ranking is a ranking selection function based on a non-stationary penalty function, which is a function of the generation number. As the number of generations increase, the penalty increases, which puts more selective pressure on the RCGA to find a feasible solution. Generally, a high-rank chromosome will have a higher chance of being selected. Ranking methods only require the evaluation function to map the solutions to a partially ordered set. Ranking methods assign chromosome \mathbf{p}_i based on the rank of solution i when all the solution are sorted. The normalized geometric ranking selection can be done by assigning a probability q_i to the chromosome \mathbf{p}_i as follows:

$$q_i = q' \times (1 - q_{best})^{rank-1}, i = 1, 2, \dots, pop_size \quad (4.28)$$

$$q' = \frac{q_{best}}{1 - (1 - q_{best})^{pop_size}} \quad (4.29)$$

where q_{best} is the probability of selecting the best chromosome which is chosen by the users, $rank$ is the rank of the chromosome with the best at $rank = 1$.

Tournament selection, like the ranking methods that only require the evaluations of the fitness function to map solutions to a partially ordered set, involves no probability assignment in the method. In tournament selection, a number $C_{tour} \in [2 \quad pop_size]$ is the tournament size. The best chromosomes from this group are selected as parents to produce the offspring.

In single-point crossover exchanges information comes from two selected chromosomes (\mathbf{p}_1 and \mathbf{p}_2), where,

$$\mathbf{p}_1 = [p_{1_1} \quad p_{1_2} \quad p_{1_3} \quad \dots \quad p_{1_{no_var}}] \quad (4.30)$$

$$\mathbf{p}_2 = [p_{2_1} \quad p_{2_2} \quad p_{2_3} \quad \dots \quad p_{2_{no_var}}] \quad (4.31)$$

It generates a random integer number r from a uniform distribution from 1 to no_var , and creates new offspring $\mathbf{o}_{s_c^1}$ and $\mathbf{o}_{s_c^2}$ as follows:

$$\mathbf{o}_{s_c^1} = [p_{1_1} \quad p_{1_2} \quad \dots \quad p_{1_r} \quad p_{2_{r+1}} \quad \dots \quad p_{2_{no_var}}] \quad (4.32)$$

$$\mathbf{o}_{s_c^2} = [p_{2_1} \quad p_{2_2} \quad \dots \quad p_{2_r} \quad p_{1_{r+1}} \quad \dots \quad p_{1_{no_var}}] \quad (4.33)$$

where, no_var denotes the number of variables (genes) in a chromosome.

Arithmetic crossover is a linear combination of two selected chromosomes (\mathbf{p}_1 and \mathbf{p}_2). The resulting offspring o_{s_c} is defined as,

$$\mathbf{o}_{s_c} = C_{arith} \times \mathbf{p}_1 + (1 - C_{arith}) \times \mathbf{p}_2 \quad (4.34)$$

This operation depends on a coefficient C_{arith} , which should be a constant when the operation is a uniform convex crossover. Otherwise, if $C_{arith} \in [0, 1]$ varies, it is a non-uniform convex crossover.

Heuristic crossover produces a linear extrapolation of two chromosomes. It is an operation that utilizes the fitness value information. The resulting offspring is given as follows:

$$\mathbf{o}_{s_c} = \mathbf{p}_1 + (\mathbf{p}_1 - \mathbf{p}_2) \times r_d \quad (4.35)$$

where $\mathbf{o}_{s_c} = [o_{s_1}, o_{s_2}, \dots, o_{s_{no_vars}}]$, r_d is a random number in $[0, 1]$, and \mathbf{p}_1 is better than \mathbf{p}_2 in terms of the fitness value. Define:

$$feasibility(o_{s_k}) = \begin{cases} 1 & \text{if } para_{min}^k \leq o_{s_k} \leq para_{max}^k \quad \forall k \\ 0 & \text{otherwise} \end{cases} \quad (4.36)$$

where $para_{min}^k$ and $para_{max}^k$ are the minimum and maximum values of the parameter o_{s_k} respectively. If \mathbf{o}_{s_c} is infeasible, i.e. any $feasibility(o_{s_k})=0$, a new random number r_d should be generated to create a new offspring using equation (4.35). Until \mathbf{o}_{s_c} is feasible, this process is repeated.

Extended intermediate crossover is a combination of two selected parents \mathbf{p}_1 and \mathbf{p}_2 . The resulting offspring \mathbf{o}_{s_c} is defined as,

$$\mathbf{o}_{s_c} = \mathbf{p}_1 + C_{ext}(\mathbf{p}_2 - \mathbf{p}_1), i = 1, \dots, no_vars \quad (4.37)$$

where C_{ext} is chosen randomly in the interval $[-0.25, 1.25]$.

Blend- α crossover is a combination of two selected parents \mathbf{p}_1 and \mathbf{p}_2 . The resulting offspring \mathbf{o}_{s_c} is chosen randomly from the interval $[X_i^1, X_i^2]$ following the uniform distribution:

$$X_i^1 = \min(p_{1_i}, p_{2_i}) - \alpha d_i \quad (4.38)$$

$$X_i^2 = \max(p_{1_i}, p_{2_i}) - \alpha d_i \quad (4.39)$$

where $d_i = |p_{1_i} - p_{2_i}|$, p_{1_i} and p_{2_i} are the i -th elements of \mathbf{p}_1 and \mathbf{p}_2 , respectively, and α denotes a positive constant.

Unimodal normal distribution crossover is a mixture of three selected parents \mathbf{p}_1 , \mathbf{p}_2 , and \mathbf{p}_3 . The resulting offspring is defined as,

$$\mathbf{o}_{s_c^1} = \begin{bmatrix} \mathbf{o}_{s_c^1} & \mathbf{o}_{s_c^1} & \dots & \mathbf{o}_{s_{no_vars}^1} \end{bmatrix} = \mathbf{m} + z_1 e_1 + \sum_{i=2}^{no_vars} z_i e_i \quad (4.40)$$

$$\mathbf{o}_{s_c^2} = \begin{bmatrix} \mathbf{o}_{s_c^2} & \mathbf{o}_{s_c^2} & \dots & \mathbf{o}_{s_{no_vars}^2} \end{bmatrix} = \mathbf{m} + z_1 e_1 + \sum_{i=2}^{no_vars} z_i e_i \quad (4.41)$$

where,

$$\mathbf{m} = \frac{(\mathbf{p}_1 + \mathbf{p}_2)}{2}, \quad (4.42)$$

$$z_1 = N(0, \sigma_1^2), \quad z_i = N(0, \sigma_2^2), \quad i = 2, \dots, no_vars, \quad (4.43)$$

$$\sigma_1 = C_{undx}^1 d_1, \quad \sigma_2 = \frac{C_{undx}^2 d_2}{\sqrt{\text{no_vars}}}, \quad (4.44)$$

$$e_1 = \frac{(\mathbf{p}_2 - \mathbf{p}_1)}{|\mathbf{p}_2 - \mathbf{p}_1|}, \quad (4.45)$$

$$e_m \perp e_n (m \neq n), \quad m, n = 2, \dots, \text{no_vars}, \quad (4.46)$$

$N(\cdot)$ denotes a normal-distributed random number, d_1 denotes the distance between the parent \mathbf{p}_1 and \mathbf{p}_2 , d_2 denotes the distance of \mathbf{p}_3 from the line connecting \mathbf{p}_1 and \mathbf{p}_2 , C_{undx}^1 and C_{undx}^2 are constant coefficients of UNDX.

Uniform mutation is a simple random mutation operation. It randomly selects one gene and sets it equal to a random number between its upper and lower bound.

$\mathbf{o}_{s_c} = [o_{s_1} \ o_{s_2} \ \dots \ o_{s_{\text{no_vars}}}]$ is a chromosome and a gene o_{s_c} and when it is randomly selected for mutation (the value of o_{s_c} is inside $[para_{\min}^c, para_{\max}^c]$), the resulting chromosome is then given by $\hat{\mathbf{o}}_{s_c} = [o_{s_1}, \dots, \hat{o}_{s_c}, \dots, o_{s_{\text{no_vars}}}]$, $c \in \{1, 2, \dots, \text{no_vars}\}$, and

$$\hat{o}_{s_c} = U(para_{\min}^c, para_{\max}^c), \quad (4.47)$$

where $U(para_{\min}^c, para_{\max}^c)$ is a random number between the upper and lower bounds.

Boundary mutation changes the selected gene to its lower or upper bound value. If $\mathbf{o}_{s_c} = [o_{s_1} \ o_{s_2} \ \dots \ o_{s_{\text{no_vars}}}]$ is a chromosome and a gene o_{s_c} is randomly selected for mutation (the value of o_{s_c} is inside $[para_{\min}^c, para_{\max}^c]$), the resulting chromosome is then given by $\hat{\mathbf{o}}_{s_c} = [o_{s_1}, \dots, \hat{o}_{s_c}, \dots, o_{s_{\text{no_vars}}}]$, $c \in \{1, 2, \dots, \text{no_vars}\}$, and

$$\hat{o}_{s_c} = \begin{cases} para_{\min}^c & \text{if } r_d = 0 \\ para_{\max}^c & \text{if } r_d = 1 \end{cases} \quad (4.48)$$

where r_d is a random number equal to 0 or 1 only.

Non-uniform mutation is an operation with a fine-tuning capability. Its action depends on the generation number of the population. The operation takes place as follows. If $\mathbf{o}_{s_c} = [o_{s_1} \ o_{s_2} \ \dots \ o_{s_{no_vars}}]$ is a chromosome and the gene o_{s_c} is randomly selected for mutation (the value of o_{s_c} is inside $[para_{\min}^c, para_{\max}^c]$), the resulting chromosome is then given by $\hat{\mathbf{o}}_{s_c} = [o_{s_1}, \dots, \hat{o}_{s_c}, \dots, o_{s_{no_vars}}]$, $c \in \{1, 2, \dots, no_vars\}$,

$$\hat{o}_{s_c} = \begin{cases} o_{s_c} + \Delta(\tau, para_{\max}^c - o_{s_c}) & \text{if } r_d = 0 \\ o_{s_c} - \Delta(\tau, o_{s_c} - para_{\min}^c) & \text{if } r_d = 1 \end{cases} \quad (4.49)$$

where r_d is a random number equal to 0 or 1 only. The function $\Delta(\tau, y)$ returns a value in the range $[0, y]$ such that $\Delta(\tau, y)$ approaches 0 as τ increases. It is defined as follows:

$$\Delta(\tau, y) = y \left(1 - r \left(1 - \frac{\tau}{T} \right)^{\zeta_{num}} \right), \quad (4.50)$$

where r denotes a random number in the range of $[0, 1]$, τ denotes the present generation number of the population, T denotes the maximum generation number of the population and ζ_{num} is a system parameter that determines the degree of non-uniformity.

For the GA operation, including normalized geometric ranking selection, blend- α crossover and non-uniform mutation are used in this chapter.

4.6 Results and Discussion

4.6.1 Result Classification with FFT (Feature Extractor) and GA-ANN (Classifier)

The results of the classification start with the FFT as the feature extractor and the GA-ANN as the classification algorithm to classify in three tasks classification out of six mental cognitive tasks (math, letter, cube, count, tone, and navigate).

The features dataset per subject consists of 450 units for each mental task and 1350 for the total of 3-tasks classification. This is divided into a half portion for the training set and the same amount for the testing set. The GA is employed to optimize the parameter of the neural network. The number of hidden neurons is changed using a value varying from 4 to 40 per training session of the neural network for each subject in order to find the best number that provides the highest fitness value or lowest MSE to achieve the highest classification accuracy. The population size used for the GA is 50, and the training is stopped when the training of the neural network reaches up to 2000 iterations. The GA parameters as shown in Table 4.1 are as follows: selection function using normalized geometric ranking with probability of selecting best chromosome is 0.08, crossover function based on Blend- α crossover with probability of crossover is 0.8 and non-uniform operation for the mutation function with probability of mutation is 0.1.

Table 4.1: Parameters GA and FPSOCM for ANN training

GA Parameters
Number of iterations = 2000
Population size = 50
Selection function = normalized geometric ranking; probability of selecting the best chromosome = 0.08
Crossover function = Blend- α crossover; probability of crossover = 0.8
Mutation function = non-uniform mutation; probability of mutation = 0.1

The training of the genetic algorithm-based neural network is done in combination of any three tasks classification of six mental cognitive tasks (math, letter, cube, count, tone and navigate), providing 20 combinations of training sets in total. In each combination, the training of the neural network was repeated ten times. This provided ten network weight parameters. As a result, each mental task is the averaging of ten classifications values as shown in Table 4.2, Fig. 4.10, and Table 4.3. The mean value of triplet mental tasks classification is provided in the table as the average classification accuracy.

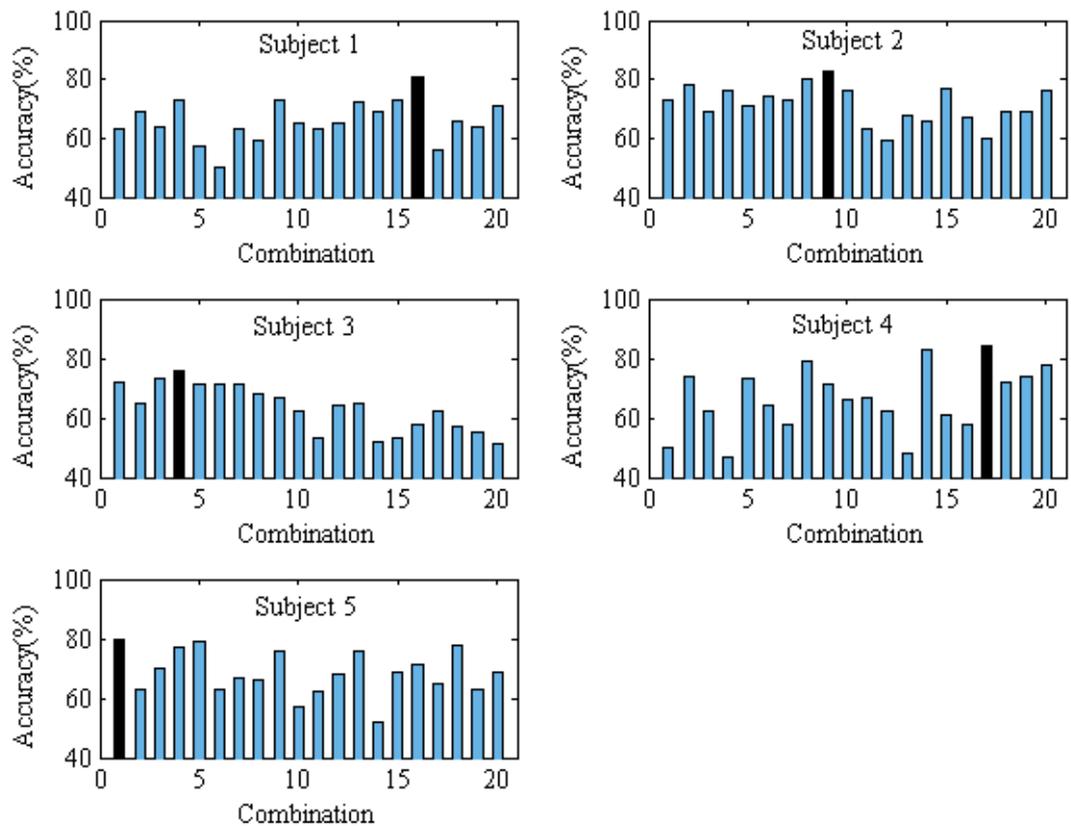


Figure 4.10: Plot of accuracy of five subjects with FFT (feature extractor) and GA-ANN (classifier)

Table 4.2: Accuracy of with FFT (feature extractor) and GA-ANN (classifier)

Number Combination	Percentage of correctly classified task; classification accuracy (%)																			
	Subject 1				Subject 2				Subject 3				Subject 4				Subject 5			
Triplet tasks combinations of = Math; 2=Letter; 3=Cube; 4=Count; 5= Tone; 6=Navigate	1 st task	2 nd task	3 rd task	Mean	1 st task	2 nd task	3 rd task	Mean	1 st task	2 nd task	3 rd task	Mean	1 st task	2 nd task	3 rd task	Mean	1 st task	2 nd task	3 rd task	Mean
1 Math(1)-Letter(2)-Cube(3)	51	86	51	63	81	75	63	73	83	84	50	72	48	38	63	50	73	94	77	81
2 Math(1)-Letter(2)-Count(4)	61	83	63	69	85	67	82	78	93	55	47	65	63	61	96	74	78	49	64	63
3 Math(1)-Letter(2)-Tone(5)	54	84	54	64	84	67	56	69	85	90	45	73	50	67	70	62	74	100	35	70
4 Math(1)-Letter(2)-Navigate(6)	60	76	83	73	77	80	71	76	88	77	64	76	56	35	50	47	66	99	65	77
5 Math(1)-Cube(3)-Count(4)	53	44	76	57	95	49	70	71	97	53	62	71	55	69	94	73	75	86	77	79
6 Math(1)-Cube(3)-Tone(5)	45	56	49	50	94	68	60	74	90	77	45	71	43	71	78	64	71	84	34	63
7 Math(1)-Cube(3)-Navigate(6)	62	56	71	63	85	56	78	73	90	61	63	71	53	59	64	58	59	80	62	67
8 Math(1)-Count(4)-Tone(5)	53	71	51	59	98	86	54	80	89	67	47	68	60	99	77	79	69	92	35	66
9 Math(1)-Count(4)-Navigate(6)	64	81	75	73	87	83	81	84	90	55	57	67	68	87	58	71	67	93	69	76
10 Math(1)-Tone(5)-Navigate(6)	52	59	83	65	88	61	79	76	85	43	58	62	48	78	72	66	63	46	62	57
11 Letter(2)-Cube(3)-Count(4)	78	62	48	63	71	48	70	63	68	45	45	53	43	65	93	67	44	84	57	62
12 Letter(2)-Cube(3)-Tone(5)	83	58	53	65	58	66	54	59	92	50	49	64	40	67	80	62	94	67	43	68
13 Letter(2)-Cube(3)-Navigate(6)	84	63	70	72	74	51	79	68	78	46	71	65	37	61	48	48	94	55	78	76
14 Letter(2)-Count(4)-Tone(5)	79	58	70	69	68	81	50	66	63	46	46	52	72	98	79	83	50	61	44	52
15 Letter(2)-Count(4)-Navigate(6)	80	64	74	73	68	80	81	77	52	43	64	53	49	87	45	61	49	63	93	69
16 Letter(2)-Tone(5)-Navigate(6)	84	79	84	82	66	53	83	67	80	44	49	58	35	85	55	58	100	35	80	71
17 Cube(3)-Count(4)-Tone(5)	57	57	55	56	57	67	55	60	51	80	54	62	73	95	87	85	76	76	42	65
18 Cube(3)-Count(4)-Navigate(6)	54	76	67	66	47	68	91	69	45	55	71	57	66	89	61	72	67	76	90	78
19 Cube(3)-Tone(5)-Navigate(6)	58	63	72	64	61	58	89	69	55	45	64	55	62	90	68	74	78	35	76	63
20 Count(4)-Tone(5)-Navigate(6)	73	65	73	71	82	52	95	76	54	46	52	51	88	88	56	78	92	35	80	69

Table 4.3: Chosen mental task of each subject with FFT (feature extractor) and GA-ANN (classifier)

Subject	Chosen tasks	Best hidden neuron	Mean of accuracy (%)	Bit rate (bits/trial)
1	Letter(2)-Tone(5)-Navigate(6)	34	82	0.7
2	Math(1)-Count(4)-Navigate(6)	24	84	0.8
3	Math(1)-Letter(2)-Navigate(6)	28	76	0.5
4	Cube(3)-Count(4)-Tone(5)	26	85	0.8
5	Math(1)-Letter(2)-Cube(3)	36	81	0.7

The resulting classification accuracy, as shown in Table 4.2, indicates a variety of values of classification accuracy across different subjects as the inter subject variability changes. With a total of 20 combinations of results, each subject has its own favourite triplet mental task combination, which yields the highest classification accuracy between 76% and 85%. In detail, subject 1 has the best 3-tasks classification between mental letter composing, ringtone imagery and spatial navigation with classification accuracy around 82%. Subject 2 achieved the highest accuracy based on a combination of mental arithmetic, visual counting and spatial navigation with accuracy at 84%. Subject 3 achieved accuracy at 76% of best triplet tasks with mental arithmetic, letter composing and spatial navigation. Next, subject 4 has the best classification accuracy between mental Rubik's cube rolling, visual counting and familiar ring tone imagery with accuracy at 85%. Subject 5 has the best classification accuracy at 81% between mental arithmetic, letter composing and Rubik's cube rolling. The table also shows alternative combination tasks on each subject with an average classification accuracy result above 70%. This could be used as the additional chosen combination.

Fig.10 and Table 4.3 provide a summary of the favourite triplet tasks for each individual. This provides the best suitable combination task and matching their circumstances and background. Generally, with user preferable three tasks classification, a satisfactory higher accuracy is achieved at between 76% and 85%.

Fig. 4.11 shows the selected best fitness value plotting, which has steady increasing value toward the highest point for five subjects involved in the study. The plotting of the number of hidden neuron versus classification accuracy in shown is Fig. 4.12. Each hidden neuron value provides a different result. The best chosen hidden neuron value which gives the highest classification accuracy for five subjects is between 24 and 36.

The information transfer rate of the classification is also provided in Table II, which has value between 0.5 to 0.8 bits per trial. The bit rate can also be illustrated as the number of bits per minutes by multiplying the bit rate (bits/trials) by system speed (trials/min). For example, the speed classification of the system is set every second which means 60 times in a minute. Therefore, the value bit rate 0.5- 0.8 bit/trial corresponded to 30-48 bits/min. A faster bit rate could be implemented by increasing the speed of the system to classify each mental task.

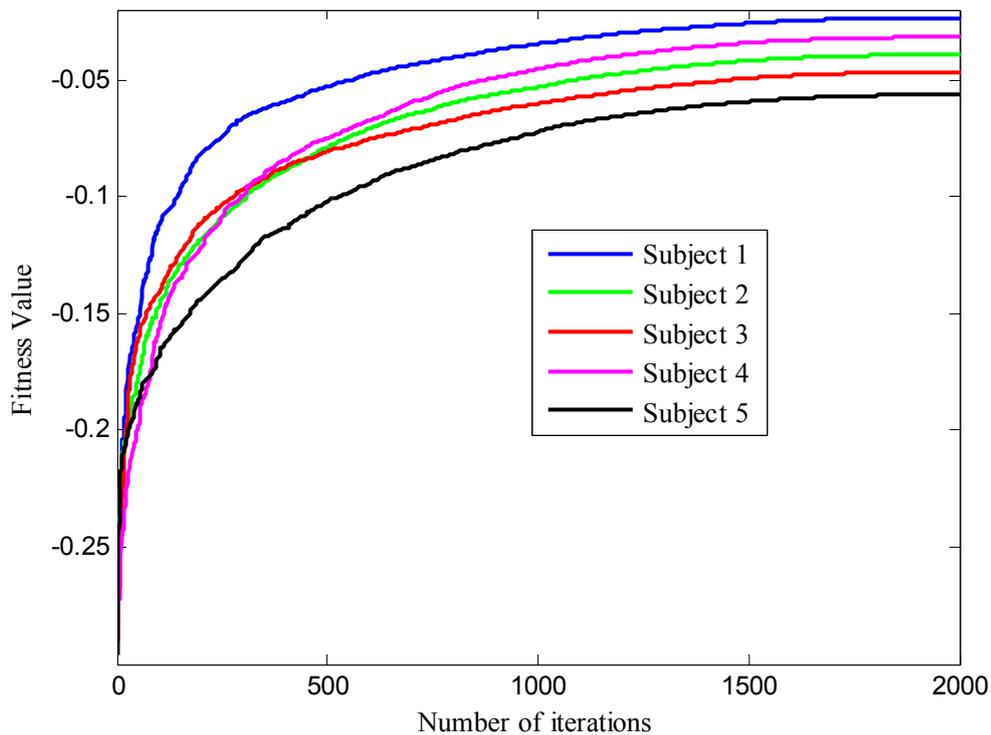


Figure 4.11: Plot fitness value of five subjects with FFT (feature extractor) and GA-ANN (classifier)

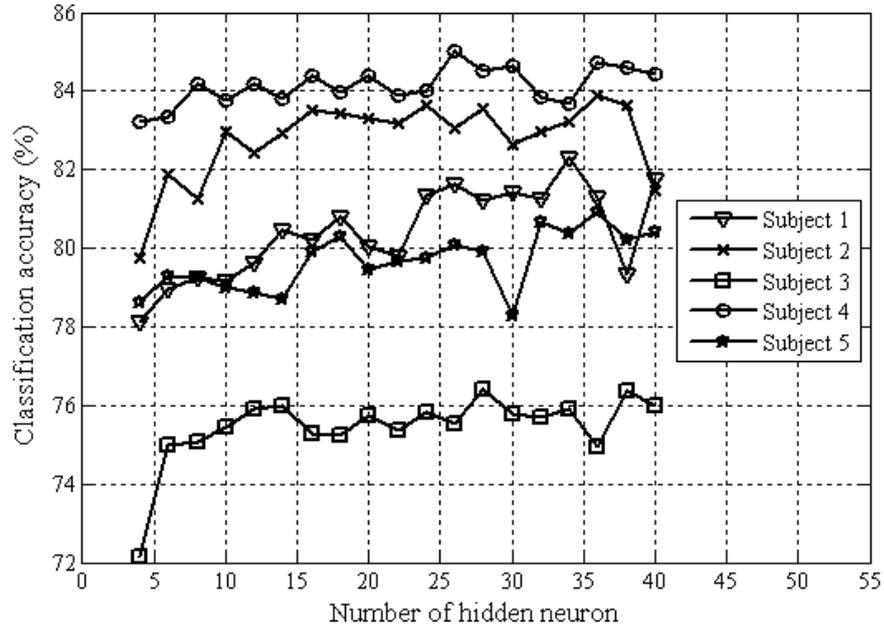


Figure 4.12: Number of hidden neuron vs. accuracy of five subjects with FFT (feature extractor) and GA-ANN (classifier)

4.6.2 Testing of the HHT Algorithm for Feature Extractor

For the initial testing of the HHT, a known combination of four sinusoidal signals is used with the equation as follows:

$$x(t) = \sin\left(\frac{\pi}{10}t\right) + 0.5 \sin(2\pi t) + \sin(10\pi t) + 0.5 \sin(40\pi t) \quad (4.50)$$

$$x_1(t) = \sin\left(\frac{\pi}{10}t\right) \quad (4.51)$$

$$x_2(t) = 0.5 \sin(2\pi t) \quad (4.52)$$

$$x_3(t) = \sin(10\pi t) \quad (4.53)$$

$$x_4(t) = 0.5 \sin(40\pi t) \quad (4.54)$$

The test signal $x(t)$ in equation (4.50) composes four components of sinusoid signals: 1) $x_1(t)$ in equation (4.51) as the first component, has an amplitude of 1 and frequency oscillation at 0.05 Hz, 2) $x_2(t)$ in equation (4.52) as the second component, has amplitude of 0.5 and frequency oscillation at 1 Hz, 3) $x_3(t)$ in equation (4.53) as the third component, has amplitude of 1 and frequency oscillation at 5 Hz, 4) $x_4(t)$ in equation (4.54) as the fourth component, has amplitude of 0.5 and frequency oscillation at 20 Hz.

The result of the EMD process is shown in Fig.4.13. EMD as the first step of HHT for this signal is composed correctly of five IMFs with the first four IMFs representing the four defined sinusoidal signals at different frequencies and the fifth IMF representing the residue.

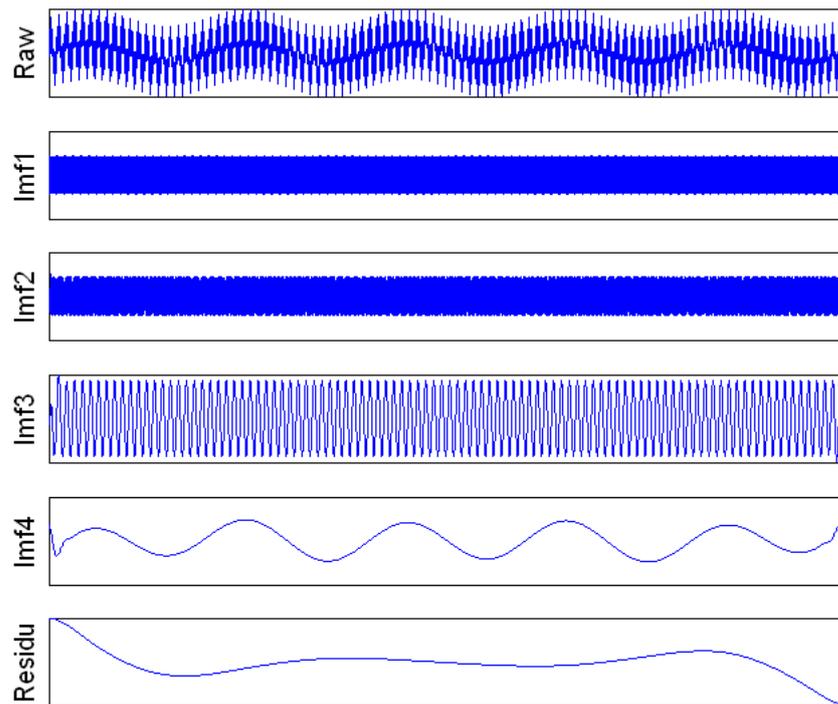


Figure 4.13: EMD process for test signal $x(t)$

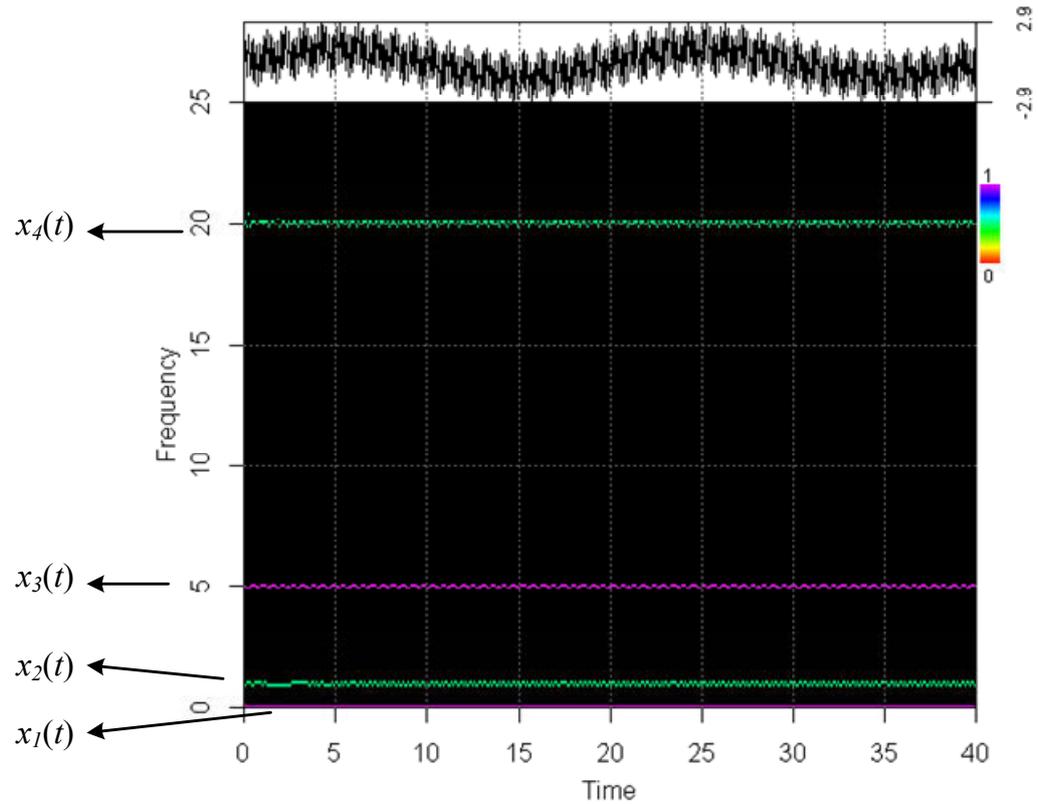


Figure 4.14: HHT process (HT spectrum) for test signal $x(t)$

The second process of the HHT is to apply the HT to the IMFs to provide instantaneous frequency and amplitude. A plot on Fig. 4.14 shows the Hilbert spectrum, or HHT spectrum, as the amplitude distribution on the time-frequency plane. It shows a clearly four frequency bands and amplitude that represents the four components of the signals. The very bottom purple signal is the $x_1(t)$ component of the signal $x(t)$ that has a frequency of 0.05Hz with amplitude at 1 (purple). The top of $x_1(t)$ is the green signal for $x_2(t)$ that has a frequency of 1 Hz and amplitude at 0.5 (green). The upper signal of the $x_2(t)$ is the purple signal for $x_3(t)$ that has a frequency of 5 Hz with amplitude of 1 (purple). The last signal is $x_4(t)$ that has a frequency of 20 Hz with amplitude of 0.5 (green). This shows the HHT algorithm is able to recognize the whole four components of the test signals, including their frequencies and amplitudes.

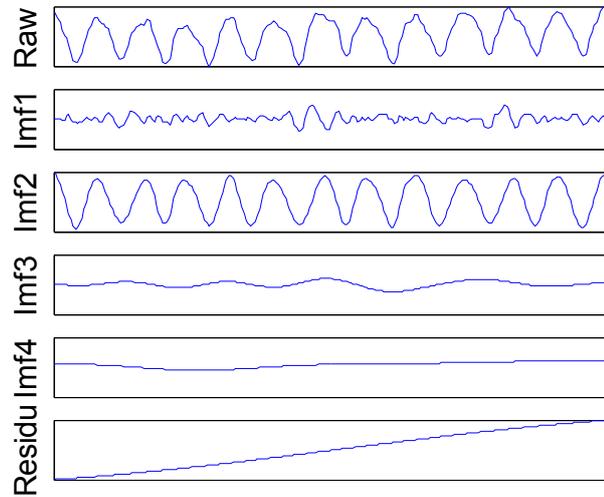


Figure 4.15: EMD process for eyes closed signal at 1s time-window

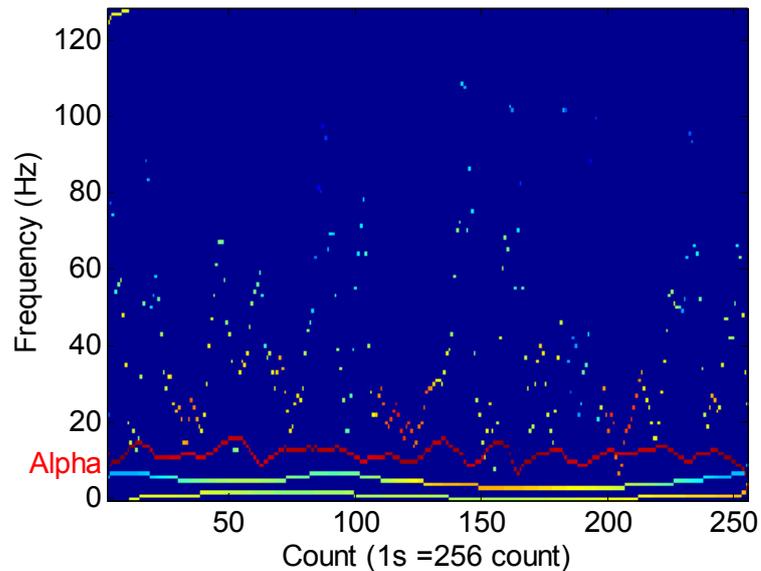


Figure 4.16: HHT process (HT spectrum) of eyes closed signal at a 1s time-window

The next testing of the HHT is to apply the algorithm for the EEG signal during eyes closed. It has been known that during the eyes closed task, there is a dominant feature in the alpha band of EEG (8-13Hz) (Craig et al. 1999; Craig et al. 2002; Craig et al. 2000). This unique feature can be used for further testing to ensure the HHT method

is correctly converting the raw EEG signal into correct features. The segmented EEG feature data (eyes closed task) was processed and converted into a series of IMFs and residue in the EMD process, as shown in Fig. 4.15. This is followed by applying the HT method to the IMFs, resulting in the amplitude and instantaneous frequency as functions of time. The plotting of the HHT spectrum in Fig.4.16 for the eyes closed task shows a clear dominant feature of the instantaneous frequency of the alpha EEG band (8-13Hz). This shows the features extraction method has correctly converted EEG data for the eyes closed task into the proper feature.

4.6.3 Result Classification with HHT (Feature Extractor) and GA-ANN (Classifier)

Next, the number of EEG channels is reduced from eight to two channels. GA-ANN training is performed based on the chosen mental task for each subject. Two features extraction methods, FFT and HHT, are compared to give a better features extraction algorithm. The result for the two channels' combination of the chosen subject specific task is provided in Table 4.4. To give the best accuracy, each preferable task on different subjects has the difference of the best two channels.

In detail, subject 1 with the chosen triplet task (letter-tone-navigate) has the best accuracy using O1-T4 pair with improved accuracy from 65% (FFT) to 77% (HHT). Another option is at T3-T4 with improved accuracy at 75 % (HHT). Subject 2 with the chosen three tasks (math-count-navigate) also provided an improvement of best accuracy from 67% (FFT) to 71% (HHT) using C3-T4 and optional P3-T4 with improved accuracy to 70% (HHT) from 65 % (FFT). Subject 3 with three tasks (math-letter-navigate) has best improved accuracy if using HHT with pairs: C3-T4 at 71% from 69% (FFT). Subject 4 has more channel pairs with accuracies above 70%, including C3-T4, P3-T4, O1-T4 and T3-T4. The best pair is O1-T4 location with improved accuracy at 79% using the HHT compared to FFT at 77%. Subject 5 also has more channel pairs and an improved accuracy using the HHT method, such as: C3-O2, C3-T4, P3-O2, O1-O2, O1-T4 and T3-O2 with the best accuracy at 84% (HHT)

improving from 79% (FFT) using O1-T4 channel. The rest of the channel combinations for each subject have improved accuracy using the HHT with the accuracy lower than 70%.

Generally, the resulting accuracies with only two channels for five subjects using FFT is lower than the original eight channels classification at values between 65 % and 79%. However, by using the HHT based feature extractor, these accuracies are improved with values between 70% and 84% across five subjects with two channels.

The best chosen mental task for each subject and the combination two channels EEG that provided improved accuracy at above 70%. Subject 1 has the best two channels at O1-T4 and T3-T4 using mental tasks: letter, tone, and navigation. Subject 2 has the best two channels at C3-T4 and P3-T4 using three mental tasks: math, count and navigate. Subject 3 has the best two channels at C3-T4 using mental tasks: cube, count, and tone. Subject 4 has the best two channels at C3-T4, P3-T4, O1-T4 and T3-T4 using mental tasks: cube, count and tone. Subject 5 has the best two channels at C3-O2, C3-T4, P3-O2, O1-O2, O1-T4 and T3-O2.

Comparisons on the classifiers (GA-ANN and ANN) and feature extractors (FFT, HHT) are elaborated in Table 4.5. The type of the ANN is the same one used in chapter 3 which is ANN trained with the Levenberg-Marquardt algorithm. The classifier comparison has classification accuracies for a chosen three mental tasks, which were lower using ANN compared to GA-ANN classifiers for both FFT and HHT feature extractors. In regards to the feature extractor comparison, the accuracies using the HHT as the feature extractor and ANN as the classifier were also higher compared to accuracy results with the feature extractor using FFT in the same ANN classifier. As a result, in this study, GA-ANN as the classifier and HHT as the feature extractor provided improved classification accuracy compared to ANN as the classifier and FFT as the feature extractor.

Table 4.4: Classification accuracy of chosen mental task of 2 channels EEG combinations and comparison feature extractors (FFT and HHT) with GA-ANN as classifier

Subjects	Chosen Subject Combination	Features Extractor	Two channels Combination															
			C3-C4	C3-P4	C3-O2	C3-T4	P3-C4	P3-P4	P3-O2	P3-T4	O1-C4	O1-P4	O1-O2	O1-T4	T3-C4	T3-P4	T3-O2	T3-T4
S1	Letter(2) - Tone (5) - Navigate (6)	FFT	53	53	48	56	51	51	53	59	55	59	57	65	57	58	57	70
		HHT	54	56	56	72	57	57	56	73	56	61	58	72	57	59	58	75
S2	Math(1) - Count (4) - Navigate (6)	FFT	48	51	54	67	43	48	57	65	47	51	59	67	52	51	60	66
		HHT	48	52	54	71	47	52	54	70	49	56	58	68	51	55	58	68
S3	Math(1) - Letter (2) - Navigate (6)	FFT	44	45	45	69	48	47	46	64	45	43	46	65	48	49	52	67
		HHT	52	52	49	70	51	52	49	66	46	48	46	64	51	53	53	66
S4	Cube(3) - Count (4) - Tone (5)	FFT	51	50	61	68	52	57	64	72	61	67	58	77	53	50	64	68
		HHT	51	49	60	72	54	55	65	73	68	68	68	79	50	51	63	70
S5	Math(1) - Letter (2) - Cube (3)	FFT	48	49	69	62	47	49	71	60	65	64	73	79	47	52	68	58
		HHT	54	56	74	71	51	55	75	66	69	69	78	84	53	55	75	64

Table 4.5: Comparisons between classifiers ANN and GA-ANN; between feature extractor FFT and HHT

Subjects	The chosen three task 1=Math; 2=Letter 3=Cube; 4=Count 5= Tone; 6=Navigate	Feature Extraction		C3-O2		C3-T4		P3-O2		P3-T4		O1-O2		O1-T4		T3-O2		T3-T4		
		FFT	HHT	ANN	GA-ANN															
S1	Letter(2) - Tone (5) - Navigate (6)	FFT												60	65			65	70	
		HHT													73	77			72	75
S2	Math(1) - Count (4) - Navigate (6)	FFT		61	67			63	65											
		HHT		68	71			65	70											
S3	Math(1) - Letter (2) - Navigate (6)	FFT		65	69															
		HHT		68	70															
S4	Cube(3) - Count (4) - Tone (5)	FFT		63	68			68	72					70	77			66	68	
		HHT		70	72			70	73					75	79			69	70	
S5	Math(1) - Letter (2) - Cube (3)	FFT		67	69	58	62	69	71					70	73	71	79	65	68	
		HHT		71	74	67	71	71	75					73	78	77	84	70	75	

4.7 Conclusion

A genetic algorithm has been successfully applied for the optimal training of the neural networks in order to provide classification outputs of three mental non-motor imagery tasks for the application of wheelchair movement control. The results show that each subject who participated in the study was able to have their own best triplet of mental task combination with a classification mean accuracy between 76% and 85% and a bit rate value of around 0.5 to 0.8 bits per trial. This would give more flexibility to select a suitable combination of non-motor imagery mental tasks as an alternative solution for disabled individuals who could not perform well using motor imagery tasks of a BCI system.

Two-channel mono-polar EEG classification has been successfully applied as a replacement to the original eight channels to discriminate three mental non-motor cognitive tasks tested for five subjects. As a result, two EEG channels with fewer electrodes provide more portability and more convenient setting-up in the practical BCI wheelchair control especially for severely disabled individuals. The original eight channels classification resulted in accuracies between 76% and 85%. With two channels, the accuracy using FFT features measured between 65% and 79%. Moreover, the HHT based feature extraction method provides a better performance compared to the FFT based method with an improved accuracy between 70% and 84%.

In this chapter, for the classifier, GA-ANN provided improved accuracy for all subjects compared to ANN, and HHT provided improved accuracy for all subjects compared to FFT. An Advanced BCI algorithm will be discussed in the next chapter together with the study working with patients with tetraplegia.

Chapter 5

Advanced BCI System using Fuzzy Particle Swarm Optimization of Artificial Neural Network

5.1 Introduction

Chapter 4 was focused on the features extractor method and classifier to optimize the BCI using the mental tasks. In this chapter, another advanced classifier method will be used to find improvement of the classification accuracy of the detecting BCI using the mental tasks. For comparison reasons, different classifiers and feature extractors will be used.

The reason this study is continuing further with the BCI using mental tasks are as follows:

- As discussed in chapter 2 (literature review), mental task imagery is the same as the popular sensorimotor rhythm (SMR)-based BCI which relies on the spontaneous mental signals initiated by the users themselves (self-paced system). So, both share the same advantage of not using the external stimuli if compared to the other types of BCI (SSVEP and P300). In mental task-based BCI, the user just needs to imagine the non-motor imagery task, such as letter composing, arithmetic and figure rotation. These tasks are not complicated and most people are familiar with them.
- Although many reports in SMR-based BCI research can be found, there are still some people who are unable to use SMR, and this is known as the BCI illiteracy phenomenon (Allison & Neuper 2010).
- The selective SMR or motor imagery task defects in severely disabled patients were also reported (Conson et al. 2008).
- Furthermore, individuals who have been paralysed or have been amputees for years may be unable to perform SMR tasks effectively (Birch, Bozorgzadeh & Mason 2002; Curran et al. 2004). The SMR task would be problematic for people with impairment in the particular area, such as stroke patients, who mostly have impairment on the motor cortex area.

One of the applications of the BCI is providing an alternative solution for hands-free wheelchair control to assist severely disabled individuals who are unable to move their body or head. Basically, to drive a wheelchair using a thought controller, at least three mental commands are needed to provide wheelchair steering control of turning left, turning right and moving forward (Craig & Nguyen 2007; Millan et al. 2009). The backward command is not used here for safety reasons. As a result, for the BCI using mental tasks paradigm, three mental tasks can be used for the three wheelchair commands. Each mental task is mapped into a particular command to control the wheelchair, as illustrated in Fig. 5.1. This is started by data collection using EEG for the mental tasks. The data is processed by a signal pre-processing module using the digital signal processing. Next, a features extraction module transforms the signal into features to be used for the classification algorithm. Advanced optimization technique is used

here to improve the ANN classifier to classify the three mental tasks which are mapped into three wheelchair commands.

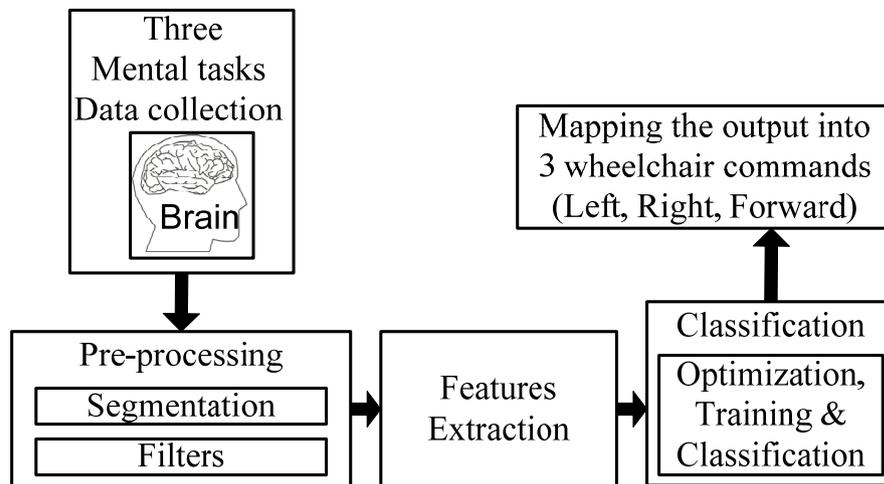


Figure 5.1: Components of mental tasks-based BCI for wheelchair commands

However, most of the results of BCI classification experiments, especially with the mental task non-motor imagery BCI, have reported on the able-bodied only, without including individuals with severe disabilities, who are an important target group for BCI technology. This chapter includes the experiment for both groups for comparison purposes.

As was previously mentioned, using FFT as the feature extractor requires the system to be linear and periodic (stationary). Data from EEG has the tendency to be a non-linear and non-stationary signal. One of the ways to overcome the drawback with the FFT feature extractor, a time-frequency wavelet analysis can be explored. In wavelet analysis, a characteristic time-limited pulse shape called the mother wavelet is used to construct a template. A different mother wavelet can be found with each, and has specific time-frequency characteristic and mathematical properties. Just as FFT provides an efficient computational of Fourier Transform for digital signals, the discrete wavelet transform (DWT) provides an efficient computation of wavelet using specific scale and shift factors that minimize redundancy in the time-frequency representation.

This chapter uses the wavelet analysis as the feature extractor for comparison purposes. Moreover, the HHT discussed in the previous chapter is also known as a time-domain analysis that provides improved results compared to the conventional FFT.

Before the classification using ANN, a network training/learning is required. GA is used in the previous chapter to optimize the parameters of the ANN to form as GA-ANN. Swarm intelligence is a new approach to improve optimization problems. This is done with inspiration from the social behaviours of animals or insects (Kennedy 2010). The particle swarm optimization (PSO) has comparable or even superior search if compared with other population-based stochastic optimization methods such as EA. Furthermore, PSO has memory of the previously visited best position, which differs from EA that does not keep the important information such as population changes. Recently, an improved optimization technique using a fuzzy particle swarm optimization with the cross-mutated operation (FPSOCM) was proposed (Ling et al. 2012). Here, a fuzzy inertia weight with the value is obtained from a fuzzy inference system. Fuzzy logic is good in representing expert knowledge and experience in some linguistic rules, and can be easily understood by human beings. Using fuzzy inference system to determine the inertia weight of PSO, the characteristic function of the inertia becomes nonlinear. The nonlinear characteristic of the inertia weight offers better solution quality. Moreover, a new operation called the cross-mutated (CM) operation can effectively handle the drawback of trapping in local optima. With the new FPSOCM optimization method being proposed, this study combines the FPSOCM with the ANN to form a FPSOCM-ANN for the BCI using mental tasks classifier.

The contribution of this chapter is as follows: first, different feature extractors are explored, including FFT, wavelet, HHT to find a better feature extractor. Second, two optimization techniques were used for the ANN training, including the GA and FPSOCM. These optimization techniques combined into the ANN classifier to form GA-ANN and FPSOCM-ANN classifiers for mental task-based BCI in order to find an improved classifier with the highest classification accuracy. The mental tasks used are mental letter composing, mental arithmetic calculation and mental figure Rubik's cube

rolling forward. These mental tasks can be mapped for the three wheelchair commands: left, right and forward. An additional eyes closed task is recorded for the feature extractor testing and on-off command. For comparison purposes, other classifiers, such as support vector machine (SVM), linear discriminant analysis (LDA) and linear perceptron are presented. Third, this paper includes five able-bodied subjects and five patients with tetraplegia with comparison for both groups. Fourth, different time-windows of data are investigated to find the best data windowing with an improved result of classification accuracy. Fifth, for practical reasons, results of combinations of two channels classification are presented to find a suitable two channels EEG for mental task-based BCI.

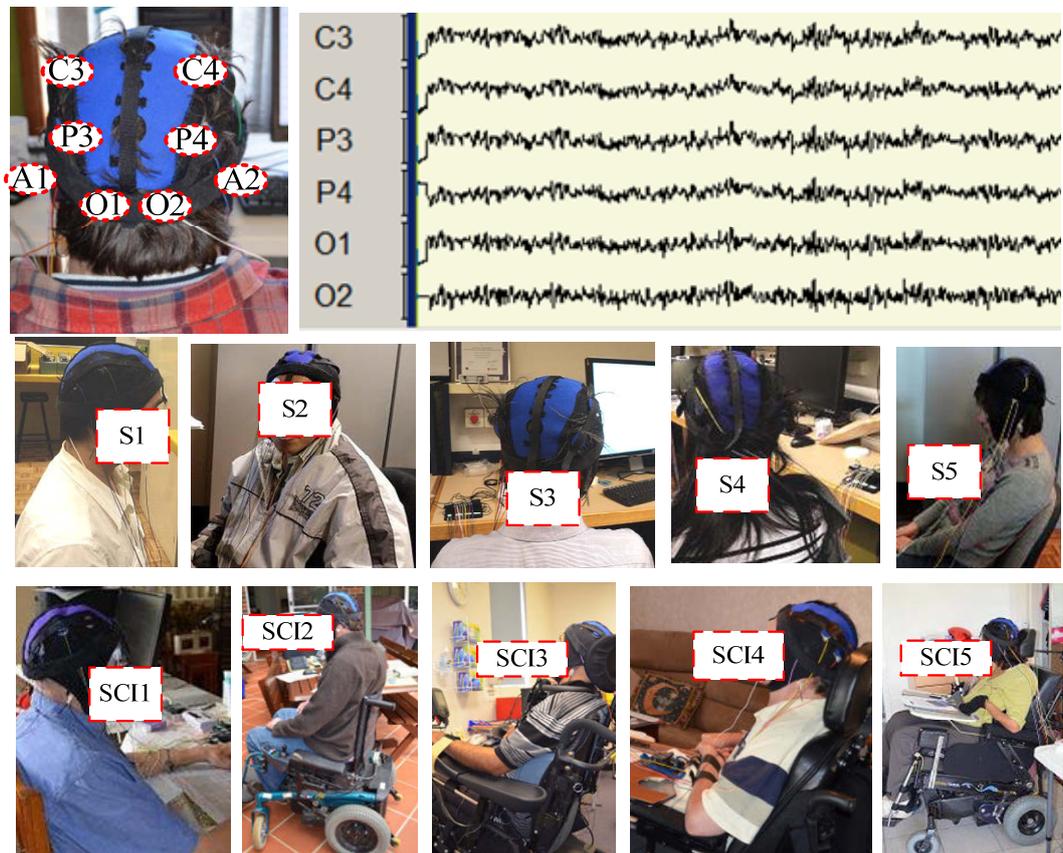
5.2 Data Collection

The Human Research Ethics Committee of University of Technology, Sydney approved this study. A total of ten participants were involved: five able-bodied subjects (S1-S5) aged between 25 and 35 years and five patients with tetraplegia (T1-T5) aged between 45 and 80 years who suffer a high-level of spinal cord injury (SCI) in the cervical area at level C3, C4, C5 and C6 with the details as shown in Fig. 5.2 and Table 5.1.

A commercial EEG system, Siesta from Compumedics, was used with the sampling rate of the system at 256 Hz. This study used 6 channels with the electrodes positioned at locations C3, C4, P3, P4, O1 and O2. The left earlobe (A1) was used as the reference and the GND electrode was attached to the right earlobe (A2) as shown in Fig 5.2. This configuration refers to the standard 10-20 electrode montage system (Klem et al. 1999).

Table 5.1: Details of participants: able-bodied subjects and patients with tetraplegia

Participants	Age	Description
S1, S2, S3,S4, S5	25-35	Healthy Subjects
SCI1	80	Tetraplegia; cervical SCI at level C3 and C4
SCI2	59	Tetraplegia; cervical SCI at level C5 and C6
SCI3	50	Tetraplegia; cervical SCI at level C3 and C4
SCI4	55	Tetraplegia; cervical SCI at level C3 and C4
SCI5	45	Tetraplegia; cervical SCI at level C5 and C6

**Figure 5.2:** Experiment setup of EEG-BCI with patients with able-bodied subjects and patients with tetraplegia

To ensure a proper recording of the EEG signal for participants in this study, there are some procedures to follow:

- First, the Compumedics-Siesta uses the standard wet electrodes (gold). In a standard wet electrode, to ensure a proper electrical contact, EEG gel was applied to keep the impedance low. The subjects were asked to wear the EEG cap to tighten the electrodes at the same position to ensure no electrodes shifted during usage.
- Second, the impedance was measured and maintained below $5k\Omega$ to justify quality of electrodes connection with supported software as shown in Fig.5.3.
- Third, participants were asked to keep their eye blinks and unnecessary movements minimal during the data collection. Data with strong presence of artefacts will be discarded.

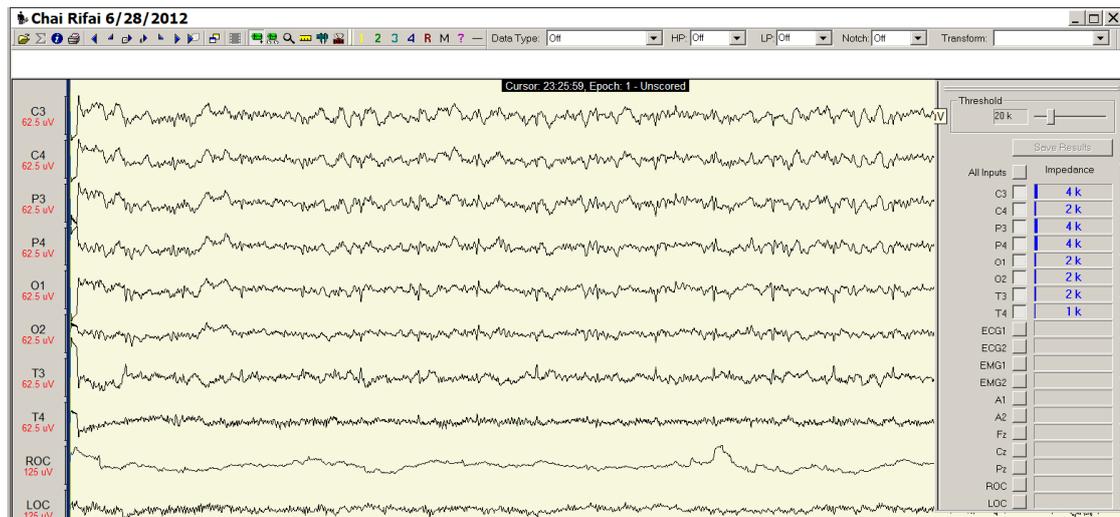


Figure 5.3: Impedance checking using the recording software

A total of three non-motor imagery mental tasks were used, including mental letter composing, arithmetic and a figure of Rubik's cube rolling. Prior to the beginning of the session, participants were shown a video as a guidance to perform mental tasks for the standard protocol of measurement. Fig. 5.4 shows the figure which was taken from the

video guidance. Participants were asked to mentally perform the following tasks: composing words in the mind, solving multiplication problems, and imagining a Rubik's cube rolling in a forward direction. Additional eyes closed and open tasks were also recorded for testing. The experiment was recorded at 10 sessions for each mental task with each session lasting 15s of recording time. The first 3s of data were discarded for preparation time, and the remaining 12s of data were used for further signal processing.

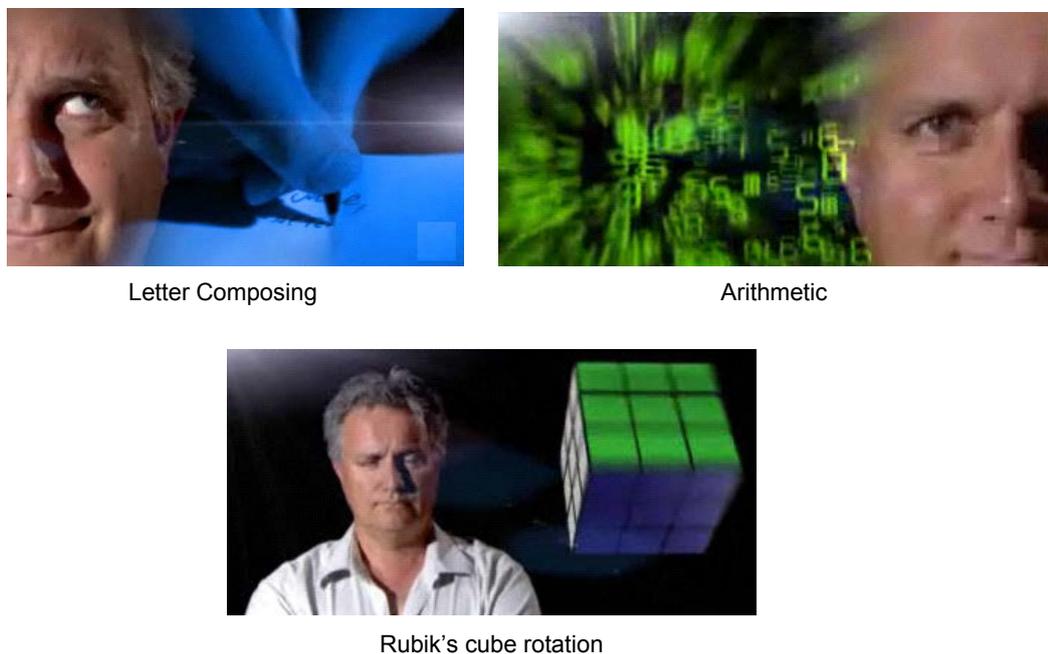


Figure 5.4: Pictures from the video guidance for the participants

5.3 Signal Pre-Processing and Feature Extraction

5.3.1 Signal Pre-processing

For digital signal pre-processing, different moving time-window segmentations from 1s until 10s, were applied with a quarter second segment. As a result, for 10 sessions in each mental task, each participant provided data of 450 sets for a 1s time-window, 410 sets for a 2s time-window, 370 sets for a 3s time-window, 330 sets for a 4s

time-window, 290 sets for a 5s time-window, 250 sets for 6s time-window, 210 sets for 7s time-windows, 170 sets for 8s time-window, 130 sets for 9s time-window and 90 sets for 10s time window.

This was further processed by applying digital signal processing filters using a Butterworth band-pass filter (0.1-100 Hz) and a Butterworth notch filter at 50 Hz for a signal to noise ratio improvement. The feature extraction process was the next step after completing the signal pre-processing process. For comparison purposes different feature extractors were used in this chapter including FFT, Wavelet, original HHT with its EMD process (HHT-EMD) and HHT with improved EMD, known as ensemble EMD (HHT-EEMD).

5.3.2 Feature Extractor using Hilbert-Huang Transform

(HHT)

The HHT as the features extraction has been discussed in detail in chapter 4. This method is a time-frequency method consisting of two parts. During the first part, empirical mode decomposition (EMD), the signal is decomposed into the implicit mode function (IMF), putting forward the scale characteristic imbedded in the signal. The second part, the Hilbert transform, is applied to the IMF's, yielding a time-frequency representation (Hilbert spectrum) for each IMF (Huang et al. 1998). HHT has been applied in different fields of science; geophysics (Huang, Shen & Long 1999), image processing (Tan et al. 2006), mechanical fault diagnosis (Li, Zheng & Tang 2005) and financial applications (Huang et al. 2003). The EMD process is basically a sifting process comprising the following procedures:

- 1) Identification of all extrema (maxima and minima) of the signal $x(t)$;
- 2) Generation of the upper envelope ($E_u(t)$) and lower envelope ($E_l(t)$) via cubic spline interpolation among all the maxima and minima, respectively.
- 3) Point by point averaging of the two envelopes to compute a local mean series, $m(t) = (E_u(t) + E_l(t)) / 2$.

- 4) Subtraction of $m(t)$ from the data to obtain an IMF candidate, $c(t) = x(t) - m(t)$ and checking the properties of $c(t)$ as follows:
 - a. If $c(t)$ is not an IMF, for instance if it does not satisfy the previously defined properties, replace $x(t)$ with $c(t)$ and repeat the procedure from step 1.
 - b. If $c(t)$ is an IMF, evaluate the residue, $r(t) = x(t) - c(t)$.
- 5) Repeat the procedure from step 1 to 4 by sifting the residual signal. The sifting process ends when the residue $r(t)$ satisfies a predefined stopping criterion.

Having obtained the IMFs using the EMD method, the Hilbert transform was applied to each IMF component to obtain the instantaneous frequency and amplitude, which can be plotted in a three-dimensional plot with the amplitude at its height in the time-frequency plane. The process of the EMD with the HT in this study is known as HHT, with its original EMD process, Hilbert-Huang transform using empirical mode decomposition (HHT-EMD).

There are improvements on the original HHT, especially with the EMD analysis available (Chen & Feng 2003; Wu & Huang 2009). The HHT using the ensemble empirical mode decomposition (EEMD) is the latest improvement on the EMD (Wu & Huang 2009). One of the issues of the original EMD is the frequent appearance of mode mixing, which is defined as a single IMF consisting of signals of widely disparate scales. The mode mixing is often a consequence of signal intermittency, which could cause aliasing in the time-frequency distribution and make unclear the physical meaning of individual IMF. To overcome the scale separation issue, a noise-assisted data analysis (NADA) method is used, called the ensemble EMD (EEMD), which defines the true IMF components as the mean of an ensemble of trials, each consisting of the signal plus a white noise of finite amplitude.

The principle of the EEMD is that the added white noise would populate the whole time-frequency space uniformly with the constituting components of different scales. When a signal is added to this uniformly distributed white background, the bits of signal of different scales are automatically projected in proper scales of reference established

by the white noise in the background. Each individual trial may produce very noisy results, for each of the noise-added decompositions consists of the signal and the added white noise. With the noise in each trial different in separate trials, it is cancelled out in the ensemble mean of enough trials which are shown in the following equation:

$$x_i(t) = x(t) + w_i(t) \quad (5.1)$$

where $x_i(t)$ denotes the i th artificial observation, $x(t)$ denotes the single data set, and $w_i(t)$ denotes the white noise. The EEMD is developed as follows:

- 1) Add a series of white noise to the data,
- 2) Decompose the data with added white noise into IMFs,
- 3) Repeat step 1 and step 2, but with different white noise series each time, and
- 4) Obtain the ensemble means of corresponding IMF of the decomposition as the final result.

The effects of the decomposition using the EEMD are that the added white noise series cancel each other, and the mean IMFs stay within the natural dyadic filter windows, significantly reducing the chance of mode mixing and preserving the dyadic property. This EEMD with the HT in this study is known as the HHT-EEMD method. For the EEG data, the spectrum of the HHT for the features covers the following EEG bands: δ (1-3Hz), θ (4-7Hz), α (8-13Hz) and β (14-30Hz) or in the frequency range from 1 to 30 Hz. Consequently, the input feature on each channel has 30 units and six EEG channels resulting in 180 units of input features and with two EEG channels, resulting in 60 units.

5.3.3 Other Feature Extractors for Comparison Purpose

5.3.3.1 Fast Fourier Transform (FFT)

The FFT features extractor, which has been discussed and used previously in chapters 3 and 4, is a spectral-based analysis, which requires the system to be linear and

periodic (stationary); otherwise, the spectrum of the FFT will make little physical sense. The FFT also requires linearity in its analysis. Many natural phenomena can be approximated by using the linear analysis, but they also have the tendency to be nonlinear whenever their variations become finite in amplitude. EEG is known as a non-linear and non-stationary signal, which is composed of many different frequency components as a function of time of the nature changes (Andrzejak et al. 2001; Sannelli et al. 2008).

5.3.3.2 Wavelet

The basic idea underlying wavelet analysis consists of expressing a signal as a linear combination of a particular set of wavelet transforms (WT) which is obtained by shifting and dilating a mother wavelet function. The decomposition of the signal leads to a set of wavelet coefficients. As a result, the signal can be reconstructed as a linear combination of the wavelet functions weighted by the wavelet coefficients. In order to obtain an exact reconstruction of the signal, an adequate number of coefficients must be computed. The key feature of wavelets is the time-frequency localisations, which means that most of the energy of the wavelet is restricted to finite time intervals. When compared to the FFT, the wavelet analysis varies the time-frequency aspect ratio to produce good frequency localisation at low frequencies and good time localisation at high frequency. This produces segmentation of the time-frequency plane that is appropriate for most physical signals, especially those of a transient nature, such as the EEG signal. The continuous wavelet decomposition is given as follows:

$$W_{\psi}(a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} x(t) \psi^* \left(\frac{t-b}{a} \right) dt \quad (5.2)$$

where $(.)^*$ is the complex conjugate and $\psi(t)$ is the analysing wavelet function, which have to satisfy certain restrictions. The two other parameters are a for scaling and b for

position. When comparing with the FFT, the wavelet transform substitutes for the window and the Fourier-like exponent function with a kernel wavelet.

The wavelet transform can be done in three different approaches, which include discrete wavelet transform (DWT), continuous wavelet transform and wavelet series. DWT is sufficient to decompose and reconstruct most power quality problems which provides enough information and offers high reduction in the computational time (Chui 1997). In DWT, the number of decomposition levels is chosen based on the dominant frequency components of the signal that correlates well with the frequencies necessary for classification of the signal which are retained in the wavelet coefficients. There are some issues with the wavelet transform when applied to nonlinear and non-stationary data. The wavelet transform depends on the predetermined wavelet function. This distorts the results, and the frequency distribution does not appear clear enough. Wavelet analysis is linear, whether it is discrete or continuous. As a result, it may not even fit for non-linear data.

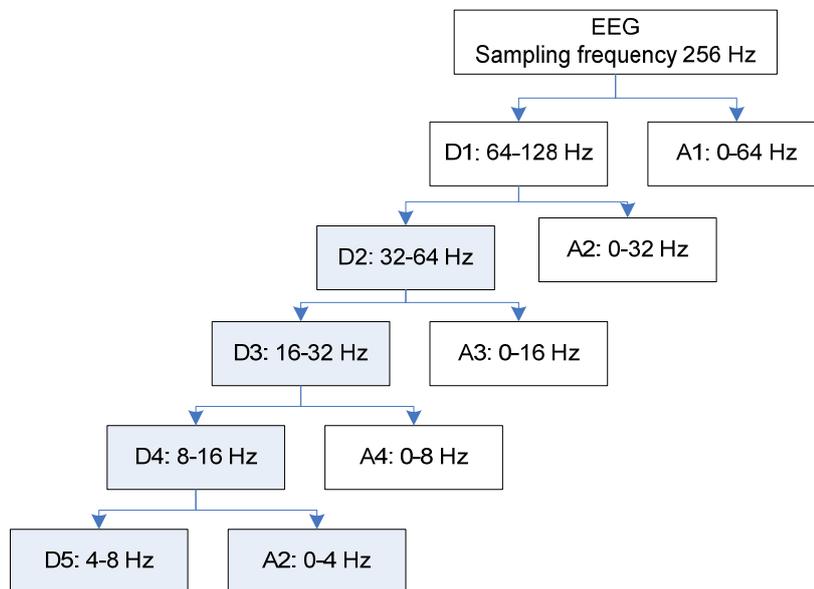


Figure 5.5: Wavelet decomposition of EEG signal with sampling frequency at 256Hz

The wavelet decomposition of the EEG recording is illustrated in Fig. 5.5 and Table 5.2 provides the equivalent of the wavelet decomposition of the EEG rhythms. With the EEG sampling frequency at 256 Hz, it needed a total of 5 levels of the wavelet decomposition and resulted in five EEG bands. The decomposition can be constructed from the Daubechies wavelet (Daubechies 1992) for mother wavelets to do the 5 levels of decomposition. Delta rhythm is reconstructed by the sum of the wavelet component at level 5 of the approximation (A5). The theta rhythm is reconstructed by the wavelet component at level 5 of the decomposition (D5). The alpha rhythm is reconstructed by the wavelet component at level 4 of the decomposition (D4). The beta rhythm is reconstructed by the wavelet component at level 3 of the decomposition (D3). The gamma rhythm is reconstructed by the wavelet component at level 2 of the decomposition (D2).

Table 5.2: EEG Frequencies Corresponding to Wavelet Decomposition

Decomposed Signal	Frequency range	EEG Rhythms
Decomposition 1 (D1)	64 - 128 Hz	-
Decomposition 2 (D2)	32 - 64 Hz	Gamma
Decomposition 3 (D3)	16 - 32 Hz	Beta
Decomposition 4 (D4)	8 - 16 Hz	Alpha
Decomposition 5 (D5)	4 - 8 Hz	Theta
Approximation 5 (A5)	0 - 4 Hz	Delta

5.4 Classification Algorithms

5.4.1 Fuzzy Particle Swarm Optimization with Cross-Mutated Operation of Artificial Neural Network (FPSOCM-ANN)

For the classification algorithm, an advanced method is used by combining the neural network with the latest proposed optimization technique. The core of the classification method in this study is the ANN as the non-linear algorithm. This chapter,

the same as in chapters 3 and 4, uses a three layers feed forward neural network as shown in Fig.5.6.

The ANN model and normalization are as follows:

$$z_k(x, w) = f_1 \left(b_k + \sum_{j=1}^m w_{kj} f_2 \left(b_j + \sum_{i=1}^n w_{ji} x_n^* \right) \right) \quad (5.3)$$

$$x^* = (x - x_{min}) / (x_{max} - x_{min}) \quad (5.4)$$

where f_1 and f_2 , are the activation functions and log-sigmoid function used in this paper, n refers to the number of input nodes, m refers to the number of hidden nodes, k refers to the number of output nodes, b_j and b_k are the biases, w_{ji} refers to the weight to the hidden unit y_j from input unit x_i , w_{kj} refers to the weight to output unit z_k from hidden unit y_j , x^* is the input features after normalization, x is the input features before normalization, x_{min} represents the minimum, and x_{max} refers to the maximum value of the input. The activation function uses log-sigmoid function; therefore, prior to ANN training, the features need to be normalized into the range of zero to one using equation (5.4).

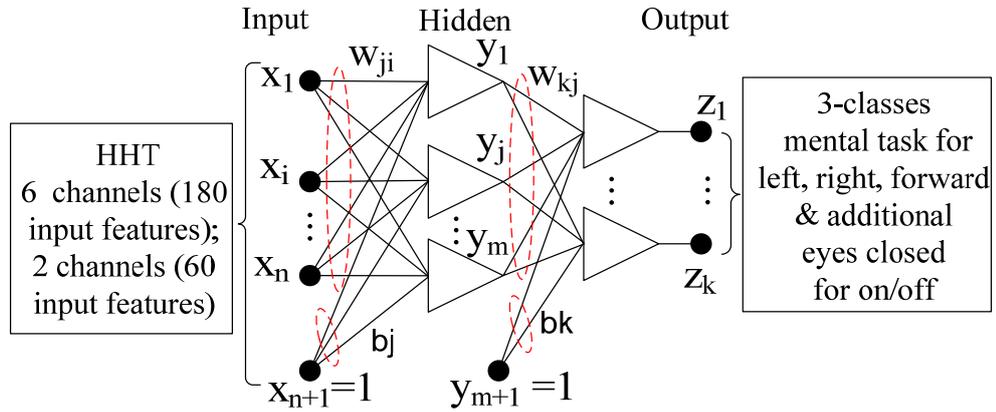


Figure 5.6: The ANN structure for mental task-based BCI

Recently, an improved optimization technique using fuzzy particle swarm optimization with the cross-mutated operation (FPSOCM) was proposed with its

application in industrial application of economic load dispatch (Ling et al. 2012). It uses the FPSOCM method to optimize the learning/training of the ANN to form a fuzzy particle swarm optimization with the cross-mutated operation of artificial neural network (FPSOCM-ANN). This FPSOCM-ANN classifier was used to classify three mental tasks in this study. More details about the FPSOCM will be discussed in the next section.

Particle swarm optimization (PSO) is a stochastic optimization algorithm that models the processes of the sociological behaviour associated with bird flocking and fish schooling (Kennedy 2010). The swarm is formed by a number of particles, and each of them transverses the search space looking for the global optimum. A PSO with constriction factor was introduced in (Clerc & Kennedy 2002) order to improve the searching ability over the standard PSO. The fuzzy particle swarm optimization with cross-mutated operation was proposed for the improvement of the PSO with the procedure as shown in Fig. 5.7.

```

begin
   $t = 0$ 
  Initialize  $X(t)$ 
  Define probability of CM operation  $p_{cm}$ 
  Output ( $f(X(t))$ )
  while (not termination condition) do
     $t = t + 1$ 
    output ( $t/T$ )
    output  $\| \zeta(t) \|$ 
    Find inertia weight  $\tilde{\omega}_k(t)$  using FIS
    Update velocity  $\mathbf{v}(t)$ 
    Find the control parameter  $\beta(t)$  by using FIS
    Generate a random number  $R_{cm}$ 
    if  $R_{cm} > p_{cm}$ 
      then Perform cross-mutated operation
    output ( $f(X(t))$ )
  end
  return the best particle,  $g$ 
end

```

Figure 5.7: The FPSOCM procedure

A fuzzy inertia weight $\tilde{\omega}(t)$ and a cross-mutated (CM) operation were introduced for performance searching improvement and to tackle the issue of trapping in local optima. The FPSOCM process was started by the initialization of the particle swarm $X(t)$ with generation number $t = 0$. The $X(t)$ was constructed with w_{ji} , w_{kj} , b_j and b_k of ANN. Each particle was evaluated by the cost-objective (fitness) function, $f(X(t))$. The probability of the CM operation for each particle (p_{cm}) was defined before the iteration sequences had begun. The value of the inertia weight ($\tilde{\omega}(t)$) is selected to improve the searching performance and is controlled by two inputs of the fuzzy inference system (FIS), the normalized standard deviation of the cost value among all the particles, $\|\zeta(t)\|$ and the iteration stage, t/T . After the $\tilde{\omega}(t)$ has been found, the particle velocity ($\mathbf{v}(t)$) was updated. This was continued by finding the control parameter, $\beta(t)$ of the CM by using the fuzzy inference system (FIS). The velocity of each element swarm particle was evaluated by the CM operation. A random particle (R_{cm}) in the range of 0 and 1 is generated. If the value of R_{cm} is more than the value of p_{cm} , the CM operation will be undertaken on that particular element. Next, a new particle swarm was generated. The process was continued and repeated until the iteration number (T), as defined in the beginning, was met. In detail, the fuzzy inertia weight $\tilde{\omega}(t)$ is governed by the following fuzzy rules:

$$\text{Rule } j : \text{IF } \|\zeta(t)\| \text{ is } N_1^j, \text{ AND } t/T \text{ is } N_2^j \text{ THEN } \tilde{\omega}(t) = \sigma_j, \quad j = 1, 2, \dots, \varepsilon \quad (5.5)$$

where N_1^j and N_2^j are fuzzy terms of rule j , ε denotes the number of rules, $\sigma_j \in [\omega_{win} \ \omega_{max}]$ is the singleton to be determined, here $\omega_{win} = 0.1$ and $\omega_{max} = 1.1$. The final $\tilde{\omega}(t)$ is given with the following:

$$\tilde{\omega}(t) = \sum_{j=1}^{\varepsilon} m_j(t) \sigma_j \quad (5.6)$$

Where

$$m_j(t) = \frac{\mu_{N_1^j}(\|\zeta(t)\|) \times \mu_{N_2^j}(t/T)}{\sum_{j=1}^{\varepsilon} (\mu_{N_1^j}(\|\zeta(t)\|) \times \mu_{N_2^j}(t/T))} \quad (5.7)$$

$\mu_{N_1^j}(\|\zeta(t)\|)$ and $\mu_{N_2^j}(t/T)$ are the membership function values corresponding to $\mu_{N_1^j}$ and $\mu_{N_2^j}$ respectively. A new velocity

$$v_j^i(t) = k \cdot \left\{ \tilde{\omega}(t) \cdot v_j^i(t-1) + \varphi_1 \cdot r_1 \cdot (p_j^i - x_j^i(t-1)) \varphi_2 \cdot r_2 \cdot (g_j^i - x_j^i(t-1)) \right\} \quad (5.8)$$

There are three membership functions for each input which are Low (L), Medium (M), and High (H). Five terms Very Low (VL), Low (L), Medium (M), High (H), and Very High (VH) are used as output singleton variables. Based on the characteristic of $\|\zeta(t)\|$ and t/T , the 9 linguistic fuzzy rules for determining $\tilde{\omega}(t)$ are as shown in Fig. 5.8.

Rule 1: IF $\ \zeta(t)\ $ is “L” AND t/T is “L”, THEN $\tilde{\omega}(t)$ is “VH”(=1.1)
Rule 2: IF $\ \zeta(t)\ $ is “M” AND t/T is “L”, THEN $\tilde{\omega}(t)$ is “H”(=0.85)
Rule 3: IF $\ \zeta(t)\ $ is “H” AND t/T is “L”, THEN $\tilde{\omega}(t)$ is “VH”(=1.1)
Rule 4: IF $\ \zeta(t)\ $ is “L” AND t/T is “M”, THEN $\tilde{\omega}(t)$ is “M”(=0.6)
Rule 5: IF $\ \zeta(t)\ $ is “M” AND t/T is “M”, THEN $\tilde{\omega}(t)$ is “M”(=0.6)
Rule 6: IF $\ \zeta(t)\ $ is “H” AND t/T is “M”, THEN $\tilde{\omega}(t)$ is “H”(=0.85)
Rule 7: IF $\ \zeta(t)\ $ is “L” AND t/T is “H”, THEN $\tilde{\omega}(t)$ is “VL”(=0.1)
Rule 8: IF $\ \zeta(t)\ $ is “M” AND t/T is “H”, THEN $\tilde{\omega}(t)$ is “VL”(=0.1)
Rule 9: IF $\ \zeta(t)\ $ is “H” AND t/T is “H”, THEN $\tilde{\omega}(t)$ is “L”(=0.35)

Figure 5.8: The Fuzzy rules for inertia weight, $\tilde{\omega}(t)$

The value of $\tilde{\omega}(t)$ is determined by the fuzzy inputs $\|\zeta(t)\|$ and t/T . The value of t/T represents the iteration stage. The value of $\tilde{\omega}(t)$ should be higher when the value of t/T is smaller so that a larger value of velocity of the particle element is given for global

searching. A large value of t/T implies a smaller value of velocity of the particle element for local searching and fine-tuning. Therefore, the value of $\tilde{\omega}(t)$ of the fuzzy rules 1, 2, and 3 (t/T is “L”) are larger than those of the rules 4, 5, and 6 (t/T is “M”). In rules 1 to 3, the searching process is in its early stage (t/T is “L”). When $\|\zeta(t)\|$ is “H”, the wide spread of particle location means a larger value of $\tilde{\omega}(t)$ should be given for enhancing global exploration. When $\|\zeta(t)\|$ is “L” in the early stage, however, the value of $\tilde{\omega}(t)$ is set to be larger, because the chance that the solution being trapped in a local optimum tends to be high. In rule 2, the value of $\|\zeta(t)\|$ is “M”, and the value of $\tilde{\omega}(t)$ is set to be slightly smaller than that in rules 1 and 3 in the early stage (t/T is “L”). In rules 4 to 6, the searching process is in its middle stage (t/T is “M”). The rationale for the suggested value of $\tilde{\omega}(t)$ is similar to that for rules 1-3. However, there is one difference that when $\|\zeta(t)\|$ is “L”, the value of $\tilde{\omega}(t)$ is smaller than that when $\|\zeta(t)\|$ is “H”. It is because the optimal solution may be found in the middle stage, where a smaller value of $\tilde{\omega}(t)$ is given. In rules 7 to 9, the searching process is undergoing a fine tuning process to reach optimal solution. Thus, when the value of $\|\zeta(t)\|$ is “L”, the locations of particles are close to one another and near the optimal solution, as a result, the smallest value of $\tilde{\omega}(t)$ is used.

Furthermore, a cross-mutated (CM) operation was used to merge the ideas of the crossover and mutation operation of the genetic algorithm (Michalewicz 1994) with the aim of helping the particles escape from a local optima. A control parameter $\beta(t)$ is introduced into the CM operation, which is used by other fuzzy rules. By using the CM operation, the performance of the PSO method improves, especially when it used to handle some multimodal optimization issues with many local minima.

The velocity of every particle element of the swarm will have a chance to undergo CM operation governed by a probability of CM operation, $p_{cm} \in [0 \ 1]$, which is defined by the user. The resulting velocity of a particle element under the CM operation is :

$$\bar{v}_j^i(t) = \begin{cases} (1 - \beta(t)) v_j^i(t) + \beta(t) \tilde{v}_j^i(t), r_3 > 0.5 \\ (1 - \beta(t)) v_j^i(t) + \beta(t) \tilde{v}_j^i(t), r_3 \leq 0.5 \end{cases} \quad (5.9)$$

$$\bar{v}_j^i(t) = 0.25 \left\{ r_4 \cdot (\rho_{\max_j} - \rho_{\min_j}) + \rho_{\min_j} \right\} \quad (5.10)$$

where $\bar{v}_j^i(t)$ is a random velocity of a particle element and its value is randomly generated and bounded within 0.25 of the range of the particle element value. The maximum velocity and minimum velocity are 0.25 of the range of particle element value. This value is chosen by trial and error through experiments. If this value is too large or too small, the searching performance might be degraded. The $r_3, r_4 \in [0 \ 1]$ is a random number. The resulting velocity of particle element $\bar{v}_j^i(t)$ combines the information $v_j^i(t)$ and $\tilde{v}_j^i(t)$. This information exchange process exhibits the characteristic of the crossover operation over g_j . However, $\bar{v}_j^i(t)$ is mutated individually and exhibits the characteristic of the mutation operation, called the cross-mutated (CM) operation. The control parameter $\beta(t)$ provides balance to control the resulting velocity $\bar{v}_j^i(t)$ converging toward $v_j^i(t)$ or $\tilde{v}_j^i(t)$. If $\beta(t)$ is at a value near to 0, $\bar{v}_j^i(t)$ will reach $v_j^i(t)$. On the other hand, when $\beta(t)$ is reaching 1, $\bar{v}_j^i(t)$ will reach $\tilde{v}_j^i(t)$. The random velocity $\tilde{v}_j^i(t)$ provides a means for the particle element to escape from a local optimum through a random movement. The $\beta(t)$ is to be governed by fuzzy rules as follows:

$$\text{Rule } j : \text{IF } \|\zeta(t)\| \text{ is } N_1^j, \text{ AND } t/T \text{ is } N_2^j \text{ THEN } \beta(t) = \chi_j, \quad j = 1, 2, \dots, \varepsilon \quad (5.11)$$

where χ_j is a singleton to be determined. The final value of $\beta(t)$ is as follows:

$$\beta(t) = \sum_{j=1}^{\varepsilon} m_j(t) \chi_j \quad (5.12)$$

where $m_j(t)$ is given in equation (5.7). The fuzzy rules for determining $\beta(t)$ are as follows:

Rule 1: IF $\ \zeta(t)\ $ is “L” AND t/T is “L”, THEN $\beta(t)$ is “VH”(=0.5)
Rule 2: IF $\ \zeta(t)\ $ is “M” AND t/T is “L”, THEN $\beta(t)$ is “H”(=0.4)
Rule 3: IF $\ \zeta(t)\ $ is “H” AND t/T is “L”, THEN $\beta(t)$ is “VH”(=0.5)
Rule 4: IF $\ \zeta(t)\ $ is “L” AND t/T is “M”, THEN $\beta(t)$ is “M”(=0.4)
Rule 5: IF $\ \zeta(t)\ $ is “M” AND t/T is “M”, THEN $\beta(t)$ is “M”(=0.3)
Rule 6: IF $\ \zeta(t)\ $ is “H” AND t/T is “M”, THEN $\beta(t)$ is “H”(=0.4)
Rule 7: IF $\ \zeta(t)\ $ is “L” AND t/T is “H”, THEN $\beta(t)$ is “VL”(=0.1)
Rule 8: IF $\ \zeta(t)\ $ is “M” AND t/T is “H”, THEN $\beta(t)$ is “VL”(=0.2)
Rule 9: IF $\ \zeta(t)\ $ is “H” AND t/T is “H”, THEN $\beta(t)$ is “L”(=0.2)

Figure 5.9: The Fuzzy rules for control parameter, $\beta(t)$

The value of $\beta(t)$ is determined by $\|\zeta(t)\|$ and t/T . Rules 1-3 with t/T being “L” correspond to the searching process in its early stage. Rules 7-9 correspond to the searching process in its late stage. The values of $\beta(t)$ in rules 1-3 are larger than those in rules 7-9. This is because a more significant random velocity provides more global exploration in the early stage. On the other hand, the effect of random velocity should be reduced in the later stage for more fine-tuning. In the early stage, when $\|\zeta(t)\|$ is “L”, meaning that the locations of particles are close to one another, the value of $\beta(t)$ has to be set to be larger than that when $\|\zeta(t)\|$ is “M”. It is because the chance that the solution is trapped in a local optimum is high. In the later stage, the searching process is doing fine-tuning. Thus, when the value $\|\zeta(t)\|$ is “L”, smallest value of $\beta(t)$ should be used. After the CM operation, an updated swarm is generated. The iterative process will be repeated when a defined number of iteration T is reached.

The objective of the FPSOCM-ANN training is to minimize the fitness (cost) values interactively with the fitness function:

$$fitness = 1 / (1 + err) \quad (5.13)$$

where *fitness* denotes the fitness value and *err* is the mean square error (MSE). It can be seen from the defining formula that a larger fitness value implies a smaller MSE.

5.4.2 Other Classifiers for Comparison Purposes

5.4.2.1 Linear Perceptron

Since it is for three classes of mental tasks classification, a multi-class linear perceptron classification was used in this study. Originally, linear perceptron is for binary classification. The Kesler's construction is used for the multi-class linear perceptron classification (Franc & Hlavac 2004; Schlesinger & Hlavac 2002). If the input is a set $T = \{(x_1, y_1), \dots, (x_l, y_l)\}$ of binary labelled $y_i \in \{1, \dots, c\}$ training vectors $x_i \in \mathbb{R}^n$. The problem of training the separating hyperplane can be defined in a form as follows:

$$\langle v \cdot z_i^y \rangle > 0, \quad i = 1, \dots, l, \quad y = Y \setminus \{y_i\} \quad (5.14)$$

where the vector $v \in \mathfrak{R}^{(n+1)c}$ is constructed as

$$v = [w_1; b_1; w_2; b_2; \dots; w_c; b_c] \quad (5.15)$$

and transformed training data $Z = \{z_1, \dots, z_l\}$ are defined as:

$$z_i = \begin{cases} [x_i; 1], & \text{for } j = y_i, \\ -[x_i; 1], & \text{for } j = y, \\ 0, & \text{otherwise} \end{cases} \quad (5.16)$$

The perceptron algorithm is an iterative procedure which builds a series of vectors $v^{(0)}, v^{(1)}, \dots, v^{(t)}$ until the set of inequalities of equation (5.14) is satisfied. The initial vector $v^{(0)}$ can be set arbitrarily. The Novikoff theorem ensures that the Perceptron algorithm stops after a finite number of iterations t if the training data are linearly separable.

5.4.2.2 Linear Discriminant Analysis (LDA)

The basic idea of linear discriminant analysis (LDA) (Delorme et al. 2010; Lu, Plataniotis & Venetsanopoulos 2003) is to find a linear transformation that best discriminates among classes, and the classification is then performed in the transformed space based on some metric, such as Euclidean distance. The same as the linear perceptron classification above, a multi-class LDA is needed here to classify three mental tasks. LDA was first introduced for two classes with its idea to transform the multivariate observation x_1, x_2, \dots, x_m where $x_i = (x_{i1}, \dots, x_{ip})$ belonging to two different classes (c_1 and c_2). In the two classes, the scatter matrices are as follows:

$$S_i = \sum_{x \in c_i} (x - \bar{x}_i)(x - \bar{x}_i)', \quad (5.17)$$

where $\bar{x}_i = \frac{1}{m_i} \sum_{x \in c_i} x$ and m_i is the number of samples in c_i . Thus, the total intra-class scatter matrix is given as follows:

$$\hat{\Sigma}_w = S_1 + S_2 = \sum_i \sum_{x \in c_i} (x - \bar{x}_i)(x - \bar{x}_i)' \quad (5.18)$$

The inter-class scatter matrix is given as follows:

$$\hat{\Sigma}_b = (\bar{x}_1 - \bar{x}_2)(\bar{x}_1 - \bar{x}_2)' \quad (5.19)$$

Fisher's criterion used the linear transformation Φ to maximize the Rayleigh coefficient as the ratio of the determinant of the inter-class scatter matrix of the projected samples to the intra-class scatter matrix of the projected samples as follows:

$$J(\Phi) = \frac{|\Phi^T \hat{\Sigma}_b \Phi|}{\Phi^T \hat{\Sigma}_w \Phi} \quad (5.20)$$

If $\hat{\Sigma}_w$ is non-singular, the equation (5.20) can be solved as a conventional eigenvalue problem and Φ is given by the eigenvector of matrix $\hat{\Sigma}_b / \hat{\Sigma}_w$.

In a multi-class LDA with a number of classes more than two, a natural extension of Fisher linear discriminant exists using multiple discriminant analysis. Suppose there are n classes, the intra-class matrix is calculated as follows:

$$\hat{\Sigma}_w = S_1 + \dots + S_n = \sum_{i=1}^n \sum_{x \in c_i} (x - \bar{x}_i)(x - \bar{x}_i)' \quad (5.21)$$

The inter-class scatter matrix is given by

$$\hat{\Sigma}_b = \sum_{i=1}^n m_i (\bar{x}_i - \bar{x})(\bar{x}_i - \bar{x})' \quad (5.22)$$

where m_i is the number of training samples for each class, \bar{x}_i is the mean for each class and \bar{x} is total mean vector given by $\bar{x} = \frac{1}{m} \sum_{i=1}^n m_i \bar{x}_i$. After obtaining $\hat{\Sigma}_b$ and $\hat{\Sigma}_w$, the linear transformation Φ should still maximize (5.20). It can be shown that the transformation Φ can be obtained by solving the generalised eigenvalue problem:

$$\hat{\Sigma}_b \Phi = \lambda \hat{\Sigma}_w \Phi \quad (5.23)$$

The upper bounds of the rank $\hat{\Sigma}_w$ and $\hat{\Sigma}_b$ are respectively $m-n$ and $n-1$.

5.4.2.3 Support Vector Machine (SVM)

The motivation behind the support vector machine (SVM) classification is to map the input into a high dimensional feature space, in which the data might be linearly separable. The training of the SVM is defined in the first place for the case of a binary classification problem, for which a linear decision surface exists that can perfectly classify the training data. The assumption of linear separability means that there exists some hyperplane, which perfectly separates the data. This hyperplane is a decision surface of the form:

$$w \cdot x + b = 0 \quad (5.24)$$

where w is an adjustable weight vector, x is an input vector, and b is a bias term. In order to generalise the SVM to non-linear decision, a kernel function can be used in the SVM.

The SVM is originally designed for binary classification. Since this study need to classify three mental tasks, a multi-class SVM will be required. Several multi-class SVM have been proposed such as: multi-class SVM using one-against-all (OAA) decomposition, multi-class SVM using one-against-one (OAO) decomposition, and multi-class BSVM formulation (Fatma Guler & Ubeyli 2007; Franc & Hlavac 2004). This study used a multi-class biased support vector machine (BSVM) for the comparison classification methods. The letter ‘‘B’’ in BSVM stands for the bias term added to the objective. The multi-class BSVM formulation is defined as:

$$\begin{aligned}
(\mathbf{W}^*, \mathbf{b}^*, \boldsymbol{\xi}^*) &= \underset{\mathbf{W}, \mathbf{b}}{\operatorname{argmin}} \underbrace{\frac{1}{2} \sum_{y \in Y} (\|w_y\|^2 + b_y^2)}_{F(\mathbf{W}, \mathbf{b}, \boldsymbol{\xi})} + C \sum_{i \in I} \sum_{y \in Y \setminus \{y_i\}} (\xi_i)^p \\
\text{subject to} & \\
\langle w_{y_i} \cdot x_i \rangle + b_{y_i} - (\langle w_y \cdot x_i \rangle + b_y) &\geq 1 - \xi_i^y, \quad i \in I, y \in Y \setminus \{y_i\}, \\
\xi_i^y &\geq 0, \quad i \in I, y \in Y \setminus \{y_i\}
\end{aligned} \tag{5.25}$$

where $\mathbf{W} = [w_1, \dots, w_c]$ is a matrix of normal vectors, $\mathbf{b} = [b_1, \dots, b_c]^T$ is a vector of biases, $\boldsymbol{\xi}$ is a vector of slack variables and $I = [1, \dots, l]$ is a set of indices. The $L1$ -soft margin is applied for $p=1$ and $L2$ -soft margin for $p=2$.

The multi-class rule is composed of the set of discriminate function $f_y : X \rightarrow \mathbb{R}$ which is computed as follow:

$$f_y(x) = \sum_{i \in I} \langle x_i \cdot x \rangle \sum_{u \in Y \setminus \{y_i\}} \alpha_i^u (\delta(u, y_i) - \delta(u, y)) + b_y, \quad y \in Y \tag{5.26}$$

where the bias b_y , $y \in Y$ is given by following:

$$b_y = \sum_{i \in I} \sum_{y \in Y \setminus \{y_i\}} \alpha_i^u (\delta(y, y_i) - \delta(y, u)), \quad y \in Y \tag{5.27}$$

The non-linear kernel classifier is obtained by substituting the selected kernel $k : X \times X \rightarrow \mathbb{R}$ for the dot product $\langle x \cdot x' \rangle$ to (5.26) and (5.27).

5.4.2.4 Genetic Algorithm of Artificial Neural Network (GA-ANN)

GA-ANN for BCI using mental task classification has been discussed in chapter 4. It is using the GA to optimize parameters during the training of the ANN. In GA optimization, three operations are needed: 1) selection operation, which is used to select

the chromosomes from the population with respect to some probability distribution based on some fitness values. This study used the normalized geometric ranking for the selection operation; 2) crossover operation which is used to combine the information of the selected parents' chromosomes and generate the offspring. A blend- α crossover operation was used in this study; 3) mutation operation, which is used to change the offspring variables. A non-uniform mutation was applied in this study.

5.5 Results

5.5.1 Testing of the HHT as the Features Extraction

The dominant feature on the alpha band of EEG (8-13Hz) can be found during the eyes closed task (Craig et al. 2002). This unique feature can be used for testing of the HHT feature extraction method to ensure the method is correctly converting the raw EEG signal into proper features. The HHT spectrum as shown in Fig. 5.10 for the eyes closed task shows a clear dominant feature of the instantaneous frequency of the alpha EEG band (8-13Hz). This shows the features extraction method has correctly converted the eyes closed task into the proper feature.

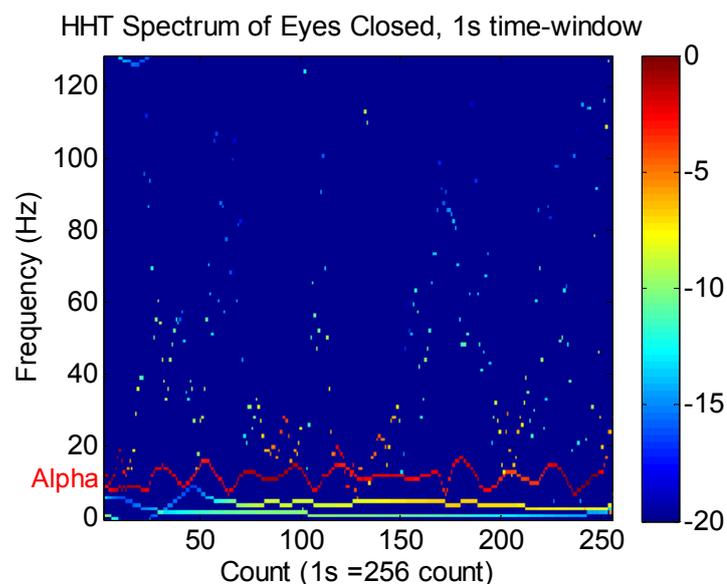


Figure 5.10: HHT spectrum for eyes closed 1s time-window

5.5.2 Classification using FPSOCM–ANN and Comparison with GA-ANN at Different Time-Windows

Table 5.3 shows the parameters of the GA and FPSOCM for ANN optimization. The parameters used in the FPSOCM-ANN training are as follows: the training of the neural network was repeated 10 times for each different hidden neuron; thus, the reported accuracies were the mean value of 10 results of accuracies.

The number of hidden neurons was varied from 4 to 30 units to obtain the best number with the lowest mean square error (MSE) and highest classification accuracy. The parameters for FPSOCM were as follows: The swarm size is 50, the number of iterations is 2000, both acceleration constants are 2.05, the maximum velocity is 0.2 and the probability of each CM is 0.005. A 3-fold cross-validation was used to evaluate the performance BCI classification accuracy.

Table 5.3: Parameters GA and FPSOCM for ANN training

GA Parameters	FPSOCM Parameters
Number of iteration = 2000	Number of iteration = 2000
Population size = 50	Swarm size = 50
Selection function = normalised geometric ranking; probability of selecting the best chromosome = 0.08	Acceleration constants = 2.05
Crossover function = Blend- α crossover; probability of crossover = 0.8	Maximum velocity = 0.2
Mutation function = non-uniform mutation; probability of mutation = 0.1	Probability of CM = 0.005

The GA-ANN from the previously applied mental task-based BCI method in chapter 4 was also used for initial comparison with FPSOCM-ANN. The population size used for the GA is 50, and the training is stopped when the training of the neural network reaches up to 2000 iterations. The GA parameters include the following

characteristics: real-code genetic algorithm, selection operator using normalised geometric ranking with the probability of selecting the best chromosome 0.08, crossover operator based on Blend- α crossover with probability of crossover 0.8 and non-uniform operation for the mutation operation with probability of mutation 0.1. The features from different time-windows were divided into training and testing sets. A 3-fold cross-validation was used to evaluate the performance BCI classification accuracy.

The accuracy of the FPSOCM-ANN classifications in Fig. 5.11 that shows the eyes closed and open experiments resulted in average accuracy above 90% for able-bodied participants and patients with tetraplegia. The high accuracy for eyes closed-open classification demonstrates that the HHT algorithm was able to generate distinct features for FPSOCM-ANN classification. The eyes closed task can also be used for the additional on-off command in the application of BCI based wheelchair control.

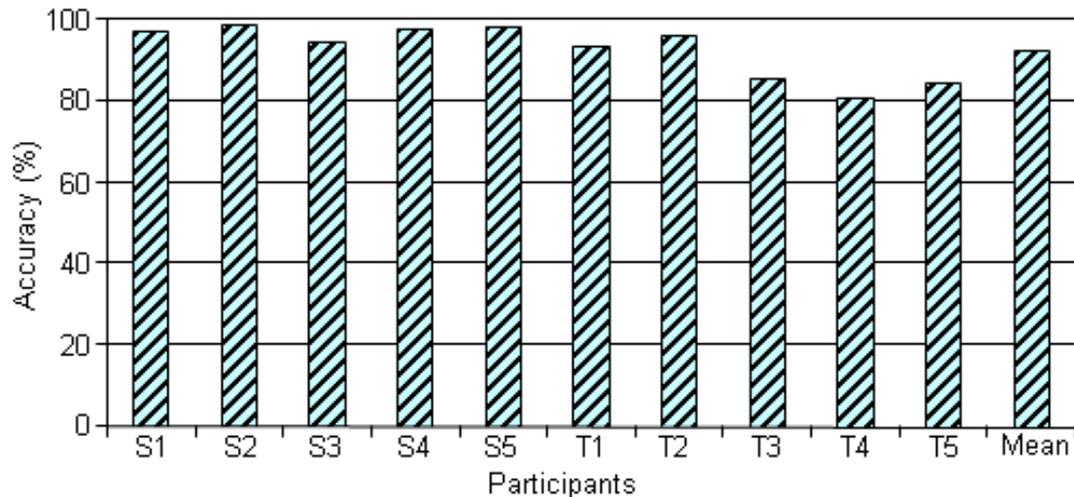


Figure 5.11: Accuracy of eyes closed-open action at 1s time-window

Table 5.4 shows the accuracies between GA-ANN and FPSOCM-ANN to classify three mental tasks (letter composing, arithmetic and figure of Rubik's cube rolling forward) in different time-windows (1s to 10s) with five able-bodied subjects (S1-S5) and five patients with tetraplegia (T1-T5). Between 1s to 6s time-windows, the average accuracies for able-bodied subjects were increased in each higher time-window between

67.3±5.7% and 76.5±7.9% using GA-ANN. These were improved using FPSOCM-ANN compared to GA-ANN with improved accuracies between 72.0±3.9% and 83.9±6.7%. Patients with tetraplegia resulted in accuracies between 59.7±1.5% and 74.4±4.2% using GA-ANN and improved accuracies based on FPSOCM-ANN with an average accuracy between 63.0±1.0% and 81.8±3.8%. The overall mean accuracy of three mental tasks classification at between 1s to 6s time-windows for both groups were between 63.5±5.6% and 75.5±6.1% with GA-ANN and improved accuracies between 67.5±5.5% and 82.9±5.2% were found with FPSOCM-ANN.

The highest overall accuracy was reached at 7s time-window with improved accuracies for both groups, compared to previous time-windows. In details, the average accuracy of the five able-bodied subjects was 79.0±8.6% using GA-ANN method and improved accuracy at 85.5±5.5% using FPSOCM-ANN. In the five patients with tetraplegia, the average accuracy resulted at 75.9±5.3% using GA-ANN and this accuracy was increased using FPSOCM-ANN with an average accuracy of 83.3±6.0%. The overall average accuracy of three mental tasks classification for the two groups combined at a 7s time-window was 77.4±6.9% using GA-ANN and improved accuracy at 84.4±5.5% using FPSOCM-ANN.

Compared to the previous time-window, the overall accuracies between 8s and 10s time windows were much more in a steady value. The average accuracies for able-bodied subjects were between 77.2±7.6% and 78.0±6.9% with GA-ANN. These were improved using FPSOCM-ANN compared to GA-ANN with accuracies of around 84.6±5.3%. For patients with tetraplegia, accuracies using GA-ANN were between 75.4±4.8 % and 76.9±6.4 and improved accuracies based on FPSOCM-ANN with an average accuracy of around 83.4±5.4%. The overall accuracies of three mental tasks classification is between 8s to 10s time-windows for both groups were between 76.5±6.7% and 77.1±6.6% with GA-ANN and improved accuracies of around 84.0±5.1% using FPSOCM-ANN.

Table 5.4: Comparison accuracy between GA-ANN and FPSOCM-ANN of 6 channels for 3 mental task classifications in different time-windows with 5 able-bodied subjects and 5 patients with tetraplegia

Subjects S1-S5=able-bodied; T1-T5=Tetraplegia	Classification Algorithms	Mean of Accuracies in different time-windows (1s-10s) 3-fold cross-validation, each fold is repeated 10 times, and the mean value is used									
		1s (%±std)	2s (%±std)	3s (%±std)	4s (%±std)	5s (%±std)	6s (%±std)	7s (%±std)	8s (%±std)	9s (%±std)	10s (%±std)
S1	GA-ANN	71.0±1.3	71.0±1.7	74.0±2.5	74.1±2.7	76.2±3.7	80.7±2.9	82.9±1.5	81.2±2.6	80.6±3.6	82.3±2.8
	FPSOCM-ANN	72.9±2.1	73.4±1.5	78.7±2.6	80.3±3.1	83.9±2.0	86.5±2.4	86.3±2.1	83.1±2.3	85.5±3.4	85.7±1.8
S2	GA-ANN	61.6±3.7	63.1±5.2	68.4±3.0	74.9±3.7	74.6±1.7	74.6±3.8	77.8±1.9	75.0±2.2	76.1±2.6	77.5±3.5
	FPSOCM-ANN	68.2±2.5	69.9±4.8	71.8±3.5	77.5±2.5	77.5±2.6	79.9±3.3	85.3±1.2	84.4±1.5	85.0±1.1	86.1±1.4
S3	GA-ANN	70.7±1.9	74.0±2.5	80.3±3.6	79.5±1.4	81.4±2.6	81.2±2.6	85.6±2.3	81.7±3.4	80.7±1.2	79.6±3.3
	FPSOCM-ANN	76.0±1.5	83.3±1.5	87.2±2.1	89.8±1.7	89.4±1.0	90.7±2.4	90.1±1.7	89.3±1.5	88.6±8.2	88.2±2.5
S4	GA-ANN	72.5±3.9	77.7±2.3	80.8±1.7	79.2±1.5	81.3±1.4	82.8±4.5	84.1±1.9	83.2±3.6	83.0±1.5	83.4±1.6
	FPSOCM-ANN	75.5±3.5	81.8±1.9	82.7±1.9	83.6±1.7	86.5±1.6	88.2±2.5	89.5±0.5	88.1±1.2	88.3±0.5	87.6±1.1
S5	GA-ANN	60.7±3.8	60.9±4.9	61.7±4.8	62.3±3.2	63.1±3.6	63.5±3.3	64.6±3.7	64.8±4.4	65.1±3.1	66.1±3.7
	FPSOCM-ANN	67.6±2.8	69.7±4.6	70.9±4.9	72.7±2.9	73.5±3.1	74.4±3.8	76.4±1.1	78.0±1.5	76.9±1.8	75.3±1.9
Mean S1-S5	GA-ANN	67.3±5.7	69.3±7.1	73.0±8.1	74.0±7.0	75.3±7.5	76.5±7.9	79.0±8.6	77.2±7.6	77.1±7.1	78.0±6.9
	FPSOCM-ANN	72.0±3.9	75.6±6.5	78.3±7.0	80.8±6.4	82.2±6.6	83.9±6.7	85.5±5.5	84.6±4.5	84.9±4.7	84.6±5.3
T1	GA-ANN	59.0±6.8	67.4±3.4	67.5±5.1	70.1±2.8	73.7±3.3	74.2±7.4	78.6±3.0	78.9±2.5	81.8±1.7	77.8±4.4
	FPSOCM-ANN	62.6±4.1	66.7±5.0	67.9±7.1	72.0±4.8	76.6±4.1	80.4±6.0	85.2±2.5	83.5±1.8	83.5±1.2	84.2±2.4
T2	GA-ANN	61.6±3.7	68.0±1.7	72.1±1.6	74.2±1.5	78.3±2.3	80.0±2.4	83.3±1.4	83.9±8.1	82.7±1.4	81.1±3.9
	FPSOCM-ANN	61.5±4.3	68.5±4.0	73.9±2.4	76.2±2.8	83.3±1.7	83.6±2.8	84.4±1.9	84.7±4.2	84.4±1.6	84.5±1.5
T3	GA-ANN	58.9±5.2	64.1±3.5	64.4±5.2	69.0±4.9	70.1±2.3	72.6±4.0	72.5±3.5	75.8±3.0	73.8±3.9	74.6±3.8
	FPSOCM-ANN	63.4±5.6	69.9±3.6	77.0±3.6	76.8±2.6	77.7±1.1	80.1±2.3	81.0±1.2	83.0±3.6	80.1±2.7	84.8±1.5
T4	GA-ANN	60.1±3.8	63.0±3.5	68.2±3.8	70.8±3.6	69.9±2.7	68.8±2.8	69.8±3.4	66.8±5.4	65.3±5.0	68.1±2.7
	FPSOCM-ANN	63.9±3.9	68.1±4.5	73.4±4.2	75.5±4.1	77.2±3.8	77.5±2.2	74.8±1.7	77.0±2.4	80.4±2.3	74.3±4.8
T5	GA-ANN	59.0±4.8	66.4±1.4	72.6±2.8	73.0±3.4	75.0±3.2	76.2±3.7	75.2±3.7	79.3±1.2	76.3±3.5	75.2±1.6
	FPSOCM-ANN	63.9±3.2	69.5±1.7	79.9±2.9	85.3±0.5	85.9±1.0	87.4±1.4	91.0±0.5	88.7±1.1	88.9±1.2	89.0±1.0
Mean T1-T5	GA-ANN	59.7±1.2	65.8±2.1	69.0±3.4	71.4±2.2	73.4±3.5	74.4±4.2	75.9±5.3	76.9±6.4	76.0±7.0	75.4±4.8
	FPSOCM-ANN	63.0±1.0	68.6±1.3	74.4±4.5	77.1±4.9	80.2±4.2	81.8±3.8	83.3±6.0	83.4±4.2	83.5±3.6	83.4±5.4
Overall mean	GA-ANN	63.5±5.6	67.5±5.3	71.0±6.2	72.7±5.1	74.4±5.6	75.5±6.1	77.4±6.9	77.1±6.6	76.5±6.7	76.6±5.7
	FPSOCM-ANN	67.5±5.5	72.1±5.8	76.3±5.9	79.0±5.7	81.2±5.3	82.9±5.2	84.4±5.5	84.0±4.2	84.1±4.0	84.0±5.1

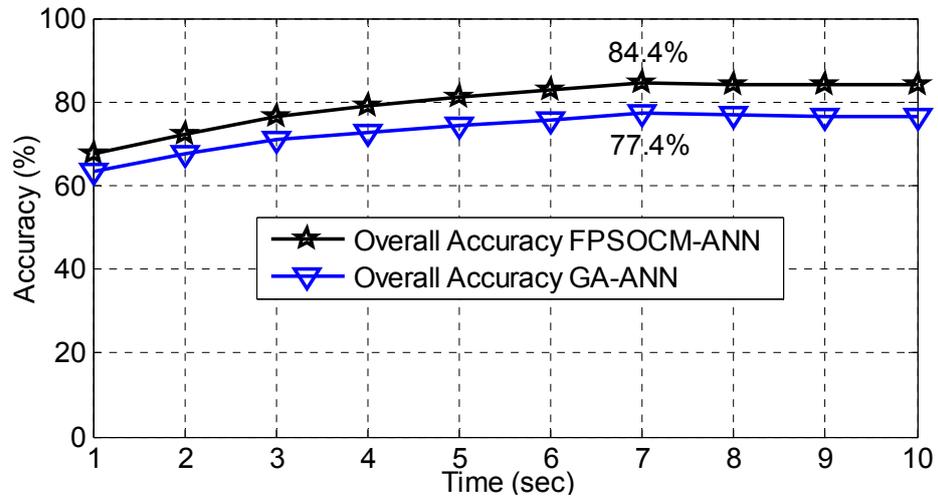


Figure 5.12: Plot of overall accuracy FPSOCM-ANN vs. GA-ANN

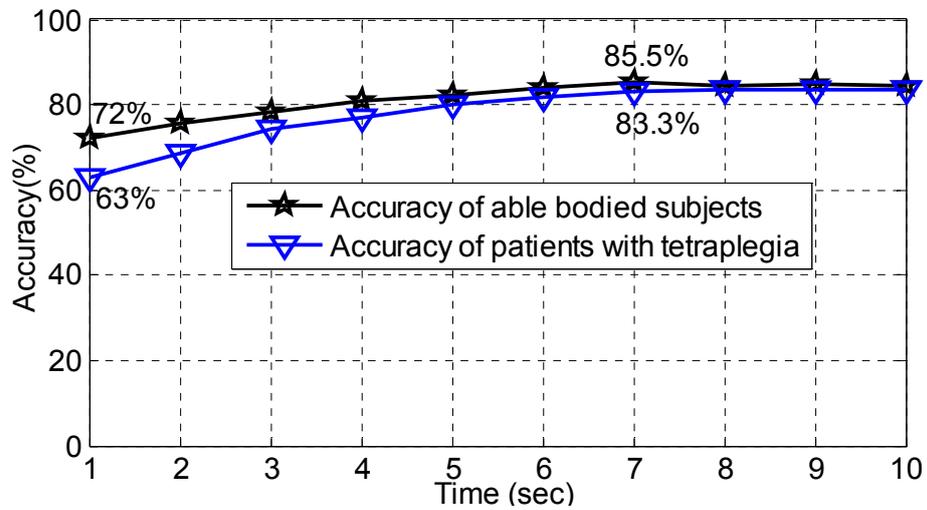


Figure 5.13: Plot of accuracy for able bodied subjects vs. patients with tetraplegia, time-windows at 1-10s; and classification based on FPSOCM-ANN.

In general, as shown in Fig 5.12 and Fig 5.13, for three mental task classifications, the results show an overall improved accuracy for the classification algorithm using FPSOCM-ANN compared to GA-ANN across both groups and in different time-windows. For comparison between able-bodied subjects and patients with tetraplegia, the patients' group provided lower classification accuracy. However, the accuracy was improved by increasing the duration of the time-window with at best achieved accuracy for a 7s time-window of 77.4% based on GA-ANN and improved accuracy of 84.4% using FPSOCM-ANN. From this experiment, patients with tetraplegia tend to have better classification accuracy when increasing the time-window, which means they need more time to perform the mental tasks due to their disability. This can be found using FPSOCM-ANN, where for a 1s time-window, patients with tetraplegia achieve an accuracy of 63% compared with 72% for able-bodied subjects. The patients group had accuracy around 9% lower than the able-bodied group. At 7s time-window, the accuracy difference between both groups was much smaller, at only 2.2% with an accuracy for patients with tetraplegia of 83.3% and able-bodied subject of 85.5%.

5.5.3 Further Comparison Classifiers and Features Extractors

Further comparisons were carried out between different classifiers and feature extractors, as shown in Table 5.5 and Fig. 5.15 using the 7s time-window of data as the best time window. Because to the output classifier provides three mental tasks, instead of using a binary classifier, the comparison classifiers needs a multi-class model. These classifiers are: multi-class linear perceptron classifier with the Kesler's construction, multi-class linear discriminant analysis (LDA) with the fisher kernel and multi-class support vector machine (SVM) using BSVM. For the feature extraction methods comparison, FFT and wavelet methods are presented. There are improvements on the original HHT especially with the EMD analysis available. The HHT using the ensemble empirical mode decomposition (EEMD) is also included in this study.

The result shows the multi-class linear perceptron classifier provided an overall mean accuracy of: $63.7 \pm 2.5\%$ using the FFT, $67.1 \pm 2.8\%$ using the wavelet, $71 \pm 4.5\%$

using the HHT-EMD and 71.7 ± 3.8 % using the HHT-EEMD. The accuracies are slightly improved across different feature extractors when using the multi-class LDA classifier with the overall mean accuracy of: 64.8 ± 3.3 % using the FFT, 68.1 ± 2.7 % using the wavelet, 72.1 ± 4.5 % using the HHT-EMD and 72.5 ± 4.4 % using the HHT-EEMD. The multi-class SVM classifier provided improved results across different features extractors to the previous two methods with overall means accuracy of: 66.9 ± 2.7 % using FFT, 70.6 ± 3.2 % using wavelet, 77.0 ± 6.6 % using HHT-EMD, 63.7 ± 2.5 % and 76.8 ± 5.6 % using HHT-EEMD. For the GA-ANN classifier, the accuracy is further improved compare to previous classifiers with the overall accuracy of: 67.8 ± 3.4 % using FFT, 71.8 ± 3.3 % using wavelet, 77.4 ± 6.9 % using HHT-EMD and 77.9 ± 6.7 % using the HHT-EEMD. The FPSOCM-ANN classifier improved at the most with the overall mean accuracy of: 70.7 ± 2.9 % using FFT, 75.7 ± 4.1 % using wavelet, 84.4 ± 5.5 % using HHT-EMD and 84.5 ± 4.9 % using HHT-EEMD.

The neural networks classifiers (GA-ANN and FPSOCM-ANN) with the optimization methods in this study provided better results compared to the linear perceptron, LDA and SVM methods. The FPSOCM-ANN provided the best classifier among other classifiers. The HHT as the feature extractor has resulted in a better method of accuracy compared to FFT and wavelet methods. In the FPSOCM-ANN, it can also be seen that there is a comparable accuracy between HHT-EMD and HHT-EEMD in this study, thus both methods can be used. A statistical significance test using p-value is presented in Table 5.6 to evaluate the significant difference between the two algorithms. The results show the FPSOCM-ANN classifier compared to the GA-ANN, SVM, LDA and linear perceptron with all of the p values less than 0.05. A p-value less than 0.05 is considered statistically significant with 95% confidence level.

Table 5.5: Comparison between different classifiers and feature extractors using 7s time-window of data

3-fold cross-validation, each fold is repeated 10 times, and the mean value is used						
Subjects S1-S5=Able-bodied T1- T5=Tetraplegia	Feature Extractors	Linear Perceptron (%±std)	LDA (%±std)	SVM (%±std)	GA- ANN (%±std)	FPSOC M- ANN (%±std)
S1	FFT	66.2±1.4	64.0±2.8	67.2±3.8	68.4±3.4	70.8±4.3
	Wavelet	70.3±1.8	69.8±1.9	70.9±2.1	70.7±2.3	78.9±2.6
	HHT-EMD	77.5±1.3	78.2±1.8	83.1±0.2	82.9±1.5	86.3±2.1
	HHT-EEMD	73.0±4.2	78.0±1.2	78.2±1.4	82.5±1.2	86.4±1.2
S2	FFT	69.0±2.7	67.0±3.9	72.2±1.9	73.3±1.9	75.4±3.3
	Wavelet	68.6±1.2	69.0±2.3	70.6±1.3	73.3±2.2	79.7±2.0
	HHT-EMD	70.3±2.8	73.8±3.6	77.6±0.7	77.8±1.9	85.3±1.2
	HHT-EEMD	72.9±2.7	74.5±2.5	79.3±2.5	80.1±3.4	86.2±2.7
S3	FFT	64.0±3.5	72.9±1.9	68.6±4.5	70.3±3.0	73.1±2.8
	Wavelet	71.3±2.0	73.3±1.3	74.3±2.1	73.9±4.1	79.4±4.1
	HHT-EMD	75.7±2.1	78.1±2.7	83.4±1.4	85.6±2.3	90.1±1.7
	HHT-EEMD	75.1±1.6	77.5±2.3	84.7±0.5	85.7±1.1	89.1±2.3
S4	FFT	65.5±1.9	65.8±3.1	70.3±3.6	73.0±3.6	75.6±3.2
	Wavelet	66.5±2.2	67.6±2.0	75.7±2.5	78.2±2.3	81.0±4.6
	HHT-EMD	75.9±1.9	70.2±4.8	85.4±1.3	84.1±1.9	89.5±0.5
	HHT-EEMD	75.5±0.8	73.7±3.6	84.4±1.8	86.5±0.4	87.8±2.7
S5	FFT	62.2±1.8	62.5±2.4	64.5±2.2	65.9±4.7	67.6±4.0
	Wavelet	65.0±0.9	64.2±3.7	66.8±3.1	69.0±1.7	70.3±4.8
	HHT-EMD	64.1±2.2	64.0±2.8	64.5±2.1	64.6±3.7	76.4±1.1
	HHT-EEMD	65.1±2.4	65.0±2.5	65.7±1.4	66.0±2.9	77.5±2.7
T1	FFT	63.2±3.9	63.6±2.5	64.9±2.8	65.4±4.1	69.0±1.3
	Wavelet	66.0±4.2	69.2±1.1	69.3±1.4	70.8±2.3	71.6±1.2
	HHT-EMD	69.4±2.5	72.1±3.2	77.3±1.3	78.6±3.0	85.2±2.5
	HHT-EEMD	70.6±2.4	72.5±3.6	76.3±0.4	74.1±5.1	84.9±3.1
T2	FFT	62.3±2.2	62.9±3.8	63.9±3.2	64.0±5.1	69.4±2.4
	Wavelet	65.1±3.4	64.4±4.2	68.1±2.9	68.9±4.6	74.9±2.5
	HHT-EMD	71.6±2.7	72.5±4.9	80.9±2.7	83.3±1.4	84.4±1.9
	HHT-EEMD	73.3±1.2	73.6±2.5	75.0±4.2	81.9±3.4	82.5±3.0
T3	FFT	61.7±3.9	63.0±2.7	66.5±0.6	65.4±4.9	68.9±3.2
	Wavelet	63.3±2.5	67.5±1.5	70.8±1.3	71.6±3.6	72.9±2.1
	HHT-EMD	67.4±4.1	72.4±2.6	73.9±2.0	72.5±3.5	81.0±1.2
	HHT-EEMD	68.8±3.6	73.3±2.4	76.5±1.5	75.7±2.3	84.0±1.4
T4	FFT	62.1±4.0	64.9±3.8	64.5±2.7	65.0±2.2	68.4±4.1
	Wavelet	64.8±2.4	66.4±1.7	65.8±1.8	67.1±4.2	70.6±2.5
	HHT-EMD	65.4±3.2	66.1±2.3	69.5±1.6	69.8±3.4	74.8±1.7
	HHT-EEMD	66.4±1.3	65.3±3.0	71.6±0.7	70.0±2.1	75.8±2.3
T5	FFT	60.6±3.7	61.4±2.3	66.5±2.5	67.1±2.6	68.9±2.7
	Wavelet	70.3±2.2	69.6±1.8	73.3±1.3	74.8±2.3	77.3±1.8
	HHT-EMD	72.4±3.1	73.5±2.5	74.2±1.4	75.2±3.7	91.0±0.5
	HHT-EEMD	75.9±1.0	71.1±3.5	76.0±0.7	76.6±2.0	91.2±1.6
Overall Mean	FFT	63.7±2.5	64.8±3.3	66.9±2.7	67.8±3.4	70.7±2.9
	Wavelet	67.1±2.8	68.1±2.7	70.6±3.2	71.8±3.3	75.7±4.1
	HHT-EMD	71.0±4.5	72.1±4.5	77.0±6.6	77.4±6.9	84.4±5.5
	HHT-EEMD	71.7±3.8	72.5±4.4	76.8±5.6	77.9±6.7	84.5±4.9

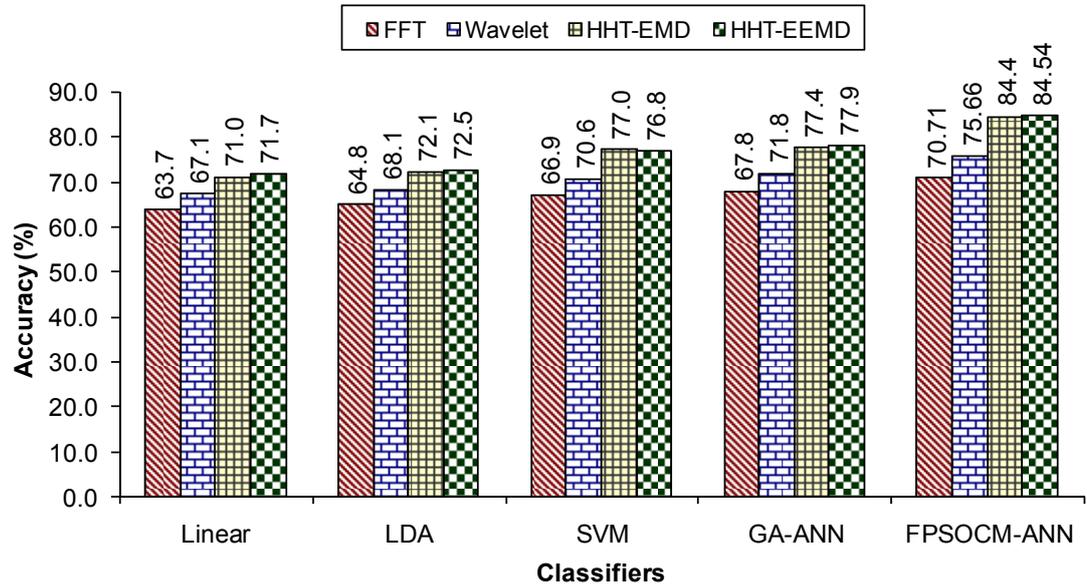


Figure 5.14: Plot of overall comparison between different classifiers and features extractors at 7s time-window of data.

Table 5.6: Statistical significant using overall means classification accuracies at 7s of time-window

Classifiers comparison	FFT (<i>p</i> -value)	Wave (<i>p</i> -value)	HHT-EMD (<i>p</i> -value)	HHT-EEMD (<i>p</i> -value)
FPSOCM-ANN vs. GA-ANN	0.034	0.023	0.017	0.016
FPSOCM-ANN vs. SVM	0.008	0.006	0.012	0.005
FPSOCM-ANN vs. LDA	0.0011	0.0004	0.0002	0.0001
FPSOCM-ANN vs. Linear perceptron	0.00014	0.0002	0.00011	0.00005

5.6 Discussion

5.6.1 Comparison between the Globally Trained Classifier and the Subject's Specifically Trained Classifier

The large differences in EEG-based BCI, known as inter-subject variability, could affect different performances (Blankertz et al. 2006; Grosse-Wentrup & Schölkopf 2013). This study also presents the comparison between the globally trained classifier and subject's specifically trained classifier, which can be seen in Fig. 5.15 using the 7s as the best time-window with the total dataset; each mental task is 210 sets per participants. A 3-fold cross validation was used to evaluate the performance of the BCI classification accuracy. For the globally trained classifier, the datasets from 6 participants were used for classifier training including portions of data from 3 able-bodied (S1, S2 and S3) and portions of data from 3 patients with tetraplegia (T1, T2, and T3) at the best time-window. The same 6 participants (S1, S2, S3, T1, T2, and T3) provide portions of data for the testing dataset. Also, the different participants (S4, S5, T4 and T5) were then treated as new users with their full portions of data used for the testing set only. For the subject's specifically trained classifier, there were 10 participants, 5 able-bodied subjects (S1, S2, S3, S4, and S5) and 5 patients with tetraplegia (T1, T2, T3, T4, and T5) who provided their portions of data for training and testing at the best time-window.

The result shows comparable accuracies only for able-bodied subjects and patients with tetraplegia (S1, S2, S3, T1, T2 and T3) who have been provided with their portions of dataset for the training of the classifier. For the S4, S5, T4, and T5, who were treated as new users of the system, a large drop of accuracies resulted. The drop in these accuracies was large for the globally trained classifier: S4 at 75.11%, S5 at 67.17%, T4 at 63.08% and T5 at 78.5%. If this is compared to the user's specifically trained classifier: S4 at 89.5%, S5 at 76.4%, T4 at 91 % and T5 at 84.4%. As a result, the overall accuracy for the globally trained classifier has also dropped with mean accuracy at 78.5%, compared to each subject's specifically trained classifier with means of

accuracy at 84.4%. This shows the inter-subject variability issue for the EEG-based BCI system. As a result, the subject's specifically trained classifier is preferable. The other option is the globally trained classifier with adaptive re-training to ensure the classifier is able to generalize the data from the new subject which was previously unseen by it.

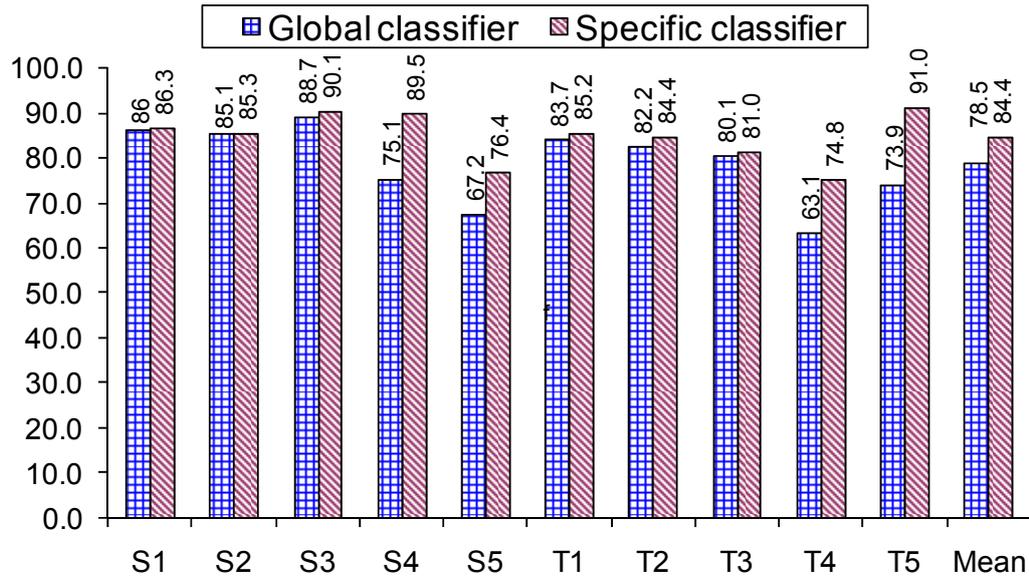


Figure 5.15: Comparison accuracy between globally trained classifier and subject's specifically trained classifier

5.6.2 Computational time of Classifiers

In terms of computation time of classifiers, Table 5.7 shows the training time of globally and subject's specifically trained classifiers, which are presented with the same value classification (execution) time. This computational time is estimated by MATLAB's built-in tic/toc functions whereas the tic function was called before the program and the toc afterwards on the computer (Intel Core i5-2400 processor 3.10GHz, 4GB RAM). The results show the training times of linear perceptron, LDA, and SVM classifiers were faster than ANN based classifiers being less than 10 sec for the subject's specifically trained classifier and less than 2 min for the globally trained classifier. For the ANN classifiers, the training time for FPSOCM-ANN was faster

compared to the GA-ANN at around 3 min for the subject's specifically trained classifier and at around 15 min for the globally trained classifier. The GA-ANN required the longest duration of time for training, compared to other classifiers at 11 minutes, for the subject's specifically trained classifier and 45 min for the globally trained classifier. In terms of the classification/execution time, all classifiers were able to complete the task in less than a second. The program developed in MATLAB takes more computational time compared to C language with significantly faster training time.

Table 5.7: Comparison of training times and classification times

Classifiers	Training time (s)		Classification (execution) time (s)
	Subject's Specifically	Globally	
Linear Perceptron	3	60	0.008
LDA	5	72	0.03
SVM	6	80	0.04
GA-ANN	660	2700	0.092
FPSOCM-ANN	175	896	0.091

Note that for the classification, the reason that ANN-based classifiers (GA-ANN and FPSOCM-ANN) were able to perform as fast as other classifiers was because, after the ANN training, final weights were treated as constants in the ANN feed-forward classification routine. This was the same for the real-time BCI environment. There was no need to perform the training classifier again during the real-time classification. Furthermore, the time-window was needed for the BCI real-time operation, as discussed in the previous section with the best at 7s. As a result, there is sufficient time available, even using a 1s time-window for FPSOCM-ANN as the best classifier with highest accuracy, to perform continuous real-time classification until the next time-window elapses.

5.6.3 Two Channels Classification for Practical Application

For more practical application purposes, the number of EEG channels was reduced from six channels into two channels. From the asymmetry of the channels location point of view, the locations can be divided into left and right groups: *C3*, *P3* and *O1* for the left channels group; *C4*, *P4*, *O2* for the right channels group.

As a result, there are nine two-channel combinations found from the two groups which are: *C3&C4*, *C3&P4*, *C3&O2*, *P3&C4*, *P3&P4*, *P3&O2*, *O1&C4*, *O1&P4*, and *O1&O2*. Table 5.8 shows the accuracy for nine combinations of two EEG channels of the three mental task classifications at 7s-time window using FPSOCM-ANN. The results show that there was a variety of chosen channels for each participant that provided the best accuracy. The first able-bodied subject (S1) has best means of accuracy of 81.6% using the *O1&O2* channels. The fifth able-bodied subject (S5) has best means of accuracy of 75.7% using the *O1&P4* channels. The second patient with tetraplegia (T2) has best means of accuracy of 82.4% using the *P3&C4* channels. Five participants have the best accuracy using the *O1&C4* channels including the second able-bodied subject (S2) with accuracy of 76.2%, third able-bodied subject (S3) with accuracy of 87.1%, the first patient with tetraplegia (T1) with accuracy of 83.6%, the fourth patient with tetraplegia (T4) with an accuracy of 76.5% and the fifth patient with tetraplegia (T5) with accuracy of 88.1%. Two participants have the best accuracy using the *P3-O2* channels, including the fourth able-bodied subject (S4) with an accuracy of 85.7%, and the second patient with tetraplegia and the third patient with tetraplegia (T3) with an accuracy of 79.4%.

For the overall mean of accuracy as a generalisation, *O1&C4* are the best two channels with accuracy at $80.5\pm 4.8\%$. This is followed by the second option for two-channel EEG at *P3&O2*, with an overall mean of accuracy of $76.4\pm 5.6\%$. The third option of two channels EEG was *C3&O2* with an accuracy of $75.4\pm 5\%$. For comparison overall accuracy of both groups between six channels (*C3*, *C4*, *P3*, *P4*, *O1* and *O2*) and the best two channels (*O1&C4*) at the best 7s time-window, the accuracy of six

channels was $84.4 \pm 5.5\%$ and compared to two channels (O1&C4) with almost comparable accuracy at $80.5 \pm 4.8\%$. The two channels option with less EEG electrodes is preferable for a practical system as it still provides accuracy above 80%.

Table 5.8: Accuracies of two channels combinations of 3 mental tasks classification in 7s time-window with 5 able-bodied subjects (S1-S5) and 5 patients with tetraplegia (T1-T5) using FPSOCM-ANN

Subjects	Accuracy(%) of combination two channels at 7s time-window								
	C3&C4	C3&P4	C3&O2	P3&C4	P3&P4	P3&O2	O1&C4	O1&P4	O1&O2
S1	68.9	61.8	77.1	61.0	60.7	76.8	78.3	69.9	<u>81.6</u>
S2	71.4	66.8	72.9	73.3	68.1	74.0	<u>76.2</u>	70.8	70.5
S3	67.6	81.1	81.6	66.3	73.5	84.8	<u>87.1</u>	74.8	71.9
S4	80.0	82.9	81.0	83.3	81.3	<u>85.7</u>	82.7	81.0	80.5
S5	74.0	74.1	73.7	63.3	68.4	70.3	74.6	<u>75.7</u>	74.8
T1	76.0	61.4	75.7	79.8	71.1	77.6	<u>83.6</u>	67.5	76.2
T2	80.3	74.3	75.2	<u>82.4</u>	73.0	72.7	80.8	68.9	73.3
T3	67.9	55.2	77.2	70.4	57.9	<u>79.4</u>	76.9	67.6	74.0
T4	70.0	67.8	63.7	75.4	70.5	70.0	<u>76.5</u>	75.1	67.3
T5	78.6	79.5	76.0	73.6	80.3	72.5	<u>88.1</u>	75.2	72.9
Average	73.5	70.5	<u>75.4</u>	72.9	70.5	<u>76.4</u>	<u>80.5</u>	72.6	74.3
Std	5.6	9.3	5.0	7.7	7.4	5.6	4.8	4.4	4.3

Table 5.9 shows the accuracy with correctly classified rates for each mental tasks as the three-class classification using FPSOCM-ANN at 7s time windows of six EEG channels and the best two EEG channels (O1&C4). Overall, mental arithmetic has the best classification accuracy compared to other tasks with an average of the correctly classified rate of $89.3 \pm 5.5\%$ for six EEG channels and $85.2 \pm 5.4\%$ for the best two EEG channels. This is followed by mental Rubik's cube rolling forward with an average correctly classified rate of $84.6 \pm 6.8\%$ for six EEG channels and $78.5 \pm 9.1\%$ for the best

two EEG channels. The mental letter composing has a lower correctly classified rate compared to the previous two tasks of $79.3 \pm 8.7\%$ for six EEG channels and $77.8 \pm 5.5\%$ for two EEG channels.

Table 5.9: Accuracies and correctly classified rates of each mental task in 7s time-window with 5 able-bodied subjects (S1-S5) and 5 patients with tetraplegia (T1-T5) using FPSOCM-ANN for 6 channels (C3-C4-P3-P4-O1-O2) and the best 2 channels (O1-O4) EEG

Subjects	Acc = % accuracy for 3 mental tasks; R= % correctly classified for arithmetic task; L= % correctly classified for letter composing task; F= % correctly classified for Rubik's cube rolling forward task							
	6 channels:O1O2C3C4P3P4				2 channels:O1C4			
	Acc	R	L	F	Acc	R	L	F
S1	86.3	90.8	87.2	81.0	78.3	76.7	84.6	73.6
S2	85.3	90.0	85.0	81.0	76.2	83.3	74.5	70.7
S3	90.1	91.2	86.5	92.7	87.1	91.5	79.2	90.5
S4	89.5	91.1	90.5	87.0	82.7	89.8	80.0	78.3
S5	76.4	78.9	68.1	82.1	74.6	78.3	71.4	74.1
T1	85.2	93.1	70.2	92.4	83.6	89.3	78.5	83.0
T2	84.4	95.7	71.4	86.2	80.8	89.9	68.2	84.3
T3	81.0	88.9	78.2	75.8	76.9	82.9	84.3	63.7
T4	74.8	80.3	69.8	74.3	76.5	80.8	74.8	73.9
T5	91.0	93.1	86.5	93.2	88.1	89.0	82.7	92.6
Average	84.4	89.3	79.3	84.6	80.5	85.2	77.8	78.5
Std	5.5	5.5	8.7	6.8	4.7	5.4	5.5	9.1

The classifier training is done automatically by using the FPSOCM-ANN classifier as a machine learning procedure. The weights for the best accuracy are saved as a constant to provide a feed-forward ANN classification. For the classification, it is a straight forward calculation based on equation (7) and does not require training at all.

Therefore, the classification function is the final detection of the mental task and the calculation takes usually less than a second.

5.7 Conclusion

The improved classifications of three non-motor imagery mental tasks have been applied in this chapter with ten participants (five able-bodied subjects and five patients with tetraplegia). Note that this study has included individuals with severe disabilities (patients with tetraplegia), who are an important target group for BCI technology for wheelchair application.

The HHT was used for the features extraction method and the FPSOCM-ANN for the classification algorithm. The HHT was compared to other features extractors (FFT and wavelet) and the FPSOC-ANN was compared with different classifiers (linear perceptron, LDA and SVM). The number of EEG channels used at first was six channels (C3, C4, P3, P4, O1 and O2). Initial testing of the feature extractor showed that the HHT composed correctly for the EMD process with the known combinations of sinusoid signal. Further HHT testing for the eyes closed EEG signal showed a correct dominant alpha (8-13Hz) wave with high classification accuracy for both groups of participants.

The HHT as the feature extractor has resulted in a better method of accuracy compared to FFT and wavelet methods. Results for three mental tasks classification showed improved accuracy of FPSOCM-ANN compared to GA-ANN and other classifiers in multi-class mode (SVM, LDA and linear perceptron) across both groups and different time-windows. Although the patients' group had lower classification accuracy, this was improved by increasing the time windows with the best classification accuracy achieved at a 7s time-window.

The result also shows comparable accuracies only for able-bodied subjects and patients with tetraplegia, who have been provided with their portions of dataset for the training of the classifier, while for participants who are treated as new users of the system, a large drop of accuracies are found. This shows the inter-subject variability

issue for the EEG-based BCI system. As a result, the subject's specifically trained classifier is preferable. The other option is the globally trained classifier with adaptive re-training to ensure the classifier is capable of generalising the data from the new subject, which was previously unseen by it.

For practical use of a BCI, combinations of two channels EEG have also been presented with the variation result of the best two channels in each subject. For overall generalisation of both groups, O1 and C4 are the best two channels, followed by the second best at P3 and O2, and the third best at C3 and O2 channels. For each correctly classified mental task, the arithmetic task has the highest rate of classification, followed by the rolling cube task and letter composing task.

These three non-motor imagery mental tasks can be directly mapped into three wheelchair commands. For example, the mental letter composing task is used for left command, arithmetic mental task for right command and imagining a rolling cube for forward command. An additional eyes closed task can be used for on-off command. The users perform the imagery mental tasks at will which are initiated by them (self-paced) for the wheelchair control application.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

BCI research is a cutting-edge technology and is in an interdisciplinary area of research, involving engineering, neuroscience, psychology, computer science, clinical rehabilitation and many others. The BCI study has been explored in this thesis, with the main aim of the research being to develop a BCI system to assist mobility as hand-free technology for people with severe disability, with improved accuracy, which provides effective classification to be used in the application of wheelchair control. This is because there is still a need for a variety of technologies for disabled individuals who suffer from locked-in syndromes, such as amyotrophic lateral sclerosis (ALS), cervical spinal cord injury (SCI) or tetraplegia, brain stem stroke and other diseases.

For this wheelchair control, at least three commands are necessary to control the wheelchair: to signal turn left, turn right and move forward. BCI-EEG is the chosen technology as being non-invasive, portable, and inexpensive. Furthermore, there have been many reports in research using SMR-based BCI. In this thesis, BCI using the mental task is the chosen method as an alternative compared to other BCI-EEG methods especially for individuals with severe disability. Some key components of the BCI have been further explored in this thesis, especially for this reason. This includes the development of the wireless embedded EEG system mental task classification, a study on computational intelligence of BCI with optimized feature extraction, and classification algorithms, and a study of the improved classifier together with the inclusion of tetraplegia patients as the main target group.

BCI-EEG instrumentation covers aspects in which brain signals are acquired, measured, digitised, stored, and analysed. It includes the sensors that detect the electrical brain activity, in this case an EEG, an amplifier, analog to digital converter (ADC), a computer to process the digitized signals, real-time computational intelligence (signal pre-processing, features extraction and classification), and the output conversion that achieve the user's intent. BCI typically make use of a personal computer (PC) for the sophisticated signal processing and computational intelligence required for the classifier. However, with the rapid development of embedded systems (microcontroller-based systems) and wireless technology, a microcontroller-based BCI system with a separate head-mounted battery-operated wireless EEG sensor provides maximum portability, convenience and cost effectiveness. This study has been designed with portable instrumentations comprised of a two channels wireless EEG system, wireless receiver and embedded system for mental task classification. The developed microcontroller-based BCI has maximized the aspects of portability, low power, real-time operation and low cost. The developed wireless EEG provides a good CMRR performance, a compact size and a low current consumption coin cell battery for power.

The testings have been done first for the developed system by connecting with the EEG simulator and applying a sinusoid signal with the amplitudes in the EEG range.

For EEG measurement, it is well-known that there is an increase of alpha wave on the visual cortex which is in the occipital lobe for most people during closed eyes action. As a result, the dominant alpha wave during eyes closed can be used for the testing. The results have shown that the developed BCI system was able to detect the dominant alpha wave between 8-13Hz during eyes closed on the occipital lobe O2. The developed system also has provided a comparable result with the medical EEG such as Procomp2-Thought Technology. Further testing of the system was done to measure other bio-potential signals, electrocardiography (ECG) for the heart signals measurement. The result has shown a clear comparable ECG signal, PQRST waves between the developed systems compared to the medical ECG, Compumedics-Siesta system.

In the first experiment, mental tasks data was collected from six healthy participants aged between 25 and 35 using the developed BCI prototype. The relevant mental tasks used are arithmetic, figure rotation, letter composing and counting task with additional eyes closed task. For the computational intelligence, the features extraction in this first experiment was based on FFT and the mental tasks were classified using a feed-forward back-propagation artificial neural network (ANN) trained with the Levenberg-Marquardt algorithm. The overall classification for the closed and open eyes action has reached high accuracy. The dominant alpha wave during eyes closed action is good enough as a command to switch the BCI system on-off or as an emergency stop. An overall accuracy of around 70% was achieved with bit rate at around 0.4 bits/trial for six subjects tested to select between three mental tasks combinations. The FFT algorithm and the final weights of ANN was applied to the embedded system to provide a real-time mental task and eyes closed task classification/ detection. This showed the developed wireless embedded EEG system is able to be used for the BCI using mental task classification.

The study was continued to the second experiment for BCI optimization using Hilbert Huang transform (HHT) as the feature extractor and genetic algorithm (GA) to optimize the ANN to form the GA-ANN classifier. A total of five able-bodied subjects aged between 22 and 40 years participated in the experiment. The medical EEG,

Compumedic-Siesta, was used with channels that were attached to the scalp at locations *C3*, *C4*, *P3*, *P4*, *O1*, *O2*, *T3*, and *T4*. A reference and GND electrodes were placed at locations *A2* and *A1* according to the standard international 10-20 system. More non motor imagery mental tasks were used, including: arithmetic, letter composing, Rubik's cube rolling, visual counting, ringtone, and spatial navigation. An additional eyes closed task was also recorded for the HHT feature extraction testing. Initial testing of the feature extractor showed that the HHT composed correctly for the EMD process with the known combinations of sinusoid signal.

From the results of the second experiment, the HHT testing for the eyes closed EEG signal showed a correct dominant alpha (8-13Hz) wave with high classification. The result for the three chosen discriminative tasks classification across combinations of mental tasks showed, using the GA-ANN classifier and FFT as the feature extractor on the 8 channels EEG, accuracy between 76% and 85%. With only two channels in combination of the best chosen task using an FFT feature extractor and the same GA-ANN, the accuracy was reduced to measure between 65% and 79%. However, the HHT based method provided an improved accuracy between 70% and 84% on the user chosen combination tasks for the classification of three discriminative tasks using two EEG channels.

The third experiment was the study for the advanced BCI system, particularly for the advanced classifier using fuzzy particle swarm optimization with cross mutated-based artificial neural network (FPSOCM-ANN). This experiment also focused on the involvement of patients with tetraplegia as the target group of the BCI system. A total of ten participants were involved: five able-bodied subjects aged between 25 and 35 years and five patients with tetraplegia aged between 45 and 80 years who suffer a high level of SCI in the cervical area. Previously in the second experiment, the study used the best user-chosen three task combination. The third experiment focused on specific three mental tasks only for mental task generalization purposes. The three relevant mental tasks used for the BCI concentrated on mental letter composing, mental arithmetic and mental Rubik's cube rolling forward, and these were associated with

three wheelchair commands: left, right and forward, respectively. An eyes closed task was also recorded for testing and used as an additional on/off command. The Compumedics-Siesta medical EEG system was used with the sampling rate of the system at 256 Hz. This study used 6 channels with the electrodes positioned at locations C3, C4, P3, P4, O1 and O2. The left earlobe (A1) was used as the reference, and the GND electrode was attached to the right earlobe (A2). The HHT was used for the feature extractor. For comparison features extractor purpose, FFT and wavelet were also included. The classifier was based on the FPSOCM-ANN. For the classifier comparison, other classifiers, including GA-ANN, multi-class linear perceptron, multi-class LDA, and multi-class BSVM were also included. Different time-windows of data were investigated to find the best data windowing with an improved result of classification accuracy.

The HHT testing for the eyes closed EEG signal shows a correct dominant alpha (8-13Hz) wave with high classification accuracy for both groups (able bodied subjects and patients with tetraplegia). Results for three mental tasks classification showed improved accuracy of FPSOCM-ANN compared to GA-ANN and other classifiers in multi-class mode (SVM, LDA and linear perceptron) across both groups and different time-windows. Although the patients group had lower classification accuracy, this was improved by increasing the time windows with the best classification accuracy achieved in a 7s time-window. The HHT as the feature extractor has resulted in a better method of accuracy compared to FFT and wavelet methods. In the FPSOCM-ANN, it can also be seen that there was a comparable accuracy between HHT-EMD and HHT-EEMD in this study, thus both methods can be used.

The neural networks classifiers, GA-ANN and FPSOCM-ANN with the optimization methods in this study provided better results compared to the linear perceptron, LDA and SVM methods. With the HHT as the feature extractor, the FPSOCM-ANN provided the best classifier among other classifiers. The FPSOCM-ANN provided improved accuracies compared to the genetic algorithm-based artificial neural network (GA-ANN), SVM, LDA, linear perceptron for three mental tasks-based

BCI classification with the best classification accuracy achieved for a 7s time-window: 84.4% (FPSOCM-ANN) compared to 77.4% (GA-ANN), 77.0% (SVM), 72.1%(LDA), and 71.0%(linear perceptron). The result shows the FPSOCM-ANN classifier compared to GA-ANN, SVM, LDA and linear perceptron with all of the p-values being less than 0.05. A p-value less than 0.05 is considered as statistically significant with 95% confidence level.

The result shows comparable accuracies only for able-bodied subjects and patients with tetraplegia, who have been provided with their portions of dataset for the training of the classifier, while for the participants from both groups who are treated as new users of the system, a drop in accuracies was found. This showed the inter-subject variability issue for the EEG-based BCI system. As a result the subject's specifically trained classifier is preferable. The other option is the globally trained classifier with adaptive re-training to ensure the classifier is capable of generalising the data from the new subject, which was previously unseen by the classifier.

For the computation time in terms of classification/execution, although the training of ANN-based classifiers (FPSOCM-ANN and GA-ANN) was slower than the other compared classifiers (linear, LDA, SVM), all classifiers were able to complete the task in less than a second. The reason that ANN-based classifiers (GA-ANN and FPSOCM-ANN) were able to perform as fast as other classifiers was because after the ANN training, final weights were treated as constants in the ANN feed-forward classification routine. The time-window was needed for the BCI real-time operation, with the best at 7s. As a result, there was sufficient time available, even using a 1s time-window for FPSOCM-ANN as the best classifier with highest accuracy to perform continuous real-time classification until the next time-window elapsed.

Furthermore, in early BCI research, subjects had to undergo intensive training in order to acquire the new skill of operating a BCI without using advanced methods of computational intelligence. The machine learning technique has shortened the subject's training procedure, especially using the improved classifier FPSOCM-ANN. The training of the classifier can be done automatically as the machine learning technique.

After this classifier training is done, all the weights (parameters) are saved as constants. For the operation of real-time classification, it is not necessary to do the classifier training anymore. The classifier just needs to compute the feed forward ANN function with the saved weight parameters and particular features in a specified time-window for the pattern classification system. Thus, FPSOCM-ANN classification time in the runtime mode (execution) is fast, taking less than a second.

For practical use of a BCI, the combination of two channels EEG has also been presented with the variation result of the best two channels in each subject. For overall generalisation of both groups, O1 and C4 are the best two channels at 80.5% of accuracy, followed by the second best at P3 and O2 at 76.4% of accuracy, and the third best at C3 and O2 channels at 75.4% of accuracy. For each correctly classified mental task, the arithmetic task has the highest rate of classification, followed by the rolling cube task and letter composing task. In the online application as future work, these three non-motor imagery mental tasks can be directly mapped into three wheelchair commands.

The classifier training is done automatically by using the FPSOCM-ANN classifier as a machine learning procedure. The weights for the best accuracy are saved as constants to provide a feed-forward ANN classification. For the classification, it is a straightforward calculation and does not require training. Therefore, the classification function is the final detection of the mental task, and the calculation takes less than a second.

6.2 Future Work

In terms of the hardware, an improvement can be further done for the instrumentation, particularly with the EEG electrodes. The gels used with the EEG electrodes are necessary, but not ideal for a long term recording type of application, because they usually dry out after a long period of time. Also, skin preparation is needed if using the gel-based electrode system. A dry and contactless sensor technology can be used to improve the BCI instrumentation further.

In terms of the operation for wheelchair control, a combination between mental task-based BCI with the autonomous wheelchair technology can be explored further to create a semi-autonomous wheelchair share control with sensors on the wheelchair to avoid obstacles, which ensure the safety aspect in a real-time system.

For a further application, in this thesis, the focus of the BCI is for wheelchair control using a mental task paradigm for severely disabled individuals such as those with tetraplegia. This basically is the use of BCI as a replacement function. BCI replaces natural output that has been lost as a result of injury or disease. More applications can be explored using BCI technology. Besides acting as a hand-free technology for wheelchair control, BCI can also be applied as a replacement function to a person who can no longer speak for communication, such as to translating typed words into voice by combining with a speech synthesizer.

There is a possibility for a BCI to be applied to restore lost natural output, such as SCI patients whose arms and hands are paralysed, and who might use a BCI to stimulate the paralysed muscles to move the limbs.

A different study can be explored for BCI to be used as a natural enhancement to the central nervous system. For example, the BCI could be used when a person is performing a driving task which requires continuous attention. Here the BCI can be used for fatigue detection and an output can alert a driver, thus preventing attention lapses which might cause a traffic accident. The BCI in this application would enhance the natural central nervous system output.

In another application, a BCI might be used as a supplement to the natural nervous system output. For example, for controlling the position of a computer cursor with hand-operated joystick/mouse, a BCI might be used to select items that the cursor reaches. In this case, the BCI supplements natural neuromuscular output with an additional artificial output.

A BCI might also be used to improve the nature of the nervous system, such as in stroke rehabilitation. A person whose arm movements have been impaired by a stroke may involve a BCI that can measure signals from the damaged cortical area, around the sensorimotor cortex area and then stimulate muscles to improve arm movements. The BCI in this application might help the stroke survivor to regain more normal arm control.

Appendix A

Research Ethics Clearance

A.1 Ethics Clearance Letter

30 June 2011

Professor Hung Nguyen
Engineering & Information Technology
CB10.03.573
UNIVERSITY OF TECHNOLOGY, SYDNEY

Dear Hung,

UTS HREC 2011-217 – NGUYEN, Professor Hung, HUNTER, Dr. Gregory, LING, Dr Steve (for CHAI, Mr Rifai, PhD student) – “Brain Computer Interface for Wheelchair Control”

Thank you for your response to my email dated 22/06/11. Your response satisfactorily addresses the concerns and questions raised by the Committee, and I am pleased to inform you that ethics clearance is now granted.

Your clearance number is UTS HREC REF NO. 2011-217A

Please note that the ethical conduct of research is an on-going process. The *National Statement on Ethical Conduct in Research Involving Humans* requires us to obtain a report about the progress of the research, and in particular about any changes to the research which may have ethical implications. This report form must be completed at least annually, and at the end of the project (if it takes more than a year). The Ethics Secretariat will contact you when it is time to complete your first report.

I also refer you to the AVCC guidelines relating to the storage of data, which require that data be kept for a minimum of 5 years after publication of research. However, in NSW, longer retention requirements are required for research on human subjects with potential long-term effects, research with long-term environmental effects, or research considered of national or international significance, importance, or controversy. If the data from this research project falls into one of these categories, contact University Records for advice on long-term retention.

If you have any queries about your ethics clearance, or require any amendments to your research in the future, please do not hesitate to contact the Ethics Secretariat at the Research and Innovation Office, on 02 9514 9772.

Yours sincerely,

Professor Marion Haas
Chairperson
UTS Human Research Ethics Committee

A.2 Ethics Amendment Clearance Letter

Subject: Eth: HREC Amendment Clearance Letter - UTS HREC 2011-217A
From: Ethics Secretariat <Research.Ethics@uts.edu.au>
Date: 11/11/2011 3:50 PM
To: Prof Hung T Nguyen <Hung.Nguyen@uts.edu.au>
CC: Ethics Secretariat <Research.Ethics@uts.edu.au>, Mr Rifai Chai <Rifai.Chai@student.uts.edu.au>

Dear Hung and Rifai,

Re: UTS HREC REF NO. 2011-217A - "Brain Computer Interface for Wheelchair Control"

The UTS Human Research Ethics Committee reviewed your amendment application and approved your request to amend the application as follows:

1. Use of two EEG systems (wireless EEG and Clinical EEG);
2. Greater number of mental cognitive tasks involved;
3. Able body subjects will have their interviews go for 2-3 hours whereas disabled subjects interviews will go for 1-2 hours;
4. Increased number of participants increased from 5 to 20.

You should consider this your official letter of approval. If you require a hardcopy please contact the Research Ethics Officer (Research.Ethics@uts.edu.au).

If you wish to make any further changes to your research, please contact the Research Ethics Officer in the Research and Innovation Office, Ms Racheal Laugery on 02 9514 9772.

In the meantime I take this opportunity to wish you well with the remainder of your research.

Yours sincerely,

Professor Marion Haas
Chairperson
UTS Human Research Ethics Committee

C/- Research & Innovation Office
University of Technology, Sydney
Level 14, Tower Building
Broadway NSW 2007
Ph: 02 9514 9772
Fax: 02 9514 1244
Web: <http://www.research.uts.edu.au/policies/restricted/ethics.htm>

A.3 Information Sheet



UNIVERSITY OF TECHNOLOGY, SYDNEY

INFORMATION SHEET

BRAIN COMPUTER INTERFACE FOR WHEELCHAIR CONTROL

WHO IS DOING THE RESEARCH?

My name is Rifai Chai and I am a research student at UTS. My supervisors are Prof. Hung Nguyen as principal supervisor, Dr. Greg Hunter and Dr. Steve Ling as co-supervisor.

WHAT IS THIS RESEARCH ABOUT?

The aim of the research is to develop a prototype of Brain Computer Interface (BCI) for controlling an electric wheelchair using thought patterns consisting of different mental tasks.

WHAT WILL IT INVOLVE?

This will involve data collection using the developed wireless EEG or clinical EEG in a single session lasting 1-3 hours which I will be seated on a chair and asked to perform several mental cognitive tasks as guided by the researcher. At the start of the session, electrodes for EEG measurement will be attached and remain there for the duration of the session. At each electrode location, my hair will be parted, Skin prepping and EEG gels will be applied and the electrode placed on top. Additional two electrodes will be attached to face area for EOG detection if necessary. The mental tasks I will be asked to perform will include mental arithmetic (non-trivial and trivial), Rubik's cube rolling forward, and figure rotation (3D car), mental visual counting, letter composing, audio (ring tone) imagery, familiar face imagery, spatial navigation, eyes closed and eyes opened actions. Fewer tasks will be used for disabled body participants; therefore the duration of the data collection is adjusted at 1-2 hours.

ARE THERE ANY RISKS?

During data collection, participants will be asked to keep minimum of unnecessary movement and eyes blinks which may cause fatigue. Skin prepping and EEG gels will be applied to the specific area on the head therefore some residue of the gel may be stayed on the hair after data collection.

HOW TO MINIMISE RISKS?

The participants will be supervised all times. There will be a gap between each session for participant to relax. If participants are experiencing fatigue or any kinds of discomforts during data collection, the experiment will be stopped immediately. After data collection, gels will be cleaned off and participants will be given the opportunity to wash their hair.

IS THERE ANY OBLIGATION TO PARTICIPATE?

There is no obligation to participate in this research. Participants can withdraw at any time without giving a reason.

WHAT IF THERE ARE CONCERNS OR COMPLAINTS?

If any concerns about the research, please feel free to contact us on : 95147531 (Rifai Chai), 9514 2316 (Dr. Greg Hunter), 9514 2435 (Dr. Steve Ling) and 9514 4441 (Prof. Hung Nguyen).

If you would like to talk to someone who is not connected with the research, you may contact the Research Ethics Officer on 02 9514 9772.

A.4 Consent Form



UNIVERSITY OF TECHNOLOGY, SYDNEY CONSENT FORM

I _____ agree to participate in the research project "Brain Computer Interface for Wheelchair Control" being conducted by Prof. Hung Nguyen (CB10.03.573, ph: 9514 4441), Dr. Greg Hunter (CB01.18.23, Ph 9514 2316), Dr. Steve Ling (CB01.24.10E ph: 9514 2435) and Rifai Chai (CB01.18.01 Ph 95147531) of the University of Technology, Sydney as part of a doctoral research project.

I understand that the purpose of this study is to develop a prototype for controlling an electric wheelchair using thought patterns only. This system, called a Brain Computer Interface (BCI), comprises a battery operated wireless Electroencephalogram (EEG) and a separate controller with a wireless receiver which collects the EEG signals and translates them into commands for wheelchair control. Additional clinical EEG is used as well for the comparison, optimization and improvement of the developed BCI prototype.

I understand that my participation in this research will involve data collection using the developed wireless EEG or clinical EEG in a single session lasting 2-3 hours which I will be seated on a chair and asked to perform several mental cognitive tasks as guided by the researcher. At the start of the session, electrodes for EEG measurement will be attached and remain there for the duration of the session. At each electrode location, my hair will be parted, Skin prepping and EEG gels will be applied and the electrode placed on top. Additional two electrodes will be attached to face area for EOG detection if necessary. The mental tasks I will be asked to perform will include mental arithmetic (non-trivial and trivial), Rubik's cube rolling forward, and figure rotation (3D car), mental visual counting, letter composing, audio (ring tone) imagery, familiar face imagery, spatial navigation, eyes closed and eyes opened actions. Fewer tasks will be used for disabled body participants; therefore the duration of the data collection is adjusted at 1-2 hours.

I am aware that I can contact Prof. Hung Nguyen, Dr. Greg Hunter, Dr. Steve Ling and Rifai Chai if I have any concerns about the research. I also understand that I am free to withdraw my participation from this research project at any time I wish.

I confirm that all my questions have been fully and clearly answered.

I agree that the research data gathered from this project may be published in a form that does not identify me in any way.

_____/_____/_____
Signature (participant / caregiver / authorised representative)

_____/_____/_____
Signature (researcher / delegate)

NOTE:

This study has been approved by the Human Research Ethics Committee of the University of Technology, Sydney. If you have any complaints or reservations about any aspect of your participation in this research which you cannot resolve with the researcher, you may contact the Ethics Committee through the Research Ethics Officer (ph: +61 2 9514 9772 Research.Ethics@uts.edu.au), and quote the UTS HREC reference number. Any complaint you make will be treated in confidence and investigated and you will be informed of the outcome.

Appendix B

Firmware and Software Code

B.1 Firmware of Wireless EEG

```
/******  
head.c  
-----  
(C) Aug 2010 by Rifai Chai  
  
For wireless EEG V2  
Send data to UART0 using protocol (17 Bytes)  
Interrupt timer1 to create 256Hz of sampling rate  
Interrupt ADC 12-bit differential  
Interrupt UART Tx to transmit the data byte to UART0  
Interrupt RF (RFIRQ)  
*****/  
#define INITIALIZE  
#include "general.h"  
#include <Nordic\reg24le1.h>  
#include <stdint.h>  
#include <stdbool.h>  
#include <nordic_common.h>  
#include "head.h"  
#include "com.h"  
#include "radio.h"  
#include "timer.h"  
#include "uart_new.h"  
#include "adc.h"  
#include "delay.h"
```

```

#include "hal_adc.h"

char const channel_order[]= { 0, 3, 1, 4, 2, 5 };
uint8_t xdata TXBuf[PACKETLEN];
volatile uint8_t TXIndex,ADC_cnt;
volatile uint8_t CurrentCh;
volatile uint16_t buff; //buffer for each reading
volatile uint16_t out; //return of total reading, left justified

void disconnect_p0_input()
{
    P0CON = 0x70;
    P0CON = 0x71;
    P0CON = 0x72;
    P0CON = 0x73;
    P0CON = 0x74;
    P0CON = 0x75;
    P0CON = 0x76;
    P0CON = 0x77;
}

void disconnect_p1_input()
{
    P1CON = 0x70;
    P1CON = 0x71;
    P1CON = 0x72;
    P1CON = 0x73;
    P1CON = 0x74;
    P1CON = 0x75;
    P1CON = 0x76;
    P1CON = 0x77;
}

//*****
// Initialize buffer for EEG protocol
//*****
void Init_protocol(void)
{
    TXBuf[0] = 0xa5;           //Sync 0
    TXBuf[1] = 0x5a;           //Sync 1
    TXBuf[2] = 2;              //Protocol version
    TXBuf[3] = 0;              //Packet counter
    TXBuf[4] = 0;              //ch1 hi
    TXBuf[5] = 0;              //ch1 lo
    TXBuf[6] = 0;              //ch2 hi
    TXBuf[7] = 0;              //ch2 lo
    TXBuf[8] = 0;              //reserve
    TXBuf[9] = 0;              //reserve
    TXBuf[10] = 0;             //reserve
    TXBuf[11] = 0;             //reserve
    TXBuf[12] = 0;             //reserve
    TXBuf[13] = 0;             //reserve
    TXBuf[14] = 0;             //reserve
    TXBuf[15] = 0;             //reserve
    TXBuf[16] = 0;             //reserve
}

```

```

//*****
// RTC2 interrupt service routine
//*****
void rtc_interrupt(void) interrupt INTERRUPT_TICK
{
//   bit_toggle(P15);
//   uart_putc(0x32);

    CurrentCh = 0;
    TXBuf[3]++;
    TXBuf[2 * NUMCHANNELS + HEADERLEN]=0; // swicth
    out=0;
    buff =0;
    ADC_cnt=0;
    hal_adc_set_input_channel(HAL_ADC_INP_AIN10);
//AIN10-channel1 EEG
    hal_adc_start();
    MISC = 1; //Enable interrupt ADC
}

//*****
// ADC interrupt sevice routine
//*****
void adc_irq(void) interrupt INTERRUPT_MISCIQ
{
    volatile uint8_t i;

    ADC_cnt++;
    buff = (ADCDATL) + (((uint16_t) ADCDATH)<<8);
    out +=buff;

    if (ADC_cnt ==8){
        out = out >>3;
        i = 2 * CurrentCh + HEADERLEN;
        TXBuf[i+1] = LSB(out);
        TXBuf[i] = MSB(out);
        i = 2 * CurrentCh + HEADERLEN;
        CurrentCh++;
        ADC_cnt=0;
    }
    if (CurrentCh < NUMCHANNELS){
        if (CurrentCh == 0){
            hal_adc_set_input_channel(HAL_ADC_INP_AIN10);
// AIN0-channel12 EEG
        }
        else if (CurrentCh == 1){
            hal_adc_set_input_channel(HAL_ADC_INP_AIN1);
// AIN0-channel11 EEG
        }

        hal_adc_start();
        MISC = 1; //Enable interrupt ADC for the next channel
    }
    else{

```

```

        MISC = 0;           // Disable ADC interrupt
        radio_init(1);
        RadioOn();
        send_packet(TXBuf); // Send data wirelessly RFIRQ
        TI0=0;             // clear Tx flag
        ES0 = 1;           // UART interrupt

        S0BUF = TXBuf[0];
        TXIndex = 1;
    }
}

//*****
// UART interrupt service routine
//*****
void uart0_irq(void) interrupt INTERRUPT_UART0
{
    if (RI0 == 1){
        RI0=0;
    }

    if (TI0 == 1){ // TX interrupt
        TI0=0;
        S0BUF= TXBuf[TXIndex];
        TXIndex++;
        if (TXIndex >=PACKETLEN){
            ES0 = 0;
        }
    }
}

//*****
// RF interrupt service routine
//*****
void radio_irq(void) interrupt RF_READY_INT_VECT
{
    radio_interrupt();
}

//*****
// Main
// Handles startup and run
//*****

void main(void)
{
    delay_ms(10); // stabilise the power
    rtc2_init();
    CLKCTRL = 0x20; //set system clock to xtal only
    CLKLFCTRL = 0x01; //set low frequency clock source for RTC2
    HeadParm.bPowerUp = 1; //set power up flag to 1
    disconnect_p0_input();
    disconnect_p1_input();

//Init RTC2

```

```

    RTC2CON = 0;          //stop RTC2
    RTC2CMP1 = 0x72;     //set comparator to 33,333 or 0x7235
    RTC2CMP0 = 0x35;
    RTC2CON = 0x05;     //start RTC2 in continuous run
    RTC2CON = 0 0101b

//read RTC2
    RTC2CON |= (1 << 4);

// Initilize Buffer for testing of Interrupt UART
    Init_protocol();

//set P0.3 as UART TXD output
    P0DIR &= ~0x0C;
    P02 = 1; // turn on AD8553 and OPA333

//initialize UART
    uart_init(UART_BAUD_57K6);
    uart_puts("\r\nwireless EEGV2.1 Head\n\r");

// Initilize RF
    RFCKEN = 1;        // enable L01 clock
    RF = 1;           // enable RF interrupt
    radio_init(1);    // Enter PTX mode (Transmitter = 1)
    delay_ms(10);    // RF initialization delay
    timer1_init();   // initialization timer1 of 16bit for 256Hz
    timer0_init();   // interrupt every 10ms

//Configure ADC
    AdcInit();
    hal_adc_set_input_channel(HAL_ADC_INP_AIN10);
    hal_adc_set_reference(HAL_ADC_REF_INT);
    hal_adc_set_input_mode(HAL_ADC_DIFF_AIN6);
    //Differential; AIN6 as input
    hal_adc_set_conversion_mode(HAL_ADC_SINGLE_STEP);
    hal_adc_set_power_down_delay(HAL_ADC_PDD_24US);
    hal_adc_set_acq_window(HAL_ADC_AQW_075US);
    //Input acquisition window = 12us
    hal_adc_set_resolution(HAL_ADC_RES_12BIT);
    //resolution 12 bit
    hal_adc_set_data_just(HAL_ADC_JUST_RIGHT);
    //right justify
// Enable MISC interrupts to enable ADC interrupt
    MISC = 1;
// Start ADC
    hal_adc_start();
// start global interrupt
    EA=1;

    while( 1 ){
        EA=0;
        PWRDWN = 0x04;//register retention
        PWRDWN = 0x07;//standby
        EA=1;
    }
}

```

```

/*****
adc.c
-----
ADC functions
*****/
#include "general.h"
#include <Nordic\reg24le1.h>
#include <stdint.h>
#include <stdbool.h>
#include "adc.h"
#include "uart.h"

uint16_t adc_hi(uint8_t set);

/*****
// Function to initialize analog to digital converter
*****/
void AdcInit(void)
{
//configure digital inputs, assume direction bits set as inputs
//disable input buffer, pin 0.0
    P0CON = 0x70; // 0111 0000
//disable input buffer, pin 0.1
    P0CON = 0x71; // 0111 0001
//disable input buffer, pin 0.3
    P0CON = 0x73; // 0111 0011
//set ADCCON3: 12 bit resolution, right justified output
    ADCCON3 = 0xE0; // 1110 0000
}

/*****
// Function to read ADC value with input parameter of channel
*****/
uint16_t AdcRead(uint8_t channel)
{
    uint16_t out; //return of total reading, left justified
    uint16_t buff; //buffer for each reading
    uint8_t j,set;
//initialize variables
    out = 0;
//read depending on channel selection
    switch (channel)
    {
        case BATTERY:{
            ADCCON2 = 0x8A; // 0000 1010
            set = 0x80 + (BATTERY<<2); // 0100 0000 + channel<<2
            ADCCON1 = set;
            //wait for busy bit to settle
            j = 1;
            while(j--);
            //wait for end of conversion
            while((ADCCON1&0x40) !=0);
            //read and move to total
            buff = (ADCDATL) + (((uint16_t) ADCDATH)<<8);
            out=buff;
            break;
        }

        case ACCELX:{
            ADCCON2 = 0x8A; // diff mode with AIN6

```

```

        set = 0x80 + (BATTERY<<2);
        out = adc_hi(set);
        out = out<<2;
        break;
    }

    case ACCELY:{
        ADCCON2 = 0x0A; // 0000 1010
        //setting for ADCCON1: VDD ref
        set = 0x81 + (2<<2); // 1000 0001 + channel<<2
        //run high resolution conversion
        out = adc_hi(set);
        break;
    }

    case ACCELZ:{
        ADCCON2 = 0x0A; // 0000 1010
        //setting for ADCCON1: VDD ref
        set = 0x81 + (0<<2); // 1000 0001 + channel<<2
        //run high resolution conversion
        out = adc_hi(set);
        break;
    }

    case AMP1:{
        ADCCON2 = 0x8A; // 1000 1010
        //setting for ADCCON1
        set = 0x80 + (AMP1<<2); // 1000 0000 + channel<<2
        //run high resolution conversion
        out = adc_hi(set);
        break;
    }

    default:{
        out = 0;
        break;
    }
}
return out;
}

//-----
// Function for ADC conversion (high resolution)
//-----
uint16_t adc_hi(uint8_t set)
{
    uint16_t buff; //buffer for each reading
    uint16_t out; //return of total reading, left justified
    uint8_t i,j;
    //initialize variables
    out = 0;
    //start conversion by writing to ADCCON1
    ADCCON1 = set;
    //wait for busy bit to settle
    j = 1;
    while(j--);
    //wait for end of conversion
    while((ADCCON1&0x40) !=0);
    //run 4 conversions, channel 1 and sum results

```

```

        for(i=0;i<4;i++){
            ADCCON1 = set;
//wait for busy bit to settle
            j = 1;
            while(j--);
//wait for end of conversion
            while((ADCCON1&0x40) !=0);
//read and add to total
            buff = (ADCDATL) + (((uint16_t) ADCDATH)<<8);
            out += buff;
        }
        return out;
    }
}

/*****
delay.c
-----
Delay functions
*****/
#include "delay.h"

void delay_us(unsigned short microseconds)
{
    unsigned short count;

    if(microseconds == 0 || microseconds == 1)
        return;
    else
        microseconds--;

    for(count = 0; count < microseconds; count++)
    {
        NOP();
        NOP();
    }
}

void delay_ms(unsigned short milliseconds)
{
    unsigned short count;

    for(count = 0; count < milliseconds; count++)
    {
        delay_us(1000);
    }
}

void delay_s(unsigned short seconds)
{
    unsigned short count;

    for(count = 0; count < seconds; count++)
    {
        delay_ms(1000);
    }
}

```

```

/*****
radio.c
-----
Wireless RF functions
*****/
#define TRUE 1
#define FALSE 0
#include "nordic_common.h"
#include "hal_nrf.h"
#include "radio.h"
#include "uart_new.h"
#include "general.h"

code const uint8_t address[HAL_NRF_AW_4BYTES] = {0x00,0x00,0x00,0x04};
static uint8_t radio_status;          // Global radio status

//debug
uint8_t irq_flags;
uint8_t xdata packet[RADIO_MAX_PL];
extern uint8_t xdata TXBuf[PACKETLEN];

/*****
// Function to turn on RF power
*****/
void RadioOn(void)
{
//set CE low
CE_LOW();
//power up device
hal_nrf_set_power_mode(HAL_NRF_PWR_UP);
//wait until it is ready
while(radio_busy());
}

/*****
// Function to turn off RF power
*****/
void RadioOff(void)
{
hal_nrf_set_power_mode(HAL_NRF_PWR_DOWN);
}

/*****
// Function to configure TX and RX address
*****/
void RadioAddress(uint8_t *address)
{
//load TX address
hal_nrf_set_address(HAL_NRF_TX, address);
//load pipe 0 address
hal_nrf_set_address(HAL_NRF_PIPE0, address);
}

/*****
// Function to prepare and load data for the RF transmission
*****/
void RadioTXLoad(uint8_t *bufPt, uint8_t bytes)
{
hal_nrf_flush_tx();

```

```

//load buffer without acknowledge
hal_nrf_write_tx_payload_noack(bufPt,bytes);
}

//*****
// Function to turn the RF transmission on
//*****
void RadioTXOn(void)
{
//set CE low
CE_LOW();
//switch to transmit mode
hal_nrf_set_operation_mode(HAL_NRF_PTX);
}

//*****
// Function to transmit data and wait for completion of RF
//*****
void RadioTXSend(void)
{
while(hal_nrf_tx_fifo_empty()!=1)
{
//pulse CE high to send packet
CE_PULSE();
//set status to busy, waiting for interrupt
radio_status = RF_BUSY;
//wait until radio ready
while(radio_busy());
}
}

//*****
// Function to load data into buffer for the RF transmission
//*****
void RadioTXBuf(uint8_t *bufPt, uint8_t bytes)
{
RadioTXLoad(bufPt,bytes);
RadioTXOn();
RadioTXSend();
RadioRXOn();
}

//*****
// Function to set the RF in receive mode
//*****
void RadioRXOn(void)
{
CE_LOW();
hal_nrf_set_operation_mode(HAL_NRF_PRX);
CE_HIGH();
}

//*****
// Function for RF data receiving
//*****
uint8_t RadioRXBuf(uint8_t *rxBufPt)
{
uint8_t r; //return value
uint8_t i;

```

```

//set return value for no data received
r = 0xFF;
//test if packet received
if(radio_status == HAL_NRF_RX_DR)
{
//packet received (0,1)#
//load packet into data buffer and retrieve length
r = LSB(hal_nrf_read_rx_payload(rxBufPt));
//set status to idle if no more data
if (hal_nrf_get_rx_fifo_status() == FIFO_EMPTY)
{
radio_status = RF_IDLE;
}
for (i=0;i<50;i++) ; //53.3 for 10us. must be > 30
}
return r;
}

//*****
// Function to initialize the RF
//*****
void radio_init(uint8_t operation_mode)
{
CE_LOW(); // Disable radio
hal_nrf_enable_ack_payload(true); // Enable ack
hal_nrf_enable_dynamic_ack(true); // Enable dynamic ack
hal_nrf_enable_dynamic_payload(true); // Enable dynamic payload
hal_nrf_setup_dynamic_payload(0xFF); // All pipes uses dynamic ack.
hal_nrf_close_pipe(HAL_NRF_ALL); // First close all radio pipes
hal_nrf_open_pipe(HAL_NRF_PIPE0, true); // Open pipe0, with autoack
hal_nrf_set_crc_mode(HAL_NRF_CRC_16BIT);
hal_nrf_set_auto_retr(15, 250);
hal_nrf_set_address_width(HAL_NRF_AW_4BYTES);
hal_nrf_set_address(HAL_NRF_TX, address); // Set device's addresses
hal_nrf_set_address(HAL_NRF_PIPE0, address);
if(operation_mode == 0) // Mode dependant settings
{
hal_nrf_set_operation_mode(HAL_NRF_PRX); // Enter RX mode
CE_HIGH(); // Enable radio
}
else // Mode dependant settings
{
hal_nrf_set_operation_mode(HAL_NRF_PTX); // Enter TX mode
}
hal_nrf_set_rf_channel(25); // Using channel 25
hal_nrf_set_power_mode(HAL_NRF_PWR_UP); // Power up device
// hal_nrf_set_output_power(HAL_NRF_6DBM); // Power TX -6dBm
radio_status = RF_IDLE; // Radio ready, i.e.
RF_IDLE
}

bool radio_busy(void)
{
return(radio_status == RF_BUSY);
}

```

```

//*****
// Function for RF interrupt
//*****
void radio_interrupt(void)
{
//if(radio_status == RF_BUSY) uart_puts("\r\nI");
//set CE low to stop activity- no good
// CE_LOW();
irq_flags = hal_nrf_get_clear_irq_flags();
switch(irq_flags)
{
case (1<<HAL_NRF_MAX_RT): // Max retries reach LED2_flash
hal_nrf_flush_tx(); // Flush tx fifo, avoid jamming
radio_status = HAL_NRF_MAX_RT;
break;

case (1<<HAL_NRF_TX_DS): // Packet sent, LED1_flash
RadioOff();
PWRDWN = 0x04;
radio_status = HAL_NRF_TX_DS;
break;

case (1<<HAL_NRF_RX_DR): // Packet received, LED1_flash
radio_status = HAL_NRF_RX_DR;
break;

default:
break;
}
}

//*****
// Function to send the packet of RF data
//*****
void send_packet(uint8_t *Buf)
{
RadioTXLoad(Buf, PACKETLEN);
RadioTXOn();
if (hal_nrf_tx_fifo_empty() != 1)
CE_PULSE();
}

```

```

/*****
uart.c
-----
UART functions
*****/
#include <Nordic\reg24le1.h>
#include "uart.h"
#include <stdint.h>
#include <stdbool.h>

#define BAUD_57K6 1015 // = Round(1024 - (2*16e6)/(64*57600))
#define BAUD_38K4 1011 // = Round(1024 - (2*16e6)/(64*38400))
#define BAUD_19K2 998 // = Round(1024 - (2*16e6)/(64*19200))
#define BAUD_9K6 972 // = Round(1024 - (2*16e6)/(64*9600))

void uart_off(void) // Turn off UART
{
    REN0 = 0; // Disable receiver
}

void uart_init(hal_uart_baudrate_t baud)
{
    uint16_t temp;

    ES0 = 0; // Disable UART0 interrupt while initializing
    REN0 = 1; // Enable receiver
    SM0 = 0; // Mode 1..
    SM1 = 1; // ..8 bit variable baud rate
    PCON |= 0x80; // SMOD = 1
    ADCON |= 0x80; // Select internal baud rate generator
    switch(baud)
    {
        case UART_BAUD_57K6:
            temp = BAUD_57K6;
            break;
        case UART_BAUD_38K4:
            temp = BAUD_38K4;
            break;
        case UART_BAUD_9K6:
            temp = BAUD_9K6;
            break;
        case UART_BAUD_19K2:
        default:
            temp = BAUD_19K2;
            break;
    }
    S0RELL = (uint8_t)temp;
    S0RELH = (uint8_t)(temp >> 8);
    //clear interrupt flag
    TIO = 0;
}

void uart_putc(char c) // Put Character to UART
{
    S0BUF=c;
    while(TIO==0);TIO=0;
}

char uart_getc(void) // Get Character from UART

```

```

{
    while(RI0==0);RI0=0;
    return S0BUF;
}

void uart_puts(char *s) // Put String to UART, Require uart_putc()
{
    while(*s!=0){uart_putc(*s);s++;}
}

void h2s(unsigned int i,char *s) // Convert Hex to String
{
    unsigned int temp;short len;char*p;
    len=0;p=s;
    do
    {
        temp = i&0x000F;
        if(temp < 10)
            *s=temp+'0';
        else
            *s=(temp-10)+'A';
        len++;
        s++;
        i = i >> 4;
    }while(i!=0);
    *s='x';s++;len++;
    *s='0';s++;len++;
    for(i=0;i<len/2;i++){p[len]=p[i];p[i]=p[len-1-i];p[len-1-i]=p[len];}
    p[len]=0;
}

void i2s(int i,char *s) // Convert interger to string
{
    char sign;short len;char *p;
    sign='+';len=0;p=s;
    if(i<0){sign='-';i=-i;}
    do{*s=(i%10)+'0';s++;len++;i/=10;}while(i!=0);
    if(sign=='-'){*s='-';s++;len++;}
    for(i=0;i<len/2;i++){p[len]=p[i];p[i]=p[len-1-i];p[len-1-i]=p[len];}
    p[len]=0;
}

void uart_puth(unsigned int i) // Put Hex to UART, Require
h2s(),uart_puts()
{
    char s[7];
    h2s(i,s);uart_puts(s);
}

void uart_puti(int i) // Put Integer to UART, Require
i2s(),uart_puts()
{
    char s[7];
    i2s(i,s);uart_puts(s);
}

```

B.2 Firmware of Wireless Receiver

```

/*****
chair.c
-----
(C) Aug 2010 by Rifai Chai
For wireless receiver to be paired with wireless EEG V2
*****/
#define INITIALIZE
#include "general.h"
#include <Nordic\reg24le1.h>
#include <stdint.h>
#include <stdbool.h>
#include <nordic_common.h>
#include "chair.h"
#include "com.h"
#include "radio.h"
#include "timer.h"
#include "uart.h"
#include "delay.h"
#include "spi.h"
volatile uint8_t TXBuf[PACKETLEN];
volatile uint8_t TXIndex;
volatile uint8_t TXIndex2;
volatile bool send_spi_byte = false;
volatile bool send_uart_byte = false;
//*****
// UART interrupt service routine
//*****
void uart0_irq(void) interrupt INTERRUPT_UART0
{
    if (TI0 == 1){ // TX interrupt
        TI0=0;

        TXIndex++;
        if (TXIndex >PACKETLEN){
            ESO = 0;
        }
        send_uart_byte=true;
    }
}

//*****
// RF interrupt service routine
//*****

void radio_irq(void) interrupt RF_READY_INT_VECT
{
    radio_interrupt();
}

//*****
// SPI interrupt service routine
//*****
void spi_irq(void) interrupt INTERRUPT_SERIAL
{
    delay_us(24);
    TXIndex2++;
    if (TXIndex2 < PACKETLEN){
        send_spi_byte = true;
    }
}

```

```

    }
    else{
        SPI=0;
        SpiOff();
    }
}

//*****
// main
//*****
void main(void)
{
    delay_ms(10); // stabilise the power
    CLKCTRL = 0x20; //set system clock to xtal only
    CLKLFCTRL = 0x01;

    // Init Buffer for Interrupt UART testing
    TXBuf[0] = 0xa5; //Sync 0
    TXBuf[1] = 0x5a; //Sync 1
    TXBuf[2] = 2; //Protocol version
    TXBuf[3] = 0; //Packet counter
    P0DIR &= ~0x09;
    P03 = 1;
    P00 = 0;

    //initialize uart
    uart_init(UART_BAUD_57K6);
    uart_puts("\r\nwireless EEGV2.1 Chair\n\r");

    // Initilize RF
    RFCKEN = 1; // enable L01 clock
    RF = 1; // enable RF interrupt
    EA = 1; // Global interrupt enable
    radio_init(0); // Enter PRX mode (Receiver = 0)
    delay_ms(10);
    RadiorXOn();

    //initialize spi
    INTEXP |= 0x02;
    Spiinit();
    SPIMCON1 = 0x0D; // 0000 1100

    // start global interrupt
    EA=1;
    while( 1 ){
        if (send_uart_byte){
            S0BUF= TXBuf[TXIndex];
            send_uart_byte = false; // reset the SPI flag
        }

        if (send_spi_byte){
            if (TXIndex2 ==0)
                SpiOn();
            SPIMDAT=TXBuf[TXIndex2];
            send_spi_byte = false; // reset the SPI flag
        }
    }
}

```

B.3 Firmware of Embedded System

```

/*****
main.cpp
-----
For main Embedded system of Wheelchair Controller (Netburner-MOD5213-
Freescale ColdFire 5213)
*****/
#include "predef.h"
#include <stdio.h>
#include "constants.h"
#include "basicypes.h"
#include "ucos.h"
#include "ucosmcfc.h"

extern "C"
{

DWORD   MainStk[USER_TASK_STK_SIZE] __attribute__( ( aligned( 4 ) ) );

extern void UserMain( void *pd );

typedef void (*func_ptr) (void);
extern func_ptr __CTOR_LIST__[];
extern unsigned char etext;
extern unsigned char copy_start;
extern unsigned char copy_end;

void __do_global_ctors()
{
    unsigned long nptrs = (unsigned long) __CTOR_LIST__[0];
    unsigned i;
    for (i = nptrs; i >= 1; i--) __CTOR_LIST__[i] ();
}

int main()
{
    unsigned char * cpf=&etext;
    unsigned char * cpt=&copy_start;

    while (cpt < &copy_end) {*cpt++=*cpf++;}

    OSISRLevel = 0x2000; //Start with all interrupts enabled.
    UCosSetup();
    //Start the real "Main Task" UserMain
    OSTaskCreate( UserMain,
                 NULL,
                 ( void * ) &MainStk[USER_TASK_STK_SIZE],
                 ( void * ) MainStk,
                 10 );

    UCosBegin();
    return 0;
}
} //extern "C"

```

```

/*****
spi.cpp
-----
SPI functions, ANN, FFT Initialization
*****/
#include "general.h"
#include "predef.h"
#include <stdio.h>
#include <ctype.h>
#include <basicctypes.h>
#include <serialirq.h>
#include <system.h>
#include "constants.h"
#include "ucos.h"
#include <serialupdate.h>
#include "sim5213.h"
#include <cfinter.h>
#include <utils.h>
#include <pins.h>
#include <math.h>
#include "spi.h"
#include "eeprom.h"

//Macros
#define clrPIN4 sim.gpio.clras = ~0x02
#define setPIN4 sim.gpio.setas = 0x02
#define SPI_CRC_DEFAULT 0x53

typedef struct
{
    tSpiSendParm SBuf;           //Data to be sent to Nordic transceiver
    tSpiAddressParm ABuf;       //Address of wheelchair transceiver to be
    sent to Nordic transceiver
    BYTE NetCRC;                //8 bit CRC for data check
}tSpiWrite;

static tSpiWrite sWriteBuf0;
static tSpiWrite sWriteBuf1;
static tSpiWrite * pWriteBuf;
static tSpiReceiveParm sReadBuf;
static BYTE sNewReadCount;

#ifdef _UCOS_H
    OS_MBOX SpiMbox;
#endif

extern "C"
{
    void SetIntc( long func, int vector, int level, int prio );
    void SpiIRQ();
}

```

```

/*****
 * Function to Initialize pins, interrupt IRQ1, SPI variables
 *****/
void SpiInit()
{
    BYTE err;
    pwriteBuf = &swriteBuf0;
    swriteBuf0.SBuf.NetFlags.c = 0;
    swriteBuf0.SBuf.NetInterval = COM_DEFAULT_INTERVAL;
    swriteBuf0.ABuf.NetAddress3 = 3;
    swriteBuf0.ABuf.NetAddress2 = 2;
    swriteBuf0.ABuf.NetAddress1 = 1;
    swriteBuf0.ABuf.NetAddress0 = 0;
    swriteBuf1.ABuf.NetAddress3 = 3;
    swriteBuf1.ABuf.NetAddress2 = 2;
    swriteBuf1.ABuf.NetAddress1 = 1;
    swriteBuf1.ABuf.NetAddress0 = 0;
    swriteBuf0.NetCRC = SPI_CRC_DEFAULT;
    swriteBuf1.NetCRC = 0;
    sReadBuf.LGyro = 0;
    //set up RTOS message box for pointer to Spi received data
    //configure pins not used as GPIO inputs, the default configuration
    sim.gpio.pnqpar &= ~0x03;
    sim.gpio.pnqpar |= ( 1 );
    sim.gpio.pnqpar &= ~0x10;
    sim.gpio.pnqpar |= ( 0 << 4 );
    sim.gpio.ddrnq &= ~0x10;
    sim.gpio.pnqpar &= ~0x80;
    sim.gpio.pnqpar |= ( 0 << 7 );
    sim.gpio.clrnq = ~0x80;
    sim.gpio.ddrnq |= 0x80;
    sim.eport.eppar &= ~0x0004;
    sim.eport.eppar |= 0x0008;
    //set up interrupt vector, level and priority using Netburner SetIntc
    function
    //Note that the level and priority are ignored for IRQ1 since they are
    hardware fixed
    SetIntc((long) &SpiIRQ, 1, 1, 1);
    //clear interrupt flag
    sim.eport.epddr &= ~0x02;
    //enable interrupt
    sim.eport.epier |= 0x02;
    sReadBuf.SPIDatCnt=0x00; // as writepos in moving window averaging
    sReadBuf.DispCloseFlag=true;
    sReadBuf.DispOpenFlag=true;
    sReadBuf.interval = 5;
    sReadBuf.accumulator1 =0;
    sReadBuf.accumulator2 =0;
    sReadBuf.added = 0;
    sReadBuf.FirstFlag=false;
    sReadBuf.SecondFlag=true;
    sReadBuf.MovGap = 256;
    sReadBuf.RawMax =0; // to find raw min max to remove noise
    sReadBuf.RawMin =0; // to find raw min max to remove noise
}

```

```

/*****
 * Function for FFT variable initialization
 *****/
void InitImX(void)
{
    int z;
    for(z=0; z<256; z++){
        sReadBuf.ImX[z] =0;
        sReadBuf.ImX2[z] =0;
    }
}

/*****
 * Function of FFT initialization to calculate for twiddle factor
 *****/
void FFTInit(void)
{
    int Tcnt;

    // lookup table for twiddle factor
    Tcnt=0;
    while( Tcnt < N/2 ){
        sReadBuf.angle = ( M_TWOPi / (double)(N) ) * Tcnt;
        sReadBuf.twiddleFactor[ 2*Tcnt ] = D_TO_F32(-cos( sReadBuf.angle
) ); // -wr.
        sReadBuf.twiddleFactor[ 2*Tcnt+1 ] = D_TO_F32( -sin(
sReadBuf.angle ) ); // -wi.
        Tcnt++;
    }
    for(Tcnt=0; Tcnt<256; Tcnt++){
        sReadBuf.ImX[Tcnt] =0;
        sReadBuf.ImX2[Tcnt] =0;
    }
}

/*****
 * Function for Neural Network parameters initialization
 *****/
void NeuralNetInit(void)
{
    sReadBuf.InputNum = 54;
    sReadBuf.HiddenNeurons = 6;
    sReadBuf.OutputNum =2;
}

/*****
 * Function to write wheelchair tranceiver address via SPI
 *****/
WORD SpiAddress(const tSpiAddressParm * pData)
{
    //write new address to buffers
    sWriteBuf0.ABuf.NetAddress3 = pData->NetAddress3;
    sWriteBuf1.ABuf.NetAddress3 = pData->NetAddress3;
    sWriteBuf0.ABuf.NetAddress2 = pData->NetAddress2;
    sWriteBuf1.ABuf.NetAddress2 = pData->NetAddress2;
    sWriteBuf0.ABuf.NetAddress1 = pData->NetAddress1;
    sWriteBuf1.ABuf.NetAddress1 = pData->NetAddress1;
    sWriteBuf0.ABuf.NetAddress0 = pData->NetAddress0;
    sWriteBuf1.ABuf.NetAddress0 = pData->NetAddress0;
}

```

```

/*****
 * Function to write data to Nordic transceiver via SPI
 *****/
WORD SpiWrite(const tSpiSendParm * pData)
{
    tSpiWrite *pA;
    //find back-up write buffer
    if(pWriteBuf == &swriteBuf0)
    {
        pA = &swriteBuf1;
    }
    else
    {
        pA = &swriteBuf0;
    }
    //load in data to back-up write buffer
    pA->SBuf.NetFlags.c = pData->NetFlags.c;
    pA->SBuf.NetInterval = pData->NetInterval;
    //load CRC. Set to dummy value for now.
    pA->NetCRC = SPI_CRC_DEFAULT;
    //switch to back-up write buffer
    pWriteBuf = pA;
    //return 0: no error
    return 0;
}

/*****
 * Function to write new sample interval time to transceiver via SPI
 *****/
WORD spiWriteInterval(BYTE ticks)
{
    tSpiWrite *pA;
    tSpiWrite *pB;
    //find back-up write buffer
    if(pWriteBuf == &swriteBuf0)
    {
        pA = &swriteBuf1;
    }
    else
    {
        pA = &swriteBuf0;
    }
    //copy data to back-up write buffer
    pA->SBuf.NetFlags.c = pWriteBuf->SBuf.NetFlags.c;
    pA->SBuf.NetInterval = pWriteBuf->SBuf.NetInterval;
    //Change Interval to new value
    pA->SBuf.NetInterval = ticks;
    //load CRC. Set to dummy value for now.
    pA->NetCRC = SPI_CRC_DEFAULT;
    //switch to back-up write buffer
    pWriteBuf = pA;
    //return 0: no error
    return 0;
}

```

```

/*****
 * Interrupt service routine for SPI
 * used to implement software SPI bus
 *****/
INTERRUPT( SpiIRQ, 0x2600)
{
    BYTE a;
    tBitByte b;
    tSpiWrite *pWBuf;
    b.c = 0x55;
    sReadBuf.Syn1      = SpiReadWriteByte(b);
    sReadBuf.Syn2      = SpiReadWriteByte(b);
    sReadBuf.Ver       = SpiReadWriteByte(b);
    sReadBuf.Count     = SpiReadWriteByte(b);
    sReadBuf.EEGCh1Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh1Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.EEGCh2Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh2Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.EEGCh3Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh3Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.EEGCh4Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh4Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.EEGCh5Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh5Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.EEGCh6Byts = ((WORD)SpiReadWriteByte(b))<<8;
    sReadBuf.EEGCh6Byts += (WORD)SpiReadWriteByte(b);
    sReadBuf.Swtch     = SpiReadWriteByte(b);

    FifoIn(sReadBuf.EEGCh1Byts, sReadBuf.EEGCh2Byts);
    sReadBuf.SPIDatCnt++;

    //debug
    setPIN4;
    #ifdef DEBUG1
        writestring(0,"Interrupt\r\n");
    #endif
    //send message containing received data to SpiMbox message box
    #ifdef _UCOS_H
        if (sReadBuf.SecondFlag==true){
            a = OSMBboxPost(&SpiMbox,(void *)&sReadBuf);
            if (a == OS_MBOX_FULL)
                writestring(0,"Interrupt: mail box full\r\n");
        }
    #endif
    #ifdef DEBUG1
        if (a == OS_NO_ERR)
            writestring(0,"Interrupt: message posted\r\n");
    #endif
    #endif
    //wait till pinNORDIC_CS goes high to prevent further interrupts
    // while ((sim.eport.eppdr & 0x02) == 0) ; not needed with edge
    triggered interrupt
    //clear IRQ1 interrupt flag
    sim.eport.epfr |= 0x02;
}

```

```

/*****
 * Function to write/read byte parameter to/from Nordic via SPI
 *****/
#define clrNORDIC_DIN sim.gpio.clrnq = ~0x80
#define setNORDIC_DIN sim.gpio.setnq = 0x80
#define readNORDIC_CLK (sim.gpio.setnq & 0x10)
#define readNORDIC_DOUT (sim.gpio.setas & 0x01)
BYTE SpiReadWriteByte(tBitByte byte_buf)
{
    int i,b;
    tBitByte r_byte,temp_byte;
    //read and write bit with shift (8 times)
    for(i=0;i<8;i++)
    {
        //put high bit of byte_buf on Dout
        if ((byte_buf.c & 0x80) == 0)
        {
            clrNORDIC_DIN;
        }else
        {
            setNORDIC_DIN;
        }
        //shift r_byte left
        r_byte.c = r_byte.c << 1;
        //wait for clock to go low
        do
        {
            b = readNORDIC_CLK;
            //b = pinNORDIC_CLK;
        }
        while(b!=0);
        do
        {
            clrPIN4;
            b = readNORDIC_CLK;
            setPIN4;
            //b = pinNORDIC_CLK;
        }
        while(b==0);
        temp_byte.c = sim.gpio.setqs;
        r_byte.b.bit0 = temp_byte.b.bit3;
        byte_buf.c = byte_buf.c << 1;
    }
    return r_byte.c;
}

/*****
//Function for FIFO buffer initialization
 *****/
void FifoInit(void)
{
    int i,j;
    sReadBuf.ptrin = 0;
    sReadBuf.ptrout = 0;
    for(j=0;j<FIFO_DEPTH;j++)
    {
        for(i=0;i<FIFO_LENGTH;i++)
            sReadBuf.fifo[j][i] = '\0';
    }
}

```

```

//*****
// Function for moving average window
//*****
char MovingAvg(WORD c, WORD d)
{
    WORD *pr1, *pr2, r;
    r=1;
    sReadBuf.accumulator1 += c;
    sReadBuf.accumulator2 += d;
    sReadBuf.added++;
    if (sReadBuf.added > sReadBuf.interval){
        sReadBuf.oldest = (( sReadBuf.ptrin&FIFO_MASK) -
        sReadBuf.interval);
        if (sReadBuf.oldest < 0)
            sReadBuf.oldest +=FIFO_LENGTH;
        ReadBuf.accumulator1 -= sReadBuf.fifo[0][sReadBuf.oldest];
        sReadBuf.accumulator2 -=
        sReadBuf.fifo[1][sReadBuf.oldest];
        sReadBuf.added = sReadBuf.interval;
    }
    if (sReadBuf.added){
        r=0;
        pr1 = &sReadBuf.Avg[0][ sReadBuf.ptrin&FIFO_MASK];
        pr2 = &sReadBuf.Avg[1][ sReadBuf.ptrin&FIFO_MASK];
        *pr1 = sReadBuf.accumulator1 / sReadBuf.added;
        *pr2 = sReadBuf.accumulator2 / sReadBuf.added;
    }
    return r;
}

//*****
//Function to input data into FIFO buffer
//*****
char FifoIn(WORD c, WORD d)
{
    char r;
    r = 1;
    if(((int)(sReadBuf.ptrin-sReadBuf.ptROUT)<FIFO_LENGTH){
        r = 0;
        sReadBuf.fifo[0][sReadBuf.ptrin&FIFO_MASK] = c;
        sReadBuf.fifo[1][sReadBuf.ptrin&FIFO_MASK] = d;
        MovingAvg(c,d);
        sReadBuf.ptrin++;
    }
    return r;
}

//*****
//Function to output data into FIFO buffer
//*****
int FifoOut (void)
{
    int pr;

    if (sReadBuf.ptrin != sReadBuf.ptROUT){
        pr = sReadBuf.ptROUT;
        sReadBuf.ptROUT++;
    }
}

```

```

    else
        pr = sReadBuf.ptrout;
    return pr;
}

//*****
//Function to output packet of data into FIFO buffer
//*****
char FifoOutpacket(void)
{
    BYTE err,i; // 0 = no err; 1=error
    WORD data_out;
    WORD rawmax1, rawmin1;
    err=0;
    for(i=0; i<256; i++){
        data_out = FifoOut();
        if (data_out !=sReadBuf.ptrout){
            sReadBuf.ReX[i]
            =sReadBuf.Avg[0][data_out&FIFO_MASK];
            sReadBuf.ReX2[i]=sReadBuf.Avg[1][data_out&FIFO_MASK];
            if (i ==0){
                if (sReadBuf.ReX[i]    >= sReadBuf.ReX2[i]){
                    sReadBuf.RawMax = sReadBuf.ReX[i];
                    sReadBuf.RawMin = sReadBuf.ReX2[i];
                }
                else{
                    sReadBuf.RawMax = sReadBuf.ReX2[i];
                    sReadBuf.RawMin = sReadBuf.ReX[i];
                }
            }
            else{
                if (sReadBuf.ReX[i]    >= sReadBuf.ReX2[i]){
                    rawmax1 = sReadBuf.ReX[i];
                    rawmin1 = sReadBuf.ReX2[i];

                    if (rawmax1 > sReadBuf.RawMax)
                        sReadBuf.RawMax =rawmax1;
                    if (rawmin1 < sReadBuf.RawMin)
                        sReadBuf.RawMin =rawmin1;
                }
                else{
                    rawmax1 = sReadBuf.ReX2[i];
                    rawmin1 = sReadBuf.ReX[i];
                    if (rawmax1 > sReadBuf.RawMax)
                        sReadBuf.RawMax =rawmax1;
                    if (rawmin1 < sReadBuf.RawMin)
                        sReadBuf.RawMin =rawmin1;
                }
            }
        }
        else{
            err=i;
            sReadBuf.ptrout -=i;
            break;
        }
    }
    return err;
}

```

```

/*****
data.cpp
-----
Data processing and ANN classification
*****/
#include "general.h"
#include "predef.h"
#include <stdio.h>
#include <ctype.h>
#include <basicctypes.h>
#include <serialirq.h>
#include <system.h>
#include <math.h>
#include "constants.h"
#include "ucos.h"
#include <serialupdate.h>
#include "sim5213.h"
#include <cfinter.h>
#include <utils.h>
#include <pins.h>
#include "spi.h"
#include "display.h"
#include "data.h"
#include "calc.h"

//Macros
BYTE cnt;
#define clrPIN4 sim.gpio.clras = ~0x02
#define setPIN4 sim.gpio.setas = 0x02

//Local structures
typedef struct
{
    tSpiSendParm SBuf;           //Data to be sent to Nordic transceiver
    tSpiAddressParm ABuf;
    BYTE NetCRC;                //8 bit CRC for data check
}tData;

//Local variables
namespace
{
    char display_mode;
}
namespace data
{
    tSpiReceiveParm * pData;
}
using namespace data;

```

```

/*****
 * Function for data main processing
 * Task running under real-time operationg system
 *****/
void DataMain(void *psignal)
{
  BYTE err;
  int num1, num2,n,num3, data_out;
  psignal = psignal;
  //indicate on display waiting for head sensor
  Display("\a1waiting for Head");
  Display("\a2");
  display_mode = 1;
  //Task infinite loop
  //debug
  #ifdef DEBUG1
    writestring(0,"Task started\r\n");
  #endif
  while(1)
  {
    //wait for message from spi
    pData=(tSpiReceiveParm*) OSMboxPend(& SpiMbox,
    2*TICKS_PER_SECOND,&err);
    if (pData==NULL)
      { //timed out the 5 seconds: indicate fault
        if (display_mode != 2)
          {
            Display("\a1 Faulty Radio");
            Display("\a2          ");
            display_mode = 2;
          }
      }
    #ifdef DEBUG1
      writestring(0,"message timeout\r\n");
    #endif
    }
    else
      { //We got the message
        //test if head sensor not found
        if (pData->ChairFlags.b.bit0 != 0)
          {
            //head sensor not found (0,1)#
            //indicate not found
            if (display_mode != 3)
              {
                Display("\a1 Head not found");
                display_mode = 3;
              }
            //run CalcMain in no head mode
          }
          else
            {
              //head sensor found #(0,1)
              //indicate head sensor connected
              if (display_mode != 4)
                {
                  Pins[26]=1;
                  Display("\a1 W.EEG connected");
                  Display("\a2L  F  O/F  R");
                  display_mode = 4;
                  pData->DispOpenFlag =true;
                }
            }
          }

```

```

        pData->DispCloseFlag =true;
    }
    if ((pData->SPIDatCnt>=256)&& (pData->FirstFlag==false)){
        pData->SPIDatCnt=0;
        pData->FirstFlag = true;
        pData->SecondFlag = true;
        data_out = FifoOutpacket();
    }
    else if ((pData->SPIDatCnt>=pData->MovGap)&& (pData-
>FirstFlag==true)&&(pData->SecondFlag==true)){
        pData->SPIDatCnt=0;
        pData->FirstFlag = true;
        pData->SecondFlag = false;
        pData->ptrout = pData->ptrout - 256 + pData->MovGap;
// apply moving window
        data_out = FifoOutpacket();
        if (data_out ==0){
            InitImX(); // Imx =0
//FFT
            realFFT( pData->ReX, pData->ImX, pData-
>twiddleFactor, log2_N ); //Call FFT function
            realFFT( pData->ReX2, pData->ImX2, pData-
>twiddleFactor, log2_N ); //Call FFT function
//calculate Abs(FFT)
            for (num2=4; num2<=30;num2++){
                pData->AbsX[num2] = sqrt((pData-
>ReX[num2]*pData->ReX[num2]) +(pData->ImX[num2]*pData->ImX[num2]));
                pData->AbsX[num2] = pData-
>AbsX[num2]*pData->AbsX[num2]; // PSD left
                pData->Fe[num2-4] = (double)(((pData-
>AbsX[num2] - xmin)/ (xmax- xmin))*1.0);
                pData->AbsX2[num2] = sqrt((pData-
>ReX2[num2]*pData->ReX2[num2]) +(pData->ImX2[num2]*pData-
>ImX2[num2]));
                pData->AbsX2[num2] = pData-
>AbsX2[num2]*pData->AbsX2[num2]; // PSD right
                pData->Fe[num2+23] = (double)(((pData-
>AbsX2[num2] - xmin)/ (xmax- xmin))*1.0);
            }
//Neural Networks classification
//Update for the hidden neuron layer
            for(num2=1; num2<=pData-
>HiddenNeurons;num2++){
                pData->NetInput = 0.0;
                for (num3=1; num3<=pData-
>InputNum;num3++){
                    pData->NetInput = (double)(pData-
>NetInput +(weights1[num3-1][num2-1]*pData->Fe[(num3-1)]));
                }
                pData->NetInput = (double)(pData-
>NetInput + (weights1[num3-2+1][num2-1]*(1))); // -1 is Bias
// apply activation function use Taylor series symenrical
approximation for negative value
                if (pData->NetInput < 0){
                    pData->NetInput =(double) pData-
>NetInput* (-1.0);

```

```

        pData->Output[num2-1][0] =
(double) (0.5 - ((1.0/(1.0+( 1.0/( (1.0+pData->NetInput*(1.0+pData-
>NetInput/2.0*(1.0+pData->NetInput/3.0*(1.0+pData-
>NetInput/4.0*(1.0+pData->NetInput/5.0*(1.0+pData-
>NetInput/6.0*(1.0+pData->NetInput/7.0))))))))) - 0.5));
    }
    else
        pData->Output[num2-1][0] =
(double) (1.0/(1.0+( 1.0/( (1.0+pData->NetInput*(1.0+pData-
>NetInput/2.0*(1.0+pData->NetInput/3.0*(1.0+pData-
>NetInput/4.0*(1.0+pData->NetInput/5.0*(1.0+pData-
>NetInput/6.0*(1.0+pData->NetInput/7.0))))))))) );
}

//Update for the output layer
for(num2=1; num2<=pData->OutputNum;num2++){
    pData->NetInput = 0;
    for (num3=1; num3<=pData-
>HiddenNeurons;num3++){
        pData->NetInput = (double)(pData-
>NetInput + (weights2[num3-1][num2-1] * pData->Output[num3-1][0]));
    }
    pData->NetInput = (double)(pData-
>NetInput + (weights2[num3-2+1][num2-1]*(1))); // -1 is Bias
    // apply activation function use Taylor series symmetrical
    // approximation for negative value
    if (pData->NetInput < 0){
        pData->NetInput =(double) pData-
>NetInput* (-1.0);
        pData->Output[num2-1][1] =
(double) (0.5 - ((1.0/(1.0+( 1.0/( (1.0+pData->NetInput*(1.0+pData-
>NetInput/2.0*(1.0+pData->NetInput/3.0*(1.0+pData-
>NetInput/4.0*(1.0+pData->NetInput/5.0*(1.0+pData-
>NetInput/6.0*(1.0+pData->NetInput/7.0))))))))) - 0.5));
    }
    else
        pData->Output[num2-1][1] =
(double) (1.0/(1.0+( 1.0/( (1.0+pData->NetInput*(1.0+pData-
>NetInput/2.0*(1.0+pData->NetInput/3.0*(1.0+pData-
>NetInput/4.0*(1.0+pData->NetInput/5.0*(1.0+pData-
>NetInput/6.0*(1.0+pData->NetInput/7.0))))))))) );
}

// classification
    if(( pData->Output[0][1] > pData-
>Output[1][1]) && ( pData->Output[0][1] > pData->Output[2][1]) && (
pData->Disptask1Flag <1)){
        Display("\a1 R --> right");
        pData->Disptask1Flag++;
        pData->Disptask2Flag = 0;
        pData->Disptask3Flag = 0;
    }
    else if(( pData->Output[1][1] > pData-
>Output[0][1]) && ( pData->Output[1][1] > pData->Output[2][1]) && (
pData->Disptask2Flag <1)){
        Display("\a1 L --> left");
        pData->Disptask2Flag++;
    }
}

```



```

| ImX[j]=ImX[i];
|   move.l    (%a3), (0,%a1,%a4.l*1)
| ReX[i]=tr;
|   move.l    %d3, (%a2)
| ImX[i]=ti;
|   move.l    %d4, (%a3)

jk:
| d7 is k, k=nd2*4
| '4' is the size of one sample (4 bytes);
|   move.l    %d5,%d7

|   cmp.l     %d7,%d2           | while (k<=j){
|   bcs.b     j

while:
|   sub.l     %d7,%d2           | j=j-k;
|   lsr.l     #1,%d7           | k=k>>1;
|   cmp.l     %d7,%d2
|   bcc.b     while

j:
|   add.l     %d7,%d2           | j=j+k;

|   addq.l    #4,%d1           | increment i
|   cmp.l     %d6,%d1
|   bcs.b     for

rts

#####
|# Real FFT
|# void realFFT (FRAC32 * ReX, FRAC32 * ImX);
|#####
|# Upon entry, ReX[ ] contains the real input signal,
|# while values in ImX[ ] are ignored.
|# Upon return, ReX[ ] and ImX[ ] contain the DFT output.
|# All signals run from 0 to 1023.
|#####
.global    realFFT

realFFT:
| store contents of all registers into stack
|   lea.l     (-72,%a7),%a7
|   movem.l   %d0-%d7/%a0-%a6, (%a7)
|   move.l    #0xB0, %MACSR

/* Initialize remaining eMAC registers. */
|   move.l    #0, %ACC0
|   movem.l   (76,%a7),%a0-%a1
|   move.l    (88,%a7),%d1           | d1=log2_N
|   moveq.l   #1,%d0
|   asl.l     %d1,%d0           | d0=N
|   move.l    %d0,%d1           | d1=N (counter)
|   movea.l   %a0,%a2           | a2=dstRe
|   movea.l   %a1,%a3           | a3=dstIm
|   movea.l   %a0,%a4           | a4=srcRe

reorder:
|   move.l    (%a4)+, (%a2)+           | ReX[i]=ReX[2*i];

```

```

        move.l    (%a4)+, (%a3)+          | ImX[i]=ReX[2*i+1];

| modification of loop counter
    subq.l    #2, %d1
    bne      reorder
    move.l    %a1, -(%a7) | push address of ImX[] buffer into the stack
    move.l    %a0, -(%a7) | push address of ReX[] buffer into the stack
    move.l    %d0, -(%a7) | push N into the stack
    jsr      _rev_addr_sort | jump to subroutine
    lea.l    (12, %a7), %a7 | restore stack pointer after subroutine

| Register assignment during first stage of FFT:
| first stage of FFT: start
    move.l    %d0, %d7 | backup: d7=N
    movea.l   %a0, %a2 | backup: a2=ReX
    movea.l   %a1, %a3 | backup: a3=ImX
    move.l    %d0, %d6 | d6=N
    asr.l     #2, %d6 | Counter of sub-DFTs: d6=N/4

first_stage:
    movem.l   (%a0), %d0/%d2 | d0 = ar, d2 = br
    movem.l   (%a1), %d1/%d3 | d1 = ai, d3 = bi

| Normalize all inputs (divide by 2).
    asr.l     #1, %d0 | Normalize ar
    asr.l     #1, %d1 | Normalize ai
    asr.l     #1, %d2 | Normalize br
    asr.l     #1, %d3 | Normalize bi
    move.l    %d2, %d4 | Temporary save br.
    move.l    %d3, %d5 | Temporary save bi.
    add.l     %d0, %d2 | xr = ar + br
    move.l    %d2, (%a0)+
    add.l     %d1, %d3 | xi = ai + bi
    move.l    %d3, (%a1)+
    sub.l     %d4, %d0 | yr = ar - br
    move.l    %d0, (%a0)+
    sub.l     %d5, %d1 | yi = ai - bi
    move.l    %d1, (%a1)+
    subq.l    #1, %d6
    bne      first_stage
| first stage of FFT: end

| second stage of FFT: start
    move.l    %d7, %d0 | from backup: d0=N
    asr.l     #3, %d0 | Counter of sub-DFTs: d0=N/8
                    | Each sub-DFT consists of 2

butterflies.
    movea.l   %d0, %a5 | Counter of sub-DFTs: a5=N/8

    movea.l   %a2, %a0 | from backup: a0=&ReX
    movea.l   %a3, %a1 | from backup: a1=&ImX
second_stage:
    movem.l   (%a0), %d0-%d3 | d0 = ar0, d1 = ar1, d2 = br0,
d3 = br1
    movem.l   (%a1), %d4-%d7 | d4 = ai0, d5 = ai1, d6 = bi0,
d7 = bi1

| Normalize all inputs (divide by 2).

```

```

asr.l    #1,%d0          | Normalize ar0
asr.l    #1,%d1          | Normalize ar1
asr.l    #1,%d2          | Normalize br0
asr.l    #1,%d3          | Normalize br1
asr.l    #1,%d4          | Normalize ai0
asr.l    #1,%d5          | Normalize ai1
asr.l    #1,%d6          | Normalize bi0
asr.l    #1,%d7          | Normalize bi1

| First (even) butterfly of second stage.
movea.l  %d0,%a2        | Temporary save a2 = ar0
movea.l  %d1,%a3        | Temporary save a3 = ar1
add.l    %d2,%d0        | xr0 = ar0 + br0
move.l   %d0,(%a0)+
add.l    %d7,%d1        | xr1 = ar1 + bi1
move.l   %d1,(%a0)+
suba.l   %d2,%a2        | yr0 = ar0 - br0
move.l   %a2,(%a0)+
suba.l   %d7,%a3        | yr1 = ar1 - bi1
move.l   %a3,(%a0)+

| Second (odd) butterfly of second stage.
movea.l  %d4,%a2        | Temporary save a2 = ai0
movea.l  %d5,%a3        | Temporary save a3 = ai1
add.l    %d6,%d4        | xi0 = ai0 + bi0
move.l   %d4,(%a1)+
sub.l    %d3,%d5        | xi1 = ai1 - br1
move.l   %d5,(%a1)+
suba.l   %d6,%a2        | yi0 = ai0 - bi0
move.l   %a2,(%a1)+
adda.l   %d3,%a3        | yi1 = ai1 + br1
move.l   %a3,(%a1)+

subq.l   #1,%a5
cmpa.l   #0,%a5
bne      second_stage
| second stage of FFT: end

| Regular stages 3..(log2_N-1): start
move.l   (88,%a7),%d1   | d1=log2_N
moveq.l  #1,%d0
asl.l    %d1,%d0        | d0=N
move.l   %d0,%d6        | step in the table of twiddle factors
                        | N/4*4=N
                        | (multiplied by 4 because of the size
                        | of coefficients)
asr.l    #4,%d0        | Counter of sub-DFTs: d0=N/16
                        | Each sub-DFT consists of 4 butterflies.
                        | Quantity of sub-DFTs on given stage.
move.l   %d0,(64,%a7)
moveq.l  #3,%d0
move.l   %d0,(68,%a7)
movea.l  #(4*4),%a5    | stage loop counter (starts from 3rd stage)
                        | a5 contains the number of butterflies
                        | per one sub DFT multiplied
                        | by 4 (the size of values)

next_stage:
moveq.l  #0,%d0
move.l   %d0,(60,%a7)
movem.l  (76,%a7),%a0-%a1 | sub DFT loop counter = 0
                        | a0 points to ar0, a1 points to ai0
movea.l  %a0,%a2
movea.l  %a1,%a3

```

```

    adda.l    %a5,%a2          | a2 points to br0
    adda.l    %a5,%a3          | a3 points to bi0

next_subDFT:
    movea.l   (84,%a7),%a4     | a4 points to the first twiddle factor
    move.l    (%a0),%d2        | ar -> d2
    move.l    (%a2),%d4        | br -> d4
    moveq.l   #0,%d7           | d7 contains counter for butterfly loop
                                | in the current sub DFT multiplied
                                | by 4 (the size of values)

| start of butterflies loop of the current sub DFT
next_bf:
| Normalize ar and br inputs (divide by 2).
    asr.l     #1,%d2           | Normalize ar
    asr.l     #1,%d4           | Normalize br
    movem.l   (%a4),%d0-%d1    | wr -> d0, wi -> d1
    move.l    %d2,%ACC0        | ar -> %ACC0
    msac.l   %d0,%d4,(%a3),%d5 | ar-br*wr -> %ACC0, bi -> d5
    asr.l     #1,%d5           | Normalize bi
    msac.l   %d1,%d5,(%a1),%d3 | ar-br*wr-bi*wi = xr -> %ACC3, ai -> d3
    asr.l     #1,%d3           | Normalize ai
    move.l    %ACC0,%a6        |
    move.l    %a6,(%a0)+       | xr -> memory
    add.l     %d2,%d2          | 2*ar -> d2
    sub.l     %a6,%d2          | 2*ar-xr = yr -> d2
    move.l    %d2,(%a2)+       | yr -> memory
    move.l    %d3,%ACC0        | ai -> %ACC3
    mac.l    %d1,%d4,(%a0),%d2 | ai+br*wi -> %ACC3, ar -> d2
    msac.l   %d0,%d5,(%a2),%d4 | ai+br*wi-bi*wr = xi -> %ACC3, br -> d4

| Normalization of above loaded ar and br inputs
| is implemented at start of butterflies loop.
    move.l    %ACC0,%a6        |
    move.l    %a6,(%a1)+       | xi -> memory
    add.l     %d3,%d3          | 2*ai -> d3
    sub.l     %a6,%d3          | 2*ai-xi = yi -> d3
    move.l    %d3,(%a3)+       | yi -> memory
    adda.l    %d6,%a4          | modify pointer to the twiddle factor
                                | for the calculation of the next butterfly

    addq.l    #4,%d7           |
    cmp.l     %a5,%d7          |
    bcs.b     next_bf         |

| end of butterflies loop of the current sub DFT
    adda.l    %a5,%a0          | a0 - a3 point to the input values
    adda.l    %a5,%a1          | for the first butterfly
    adda.l    %a5,%a2          | of the next sub DFT
    adda.l    %a5,%a3          |

    move.l    (60,%a7),%d0     |
    addq.l    #1,%d0           |
    move.l    %d0,(60,%a7)     | increment sub DFT loop counter
    cmp.l     (64,%a7),%d0     | compare sub DFT loop counter with
                                | the number of sub DFTs on this stage
    bcs.w     next_subDFT     | end of sub DFTs loop

    adda.l    %a5,%a5          | multiply contents of a5 (the number of
                                | butterflies per one sub DFT) by 2 for the
                                | next stage
    lsr.l     #1,%d6          | divide step in the table of twiddle

```

```

    move.l   (64,%a7),%d0      | factors by 2
    lsr.l   #1,%d0           | divide the number of sub DFTs for the
    move.l   %d0,(64,%a7)     | next stage by 2
    move.l   (68,%a7),%d0     | increment stage loop counter
    addq.l  #1,%d0
    move.l   %d0,(68,%a7)
    cmp.l   (88,%a7),%d0
    bcs.w   next_stage      | end of stage loop
| Regular stages 3..(log2_N-1): end
| Even/odd frequency domain decomposition: start
    movem.l (76,%a7),%a0-%a1 | point a0 and a1 to ReX and ImX buffers
    lea.l   (4,%a0),%a2      | point a2 to ReX[i], i=1
    lea.l   (4,%a1),%a3      | point a3 to ImX[i], i=1
    move.l   (88,%a7),%d1    | d1=log2_N (temporary)
    moveq.l  #1,%d6
    asl.l   %d1,%d6          | d6=N (temporary)
    movea.l  %d6,%a6         | a6=N (stored value)
    move.l   %d6,%d0         | d0=N (temporary)
    asr.l   #1,%d0          | d0=N/2=nd2 (temporary)
    lea.l   (0,%a0,%d0.1*4),%a4
    lea.l   (0,%a1,%d0.1*4),%a5
    addq.l  #1,%d0          | ip2=i+nd2, i=1
    subq.l  #1,%d6          | ipm=im+nd2=2*nd2-i=N-i, i=1

adjust:
    move.l   (%a3),%d3       | d3 = ImX[i]
    move.l   %d3,%d1        | d1 = ImX[i]
    move.l   -(%a5),%d5     | d5 = ImX[im]
    add.l   %d5,%d3        | d3 = ImX[i] + ImX[im]
    asr.l   #1,%d3         | d3 = (ImX[i] + ImX[im]) / 2
    move.l   %d3,(0,%a0,%d0.1*4) | ReX[ip2] = (ImX[i] + ImX[im]) / 2

| ReX[ipm] = ReX[ip2]
    move.l   %d3,(0,%a0,%d6.1*4) | ReX[ipm] = ReX[ip2]

| ImX[ip2]=(ReX[im]-ReX[i])/2;
    move.l   -(%a4),%d4     | d4 = ReX[im]
    move.l   %d4,%d7        | d7 = ReX[im]
    move.l   (%a2),%d2      | d2 = ReX[i]
    sub.l   %d2,%d4        | d4 = ReX[im] - ReX[i]
    asr.l   #1,%d4         | d4 = (ReX[im] - ReX[i]) / 2
    move.l   %d4,(0,%a1,%d0.1*4) | ImX[ip2]=(ReX[im]-ReX[i])/2

| ImX[ipm] = -ImX[ip2];
    neg.l   %d4            | d4 = -ImX[ip2]
    move.l   %d4,(0,%a1,%d6.1*4) | ImX[ipm] = -ImX[ip2]

| ReX[i]=(ReX[i]+ReX[im])/2;
    add.l   %d7,%d2        | d2 = ReX[i] + ReX[im]
    asr.l   #1,%d2         | d2 = (ReX[i] + ReX[im]) / 2
    move.l   %d2,(%a2)+    | ReX[i] = (ReX[i] + ReX[im]) / 2

| ReX[im] = ReX[i]
    move.l   %d2,(%a4)     | ReX[im] = ReX[i]

| ImX[i]=(ImX[i]-ImX[im])/2;
    sub.l   %d5,%d1        | d1 = ImX[i] - ImX[im]
    asr.l   #1,%d1         | d1 = (ImX[i] - ImX[im]) / 2
    move.l   %d1,(%a3)+    | ImX[i] = (ImX[i] - ImX[im]) / 2

```

```

| ImX[im]=-ImX[i];
  neg.l    %d1
  move.l   %d1, (%a5)
| d1 = -ImX[i]
| ImX[im]=-ImX[i];

| loop processing
  addq.l   #1,%d0
  subq.l   #1,%d6
  cmp.l    %d0,%d6
  bgt.b    adjust
| ip2=i+nd2, after increment of i
| ipm=2*nd2-i, after increment of i
| ipm-ip2 <=> (2*nd2-i)-(i+nd2)

| finish of even/odd frequency domain decomposition
  movea.l  %a1,%a4
  adda.l   %a6,%a4
  movea.l  %a0,%a3
  adda.l   %a6,%a3
  adda.l   %a6,%a3
  moveq.l  #0,%d0
  move.l   (%a1), (%a3)
  adda.l   %a6,%a3
  move.l   (%a4), (%a3)
  move.l   %d0, (%a4)
  adda.l   %a6,%a4
  move.l   %d0, (%a4)
  adda.l   %a6,%a4
  move.l   %d0, (%a4)
  move.l   %d0, (%a1)
| a4=&ImX[0]
| a4=&ImX[n4]
| a3=&ReX[0]
| a3=&ReX[n4]
| a3=&ReX[n4+n4]=&ReX[nd2]
| ReX[nd2]=ImX[0];
| a3=&ReX[nd2+n4]=&ReX[n34]
| ReX[n34]=ImX[n4];
| ImX[n4]=0;
| a4=&ImX[n4+n4]=&ImX[nd2]
| ImX[nd2]=0;
| a4=&ImX[nd2+n4]=&ImX[n34]
| ImX[n34]=0;
| ImX[0]=0;
| Even/odd frequency domain decomposition: end
| Last (log2_N) stage of FFT: start
| The previous log2(N)-1 stages

  movea.l  %a6,%a5
  adda.l   %a6,%a5
  movea.l  (84,%a7), %a4
  movea.l  %a0,%a2
  movea.l  %a1,%a3
  adda.l   %a5,%a2
  adda.l   %a5,%a3
  move.l   (%a0), %d2
  move.l   (%a2), %d4
  moveq.l  #0,%d7
| a5=N=n4*4
| a5=2*N=nd2*4
| a4 points to the first twiddle factor
| a0 points to ar0
| a1 points to ai0
| a2 points to br0
| a3 points to bi0
| ar -> d2
| br -> d4
| counter for butterfly loop

fin_stage:
  movem.l  (%a4), %d0-%d1
  move.l   %d2, %ACC0
  msac.l  %d0,%d4, (%a3), %d5
  msac.l  %d1,%d5, (%a1), %d6
  move.l   %ACC0, %d3
  move.l   %d3, (%a0)+
  add.l   %d2, %d2
  sub.l   %d3, %d2
  move.l   %d2, (%a2)+
  move.l   %d6, %ACC0
  mac.l   %d1,%d4, (%a0), %d2
  msac.l  %d0,%d5, (%a2), %d4
| wr -> d0, wi -> d1
| ar -> %ACC
| ar-br*wr -> %ACC, bi -> d5
| ar-br*wr-bi*wi = xr -> %ACC, ai -> d6
| xr -> memory
| 2*ar -> d2
| 2*ar-xr = yr -> d2
| yr -> memory
| ai -> %ACC0
| ai+br*wi -> %ACC0, ar -> d2
| ai+br*wi-bi*wr = xi -> %ACC0, br -> d4

  move.l   %ACC0, %d3
  move.l   %d3, (%a1)+
  add.l   %d6, %d6
  sub.l   %d3, %d6
  move.l   %d6, (%a3)+
| xi -> memory
| 2*ai -> d6
| 2*ai-xi = yi -> d6
| yi -> memory

```

```
    addq.l    #8,%a4                | modify pointer to the twiddle factor
                                        | for the next butterfly
    addq.l    #4,%d7
    cmp.l     %a5,%d7
    bcs.b     fin_stage            | end of butterfly loop
| Last (log2_N) stage of FFT: end
    movem.l   (%a7),%d0-%d7/%a0-%a6
    lea.l     (+72,%a7),%a7
    rts
|#####
| End section on aligned boundary
    .p2align 2,0
```

B.4 Software of ANN

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ANN_training.m
%% Neural network training program
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
close all
%Changing Parameters
%Position Num

FLAG1=1;
%get network info
info=xlsread('data','InfoData');
PosNum=info(1,1);
HiddenNuerons=15
Bias=info(3,1)
Lrate=info(4,1);
numCYCLES=info(5,1)
ValStep=info(6,1)
BestCycle=info(9,1)

%get data
for a=1:PosNum
    num{a} = xlsread('data', strcat('Pos',int2str(a)));
end

if ValStep==1
    BestValCycle=info(10,1)
    for a=1:PosNum
        numVal{a} = xlsread('data', strcat('Pos',int2str(a),'Val'));
    end
end

%check data consistancy
chksize=size(num{1},1);
chkstate=0;
for a=1:PosNum
    for b=1:size(num{a},2)
        if chksize~=size(num{a},1)
            chkstate=1;
            'inconsistant data'
            stop();
            break;
        end
    end
end
if ValStep==1
    for b=1:size(numVal{a},2)
        if chksize~=size(numVal{a},1)
            chkstate=1;
            'inconsistant data'
            stop();
            break;
        end
    end
end
end
if chkstate==0
    'data consistant'
    OutputNum=PosNum;
end

```

```

    InputNum=chksize;
end

%Create Output identity
%ActualOutput=eye(PosNum,PosNum);
for i=1:PosNum
    for j=1:PosNum
        if i==j
            ActualOutput(i,j)=1;
        else
            ActualOutput(i,j)=0;
        end
    end
end

TryNum=0;

%A LOOP LEARNING
while FLAG1==1
hold on;

%generate weights
%innerlayer

for i=1:HiddenNuerons
    for a=1:InputNum+1
        weights{1}(a,i)=2*randn();
    end
end
%outtterLayer
for i=1:OutputNum
    for a=1:HiddenNuerons+1
        weights{2}(a,i)=2*randn();
    end
end

% TryNum=TryNum+1;
for i=1:numCYCLES
    CycleERROR=0;
    ValCycleERROR=0;

    for b=1:PosNum
        for a=1:size(num{b},2)
            [Output]=UpdateOutput(num{b}(:,a),weights,Bias);
            error=ActualOutput(:,b)-Output(1:OutputNum,2);
            %UPDATEWEIGHT
            [weights,Delta]=Updateweights(weights,num{b}(:,a),
            Output,error,Lrate,Bias);
            for c=1:PosNum
                CycleERROR=CycleERROR+0.5*(error(c,1)^2);
            end
        end
    end
    % plot(i,CycleERROR,'rx-')
    % pause(0.01)
    CycleERROR_array{i,1}=CycleERROR;
    dlmcell('CycleErr.txt',CycleERROR_array);
% CycleERROR
    if ValStep==1
        for b=1:PosNum

```

```

        for a=1:size(numVal{b},2)
            [Output]=UpdateOutput(num{b}(:,a),weights,Bias);

            error=ActualOutput(:,b)-Output(1:OutputNum,2);
            %UPDATEWEIGHTS
            for c=1:PosNum
                ValCycleERROR=ValCycleERROR+0.5*(error(c,1)^2);
            end
        end
    end
    ValCycleERROR_array{i,1} =ValCycleERROR;
    dlmcell('ValCycleErr.txt',ValCycleERROR_array);
    % plot(i,ValCycleERROR,'bx-')
    % hold on;
end

if i==1
    oldCycle= CycleERROR;
    RecCycleERROR_array{i,1} = CycleERROR;
    dlmcell('RecCycleErr.txt',RecCycleERROR_array);
elseif CycleERROR <= oldCycle
    oldCycle=CycleERROR;
    save weights
    RecCycleERROR_array{i,1} = CycleERROR;
    dlmcell('RecCycleErr.txt',RecCycleERROR_array);
else
    RecCycleERROR_array{i,1} = oldCycle;
    dlmcell('RecCycleErr.txt',RecCycleERROR_array);
end

%Printing out
if ValStep==1
    sprintf('CE: %.2f BCE: %.2f VCE: %.2f BVCE: %.2f CYCNUM# %d
Try# %d',CycleERROR,BestCycle,ValCycleERROR,BestValCycle,i,TryNum)
else
    sprintf('CE: %.2f BCE: %.2f CYCNUM# %d Try#
%d',CycleERROR,BestCycle,i,TryNum)
end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% UpdateOutput.m
%% Function to update output layer of ANN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Output]=UpdateOutput(Input,weights,Bias)
    InputNum=size(Input,1);
    HiddenNeurons=size(weights{2},1)-1;
    OutputNum=size(weights{2},2);
    %hidden layers
    for i=1:HiddenNeurons
        NetInput=0;
        for a=1:InputNum
            NetInput=NetInput+weights{1}(a,i)*Input(a);
        end
        NetInput=NetInput+weights{1}(a+1,i)*Bias;
        Output(i,1)=ActivationFunc(NetInput,1,2); %sigmoid logistics
    end
end

```

```

    for i=1:OutputNum
        NetInput=0;
        for a=1:HiddenNeurons
            NetInput=NetInput+Weights{2}(a,i)*Output(a,1);
        end
        NetInput
        a
        NetInput=NetInput+Weights{2}(a+1,i)*Bias;
        Output(i,2)=ActivationFunc(NetInput,1,2);
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% UpdateWeights.m
%% Function to update weight of ANN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Weights,Delta]
=UpdateWeights(Weights,Input,Output,error,Lrate,Bias)
    InputNum=size(Weights{1},1)-1;
    HiddenNeurons=size(Weights{2},1)-1;
    OutputNum=size(Weights{2},2);
    %outta delta value
    for i=1:size(error)
        Delta(i,2)=error(i)*DiffFunc(Output(i,2));
    end

    %inner delta
    for i=1:HiddenNeurons
        Delta(i,1)=0;
        for a=1:OutputNum
            dout=DiffFunc(Output(i,1));
            Delta(i,1)=Delta(i,1)+dout*Delta(a,2)*Weights{2}(i,a);
        end
    end
    %inner weights
    for i=1:HiddenNeurons
        for a=1:InputNum

Weights{1}(a,i)=Weights{1}(a,i)+Lrate*Delta(i,1)*Input(a,1);
        end
        weights{1}(a+1,i)=Weights{1}(a+1,i)+Lrate*Delta(i,1)*Bias;
    end
    %outter weights
    for i=1:OutputNum
        for a=1:HiddenNeurons
Weights{2}(a,i)=Weights{2}(a,i)+Lrate*Delta(i,2)*Output(a,1);
        end
        weights{2}(a+1,i)=Weights{2}(a+1,i)+Lrate*Delta(i,2)*Bias;
    end
end

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ActivationFunc.m
%% Function for activation functions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function output=ActivationFunc(input,response,Mode)
    Threshold=0;
    switch Mode
    case 1
        %bipolar logistics
        output= (1-exp(-input))/(1+exp(-input));
    case 2
        %sigmoid
        output= 1/(1+exp(-input/response));
    case 3
        %TLU
        if input>=Threshold
            output=1;
        else
            output=-1;
        end
    case 4 %polynomial approximation sigmoid function
        if (input < (-5))
            output = 0;
        elseif ((input >=(-5))&&((input <=(-4 )))
            output = 0.01129*input + 0.06248;
        elseif ((input >(-4))&&((input <=(-3 )))
            output = 0.02943*input + 0.13404;
        elseif ((input >(-3))&&((input <=(-2 )))
            output = 0.07177*input + 0.25602;
        elseif ((input >(-2))&&((input <=(-1 )))
            output = 0.14973*input + 0.41285;
        elseif ((input >(-1))&&((input <=(0 )))
            output = 0.23105*input+ 0.49653;
        elseif ((input >(0))&&((input <=(1 )))
            output = 0.23105*input+ 0.50346;
        elseif ((input >(1))&&((input <=(2 )))
            output = 0.14973*input+ 0.58714;
        elseif ((input >(2))&&((input <=(3 )))
            output = 0.07177*input+ 0.74097;
        elseif ((input >(3))&&((input <=(4 )))
            output = 0.02943*input+ 0.86595;
        elseif ((input >(4))&&((input <=(5 )))
            output = 0.01129*input+ 0.93751;
        elseif (input >5)
            output = 1;
        end
    case 5 % taylor series approximatoin of sigmoid function
        output = 1./(1 + (1/ExpTaylor(input)));
    case 6 % taylor series and polynomial approximation
        if (input < (-5))
            output = 0;
        elseif ((input >=(-5))&&((input <=(-4 )))
            output = 0.01129*input + 0.06248;
        elseif ((input >(-4))&&((input <=(-3 )))
            output = 0.02943*input + 0.13404;
        elseif ((input >(-3))&&((input <(-2 )))
            output = 0.07177*input + 0.25602;
        elseif (input >=-2)
            output = 1./(1 + (1/ExpTaylor(input)));
        end
    end
end

```

```

        case 7 %taylor series symmetrical approximation
            if (input <0)
                output = (0.5 - ((1./(1 + (1/ExpTaylor(input*(-1)))))) -
0.5));
            else
                output = 1./(1 + (1/ExpTaylor(input)));
            end
        end
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%NeuralNet_toolbox_training.m
%ANN training using Matlab toolbox
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
data = 'Data_mental_task';

%load features dataset
load (strcat('features',strcat(data, '.mat')))

%distribution dataset 25%combination each mental task
[inputs targets] = dist_data(numVal,204);

% Create a Pattern recognition network
hiddenLayerSize =4;

net=newff(inputs,targets,hiddenLayerSize,{'tansig','tansig'}'trainlm')

% Setup division of data for training, validation and testing
net.divideParam.trainRatio=0.5;
net.divideParam.valRatio=0.5;
net.divideParam.testRatio=0.5;

% Override preprocessing and postprocessing
net.inputs{1}.processFcns={'mapminmax'};
net.outputs{2}.processFcns={'mapminmax'};

% Using LevenbergM as training function
net.trainFcn = 'trainlm';
net.trainParam.show = 50;
net.trainParam.epochs = 200;
net.trainParam.goal = 1e-7;
net.trainParam.mu_max = 1e50;
net.trainParam.min_grad = 1e-15;

randn('seed',192736547);
net = init(net);

net.plotFcns =
{'plotperform','plottrainstate','plotconfusion','plotroc'};

% Train neural network
[net,tr] = train(net,inputs,targets);

outputs = sim(net,inputs);

%view(net)

```

```
% Save weights
IW=net.IW{1,1};
LW=net.LW{2,1};
b1=net.b{1};
b2=net.b{2};

% plotperform(tr)
% plotperf(tr)
% plotperf(tr,NaN,' ',tr.best_epoch);
% plotconfusion(targets,outputs)

% indices and performance data
info=[];
info.train.indices = tr.trainInd;
info.validation.indices = tr.valInd;
info.test.indices = tr.testInd;
i1 = info.train.indices;
t1 = targets(:,i1);
y1 = outputs(:,i1);
i2 = info.validation.indices;
t2 = targets(:,i2);
y2 = outputs(:,i2);
i3 = info.test.indices;
t3 = targets(:,i3);
y3 = outputs(:,i3);
t4 = [t1 t2 t3];
y4 = [y1 y2 y3];
% plotconfusion(targets,outputs);

plotconfusion(t1,y1, 'Training', t2,y2, 'Validation', t3,y3, 'Test', t4,y4, '
All');

% SaveTrainedWeights(net,IW,b1,LW,b2);
save weight.mat

% Testing the test data
testX = inputs(:,tr.testInd);
testT = targets(:,tr.testInd);
testY = net(testX);
testIndices = vec2ind(testY)
plotconfusion(testT,testY)
[c,cm] = confusion(testT,testY)
fprintf('Percentage Correct Classification : %f%%\n', 100*(1-c));
fprintf('Percentage Incorrect Classification : %f%%\n', 100*c);
```

B.5 Software of GA-ANN

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% GA_ANN_training.m
%% Genetic Algorithm of Artificial Neural NETWORK (GA-ANN) training
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
clc;

load combi3b.mat % for 3 task combination

Subject = 11;
global xx desire no_in no_out N no_iters
hidden_list = [3 4 5 6 7 8 9 10 12 15 20 25 30];

num_repeat = 10; %total number of repeat

ch = 'C3C4P3P4O1O2';
i=1; %first combination (Aritmetic, Rubik's Cube and letter composing)

ntrain=210;
for fet=1:4
    if fet==1
        fetn='HHT1';
        no_in = 180; %number of input
    elseif fet==2
        fetn='HHT2';
        no_in = 180; %number of input
    elseif fet==3
        fetn='FFT';
        no_in = 180; %number of input
    elseif fet==4
        fetn='Wave';
        no_in = 24; %number of input
    end

    for fol=1:3
        for j =1:length(hidden_list)
            clc;
            for k =1 :num_repeat
                clear inputs num_val targets task_sel resultx
                trace endPop bPop t
                numfile = ((combi3b(i,1)*100) + (combi3b(i,2)*10)
                + (combi3b(i,3)*1))
                eval(['load S' num2str(Subject) '_features_' fetn
                '_6CH_' num2str(numfile) '.mat;']);
                %create dataset for 3-fold cross validation
                inputs1= (inputs(1:no_in,1:ntrain));
                target1= (targets(:,1:ntrain));
                inputs2= (inputs(1:no_in,ntrain+1:ntrain*2));
                target2= (targets(:,ntrain+1:ntrain*2));
                inputs3= (inputs(1:no_in,(ntrain*2)+1:ntrain*3));
                target3= (targets(:,(ntrain*2)+1:ntrain*3));

                inputs11 = horzcat(inputs1, inputs2);
                target11 = horzcat(target1, target2);

                inputs22 = horzcat(inputs1, inputs3);
                target22 = horzcat(target1, target3);
            end
        end
    end
end

```

```

inputs33 = horzcat(inputs2, inputs3);
target33 = horzcat(target2, target3);

if fo1 ==1
%kfo1=1
    xx = (inputs11)';
    desire = (target11)';

elseif fo1==2
%kfo1=2
    xx = (inputs22)';
    desire = (target22)';

elseif fo1==3
%kfo1=3
    xx = (inputs33)';
    desire = (target33)';
end

no_out=3; %number of output
N=hidden_list(1,j) %hidden neuron
no_iters=2000; %number of training iteration
rep=k
% run GA-ANN training
[resultx,trace]=GA_optimization('200',1,1);
eval(['save S' num2str(Subject) '_weights_GA_'
fetn '_6CH_' num2str(numfile) '_KFOL' num2str(fo1) '_H'
num2str(hidden_list(j)) '_' num2str(k) '.mat'];]); % save file
end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% GA_optimization.m
%% Genetic Aloritm Optimization for ANN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [resultx,trace]=GA_optimization(fun_select, kk, pp)

display1=1; %0=off
no_sample=pp;
global no_iters

% ----- BG Problem -----
elseif strcmp(fun_select, '200')==1
    global no_in no_out N no_iters
    pb=4;
    vlb = [ones(1, (no_in+1)*N)*(-pb) ones(1, (N+1)*no_out)*(-pb)];%FFNN
    vub = [ones(1, (no_in+1)*N)*(pb) ones(1, (N+1)*no_out)*(pb)];
    bounds = [vlb' vub'];
    initvar= bounds;
    pop_size=50;
    pc=0.8;
    pm=0.1;
    mu_b=2;
    fun='f200';
end

```

```

%-----Method Selection-----

for j=1:no_sample
    disp(j);
    initPop=initializega(pop_size,initvar,fun);

    switch(kk)
    case(1)
        [bstX,endPop,bPop,trace]=ga1(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['BlendXover']
        ,[1 0.5], 'nonUnifMutation',[1 no_iters mu_b],pc,pm);
    case(2)
        [bstX,endPop,bPop,trace]=ga2(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['AveXover
        BoundXover'],[1 0.5;1 0.5], 'nonUnifMutation1',[1 no_iters
        mu_b],pc,pm);
    case(3)
        [bstX,endPop,bPop,trace]=ga2(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['UNDXXover
        UNDXXover'],[1 0.336 ;1 .336], 'nonUnifMutation',[1 no_iters
        mu_b],pc,pm);
    case(4)
        [bstX,endPop,bPop,trace]=ga2(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['EIXover
        EIXover'],[1 0.366 ;1 0], 'nonUnifMutation',[1 no_iters mu_b],pc,pm);
    case(5)
        [bstX,endPop,bPop,trace]=ga1(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['FractalXover
        '],[1 Npara], 'nonUnifMutation',[1 no_iters mu_b],pc,pm);
    case(6)
        [bstX,endPop,bPop,trace]=ga2(bounds,fun,[],initPop,[1e-6 1
        display1], 'maxGenTerm',no_iters,'normGeomSelect',[0.08],['BlendXover']
        ,[1 0.5], 'nonUnifMutation1',[1 no_iters mu_b],pc,pm);

    end

%----- Stat results -----
h=zeros(1,no_iters);
h(trace(:,1))=trace(:,2);

for i=1:(no_iters-1)
    if h(i+1)<=h(i)
        h(i+1)=h(i);
    end
end
pa_h(j,:)=h;
resultx(j,:)=bstX;
end

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ga1.m
%% Genetic Algorithm optimization function
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,endPop,bPop,traceInfo] =
ga1(bounds,evalFN,evalOps,startPop,opts,...
termFN,termOps,selectFN,selectOps,xOverFNs,xOverOps,mutFNs,mutOps,prob
_xover,prob_mut)

evalstr = [evalFN,'(x,0)'];
n=nargin;
if n<2 | n==6 | n==10 | n==12
    disp('Insufficient arguements')
end
if n<3 %Default evaluation opts.
    evalOps=[];
end
if n<5
    opts = [1e-6 1 0];
end
if isempty(opts)
    opts = [1e-6 1 0];
end

if any(evalFN<48) %Not using a .m file
    if opts(2)==1 %Float ga
        e1str=['x=c1; c1(xZomeLength)=' , evalFN ';;'];
        e2str=['x=c2; c2(xZomeLength)=' , evalFN ';;'];
    else %Binary ga
        e1str=['x=b2f(endPop(j,:),bounds,bits);
endPop(j,xZomeLength)=' ,...
evalFN ';;'];
    end
else %Are using a .m file
    if opts(2)==1 %Float ga
        e1str=['[c1 c1(xZomeLength)]=' evalFN '(c1,[gen evalOps]);'];
        e2str=['[c2 c2(xZomeLength)]=' evalFN '(c2,[gen evalOps]);'];
    else %Binary ga
        e1str=['x=b2f(endPop(j,:),bounds,bits);[x v]=' evalFN ...
'(x,[gen evalOps]); endPop(j,:)=f2b(x,bounds,bits) v;'];
    end
end

if n<6 %Default termination information
    termOps=[100];
    termFN='maxGenTerm';
end
if n<12 %Default mutation information
    if opts(2)==1 %Float GA
        mutFNs=['boundaryMutation multiNonUnifMutation nonUnifMutation
unifMutation'];
        mutOps=[4 0 0;6 termOps(1) 3;4 termOps(1) 3;4 0 0];
    else %Binary GA
        mutFNs=['binaryMutation'];
        mutOps=[0.05];
    end
end
if n<10 %Default crossover information
    if opts(2)==1 %Float GA

```

```

    xOverFNS=['arithXover heuristicXover simplexover'];
    xOverOps=[2 0;2 3;2 0];
    else %Binary GA
    xOverFNS=['simplexover'];
    xOverOps=[0.6];
end
end
if n<9 %Default select opts only i.e. roulette wheel.
selectOps=[];
end
if n<8 %Default select info
selectFN=['normGeomSelect'];
selectOps=[0.08];
end
if n<6 %Default termination information
termOps=[100];
termFN='maxGenTerm';
end
if n<4 %No starting population passed given
startPop=[];
end
if isempty(startPop) %Generate a population at random
%startPop=zeros(80,size(bounds,1)+1);
startPop=initializega(80,bounds,evalFN,evalOps,opts(1:2));
end

if opts(2)==0 %binary
bits=calcbits(bounds,opts(1));
end

xOverFNS=parse(xOverFNS);
mutFNS=parse(mutFNS);

xZomeLength = size(startPop,2);
numVar      = xZomeLength-1;           %Number of variables
popSize     = size(startPop,1);       %Number of individuals in the pop
endPop      = zeros(popSize,xZomeLength);
c1          = zeros(1,xZomeLength);
c2          = zeros(1,xZomeLength);
numXovers   = size(xOverFNS,1);       %Number of Crossover operators
numMut      = size(mutFNS,1);         %Number of Mutation operators
epsilon     = opts(1);                 %Threshold for two fitness to differ
oval        = max(startPop(:,xZomeLength)); %Best value in start pop
bFoundIn    = 1;                       %Number of times best has changed
done        = 0;                       %Done with simulated evolution
gen         = 1;                       %Current Generation Number
collectTrace = (nargout>3);            %Should we collect info every gen
floatGA     = opts(2)==1;              %Probabilistic application of ops
display     = opts(3);                 %Display progress
a=1;
bval_prev=0;
if display
disp(' fitness      gen/T')
disp('-----')
end
while(~done)

[bval,bindx] = max(startPop(:,xZomeLength)); %Best of current pop
best = startPop(bindx,:);

```

```

if collectTrace
    traceInfo(gen,1)=gen; %current generation
    traceInfo(gen,2)=startPop(bindx,xZomeLength); %Best fitness
    traceInfo(gen,3)=mean(startPop(:,xZomeLength)); %Avg fitness
    traceInfo(gen,4)=std(startPop(:,xZomeLength));
end

if ( (abs(bval - oval)>epsilon) | (gen==1))
    if display
        fprintf(1,'\n %f %d/%d %f ',bval,gen,termOps,best(numVar));
%Update the display
    end
    if floatGA
        bPop(bFoundIn,:)= [gen startPop(bindx,:)]; %Update bPop Matrix
    else
        bPop(bFoundIn,:)= [gen
b2f(startPop(bindx,1:numVar),bounds,bits)...
startPop(bindx,xZomeLength)];
    end
    bFoundIn=bFoundIn+1; %Update number of changes
    oval=bval; %Update the best val

end

endPop = feval(selectFN,startPop,[gen selectOps]); %Select
count=0;
if floatGA
for hh=1:popSize
if rand<=(prob_xover/2) %prob crossover
    %ctemp=[];
    for i=1:numXovers,
        for j=1:xOverOps(i,1),
            a = round(rand*(popSize-1)+1); %Pick a parent
            b = round(rand*(popSize-1)+1); %Pick another parent
            c = round(rand*(popSize-1)+1); %Pick another parent2
            xN=deblank(xOverFNs(i,:)); %Get the name of crossover function
            [c1 c2] = feval(xN,endPop(a,:),endPop(b,:),endPop(c,:),bounds,[gen
xoverOps(i,:)]);

            if c1(1:numVar)==endPop(a,(1:numVar)) %Make sure we created a new
                c1(xZomeLength)=endPop(a,xZomeLength);
            elseif c1(1:numVar)==endPop(b,(1:numVar))
                c1(xZomeLength)=endPop(b,xZomeLength);
            else
                %[c1(xZomeLength) c1] = feval(evalFN,c1,[gen evalOps]);
                eval(e1str);
            end
            if c2(1:numVar)==endPop(a,(1:numVar))
                c2(xZomeLength)=endPop(a,xZomeLength);
            elseif c2(1:numVar)==endPop(b,(1:numVar))
                c2(xZomeLength)=endPop(b,xZomeLength);
            else
                eval(e2str);
            end
            endPop(a,:)=c1;
            endPop(b,:)=c2;
        end
    end
end
end
end

```

```

for gg=1:numVar
    if rand<=prob_mut %prob mutation
        for i=1:numMuts,
            for j=1:mutOps(i,1),
                a = round(rand*(popSize-1)+1);
                c1 = feval(deblank(mutFNS(i,:)),endPop(a,:),bounds,[gen
mutOps(i,:),evalstr]);

                if c1(1:numVar)==endPop(a,(1:numVar))
                    c1(xZomeLength)=endPop(a,xZomeLength);
                else
                    eval(e1str);
                end
                endPop(a,:)=c1;
            end
        end
    end
end

else
    for i=1:numXOvers,
        xN=deblank(xOverFNS(i,:)); %Get the name of crossover function
        cp=find(rand(popSize,1)<xOverOps(i,1)==1);
        if rem(size(cp,1),2) cp=cp(1:(size(cp,1)-1)); end
        cp=reshape(cp,size(cp,1)/2,2);
        for j=1:size(cp,1)
            a=cp(j,1); b=cp(j,2);
            [endPop(a,:) endPop(b,:)] = feval(xN,endPop(a,:),endPop(b,:),...
bounds,[gen xoverOps(i,:)]);
        end
    end
    for i=1:numMuts
        mN=deblank(mutFNS(i,:));
        for j=1:popSize
            endPop(j,:) = feval(mN,endPop(j,:),bounds,[gen mutOps(i,:)]);
        end
    end
end

gen=gen+1;
done=feval(termFN,[gen termOps],bPop,endPop); %See if the ga is done
startPop=endPop; %Swap the populations

[bval,bindx] = min(startPop(:,xZomeLength)); %Keep the best solution
startPop(bindx,:) = best; %replace it with the worst
end

[bval,bindx] = max(startPop(:,xZomeLength));

if display
    fprintf(1,'\n%d %F\n',gen,bval);
end

x=startPop(bindx,:);
if opts(2)==0 %binary

```

```

x=b2f(x,bounds,bits);
bPop(bFoundIn,:)=[gen b2f(startPop(bindx,1:numVar),bounds,bits)...
startPop(bindx,xZomeLength)];
else
bPop(bFoundIn,:)=[gen startPop(bindx,:)];
end

if collectTrace
traceInfo(gen,1)=gen; %current generation
traceInfo(gen,2)=startPop(bindx,xZomeLength); %Best fitness
traceInfo(gen,3)=mean(startPop(:,xZomeLength)); %Avg fitness
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% f200.m
%% mapping the data and link to the ANN
%% Genetic Algorithm Optimization for ANN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,f]=f200(x,parameters)
global xx desire no_in no_out N
[pat,row]=size(xx);
[yy]=nnet_trad(x,N,no_in,no_out,xx','logsig','logsig');
f=-mse(desire-yy);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% nnet_trad.m
%% Artificial Neural Network
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [y]=nnet_trad(var,N,no_in,no_out,ins, hid_tf,op_tf)

[row,no_pat]=size(ins);
xin=[ins;ones(1,no_pat)*-1];
[no_in,a]=size(xin);
v=reshape(var(1:N*no_in),N,no_in)';
ow=reshape(var(N*no_in+1:N*no_in+(N+1)*no_out),N+1,no_out)';
hid=ttf(xin'*v,hid_tf);
[r c]=size(hid);
y=[hid -1*ones(r,1)]*ow';
y=logsig(y);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% ttf.m
%% Transfer functions for Artificial Neural Network
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function outtf=ttf(tf_in,index)

if strcmp(index,'logsig')==1
outtf=logsig(tf_in);
elseif strcmp(index,'tansig')==1
outtf=tansig(tf_in);
elseif strcmp(index,'purelin')==1
outtf=(tf_in);
end

```

B.6 Software of FPSOCM-ANN

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% FPSOCM_ANN_training.m
%% fuzzy particle swarm optimization with cross mutated-based
%% artificial neural network (FPSOCM-ANN) training
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
clc;
load combi3b.mat % for 3 task combination
Subject = 11;
global xx desire no_in no_out N no_iter
hidden_list = [10];
num_repeat = 10; %total number of repeat

ch = 'C3C4P3P4O1O2';
i=1; %1'st combination(arithmetic, Rubik's cube and letter composing)
ntrain=210;

for fet=1:4
    if fet==1
        fetn='HHT1';
        no_in = 180; %number of input
    elseif fet==2
        fetn='HHT2';
        no_in = 180; %number of input
    elseif fet==3
        fetn='FFT';
        no_in = 180; %number of input
    elseif fet==4
        fetn='wave';
        no_in = 24; %number of input
    end

    for fol=1:3
        for j =1:length(hidden_list)
            clc;
            for k =1 :num_repeat
                clear inputs num_val targets task_sel resultx
                trace endPop bPop t
                numfile = ((combi3b(i,1)*100) + (combi3b(i,2)*10)
                + (combi3b(i,3)*1))
                eval(['load S' num2str(Subject) '_features_' fetn
                '_6CH_' num2str(numfile) '.mat;']);
                %create dataset for 3-fold cross validation
                inputs1= (inputs(1:no_in,1:ntrain));
                target1= (targets(:,1:ntrain));
                inputs2= (inputs(1:no_in,ntrain+1:ntrain*2));
                target2= (targets(:,ntrain+1:ntrain*2));
                inputs3= (inputs(1:no_in,(ntrain*2)+1:ntrain*3));
                target3= (targets(:,(ntrain*2)+1:ntrain*3));

                inputs11 = horzcat(inputs1, inputs2);
                target11 = horzcat(target1, target2);

                inputs22 = horzcat(inputs1, inputs3);
                target22 = horzcat(target1, target3);

                inputs33 = horzcat(inputs2, inputs3);
                target33 = horzcat(target2, target3);
            end
        end
    end
end

```

```

        if fo1 ==1
            %kfo1=1
            xx = (inputs11)';
            desire = (target11)';

        elseif fo1==2
            %kfo1=2
            xx = (inputs22)';
            desire = (target22)';

        elseif fo1==3
            %kfo1=3
            xx = (inputs33)';
            desire = (target33)';
        end

        no_out=3; %number of output
        N=hidden_list(1,j) %hidden neuron
        no_iter=2000; %number of training iteration
        rep=k

    %run FPSOCM-ANN training
    [t,resultx,endPop,bPop,trace]=pso_main('200',1, 0.005);
        eval(['save S' num2str(Subject) '_weights_FPSOCM_'
fetn '_6CH_' num2str(numfile) '_KFOL' num2str(fo1) '_H'
num2str(hidden_list(j)) '_' num2str(k) '.mat'];]); % save file
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% pso_main.m
%% initialization variable FPSOCM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function
[t,x,endPop,bPop,traceInfo]=pso_main(fun_select,method,shape_m)
global no_iter
if strcmp(fun_select,'200')==1
c1 = 2.05; % PSO parameter C1
c2 = 2.05; % PSO parameter C2
k1=shape_m; %wavelet mutation shape parameter
pm=0.1; %prob of mutation operation
vmax=0.2; %max value of velocity of PSO
dt=0.00001; % parameter for pso_noel ONLY
psopts=[c1,c2,vmax,k1,pm,dt];
dim=10;
pop_size=50;

global no_in no_out N no_iter

pb=5;
vlb = [ones(1,(no_in+1)*N)*(-pb) ones(1,(N+1)*no_out)*(-pb)];%FFNN
vub = [ones(1,(no_in+1)*N)*(pb) ones(1,(N+1)*no_out)*(pb)];
bounds = [vlb' vub'];
startPop=initialize(pop_size,bounds);
fun='fun200';
end

```

```

if method==1
    tic;
    [x,endPop,bPop,traceInfo] =
    pso_ling(bounds,fun,[],startPop,psopts,'maxGenTerm',no_iter);
    t=toc;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% pso_ling.m
%% FPSOCM Optimisation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x,endPop,bPop,traceInfo] =
pso_ling(bounds,evalFN,evalOps,startPop,opts,termFN,termOps)
n=nargin;
if n<2
    disp('Insufficient arguments')
end
if n<3    %Default evaluation opts.
    evalOps=[];
end

if n<4 %No starting population passed given
    startPop=[];
end
if length(startPop)==0 %Generate a population at random
    startPop=initialize(30,bounds);
end

if n<5
    opts = [];
end
if length(opts)==0
    opts = [2.05, 2.05 0.2 0.001];
end

%parameters for pso
c1=opts(1);c2=opts(2);vmax=opts(3);k1=opts(4);
c=c1+c2;
kk=2/abs(2-c-sqrt(c^2-4*c)); %kk=0.7298

if n<6 %Default termination information
    termFN=[];
end
if length(termFN)==0
    termFN='maxGenTerm';
end

if n<7
    termOps=[];
end
if length(termOps)==0
    termOps=50;
end
%%determine the vbounds- a matrix of upper and lower
%%bounds of the velocities
vbounds=vmax*[(bounds(:,1)-bounds(:,2)),(bounds(:,2)-bounds(:,1))];

```

```

evalstr=['[endPop(ip,numVar+1) endPop(ip,numVar+2)]= ' evalFN
'(endPop(ip,:),[evalOps]);'];

xZomeLength = size(startPop,2);    %Length of the
xzome=numVars+fitness
numVar      = xZomeLength-3;      %Number of variables
popSize     = size(startPop,1);   %Number of individuals in the pop
endPop      = startPop;          %A secondary population matrix

for ip=1:popSize
    eval(evalstr);
end

pBest = endPop;
oval  = 0;

bFoundIn = 1;    %Number of times best has changed
done      = 0;    %Done with simulated evolution
gen       = 1;    %Current Generation Number
collectTrace = (nargout>3);    %Should we collect info every gen

minbounds=ones(popSize,1)*bounds(:,1)';
maxbounds=ones(popSize,1)*bounds(:,2)';
minvbounds=ones(popSize,1)*vbounds(:,1)';
maxvbounds=ones(popSize,1)*vbounds(:,2)';

%%--- initilize the velocities
%% v1 with popSize rows, numVar columns
ve=vmax*(-1+2*rand(popSize,numVar)).*(maxbounds-minbounds);

while(1)
    %disp(['gen =',int2str(gen)]);
    [bval,bindx] = min(endPop(:,numVar+1));
    % [bval] = (endPop(bindx,numVar+1));
    fprintf(1,'\n %10.15f %d ',bval,gen);
    gBest = endPop(bindx,:);
    mgBest=ones(popSize,1)*gBest;

    for ipop=1:popSize    %update pBest
        if ( (endPop(ipop,numVar+1) < pBest(ipop,numVar+1)) | (gen==1) )
            pBest(ipop,:)=endPop(ipop,:);
        end
    end

    if ( (bval <= oval) | (gen==1))
        bPop(bFoundIn,:)= [gen endPop(bindx,:)];    %Update bPop Matrix
        bFoundIn=bFoundIn+1;    %Update number of
changes
        oval=bval;    %Update the best val
    end

    if collectTrace
        traceInfo(gen,1)=gen; %current generation
        traceInfo(gen,2:xZomeLength+1)=endPop(bindx,:); %Best fitness
        traceInfo(gen,xZomeLength+2)=mean(endPop(:,numVar+1));%Avgfitness
        traceInfo(gen,xZomeLength+3)=std(endPop(:,numVar+1));
    end

    done=feval(termFN,[gen termOps],bPop,endPop);
    if done==1

```

```

        break;
    end
    dfdx=std((endPop(:,numVar+1))/norm(endPop(:,numVar+1)));
    mdfdx=[zmf(dfdx, [0.03 0.1]) gaussmf(dfdx,[0.035 0.1]) smf(dfdx, [0.1
    0.17])];
    tg=gen/termOps;
    mtf=[zmf(tg, [0.15 0.5]) gaussmf(tg,[0.25 0.5]) smf(tg, [0.5 0.85])];
    m=[mdfdx*mtf(1) mdfdx*mtf(2) mdfdx*mtf(3)];
    omf=[1.1 0.85 1.1 0.6 0.6 0.85 0.1 0.35 0.35]'; % most updated
    cmf=[0.5 0.4 0.5 0.4 0.3 0.5 0.1 0.1 0.2]';
    ww=((m*omf)/sum(m));
    c=((m*cmf)/sum(m));
    ve=kk*(ww*ve+c1*rand(popSize,numVar).*(pBest(:,1:numVar)-
    endPop(:,1:numVar))+c2*rand(popSize,numVar).*(mgBest(:,1:numVar)-
    endPop(:,1:numVar)));

    if rand>0.5
        ve1=((rand(popSize,numVar).*(maxbounds-minbounds))+minbounds)*0.25;
    else
        ve1=(maxbounds-((rand(popSize,numVar).*(maxbounds-
    minbounds))))*0.25;
    end

    pcm_choice=0;
    if (pcm_choice==1)

    b_std=std((endPop(:,numVar+1)));
    b_mean=mean((endPop(:,numVar+1)));

    f_std(gen)=b_std;
    f_mean(gen)=b_mean;
    gg=[0.5 0.1 0.05 0.02 0.01];

    window=20;
    if gen<window+1
        r=ceil(rand*5);
        pcm=gg(r);
    else
        ttest=(f_mean(gen-window)-
    f_mean(gen))/sqrt(f_std(gen)^2/popSize+f_std(gen-window)^2/popSize);

        if ttest<0.5
            m=0.5;
        elseif ttest<1
            m=0.5;
        elseif ttest<1.5
            m=1;
        elseif ttest<2
            m=0.5;
        else
            m=0.2;
        end
        gg=[0.5 0.1 0.05 0.02 0.01]*(1-tg)^m;
        pcm=gg(r);
    end
    %pause
    else
    pcm=k1;
    end
    for u=1:(popSize)

```

```

        for z=1:numVar
            if rand<pcm
                if rand<0.5
                    ve(u,z)=ve(u,z)*(1-c)-ve1(u,z)*(c) ;
                else
                    ve(u,z)=ve(u,z)*(1-c)+ve1(u,z)*(c);
                end
            end
        end

        change1 = ve > maxvbounds;
        ve      = ve.*(~change1)+maxvbounds.*change1;
        change2 = ve < minvbounds;
        ve      = ve.*(~change2)+minvbounds.*change2;

        endPop(:,1:numVar)=endPop(:,1:numVar)+ve;

        change1 = endPop(:,1:numVar) > maxbounds;
        endPop(:,1:numVar) =
        endPop(:,1:numVar).*(~change1)+maxbounds.*change1;
        change2 = endPop(:,1:numVar) < minbounds;
        endPop(:,1:numVar) =
        endPop(:,1:numVar).*(~change2)+minbounds.*change2;
        gen=gen+1;
        endPop=regularVar(endPop,bounds);
        for ip=1:popSize
            eval(evalstr);
        end

    end % while end
    [bval,bindx] = min(endPop(:,numVar+1));
    if ( (bval < oval) )
        bPop(bFoundIn,:)= [gen endPop(bindx,:)]; %Update bPop Matrix
        bFoundIn=bFoundIn+1;
    end
    x=bPop(bFoundIn-1,2:numVar+4);

    if collectTrace
        traceInfo(gen,1)=gen; %current generation
        traceInfo(gen,2:xZomeLength+1)=endPop(bindx,:); %Best fitness
        traceInfo(gen,xZomeLength+2)=mean(endPop(:,numVar+1)); %Avg fitness
        traceInfo(gen,xZomeLength+3)=std(endPop(:,numVar+1));
    end
end

```

Bibliography

- Al-Fahoum, A.S. & Al-Fraihat, A.A. 2014, 'Methods of EEG Signal Features Extraction Using Linear Analysis in Frequency and Time-Frequency Domains', *International Scholarly Research Notices*, vol. 2014.
- Allison, B., Brunner, C., Kaiser, V., Müller-Putz, G., Neuper, C. & Pfurtscheller, G. 2010a, 'Toward a hybrid brain–computer interface based on imagined movement and visual attention', *Journal of Neural engineering*, vol. 7, no. 2, p. 026007.
- Allison, B., Dunne, S., Leeb, R., Millán, J.D.R. & Nijholt, A. 2012, *Towards Practical Brain-Computer Interfaces*, Springer.
- Allison, B., Jin, J., Zhang, Y. & Wang, X. 2014, 'A four-choice hybrid P300/SSVEP BCI for improved accuracy', *Brain-Computer Interfaces*, no. ahead-of-print, pp. 1-10.
- Allison, B., Luth, T., Valbuena, D., Teymourian, A., Volosyak, I. & Graser, A. 2010b, 'BCI Demographics: How Many (and What Kinds of) People Can Use an SSVEP BCI?', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 18, no. 2, pp. 107-116.
- Allison, B. & Neuper, C. 2010, 'Could Anyone Use a BCI?', in D.S. Tan & A. Nijholt (eds), *Brain-Computer Interfaces, Applying our Minds to Human-Computer Interaction*, Springer London, pp. 35-54.
- Allison, B.Z., McFarland, D.J., Schalk, G., Zheng, S.D., Jackson, M.M. & Wolpaw, J.R. 2008, 'Towards an independent brain-computer interface using steady state visual evoked potentials', *Clinical Neurophysiology*, vol. 119, no. 2, pp. 399-408.
- Anderson, C.W. & Bratman, J.A. 2008, 'Translating thoughts into actions by finding patterns in brainwaves', *Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems*, pp. 1-6.
- Anderson, C.W., Devulapalli, S.V. & Stolz, E.A. 1995a, 'Determining Mental State from EEG Signals Using Parallel Implementations of Neural Networks', *Scientific Programming*, vol. 4, no. 3, pp. 171-183.

-
- Anderson, C.W., Devulapalli, S.V. & Stolz, E.A. 1995b, 'EEG signal classification with different signal representations', *Neural Networks for Signal Processing [1995] V. Proceedings of the 1995 IEEE Workshop*, pp. 475-483.
- Anderson, C.W., Knight, J.N., O'Connor, T., Kirby, M.J. & Sokolov, A. 2006, 'Geometric subspace methods and time-delay embedding for EEG artifact removal and classification', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 142-146.
- Anderson, C.W. & Sijercic, Z. 1996, 'Classification of EEG signals from four subjects during five mental tasks', *Solving Engineering Problems with Neural Networks: Proceedings of the Conference on Engineering Applications in Neural Networks (EANN'96)*, Turkey, pp. 407-414.
- Anderson, C.W., Stolz, E.A. & Shamsunder, S. 1998, 'Multivariate autoregressive models for classification of spontaneous electroencephalographic signals during mental tasks', *IEEE Transactions on Biomedical Engineering*, vol. 45, no. 3, pp. 277-286.
- Andrzejak, R.G., Lehnertz, K., Mormann, F., Rieke, C., David, P. & Elger, C.E. 2001, 'Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state', *Physical Review E*, vol. 64, no. 6, p. 061907.
- Asvestas, P., Ventouras, E., Matsopoulos, G. & Karanasiou, I. 2014, 'Classification of Evoked Potentials Associated with Error Observation Using Artificial Neural Networks', *The 15th International Conference on Biomedical Engineering*, Springer, pp. 565-568.
- Australian Bureau of Statistics 2009, *Disability, Ageing and Carers, Australia: Summary of Findings*, viewed February 2010 <<http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/4430.02009>>.
- Bäck, T. & Schwefel, H.-P. 1993, 'An overview of evolutionary algorithms for parameter optimization', *Evolutionary computation*, vol. 1, no. 1, pp. 1-23.
- Bai, O., Rathi, V., Lin, P., Huang, D., Battapady, H., Fei, D.-Y., Schneider, L., Houdayer, E., Chen, X. & Hallett, M. 2011, 'Prediction of human voluntary movement before it occurs', *Clinical Neurophysiology*, vol. 122, no. 2, pp. 364-372.

-
- Balakrishnan, D. & Puthusserypady, S. 2005, 'Multilayer perceptrons for the classification of brain computer interface data', *Bioengineering Conference, 2005. Proceedings of the IEEE 31st Annual Northeast*, IEEE, pp. 118-119.
- Ball, T., Kern, M., Mutschler, I., Aertsen, A. & Schulze-Bonhage, A. 2009, 'Signal quality of simultaneously recorded invasive and non-invasive EEG', *NeuroImage*, vol. 46, no. 3, pp. 708-716.
- Barcelo, F., Perianez, J.A. & Nyhus, E. 2007, 'An information theoretical approach to task-switching: evidence from cognitive brain potentials in humans', *Frontiers in Human Neuroscience*, vol. 1.
- Barreto, A.B., Taberner, A.M. & Vicente, L.M. 1996, 'Classification of spatio-temporal EEG readiness potentials towards the development of a brain-computer interface', *Southeastcon'96. Bringing Together Education, Science and Technology*., *Proceedings of the IEEE*, IEEE, pp. 99-102.
- Barreto, G.A., Frota, R.A. & de Medeiros, F.N. 2004, 'On the classification of mental tasks: a performance comparison of neural and statistical approaches', *Machine Learning for Signal Processing, 2004. Proceedings of the 2004 14th IEEE Signal Processing Society Workshop*, IEEE, pp. 529-538.
- Başar, E., Özgören, M., Öniz, A., Schmiedt, C. & Başar-Eroğlu, C. 2007, 'Brain oscillations differentiate the picture of one's own grandmother', *Int. J. Psychophysiol*, vol. 64, no. 1, pp. 81-90.
- Bashashati, A., Fatourechi, M., Ward, R.K. & Birch, G.E. 2007, 'A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals', *J. Neural engineering*, vol. 4, p. R32.
- BCInet 2012, *NIA Game Controller*, viewed March 2011 <<http://www.bcinet.com/products/>>.
- Bear, M.F., Connors, B.W. & Michael, A. 2007, *Neuroscience: exploring the brain*, 3rd edn, Lippincott Williams & Wilkins.
- Bear, M.F., Connors, B.W. & Paradiso, M.A. 2007, *Neuroscience: Exploring the Brain*, 3rd Edition edn, Wolters Kluwer Health.
- Besserve, M., Garnero, L. & Martinerie, J. 2007, 'Cross-Spectral Discriminant Analysis (CSDA) for the classification of Brain Computer Interfaces', *Neural Engineering, 2007. CNE'07. 3rd International IEEE/EMBS Conference on*, IEEE, pp. 375-378.

-
- Beuchat, N.J., Chavarriaga, R., Degallier, S. & Millán, J.d.R. 2013, 'Offline decoding of upper limb muscle synergies from EEG slow cortical potentials', *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, IEEE, pp. 3594-3597.
- Birbaumer, N. & Cohen, L.G. 2007, 'Brain–computer interfaces: communication and restoration of movement in paralysis', *The Journal of Physiology*, vol. 579, no. 3, pp. 621-636.
- Birbaumer, N., Ghanayim, N., Hinterberger, T., Iversen, I., Kotchoubey, B., Kubler, A., Perelmueter, J., Taub, E. & Flor, H. 1999, 'A spelling device for the paralysed', *Nature*, vol. 398, no. 6725, pp. 297-298.
- Birbaumer, N., Hinterberger, T., Kubler, A. & Neumann, N. 2003, 'The thought-translation device (TTD): neurobehavioral mechanisms and clinical outcome', *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol. 11, no. 2, pp. 120-123.
- Birch, G.E., Bozorgzadeh, Z. & Mason, S.G. 2002, 'Initial on-line evaluations of the LF-ASD brain-computer interface with able-bodied and spinal-cord subjects using imagined voluntary motor potentials', *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 10, no. 4, pp. 219-224.
- Biswal, B.B., Mennes, M., Zuo, X.-N., Gohel, S., Kelly, C., Smith, S.M., Beckmann, C.F., Adelstein, J.S., Buckner, R.L. & Colcombe, S. 2010, 'Toward discovery science of human brain function', *Proceedings of the National Academy of Sciences*, vol. 107, no. 10, pp. 4734-4739.
- Blankertz, B., Curio, G. & Müller, K.-R. 2002, 'Classifying single trial EEG: Towards brain computer interfacing', *Advances in neural information processing systems*, vol. 1, pp. 157-164.
- Blankertz, B., Dornhege, G., Lemm, S., Krauledat, M., Curio, G. & Müller, K.-R. 2006, 'The Berlin Brain-Computer Interface: Machine Learning Based Detection of User Specific Brain States', *J. UCS*, vol. 12, no. 6, pp. 581-607.
- Blankertz, B., Sannelli, C., Halder, S., Hammer, E.M., Kübler, A., Müller, K.-R., Curio, G. & Dickhaus, T. 2010, 'Neurophysiological predictor of SMR-based BCI performance', *NeuroImage*, vol. 51, no. 4, p. 1303.
- Bonnet, L., Lotte, F. & Lécuyer, A. 2013, 'Two Brains, One Game: Design and Evaluation of a Multiuser BCI Video Game Based on Motor Imagery', *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 5, no. 2, pp. 185-198.
-

-
- Boord, P., Craig, A., Tran, Y. & Nguyen, H. 2010, 'Discrimination of left and right leg motor imagery for brain-computer interfaces', *Medical & biological engineering & computing*, vol. 48, no. 4, pp. 343-350.
- Bostanov, V. 2004, 'BCI competition 2003-data sets Ib and IIb: feature extraction from event-related brain potentials with the continuous wavelet transform and the t-value scalogram', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 1057-1061.
- Brunner, P., Bianchi, L., Guger, C., Cincotti, F. & Schalk, G. 2011, 'Current trends in hardware and software for brain-computer interfaces (BCIs)', *Journal of Neural engineering*, vol. 8, no. 2, p. 025001.
- Buch, E., Weber, C., Cohen, L.G., Braun, C., Dimyan, M.A., Ard, T., Mellinger, J., Caria, A., Soekadar, S. & Fourkas, A. 2008, 'Think to move: a neuromagnetic brain-computer interface (BCI) system for chronic stroke', *Stroke*, vol. 39, no. 3, pp. 910-917.
- Burke, D.P., Kelly, S.P., de Chazal, P., Reilly, R.B. & Finucane, C. 2005, 'A parametric feature extraction and classification strategy for brain-computer interfacing', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 13, no. 1, pp. 12-17.
- Burmistrov, E., Matlashov, A., Sandin, H., Schultz, L., Volegov, P. & Espy, M. 2013, 'Optimization and Configuration of SQUID Sensor Arrays for a MEG-MRI System', *Applied Superconductivity, IEEE Transactions on*, vol. 23, no. 3, pp. 1601304-1601304.
- Cabrera, A.F. & Dremstrup, K. 2008, 'Auditory and spatial navigation imagery in Brain-Computer Interface using optimized wavelets', *Journal of Neuroscience Methods*, vol. 174, no. 1, pp. 135-146.
- Chai, R., Hunter, G.P., Ling, S.H. & Nguyen, H.T. 2012a, 'Real-Time Microcontroller based Brain Computer Interface for Mental Task Classifications using Wireless EEG Signals from Two Channels ', *Proceedings of the 9th IASTED International Conference on Biomedical Engineering (BioMed 2012)*, ACTA Press, Innsbruck, pp. 336-342.
- Chai, R., Ling, S.H., Hunter, G.P. & Nguyen, H.T. 2012b, 'Mental non-motor imagery tasks classifications of brain computer interface for wheelchair commands using genetic algorithm-based neural network', *Proc. of the 2012 International Joint Conference on the Neural Networks (IJCNN)*, pp. 978-984.

-
- Chai, R., Ling, S.H., Hunter, G.P. & Nguyen, H.T. 2012c, 'Mental task classifications using prefrontal cortex electroencephalograph signals', *Proc. of the 34 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* pp. 1831-1834.
- Chai, R., Ling, S.H., Hunter, G.P. & Nguyen, H.T. 2012d, 'Toward fewer EEG channels and better feature extractor of non-motor imagery mental tasks classification for a wheelchair thought controller', *Proc. of the 34 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 5266-5269.
- Chai, R., Ling, S., Hunter, G., Tran, Y. & Nguyen, H. 2013a, 'Brain Computer Interface Classifier for Wheelchair Commands using Neural Network with Fuzzy Particle Swarm Optimization', *IEEE Journal of Biomedical and Health Informatics (retitled from IEEE Transaction on Information Technology in Biomedicine)* (In Press).
- Chai, R., Ling, S.H., Hunter, G.P., Tran, Y. & Nguyen, H.T. 2013b, 'Classification of Wheelchair Commands using Brain Computer Interface: Comparison between Able-Bodied Persons and Patients with Tetraplegia ', *Proc. of the 35 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2013, Osaka*, pp. 989-992.
- Chen, Y. & Feng, M.Q. 2003, 'A technique to improve the empirical mode decomposition in the Hilbert-Huang transform', *Earthquake Engineering and Engineering Vibration*, vol. 2, no. 1, pp. 75-85.
- Cheolsoo, P., Looney, D., ur Rehman, N., Ahrabian, A. & Mandic, D.P. 2013, 'Classification of Motor Imagery BCI Using Multivariate Empirical Mode Decomposition', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 21, no. 1, pp. 10-22.
- Cherry, S.R. & Phelps, M.E. 2002, 'Imaging brain function with positron emission tomography', in T. AW & M. JC (eds), *Brain mapping: The methods*, New York: Academic.
- Chiappa, S., Donckers, N., Bengio, S. & Vrins, F. 2004, 'HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems', *ESANN*, pp. 193-204.
- Chui, C.K. 1997, *Wavelets: a mathematical tool for signal analysis*, vol. 1, Siam.
- Chun-Hsiang, C., Li-Wei, K., Yuan-Pin, L., Tzyy-Ping, J. & Chin-Teng, L. 2014, 'Independent Component Ensemble of EEG for Brain-Computer Interface',

Neural Systems and Rehabilitation Engineering, *IEEE Transactions on*, vol. 22, no. 2, pp. 230-238.

- Cincotti, F., Mattia, D., Aloise, F., Bufalari, S., Schalk, G., Oriolo, G., Cherubini, A., Marciani, M. & Babiloni, F. 2008, 'Non-invasive brain-computer interface system: Towards its application as assistive technology', *Brain research bulletin*, vol. 75, no. 6, pp. 796-803.
- Clerc, M. & Kennedy, J. 2002, 'The particle swarm-explosion, stability, and convergence in a multidimensional complex space', *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 1, pp. 58-73.
- Comerchero, M.D. & Polich, J. 1999, 'P3a and P3b from typical auditory and visual stimuli', *Clinical Neurophysiology*, vol. 110, no. 1, pp. 24-30.
- Conson, M., Sacco, S., Sarà, M., Pistoia, F., Grossi, D. & Trojano, L. 2008, 'Selective motor imagery defect in patients with locked-in syndrome', *Neuropsychologia*, vol. 46, no. 11, pp. 2622-2628.
- Coyle, S., Ward, T., Markham, C. & McDarby, G. 2004, 'On the suitability of near-infrared (NIR) systems for next-generation brain-computer interfaces', *Physiological Measurement*, vol. 25, p. 815.
- Coyle, S.M., Ward, T.E. & Markham, C.M. 2007, 'Brain-computer interface using a simplified functional near-infrared spectroscopy system', *Journal of Neural engineering*, vol. 4, no. 3, p. 219.
- Craig, A., McIsaac, P., Tran, Y., Kirkup, L. & Searle, A. 1999, 'Alpha wave reactivity following eye closure: a potential method of remote hands free control for the disabled', *Technology and Disability*, vol. 10, no. 3, pp. 187-194.
- Craig, A., Moses, P., Tran, Y., McIsaac, P. & Kirkup, L. 2002, 'The effectiveness of a hands-free environmental control system for the profoundly disabled', *Archives of physical medicine and rehabilitation*, vol. 83, no. 10, pp. 1455-1458.
- Craig, A., Tran, Y., McIsaac, P., Moses, P., Kirkup, L. & Searle, A. 2000, 'The effectiveness of activating electrical devices using alpha wave synchronisation contingent with eye closure', *Applied Ergonomics*, vol. 31, no. 4, pp. 377-382.
- Craig, A., Tran, Y., Wijesuriya, N. & Nguyen, H. 2012, 'Regional brain wave activity changes associated with fatigue', *Psychophysiology*, vol. 49, no. 4, pp. 574-582.
- Craig, D.A. & Nguyen, H.T. 2007, 'Adaptive EEG Thought Pattern Classifier for Advanced Wheelchair Control', *Proc. IEEE 29th Annu. Int. Conf. Eng. Med. Biol. Soc.*, pp. 2544-2547.

-
- Craig, D.A., Nguyen, H.T. & Burchey, H.A. 2006, 'Two Channel EEG Thought Pattern Classifier', *Proc. of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 1291-1294.
- Croft, R.J., Chandler, J.S., Barry, R.J., Cooper, N.R. & Clarke, A.R. 2005, 'EOG correction: A comparison of four methods', *Psychophysiology*, vol. 42, no. 1, pp. 16-24.
- Curran, E., Sykacek, P., Stokes, M., Roberts, S.J., Penny, W., Johnsrude, I. & Owen, A.M. 2004, 'Cognitive tasks for driving a brain-computer interfacing system: a pilot study', *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 12, no. 1, pp. 48-54.
- Curran, E.A. & Stokes, M.J. 2003, 'Learning to control brain activity: A review of the production and control of EEG components for driving brain-computer interface (BCI) systems', *Brain and Cognition*, vol. 51, no. 3, pp. 326-336.
- Daubechies, I. 1992, *Ten lectures on wavelets*, vol. 61, SIAM.
- de Kruif, B.J., Schaefer, R. & Desain, P. 2007, 'Classification of Imagined Beats for use in a Brain Computer Interface', *Proc. of the 29 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 678-681.
- De Massari, D., Matuz, T., Furdea, A., Ruf, C.A., Halder, S. & Birbaumer, N. 2013, 'Brain-computer interface and semantic classical conditioning of communication in paralysis', *Biological Psychology*, vol. 92, no. 2, pp. 267-274.
- Decety, J., Perani, D., Jeannerod, M., Bettinardi, V., Tadary, B., Woods, R., Mazziotta, J.C. & Fazio, F. 1994, 'Mapping motor representations with positron emission tomography'.
- Delorme, A., Kothe, C., Vankov, A., Bigdely-Shamlo, N., Oostenveld, R., Zander, T.O. & Makeig, S. 2010, 'MATLAB-based tools for BCI research', in *Brain-Computer Interfaces*, Springer, pp. 241-259.
- Dien, J., Spencer, K.M. & Donchin, E. 2003, 'Localization of the event-related potential novelty response as defined by principal components analysis', *Cognitive Brain Research*, vol. 17, no. 3, pp. 637-650.
- Donoghue, J.P. 2002, 'Connecting cortex to machines: recent advances in brain interfaces', *Nature Neuroscience*, vol. 5, pp. 1085-1088.
- Emotiv 2010, *Epoc Neuro headset*, viewed 10 June 2010 <<http://www.emotiv.com/apps/epoc/299/>>.
-

-
- Eshelman, L.J. & Schaffer, J.D. 1993, 'Real-coded genetic algorithms and interval-schemata', *Foundation of Genetic Algorithms 2*, ed. D. Whitley, Morgan Kaufmann., pp. 187-202.
- Faradji, F., Ward, R.K. & Birch, G.E. 2009, 'Design of a mental task-based brain-computer interface with a zero false activation rate using very few EEG electrode channels', *Proc. 4th Int. IEEE/EMBS Conf. on Neural Eng.*, pp. 403-406.
- Farwell, L. & Donchin, E. 1988, 'Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials', *Electroencephalography and clinical Neurophysiology*, vol. 70, no. 6, pp. 510-523.
- Fatma Guler, N. & Ubeyli, E.D. 2007, 'Multiclass Support Vector Machines for EEG-Signals Classification', *IEEE Trans. Inf. Technol. in Biomed.*, vol. 11, no. 2, pp. 117-126.
- Fatourehchi, M., Bashashati, A., Ward, R.K. & Birch, G.E. 2007, 'EMG and EOG artifacts in brain computer interface systems: A survey', *Clinical Neurophysiology*, vol. 118, no. 3, pp. 480-494.
- Fazli, S., Mehnert, J., Steinbrink, J., Curio, G., Villringer, A., Müller, K.-R. & Blankertz, B. 2012, 'Enhanced performance by a hybrid NIRS-EEG brain computer interface', *NeuroImage*, vol. 59, no. 1, pp. 519-529.
- Felzer, T. & Freisieben, B. 2003, 'Analyzing EEG signals using the probability estimating guarded neural classifier', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 4, pp. 361-371.
- Fisch, B.J. & Spehlmann, R. 1999, *EEG Primer: Basic Principles of Digital and Analog EEG*, Elsevier.
- Franc, V. & Hlavac, V. 2004, *Statistical pattern recognition toolbox for Matlab*, Center for Machine Perception, Czech Technical University, <<http://cmp.felk.cvut.cz/cmp/software/stprtool/>>.
- Freeman, W.J., Rogers, L.J., Holmes, M.D. & Silbergeld, D.L. 2000, 'Spatial spectral analysis of human electrocorticograms including the alpha and gamma bands', *Journal of Neuroscience Methods*, vol. 95, no. 2, pp. 111-121.
- Friedrich, E.V., McFarland, D.J., Neuper, C., Vaughan, T.M., Brunner, P. & Wolpaw, J.R. 2009, 'A scanning protocol for a sensorimotor rhythm-based brain-computer interface', *Biological Psychology*, vol. 80, no. 2, pp. 169-175.

-
- Friedrich, E.V., Scherer, R. & Neuper, C. 2012, 'The effect of distinct mental strategies on classification performance for brain-computer interfaces', *International Journal of Psychophysiology*, vol. 84, no. 1, pp. 86-94.
- g.tech 2011, *Intendix*, viewed October 2011 <<http://www.intendix.com/>>.
- Gangadhar, G., Chavarriaga, R. & del R Millán, J. 2009, 'Fast recognition of anticipation-related potentials', *Biomedical Engineering, IEEE Transactions on*, vol. 56, no. 4, pp. 1257-1260.
- Gao, X., Xu, D., Cheng, M. & Gao, S. 2003, 'A BCI-based environmental controller for the motion-disabled', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 137-140.
- Gaona, C.M., Sharma, M., Freudenburg, Z.V., Breshears, J.D., Bundy, D.T., Roland, J., Barbour, D.L., Schalk, G. & Leuthardt, E.C. 2011, 'Nonuniform high-gamma (60–500 Hz) power changes dissociate cognitive task and anatomy in human cortex', *The Journal of Neuroscience*, vol. 31, no. 6, pp. 2091-2100.
- Garcia, G.N., Ebrahimi, T. & Vesin, J.-M. 2003, 'Support vector EEG classification in the Fourier and time-frequency correlation domains', *Neural Engineering, 2003. Conference Proceedings. First International IEEE EMBS Conference on*, IEEE, pp. 591-594.
- Garrett, D., Peterson, D.A., Anderson, C.W. & Thaut, M.H. 2003, 'Comparison of linear, nonlinear, and feature selection methods for EEG signal classification', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, no. 2, pp. 141-144.
- Georgieva, P., Silva, F., Milanova, M. & Kasabov, N. 2014, 'EEG Signal Processing for Brain-Computer Interfaces', in, *Springer Handbook of Bio-/Neuroinformatics*, Springer, pp. 797-812.
- Goldberg, D.E. & Deb, K. 1991, 'A comparative analysis of selection schemes used in genetic algorithms', *Urbana*, vol. 51, pp. 61801-62996.
- Goncharova, I., McFarland, D.J., Vaughan, T.M. & Wolpaw, J.R. 2003, 'EMG contamination of EEG: spectral and topographical characteristics', *Clinical Neurophysiology*, vol. 114, no. 9, pp. 1580-1593.
- Graimann, B., Huggins, J., Levine, S. & Pfurtscheller, G. 2002, 'Visualization of significant ERD/ERS patterns in multichannel EEG and ECoG data', *Clinical Neurophysiology*, vol. 113, no. 1, pp. 43-47.

-
- Graimann, B., Huggins, J.E., Levine, S.P. & Pfurtscheller, G. 2004, 'Toward a direct brain interface based on human subdural recordings and wavelet-packet analysis', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 954-962.
- Grefenstette, J.J., Gopal, R., Rosmaita, B.J. & Gucht, D.V. 1985, 'Genetic algorithms for the traveling salesman problem', *Proceedings of the 1st international conference on genetic algorithms*, L. Erlbaum Associates Inc., pp. 160-168.
- Grill, W.M. & Mortimer, J.T. 2000, 'Neural and connective tissue response to long - term implantation of multiple contact nerve cuff electrodes', *Journal of biomedical materials research*, vol. 50, no. 2, pp. 215-226.
- Grosse-Wentrup, M. & Schölkopf, B. 2013, 'A Review of Performance Variations in SMR-Based Brain-Computer Interfaces (BCIs)', in C. Guger, B.Z. Allison & G. Edlinger (eds), *Brain-Computer Interface Research*, Springer Berlin Heidelberg, pp. 39-51.
- Guo, S., Cooper, R.A., Boninger, M.L., Kwarciak, A. & Ammer, B. 2002, 'Development of power wheelchair chin-operated force-sensing joystick', *Proc. of the Second Joint Engineering in Medicine and Biology, 2002. 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society EMBS/BMES Conference*, vol. 3, pp. 2373-2374 vol.2373.
- Hagan, M.T. & Menhaj, M.B. 1994, 'Training feedforward networks with the Marquardt algorithm', *Neural Networks, IEEE Transactions on*, vol. 5, no. 6, pp. 989-993.
- Ham, F.M. & Kostanic, I. 2000, *Principles of neurocomputing for science and engineering*, McGraw-Hill Higher Education.
- Hammon, P.S. & de Sa, V.R. 2007, 'Preprocessing and meta-classification for brain-computer interfaces', *Biomedical Engineering, IEEE Transactions on*, vol. 54, no. 3, pp. 518-525.
- Haselsteiner, E. & Pfurtscheller, G. 2000, 'Using time-dependent neural networks for EEG classification', *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 4, pp. 457-463.
- He, B., Gao, S., Yuan, H. & Wolpaw, J. 2013, 'Brain-Computer Interfaces', in B. He (ed.), *Neural Engineering*, Springer US, pp. 87-151.

-
- Henle, C., Schuettler, M., Rickert, J. & Stieglitz, T. 2013, 'Towards Electrocorticographic Electrodes for Chronic Use in BCI Applications', in, *Towards Practical Brain-Computer Interfaces*, Springer, pp. 85-103.
- Hermes, D., Miller, K.J., Vansteensel, M.J., Aarnoutse, E.J., Leijten, F.S. & Ramsey, N.F. 2012, 'Neurophysiologic correlates of fMRI in human motor cortex', *Human brain mapping*, vol. 33, no. 7, pp. 1689-1699.
- Herrera, F., Lozano, M. & Verdegay, J.L. 1998, 'Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis', *Artificial intelligence review*, vol. 12, no. 4, pp. 265-319.
- Hinterberger, T., Neumann, N., Pham, M., Kübler, A., Grether, A., Hofmayer, N., Wilhelm, B., Flor, H. & Birbaumer, N. 2004, 'A multimodal brain-based feedback and communication system', *Experimental Brain Research*, vol. 154, no. 4, pp. 521-526.
- Hochberg, L.R., Bacher, D., Jarosiewicz, B., Masse, N.Y., Simeral, J.D., Vogel, J., Haddadin, S., Liu, J., Cash, S.S. & van der Smagt, P. 2012, 'Reach and grasp by people with tetraplegia using a neurally controlled robotic arm', *Nature*, vol. 485, no. 7398, pp. 372-375.
- Hochberg, L.R. & Donoghue, J.P. 2006, 'Sensors for brain-computer interfaces', *Engineering in Medicine and Biology Magazine, IEEE*, vol. 25, no. 5, pp. 32-38.
- Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Chen, D., Penn, R.D. & Donoghue, J.P. 2006, 'Neuronal ensemble control of prosthetic devices by a human with tetraplegia', *Nature*, vol. 442, no. 7099, pp. 164-171.
- Hoffmann, U., Garcia, G., Vesin, J.-M., Diserens, K. & Ebrahimi, T. 2005, 'A boosting approach to P300 detection with application to brain-computer interfaces', *Neural Engineering, 2005. Conference Proceedings. 2nd International IEEE EMBS Conference on*, IEEE, pp. 97-100.
- Holly, T.A., Abbott, B.G., Al-Mallah, M., Calnon, D.A., Cohen, M.C., DiFilippo, F.P., Ficaro, E.P., Freeman, M.R., Hendel, R.C. & Jain, D. 2010, 'Single photon-emission computed tomography', *Journal of nuclear cardiology*, vol. 17, no. 5, pp. 941-973.
- Horki, P., Solis-Escalante, T., Neuper, C. & Müller-Putz, G. 2011, 'Combined motor imagery and SSVEP based BCI control of a 2 DoF artificial upper limb', *Medical & biological engineering & computing*, vol. 49, no. 5, pp. 567-577.
-

-
- Hoya, T., Hori, G., Bakardjian, H., Nishimura, T., Suzuki, T., Miyawaki, Y., Funase, A. & Cao, J. 2003, 'Classification of single trial EEG signals by a combined principal+ independent component analysis and probabilistic neural network approach', *International Symposium on Independent Component Analysis and Blind Signal Separation*, pp. 197-202.
- Huan, N.-J. & Palaniappan, R. 2005, 'Classification of mental tasks using fixed and adaptive autoregressive models of EEG signals', *Neural Engineering, 2005. Conference Proceedings. 2nd International IEEE EMBS Conference on*, IEEE, pp. 633-636.
- Huang, N.E., Shen, Z. & Long, S.R. 1999, 'A new view of nonlinear water waves: The Hilbert Spectrum 1', *Annual review of fluid mechanics*, vol. 31, no. 1, pp. 417-457.
- Huang, N.E., Shen, Z., Long, S.R., Wu, M.C., Shih, H.H., Zheng, Q., Yen, N.C., Tung, C.C. & Liu, H.H. 1998, 'The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis', *Proc. Roy. Soc. London A*, vol. 454, no. 1971, pp. 903-995.
- Huang, N.E., Wu, M.L., Qu, W., Long, S.R. & Shen, S.S. 2003, 'Applications of Hilbert - Huang transform to non - stationary financial time series analysis', *Applied Stochastic Models in Business and Industry*, vol. 19, no. 3, pp. 245-268.
- Huang, N.E. & Wu, Z. 2008, 'A review on Hilbert-Huang transform: Method and its applications to geophysical studies', *Reviews of Geophysics*, vol. 46, no. 2.
- Huggins, J.E., Wren, P.A. & Gruis, K.L. 2011, 'What would brain-computer interface users want? Opinions and priorities of potential users with amyotrophic lateral sclerosis', *Amyotrophic Lateral Sclerosis*, vol. 12, no. 5, pp. 318-324.
- Huo, X. & Ghovanloo, M. 2010, 'Evaluation of a wireless wearable tongue-computer interface by individuals with high-level spinal cord injuries', *Journal of Neural engineering*, vol. 7.
- Jasper, H.H. 1958, 'The ten-twenty electrode system of the International Federation', *Electroencephalography and clinical Neurophysiology*, vol. 10, no. 1, pp. 371-375.
- Joines, J.A. & Houck, C.R. 1994, 'On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's', *Proc. of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pp. 579-584.
-

-
- Joseph, T. & Nguyen, H. 1998, 'Neural network control of wheelchairs using telemetric head movement', *Proc. of the 20 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 1998*, vol. 5, pp. 2731-2733 vol.2735.
- Juweid, M.E. & Hoekstra, O.S. 2011, *Positron emission tomography*, Springer.
- Kaper, M., Meinicke, P., Grossekhoefer, U., Lingner, T. & Ritter, H. 2004, 'BCI competition 2003-data set IIb: Support vector machines for the P300 speller paradigm', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 1073-1076.
- Käthner, I., Ruf, C.A., Pasqualotto, E., Braun, C., Birbaumer, N. & Halder, S. 2013, 'A portable auditory P300 brain-computer interface with directional cues', *Clinical Neurophysiology*, vol. 124, no. 2, pp. 327-338.
- Keirn, Z.A. & Aunon, J.I. 1990, 'A new mode of communication between man and his surroundings', *IEEE Trans. Biomedical Engineering*, vol. 37, no. 12, pp. 1209-1214.
- Kelly, S.P., Lalor, E.C., Reilly, R.B. & Foxe, J.J. 2005, 'Visual spatial attention tracking using high-density SSVEP data for independent brain-computer communication', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 13, no. 2, pp. 172-178.
- Kennedy, J. 2010, 'Particle swarm optimization', in *Encyclopedia of Machine Learning*, Springer, pp. 760-766.
- Kim, D.-W., Hwang, H.-J., Lim, J.-H., Lee, Y.-H., Jung, K.-Y. & Im, C.-H. 2011, 'Classification of selective attention to auditory stimuli: toward vision-free brain-computer interfacing', *Journal of Neuroscience Methods*, vol. 197, no. 1, pp. 180-185.
- Kita, H., Ono, I. & Kobayashi, S. 1998, 'Theoretical analysis of the unimodal normal distribution crossover for real-coded genetic algorithms', *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, IEEE, pp. 529-534.
- Klem, G.H., Lüders, H., Jasper, H. & Elger, C. 1999, 'The ten-twenty electrode system of the International Federation. The International Federation of Clinical Neurophysiology', *Electroenceph. Clin. Neurophysiol.*, vol. 52, p. 3.
- Kohlmorgen, J., Dornhege, G., Braun, M., Blankertz, B., Müller, K.-R., Curio, G., Hagemann, K., Bruns, A., Schrauf, M. & Kincses, W. 2007, 'Improving human

-
- performance in a real operating environment through real-time mental workload detection', *Toward Brain-Computer Interfacing*, pp. 409-422.
- Kostov, A. & Polak, M. 2000, 'Parallel man-machine training in development of EEG-based cursor control', *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 203-205.
- Krusienski, D.J., Grosse-Wentrup, M., Galán, F., Coyle, D., Miller, K.J., Forney, E. & Anderson, C.W. 2011, 'Critical issues in state-of-the-art brain-computer interface signal processing', *J. Neural Eng.*, vol. 8, no. 2, p. 025002.
- Kubler, A. & Muller, K.-R. 2007, 'An Introduction to Brain-Computer Interfacing', in G. Dornhege, J.R. Millan, T. Hinterberger, D.J. McFarland & K.-R. Muller (eds), *Toward Brain-Computer Interfacing*, The MIT Press, pp. 1-25.
- Kubler, A., Mushahwar, V.K., Hochberg, L.R. & Donoghue, J.P. 2006, 'BCI meeting 2005-workshop on clinical issues and applications', *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 14, no. 2, pp. 131-134.
- Kubler, A., Nijboer, F., Mellinger, J., Vaughan, T.M., Pawelzik, H., Schalk, G., McFarland, D.J., Birbaumer, N. & Wolpaw, J.R. 2005, 'Patients with ALS can use sensorimotor rhythms to operate a brain-computer interface', *Neurol.*, vol. 64, no. 10, pp. 1775-1777.
- Lal, S.K.L., Craig, A., Boord, P., Kirkup, L. & Nguyen, H. 2003, 'Development of an algorithm for an EEG-based driver fatigue countermeasure', *Journal of Safety Research*, vol. 34, no. 3, pp. 321-328.
- Lalor, E.C., Kelly, S.P., Finucane, C., Burke, R., Smith, R., Reilly, R.B. & Mcdarby, G. 2005, 'Steady-state VEP-based brain-computer interface control in an immersive 3D gaming environment', *EURASIP journal on applied signal processing*, vol. 2005, pp. 3156-3164.
- 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', *Comput. intell. and neurosci.*, vol. 2007, pp. 1-8.
- Lei, G., Youxi, W., Lei, Z., Ting, C., Weili, Y. & Xueqin, S. 2011, 'Classification of Mental Task From EEG Signals Using Immune Feature Weighted Support Vector Machines', *Magnetics, IEEE Transactions on*, vol. 47, no. 5, pp. 866-869.

-
- Lemm, S., Schafer, C. & Curio, G. 2004, 'BCI competition 2003-data set III: probabilistic modeling of sensorimotor μ rhythms for classification of imaginary hand movements', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 1077-1080.
- Leuthardt, E.C., Miller, K.J., Schalk, G., Rao, R.P. & Ojemann, J.G. 2006, 'Electrocorticography-based brain computer interface-the Seattle experience', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 194-198.
- Leuthardt, E.C., Schalk, G., Wolpaw, J.R., Ojemann, J.G. & Moran, D.W. 2004, 'A brain-computer interface using electrocorticographic signals in humans', *Journal of Neural engineering*, vol. 1, p. 63.
- Levine, S.P., Huggins, J.E., BeMent, S.L., Kushwaha, R.K., Schuh, L.A., Passaro, E.A., Rohde, M.M. & Ross, D.A. 1999, 'Identification of electrocorticogram patterns as the basis for a direct brain interface', *Journal of clinical neurophysiology*, vol. 16, no. 5, p. 439.
- Li, H., Zheng, H. & Tang, L. 2005, 'Hilbert-Huang transform and its application in gear faults diagnosis', *Key Engineering Materials*, vol. 291, pp. 655-660.
- Llera, A., Gómez, V. & Kappen, H. 2014, 'Adaptive Multiclass Classification for Brain Computer Interfaces', *Neural computation*, vol. 26, no. 6, pp. 1108-1127.
- Ling, S.H., Nguyen, H.T., Leung, F.H.F., Chan, K.Y. & Jiang, F. 2012, 'Intelligent fuzzy particle swarm optimization with cross-mutated operation', *Proc. 2012 IEEE Congress on Evolutionary Computation (CEC)*, pp. 3009-3016.
- Long, J., Li, Y., Wang, H., Yu, T., Pan, J. & Li, F. 2012, 'A hybrid brain computer interface to control the direction and speed of a simulated or real wheelchair', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 20, no. 5, pp. 720-729.
- Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F. & Arnaldi, B. 2007, 'A review of classification algorithms for EEG-based brain-computer interfaces', *J. Neural Eng.*, vol. 4, pp. R1-R13.
- Lu, J., Plataniotis, K.N. & Venetsanopoulos, A.N. 2003, 'Face recognition using LDA-based algorithms', *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 195-200.
- Makeig, S., Kothe, C., Mullen, T., Bigdely-Shamlo, N., Zhang, Z. & Kreutz-Delgado, K. 2012, 'Evolving signal processing for brain-computer interfaces', *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1567-1584.
-

-
- Marquardt, D.W. 1963, 'An algorithm for least-squares estimation of nonlinear parameters', *Journal of the Society for Industrial & Applied Mathematics*, vol. 11, no. 2, pp. 431-441.
- Martens, S., Hill, N., Farquhar, J. & Schölkopf, B. 2009, 'Overlap and refractory effects in a brain-computer interface speller based on the visual P300 event-related potential', *Journal of Neural engineering*, vol. 6, no. 2, p. 026003.
- Martinez, P., Bakardjian, H. & Cichocki, A. 2007, 'Fully online multicommand brain-computer interface with visual neurofeedback using SSVEP paradigm', *Computational intelligence and neuroscience*, vol. 2007, pp. 13-13.
- Mason, S.G. & Birch, G.E. 2000, 'A brain-controlled switch for asynchronous control applications', *Biomedical Engineering, IEEE Transactions on*, vol. 47, no. 10, pp. 1297-1307.
- Mason, S.G. & Birch, G.E. 2003, 'A general framework for brain-computer interface design', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 1, pp. 70-85.
- McCane, L.M., Sellers, E.W., McFarland, D.J., Mak, J.N., Carmack, C.S., Zeitlin, D., Wolpaw, J.R. & Vaughan, T.M. 2014, 'Brain-computer interface (BCI) evaluation in people with amyotrophic lateral sclerosis', *Amyotrophic Lateral Sclerosis and Frontotemporal Degeneration*, no. 0, pp. 1-9.
- McCarthy, G. & Donchin, E. 1981, 'A metric for thought: A comparison of P300 latency and reaction time', *Science*, vol. 211, no. 4477, pp. 77-80.
- McFarland, D.J., Anderson, C.W., Muller, K.R., Schlogl, A. & Krusienski, D.J. 2006, 'BCI meeting 2005-workshop on BCI signal processing: feature extraction and translation', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 135-138.
- McFarland, D.J., McCane, L.M., David, S.V. & Wolpaw, J.R. 1997, 'Spatial filter selection for EEG-based communication', *Electroencephalography and clinical Neurophysiology*, vol. 103, no. 3, pp. 386-394.
- McFarland, D.J. & Wolpaw, J.R. 2005, 'Sensorimotor rhythm-based brain-computer interface (BCI): feature selection by regression improves performance', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 13, no. 3, pp. 372-379.

-
- McFarland, D.J. & Wolpaw, J.R. 2008, 'Sensorimotor rhythm-based brain-computer interface (BCI): model order selection for autoregressive spectral analysis', *Journal of Neural engineering*, vol. 5, no. 2, p. 155.
- Mellinger, J., Schalk, G., Braun, C., Preissl, H., Rosenstiel, W., Birbaumer, N. & Kübler, A. 2007, 'An MEG-based brain-computer interface (BCI)', *NeuroImage*, vol. 36, no. 3, pp. 581-593.
- Menon, V. & Desmond, J.E. 2001, 'Left superior parietal cortex involvement in writing: integrating fMRI with lesion evidence', *Cognitive Brain Research*, vol. 12, no. 2, pp. 337-340.
- MettingVanRijn, A., Peper, A. & Grimbergen, C. 1994, 'Amplifiers for bioelectric events: A design with a minimal number of parts', *Medical and Biological Engineering and Computing*, vol. 32, no. 3, pp. 305-310.
- Michalewicz, Z. 1994, 'Genetic algorithms + data structures = evolution programs', 2nd, extended edition, *Springer-Verlag*, no. 1994.
- Middendorf, M., McMillan, G., Calhoun, G. & Jones, K.S. 2000, 'Brain-computer interfaces based on the steady-state visual-evoked response', *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 211-214.
- Milivojevic, B., Hamm, J.P. & Corballis, M.C. 2009, 'Hemispheric dominance for mental rotation: it is a matter of time', *Neuroreport*, vol. 20, no. 17, pp. 1507-1512.
- Millan, J. & Mouriño, J. 2003, 'Asynchronous BCI and local neural classifiers: an overview of the adaptive brain interface project', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 159-161.
- Millán, J.d.R., Franzé, M., Mouriño, J., Cincotti, F. & Babiloni, F. 2002, 'Relevant EEG features for the classification of spontaneous motor-related tasks', *Biological cybernetics*, vol. 86, no. 2, pp. 89-95.
- Millan, J.d.R., Galan, F., Vanhooydonck, D., Lew, E., Philips, J. & Nuttin, M. 2009, 'Asynchronous non-invasive brain-actuated control of an intelligent wheelchair', *Proc. IEEE 31st Annu. Int. Conf. Eng. Med. Biol. Soc.*, pp. 3361-3364.
- Millan, J.d.R. & Mourino, J. 2003, 'Asynchronous BCI and local neural classifiers: an overview of the adaptive brain interface project', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 159-161.

-
- Millán, J.d.R., Rupp, R., Müller-Putz, G.R., Murray-Smith, R., Giugliemma, C., Tangermann, M., Vidaurre, C., Cincotti, F., Kübler, A. & Leeb, R. 2010, 'Combining brain-computer interfaces and assistive technologies: state-of-the-art and challenges', *Frontiers in neuroscience*, vol. 4.
- Millan, J.R., Renkens, F., Mourino, J. & Gerstner, W. 2004, 'Noninvasive brain-actuated control of a mobile robot by human EEG', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 1026-1033.
- Millán, J.d.R. 2013, 'Brain-computer interfaces', *Introduction to Neural Engineering for Motor Rehabilitation*, vol. 40, p. 237.
- Mrachacz-Kersting, N., Niazi, I.K., Jiang, N., Pavlovic, A., Radovanović, S., Kostic, V., Popovic, D., Dremstrup, K. & Farina, D. 2013, 'A novel brain-computer interface for chronic stroke patients', in *Converging Clinical and Engineering Research on Neurorehabilitation*, Springer, pp. 837-841.
- Mühl, C., Allison, B., Nijholt, A. & Chanel, G. 2014, 'A survey of affective brain computer interfaces: principles, state-of-the-art, and challenges', *Brain-Computer Interfaces*, no. ahead-of-print, pp. 1-19.
- Mühlenbein, H. & Schlierkamp-Voosen, D. 1993, 'Predictive models for the breeder genetic algorithm i. continuous parameter optimization', *Evolutionary computation*, vol. 1, no. 1, pp. 25-49.
- Müller-Putz, G.R., Pokorny, C., Klobassa, D.S. & Horki, P. 2013, 'A single-switch BCI based on passive and imagined movements: Toward restoring communication in minimally conscious patients', *International Journal of Neural Systems*, vol. 23, no. 02.
- Müller-Putz, G.R., Scherer, R., Pfurtscheller, G. & Rupp, R. 2005, 'EEG-based neuroprosthesis control: A step towards clinical practice', *Neuroscience Letters*, vol. 382, no. 1-2, pp. 169-174.
- Muller, K.-R., Anderson, C.W. & Birch, G.E. 2003, 'Linear and nonlinear methods for brain-computer interfaces', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 165-169.
- Müller, K.-R., Krauledat, M., Dornhege, G., Curio, G. & Blankertz, B. 2004, 'Machine learning techniques for brain-computer interfaces'.
- Muller, K. & Blankertz, B. 2006, 'Toward noninvasive brain-computer interfaces', *Signal Processing Magazine, IEEE*, vol. 23, no. 5, pp. 128-126.
-

-
- Müller, M., Malinowski, P., Gruber, T. & Hillyard, S. 2003, 'Sustained division of the attentional spotlight', *Nature*, vol. 424, no. 6946, pp. 309-312.
- Musallam, S., Corneil, B., Greger, B., Scherberger, H. & Andersen, R. 2004, 'Cognitive control signals for neural prosthetics', *Science*, vol. 305, no. 5681, pp. 258-262.
- Naito, M., Michioka, Y., Ozawa, K., KIGUCHI, M. & KANAZAWA, T. 2007, 'A communication means for totally locked-in ALS patients based on changes in cerebral blood volume measured with near-infrared light', *IEICE transactions on information and systems*, vol. 90, no. 7, pp. 1028-1037.
- Nakayama, K. & Inagaki, K. 2006, 'A Brain Computer Interface Based on Neural Network with Efficient Pre-Processing', *Intelligent Signal Processing and Communications, 2006. ISPACS '06. International Symposium on*, pp. 673-676.
- Netech, *Minisim 330, EEG Simulator*, viewed 15 Dec 2012 <http://www.netechcorporation.com/productsdetails/22/9/EEG_Simulator>.
- Neubauer, A. 1997, 'A theoretical analysis of the non-uniform mutation operator for the modified genetic algorithm', *Evolutionary Computation, 1997., IEEE International Conference on*, IEEE, pp. 93-96.
- Neuper, C., Müller, G., Kübler, A., Birbaumer, N. & Pfurtscheller, G. 2003, 'Clinical application of an EEG-based brain-computer interface: a case study in a patient with severe motor impairment', *Clinical Neurophysiology*, vol. 114, no. 3, pp. 399-409.
- Neurosky 2010, *Mindset*, viewed March 2011 <<http://store.neurosky.com/collections/hardware/products/mindset>>.
- Nguyen, H.T. 2008, 'Intelligent technologies for real-time biomedical engineering applications', *Int. J. Automation and Control*, vol. 2, Nos.2/3, no. 2, pp. 274-285.
- Nguyen, H.T., King, L.M. & Knight, G. 2004, 'Real-time head movement system and embedded Linux implementation for the control of power wheelchairs', *Proc. of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2, pp. 4892-4895.
- Nicolas-Alonso, L.F. & Gomez-Gil, J. 2012, 'Brain computer interfaces, a review', *Sensors*, vol. 12, no. 2, pp. 1211-1279.
- Nicolelis, M.A., Dimitrov, D., Carmena, J.M., Crist, R., Lehew, G., Kralik, J.D. & Wise, S.P. 2003, 'Chronic, multisite, multielectrode recordings in macaque

-
- monkeys', *Proceedings of the National Academy of Sciences*, vol. 100, no. 19, pp. 11041-11046.
- Niedermeyer, E. & da Silva, F.H.L. 2005, *Electroencephalography: basic principles, clinical applications, and related fields*, Wolters Kluwer Health.
- Nijholt, A., Bos, D.P.-O. & Reuderink, B. 2009, 'Turning shortcomings into challenges: Brain-computer interfaces for games', *Entertainment Computing*, vol. 1, no. 2, pp. 85-94.
- Nolte, J. 2002, 'The human brain: An introduction to its functional anatomy'.
- OCZ Technology, *NIA Game Controller*, viewed March 2011 <<http://www.ocztechnology.com/nia-game-controller.html>>.
- Ogawa, S., Lee, T., Kay, A. & Tank, D. 1990, 'Brain magnetic resonance imaging with contrast dependent on blood oxygenation', *Proceedings of the National Academy of Sciences*, vol. 87, no. 24, pp. 9868-9872.
- Onishi, A. & Natsume, K. 2013, 'Ensemble regularized linear discriminant analysis classifier for P300-based brain-computer interface', *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, IEEE, pp. 4231-4234.
- Ono, I., Satoh, H. & Kobayashi, S. 1999, 'A real-coded genetic algorithm for function optimization using the unimodal normal distribution crossover', *Transactions of the Japanese Society for Artificial Intelligence*, vol. 14, pp. 1146-1155.
- OpenEEG 2008, <<http://openeeg.sourceforge.net/>>.
- Ortiz-Rosario, A. & Adeli, H. 2013, 'Brain-computer interface technologies: from signal to action', *Reviews in the Neurosciences*, vol. 24, no. 5, pp. 537-552.
- Osaka, M. 1984, 'Peak Alpha Frequency of EEG during a Mental Task: Task Difficulty and Hemispheric Differences', *Psychophysiology*, vol. 21, no. 1, pp. 101-105.
- Özgören, M., Başar-Eroğlu, C. & Başar, E. 2005, 'Beta oscillations in face recognition', *International Journal of Psychophysiology*, vol. 55, no. 1, pp. 51-59.
- Palaniappan, R. 2005, 'Brain Computer Interface Design Using Band Powers Extracted During Mental Tasks', *Proc. of the 2nd International IEEE/EMBS Conference on Neural Engineering*, pp. 321-324.

-
- Palaniappan, R., Paramesran, R., Nishida, S. & Saiwaki, N. 2002, 'A new brain-computer interface design using fuzzy ARTMAP', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 10, no. 3, pp. 140-148.
- Panicker, R.C., Puthusserypady, S. & Sun, Y. 2011, 'An asynchronous P300 BCI with SSVEP-based control state detection', *Biomedical Engineering, IEEE Transactions on*, vol. 58, no. 6, pp. 1781-1788.
- Park, S.-A., Hwang, H.-J., Lim, J.-H., Choi, J.-H., Jung, H.-K. & Im, C.-H. 2013, 'Evaluation of feature extraction methods for EEG-based brain-computer interfaces in terms of robustness to slight changes in electrode locations', *Medical & biological engineering & computing*, vol. 51, no. 5, pp. 571-579.
- Peixoto, N., Nik, H.G. & Charkhkar, H. 2013, 'Voice controlled wheelchairs: Fine control by humming', *Computer Methods and Programs in Biomedicine*, vol. 112, no. 1, pp. 156-165.
- Penny, W.D., Roberts, S.J., Curran, E.A. & Stokes, M.J. 2000, 'EEG-based communication: a pattern recognition approach', *Rehabilitation Engineering, IEEE Transactions on*, vol. 8, no. 2, pp. 214-215.
- Pfurtscheller, G., Allison, B.Z., Brunner, C., Bauernfeind, G., Solis-Escalante, T., Scherer, R., Zander, T.O., Mueller-Putz, G., Neuper, C. & Birbaumer, N. 2010a, 'The hybrid BCI', *Frontiers in neuroscience*, vol. 4.
- Pfurtscheller, G., Brunner, C., Schlogl, A. & Lopes da Silva, F. 2006a, 'Mu rhythm (de) synchronization and EEG single-trial classification of different motor imagery tasks', *NeuroImage*, vol. 31, no. 1, pp. 153-159.
- Pfurtscheller, G., Flotzinger, D. & Kalcher, J. 1993, 'Brain-computer interface—a new communication device for handicapped persons', *Journal of Microcomputer Applications*, vol. 16, no. 3, pp. 293-299.
- Pfurtscheller, G. & Lopes da Silva, F. 1999, 'Event-related EEG/MEG synchronization and desynchronization: basic principles', *Clinical Neurophysiology*, vol. 110, no. 11, pp. 1842-1857.
- Pfurtscheller, G., Muller-Putz, G., Schlogl, A., Graimann, B., Scherer, R., Leeb, R., Brunner, C., Keinrath, C., Lee, F. & Townsend, G. 2006b, '15 years of BCI research at Graz university of technology: current projects', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 205-210.

-
- Pfurtscheller, G., Müller, G.R., Pfurtscheller, J., Gerner, H.J. & Rupp, R. 2003a, 'Thought' - control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia', *Neuroscience Letters*, vol. 351, no. 1, pp. 33-36.
- Pfurtscheller, G., Müller, G.R., Pfurtscheller, J., Gerner, H.J. & Rupp, R. 2003b, 'Thought' – control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia', *Neuroscience Letters*, vol. 351, no. 1, pp. 33-36.
- Pfurtscheller, G., Neuper, C., Schlogl, A. & Lugger, K. 1998, 'Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters', *Rehabilitation Engineering, IEEE Transactions on*, vol. 6, no. 3, pp. 316-325.
- Pfurtscheller, G., Solis-Escalante, T., Ortner, R., Linortner, P. & Muller-Putz, G.R. 2010b, 'Self-Paced Operation of an SSVEP-Based Orthosis With and Without an Imagery-Based "Brain Switch": A Feasibility Study Towards a Hybrid BCI', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 18, no. 4, pp. 409-414.
- Pfurtscheller, G., Stancak Jr, A. & Neuper, C. 1996, 'Post-movement beta synchronization. A correlate of an idling motor area?', *Electroencephalography and clinical Neurophysiology*, vol. 98, no. 4, pp. 281-293.
- Pham, M., Hinterberger, T., Neumann, N., Kübler, A., Hofmayer, N., Grether, A., Wilhelm, B., Vatine, J.-J. & Birbaumer, N. 2005, 'An auditory brain-computer interface based on the self-regulation of slow cortical potentials', *Neurorehabilitation and Neural Repair*, vol. 19, no. 3, pp. 206-218.
- Polich, J. 2007, 'Updating P300: an integrative theory of P3a and P3b', *Clinical neurophysiology: official journal of the International Federation of Clinical Neurophysiology*, vol. 118, no. 10, p. 2128.
- Polikov, V.S., Tresco, P.A. & Reichert, W.M. 2005, 'Response of brain tissue to chronically implanted neural electrodes', *Journal of Neuroscience Methods*, vol. 148, no. 1, pp. 1-18.
- Prasad, A. & Sanchez, J.C. 2012, 'Quantifying long-term microelectrode array functionality using chronic in vivo impedance testing', *Journal of Neural engineering*, vol. 9, no. 2, p. 026028.
- Pregenzer, M. & Pfurtscheller, G. 1999, 'Frequency component selection for an EEG-based brain to computer interface', *Rehabilitation Engineering, IEEE Transactions on*, vol. 7, no. 4, pp. 413-419.
-

-
- Prutchi, D. & Norris, M. 2005, *Design and Development of Medical Electronic Instrumentation: A Practical Perspective of the Design, Construction, and Test of Medical Devices*, John Wiley & Sons, New Jersey.
- Qin, L. & He, B. 2005, 'A wavelet-based time–frequency analysis approach for classification of motor imagery for brain–computer interface applications', *Journal of Neural engineering*, vol. 2, no. 4, p. 65.
- Rakotomamonjy, A. & Guigue, V. 2008, 'BCI competition III: dataset II-ensemble of SVMs for BCI P300 speller', *Biomedical Engineering, IEEE Transactions on*, vol. 55, no. 3, pp. 1147-1154.
- Rakotomamonjy, A., Guigue, V., Mallet, G. & Alvarado, V. 2005, 'Ensemble of SVMs for improving brain computer interface P300 speller performances', in, *Artificial Neural Networks: Biological Inspirations–ICANN 2005*, Springer, pp. 45-50.
- Ramsey, N.F., van de Heuvel, M.P., Kho, K.H. & Leijten, F.S. 2006, 'Towards human BCI applications based on cognitive brain systems: an investigation of neural signals recorded from the dorsolateral prefrontal cortex', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 214-217.
- Ranganatha, S., Hoshi, Y. & Guan, C. 2005, 'Near infrared spectroscopy based brain-computer interface', *Microtechnologies for the New Millennium 2005*, International Society for Optics and Photonics, pp. 434-442.
- Rivera, S., Reiss, A., Eckert, M. & Menon, V. 2005, 'Developmental changes in mental arithmetic: evidence for increased functional specialization in the left inferior parietal cortex', *Cerebral Cortex*, vol. 15, no. 11, pp. 1779-1790.
- Royer, A.S., Doud, A.J., Rose, M.L. & He, B. 2010, 'EEG control of a virtual helicopter in 3-dimensional space using intelligent control strategies', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 18, no. 6, pp. 581-589.
- Samar, V.J., Bopardikar, A., Rao, R. & Swartz, K. 1999, 'Wavelet analysis of neuroelectric waveforms: a conceptual tutorial', *Brain and language*, vol. 66, no. 1, pp. 7-60.
- Sanei, S., Ferdowsi, S., Nazarpour, K. & Cichocki, A. 2013, 'Advances in Electroencephalography Signal Processing [Life Sciences]', *Signal Processing Magazine, IEEE*, vol. 30, no. 1, pp. 170-176.
- Sannelli, C., Braun, M., Tangermann, M. & Müller, K.-R. 2008, 'Estimating noise and dimensionality in BCI data sets: Towards illiteracy comprehension',
-

- Sato, S. 1990, *Magnetoencephalography*, Raven Press New York.
- Schalk, G., Kubanek, J., Miller, K., Anderson, N., Leuthardt, E., Ojemann, J., Limbrick, D., Moran, D., Gerhardt, L. & Wolpaw, J. 2007, 'Decoding two-dimensional movement trajectories using electrocorticographic signals in humans', *Journal of Neural engineering*, vol. 4, no. 3, p. 264.
- Scherer, R., Lee, F., Schlogl, A., Leeb, R., Bischof, H. & Pfurtscheller, G. 2008, 'Toward Self-Paced Brain-Computer Communication: Navigation Through Virtual Worlds', *Biomedical Engineering, IEEE Transactions on*, vol. 55, no. 2, pp. 675-682.
- Scherer, R., Muller, G., Neuper, C., Graimann, B. & Pfurtscheller, G. 2004, 'An asynchronously controlled EEG-based virtual keyboard: improvement of the spelling rate', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 979-984.
- Schlesinger, M.I. & Hlaváč, V. 2002, *Ten lectures on statistical and structural pattern recognition*, Kluwer Academic Publisher.
- Schlogl, A., Lee, F., Bischof, H. & Pfurtscheller, G. 2005, 'Characterization of four-class motor imagery EEG data for the BCI-competition 2005', *Journal of Neural engineering*, vol. 2, p. L14.
- Serruya, M.D., Hatsopoulos, N.G., Paninski, L., Fellows, M.R. & Donoghue, J.P. 2002, 'Brain-machine interface: Instant neural control of a movement signal', *Nature*, vol. 416, no. 6877, pp. 141-142.
- Shenoy, K.V., Meeker, D., Cao, S., Kureshi, S.A., Pesaran, B., Buneo, C.A., Batista, A.P., Mitra, P.P., Burdick, J.W. & Andersen, R.A. 2003, 'Neural prosthetic control signals from plan activity', *Neuroreport*, vol. 14, no. 4, pp. 591-596.
- Shibasaki, H. & Hallett, M. 2006, 'What is the Bereitschaftspotential?', *Clinical Neurophysiology*, vol. 117, no. 11, pp. 2341-2356.
- Simpson, R.C. & Levine, S.P. 2002, 'Voice control of a powered wheelchair', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, no. 2, pp. 122-125.
- Slutzky, M.W., Jordan, L.R., Krieg, T., Chen, M., Mogul, D.J. & Miller, L.E. 2010, 'Optimal spacing of surface electrode arrays for brain-machine interface applications', *Journal of Neural engineering*, vol. 7, no. 2, p. 026004.

-
- Sorger, B., Reithler, J., Dahmen, B. & Goebel, R. 2012, 'A Real-Time fMRI-Based Spelling Device Immediately Enabling Robust Motor-Independent Communication', *Current Biology*, vol. 22, no. 14, pp. 1333-1338.
- Srinivas, M. & Patnaik, L.M. 1994, 'Genetic algorithms: A survey', *Computer*, vol. 27, no. 6, pp. 17-26.
- Staba, R.J., Wilson, C.L., Bragin, A., Fried, I. & Engel, J. 2002, 'Quantitative analysis of high-frequency oscillations (80–500 Hz) recorded in human epileptic hippocampus and entorhinal cortex', *Journal of neurophysiology*, vol. 88, no. 4, pp. 1743-1752.
- Sussillo, D., Nuyujukian, P., Fan, J.M., Kao, J.C., Stavisky, S.D., Ryu, S. & Shenoy, K. 2012, 'A recurrent neural network for closed-loop intracortical brain-machine interface decoders', *Journal of Neural engineering*, vol. 9, no. 2, p. 026027.
- Szarowski, D., Andersen, M., Retterer, S., Spence, A., Isaacson, M., Craighead, H., Turner, J. & Shain, W. 2003, 'Brain responses to micro-machined silicon devices', *Brain research*, vol. 983, no. 1, pp. 23-35.
- Tan, S., Huang, J., Yang, Z. & Shi, Y.Q. 2006, 'Steganalysis of JPEG2000 Lazy-Mode Steganography using the Hilbert-Huang Transform Based Sequential Analysis', *Image Processing, 2006 IEEE International Conference on*, IEEE, pp. 101-104.
- Taylor, P.B. & Nguyen, H.T. 2003, 'Performance of a head-movement interface for wheelchair control', *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, vol. 2, pp. 1590-1593 Vol.1592.
- Teli, M.N. & Anderson, C. 2009, 'Nonlinear dimensionality reduction of electroencephalogram (EEG) for Brain Computer interfaces', *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pp. 2486-2489.
- Thuraisingham, R.A., Tran, Y., Craig, A., Wijesuriya, N. & Hung, N. 2009, 'Using microstate intensity for the analysis of spontaneous EEG: Tracking changes from alert to the fatigue state', *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pp. 4982-4985.
- Trejo, L.J., Rosipal, R. & Matthews, B. 2006, 'Brain-computer interfaces for 1-D and 2-D cursor control: designs using volitional control of the EEG spectrum or steady-state visual evoked potentials', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 225-229.

-
- van den Berg, M.E.L., Castellote, J.M., Mahillo-Fernandez, I. & de Pedro-Cuesta, J. 2010, 'Incidence of spinal cord injury worldwide: a systematic review', *Neuroepidemiology*, vol. 34, no. 3, pp. 184-192.
- Van Heertum, R.L., Tikofsky, R.S. & Ichise, M. 2000, *Functional cerebral SPECT and PET imaging*, 3rd edn, Lippincott Williams & Wilkins.
- Vanni, S., Revonsuo, A., Saarinen, J. & Hari, R. 1996, 'Visual awareness of objects correlates with activity of right occipital cortex', *Neuroreport*, vol. 8, no. 1, pp. 183-186.
- Varsta, M., Heikkonen, J. & Mourino, J. 2000, 'Evaluating the performance of three feature sets for brain-computer interfaces with an early stopping MLP committee', *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 2, IEEE, pp. 907-910.
- Vaughan, T.M., Heetderks, W.J., Trejo, L.J., Rymer, W.Z., Weinrich, M., Moore, M.M., Kübler, A., Dobkin, B.H., Birbaumer, N. & Donchin, E. 2003, 'Brain-computer interface technology: a review of the Second International Meeting', *IEEE transactions on neural systems and rehabilitation engineering: a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 11, no. 2, p. 94.
- Veigl, C., *BrainBay - an OpenSource Biosignal project*, <<http://www.shifz.org/brainbay/>>.
- Vetter, R.J., Williams, J.C., Hetke, J.F., Nunamaker, E.A. & Kipke, D.R. 2004, 'Chronic neural recording using silicon-substrate microelectrode arrays implanted in cerebral cortex', *Biomedical Engineering, IEEE Transactions on*, vol. 51, no. 6, pp. 896-904.
- Villringer, A., Planck, J., Hock, C., Schleinkofer, L. & Dirnagl, U. 1993, 'Near infrared spectroscopy (NIRS): a new tool to study hemodynamic changes during activation of brain function in human adults', *Neuroscience Letters*, vol. 154, no. 1, pp. 101-104.
- Vlek, R.J., Steines, D., Szibbo, D., Kübler, A., Schneider, M.-J., Haselager, P. & Nijboer, F. 2012, 'Ethical issues in Brain-Computer interface research, development, and dissemination', *Journal of Neurologic Physical Therapy*, vol. 36, no. 2, pp. 94-99.
- Wang, Y., Wang, Y.-T. & Jung, T.-P. 2010, 'Visual stimulus design for high-rate SSVEP BCI', *Electronics letters*, vol. 46, no. 15, pp. 1057-1058.

-
- Webster, J. 2010, *Medical Instrumentation: application and design*, Forth edn, John Wiley & Sons.
- Widrow, B. & Lehr, M.A. 1990, '30 years of adaptive neural networks: perceptron, madaline, and backpropagation', *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415-1442.
- Wilson, J.A., Felton, E.A., Garell, P.C., Schalk, G. & Williams, J.C. 2006, 'ECoG factors underlying multimodal control of a brain-computer interface', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 14, no. 2, pp. 246-250.
- Wilson, J.A., Guger, C. & Schalk, G. 2012, 'BCI Hardware and Software', in J.R. Wolpaw & E.W. Wolpaw (eds), *Brain-Computer Interfaces: Principles and Practice*, Oxford University Press, pp. 165-187.
- Woldorff, M.G. 1993, 'Distortion of ERP averages due to overlap from temporally adjacent ERPs: Analysis and correction', *Psychophysiology*, vol. 30, no. 1, pp. 98-119.
- Wolpaw, J. & Wolpaw, E.W. 2012, *Brain-computer interfaces: principles and practice*, Oxford University Press.
- Wolpaw, J.R. 2007, 'Brain-computer interfaces as new brain output pathways', *The Journal of Physiology*, vol. 579, no. 3, p. 613.
- Wolpaw, J.R. 2010, 'Brain-computer interface research comes of age: traditional assumptions meet emerging realities', *Journal of motor behavior*, vol. 42, no. 6, pp. 351-353.
- Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G. & Vaughan, T.M. 2002, 'Brain-computer interfaces for communication and control', *Clin. Neurophysiol.*, vol. 113, no. no. 6, pp. 767-791.
- Wolpaw, J.R., Loeb, G.E., Allison, B.Z., Donchin, E., do Nascimento, O.F., Heetderks, W.J., Nijboer, F., Shain, W.G. & Turner, J.N. 2006, 'BCI meeting 2005-workshop on signals and recording methods', *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 138-141.
- Wolpaw, J.R. & McFarland, D.J. 1994, 'Multichannel EEG-based brain-computer communication', *Electroencephalography and clinical Neurophysiology*, vol. 90, no. 6, pp. 444-449.

-
- Wolpaw, J.R. & McFarland, D.J. 2004, 'Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans', *Proc. of the National Academy of Sciences of the United States of America*, vol. 101, no. 51, pp. 17849-17854.
- Wolpaw, J.R., McFarland, D.J., Neat, G.W. & Forneris, C.A. 1991, 'An EEG-based brain-computer interface for cursor control', *Electroencephalography and clinical Neurophysiology*, vol. 78, no. 3, pp. 252-259.
- Wolpaw, J.R., McFarland, D.J., Vaughan, T.M. & Schalk, G. 2003, 'The Wadsworth Center brain-computer interface (BCI) research and development program', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 11, no. 2, pp. 1-4.
- Wolpaw, J.R., Ramoser, H., McFarland, D.J. & Pfurtscheller, G. 1998, 'EEG-based communication: improved accuracy by response verification', *IEEE Trans. Rehabilitation Engineering*, vol. 6, no. 3, pp. 326-333.
- Wu, Z. & Huang, N.E. 2009, 'Ensemble empirical mode decomposition: a noise-assisted data analysis method', *Advances in Adaptive Data Analysis*, vol. 1, no. 01, pp. 1-41.
- Yong, Y., Hurley, N. & Silvestre, G. 2005, 'Single-trial EEG classification for brain-computer interface using wavelet decomposition', *European Signal Processing Conference, EUSIPCO 2005*.
- Zhang, X., Yuhong, L., Zhang, F., Ren, J., Sun, Y.L., Yang, Q. & Huang, H. 2012a, 'On Design and Implementation of Neural-Machine Interface for Artificial Legs', *Industrial Informatics, IEEE Transactions on*, vol. 8, no. 2, pp. 418-429.
- Zhang, Y., Jin, J., Qing, X., Wang, B. & Wang, X. 2012b, 'LASSO based stimulus frequency recognition model for SSVEP BCIs', *Biomedical Signal Processing and Control*, vol. 7, no. 2, pp. 104-111.
- Zimmermann, R., Marchal-Crespo, L., Edelmann, J., Lambercy, O., Fluet, M.-C., Riener, R., Wolf, M. & Gassert, R. 2013, 'Detection of motor execution using a hybrid fNIRS-biosignal BCI: a feasibility study', *Journal of neuroengineering and rehabilitation*, vol. 10, no. 1, p. 4.