University of Strathclyde

Department of Biomedical Engineering

DEVELOPMENT OF FEATURE EXTRACTION ALGORITHMS FOR REAL-TIME BRAIN COMPUTER INTERFACE

by

Jetsada Arnin

A thesis submitted in fulfilment of the requirements

for the degree of Doctor of Philosophy

2021

DECLARATION OF AUTHENTICITY AND AUTHOR'S RIGHTS

This thesis is the result of the original research of the author. It has been composed by the author and has not been previously submitted for examination which has led to the award of any degree.

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by University of Strathclyde Regulation 3.50. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Jetsada Arnin

October 31, 2021

To my family …

and

"Proud" to be a Strathclyder …

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude and immense appreciation to Professor Bernard Conway for his guidance, valuable and constructive suggestions, unwavering support, encouragement, and infinite patience, and most importantly, for applying pressure and continuing to push me to accomplish my goal. I am very appreciative of the kind reception I have gotten. Additionally, I am grateful to Dr Danial Khahani and Dr Heba Lakany for their invaluable guidance and encouragement during my PhD studies. They are constantly making time to assist and advise me. It has been an honour to work with them, and the experience acquired has broadened my perspective.

A special thanks to our laboratory members, Yuly and Lukas, for their essential, kind, and supportive assistance in refining my experiment.

Lastly, I would like to express my gratitude to the Royal Thai Government and the Thailand Institute of Scientific and Technological Research (TISTR) for their financial support of this PhD project.

CONTENTS

ABSTRACT

Brain-computer interface (BCI) research is a multidisciplinary field of biomedical engineering that incorporates information from neuroscience, computer engineering, and electrical engineering. A BCI attempts to decode the brain's electrical activity to comprehend the neural system's function and transform it into outputs that augment, replace, repair, or improve human activities. The primary focus of BCI research is on creating robust, dependable, and user-friendly real-time systems. One of the most challenging aspects of developing real-time BCI is minimising the delay introduced by processing pipelines, which often decreases a system's computing performance. The thesis focuses on developing a real-time BCI signal processing method, particularly on the acceleration of the feature extraction phase. The purpose of the study is three-fold: a) to evaluate a feature extraction method that could be potential for real-time BCIs b) to modify the existing feature extraction algorithms in order to support real-time processing using different accelerating techniques c) to demonstrate the use of modified algorithms for real-time BCI applications.

The first study examined potential methods for real-time BCI feature extraction using a publicly available EEG data set. The findings established a difference between time and frequency domain techniques for a BCI with two classes of motor imaginary. Furthermore, the key findings show that although varying feature extraction methods did not result in a significant increase in classification accuracy, they did result in a considerable decrease in computation time. An additional examination is parallel computing to address the remaining complexity issues in the selected feature extraction methods. As a result, boosting computing time using a parallel processing technique is theoretically possible and flexible enough for real-time processing. However, in order to further reduce system latency, both hardware and software must be optimised when utilised in a real-time environment.

Additionally to the assessment findings, the discrete Fourier transform technique offered comparable classification accuracy but was computationally costly. This

almost eliminates the possibility of processing in real time. Another study solved this problem by proposing a new technique known as the “enhanced sliding discrete Fourier transform (eSDFT).” The eSDFT method is optimal for processing time-varying physiological data in real-time or near-real-time. The computational complexity analysis revealed that the proposed approach outperformed the traditional method. Furthermore, the suggested approach can be utilised for large-scale, real-time signal processing that uses a distributed computing infrastructure comprised of many compute units.

Apart from the Fourier transform-based feature extraction method, a straightforward Fourier transform cannot be used effectively for BCI processing because of the high temporal resolution of EEG signals, which often results in the loss of time information. Hence, the discrete wavelet transform (DWT) was selected to study, as it operates in both the time and frequency domains. This study examined the feasibility of creating a parallel computing approach for extracting time-frequency-domain characteristics. A modification of the DWT method has been proposed to enable parallel computing. The results show how effective hardware acceleration is in handling enormous computational demands. The suggested processing technique may be utilised to speed up any signal processing methodology.

The last study shows how to handle real-time signal processing using recursive approaches and parallel computing technologies. On a sliding window basis, the first and second generations of DWT algorithms were chosen for research. A variation of the conventional DWT method has been suggested to minimise duplication in the computation of coefficients. The findings indicate that the recursive approach has the shortest runtime and the most outstanding speedup ratio for all input lengths. Additionally, the proposed technique may be used for a wide variety of wavelet functions without degrading performance.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AEP	Auditory Evoked Potential
ALS	Amyotrophic Lateral Sclerosis
ANN	Artificial Neural Network
ANOVA	Analysis of Variance
API	Application Programming Interface
AR	Autoregressive Component
ASIC	Application-Specific Integrated Circuit
BCI	Brain-Computer Interface
bior	Biorthogonal
CAR	Common Average Reference
Cg	C for Graphics
CMRR	Common-Mode Rejection Ratio
coif	Coiflets
CPU	Central Processing Unit
CSP	Common Spatial Pattern
CWT	Continue Wavelet Transform
db	Daubechies
DFT	Discrete Fourier Transform
DSP	Digital Signal Processing
DWT	Discrete Wavelet Transform
ECG	Electrocardiogram
ECoG	Electrocorticography
EEG	Electroencephalography
EMD	Empirical Mode Decomposition
EMG	Electromyogram
EOG	Electrooculogram
ERD	Event-Related Desynchronisation

ERP	Event-Related Potential
ERS	Event-Related Synchronisation
FBS	Filter Bank Scheme
FES	Functional Electrical Stimulation
FFT	Fast Fourier Transform
FGWT	First-Generation Wavelet Transform
FIFO	First-In-First-Out
FIR	Finite Impulse Response
fMRI	Functional Magnetic Resonance Imaging
FPGA	Field-Programmable Gate Arrays
GA	Genetic Algorithm
GPU	Graphics Processing Unit
GUI	Graphic User Interface
HMPP	Hybrid Multi-Core Parallel Programming
HPC	High-Performance Computing
ICA	Independent Component Analysis
IIR	Infinite Impulse Response
ITR	Information Transfer Rate
KNN	K-Nearest Neighbour
LAP	Laplacian Reference
LDA	Linear Discriminant Analysis
LFP	Local Field Potential
LS	Lifting Scheme
MCU	Microcontroller Unit
MEA	Microelectrode Array
MEG	Magnetoencephalography
MF	Matched Filtering
MFT	Momentary Fourier Transform
MLS	Modified Lifting Scheme
MPI	Message Passing Interface
NMD	Nonlinear Mode Decomposition
OpenCL	Open Computing Language

OpenGL	Open Graphics Library
PCA	Principal Component Analysis
PSD	Power Spectral Density
QDA	Quadratic Discriminant Analysis
rbio	Reverse Biorthogonal
REST	Reference Electrode Standardisation Technique
RFE	Recursive Feature Elimination
SBS	Sequential Backward Selection
SCP	Slow Cortical Potential
SDFT	Sliding Discreate Fourier Transform
SDK	Software Development Kit
SFS	Sequential Forward Selection
SGWT	Second-Generation Wavelet
SIMD	Single Instruction Multiple Data
SMR	Sensorimotor Rhythm
SSVEP	Steady-State Visual Evoked Potential
SVM	Support Vector Machine
sym	Symlets
TPU	Tensor Processing Unit
VEP	Visual Evoked Potential
VLSI	Very Large-Scale Integration
VSM	Virtual Shared Memory
WT	Wavelet Transform

CHAPTER 1

INTRODUCTION

This chapter considers the research context and the studies motivation, together with an overall outline of the structure of the thesis. Initially, the primary research context is described in Section 1.1. Second, Section 1.2 highlights the research objectives of the thesis that are emerged from the research context. Then, the thesis statement is declared in Section 1.3. Next, Section 1.4 summarises lists of the contributions and the novel findings that emerge. After that, the thesis's organisation is explained in Section 1.5 with a list of publications in Section 1.6. Lastly, Section 1.7 summarises the chapter.

1.1 Research Context

Brain-computer interface (BCI) is a multidisciplinary area of biomedical engineering with knowledge contributions from neuroscience, computer engineering, and electrical engineering. A BCI aims at decoding the electrical activity of the brain to understand the function of the neural system and to convert it into outputs that enhance, replace, restore, or improve human functions through communication with a device. BCI applications are ideally designed to operate in real or near-real time and need to perform in a reliable, predictable and safe way. Key major topics for BCI research development are the implementation of real-time systems that are robust, reliable, and user-friendly. The present thesis investigates non-invasive BCI from four primary research contexts: (1) the advancement of large-scale neural recordings, (2) the development of cutting-edge techniques in digital signal processing, (3) the progression of high-performance computing technologies, and (4) neurorehabilitation research for improving sensorimotor functions.

1.1.1 Large-Scale Neural Recordings

Over many decades, neuroscience has advanced understanding of the structure and function of the nervous system. Moreover, it has revealed brain activity phenomena that underlie different human behaviours. Studying properties and function of neurons or ensemble brain activity is dependant of accessing neural recordings and observing changes that occur in relation to different types of cognitive, sensory or physical activity (Morshed and Khan, 2014). In general, neural recordings can be categorised into two main groups, those made via invasive and non-invasive techniques. The main difference between these two methods is that invasive methods provide higher temporal resolution and spatial resolution than the non-invasive methods that currently remain the preferred method for studying human brain activities.

Invasive neural recordings require surgery and the implantation or placement of recording electrodes on or near intracranial structures (including the surface of the cortex) or at depth within brain tissue at key areas of interest. The technologies include the use of microneedle electrode or microelectrode arrays (MEA) that measure neural spikes and local field potentials (LFP) and the use of non-penetrating surface grids to record electrical potentials associated with brain activity from the cerebral cortex, often referred to as electrocorticography (ECoG). In contrast, non-invasive techniques can be established without surgical risk and can provide more flexibility and safety than the invasive approach. For example, the most used tool in non-invasive brain-computer interface research is electroencephalography (EEG), which is a method of recording neural oscillations caused by the massed synchronous summation of neural activity arising as electrical fields generated by the brain and recordable from the scalp. In addition to methods based on measuring the electrical activity from the scalp, other non-invasive methods use sensors that measure the equivalent magnetic fields associated with this. Termed magnetoencephalography (MEG) this method has many similarities to EEG but is dependent on high-cost infrastructure and magnetic screening of sensitive sensors that limits its use in functional settings. While EEG and MEG are direct measures of neural activity, much progress in understanding brain dynamics has come from indirect measures that are based on changes in the oxygenation and flow of blood through brain regions. Here functional Magnetic

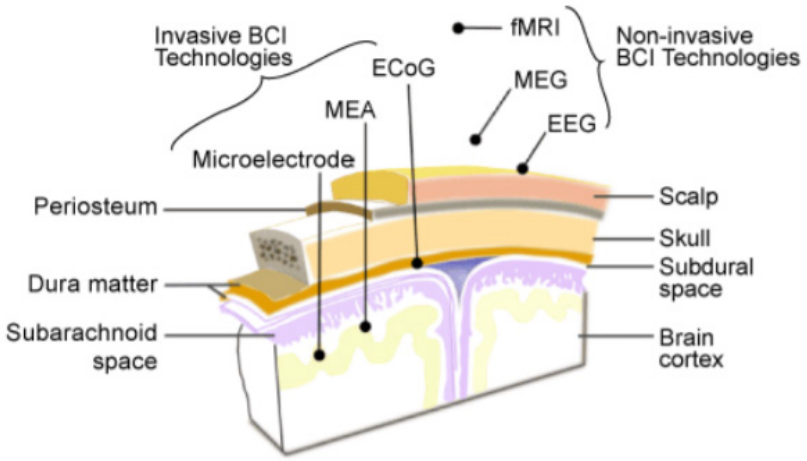

Figure 1.1. Electrode placement locations for different types of neural recordings: MEA, ECoG, EEG, MEG, and fMRI. Adapted from (Morshed and Khan, 2014).

Resonance Imaging (fMRI) has become a readily available tool for use in non-invasive research and for clinical neuroscience investigations of the brain. However, like MEG, fMRI requires significant infrastructure and can only image brain activity under conditions where the head is stationary within the scanner bore and therefore has limitations for use in BCI. A brief overview of standard neural recordings depending on electrode locations is illustrated in Figure 1.1.

1.1.2 Digital Signal Processing

Research on digital signal processing (DSP) has a long history. Many existing technologies in everyday life apply the fundamentals of the DSP, such as audio and acoustic applications, image and video applications, and communications. There are two major theoretical and conceptual frameworks for the DSP classified by types of application, i.e. offline applications and online or real-time applications. Applications related to offline processing considers computational complexity less than online processing as it is negotiable (Kuo et al., 2001). In contrast, real-time applications aim to avoid large accumulating processing latencies. Therefore, managing the latency in real-time processing is one of the most challenging tasks (Linderman et al., 2007) when dealing with complex signals and multiple channels. The differences in the characteristics between offline processing and online processing are discussed below:

1.1.2.1 Offline Signal Processing

In system identification, offline analysis is performed after gathering all the data from experiments, and the identification process happens only once (Puttige and Anavatti, 2007). The advantage of running offline is that computational resources can be extended as required by the complexity of the task and the availability and specification of the computational resources. This scheme is primarily used in offline machine learning and deep learning of large datasets (Halme and Parkkonen, 2018; Ang and Guan, 2016; Gu et al., 2009) as several training data sets can be transformed using feature engineering into larger sizes. In offline analysis, times to completion is not considered the most important application aspect of algorithm performance.

1.1.2.2 Real-Time Signal Processing

Real-time analysis can play an essential role in some specific applications, such as detecting changes in magnitude or signal oscillation in the time domain. In contrast to the offline analysis, the main difference of real-time analysis is that the parameters of the identified system can be updated after and during each experiment (Kuo et al., 2001). Generally, real-time signal processing is likely to process data at every single time step. This procedure is the most critical part as the computing power of the processing unit is usually restricted during the dual processes of acquisition and continuous running. For example, processing large amounts of data in real time frequently introduces an arithmetic overflow error when the system is not appropriately managed (Kurokawa et al., 1980). Many studies (Darwish and Fikri, 2006; Kountouris, 2009; Liu and Parhi, 2016) have been working to solve this critical issue through hardware (speed) and software (efficiency) approaches.

1.1.3 High-Performance Computing

Theoretically, high-performance computing (HPC) is related to the ability to process a considerable amount of data and perform complex calculations at high speeds (Dowd and Severance, 2010). The fundamental concept of the HPC is splitting complex problems into small parts that can perform calculations concurrently by multiple processors, to sum up for the results. Heterogeneous computing or parallel computing is another common term of the HPC. Regarding the process of parallel execution, each

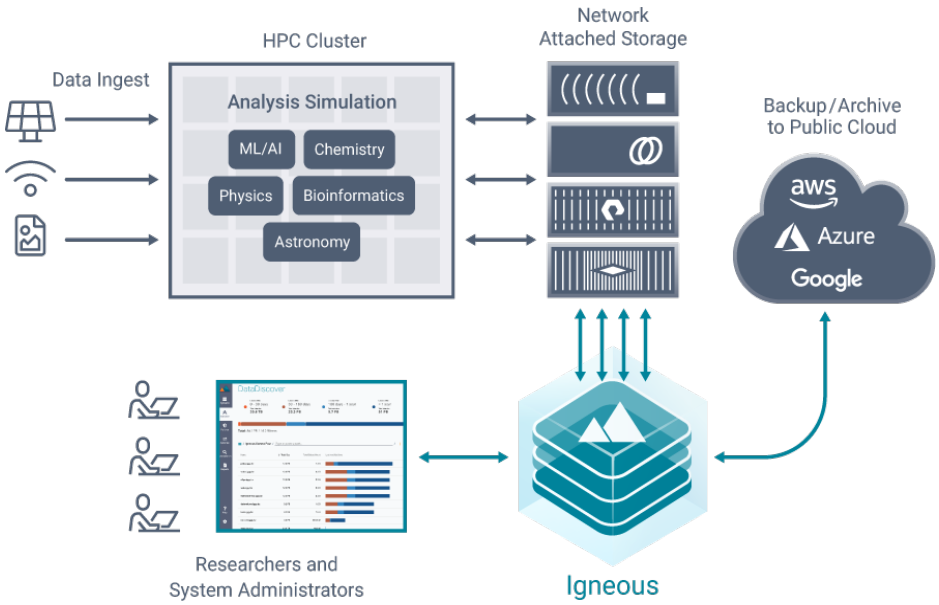

Figure 1.2. A complex architecture of high-performance computing technologies based on cloud computing platform, which the HPC cluster is responsible for complex calculations. Adapted from [https://www.igneous.io/industries/high-performance-computing].

processor communicates with others via shared memory so that the results are combined upon its completion. Presently, several machine learning and deep learning applications (Elsebakhi et al., 2015; Renggli et al., 2019; Schmidt and Hildebrandt, 2017) increasingly use HPC systems to accelerate calculations as their algorithms require a large volume of data for training and testing. According to the use of HPC, hardware acceleration has been designed to primarily support parallel operations in modern appliances, including personal laptops, mobile phones, and workstations.

Examples of hardware acceleration devices are graphics processing units (GPUs), field-programmable gate arrays (FPGA), and tensor processing units (TPUs). The devices can excessively speed up, for example, processing videos and images. Acceleration devices also offer running complex workloads remotely over the internet via cloud computing (Durao et al., 2014). The overview of the extensive usage of the HPC on cloud computing platforms is illustrated in Figure 1.2, which shows the HPC formed as a cluster executing calculations initiated by user requests. In addition to the advantages of using the HPC, recent theoretical and technical developments have revealed that hardware acceleration devices have been progressively impacting diverse areas such as physics (Pratx and Xing, 2011) and medicine (Rahman and Muniyandi,

2018; Smistad et al., 2015). Those success show that the HPC systems work well with big data, and they reduce computation time significantly.

1.1.4 Neurorehabilitation

Between the late 1980s and early 1990s, many advances in fundamental and clinical research resulted in significant improvements in neurorehabilitation treatments and the introduction of machines for neurorehabilitation. Three changes have occurred during that time period: (1) increased scientific evidence for the efficacy of intensive neurorehabilitation therapy derived from animal model studies; (2) the new promise of neuroregenerative treatments necessitating increased standardisation and outcome monitoring in rehabilitation practice; and (3) the opportunity and necessity to treat the growing neuropathic population. All three of these developments led to the understanding that properly built robotic devices and other assistive technologies may be beneficial for rehabilitation treatment on both a scientific and clinical level.

Neurorehabilitation can refer to a combination of two major processes, including an optimal adaptation to disability and a modification of disability in type or degree (Cole, 1988). The goal is to help recovery from nervous system injury or neurological disorders and to compensate for any functional changes resulting from it. With the development of rehabilitation technologies, multimodal treatments are emerging, providing an opportunity for more convenience and faster recovery. Regarding the interventions used in neurorehabilitation, there are several standard techniques. The majority of technologies depends on robotics and measurement, allowing for incorporation of long-term monitoring. For example, the integration of rehabilitation robots and neural recording devices (Do et al., 2013) was designed to monitor rehabilitation progress and provide feedback or performance during rehabilitation. The results showed that integrated systems could enhance neuromuscular functions with training in a short period. Apart from robot-assisted therapy, an effective combination of EEG-based BCI and functional electrical stimulation (FES) showed rapid rehabilitation progression in stroke patients (Lupu et al., 2018). Another novel technique (Shu et al., 2018) used magnetic stimulation for stroke rehabilitation. Figure 1.3 shows an example of a BCI-based rehabilitation system.

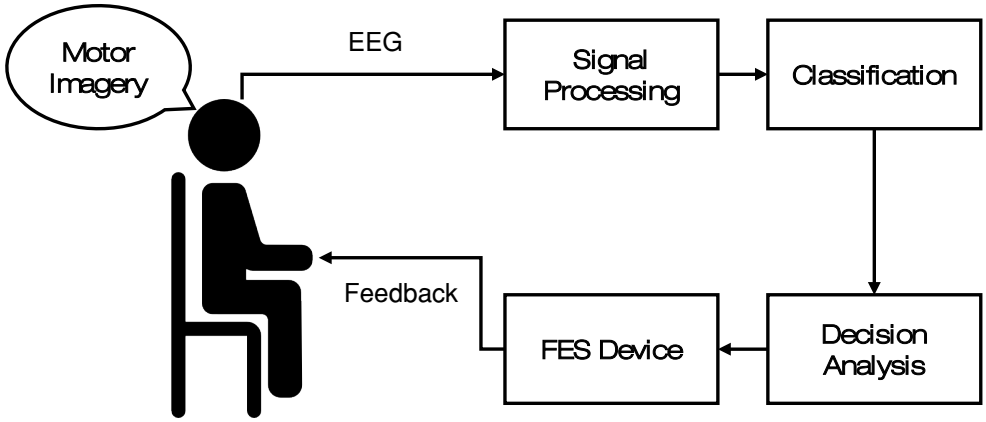

Figure 1.3. An example of general BCI-based neurorehabilitation systems. The system integrates EEG recordings and a functional electrical stimulation (FES) device to enhance neuromuscular functions. Based on this scheme, the user performs motor imagery tasks or related events and the stimulation will be triggered according to the classification results.

1.2 Research Objectives

According to the research motivation discussed in the previous section, there is a need to bring real-time BCI system processing into real-world applications that give the user fast and reliable control over assistive and complex devices (communication aids, robotic exoskeletons, personal navigation devices, etc.). In this section, the ultimate goals of the thesis and objectives and contributions are presented.

1.2.1 Evaluation of Real-Time Feature Extraction Methods

Feature extraction is an essential part of machine learning, pattern recognition and image processing. In short, feature extraction converts an initial set of raw data into a reduced number of new variables with meaningful features. Various modern classification algorithms require this kind of data transformation before passing through the algorithm. In BCI studies (Wang et al., 2018), a large number of feature extraction methods have been developed and reported upon. Those methods are applied in different scenarios based on time-domain methods and frequency-domain methods. However, implementing the feature extraction methods in real time always encounters a computational complexity issue (Sutton et al., 2018). This problem eventually leads to latency in signal processing which in BCI control leads to slowly

generated outcomes limiting the effectiveness of the assistive technology. Hence, selecting the appropriate algorithm to work in real time is required when optimising the system and its ability to control devices in relation to the users' intent. The present thesis aims at investigating the latency issue when applying feature extraction methods in real time, particularly for EEG-based BCI applications. A more detailed study and evaluation are presented in Chapter 3.

1.2.2 Development of Real-Time Feature Extraction Algorithms

Previous research showed that many feature extraction methods are designed based on sequential processing (Wierzgała et al., 2018; Pawar and Dhage, 2020). This means the calculation of the algorithm needs to run in a specific order. With the advantages of modern computing hardware offering parallel and distributed processing, multiple numerical calculations can be done within a single instruction, resulting in faster computation time. Therefore, these techniques have the potential to solve contemporary problems with high computational complexity. For instance, applying the parallel computing concept within proper hardware acceleration units can make real-time feature extraction possible (Ramkumar et al., 2019; Cho et al., 2018). In order to benefit from this advantage, any mathematical computation needs to be considered in a parallel computing scheme. As many feature extraction algorithms prefer running in order, not all algorithms can be modified to fit parallel computing schemes. However, there are often limitations for those algorithms that can be modified to fit the parallel architecture (Xiong et al., 2020). In the present thesis, feature extraction methods which are modified to be implemented in a parallel computing environment are proposed in Chapters 4 and 5.

1.2.3 Implementation of Multichannel Feature Extraction Techniques for Real-Time Signal Processing

Parallel computing is one of the core concepts of high-performance computing technologies (Dowd and Severance, 2010). Another important aspect is managing communications between HPC devices (Aluru and Jammula, 2013). If this part is not managed correctly, it can introduce additional and significant internal latencies that

paradoxically result in a slower computation time. However, understanding hardware acceleration devices can help to reduce the risk of this pitfall. In general, there are several parameters related to parallel computing schemes (Mittal and Vetter, 2015), such as buffer sizes, synchronisation, and types of variables. In order to use the HPC devices in real-time processing, setting up those parameters to optimum values is essential. Furthermore, as mentioned in Section 1.2.2 of the feature extraction algorithms developed for parallel computing schemes, it is of interest to apply them to process physiological signals that may be usable within BCI systems. Accordingly, the present thesis aims at implementing classification algorithms for the challenging problem of processing high-density EEG in real-time. Chapter 5 presents, in detail, the overall concept of a real-time BCI system, including hardware implementation and software development.

1.2.4 Real-Time Detection of Changes in Brain Activity

Several online BCI studies increasingly show the success rate of using BCI in rehabilitation (Young et al., 2014; Meng et al., 2008; Frolov et al., 2017; Ang et al., 2015). The most famous cases consist of applications for stroke patients and individuals with severe spinal cord injury. Most of those studies employed EEG-based BCI to enable rehabilitation devices, such as robot-assisted therapy and electric wheelchairs to assist patients with lost motor and movement disabilities. Using such an integrated system shows significant improvement in patient capability, especially those recovering from a stroke (Cervera et al., 2018). Apart from neurorehabilitation application, EEG-based BCIs have also been applied in the expanding domain of neuromodulation (Farzan et al., 2016), that is the modulation of nerve activity by delivering electrical, magnetic, or pharmaceutical agents directly to a target area in response to sensed neural activation patterns.

An FDA-approved example is the combination of transcranial magnetic stimulation with electroencephalography (TMS-EEG) (Esposito et al., 2020). This multimodal approach works effectively on various disorders, including depression, ADHD (Alyagon et al., 2020), and OCD (Metin et al., 2020). Hence, it would be of particular interest to detect changes in brain activity in real time, which can be used in

such applications. In the present thesis, real-time detection of changes in brain activity during self-paced motor execution using multichannel EEG is explored, as presented in Chapter 3 through Chapter 6, and builds from a research focus on restoring motor capability to those worst affected by spinal cord injury, stroke or other conditions affecting movement control.

1.3 Thesis Statement

Brain-computer interface is a promising technology enabling borderless communication between humans and machines. Many offline BCI studies have shown remarkable results using advanced techniques for neurological rehabilitation (Pfurtscheller et al., 2008; Daly and Wolpaw, 2008; Ang et al., 2015). However, implementing the same techniques into an online system does not often provide similar outcomes and system latency is a common problem. This is a considerable limitation for BCI systems intended for rehabilitation purposes as short processing time is highly significant and functionally desirable. Besides, developing a reliable real-time BCI system is exceptionally challenging. The majority of the online BCI studies reported in the literature focus on the use of EEG during movement intention and execution. This efficient EEG modality is simple to set-up and can be used in modern neurological rehabilitation. Hence, it is of interest to identify stages of the EEG capture and analysis in real time so that it can be used as a source for BCI commands. The present thesis hypothesises that changes in EEG activity during self-paced motor execution can be differentiated and predicted in real time based on some feature extraction methods that support multichannel EEG processing.

1.4 List of Contributions

This thesis investigates the existing techniques used in real-time feature extraction. Furthermore, it explores the possibility to implement parallel and distributed computing concepts into a real-time BCI system through modifications of current algorithms. This section summarises the main knowledge contributions of this thesis. To the best of the author's knowledge, the findings from this research have not been

explored before nor recorded in literature. Therefore, the scientific contributions of the thesis are listed below.

1. Evaluation of feature extraction methods for time domain and frequency domain in multichannel EEG recording using a pseudo-real-time system.
2. Modification of the state-of-the-art algorithm for real-time feature extraction based on a Fourier transform approach.
3. Modifying existing algorithms for real-time feature extraction for a parallel computing scheme based on a discrete wavelet transform approach.
4. Development of a simulation of a real-time BCI system for neurorehabilitation to evaluate the proposed algorithms by using high-performance computing technologies.
5. Low-latency identification of signal amplitude and frequency changes based on different feature extraction methods for multichannel processing.

1.5 Thesis Structure

The thesis is organised into seven chapters. Chapter 1 is the current chapter. Relevant background and state-of-the-art technologies for real-time signal processing, especially for brain-computer interface applications, are presented in Chapter 2. Moreover, a comprehensive explanation and description of real-time signal processing are introduced. Chapter 3 presents the study of real-time signal processing algorithms for feature extraction in EEG-based BCI systems. Chapter 4 focuses on developing real-time feature extraction algorithms based on the Fourier transform concept applied for parallel computing environments. Chapter 5 presents an alternate potential feature extraction method based on modified discrete wavelet transform theory. In Chapter 6, the modification of the proposed feature extraction methods for use in real-time signal processing environments is demonstrated. Finally, chapter 7 summarises the research, the highlight of the thesis, new insights, and limitations, followed by a future orientation of the present research.

In addition to the main chapters, additional and technical information of the implemented BCI system is provided in appendices, including a setup of real-time simulation system (Appendix A), a MATLAB script demonstrating the proposed

feature extraction algorithm presented in Chapter 4 (Appendix B), supplementary results of Chapter 5 (Appendix C), and additional results of Chapter 6 (Appendix D).

1.6 List of Publications

Some contents of the following publications are considered to be the essential parts of the thesis. The publications, including abstracts, peer-reviewed conference papers, and peer-reviewed journals, are listed as follows:

Abstracts/Posters:

1. Arnin, A., Kahani, D., and Conway, B. A. (2018). Development of high-speed brain computer interfacing technologies for neuromodulation. Abstract/Poster, Thai Student Academic Conference (TSAC2018), Brussels, Belgium.
2. Arnin, A., Kahani, D., and Conway, B. A. (2018). Implementation and evaluation of different time and frequency domain feature extraction methods for a two class motor imagery BCI applications: a performance comparison between GPU and CPU. Abstract/Poster, BioMedEng18, London, UK.

Peer-Reviewed Conference Papers:

1. Arnin, A., Kahani, D., Lakany, H., and Conway, B. A. (2018). Evaluation of different signal processing methods in time and frequency domain for brain-computer interface applications. In Proc. 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pages 235--238.
2. Arnin, A., Kahani, D., Lakany, H., and Conway, B. A. (2019). Heterogeneous real-time multichannel time-domain feature extraction using parallel sum reduction on GPU. In Proc. 8th Graz Brain Computer Interface Conference 2019 (GBCIC), pages 178—182.

Peer-Reviewed Journals:

1. Arnin, J., Kahani, D., and Conway, B. A. (Submitted). Modified discrete wavelet transform for multichannel real-time BCI: Algorithm and parallel implementation. *IEEE Transactions on Parallel and Distributed Systems.*

2. Arnin, J., Kahani, D., and Conway, B. A. (Submitted). The enhanced sliding discrete Fourier transform (eSDFT) algorithm. *IEEE Transactions on Signal Processing*.

1.7 Chapter Summary

This chapter provided an overview of the thesis, including the research context and motivation, research aims and objectives, thesis statement, list of contributions, thesis structure, and related publications.

In the next chapter, the related background and extensive literature review are explained in detail, and a clarification of the terminology used in real-time processing is also presented thoroughly.

1.8 References

Aluru, S. & Jammula, N. 2013. A review of hardware acceleration for computational genomics. *IEEE Design & Test,* 31**,** 19-30.

Alyagon, U., Shahar, H., Hadar, A., Barnea-Ygael, N., Lazarovits, A., Shalev, H. & Zangen, A. 2020. Alleviation of ADHD symptoms by non-invasive right prefrontal stimulation is correlated with EEG activity. *NeuroImage: Clinical,* 26**,** 102206.

Ang, K. K., Chua, K. S. G., Phua, K. S., Wang, C., Chin, Z. Y., Kuah, C. W. K., Low, W. & Guan, C. 2015. A randomised controlled trial of EEG-based motor imagery brain-computer interface robotic rehabilitation for stroke. *Clinical EEG and neuroscience,* 46**,** 310-320.

Ang, K. K. & Guan, C. 2016. EEG-based strategies to detect motor imagery for control and rehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 25**,** 392-401.

Cervera, M. A., Soekadar, S. R., Ushiba, J., Millán, J. D. R., Liu, M., Birbaumer, N. & Garipelli, G. 2018. Brain-computer interfaces for post-stroke motor rehabilitation: a meta-analysis. *Annals of clinical and translational neurology,* 5**,** 651-663.

Cho, Y., Kim, M. & Woo, S. 2018. High-speed CUDA Algorithm for GPU-based Feature Extraction in Iris Images. *International Information Institute (Tokyo). Information,* 21**,** 239-248.

Cole, M. 1988. 17 - Neurorehabilitation: Thoughts and Speculations on Some Basic Aspects. *In:* DAROFF, R. B. & CONOMY, J. P. (eds.) *Contributions to Contemporary Neurology.* Butterworth-Heinemann.

Daly, J. J. & Wolpaw, J. R. 2008. Brain–computer interfaces in neurological rehabilitation. *The Lancet Neurology,* 7**,** 1032-1043.

Darwish, H. A. & Fikri, M. 2006. Practical considerations for recursive DFT implementation in numerical relays. *IEEE Transactions on Power Delivery,* 22**,** 42-49.

Do, A. H., Wang, P. T., King, C. E., Chun, S. N. & Nenadic, Z. 2013. Brain-computer interface controlled robotic gait orthosis. *Journal of NeuroEngineering and Rehabilitation,* 10**,** 111.

Dowd, K. & Severance, C. 2010. High performance computing.

Durao, F., Carvalho, J. F. S., Fonseka, A. & Garcia, V. C. 2014. A systematic review on cloud computing. *The Journal of Supercomputing,* 68**,** 1321-1346.

Elsebakhi, E., Lee, F., Schendel, E., Haque, A., Kathireason, N., Pathare, T., Syed, N. & Al-Ali, R. 2015. Large-scale machine learning based on functional networks for biomedical big data with high performance computing platforms. *Journal of Computational Science,* 11**,** 69-81.

Esposito, R., Bortoletto, M. & Miniussi, C. 2020. Integrating TMS, EEG, and MRI as an Approach for Studying Brain Connectivity. *The Neuroscientist***,** 1073858420916452.

Farzan, F., Vernet, M., Shafi, M., Rotenberg, A., Daskalakis, Z. J. & Pascual-Leone, A. 2016. Characterising and modulating brain circuitry through transcranial magnetic stimulation combined with electroencephalography. *Frontiers in neural circuits,* 10**,** 73.

Frolov, A. A., Mokienko, O., Lyukmanov, R., Biryukova, E., Kotov, S., Turbina, L., Nadareyshvily, G. & Bushkova, Y. 2017. Post-stroke rehabilitation training with a motor-imagery-based brain-computer interface (BCI)-controlled hand exoskeleton: a randomised controlled multicenter trial. *Frontiers in neuroscience,* 11**,** 400.

Gu, Y., Farina, D., Murguialday, A. R., Dremstrup, K., Montoya, P. & Birbaumer, N. 2009. Offline identification of imagined speed of wrist movements in paralysed ALS patients from single-trial EEG. *Frontiers in Neuroscience,* 3**,** 3.

Halme, H.-L. & Parkkonen, L. 2018. Across-subject offline decoding of motor imagery from MEG and EEG. *Scientific reports,* 8**,** 1-12.

Kountouris, A. 2009. A randomised algorithm for controlling the round-off error accumulation in recursive digital frequency synthesis (DFS). *Digital Signal Processing,* 19**,** 534-544.

Kuo, S., Lee, B. H. & Tian, W. 2001. *Real-time digital signal processing*, Wiley Online Library.

Kurokawa, T., Payne, J. & Lee, S. 1980. Error analysis of recursive digital filters implemented with logarithmic number systems. *IEEE Transactions on Acoustics, Speech, and Signal Processing,* 28**,** 706-715.

Linderman, M. D., Santhanam, G., Kemere, C. T., Gilja, V., O'driscoll, S., Byron, M. Y., Afshar, A., Ryu, S. I., Shenoy, K. V. & Meng, T. H. 2007. Signal processing challenges for neural prostheses. *IEEE Signal Processing Magazine,* 25**,** 18-28.

Liu, Y. & Parhi, K. K. 2016. Architectures for recursive digital filters using stochastic computing. *IEEE Transactions on Signal Processing,* 64**,** 3705-3718.

Lupu, R. G., Irimia, D. C., Ungureanu, F., Poboroniuc, M. S. & Moldoveanu, A. 2018. BCI and FES Based Therapy for Stroke Rehabilitation Using VR Facilities. *Wireless Communications and Mobile Computing,* 2018**,** 4798359.

Meng, F., Tong, K.-Y., Chan, S.-T., Wong, W.-W., Lui, K.-H., Tang, K.-W., Gao, X. & Gao, S. BCI-FES training system design and implementation for rehabilitation of stroke patients. 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008. IEEE, 4103-4106.

Metin, S. Z., Balli Altuglu, T., Metin, B., Erguzel, T. T., Yigit, S., Arıkan, M. K. & Tarhan, K. N. 2020. Use of EEG for Predicting Treatment Response to Transcranial Magnetic Stimulation in Obsessive Compulsive Disorder. *Clinical EEG and Neuroscience,* 51**,** 139-145.

Mittal, S. & Vetter, J. S. 2015. A survey of CPU-GPU heterogeneous computing techniques. *ACM Computing Surveys (CSUR),* 47**,** 1-35.

Morshed, B. I. & Khan, A. 2014. A brief review of brain signal monitoring technologies for BCI applications: challenges and prospects. *Journal of Bioengineering & Biomedical Sciences,* 4**,** 1.

Pawar, D. & Dhage, S. 2020. Feature Extraction Methods for Electroencephalography based Brain-Computer Interface: A Review. *IAENG International Journal of Computer Science,* 47.

Pfurtscheller, G., Müller-Putz, G. R., Scherer, R. & Neuper, C. 2008. Rehabilitation with brain-computer interface systems. *Computer,* 41**,** 58-65.

Pratx, G. & Xing, L. 2011. GPU computing in medical physics: A review. *Medical physics,* 38**,** 2685-2697.

Puttige, V. R. & Anavatti, S. G. Comparison of real-time online and offline neural network models for a uav. 2007 International Joint Conference on Neural Networks, 2007. IEEE, 412-417.

Rahman, M. A. & Muniyandi, R. C. 2018. Review of GPU implementation to process of RNA sequence on cancer. *Informatics in Medicine Unlocked,* 10**,** 17-26.

Ramkumar, B., Laber, R., Bojinov, H. & Hegde, R. S. 2019. GPU acceleration of the KAZE image feature extraction algorithm. *Journal of Real-Time Image Processing***,** 1-14.

Renggli, C., Ashkboos, S., Aghagolzadeh, M., Alistarh, D. & Hoefler, T. Sparcml: High-performance sparse communication for machine learning. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2019. 1-15.

Schmidt, B. & Hildebrandt, A. 2017. Next-generation sequencing: big data meets high performance computing. *Drug discovery today,* 22**,** 712-717.

Shu, X., Chen, S., Chai, G., Sheng, X., Jia, J. & Zhu, X. Neural Modulation By Repetitive Transcranial Magnetic Stimulation (rTMS) for BCI Enhancement in Stroke Patients. 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 18-21 July 2018 2018. 2272-2275.

Smistad, E., Falch, T. L., Bozorgi, M., Elster, A. C. & Lindseth, F. 2015. Medical image segmentation on GPUs–A comprehensive review. *Medical image analysis,* 20**,** 1-18.

Sutton, J. R., Mahajan, R., Akbilgic, O. & Kamaleswaran, R. 2018. PhysOnline: an open source machine learning pipeline for real-time analysis of streaming physiological waveform. *IEEE journal of biomedical and health informatics,* 23**,** 59-65.

Wang, Z., Healy, G., Smeaton, A. F. & Ward, T. E. 2018. A review of feature extraction and classification algorithms for image RSVP based BCI. *Signal processing and machine learning for brain-machine interfaces*, 243-270.

Wierzgała, P., Zapała, D., Wojcik, G. M. & Masiak, J. 2018. Most popular signal processing methods in motor-imagery BCI: a review and meta-analysis. *Frontiers in neuroinformatics,* 12, 78.

Xiong, Q., Zhang, X., Wang, W.-F. & Gu, Y. 2020. A Parallel Algorithm Framework for Feature Extraction of EEG Signals on MPI. *Computational and Mathematical Methods in Medicine,* 2020.

Young, B. M., Nigogosyan, Z., Nair, V. A., Walton, L. M., Song, J., Tyler, M. E., Edwards, D. F., Caldera, K., Sattin, J. A. & Williams, J. C. 2014. Case report: post-stroke interventional BCI rehabilitation in an individual with preexisting sensorineural disability. *Frontiers in neuroengineering,* 7, 18.

CHAPTER 2

BACKGROUND & LITERATURE REVIEW

This chapter discusses the background and literature related to designing and implementing a real-time BCI system. The contents cover the fundamentals of EEG signals, a comprehensive review of different EEG modalities for real-time processing, technological trends in brain-computer interface, real-time EEG processing, and understanding artefacts in EEG and how to deal with them. Furthermore, the advance of high-performance computing platforms, comprising progress in hardware development and common practice, is also discussed.

2.1 EEG Modalities

Many studies explore the human brain to understand its function, characteristics, and how to interpret its patterns of activity systematically. In general, several BCI applications based on EEG signal analysis can be divided into categories based on signal mode (He et al., 2013), for example, sensorimotor rhythms (SMR), slow cortical potentials (SCP), P300 potentials (P300), event-related potentials (ERPs), and local field potentials (LFP). Table 2.1 summarises the main features of different types of EEG modalities used for brain-computer interfacing.

2.1.1 Sensorimotor Rhythm (SMR)

Electromagnetic recordings of the resting brain show extensive endogenous oscillatory activity. Extemporaneous activity has been referred to as the alpha band (between 8 Hz to 13 Hz) or the mu rhythm, which can be captured from the sensorimotor cortex area. Also, the visual alpha rhythm is considered when recovered from the visual cortex. Moreover, it is well established that this idling oscillation is influenced by

Table 2.1. Summary of the main features of different EEG modality in brain-computer interface.

Modality	Summary of Features
Sensorimotor Rhythm (SMR)	Observed in mu rhythm (8-13 Hz) and Beta (14-30 Hz). Event-related desynchronization (ERD) can be found in planning and an execution of movement. Event-related synchronization (ERS) can be often found during motor imagery (an increase of amplitude). Need learning to improve accuracy . Associated with motor images in the absence of real movement. Need calibration process to perform the BCI. Used in synchronous (cue-paced) or asynchronous environments (self-paced).
Slow Cortical Potential (SCP)	As a result of changes in the degrees of dendritic depolarization in particular cortical neurons (SCPs are owned by the portion of EEG signals that are less than 1 Hz in frequency). Occurred from 0.5-10 seconds after the onset of an internal event. Positive SCP indicates increased cortical activity, while negative SCP indicates decreased cortical activation. People can be taught to generate voluntary SCP changes. Yielded 80% of average accuracy, relative low information transfer rate, longer continuous training (months).
P300 Event-Related Potential (P300)	A natural reaction that is advantageous in situations when sufficient training time is lacking or the user is not readily taught. No training session required. Considered as endogenous BCIs, arisen more than 100 milliseconds of onset latency. Low SNR (difficult to detect), very low ITR, too low accuracy.
Event-Related Potentials (ERPs)	Occurred in the EEG at a predetermined time interval after the presentation of a visual, auditory, or somatosensory input. Frequently used by auditory and VEP (SSVEP). High information transfer rate (particularly SSVEP). Considered as exogenous BCIs, happened less than 100 milliseconds of onset latency.
Spike and Local Field Potential (LFP)	Acquired through the implantation of microelectrodes using invasive methods. By using the specific firing rates of many neurons, it is possible to achieve multidimensional control for a BCI.

intricate thalamocortical neuronal connections inside complex feedback loops. In these feedback loops, the synchronous synaptic activity of the neurons produces noticeable oscillations based on recording of local fields. The fundamental membrane properties of neurons, the kinetics of synaptic processes, the strength and complexity of connections within the neural network, and the impacts of different neurotransmitter systems all contribute to the frequency and duration of oscillations.

Additionally, oscillations in the beta (between 14 Hz to 30 Hz) and gamma frequency (more than 30 Hz) bands have been observed across the sensorimotor cortex. These oscillations observed throughout the sensorimotor cortex, together with the mu rhythm, are called sensorimotor rhythm or SMR. The sensorimotor cortex has been primarily recognised as the origin of SMRs, and their presence and magnitude change with motor and somatosensory function. During idling or rest, these oscillations arise repeatedly. However, these oscillations are different in amplitude and frequency during non-idling periods, and these changes are detectable using EEG or MEG methods. In general, a variation that relates to tasks in SMRs is recognised as a reduction in the amplitude of the low-frequency components (alpha/beta band or mu-rhythm) (sometimes referred to as event-related desynchronization or ERD) (Pfurtscheller and da Silva, 1999). By contrast, increasing the amplitude of a frequency band is referred to as event-related synchronisation (ERS) (Pfurtscheller and da Silva, 1999). For example, during motor planning and execution it is common to observe reductions in the alpha and beta frequency bands compared to resting activity levels (Pfurtscheller and da Silva, 1999).

Figure 2.1 illustrates the presence of these kinds of frequency band changes in EEG activity. Numerous investigations have shown that motor imaging may cause ERD (and frequently ERS) in central sensorimotor regions. (Pfurtscheller and da Silva, 1999; Pfurtscheller et al., 1997; McFarland et al., 2000; Wang et al., 2004; Wang and He, 2004; Yamawaki et al., 2006; Miller et al., 2010). Thus, using the features associated with spectral changes in brain activity in the preparation, execution, or imagination of movement allows categorisation of different brain conditions with movement intention. This implementation is the foundation of neural control in BCIs based on movement intention detection. Associated with this, several studies have shown that individuals may acquire the ability to control the amplitude of sensorimotor rhythms across one hemisphere utilising motor imagining techniques. (Wolpaw et al., 1991; Wolpaw and McFarland, 2004; Doud et al., 2011; McFarland et al., 2010; Royer et al., 2010; Yuan et al., 2008).

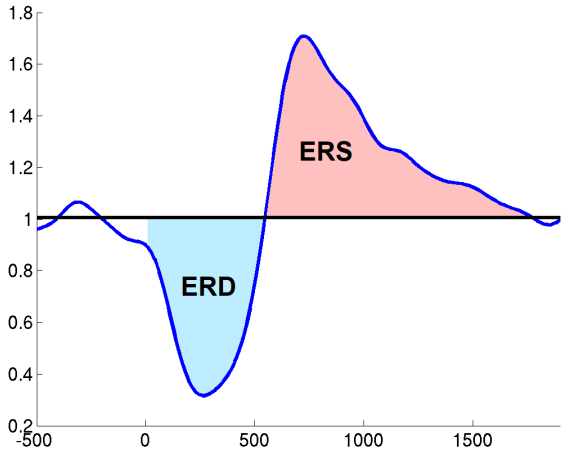

Figure 2.1. Presenting mu-rhythmic brain electric oscillatory oscillations in response to stimulus. ERD is described as a phasic relative power drop in a certain frequency band, while ERS is defined as a relative power rise in that frequency band, both of which occur in response to stimulation. After the event-onset ($t = 0$), the mu-rhythm desynchronises, visible from the decrease in averaged band power (ERD). Then it is followed by an increase in signal power (ERS), overshooting the pre-event baseline level and thus yeilding in the standard biphasic ERD-ERS response.

2.1.2 Slow Cortical Potential (SCP)

The slow cortical potentials (SCP), a unique category of signals measured by EEG, are produced by changes in the depolarisation levels of pyramidal neurons in the cortex, lasting from several hundred milliseconds to several seconds. The SCP is the lowest frequency feature in scalp-recorded EEG and has amplitudes ranging from a few microvolts (μV) to more than 100 μV. (e.g., during seizures). A negative SCP indicates cortical activation typically associated with impending voluntary movement, while a positive SCP is usually associated with reduced cortical activation. Several studies (Lutzenberger et al., 1982; Rockstroh et al., 1982) have found that the negative SCP represents lower excitation thresholds of underlying neural structures, resulting in processing facilitation during stages of behavioural or cognitive preparation.

SCPs are not spontaneous activities and are part of the family of event-related potentials often visualised using averaging techniques with multiple aspects. The fact that brain potentials in this category are time-locked to a single event is their defining feature. Physical inputs, behavioural responses, and cognitive and emotional processes are examples of external or interior occurrences. With training SCPs can be regulated and used to control a simple BCI (Birbaumer et al., 1999; Birbaumer et al., 2000).

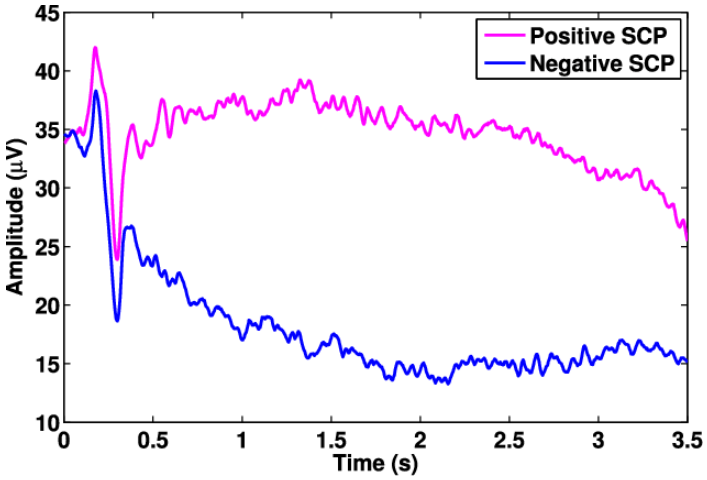

Figure 2.2. Demonstraing mean traces of the positive SCPs and the negative SCPs (Hou et al., 2017). A negative shift lowers the excitability of the underlying cortical region, whereas a positive shift is thought to be suppression of excitation and/or energy consumption.

Figure 2.2 shows example averages of a positive SCP and a negative SCP (Hou et al., 2017).

2.1.3 P300 Event-Related Potential (P300)

P300 is referred to as a standard endogenous event-related potential that is often investigated in psychological studies of cognition (Donchin and Coles, 1988). The P300 is a positive deflection in the EEG signal that emerges around 300 milliseconds after an attended stimulus is presented. The P300 is well-known and is a helpful tool for evaluating cognitive function. P300 components (for example, amplitude, latency, and energy) indicate information processing related to attention, stimulus processing speed, error awareness, and memory performance.

Figure 2.3 shows a recorded P300 waveform. In this paradigm, a user is subjected to recurrent occurrences that fall into two different types (target and non-target). Events that fall into one of the two types happen infrequently. The user is assigned a task that may be completed only via the categorization of each event. When an uncommon event occurs, the EEG displays a P300 response. The P300 component's amplitude is inversely proportional to the frequency of the intermittent occurrences. When the subject focuses on the target displayed on the screen, the P300 component appears. This ERP feature is resilient, making it particularly advantageous when a lengthy training period is impossible or when the user has been intensely trained

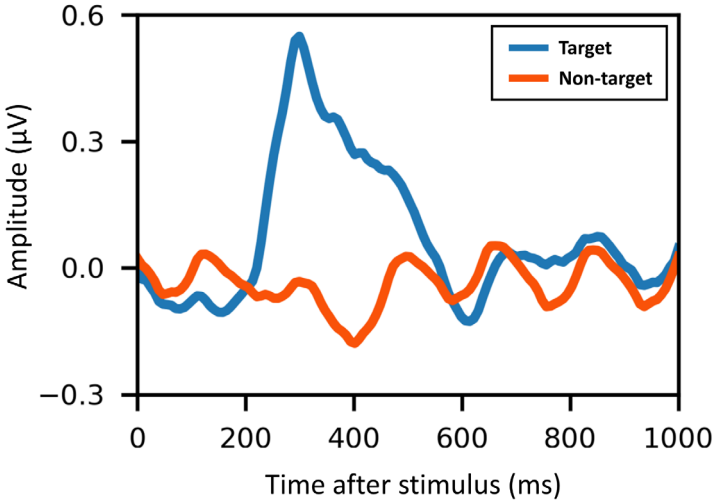

Figure 2.3. Showing the P300 component captured from the scalp EEG. When the subject focuses on the target stimulus which is displayed on the screen, the P300 component will obviously appears.

(Spencer et al., 2001). At present, the P300-based BCIs are generally configured as communication aids and are the only class of BCI systems that severely handicapped individuals use on a daily basis in their homes (Sellers et al., 2010).

2.1.4 Event-Related Potentials (ERPs)

Brain responses that may be recovered from the electroencephalogram at a defined time interval after an auditory, visual, somatosensory, or intrinsic event are called exogenous event-related potentials (ERPs) (Rugg and Curran, 2007). A widely used technique for obtaining ERP from EEG is to record signals parallel to the start of the stimulus and then average them in relation to the commencement of the event. The average number of stimuli characteristically ranges from a few to hundreds or thousands, depending on the neuroscience study and signal to noise ratio. Exogenous or endogenous ERPs are frequently interchangeably used. Exogenous ERPs, on the other hand, often have a shorter latency and are primarily affected by the triggering stimuli. In comparison, the endogenous ERP may have longer latency and are mainly influenced by the subject's concurrent brain activity or attention.

The ERPs can be measured as temporal amplitude changes or alterations in band power and are associated with the ERD and ERS described earlier. ERPs indicate an extensive portion of activity in the continuing EEG that is phase-locked by the stimuli.

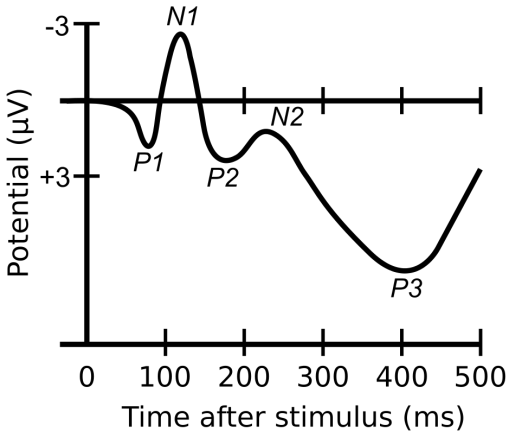

Figure 2.4. Demonstrating a wide variety of ERPs used in BCI. ERPs refer to changes in electrical activity in the brain in response to external stimuli. In general, an ERP can be detected by following (or preceding) the onset of specific visual, auditory, or other sensory stimuli, and cues signaling motor preparation, motor execution, or motor imagery.

After averaging, the ERP temporal structures usually include information on very-low-frequency elements (less than 1 Hz). Other devices are ignored throughout the averaging process via recurrences, and the information over 1 Hz is only partially displayed. An alternate approach for characterising task-related EEG data is to investigate EEG activity prior to averaging, either in power (ERD/ERS) or phase (Mayaud et al., 2013). Because no averaging is required, this technique may be utilised in single trials. As a result, it is beneficial to use in BCIs (even though the signal-to-noise ratio still constrains it). Figure 2.4 illustrates an example of well-known ERP components utilised in BCI.

Along with ERPs, the most often used ERP in BCI systems is the visual evoked potential (VEP). The steady-state visual evoked potential (SSVEP) is a widely used VEP that has been extensively developed for communication assistance, particularly for spinal cord injury patients with normal visual function. SSVEPs and other VEPs are gaze-dependent and therefore need muscular control to move the head or eyes. In general, the user views a few items flashing on a screen. The items are flashed at various frequencies in either the alpha band or beta band in order to produce coherent signals in the brain regions that react to visual inputs. The SSVEP's frequency analysis reveals the highest magnitude at the frequency of the item the user is viewing. As a consequence, the SSVEP technique may categorise the item the user wishes to choose based on the frequency of this peak. (Middendorf et al., 2000; Ortner et al., 2011).

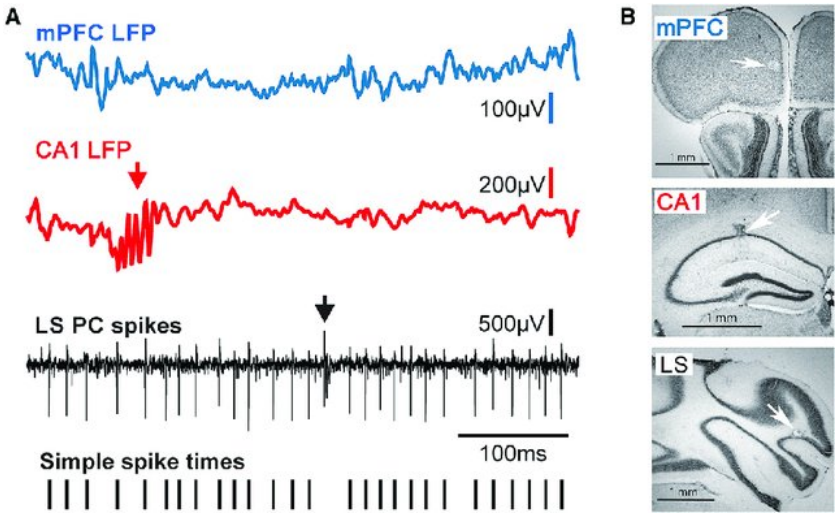

Figure 2.5. Displaying raw local-filed potential (LFP) and single-unit spike signals recorded from head-fixed mice using micro-needle electrodes (McAfee et al., 2019).

2.1.5 Spike and Local Field Potential (LFP)

The local field potential (LFP) is a signal produced by the summation of fields generated by excitatory and inhibitory dendritic potentials from a large number of neurons lying close to a recording location. In general, LFPs are used to refer to synchronous occurrences within neuronal populations (mostly at frequencies less than 300 Hz). LFPs are generated primarily by synaptic potentials, and are also the primary source of ECoG, MEG, and EEG waveform. Other soma-dendritic integrative processes may also contribute to LFPs, such as voltage-dependent membrane oscillations and after-potentials following soma-dendritic spikes. The LFPs and their many band-limited components are inextricably linked to cortical processing. Also, the LFP activity in the gamma band is also associated with almost all neurons' local spiking activity. Figure 2.5 illustrate a raw LFP recording (upper two traces) and single-unit spikes recorded from microelectrodes in the cerebellar lobulus simplex (LS) and Crus I and LFP activity in the medial prefrontal cortex (mPFC) and dorsal hippocampus CA1 region (dCA1) of mice. (McAfee et al., 2019).

Recording spikes and local field potentials have typically relied on the use of invasive microelectrodes implanted into areas of interest. Recording multi-unit spiking activity can be used to achieve multi-dimensional control for BCI applications by

analysing the neurons' firing rate and patterns of activity largely. However, this approach has been restricted to animal experiments because of the complex invasive procedures and a lack of electrodes that consistently yield long-term and stable recordings when used in human subjects.

2.2 Functional Target Potentials for BCI Usage

2.2.1 Substituting Lost Communication

Brain-computer and machine interfacing is increasingly playing an essential role in serving and augmenting medical services for those with high levels of disability by offering a new approach to communication. Present BCIs can be used for controlling electronic switches, replying to yes or no questions, providing simple word processing and computer cursor movement (Pandarinath et al., 2017; Mainsah et al., 2015).

When developing technology for communication assistance, there are options that do not require neural signals, even though such communication may be accomplished via brain control. Individuals who maintain control of one or a few muscles may communicate using this method on a regular basis. For instance, the electric activity of eyebrows, diaphragm, or finger muscles may be utilised to create a second control channel that is potentially quicker and provides better accuracy than current EEG-based BCIs (He et al., 2019). BCIs, on the other hand, may be considered mainly for those who lack complete muscular control or whose continued control is quickly exhausted. These individuals consist of those who are virtually paralysed but retain cognitive function, such as those with amyotrophic lateral sclerosis (ALS) or locked-in syndromes, as well as those who have movement problems that impair necessary muscular control, such as those with severe cerebral palsy. While individuals suffering from these diseases could have lost their capability of regulating their movement, their cognitive function can remain undamaged. As a result, people may be able to converse via a BCI. In certain instances, conventional communication methods focused on muscle movement (Kaur, 2021) may be more advantageous for these isolated individuals. Thus, even the most basic BCI-based communication capability, such as the capacity to answer yes or no, may be priceless.

Until recently, the majority of BCI research has been conducted on healthy volunteers. Several pieces of study (Lazarou et al., 2018; Kübler, 2020) have determined the potential of communication based on BCIs in specific individuals in a laboratory, hospital, or even at home. Numerous obstacles remain in bringing present BCI-based communication technologies into successful usage by severely handicapped individuals. To begin, diseases that impair voluntary muscular function may impair user control over the signal topographies utilised by BCIs. For instance, ALS can result in cortical neuron degeneration, impairing the generation or regulation of sensorimotor oscillations or even evoked potentials utilised in communication based on BCIs. As a result, it may be necessary to create a variety of BCI systems based on a variety of accessible neural signals in order to provide more treatment options for the wide variety of brain injuries. Additionally, impairment to the prefrontal cortex, such as Parkinson's disease, multiple sclerosis, or amyotrophic lateral sclerosis, may impair attention and cause indifferent BCI use. Similarly, a long training period may be troublesome for specific users (Bai et al., 2020). As a result, BCI systems with a short learning curve, such as those based on SSVEP, may be more appropriate.

2.2.2 Substituting Lost Motor Function and Supporting Plasticity

Currently, the most significant potential instructions of control in BCI development have been reached via the advancement of neuroprostheses for the restoration of motor function. Multi-dimensional and single-step control of a robotic arm or a virtual item is state-of-the-art in movement control. (Li et al., 2018b). BCI-controlled natural self-feeding has been achieved in animals utilising intracranial recordings and a robotic arm with four degrees of freedom (Velliste et al., 2008; Pan et al., 2011). Sensorimotor rhythms have been used to accomplish three-dimensional control of a computer cursor (McFarland et al., 2010) or continuous control of a virtual helicopter in real time (Doud et al., 2011; Royer et al., 2010) using non-invasive EEG. Also, consecutive interpretation of three-dimensional movement pathways based on EEG has been introduced (Bradberry et al., 2010). Their implementations may eventually include wheelchair control, vehicle operation, dexterous finger control, and a variety of other tasks performed by robots. Nonetheless, this study operated a robotic arm or a virtual

item in the majority of instances. This extra motor function may be beneficial for individuals who are paralysed to varying degrees. Numerous individuals have a permanent loss of motor function. Thus, a neuroprosthesis enables restoration of functional motor control in place of the conventional. While traditional methods based on restricted muscle activity may also offer similar capability, neuroprostheses based on BCIs may give personalised prosthetic control that is organically connected to the user's objective. Once users want to raise their arms, for example, they may do so by connecting to the BCI and driving their arms based on intention detection alone.

Another potential use of BCI tools is to aid in rehabilitation and promote neuroplasticity in order to reclaim a lost level of function. Numerous studies have revealed that training for and usage of BCIs may result in modifications of brain activity that enable using prosthetic devices, mainly applied with functional support such as electric stimulation (FES) (Moritz et al., 2008; Tam et al., 2011). These learning-related evolutions are essential for individuals who have sustained a brain injury, such as a stroke but yet to regain some motor function. In research utilising MEG recording, individuals with persistent hand hemiplegia were able to control their sensorimotor rhythms after a stroke effectively. Likewise, they were able to control a prosthetic or orthotic device by imaging the opening and closing of their paralysed hands (Buch et al., 2008). A comparison of the early and late training stages revealed that the ipsilesional hemisphere had improved sensorimotor rhythms, allowing the hemisphere to interface with the gadget's controller. Numerous randomised controlled trials have shown that supporting movement with FES linked to a BCI system may significantly improve upper-limb function in individuals with mild to severe upper-limb impairment (Alon et al., 2003; Ring and Rosenthal, 2005) or severe stroke (Daly et al., 2005). According to studies with invasive and non-invasive BCIs, learning-related changes can arise over days to months (Royer et al., 2011). Surprisingly, once users are trained via a BCI to manipulate a neuroprosthesis, they hold this capacity months later without having to override use (Wolpaw and McFarland, 2004) suggesting a longstanding learning-related modification in neural circuits. As a result, a BCI may be utilised to promote favourable neuroplasticity in neuromuscular circuits to restore motor function years after the onset of a disease.

2.2.3 Augmenting Normal Functions

BCIs are often thought of as devices to be used to help those with functional disability. However, they can also be utilised to enhance normal neuromuscular function. One potential application is to support remote navigation using BCI. One such application is controlling a computer cursor that aims to help disabled people to increase control of external devices and aid healthy people to issue external commands without using normal neuromuscular means. Much research has discovered competency in completing navigation tasks in virtual environments, such as for control of a computer cursor (Wolpaw and McFarland, 2004; McFarland et al., 2010), walking in a virtual world (Pfurtscheller et al., 2006), and real-time control of an aircraft in a 3D virtual environment (Doud et al., 2011; Royer et al., 2010).

A limitation of utilising BCI technology to augment normal neuromuscular function is the slower pace of information transmission (ITR) than conventional muscle control. To accomplish that job, a healthy individual will choose manual keying over BCI practice. On the other hand, BCI technology controls may run into situations when a high data transmission rate is unnecessary, and non-muscular control is required.

2.2.4 Neurofeedback

BCI technology has been used in a variety of fields, including traditional communication, motor rehabilitation, environmental control, mobility, and entertainment. The potential of BCI usage and training to induce brain plasticity may serve as a foundation for new medical fields. Users may use neurofeedback to gain discriminating control over certain brain regions in order to create behavioural changes in the brain. BCI-based neurofeedback has been shown to improve cognitive function (Angelakis et al., 2007; Hanslmayr et al., 2005), speech skills (Rota et al., 2009), affection (Sitaram et al., 2011), and pain management (deCharms et al., 2005). Additionally, it has been used to treat mental illnesses such as epilepsy (Walker and KozlowSki, 2005; Sterman and Egner, 2006), attention deficit (Strehl et al., 2006), schizophrenia (Schneider et al., 1992b), depression (Schneider et al., 1992a), alcohol dependence (Schneider et al., 1993), and paedophilia (Renaud et al., 2011).

Alternatively, brain signal recordings may be used to assess the state of the brain's activities in health and illness (Georgopoulos et al., 2007).

2.2.5 Others BCI Applications

Additionally, the opportunity to monitor brain impulses may be commercialised. Neuromarketing is a relatively new field of study in which neuroscientific methods are used in marketing research. Thus far, little neuromarketing research has been conducted, although some findings indicate that neuroimaging may play a significant role in a variety of marketing disciplines (Cook et al., 2011; Ambler et al., 2000; Vecchiato et al., 2010; Ambler et al., 2004). Neuromarketing may provide a more cost-effective trade-off between expenses and earnings. Neuromarketing may be used to evaluate conceptual product design, allowing for the elimination of unfavourable features early in the production process. This would result in a more well-organized source distribution since only the most promising products would be settled (Ariely and Berns, 2010). Similarly, instead of relying on data from traditional market research, neuromarketing may provide more accurate information on users' primary tendencies (Ariely and Berns, 2010). Neuroimaging may provide confidential information about customers' actual preferences that cannot be communicated clearly. The reaction of the brain to commercial advertising may be examined, as well as the effectiveness of marketing efforts.

2.3 Research Trends in BCIs

2.3.1 BCIs Based on Sensorimotor Rhythms

Pfurtscheller and colleagues (Pfurtscheller et al., 1993) produced an example of multidimensional BCI utilising mu-rhythm EEG from the sensorimotor brain. The obtained EEG signals were bandpass filtered to focus on the mu frequency (8-12 Hz) and then squared to obtain the instantaneous mu power. Five consecutive mu-power estimations obtained during ERD were used to generate a five-dimensional feature vector. The vector was categorised using a one-nearest neighbour (1-NN) classifier utilising reference vectors produced using the learning vector quantisation (LVQ)

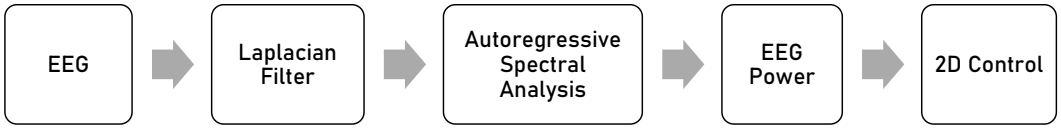

Figure 2.6. Presenting the processing pipeline of Wolpanw's work (Wolpaw and McFarland, 2004; McFarland et al., 2010). The autoregrssive spectrak analysis was used for determining the feature vector which includes the EEG power of mu-rhythm and beta-rhythm.

technique. The LVQ approach is a vector quantisation technique that partitions the high-dimensional input space into areas, each of which has an associated reference vector and class label. Unknown input vectors are categorised by associating them with the class label of the reference vector to which they are most similar in terms of feature translation.

In addition, a modern BCI system has been created that enables users to operate a three-dimensional computer cursor (Wolpaw, 2010). The EEG was collected from users who accomplished cursor management by actively manipulating mu and beta rhythm power from one or more electrode sites overlying the sensorimotor cortex. An autoregressive method was used to determine the EEG power spectra to produce the feature vector yielding multidimensional control (Wolpaw and McFarland, 2004; McFarland et al., 2010). The processing pipeline of this work is illustrated in Figure 2.6, in which the raw EEG signal was initially filtered using Laplacian filtering. In the study of subjects trained to use the system, Wolpow demonstrated performance in terms of speed and precision equivalent to that achieved with implemented microelectrode arrays in BCI systems (Wolpaw, 2010).

Doud and colleagues (Doud et al., 2011; Royer et al., 2010) explored the feasibility of employing sensorimotor rhythms to create a BCI system for controlling continuous navigation in a three-dimensional virtual environment. Control signals generated from motor imagery tasks and advanced classification methods were utilised to enhance the performance of navigation. Three-dimensional navigation was reduced to two dimensions, enabling people to fly a virtual helicopter in any direction while maintaining a constant forward flight velocity. (Royer et al., 2010). Additional research has shown that people are capable of controlling a virtual helicopter in three dimensions in a rapid, accurate, and continuous manner (Doud et al., 2011). The

technology modulated sensorimotor rhythms to trigger the helicopter's forward-backwards direction and height controls. At each simulation time step, these instructions were translated to mechanical motor forces generated by the helicopter's mechanical motor. The angle of the helicopter was linearly mapped with more precision using sensorimotor oscillations linked with other motor related intention. These numerous control choices enable interaction between coarse and fine control, as shown by the gross and fine arm and hand motions. As a result, participants flew the helicopter to random positions and orientations in three dimensions, to which the system responded every 30 milliseconds.

2.3.2 BCIs Based on P300

In 1988, Farwell and Donchin explored a BCI based on P300 (Farwell and Donchin, 1988). The P300-based BCI paradigm has grown to become one of the most utilised BCI modalities, usually being utilised in spelling or communication applications. In the ERP, the P300 is a positive deflection with a 200-700 milliseconds delay after stimulus onset. The response is activated when a subject attends to a series of stimulus actions containing an intermittently displayed target called the oddball event. Typically, the response of P300 is assessed across the central-parietal region (Donchin, 1981). The majority of P300-BCIs make use of a visual stimulus. However, systems using auditory stimuli have also been investigated.

A row and column paradigm (RCP) strategy has been used in several P300-based applications with the visual P300 ERP (Farwell and Donchin, 1988; Donchin et al., 2000). Here, a matrix comprising numbers, alphabets, and other perceptible items is displayed to users in rows and columns, as shown in Figure 2.7. The subject examines the desired object or letter and then counts the flashes of the row and column. Due to the fact that the P300 can be only detectable in responses produced by the target stimulus, the computer can recognise the rows and columns that elicit a P300 response after an appropriate number of repetitions. Accordingly, the user will recognise the item at the intersection of this row and column as the requested item.

The P300, which is based on RCP, has several intrinsic causes of inaccuracy. A target item that is next to it (flashing in the same row or column) is chosen erroneously

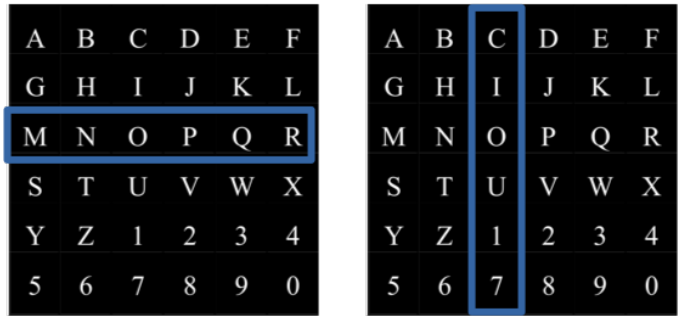

Figure 2.7. A traditional row and column paradigm in which rows and columns are randomly flashed. The target character is only unique at the intersection of the target row and column. If the target or intended character is O, then theoretically, only the flashing of row 3 and column 3 should elicit the P300 response.

more often than other non-target items. Additionally, after the row and column of the target item flash in sequence, the magnitude of the second flash may be decreased, or the morphology of the second flash may change. Therefore, another P300 paradigm named the checkerboard paradigm has been proposed by Townsend (Townsend et al., 2010), where an 8 x 9 matrix was virtually overlaid on a checkerboard. In these two matrices, the item locations were randomly organised. During a real-time operation, each stimulus series comprised six rows in the white matrix flashing sequentially in vertical, followed by six rows in the black matrix flashing sequentially in horizontal. The locations of the elements in each matrix were re-randomised for the subsequent trial after each series of stimuli. A subject observes the flashing of a virtual row or column of randomly dispersed objects.

Using the checkerboard paradigm, the problem of double-flashing is eliminated since objects are not able to visualise at least six intervening flashes. Additionally, adjacency problems are avoided in most cases since a neighbouring object cannot be put in the same flash group. According to preliminary findings, the checkerboard paradigm is quicker, more precise, and more consistent than the standard RCP. Additionally, users of BCIs are likely to find it more acceptable in usage. Jin has suggested a new P300 paradigm for reducing adjacency and double-flash mistakes (Jin et al., 2011).

Research studies also reveal that gaze change may be necessary for conventional P300 speller performance (Treder and Blankertz, 2010; Brunner et al., 2010b), and the ability to control eye gaze is, therefore, an essential requirement for successful use.

For subjects without gaze control, a gaze-independent visual P300-based BCI has been developed by Liu (Liu et al., 2011). It is distinguished from the standard RCP-P300 speller by the fact that each column or row of a 6 x 6 matrix is grouped and displayed over the location of the gaze point within a narrow near-central visual area. Their stimulus includes two phases: an image phase containing six characters in a circle and an interval phase devoid of characters. A single critical stimulus series contains 12 interval phases and 12 image phases (corresponding to the 6 x 6 matrix in a traditional P300 speller). In the image phase, subjects are required to focus on the circle's centre and memorise the target character among grouped non-target symbols. According to the results of their study, the average accuracy recorded by eight healthy subjects was more than 90%, and the speed was approximately one character per minute. Consequently, this approach could be helpful for users with a limited gaze.

In BCI systems, the P300-BCI is one of the most regularly used and one of the few BCIs that have been examined in the severely paralysed subjects (Sellers et al., 2010). Several BCI studies focused on increasing system performance, including speed, accuracy, consistency, and user comfort. Hong suggested a novel N200-speller BCI that incorporates a motion-onset visual ERP component (Hong et al., 2009). This system decreases the discomfort of bright visual stimuli as it operates at a lower luminance and contrast thresholds than used in P300 based system.

2.3.3 BCIs Based on Visual Evoked Potentials

Numerous studies (Wang et al., 2006b; Bin et al., 2009a) have been conducted on non-invasive BCI devices based on visual evoked potentials (VEPs). Sensory stimulation of a subject's visual field initiates the recording of VEPs in the occipital regions. VEPs are thought to be a reflection of visual patterns in the brain. Central visual field stimulation produces larger potentials than peripheral stimulation. BCIs based on VEP determine the target on which a user is concentrating through investigation of simultaneously recorded EEG. An individual target is wrapped by a single stimulus sequence, which leads to a distinct VEP signal. To enable accurate classification, VEPs generated from distinct stimulus sequences should be orthogonal or nearly orthogonal in some transform domain, e.g., the frequency domain.

The stimulus sequence design is critical for an SSVEP-based BCI. SSVEP-based BCIs now available may be classified into four types based on the stimulus sequence employed, including frequency-modulated VEP (f-VEP) BCIs (Middendorf et al., 2000; Cheng et al., 2002; Gao et al., 2003; Bin et al., 2009b); time-modulated VEP (t-VEP) BCIs (Guo et al., 2008; Lee et al., 2006; Lee et al., 2008); code-modulated VEP (c-VEP) BCIs (Sutter, 1992; Sutter, 1984; Momose, 2007; Bin et al., 2011); and phase-modulated VEPBCIs (p-VEP) (Kluge and Hartmann, 2007; Wilson and Palaniappan, 2009; Jia et al., 2011; Pan et al., 2011).

In a frequency-modulated (f-VEP) BCI, each target flashes at a unique frequency (Bin et al., 2009a). This results in a periodic visual evoked response with identical fundamental and harmonic frequencies as the flashing stimulus. The evoked reactions to successive flashes of the target overlap since the flicker frequency of f-VEP BCIs is regularly more significant than 6 Hz. This results in a periodic sequence of VEPs, or constant-state visual evoked potentials, that is frequency locked to the flickering target. Intrinsically, the f-VEP approach is frequently designated as SSVEP BCIs. Power spectral analysis may be used to identify targets. Numerous laboratory and clinical experiments (Won et al., 2015; İşcan and Nikulin, 2018) have been conducted over the last decades to evaluate the capacity of f-VEP BCIs. The benefits of an f-VEP BCI include ease of system setup, little user training, and a high data transmission rate.

The flash sequences of different targets are similarly self-regulating in time-modulated VEP (t-VEP) BCIs. This may be accomplished by demanding that flash sequences for distinct targets be strictly non-overlapping or by randomising the length of the ON and OFF states for each target. Visual evoked potentials are activated with short latencies and durations by the flashing stimuli. In the t-VEP system, a synchronised signal must be supplied to the EEG amplifier to identify the flash onset of each target. Following the beginning of a visual stimulus, t-VEPs are both time- and phase-locked. Thus, averaging across many brief epochs synchronised according to the flash target onset time will generate VEPs for each potential target since all targets are mutually independent. Because a foveal VEP is more significant than a peripheral VEP, the target generating the most outstanding average peak-to-valley VEP magnitude may be recognised as the fixation target. Averaging across several epochs

is required for precise target recognition in a t-VEP BCI. Likewise, t-VEP BCIs typically have low stimulation rates to avoid the overlap of two successive VEPs, and thus have a reasonably poor information transmission rate.

In opposition, pseudorandom stimulus sequences applied to code-modulated (c-VEP) BCI have been proposed. In c-VEP BCIs, the M-sequence is one of the most customarily used techniques for a pseudorandom sequence. M-sequences include autocorrelation functions that roughly resemble a unit impulse function and are almost orthogonal to its time lag sequence. Thus, an M-sequence and its time lag sequence may be utilised in c-VEP BCIs to target a variety of diverse stimuli. An asynchronous signal providing a trigger for target identification is provided to the EEG amplifier at the beginning of each stimulation cycle. To identify the target, the template matching technique is commonly employed. In 1984, Sutter (Sutter, 1984) developed a BCI system based on c-VEP. Bin (Bin et al., 2011) explained and validated a c-VEP BCI system based on a personal computer. The result was an average ITR of 108 bits/min, and with one subject achieving the highest rate of 123 bits/min.

Multiple targets flicker at a similar frequency with different phases in a phase-modulated VEP (p-VEP) BCI, allowing for the presentation of more targets in less time. Recently, a coding method has been proposed to group frequency and phase information (Jia et al., 2011). They built their system using this approach, using 15 targets and just three stimulation frequencies. The average ITR achieved 60 bits/min during an online stimulation after optimising lead location, reference phase, data segment length, and harmonic components.

The summary of the advantages and disadvantages of BCIs based on VEP can be found in the literature (Wang et al., 2006b; Bin et al., 2009a), which can be described as simplicity, low training time, and high information transfer rate. With these advantages, VEP based BCI has been increasingly used as the basis for communication aids. In comparison, the disadvantages include the need for adequate gaze control (which may be absent in individuals with severe neuromuscular impairments) and visual fatigue caused by prolonged fixation.

Subjects change their gaze location in conventional SSVEP BCIs to focus on one of the target stimuli. The target stimulus elicits more substantial SSVEP responses at

the matched frequency throughout occipital brain regions. Correspondingly, this device is classified as a dependent BCI due to the possibility that muscular action such as gaze generation is required. As a result, it may be ineffective in the case of individuals who have lost control of their gaze direction.

2.3.4 BCIs Based on Auditory Potentials

As discussed earlier, BCIs using visual stimuli have been shown to be efficient. Still, specific individuals with severe disabilities may have difficulties utilising such BCIs owing to the need for enduring vision or lack of eye movement control. However, even with severe brain degeneration diseases such as amyotrophic lateral sclerosis (ALS), hearing is generally well-maintained. As a result, an auditory evoked potential-based BCI paradigm (AEP-BCI) may be utilised.

Auditory evoked potentials (AEPs) used in BCI related applications can be categorised into two groups. The first type employs auditory stimuli as feedback to aid subjects in controlling their sensorimotor rhythms (Nijboer et al., 2008) or controlling their slow waves (Hinterberger et al., 2004a; Pham et al., 2005). Another type employs an oddball auditory paradigm (Hill et al., 2004; Sellers and Donchin, 2006; Furdea et al., 2009; Guo et al., 2010) and is the most common form of AEP-based BCI (Furdea et al., 2009; Guo et al., 2010). As with the visual P300, auditory stimuli in strange auditory BCIs are classified as often shown non-targets and rarely presented targets. For example, spoken numbers may be composed of a sequence of stimuli. The numbers would be presented at random and would correspond to the possible choices. Except for one target stimulus in the sequence (the user's preferred option), all the numbers would be standard non-target stimuli. After the target number is spoken, the individual is taught to focus only on it and perform a mental activity (counting each time when heard). Auditory ERPs react to hearing the target stimuli in a manner similar to visual P300-based BCIs. In 2009, Furdea proposed an auditory spelling system (Furdea et al., 2009) and another was studied in ALS patients (Kubler et al., 2009). To compare performance across visual and auditory potentials, a 5x5 visual matrix with numbers 1-5 in rows and 6-10 in columns was given to participants. Rather than using flashes, spoken numbers were used in lieu of the standard visual P300 speller. The

participants were trained to choose the row number first, followed by the column number, in order to equalise the target letter in a visual P300 speller. This auditory system was first verified in healthy subjects, achieving an average accuracy of above 70% (Furdea et al., 2009). In another study with ALS patients conducted by Kubler (Kubler et al., 2009), the users utilised the device and obtained a survival rate of more than 50%.

Guo proposed another auditory BCI system using no visual display that pronounced numbers 1–8 representing eight possible choices, and the participants completed a mental task (Guo et al., 2010). One of the numbers represents the target associated with the subject's intended selection during operation. Rather than simple silent counting, this paradigm required identifying the target number's laterality (left or right) or gender voice (male or female). The AEP's N2 component (latency 100–300 milliseconds) and late positive component (LPC) (delay 400–700 milliseconds) were utilised to recognise the target ERP's spatiotemporal pattern. The N2 component reflects the negativity in auditory processing that is amplified when endogenous attention is used intentionally, while the comprehensive LPC component represents memory-updating processes. Hence, N2 and LPC were chosen as prominent indicators of the reaction of the brain to the target. As a consequence, the task enhances the amplitude and duration of the LPC considerably (Guo et al., 2010).

The auditory speller performed worse than the visual spelling system. Additionally, the auditory ERPs' peak latencies were longer. Nonetheless, AEP-based BCIs may be a preferable method of communicating externally for severely handicapped individuals with impaired eyesight or lack of eye movement control and are therefore deserving of further investigation.

2.3.5 Attention-Based BCIs

Several neurophysiological studies reveal that individuals may surreptitiously shift their attention to different spatial places without rerouting their gaze (Posner and Petersen, 1990; Posner and Dehaene, 1994). Furthermore, changing concentration to one of the various overlaid objects can increase behavioural competency (accuracy and reaction time) and improve neuronal reactions compared to patterns in which the object

is not attended (Desimone and Duncan, 1995). In 2005, a BCI based on spatial visual selective attention was reported by Kelly (Kelly et al., 2005a; Kelly et al., 2005b). Two bilateral flickers presented the subjects with overlaid letter series. To select the target, the subjects covertly focused on one of the two bilateral flickers. This system yielded an average accuracy of more than 70%. In another study, a non-spatial visual selective attention-based BCI, Zhang (Zhang et al., 2010) presented two sets of dots of varying colours, transparent surfaces, and flicker frequencies spinning in opposing directions to elicit two overlaid sensations. They elicited noticeable SSVEPs because of surfaces flickering at different frequencies. The amplitude of SSVEP at a consistent frequency was boosted by selectively attending to one of the two surfaces. As a result, participants may choose between two distinct BCI outputs. This approach was evaluated in healthy participants using an online training programme over a three-day period. On the last training day, more than 70% was reached.

BCIs based on selective visual attention have to date only offered a binary control. Nonetheless, their good performance with gaze independence warrants additional investigation in order to create a system that allows multiple selections. This may be a viable technique for paralysed individuals who have difficulty regulating their gaze direction. Rather than altering their gaze direction, it may assist patients in controlling BCIs via covert attention shifts.

2.4 Comparison of BCI Performance

This section aims to provide a comparison of BCI performance based on different perspectives, including control signal and novelty. In addition, the advantages and disadvantages of using each category are also listed.

2.4.1 Comparison Based on Types of Control Signal

A primary goal of a general BCI is to translate user intentions by using signals representing cerebral activity associated with cognitive tasks. While the neural coding underlying cognition is still unresolved to create BCIs that interpret user intentions, physiological phenomena that correlate with decision making and execution that

Table 2.2. Summary of different control signals in BCI.

Signal	Physiological Phenomena	Number of Choices	Training Needed	Information Transfer Rate
SCP	Brain signals undergo gradual voltage shifts.	Low	Yes	5-12 bits/min
VEP	Modulations of brain signals in the visual cortex.	High	No	60-100 bits/min
P300	Positive peaks as a result of a rare stimulus.	High	No	20-25 bits/min
SMR	Modulations in sensorimotor rhythms synchronized to motor activities.	Low	Yes	3-35 bits/min

people can modulate at will can be used. These recoverable signals are considered potential control signals in BCIs.

A growing body of research has explored an enormous variety of brainwaves that might function as control signals in BCI systems. However, only those control signals that are currently used in BCI systems will be addressed here, including slow cortical potentials (SCPs), visual evoked potentials (VEPs), P300 evoked potentials (P300s), and sensorimotor rhythms (SMRs). In Table 2.2, all of the available signal controllers are listed, along with some of their primary characteristics. (Nicolas-Alonso and Gomez-Gil, 2012).

2.4.2 Comparison Based on Types of Novelty

The types of BCIs based on their novelty can be divided into (1) exogenous or endogenous and (2) synchronous (cue-paced) or asynchronous (self-paced). The various kinds of BCI are listed in Tables 2.3 and 2.4, together with information on the brain signals that can be changed to transmit information and their related benefits and drawbacks. Similarly, BCIs may be classified as dependent or independent. (Wolpaw et al., 2002).

BCIs can be considered exogenous or endogenous with respect to the input signal. Exogenous BCI employs the activity of neurons evoked in the brain by an external stimulus, for example, VEPs and auditory evoked potentials (AEPs) (Kleber and Birbaumer, 2005). Exogenous BCIs do not need much training since their control signals (SSVEPs and P300) are clearly defined and very simple to record using supplementary stimulation setups. Only one well-placed EEG channel may be needed

Table 2.3. Differences between synchronous and asynchronous BCI.

Approach	Advantage	Disadvantage
Synchronous BCI	Design and performance assessment are simplified. The user may prevent artefact generation while still performing blinks and other eye movements while brain signals are not being analysed.	Does not provide a more natural means of communication.
Asynchronous BCI	No external cues are required. Provides a more natural way of interaction.	Design is more complicated. Evaluation is more challenging.

Table 2.4. Differences between exogenous and endogenous BCI.

Approach	Brain Signal	Advantage	Disadvantage
Exogenous BCI	SSVEP P300	Minimal training. Setup of the control signal is simple and fast. High bit rate (60 bits/min). There is no need for more than one EEG channel.	Constant vigilance towards external inputs. Some users may experience fatigue.
Endogenous BCI	SCP SMR	Independent of external stimuli. Can be run on an ad hoc basis. Helpful for impaired sensory organ users. Appropriate for applications requiring cursor control.	Training is very time intensive (months/weeks). Not every user can regain control. EEG recordings on several channels are needed for optimal performance. Lower bit rate (10-30 bits/min).

as the signal source, and a high information transfer rate (ITR) of up to 60 bits/min can be achieved. Conversely, endogenous BCIs utilise localised modulation of brain oscillations or potentials without attention to external stimuli (Kleber and Birbaumer, 2005). Here, users learn to produce particular brain signal patterns through neurofeedback training, such as modulations in the SMRs (Krusienski et al., 2007) or the SCPs (Hinterberger et al., 2004b). The benefit of endogenous BCIs is that the user has voluntary control over the BCI, and in the most successful example, can achieve multidimensional control.

In comparison with endogenous BCIs, an exogenous BCI may limit the user to the selections offered and may require extended user training to achieve a satisfactory level of performance. Similarly, endogenous BCIs, while beneficial for users with advanced stages of ALS, require user attention to a specific tool and can also be

cognitively fatiguing. The differences between exogenous and endogenous BCIs are summarised in Table 2.3.

As stated by the novelty of the BCI system, it can also be categorised as synchronous or asynchronous. Synchronous BCIs examine brainwaves across predefined time intervals. Any EEG wave that occurs outside the predefined gap is ignored. As a result, users may only direct their instructions within predefined times monitored by the BCI systems. For example, the Graz BCI is an example of a synchronous BCI systems (Pfurtscheller et al., 2003). The benefit of synchronous BCIs is that the onset of mental activity may be predicted and associated with a specific trigger (Tsui and Gan, 2007). Additionally, patients may blink and change gaze during rest, creating artefacts and is one reason why BCIs do not always continuously examine brain signals as errors increase with incidence of artefacts. This takes us back to the first principles of the design and assessment of synchronous BCI. Yet, asynchronous BCIs that do continuously examine brain signals offer a more natural approach to human-machine communication than synchronous BCIs but they are more challenging to implement and require more computational complexity, including real time processing and fast pattern recognition. The discussion between synchronous and asynchronous BCIs is summarised in Table 2.4.

2.5 Real-Time EEG Processing

Since the present thesis focuses on using non-invasive recording techniques, related background and literature are provided on this topic. Although some basic principles of real-time signal processing were introduced in the previous chapter, this section explains this further by examining the essential parts of real-time EEG recordings, covering state-of-the-art processing tools and techniques.

2.5.1 Signal Acquisition

2.5.1.1 Non-Invasive EEG Electrodes

There are three typical types of electrodes used in EEG recording, including wet electrodes, dry electrodes, and non-contact electrodes, as shown in Figure 2.8. The details of each type are described as follows:

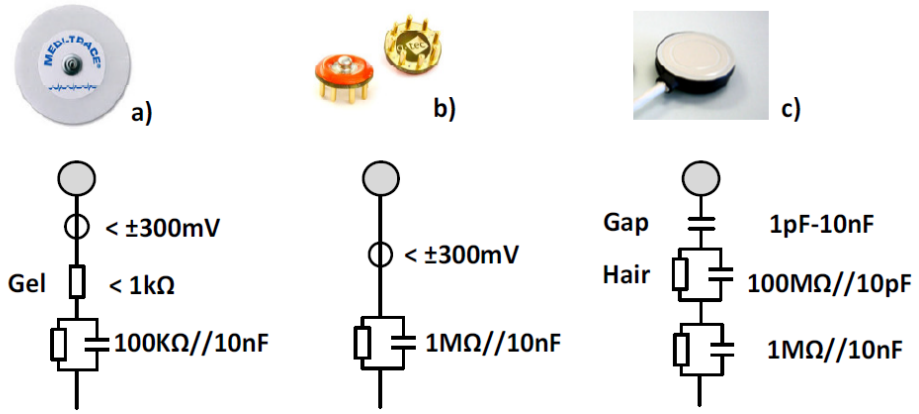

Figure 2.8. Electrical equivalents of various electrode-tissue interactions. a) wet/gel electrode based on Ag/AgCl, b) dry contact electrode developed by g.tec, c) capacitive (dry non-contact) electrode developed by QUASAR. Adapted from (Xu et al., 2017).

Wet Electrodes: Several pieces of research and clinical EEG studies use this kind of electrode, either set up as passive-wet electrodes or active-wet electrodes (Laszlo et al., 2014). Passive electrodes connect directly to a preamplifier, while active electrodes have an integrated preamplifier and signal conditioning. The use of active electrodes can offer reduced noise interference, by moderating and minimising impedance levels. Thus, it can achieve better signal quality than passive electrodes. However, care is needed in considering active electrodes to ensure the total bandwidth of signals can be recovered. This is particularly important in ERP studies where high-frequency components exist in the signals (Mathewson et al., 2017). An inherent requirement for both passive and active wet electrodes is the use of a conducting gel as an interface between the user's skin and the material of the electrode. Gel performance is, therefore, a consideration with these electrodes, as drying out of the gel over time degrades the recording quality.

Dry Electrodes: Active-dry electrodes have been designed to eliminate the need for gel, reduce setup times and may allow real-world usage by untrained uses or carers. However, current dry electrode designs operate at high impedance and as such increase the potential for receiving more interference than wet electrodes. This problem may obscure ERP information and result in decreased statistical power (Im and Seo, 2016).

Non-Contact Electrodes: Non-contact electrodes are capacitive in nature and thus do not need an ohmic connection to the body. They are capable of recording EEG data and are safe to use in case of skin disease or allergies. Using such electrodes may lower

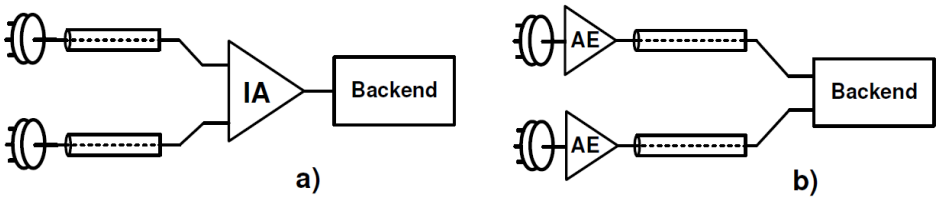

Figure 2.9. A simplified block schematic of a) a traditional EEG readout using an instrumentation amplifier (IA); b) active electrode (AE) for EEG recording. Adapted from (Xu et al., 2017).

the preparation time for EEG data acquisition but result in a signal with a modest amplitude and a high sensitivity to motion artefacts (Chi and Cauwenberghs, 2010).

Considering the use of each electrode type, a traditional wet electrode is considered as the gold standard (Lopez-Gordo et al., 2014) despite requiring skin preparation and use of conductive gel to improve the electrode-scalp interface. This procedure is time-consuming, especially when many electrodes are used. It seems inappropriate for everyday long-term use. By contrast, it is preferable to utilise a dry EEG electrode. Numerous dry electroencephalogram electrodes that do not require gel have been suggested in the literature (Sullivan et al., 2008; Lin et al., 2011; Liao et al., 2011; Gargiulo et al., 2010; Estepp et al., 2009). They can record the scalp EEG with less preparation. Also, dry electrodes have been investigated for alpha-beta rhythms (Harland et al., 2002), steady-state visual evoked potentials (SSVEPs) (Lin et al., 2013; Oehler et al., 2008), and auditory event-related potentials (ERPs) paradigms (Callan et al., 2015). However, there are still some performance drawbacks. In contrast to contemporary dry EEG electrodes, the original wet electrodes offer much higher signal quality while recording.

2.5.1.2 Amplifier Architectures

Given the small amplitude of EEG signals, care is needed in choosing the specification of the bio-amplifier to be used with different electrode types. The overview of the bio-amplifier architecture is presented in Figure 2.9.

An active electrode (AE) with an integrated preamplifier decreases noise susceptibly created by cable motion artefacts by creating a low impedance output that also removes the requirement for shielded cables (Xu et al., 2017). However, set-up flexibility can be lost as preamplifiers are optimised to the electrodes at manufacture,

Table 2.5. The internationally medical standards for EEG recording equipment and general specifications for current wearable EEG applications.

	IEC60601[a]	IFCN[b]	General Specifications
Applications	Clinical EEG	Clinical EEG	Wearable EEG
Input voltage	0.5 mV_{pp}	-	1 mV_{pp}
Input referred noise (per channel)	6 μV_{pp}	0.5 μV_{pp} (0.5-100 Hz)	1 μV_{pp} (0.5-100 Hz)
Bandwidth	0.5-50 Hz	0.16-70 Hz	0.5-100 Hz
Input impedance	-	>100 MΩ	>100 MΩ
CMRR	-	110 dB	80 dB
Safe DC current	50 μA	-	50 μA
Number of wires	-	-	Minimal

[a]IEC60601: the International Electrotechnical Commission's standards for the safety and efficacy of medical electrical equipment.
[b]IFCN: the standards of International Federation of Clinical Neurophysiology.

limiting the bandwidth and sampling capability of downstream signal conditioning. Selecting a suitable amplifier system is therefore critical in order to match the characteristics of the recoding system with the signal processing requirements of the BCI (Cencen et al., 2016).

2.5.1.3 International Standards of Electrical Performance

In clinical EEG recordings, the recording protocols and equipment have to be configured to meet accepted standards of practice and safety, including recommendations from the International Federation of Clinical Neurophysiology (IFCN) recommendations (Rossini et al., 2015); and technical standards for the safety and essential performance of medical electrical equipment, published by the International Electrotechnical Commission named IEC 60601 (Commission, 2015).

According to the standardisation, designing wearable EEG systems should be required to meet target product specifications including input voltage, input-referred noise, bandwidth, noise, input impedance, common-mode rejection ratio (CMRR) and the number and method of connection wires as presented in Table 2.5 (Xu et al., 2017).

2.5.1.4 Advances in EEG Headsets

Previously, the number of channels measured in EEG investigations typically did not exceed 64 channels and mode use of the extended international 10–20 system (Jasper,

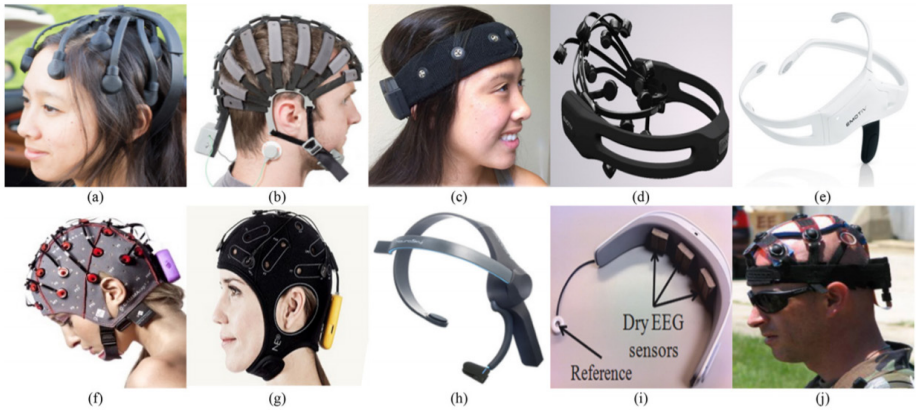

Figure 2.10. Different portable wireless EEG headsets. (a–c) Miniature dry-electrode wireless acquisition by Cognionics, (d and e) EPOC and Insight wireless EEG acquisition by Emotiv, (f) g.Nautilus wireless EEG acquisition by g.tec, (g) ENOBIO wireless EEG system by Neuroelectrics, (h) MindWave mobile EEG headset by NeuroSky, (i) wearable EEG headset, and (j) wireless EEG headset by Advanced Brain Monitoring. Adapted from (Minguillon et al., 2017; Chum et al., 2012).

1958). EEG studies over many years (Hjorth, 1975; Hellström et al., 1963; Hjorth, 1980) have explored brain functionality with this limited number of electrodes. However, with improved amplifiers offering high channel counts, new electrode standards such as 10–10 (Chatrian et al., 1985) and 10–5 (Oostenveld and Praamstra, 2001) extended EEG montages have allowed for high-density recording with 128–256 channels (Gwin et al., 2010; Pisarenco et al., 2014) to become commonplace.

With the new EEG montages, high-density EEG studies can benefit from use of spatial filtering and referencing techniques such as common average reference (CAR) for minimising interferences and identifying generators. Additionally, Laplacian montages, which also require high-density electrode recoding, have shown evidence of being highly efficient in artefact extraction and removal (Muthukumaraswamy, 2013) and eye movement (saccadic) spike potential reduction (Keren et al., 2010). This results in a better classification result compared to a low-density montage (Yacine et al., 2014). Yet the negative cost of preparation time may prevent the regular use of high-density EEG headsets in BCI application designed for daily-life activities. The criteria for a simple and fast electrical montage for everyday use are highlighted. Numerous electrode configurations have been included with contemporary EEG headsets, as shown in Figure 2.10.

2.5.2 Signal Processing Techniques

This part of the present thesis discusses current methods for signal processing commonly used in BCI study. In common, the standard processing pipeline includes signal pre-processing, electrode referencing, feature extraction and selection, and classification.

2.5.2.1 Signal Pre-Processing

Most digital signal processing applications use a filter in the pre-processing step. The main types of filters are low-pass filter, high-pass filter, band-pass filter, and band-stop. Finite impulse response (FIR) and infinite impulse response (IIR) are primary types of filtering implementation. Each filtering technique requires consideration when used because it has different properties and characteristics. The significant difference between the IIR and FIR implementations is that the IIR filter requires its filter output as input, which can be viewed as a recursive function. The advantage of using the IIR filter over the FIR filter is that it offers a lower order of filter to get a similar filter roll-off. Hence, fewer computations are required to achieve the same outcome. Although the IIR filter offers faster computations than the FIR filter, the major issues with the IIR are non-linear phase and stability (Litwin, 2000). It is a trade-off between computational performance and stability. In addition to using a digital filter, this step is mandatory for a real-world application containing signal acquisition, especially in a noisy environment.

According to the guidelines for pre-processing for large-scale EEG analysis (Bigdely-Shamlo et al., 2015), the objective is to provide adequate pre-processing to prepare the data for automated processing by standardising data statistics through collecting. The processing step, in general, is as follows:

1. Eliminate line noise without using a filtering technique.
2. Ensure that the signal is robustly referenced to an approximation of the proper average reference.
3. Identify and interpolate incorrect channels in relation to this reference.
4. Retain sufficient information to enable users to re-reference a channel using a different technique or to undo channel interpolation.

In general, applying a digital band-pass filter with a specific bandwidth compatible with the characteristics of desired signals, such as a Butterworth filter at 0.1-40 Hz used for sleep analysis (Lee et al., 2014; Liang et al., 2012); or using an adaptive notch filter to suppress power line noise (Ferdjallah and Barr, 1994) is almost the first step in signal pre-processing. Besides, a combination of FIR filters and IIR filters can be used to achieve the best results.

2.5.2.2 Effect of Electrode Referencing

Selecting the reference electrode location in EEG is challenging as it affects the signal amplitude and temporal structure. Commonly used references include a unipolar or bipolar reference, common average reference (CAR), Laplacian reference (LAP), and reference electrode standardisation technique (REST) (Hwang and Choi, 2019). The most often used location for unipolar or bipolar reference is the vertex, nose, and linked mastoids or ears. According to the CAR and the LAP, those references are a type of spatial filtering method that enhances the signal-to-noise ratio. The REST has been increasingly implemented as a re-reference technique to transform the actual electrodes into approximately zero reference ones; the idea is to create a virtual reference location at infinity. Considering the effect of reference choice, the selection of different kinds of referencing methods has an impact on power spectra (Yao et al., 2005) and functional connectivity metrics (Bastos and Schoffelen, 2016) such as coherence, phase-locking value, and phase lag index. Figure 2.11 illustrates the typical set-up of electrode reference in EEG recording, including unipolar reference, bipolar

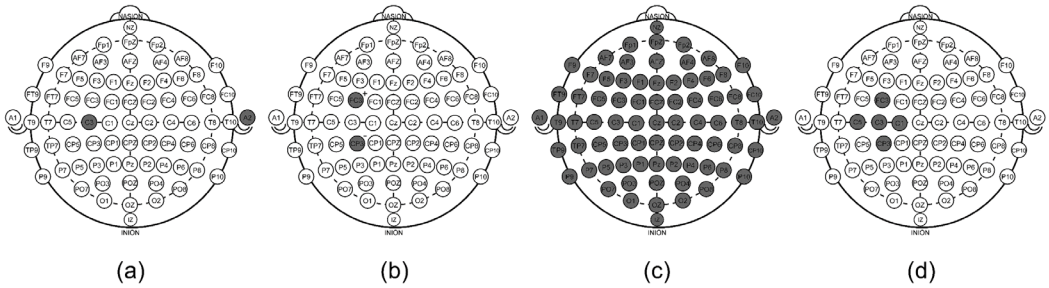

Figure 2.11. Commonly used electrode referencing in EEG recording. (a) unipolar reference using ear reference, (b) bipolar reference, (c) common average reference (CAR), (d) small Laplacian reference (LAP). A black dot represents the electrode positions required for each reference type. All types except a unipolar reference require an additional ground electrode. Note that the eletrode layout is based on 10-10 system.

reference, common average reference, and Laplacian reference. The layout is based on the 10-10 system, and the configuration can also be applied to other EEG montages.

2.5.2.3 Feature Extraction

Feature extraction primarily aims to reduce the number of input data from high dimensions to be categorised or differentiated into classes in a lower dimension. As EEG signals can be represented in many ways (Lotte et al., 2018), there are three significant types of feature extraction in BCI studies depending on purposes and domains: dimension reduction, space, and time frequency. Some frequently used techniques for dimension reduction are principal component analysis (PCA) (Wold et al., 1987) and independent component analysis (ICA) (Hyvärinen and Oja, 2000). Both techniques can also be used for noise and artefact reduction (Chawla, 2011). Besides, common spatial pattern (CSP) (Wang et al., 2006a) is a well-known space-based feature extraction technique often used in motor imagery studies and in real-time applications. Based on a time-frequency representation, various techniques have been commonly used, for example, autoregressive components (AR) (Zhang et al., 2017b), matched filtering (MF) (Boashash and Azemi, 2014), continuous wavelet transform (CWT) (Bostanov, 2004), and discrete wavelet transform (DWT) (Cvetkovic et al., 2008).

Apart from the techniques mentioned above, other commonly used approaches explore frequency-bandpower features (Brodu et al., 2011) and time-point features (Deng et al., 2013). Bandpower features consider the power of signals for a given frequency band, while time-point features consider the concatenation of signal samples. Hence, a combination of feature extraction techniques can be applied to achieve both features (Alonso et al., 2007). Selecting a proper feature method should consider the nature and characteristics of target signals (Al-Fahoum and Al-Fraihat, 2014) and consider choosing the best outcome from a wide variety of different methods or choosing some of the most influential features from the currently available features called feature selection (Dash and Liu, 1997). This topic will be explained in-depth in the next section. A summary of typically used feature extraction methods in BCI research is presented in Table 2.6.

2.5.2.4 Feature Selection

Selecting the proper and correct features is essential in machine learning as some features that may be redundant or may not be related to the target should be discarded (Guyon and Elisseeff, 2003). Moreover, the feature selection step further reduces the number of extracted features, resulting in fewer correlated parameters, more straightforward observation and better computational efficiency (Lotte et al., 2018).

In a general feature selection method, there are three main types, including the filter, wrapper, and embedded approach (Cai et al., 2018). The filter approach does not use a learning algorithm on the original data but only considers statistical characteristics of the input data and focuses on the relationship between the features and the target classes. The coefficient of determination (R-squared) (Cheng and Garg, 2014), the mutual information ranking (Vergara and Estévez, 2014), and the variance threshold (Dhanya et al., 2020) are examples of the filter methods. Regarding the wrapper and embedded methods, these methods use a classifier to determine a subset of features at the cost of longer computation time. The wrapper approach applies learning algorithms to the original data, selects a subset of relevant features based on the learning algorithm's performance, trains a model with a different combination of the features, observes the resulting performance, and then selects the features resulting in the best out-of-sample performance. On the other hand, the embedded methods combine all feature selection techniques as part of the model construction process. The recursive feature elimination (RFE) (Darst et al., 2018), genetic algorithm (GA) (Leardi, 2000), and sequential forward/backward selection (SFS/SBS) (Reeves and Zhe, 1999) are increasingly popular wrapper-type feature selection approaches. In contrast, the most often used embedded-type approaches are the decision tree (Sugumaran et al., 2007) and Lasso regularisation (Fonti and Belitser, 2017). A summary of commonly used feature selection methods in BCI research is presented in Table 2.7, which describes the essential properties of each method.

2.5.2.5 Signal Classification

In machine learning, classification refers to the process of predicting the classes (sometimes called targets or labels) of a given example of input data (Zhang et al., 2017a). The number of classes can be two categories called binary classification or

multiple categories called multi-class classification. Classification requires machine learning algorithms to employ the features extracted as independent variables to define a boundary between targets in feature space. Presently, classification algorithms are mainly considered for supervised and unsupervised classification. However, there is no precise, convincing theory on how to map those algorithms onto problem types. Instead, a practical recommendation is to search for which algorithm and algorithm configuration under controlled experiments provide the best performance for a given classification (Lotte et al., 2018). Besides, in order to attain the most appropriate algorithm, the accuracy of the model, training time, linearity of the problem, number of hyperparameters, and number of features used can be considered as additional requirements when selecting a classification algorithm.

The commonly used machine learning algorithms in BCI research can be separated into three main groups: generative, linear, and non-linear (Lotte et al., 2018). A generative model is based on a joint probability distribution statistical model, which assigns an observed feature vector to a class with the highest probability. The generative classifier attempts to learn the model that creates data by estimating the model's assumptions and distributions. It then utilises this to forecast unobserved data, assuming that the learnt model reflects the accurate model. A linear classifier makes a classification decision based on the value of a linear combination of the attributes. The linear classifier works on the principle that a hyperplane can separate two values in the target class in the feature space; thus, it is usually sensitive to outliers or intense noise. On the other hand, non-linear classifiers are designed to handle issues that are present in linear models. Nonlinear classifiers are frequently more accurate than linear classifiers when a nonlinear problem and its class borders cannot be effectively approximated using linear hyperplanes.

An example of a generative model is the Naïve Bayes classifier (Berrar, 2018), a family of simple probabilistic classifiers based on Bayes' Theorem, assuming that a particular feature in a class is unrelated to the existence of any other feature. The Naïve Bayes algorithms are commonly used in sentiment analysis, spam filtering, and recommendation systems due to their simplicity, but the main limitation is that predictors must be independent. In most real-world cases, the predictors are dependent,

which reduces the classifier's performance. In BCI work, the Naïve Bayes has also been used to detect two-class motor imagery tasks (Wang and Zhang, 2016), which achieved an overall detection performance of 96.36%.

In addition to the popular classifiers in BCI, linear discriminant analysis (LDA) (Wu et al., 2013) is one of the most frequently used as a real-time classifier because it requires few computational operations. To prevent overfitting and decrease computing costs, the linear-type LDA classifier attempts to project a data set into a lower-dimensional space with acceptable class separability. LDA has been used in real-time motor imagery classification (Sarac et al., 2013) and emotion recognition (Bhardwaj et al., 2015). Besides, logistic regression, one of the most popular machine learning algorithms, is a linear-type classifier that is very similar to linear regression. Using continuous and discrete data sets, logistic regression can generate probabilities and categorise new data. Some BCI studies related to motor imagery classification (Isa et al., 2019; Camacho and Manian, 2016) employ logistic regression as the primary classifier, which demonstrates the potential of real-time processing.

Regarding the non-linear model, k-nearest neighbour (KNN) (Liao and Vemuri, 2002), quadratic discriminant analysis (QDA) (Tharwat, 2016), support vector machines (SVM) (Suthaharan, 2016), artificial neural network (ANN) (Yegnanarayana, 2009), and deep learning (LeCun et al., 2015) have been increasingly implemented in modern BCI research, including the detection of motor imagery (Chatterjee and Bandyopadhyay, 2016; Chaudhary et al., 2019), event-related potential (ERP) (Simbolon et al., 2015; Turnip and Soetraprawata, 2013), steady-state visual evoked potential (SSVEP) (Singla and Haseena, 2014; Aznan et al., 2018), drowsiness (Li et al., 2015; Hajinoroozi et al., 2015), emotional state (Wang et al., 2014; Alhagry et al., 2017), and seizure (Zhang and Chen, 2016; Acharya et al., 2018). Those mentioned non-linear algorithms outperform classical linear techniques. However, they often incur high computation costs. Table 2.8 summarises some of the most popular classification algorithms in BCI studies.

Table 2.6. A summary of common feature extraction methods used in BCI researches.

Type	Method	Properties	Applications
Dimension Reduction	PCA	Linear transformation. Transforms a set of possibly correlated data into a set of uncorrelated variables. Data is optimally represented in terms of minimal mean-square-error. There is no assurance that a categorization will always be accurate. PCA demands that artefacts be uncorrelated with the EEG data in order to reduce noise.	(Lin and Hsieh, 2009; Lins et al., 1993; Boye et al., 2008)
	ICA	Separates a set of mixed signals into its original sources. Assumes mutual statistical independence of underlying sources. Provides powerfulness and robustness for artifact removal. For artefact removal in BCI works, the artefacts are required to be independent from the EEG signal. The power spectrum may be tainted.	(Li et al., 2009; Castellanos and Makarov, 2006; Gao et al., 2010; Flexer et al., 2005; Jung et al., 2000; Chiappa and Barber, 2006; Wallstrom et al., 2004)
Space	CSP	A type of spatial filter specifically designed for 2-class problems and can be extended for multiclass problems. Very efficient for synchronous BCI applications but less effective for asynchronous ones. Spatial resolution affects its performance.	(Ramoser et al., 2000; Dornhege et al., 2006; Grosse-Wentrup and Buss, 2008; Lemm et al., 2005; Mousavi et al., 2011)
Time-Frequency	AR	Based on spectrum model. Provides high frequency resolution for short-time segments. Suitable for signal with sharp spectral features but not suitable for non-stationary.	(Krusienski et al., 2006; Wang et al., 2010)
	MF	Detects a specific pattern that matches predefined signals or templates. Efficient for detecting waveforms with consistent temporal characteristics.	(Krusienski et al., 2007; Brunner et al., 2010a)
	CWT	Offers both frequency and temporal information. Appropriate for non-stationary signals.	(Bostanov, 2004; Senkowski and Herrmann, 2002)
	DWT	Provides the same time-frequency information as CWT but with less redundancy and complexity. Requires computational resources less than the CWT.	(Hinterberger et al., 2003; Mason and Birch, 2000)

Table 2.7. A summary of common feature selection methods in BCI researches.

Type	Method	Properties	Applications
Filter	R-squared	A relationship measure determined via Pearson correlation coefficient. Often used as feature ranking criterion. Only suitable for detection of linear dependencies.	(Chum et al., 2012; Simões et al., 2010)
	Mutual Information	A measure between two possibly multi-dimensional random variables. Determines based on the mutual information between each feature and the target variable. Suitable for nonlinear transformation and extraction of high-order statistics. Maximising this quantity is an NP-hard optimisation problem.	(Ang et al., 2012; Zhang et al., 2016; Atkinson and Campos, 2016)
	Variance Threshold	Uses a user-specified threshold to filter the variance of each feature. Assumes that features with a higher variance may contain more useful information.	(Ma et al., 2016; Kang et al., 2016)
Wrapper	RFE	Effective at selecting features in a training dataset. Uses filter-based feature selection internally. Suitable for classification problems regression problems. A number of features to select is an Important hyperparameter.	(Hidalgo-Muñoz et al., 2014; Zhong and Jianhua, 2017; Li et al., 2018a)
	GA	A stochastic technique for function optimization based on the mechanics of natural genetics and biological evolution. High resource consumption. Convergence may occur prematurely.	(Rejer, 2013; Erguzel et al., 2015; Shon et al., 2018)
	SFS/SBS	Suboptimal methods. SFS is unable to eliminate non-useful features after adding other features. SBS is incapable to re-evaluate the usefulness of an excluded feature.	(Dias et al., 2010; Lakany and Conway, 2007)
Embedded	Decision Tree	The number of legs is the most important feature. Calculated based on Gini coefficient, entropy or Chi-Square value.	(Duan et al., 2015; Sugumaran et al., 2007)
	Lasso Regularisation	Reduces the degree of overfitting or variance of a model by adding a penalty (bias). Suitable for generalized linear models.	(Yuwono et al., 2020; Zhang et al., 2020)

Table 2.8. A summary of the most popular classification methods in BCI works.

Type	Method	Properties	Applications
Generative Model	Naive Bayes	Assigns an observed feature vector to a class that has a highest probability. Creates non-linear decision boundaries. Not much popular in BCI research.	(He et al., 2015; Pires et al., 2008; Kawanabe et al., 2006)
Linear	LDA	Simple linear classifier with acceptable accuracy and requires low computation. Usually applies for two-class problems. Extended multi-class version exits. Sensitive to outliers or strong noise.	(Wu et al., 2013; Aldea and Fira, 2014; Saa and Gutierrez, 2010)
	Logistic Regression	An alternative method other than the linear regression. Uses the natural logarithm function to find the relationship between the variables and uses test data to find the coefficients. Works well in binary problems.	(Tomioka et al., 2007; Hammer et al., 2012; Kus et al., 2012; Fiebig et al., 2016)
Non-Linear	QDA	A variant of LDA that allows for non-linear separation of data. Particularly useful if there is prior knowledge that individual classes exhibit distinct covariances.	(Eva and Lazar, 2015; Shenoy et al., 2006; Sita and Nair, 2013)
	SVM	Offers linear and non-linear modalities. Uses in binary or multi-class problems. Maximises the distance between the nearest training samples and the hyperplanes. Sensitive outliers or strong noise and often requires regularisation.	(Rakotomamonjy and Guigue, 2008; Salvaris and Sepulveda, 2009; Iacoviello et al., 2015)
	KNN	Uses metric distance between the test features and their neighbors. Suitable for multi-class problems Highly efficient with low dimensional feature vectors. Sensitive to feature's dimension.	(Isa et al., 2019; Bhattacharyya et al., 2010; Rahman et al., 2020)
	ANN	Originally mimic the functionality of human brain, with neuron nodes interconnected like a web. Commonly used architectures: convolutional neural network (CNN), recurrent neural network (RNN), and long short-term memory (LSTM).	(Hamedi et al., 2014; Bevilacqua et al., 2014; Mohammadpour et al., 2017; Aguilar et al., 2015)
	Deep Learning	Extended version of ANN model that is generated from extremely connected multiple layers. A combination of advanced ANN architectures provides better results.	(Fahimi et al., 2019; Tabar and Halici, 2016; Hosseini et al., 2017; Dai et al., 2019)

2.5.3 Processing Platforms

This section of the thesis discusses current technologies related to the processing of machine learning algorithms. In order to accomplish cost-effective resource management, it is critical to select a suitable processing system. Also, the processing system is an important part when previous studies related to computation explain the real-time capabilities of their system. Recently, in signal processing, the most implemented platforms (Pauwels et al., 2011) are the central processing unit (CPU), graphics processing unit (GPU), field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and digital signal processors (DSP). Currently, the CPU and GPU play an essential role in processing machine learning algorithms for deep learning and cloud computing. While FPGA, ASIC, and DSP have been designed for working in a smaller environment, such as a mobile unit or for internet-of-thing devices. A summary of the differences between those modern technologies is displayed in Table 2.9. In addition, the processing platforms can be further categorised into sections. The following list specifics several processing systems which were discovered throughout this review:

2.5.3.1 Simulation Environment (SIM)

The purpose of computer simulation is to visualise theoretical models of a particular system (Law and Kelton, 2000). In medical applications, these models take the form of an algorithm-based system that analyses physiological data. Such systems have developed into a significant component of biomedical engineering since they may be used to investigate and therefore acquire fresh perspectives about new technologies and to estimate the performance of systems that are too complex for analytical solutions (Jacquemet et al., 2008). Phan et al., for example, developed patient-specific models to aid in their understanding of neuropsychological, behavioural, and social phenomena (Phan et al., 2012). According to Abdulghani (Abdulghani et al., 2012), they investigated the functional performance of different implementations of the compressive sensing theory in detail when applied to scalp EEG signals. Their results showed successful reconstructions, but they were computationally expensive. Besides, Olkkonen (Olkkonen et al., 2002) applied a discrete-time wavelet transform to

Table 2.9. Differences of current processing platforms (CPU, GPU, FPGA/ASIC, and DSP).

Differences	CPU	GPU	FPGA/ASIC	DSP
Embedded	Maybe	Difficult	Easy	Easy
Power Consumption	High	Very high	Low	Low
Flaoting Operation	Good	Excellent	Possible	Excellent
Integer Operation	Excellent	Excellent	Excellent	Excellent
Control Flow	Excellent	Challenging	Challenging	Fair
Number of GPIO	Fewer	No	Massive	Many
Certain Operation	Extremely versatile	Extremely strong	Not goog	Extremely strong
Pipelinging Operation	Sequential	Limited parallel	Massive parallel	Limited parallel
Execution Time	Non-deterministic	Fast and real-time	Extremely fast and real-time	Fast (limited in certain areas)
Programmability	With languages	With imaging libraries	Complicated	Proprietary tools
Appropriate tasks	Decision making	Image/video processing	Vastly parallel operations	Multiply-accumulate

decompose the EEG signal into parallel decimated sub-signals that resulted in remarkable computational savings.

2.5.3.2 Central Processing Unit (CPU)

A CPU executes a computer program's instructions by performing logic, input-output operations, and basic arithmetic (Hayes, 1998). To implement practical problem solutions in biomedical engineering, this architecture has been commonly used as a proof of concept due to its simplicity and robustness. However, in general, research in this field has been applied to non-real-time operations in which the CPU processing power is shared among multiple tasks. Significantly, such operating systems are incapable of meeting the requirements of real-time standards (Krishna, 1999). There are many examples of research working with CPU-based computing EEG, especially for brain-computer interface classification (Crevecoeur et al., 2008; Sartoretto and Ermani, 1999; Gernert et al., 1998; Kitamura et al., 1991; Meng et al., 2016; Alonso-Valerdi et al., 2015; Sivasankari and Thanushkodi, 2013).

2.5.3.3 Embedded (EMB) Systems

Embedded computer systems are designed to perform specific control tasks inside a larger system, often with restrictions on real-time computation (Noergaard, 2005). EMB systems implement very time-consuming problem remedies. Generally, such

systems are designed for practical applications, many of which include the collection and control of physiological signals. As a result, EMB systems have significant real-time requirements. Embedded systems are increasingly being utilised as operating systems, even though their performance is less than that of CPUs. These real-time operating systems ensure that the amount of processing power allocated to a given job remains constant throughout time. Currently, many embedded systems available on the market are utilised for processing the EEG, ranging from acquisition (Saptono et al., 2016; Madoš et al., 2016; Ahamed et al., 2016; Koga et al., 2013) to whole applications (Abdulhay et al., 2016; Paszkiel et al., 2016; Mirza et al., 2015; Nieva et al., 2012; Putra et al., 2017; Chai et al., 2017).

2.5.3.4 Graphics Processing Unit (GPU)

A GPU is designed to perform fast memory operations in order to accelerate picture creation in a frame buffer (Kirk and Hwu, 2010). GPUs are massively parallel processors connected through high-speed connections to sufficient amounts of random-access memory (RAM). These processing architectures are ideal for a variety of biological applications. For example, algorithms are often designed for GPU computers in order to make use of the processor's parallelism. In most recent studies, describing real-time GPU systems means lower processing time (Deng et al., 2012; Pieloth et al., 2013; Zhu, 2013; Juhasz and Kozmann, 2016; Chen et al., 2017; Wilson, 2011). These studies highlight many benefits of this hardware acceleration, including lowering the computational burden and the allocated memory on the central processor, increasing system speed and stability, scaling for large computing, and providing real-time processing.

2.5.3.5 Very Large-Scale Integration (VLSI)

VLSI is a technique for integrating thousands of transistors into a single integrated circuit (IC) (Taur and Ning, 1998). Essentially, all digital processors are based on VLSI architectures. Nevertheless, this review will only focus on the VLSI idea, such that it denotes a processing system that is developed/customised for an algorithm. This speciality can lead to economical and state-of-the-art solutions for complex problems in biomedical engineering (Athan and Landis, 1994). For instance, VLSI technology

was used as the central processor, which was integrated into implanted medical devices for on-chip deoxyribonucleic acid (DNA) analysis (Iniewski, 2008). Many applications (Pavan and Rajesh, 2017; Li et al., 2016; Zaghloul and Bayoumi, 2015; Ajay and Lourde, 2015; Shih et al., 2013; Huang et al., 2013; Abdelhalim et al., 2011) for real-time EEG computing designed using VLSI technology were proposed.

2.6 Parallel Computing Architecture

2.6.1 Challenges in Real-Time BCI Processing

Processing signals in real-time encounters difficulties that can range from the hardware level to the software level. One of the most challenging topics in online BCI processing is reducing system latency (Xu et al., 2013) so that the delay between detecting an intended action and the command to perform a BCI output is as short as possible. Typically, optimising signal processing algorithms may not be sufficient to minimise the latency. Instead, the most commonly used techniques (Oweiss, 2006) include increasing the speed of data transmission, buffering of incoming data, and the use of fast processing units. However, these may require time and effort to develop as several parameters related to those techniques need to be considered carefully.

Optimising a signal-processing pipeline requires several optimisation stages, such as increasing the memory buffer, reducing computational complexity, and increasing the data transmission rate between a computational unit and application control. One way to overcome the complexity of computational problems is to employ high-performance computing (HPC) technologies such as cloud computing or supercomputers, which offer immense computational power (Senyo et al., 2018). At present, HPC technology plays an essential role in solving complex computational problems such as simulation, modelling, and analysis (Gulo et al., 2019; Xu et al., 2015). The principal concept behind this technology is called parallel computing or heterogeneous computing, which requires massively parallel processing. This technique can solve contemporary problems by breaking a big problem into small parts, then concurrently executing individual hardware acceleration. Currently, there are many hardware acceleration devices available on the market at a reasonable price.

This includes the consumer graphics processing unit (GPU), Tensor processing unit (TPU) and the user-friendly field-programmable gate array (FPGA). Moreover, these hardware devices are programmable, which requires specific hardware languages, but inexperienced users may take a long time to understand their functionality. With the advantages of the HPC, it can be applied to many advanced computing techniques such as machine learning, deep learning, and feature extraction implemented in real time (Peker et al., 2015; Edelman et al., 2019).

2.6.2 Parallel Programming Languages

Several parallel computing architectures, such as multi-core CPU and GPU, require parallel programming in order to execute a command. The concepts of parallel programming or concurrent programming are widely established (Hansen, 1973; Andrews and Schneider, 1983) and may become an essential technique when developing a real-time system that requires continuous monitoring of simultaneous activities with critical timing constraints (Hansen, 1978). Nowadays, concurrent programming plays a vital role in parallel computing schemes, particularly digital signal processing (Athi et al., 2016; Lee et al., 2021).

Programming parallel computers involves concurrent programming languages, libraries, APIs, and parallel programming models. Based on an underlying memory architecture, the basis of parallel programming can be divided into three classes, including distributed memory, shared memory, and shared distributed memory. Programming languages based on shared memory require internal communication by working with shared memory variables. Two of the most widely used programming APIs for this memory type are POSIX Threads (Butenhof, 1997) and OpenMP (Dagum and Menon, 1998). Parallel programming based on distributed memory employs message passing, which the Message Passing Interface (MPI) (Gropp et al., 1999) is the most commonly used API. Finally, the future concept, where a part of a program commits to sending required data to another part of a program at some future point, is the fundamental concept in programming parallel programs.

Another directive-based programming model that aims to handle hardware accelerators without the complexity associated with GPU programming created by

CAPS enterprise and Pathscale for hybrid multi-core parallel programming (HMPP) is called OpenHMPP (Andión et al., 2016). Based on a set of compiler directives, the OpenHMPP can efficiently reduce computational loads on hardware accelerators and optimise data transfer between hardware memory units. In addition to the parallel programming model, one well-known framework employed in big data application is MapReduce (Dean and Ghemawat, 2008), a modern processing technique for processing large data sets on a cluster.

According to the included literature, each of the aforementioned parallel computing systems needs the development of a unique design tool. While parallel computing may be used to accelerate signal processing, it needs a thorough knowledge of parallel programming languages and the architecture of the hardware being implemented. This complicates the process of choosing a suitable system to build, and highlights a skill set necessary for multidisciplinary BCI research.

2.6.3 OpenCL

Regarding hardware acceleration, there are many portable devices that are available on the market at a reasonable price, such as a consumer GPU and a standalone FPGA module. However, rather than using a ready-to-use toolbox, those accelerating devices require specific programming languages, necessitating time for developers to learn and operate (Rodríguez-Andina et al., 2015). To overcome this problem, an Open Computing Language (OpenCL) has been designed to contribute to the open-source community (Stone et al., 2010). The significant advantage of OpenCL is that it provides a cross-platform programming language, which can be written once and run everywhere with few modifications. This strategy allows any accelerating device to execute based on one-time coding. Up to the present time, the OpenCL platform has been commonly recognised as an industry standard for programming heterogeneous computing systems.

The OpenCL platform is widely used in high-performance computing, such as multicore CPUs, GPUs, or other accelerators maintained by Khronos Group, which through heterogeneous computing, achieves execution acceleration. The OpenCL SDK is available for several microprocessor vendors, including AMD, ARM,

NVIDIA, and Intel. The advantages of parallel computing have recently been used and applied to BCI applications, speeding up signal processing pipelines such as feature selection (Escobar et al., 2016a; Escobar et al., 2016b) and frequency analysis (Wilson, 2011; Escobar et al., 2019). OpenCL provides a cross-platform, parallel computing API and a C-like language interface for heterogeneous computing devices, reducing developer learning. In principle, OpenCL could also target DSPs, Cell, and perhaps FPGAs, in which only one program can be tested on multiple platforms.

In addition to developing a real-time system based on the OpenCL platform, it is necessary to understand the heterogeneous computing concept used in parallel processing schemes and technical terms. Some terminologies primarily used for OpenCL programming are listed below:

1) Kernel: a kernel is an expressly coded function that is created to be run by one or more OpenCL-compliant devices. Usually, kernels are submitted to their intended device or devices by host applications that are run on the user's development system.

2) Context: a container used for the host device, which can manage the connected OpenCL devices. At the beginning of the OpenCL initialisation, the host selects devices and places them in a context.

3) Program: a kernel container where the host can select any functions to create a specific kernel. Then the host associates the kernel with argument data and dispatches it to a command queue.

4) Command queue: the mechanism through which the host associates the OpenCL devices with a to-do task. Multiple kernels can be associated with one command queue. Functions that are submitted to a command queue can be configured to execute in-order or out-of-order.

5) Work-item: a single implementation of the kernel on a specific set of data, in which a set of tasks to be performed on the data is identified by the kernel. For every kernel, there can be multiple work items.

6) Work-group: a combination of work items that access the same processing resources. Work-items in a single workgroup can access the same block of high-speed memory called local memory and can also be synchronised internally.

7) Global memory: host-access storage allocated on the OpenCL devices primarily transfers data from the host to the device and vice versa. This memory is ordinarily the largest memory region on the device. However, it is the slowest for work items to access.

8) Local memory: a low-level memory inside the OpenCL devices shares data between work items in the same workgroup. Accessing the local memory is much faster than accessing global memory.

9) Shared visual memory: a new feature introduced in OpenCL 2.0 that allows the host and OpenCL devices to share pointers and complex pointer-containing data structures effortlessly. This feature is designed to address the data transfer bottleneck that regularly occurs when using global memory.

10) Pipes: another feature introduced in OpenCL 2.0 that allows direct communication between kernels without interfacing the global memory and returning to the host for synchronisation between the kernels. This mechanism follows the FIFO instruction.

In general, an OpenCL program is created for a host and a kernel is then executed on the OpenCL devices, such as GPU cards. The host device is responsible for manipulating all-important tasks. It includes initiating an environment (platform, context, program, and kernel), managing device buffers, transferring data between the host and the OpenCL devices, launching the kernel, and reading the data from the OpenCL devices when it finished execution. To understand the architecture of OpenCL, Figure 2.12 illustrates the implementation overview of the framework based on the parallel scheme. Note that the OpenCL standard generally defines data types, data structures, and functions augmenting in the C/C++ languages.

2.7 Artefact Management in EEG Recording

2.7.1 Types of Artefacts

One of the biggest challenges in EEG recording is reducing or removing non-desired signals from the original signal. An acquisition system with a low signal-to-noise ratio usually encounters interference, resulting in inaccurate interpretation and

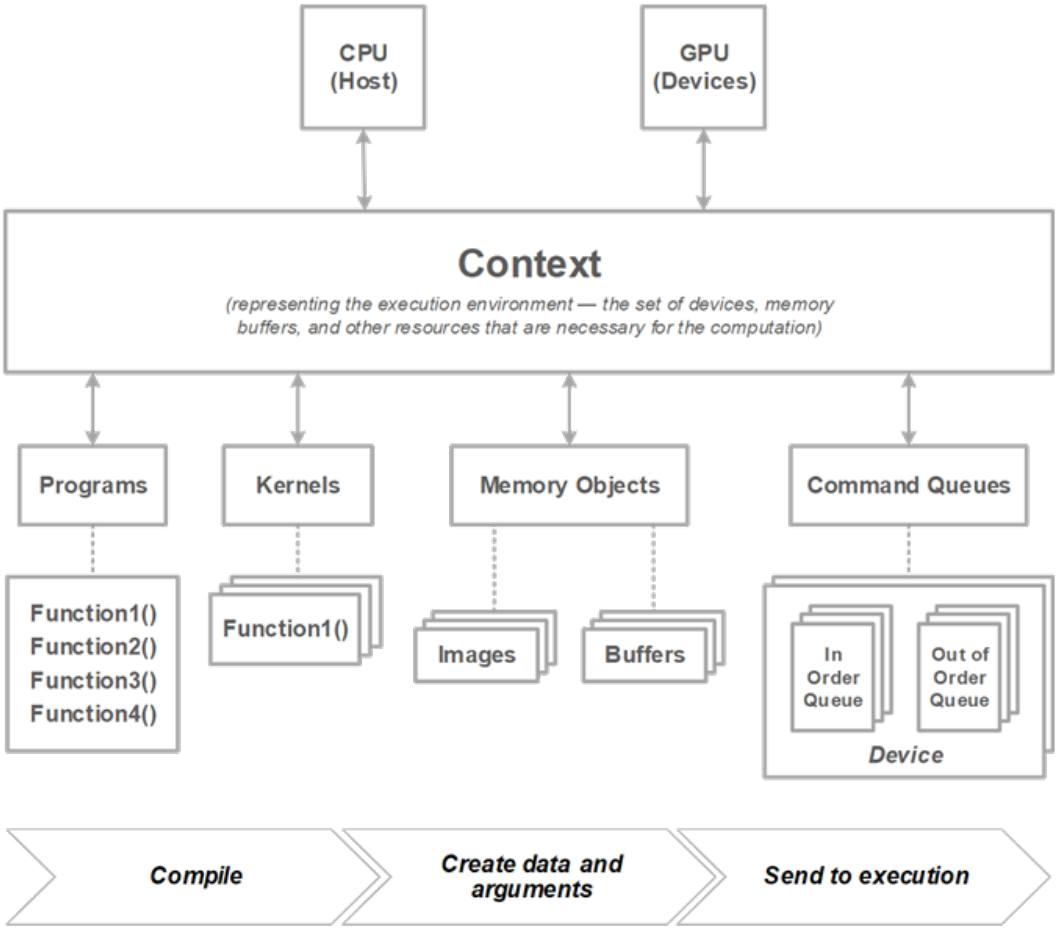

Figure 2.12. Basic concepts of OpenCL based on parallel computing scheme. The host associates devices with the context by creating kernels from programs, initialising the kernels with data and arguments, and sending the kernels to execute in command queues. Transferring data between the host and devices can be accessed using global memory. Appropriate using local memory can significantly improve the speed of execution and the efficiency of memory usage.

classification. In practice, using both analogue and digital filters, special artefact reduction techniques can be applied to minimise the effect of those unwanted signals. From the literature, the most common artefacts in EEG recording can be mainly categorised as follows: ocular artefacts (eye movement and blinks), muscular artefacts (EMG), cardiac activity (ECG), less common physiological artefacts, e.g. swallowing, speaking, coughing, deep breathing, and non-physiological artefacts such as mains electricity, cable or electrode movement artefacts. Table 2.10 summarises those artefacts' characteristics, including source, frequency range, amplitude, and morphology.

Table 2.10. Summary of the characteristic of the most occurred EEG artefacts.

Artefact	Source	Bandwidth (Hz)	Amplitude (mV)	Morphology
Occular (EOG)	Eye	0.1–10	0.05–3.5	Delta rhythm
Muscular (EMG)	Body muscle	10–500	0.1–100	Beta rhythm
Cardiac (ECG)	Heart	0.05–150	0.1–5	Epilepsy
Other physiological	Swallowing, breathing, sweating	Very low	High	Different from actual EEG
Non-physiological	mains electricity, cable/electrode movement	0.05-2, 50–60	Low and high	Delta rhythm, beta/gamma rhythm

2.7.1.1 Ocular Artefacts

The electrical potentials generated by voluntary or involuntary (saccades) eye movement or gazing create potentials that can be measured as the electrooculogram (EOG) and which can swamp the smaller fields comprising the EEG (Croft and Barry, 2000). To control this, it is recommended to record the EOG alongside EEG measurements as one way to identify when EOG interference occurs in the EEG. The EOG potentials are predominantly featured in frontal electrodes as they lie closest to the source (Romero et al., 2008; Fisch and Spehlman, 2008). The interference count has an effect on the distance between the electrodes and the eyes and the direction in which the eyes move. Additionally, blinking causes a considerable disturbance in an EEG recording. The blinking artefact is of greater amplitude than the background EEG activity (Croft and Barry, 2000) and is a short-lasting waveform widely distributed across all EEG electrodes. Ocular artefacts are referred to as OAs or EOG artefacts.

In practice, reference EOG waveforms are beneficial for ocular artefact cancellation and should be measured simultaneously with the EEG. A standard EOG recording configuration records three channels giving vertical (VEOG), horizontal (HEOG), and radial (REOG) eye movement measurements (Croft et al., 2005; Pham et al., 2011).

2.7.1.2 Muscular Artefacts

The electrical activity on the body surface caused by contracting muscles can be measured using an electromyogram (EMG). This artefact occurs in the EEG when groups of muscles are active, and this makes EEG recording during any activity that

involves movement challenging. Examples of when EMG contamination occurs include walking, talking, or swallowing (Sörnmo and Laguna, 2005a). Muscle contraction intensity, location, and the number and size of tensed muscles all have a direct effect on the shapes and amplitudes of the interferences.

The adverse effects of the EMG on the EEG activity are generally most considerable from the face, head, and neck sources (McMenamin et al., 2010; McMenamin et al., 2011) and can be difficult to remove from the EEG without affecting the EEG itself (Safieddine et al., 2012) given that the spectrum of the EMG occupies a wide distribution; hence it perturbs all frequency bands of the EEG, including alpha and beta bands. EMG can often be detected across the entire scalp (Goncharova et al., 2003) because of the capacity for conduction of myogenic activity produced by muscles from the head, face, and neck (McMenamin et al., 2010). Additionally, the EMG is combined in time with a variety of experimental manipulations, including cognitive load and vocalisation (McMenamin et al., 2010). Lastly, since the EMG is generated by the activity of geographically dispersed, functionally autonomous muscle groups with distinct topographic and spectral characteristics (Goncharova et al., 2003), it shows less replication than other biological artefacts and is, therefore, more difficult to distinguish.

2.7.1.3 Cardiac Activity

Cardiac artefacts arise from the rhythmic muscular activity of the heart and are measured by the electrocardiogram (ECG). Usually, the effect of this interference on the scalp is relatively low. However, sensitivity to EEG contamination depends on the position of referencing and grounding electrodes and varies with certain body types (Sörnmo and Laguna, 2005b). The ECG pattern is regular and shows very characteristic features, which can be misinterpreted as an epileptiform activity when the ECG is hardly visible in the scalp EEG (Benbadis et al., 2007). As the ECG is regularly recorded together with cerebral activity, the artefact is easier to rectify because a reference waveform frequently exists.

When an EEG recording electrode is placed over a pulsing vessel, such as a scalp artery, potential pulse artefacts may occur as a result of the electrode producing slow periodic waves that seem to be EEG activity (Fisch and Spehlman, 2008). This pulse

activity is much more difficult to detect than the ECG because of its low amplitude and frequency. Nevertheless, it will be isolated to one recoding electrode and can be diminished by appropriate electrode positioning (Anderer et al., 1999). Furthermore, as these pulse artefacts are correlated with the ECG, these signals can be readily detected if their presence is suspected (Benbadis et al., 2007).

2.7.1.4 Less Common Physiological Artefacts

In addition to the physiological artefacts mentioned previously, two significant interferences can occur from skin potential, including (a) slow waves caused by changes in the electrical baseline of particular EEG electrodes, perspiration artefacts, and (b) slow waves produced by sweat glands, commonly called the sympathetic skin response or galvanic skin potentials (Fisch and Spehlman, 2008). Other physiological artefacts consist of artefacts that are also based on the motion of a body part or organ, e.g., breathing artefacts, moving the tongue, dental restorations with different metals (Fisch and Spehlman, 2008).

Along with physiological artefacts, non-physiological artefacts such as transmission line noise are the most encountered interference in BCI, especially when dealing with recording equipment. However, this artefact can be straightforwardly removed using a digital or analogue filter due to its recognisable structure.

2.7.2 Denoising Techniques

Although suitable electrode attachment and the use of high-quality amplifier equipment can minimise unwanted interference, the challenge is that the real-world environment is different from a laboratory. From the literature, many techniques can be used for the removal of artefacts from an EEG. One standard attempt at eliminating artefacts from a measured EEG is simple low-pass, band-pass, or high-pass filtering. Nevertheless, this technique may result in loss of signal if EEG frequency bands overlap with the interference (Sweeney et al., 2012). Therefore, alternative techniques, such as adaptive filtering, Wiener filtering, and Bayes filtering (Sweeney et al., 2012), are needed when there is spectral overlap, as is the case with general artefacts recorded concurrently with the EEG. Additionally, linear regression (Gratton et al., 1983), EOG correction (Croft and Barry, 2000), blind source separation (Vigario and Oja, 2008)

and more modern techniques using advanced algorithms such as wavelet transform (WT) (Unser and Aldroubi, 1996), empirical mode decomposition (EMD) (Huang et al., 1998) and nonlinear mode decomposition (NMD) (Iatsenko et al., 2015) have been proposed to improve the signal-to-noise ratio of the recorded signal.

Blind source separation techniques are sometimes referred to as component-based approaches since they involve identifying the major components or independent components of the input EEG channels and executing domain transformations. Applying the inverse transformation to the corrected components reverses the process, resulting in all EEG channels being processed and estimated simultaneously. On the contrary, simple filtering, linear regression, EOG correction, wavelet transform, EMD and NMD (excluding multi-channel EMD) approximate each artefact-corrected channel separately, in both the time and frequency domains.

2.8 Chapter Summary

Chapter II provides essential background and review of the existing research related to real-time brain-computer interfacing technology. The overview of the EEG fundamentals and their applications, signal processing techniques, and the general architecture of computing hardware are revealed. Moreover, the parallel distributed processing concept and the open-source platform for that were introduced.

In Chapter III, the study of feature extraction methods that can be potentially used for real-time EEG applications is presented. Also, the chapter reveals a basic idea of how to apply the parallel computing scheme to address the latency issues in real-time signal processing.

2.9 References

Abdelhalim, K., Smolyakov, V., Shulyzki, R., Aziz, J. N. Y., Serletis, D., Carlen, P. L. & Genov, R. VLSI multivariate phase synchronization epileptic seizure detector. 2011 5th International IEEE/EMBS Conference on Neural Engineering, NER 2011, 2011. 461-464.

Abdulghani, A. M., Casson, A. J. & Rodriguez-Villegas, E. 2012. Compressive sensing scalp EEG signals: implementations and practical performance. *Medical & Biological Engineering & Computing,* 50**,** 1137-1145.

Abdulhay, E., Abdelhay, A., Kilani, A., Al-Shwiat, L. & Al-Rousan, S. 2016. Development of arduino based low cost neuro-feedback applied to ADHD. *Biomedical Research (India),* 2016**,** S31-S37.

Acharya, U. R., Oh, S. L., Hagiwara, Y., Tan, J. H. & Adeli, H. 2018. Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals. *Computers in biology and medicine,* 100**,** 270-278.

Aguilar, J. M., Castillo, J. & Elias, D. EEG signals processing based on fractal dimension features and classified by neural network and support vector machine in motor imagery for a BCI. VI Latin American Congress on Biomedical Engineering CLAIB 2014, Paraná, Argentina 29, 30 & 31 October 2014, 2015. Springer, 615-618.

Ahamed, M. A., Ahad, M. a. U., Sohag, M. H. A. & Ahmad, M. Development of low cost wireless biosignal acquisition system for ECG EMG and EOG. 2nd International Conference on Electrical Information and Communication Technologies, EICT 2015, 2016. 195-199.

Ajay, A. & Lourde, R. M. VLSI implementation of an improved multiplier for FFT computation in biomedical applications. IEEE International Conference on Electro Information Technology, 2015. 6-12.

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,* 2014.

Aldea, R. & Fira, M. 2014. Classifications of motor imagery tasks in brain computer interface using linear discriminant analysis. *International Journal of Advanced Research in Artificial Intelligence,* 3**,** 5-9.

Alhagry, S., Fahmy, A. A. & El-Khoribi, R. A. 2017. Emotion recognition based on EEG using LSTM recurrent neural network. *Emotion,* 8**,** 355-358.

Alon, G., Sunnerhagen, K. S., Geurts, A. C. H. & Ohry, A. 2003. A home-based, self-administered stimulation program to improve selected hand functions of chronic stroke. *Neurorehabilitation,* 18**,** 215-225.

Alonso-Valerdi, L. M., Salido-Ruiz, R. A. & Ramirez-Mendoza, R. A. 2015. Motor imagery based brain–computer interfaces: An emerging technology to rehabilitate motor deficits. *Neuropsychologia,* 79, Part B**,** 354-363.

Alonso, I. P., Llorca, D. F., Sotelo, M. Á., Bergasa, L. M., De Toro, P. R., Nuevo, J., Ocaña, M. & Garrido, M. Á. G. 2007. Combination of feature extraction methods for SVM pedestrian detection. *IEEE Transactions on Intelligent Transportation Systems,* 8**,** 292-307.

Ambler, T., Braeutigam, S., Stins, J., Rose, S. & Swithenby, S. 2004. Salience and choice: Neural correlates of shopping decisions. *Psychology & Marketing,* 21**,** 247-261.

Ambler, T., Ioannides, A. & Rose, S. 2000. Brands on the Brain: Neuro-Images of Advertising. *Business Strategy Review,* 11**,** 17-30.

Anderer, P., Roberts, S., Schlogl, A., Gruber, G., Klosch, G., Herrmann, W., Rappelsberger, P., Filz, O., Barbanoj, M. J., Dorffner, G. & Saletu, B. 1999. Artifact processing in computerized analysis of sleep EEG - A review. *Neuropsychobiology,* 40**,** 150-157.

Andión, J. M., Arenaz, M., Bodin, F., Rodríguez, G. & Touriño, J. 2016. Locality-aware automatic parallelization for GPGPU with OpenHMPP directives. *International Journal of Parallel Programming,* 44**,** 620-643.

Andrews, G. R. & Schneider, F. B. 1983. Concepts and notations for concurrent programming. *ACM Computing Surveys (CSUR),* 15**,** 3-43.

Ang, K. K., Chin, Z. Y., Zhang, H. & Guan, C. 2012. Mutual information-based selection of optimal spatial–temporal patterns for single-trial EEG-based BCIs. *Pattern Recognition,* 45**,** 2137-2144.

Angelakis, E., Stathopoulou, S., Frymiare, J. L., Green, D. L., Lubar, J. F. & Kounios, J. 2007. EEG neurofeedback: A brief overview and an example of peak alpha frequency training for cognitive enhancement in the elderly. *Clinical Neuropsychologist,* 21**,** 110-129.

Ariely, D. & Berns, G. S. 2010. SCIENCE AND SOCIETY Neuromarketing: the hope and hype of neuroimaging in business. *Nature Reviews Neuroscience,* 11**,** 284-292.

Athan, S. & Landis, D. L. Advanced vlsi technologies in biomedical engineering. Engineering in Medicine and Biology Society, 1994. Engineering Advances: New Opportunities for Biomedical Engineers. Proceedings of the 16th Annual International Conference of the IEEE, 1994. IEEE, 980-981.

Athi, M. V., Zekavat, S. R. & Struthers, A. A. 2016. Real-time signal processing of massive sensor arrays via a parallel fast converging svd algorithm: Latency, throughput, and resource analysis. *IEEE Sensors Journal,* 16**,** 2519-2526.

Atkinson, J. & Campos, D. 2016. Improving BCI-based emotion recognition by combining EEG feature selection and kernel classifiers. *Expert Systems with Applications,* 47**,** 35-41.

Aznan, N. K. N., Bonner, S., Connolly, J., Al Moubayed, N. & Breckon, T. On the classification of SSVEP-based dry-EEG signals via convolutional neural networks. 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2018. IEEE, 3726-3731.

Bai, Z., Fong, K. N., Zhang, J. J., Chan, J. & Ting, K. 2020. Immediate and long-term effects of BCI-based rehabilitation of the upper extremity after stroke: a systematic review and meta-analysis. *Journal of neuroengineering and rehabilitation,* 17**,** 1-20.

Bastos, A. M. & Schoffelen, J.-M. 2016. A tutorial review of functional connectivity analysis methods and their interpretational pitfalls. *Frontiers in systems neuroscience,* 9**,** 175.

Benbadis, S. R., Husain, A. M., Kaplan, P. W. & Dr. William O. Tatum, D. 2007. *Handbook of EEG Interpretation*, Springer Publishing Company.

Berrar, D. 2018. Bayes' theorem and naive Bayes classifier. *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics; Elsevier Science Publisher: Amsterdam, The Netherlands***,** 403-412.

Bevilacqua, V., Tattoli, G., Buongiorno, D., Loconsole, C., Leonardis, D., Barsotti, M., Frisoli, A. & Bergamasco, M. A novel BCI-SSVEP based approach for control of walking in virtual environment using a convolutional neural network. 2014 International Joint Conference on Neural Networks (IJCNN), 2014. IEEE, 4121-4128.

Bhardwaj, A., Gupta, A., Jain, P., Rani, A. & Yadav, J. Classification of human emotions from EEG signals using SVM and LDA Classifiers. 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), 2015. IEEE, 180-185.

Bhattacharyya, S., Khasnobish, A., Chatterjee, S., Konar, A. & Tibarewala, D. Performance analysis of LDA, QDA and KNN algorithms in left-right limb movement classification from EEG data. 2010 International conference on systems in medicine and biology, 2010. IEEE, 126-131.

Bigdely-Shamlo, N., Mullen, T., Kothe, C., Su, K.-M. & Robbins, K. A. 2015. The PREP pipeline: standardized preprocessing for large-scale EEG analysis. *Frontiers in Neuroinformatics,* 9**,** 16.

Bin, G. Y., Gao, X. R., Wang, Y. J., Hong, B. & Gao, S. K. 2009a. VEP-Based Brain-Computer Interfaces: Time, Frequency, and Code Modulations. *Ieee Computational Intelligence Magazine,* 4**,** 22-26.

Bin, G. Y., Gao, X. R., Wang, Y. J., Li, Y., Hong, B. & Gao, S. K. 2011. A high-speed BCI based on code modulation VEP. *Journal of Neural Engineering,* 8.

Bin, G. Y., Gao, X. R., Yan, Z., Hong, B. & Gao, S. K. 2009b. An online multi-channel SSVEP-based brain-computer interface using a canonical correlation analysis method. *Journal of Neural Engineering,* 6.

Birbaumer, N., Ghanayim, N., Hinterberger, T., Iversen, I., Kotchoubey, B., Kubler, A., Perelmouter, J., Taub, E. & Flor, H. 1999. A spelling device for the paralysed. *Nature,* 398**,** 297-298.

Birbaumer, N., Kubler, A., Ghanayim, N., Hinterberger, T., Perelmouter, J., Kaiser, J., Iversen, I., Kotchoubey, B., Neumann, N. & Flor, H. 2000. The thought translation device (TTD) for completely paralyzed patients. *Ieee Transactions on Rehabilitation Engineering,* 8**,** 190-193.

Boashash, B. & Azemi, G. 2014. A review of time–frequency matched filter design with application to seizure detection in multichannel newborn EEG. *Digital Signal Processing,* 28**,** 28-38.

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. *IEEE Transactions on Biomedical engineering,* 51**,** 1057-1061.

Boye, A. T., Kristiansen, U. Q., Billinger, M., Nascimento, O. F. D. & Farina, D. 2008. Identification of movement-related cortical potentials with optimized spatial filtering and principal component analysis. *Biomedical Signal Processing and Control,* 3**,** 300-304.

Bradberry, T. J., Gentili, R. J. & Contreras-Vidal, J. L. 2010. Reconstructing Three-Dimensional Hand Movements from Noninvasive Electroencephalographic Signals. *Journal of Neuroscience,* 30**,** 3432-3437.

Brodu, N., Lotte, F. & Lécuyer, A. Comparative study of band-power extraction techniques for motor imagery classification. 2011 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2011. IEEE, 1-6.

Brunner, C., Allison, B. Z., Krusienski, D. J., Kaiser, V., Muller-Putz, G. R., Pfurtscheller, G. & Neuper, C. 2010a. Improved signal processing approaches

in an offline simulation of a hybrid brain-computer interface. *J Neurosci Methods,* 188**,** 165-73.

Brunner, P., Joshi, S., Briskin, S., Wolpaw, J. R., Bischof, H. & Schalk, G. 2010b. Does the 'P300' speller depend on eye gaze? *J Neural Eng,* 7**,** 056013.

Buch, E., Weber, C., Cohen, L. G., Braun, C., Dimyan, M. A., Ard, T., Mellinger, J., Caria, A., Soekadar, S., Fourkas, A. & Birbaumer, N. 2008. Think to move: a neuromagnetic brain-computer interface (BCI) system for chronic stroke. *Stroke,* 39**,** 910-917.

Butenhof, D. R. 1997. *Programming with POSIX threads*, Addison-Wesley Professional.

Cai, J., Luo, J., Wang, S. & Yang, S. 2018. Feature selection in machine learning: A new perspective. *Neurocomputing,* 300**,** 70-79.

Callan, D. E., Durantin, G. & Terzibas, C. 2015. Classification of single-trial auditory events using dry-wireless EEG during real and motion simulated flight. *Frontiers in Systems Neuroscience,* 9.

Camacho, J. & Manian, V. Real-time single channel EEG motor imagery based Brain Computer Interface. 2016 World Automation Congress (WAC), 2016. IEEE, 1-6.

Castellanos, N. P. & Makarov, V. A. 2006. Recovering EEG brain signals: Artifact suppression with wavelet enhanced independent component analysis. *Journal of Neuroscience Methods,* 158**,** 300-312.

Cencen, V., Hirotani, M. & Chan, A. D. C. Comparison of active and passive electrodes in their optimized electroencephalography amplifier system. 2016 IEEE EMBS International Student Conference (ISC), 29-31 May 2016 2016. 1-4.

Chai, R., Naik, G. R., Ling, S. H. & Nguyen, H. T. 2017. Hybrid brain-computer interface for biomedical cyber-physical system application using wireless embedded EEG systems. *BioMedical Engineering Online,* 16.

Chatrian, G. E., Lettich, E. & Nelson, P. L. 1985. Ten Percent Electrode System for Topographic Studies of Spontaneous and Evoked EEG Activities. *American Journal of EEG Technology,* 25**,** 83-92.

Chatterjee, R. & Bandyopadhyay, T. EEG based Motor Imagery Classification using SVM and MLP. 2016 2nd international conference on Computational Intelligence and Networks (CINE), 2016. IEEE, 84-89.

Chaudhary, S., Taran, S., Bajaj, V. & Sengur, A. 2019. Convolutional neural network based approach towards motor imagery tasks EEG signals classification. *IEEE Sensors Journal,* 19**,** 4494-4500.

Chawla, M. 2011. PCA and ICA processing methods for removal of artifacts and noise in electrocardiograms: A survey and comparison. *Applied Soft Computing,* 11**,** 2216-2226.

Chen, D., Hu, Y., Cai, C., Zeng, K. & Li, X. 2017. Brain big data processing with massively parallel computing technology: challenges and opportunities. *Software - Practice and Experience,* 47**,** 405-420.

Cheng, C.-L. & Garg, G. 2014. Coefficient of determination for multiple measurement error models. *Journal of Multivariate Analysis,* 126**,** 137-152.

Cheng, M., Gao, X. R., Gao, S. G. & Xu, D. F. 2002. Design and implementation of a brain-computer interface with high transfer rates. *Ieee Transactions on Biomedical Engineering,* 49**,** 1181-1186.

Chi, Y. M. & Cauwenberghs, G. Wireless non-contact EEG/ECG electrodes for body sensor networks. 2010 International Conference on Body Sensor Networks, 2010. IEEE, 297-301.

Chiappa, S. & Barber, D. 2006. EEG classification using generative independent component analysis. *Neurocomputing,* 69**,** 769-777.

Chum, P., Park, S.-M., Ko, K.-E. & Sim, K.-B. Optimal EEG feature extraction based on R-square coefficients for motor imagery BCI system. 2012 12th International Conference on Control, Automation and Systems, 2012. IEEE, 754-758.

Commission, I. E. 2015. *Medical Electrical Equipment: Particular Requirements for the Basic Safety and Essential Performance of Medical Beds*, International Electrotechnical Commission.

Cook, I. A., Warren, C., Pajot, S. K., Schairer, D. & Leuchter, A. F. 2011. Regional brain activation with advertising images. *Journal of Neuroscience, Psychology, and Economics,* 4**,** 147.

Crevecoeur, G., Hallez, H., Van Hese, P., D'asseler, Y., Dupré, L. & Van De Walle, R. 2008. EEG source analysis using space mapping techniques. *Journal of Computational and Applied Mathematics,* 215**,** 339-347.

Croft, R. J. & Barry, R. J. 2000. Removal of ocular artifact from the EEG: a review. *Neurophysiologie Clinique-Clinical Neurophysiology,* 30**,** 5-19.

Croft, R. J., Chandler, J. S., Barry, R. J., Cooper, N. R. & Clarke, A. R. 2005. EOG correction: A comparison of four methods. *Psychophysiology,* 42**,** 16-24.

Cvetkovic, D., Übeyli, E. D. & Cosic, I. 2008. Wavelet transform feature extraction from human PPG, ECG, and EEG signal responses to ELF PEMF exposures: A pilot study. *Digital signal processing,* 18**,** 861-874.

Dagum, L. & Menon, R. 1998. OpenMP: an industry standard API for shared-memory programming. *IEEE computational science and engineering,* 5**,** 46-55.

Dai, M., Zheng, D., Na, R., Wang, S. & Zhang, S. 2019. EEG classification of motor imagery using a novel deep learning framework. *Sensors,* 19**,** 551.

Daly, J. J., Hogan, N., Perepezko, E. M., Krebs, H. I., Rogers, J. M., Goyal, K. S., Dohring, M. E., Fredrickson, E., Nethery, J. & Ruff, R. L. 2005. Response to upper-limb robotics and functional neuromuscular stimulation following stroke. *Journal of Rehabilitation Research and Development,* 42**,** 723-736.

Darst, B. F., Malecki, K. C. & Engelman, C. D. 2018. Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC genetics,* 19**,** 1-6.

Dash, M. & Liu, H. 1997. Feature selection for classification. *Intelligent data analysis,* 1**,** 131-156.

Dean, J. & Ghemawat, S. 2008. MapReduce: simplified data processing on large clusters. *Communications of the ACM,* 51**,** 107-113.

Decharms, R. C., Maeda, F., Glover, G. H., Ludlow, D., Pauly, J. M., Soneji, D., Gabrieli, J. D. E. & Mackey, S. C. 2005. Control over brain activation and pain learned by using real-time functional MRI. *Proceedings of the National Academy of Sciences of the United States of America,* 102**,** 18626-18631.

Deng, H., Runger, G., Tuv, E. & Vladimir, M. 2013. A time series forest for classification and feature extraction. *Information Sciences,* 239**,** 142-153.

Deng, Z., Chen, D., Hu, Y., Wu, X., Peng, W. & Li, X. 2012. Massively parallel non-stationary EEG data processing on GPGPU platforms with Morlet continuous wavelet transform. *Journal of Internet Services and Applications,* 3**,** 347-357.

Desimone, R. & Duncan, J. 1995. Neural mechanisms of selective visual attention. *Annual review of neuroscience,* 18**,** 193-222.

Dhanya, R., Paul, I. R., Akula, S. S., Sivakumar, M. & Nair, J. J. 2020. F-test feature selection in Stacking ensemble model for breast cancer prediction. *Procedia Computer Science,* 171**,** 1561-1570.

Dias, N. S., Kamrunnahar, M., Mendes, P. M., Schiff, S. J. & Correia, J. H. 2010. Feature selection on movement imagery discrimination and attention detection. *Med Biol Eng Comput,* 48**,** 331-41.

Donchin, E. 1981. Presidential address, 1980. Surprise!...Surprise? *Psychophysiology,* 18**,** 493-513.

Donchin, E. & Coles, M. G. H. 1988. Is the P300 Component a Manifestation of Context Updating. *Behavioral and Brain Sciences,* 11**,** 357-374.

Donchin, E., Spencer, K. M. & Wijesinghe, R. 2000. The mental prosthesis: Assessing the speed of a P300-based brain-computer interface. *Ieee Transactions on Rehabilitation Engineering,* 8**,** 174-179.

Dornhege, G., Blankertz, B., Krauledat, M., Losch, F., Curio, G. & Muller, K. R. 2006. Combined optimization of spatial and temporal filters for improving brain-computer interfacing. *Ieee Transactions on Biomedical Engineering,* 53**,** 2274-2281.

Doud, A. J., Lucas, J. P., Pisansky, M. T. & He, B. 2011. Continuous Three-Dimensional Control of a Virtual Helicopter Using a Motor Imagery Based Brain-Computer Interface. *Plos One,* 6.

Duan, L., Ge, H., Ma, W. & Miao, J. 2015. EEG feature selection method based on decision tree. *Bio-medical materials and engineering,* 26**,** S1019-S1025.

Edelman, B. J., Meng, J., Suma, D., Zurn, C., Nagarajan, E., Baxter, B., Cline, C. C. & He, B. 2019. Noninvasive neuroimaging enhances continuous neural tracking for robotic device control. *Science robotics,* 4.

Erguzel, T. T., Ozekes, S., Tan, O. & Gultekin, S. 2015. Feature selection and classification of electroencephalographic signals: an artificial neural network and genetic algorithm based approach. *Clinical EEG and neuroscience,* 46**,** 321-326.

Escobar, J. J., Ortega, J., Díaz, A. F., González, J. & Damas, M. 2019. Time-energy analysis of multilevel parallelism in heterogeneous clusters: the case of EEG classification in BCI tasks. *The Journal of Supercomputing,* 75**,** 3397-3425.

Escobar, J. J., Ortega, J., González, J. & Damas, M. Assessing parallel heterogeneous computer architectures for multiobjective feature selection on EEG classification. International Conference on Bioinformatics and Biomedical Engineering, 2016a. Springer, 277-289.

Escobar, J. J., Ortega, J., González, J. & Damas, M. Improving memory accesses for heterogeneous parallel multi-objective feature selection on EEG classification. European Conference on Parallel Processing, 2016b. Springer, 372-383.

Estepp, J. R., Christensen, J. C., Monnin, J. W., Davis, I. M. & Wilson, G. F. 2009. Validation of a Dry Electrode System for EEG. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* 53**,** 1171-1175.

Eva, O. D. & Lazar, A. M. 2015. Comparison of classifiers and statistical analysis for EEG signals used in brain computer interface motor task paradigm. *International Journal of Advanced Research in Artificial Intelligence (IJARAI),* 1**,** 8-12.

Fahimi, F., Zhang, Z., Goh, W. B., Lee, T.-S., Ang, K. K. & Guan, C. 2019. Inter-subject transfer learning with an end-to-end deep convolutional neural network for EEG-based BCI. *Journal of neural engineering,* 16**,** 026007.

Farwell, L. A. & Donchin, E. 1988. Talking Off the Top of Your Head - toward a Mental Prosthesis Utilizing Event-Related Brain Potentials. *Electroencephalography and Clinical Neurophysiology,* 70**,** 510-523.

Ferdjallah, M. & Barr, R. E. 1994. Adaptive digital notch filter design on the unit circle for the removal of powerline noise from biomedical signals. *IEEE Transactions on Biomedical Engineering,* 41**,** 529-536.

Fiebig, K.-H., Jayaram, V., Peters, J. & Grosse-Wentrup, M. Multi-task logistic regression in brain-computer interfaces. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2016. IEEE, 002307-002312.

Fisch, B. J. & Spehlman, R. 2008. *Fisch and Spehlmann's EEG Primer: Basic Principles of Digital and Analog EEG*, Elsevier.

Flexer, A., Bauer, H., Pripfl, J. & Dorffner, G. 2005. Using ICA for removal of ocular artifacts in EEG recorded from blind subjects. *Neural Networks,* 18**,** 998-1005.

Fonti, V. & Belitser, E. 2017. Feature selection using lasso. *VU Amsterdam Research Paper in Business Analytics,* 30**,** 1-25.

Furdea, A., Halder, S., Krusienski, D., Bross, D., Nijboer, F., Birbaumer, N. & Kübler, A. 2009. An auditory oddball (P300) spelling system for brain-computer interfaces. *Psychophysiology,* 46**,** 617-625.

Gao, J. F., Yang, Y., Lin, P., Wang, P. & Zheng, C. X. 2010. Automatic removal of eye-movement and blink artifacts from EEG signals. *Brain Topogr,* 23**,** 105-14.

Gao, X. R., Xu, D. F., Cheng, M. & Gao, S. K. 2003. A BCI-based environmental controller for the motion-disabled. *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 11**,** 137-140.

Gargiulo, G., Calvo, R. A., Bifulco, P., Cesarelli, M., Jin, C., Mohamed, A. & Van Schaik, A. 2010. A new EEG recording system for passive dry electrodes. *Clinical Neurophysiology,* 121**,** 686-693.

Georgopoulos, A. P., Karageorgiou, E., Leuthold, A. C., Lewis, S. M., Lynch, J. K., Alonso, A. A., Aslam, Z., Carpenter, A. F., Georgopoulos, A., Hemmy, L. S., Koutlas, I. G., Langheim, F. J. P., Mccarten, J. R., Mcpherson, S. E., Pardo, J. V., Pardo, P. J., Parry, G. J., Rottunda, S. J., Segal, B. M., Sponheim, S. R., Stanwyck, J. J., Stephane, M. & Westermeyer, J. J. 2007. Synchronous neural interactions assessed by magnetoencephalography: a functional biomarker for brain disorders. *Journal of Neural Engineering,* 4**,** 349-355.

Gernert, M., Richter, A., Rundfeldt, C. & Löscher, W. 1998. Quantitative EEG analysis of depth electrode recordings from several brain regions of mutant

hamsters with paroxysmal dystonia discloses frequency changes in the basal ganglia. *Movement Disorders,* 13**,** 509-521.

Goncharova, I. I., Mcfarland, D. J., Vaughan, T. M. & Wolpaw, J. R. 2003. EMG contamination of EEG: spectral and topographical characteristics. *Clinical Neurophysiology,* 114**,** 1580-1593.

Gratton, G., Coles, M. G. H. & Donchin, E. 1983. A New Method for Off-Line Removal of Ocular Artifact. *Electroencephalography and Clinical Neurophysiology,* 55**,** 468-484.

Gropp, W., Gropp, W. D., Lusk, E., Skjellum, A. & Lusk, A. D. F. E. E. 1999. *Using MPI: portable parallel programming with the message-passing interface*, MIT press.

Grosse-Wentrup, M. & Buss, M. 2008. Multiclass common spatial patterns and information theoretic feature extraction. *Ieee Transactions on Biomedical Engineering,* 55**,** 1991-2000.

Gulo, C. A., Sementille, A. C. & Tavares, J. M. R. 2019. Techniques of medical image processing and analysis accelerated by high-performance computing: a systematic literature review. *Journal of Real-Time Image Processing***,** 1-18.

Guo, F., Hong, B., Gao, X. R. & Gao, S. K. 2008. A brain-computer interface using motion-onset visual evoked potential. *Journal of Neural Engineering,* 5**,** 477-485.

Guo, J., Gao, S. & Hong, B. 2010. An auditory brain–computer interface using active mental response. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 18**,** 230-235.

Guyon, I. & Elisseeff, A. 2003. An introduction to variable and feature selection. *Journal of machine learning research,* 3**,** 1157-1182.

Gwin, J. T., Gramann, K., Makeig, S. & Ferris, D. P. 2010. Removal of Movement Artifact From High-Density EEG Recorded During Walking and Running. *Journal of Neurophysiology,* 103**,** 3526-3534.

Hajinoroozi, M., Mao, Z. & Huang, Y. Prediction of driver's drowsy and alert states from EEG signals with deep learning. 2015 IEEE 6th international workshop on computational advances in multi-sensor adaptive processing (CAMSAP), 2015. IEEE, 493-496.

Hamedi, M., Salleh, S.-H., Noor, A. M. & Mohammad-Rezazadeh, I. Neural network-based three-class motor imagery classification using time-domain features for BCI applications. 2014 IEEE Region 10 Symposium, 2014. IEEE, 204-207.

Hammer, E. M., Halder, S., Blankertz, B., Sannelli, C., Dickhaus, T., Kleih, S., Müller, K.-R. & Kübler, A. 2012. Psychological predictors of SMR-BCI performance. *Biological psychology,* 89**,** 80-86.

Hansen, P. B. 1973. Concurrent programming concepts. *ACM Computing Surveys (CSUR),* 5**,** 223-245.

Hansen, P. B. 1978. Distributed processes: A concurrent programming concept. *Communications of the ACM,* 21**,** 934-941.

Hanslmayr, S., Sauseng, P., Doppelmayr, M., Schabus, M. & Klimesch, W. 2005. Increasing individual upper alpha power by neurofeedback improves cognitive performance in human subjects. *Applied Psychophysiology and Biofeedback,* 30**,** 1-10.

Harland, C. J., Clark, T. D. & Prance, R. J. 2002. Remote detection of human electroencephalograms using ultrahigh input impedance electric potential sensors. *Applied Physics Letters,* 81**,** 3284-3286.

Hayes, J. P. J. P. 1998. *Computer architecture and organization*, WCB/McGraw-Hill.

He, B., Gao, S., Yuan, H. & Wolpaw, J. R. 2013. Brain–Computer Interfaces. *In:* HE, B. (ed.) *Neural Engineering.* Boston, MA: Springer US.

He, L., Hu, D., Wan, M., Wen, Y., Von Deneen, K. M. & Zhou, M. 2015. Common Bayesian network for classification of EEG-based multiclass motor imagery BCI. *IEEE Transactions on Systems, man, and cybernetics: systems,* 46**,** 843-854.

He, S., Zhou, Y., Yu, T., Zhang, R., Huang, Q., Chuai, L., Gu, Z., Yu, Z. L., Tan, H. & Li, Y. 2019. EEG-and EOG-based asynchronous hybrid BCI: a system integrating a speller, a web browser, an e-mail client, and a file explorer. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 28**,** 519-530.

Hellström, B., Karlsson, B. & Müssbichler, H. 1963. Electrode placement in EEG of infants and its anatomical relationship studied radiographically. *Electroencephalography and clinical neurophysiology,* 15**,** 115-117.

Hidalgo-Muñoz, A. R., López, M., Galvao-Carmona, A., Pereira, A. T., Santos, I. M., Vázquez-Marrufo, M. & Tomé, A. M. 2014. EEG study on affective valence elicited by novel and familiar pictures using ERD/ERS and SVM-RFE. *Medical & biological engineering & computing,* 52**,** 149-158.

Hill, N. J., Lal, T. N., Bierig, K., Birbaumer, N. & Scholkopf, B. Attention modulation of auditory event-related potentials in a brain-computer interface. Biomedical Circuits and Systems, 2004 IEEE International Workshop on, 2004. IEEE, S3/5/INV-S3/17.

Hinterberger, T., Hill, J. & Birbaumer, N. An auditory brain-computer communication device. Biomedical Circuits and Systems, 2004 IEEE International Workshop on, 2004a. IEEE, S3/6-15.

Hinterberger, T., Kubler, A., Kaiser, J., Neumann, N. & Birbaumer, N. 2003. A brain-computer interface (BCI) for the locked-in: comparison of different EEG classifications for the thought translation device. *Clin Neurophysiol,* 114**,** 416-25.

Hinterberger, T., Schmidt, S., Neumann, N., Mellinger, J., Blankertz, B., Curio, G. & Birbaumer, N. 2004b. Brain-computer communication and slow cortical potentials. *IEEE Transactions on Biomedical Engineering,* 51**,** 1011-1018.

Hjorth, B. 1975. An on-line transformation of EEG scalp potentials into orthogonal source derivations. *Electroencephalography and clinical neurophysiology,* 39**,** 526-530.

Hjorth, B. 1980. Source derivation simplifies topographical EEG interpretation. *American Journal of EEG Technology,* 20**,** 121-132.

Hong, B., Guo, F., Liu, T., Gao, X. R. & Gao, S. K. 2009. N200-speller using motion-onset visual response. *Clinical Neurophysiology,* 120**,** 1658-1666.

Hosseini, M.-P., Pompili, D., Elisevich, K. & Soltanian-Zadeh, H. 2017. Optimized deep learning for EEG big data and seizure prediction BCI via internet of things. *IEEE Transactions on Big Data,* 3**,** 392-404.

Hou, H.-R., Meng, Q.-H., Zeng, M. & Sun, B. 2017. Improving classification of slow cortical potential signals for BCI systems with polynomial fitting and voting support vector machine. *IEEE Signal Processing Letters,* 25**,** 283-287.

Huang, K. J., Chang, J. C., Feng, C. W. & Fang, W. C. A parallel VLSI architecture of singular value decomposition processor for real-time multi-channel EEG system. Proceedings of the International Symposium on Consumer Electronics, ISCE, 2013. 21-22.

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. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society of London A: mathematical, physical and engineering sciences, 1998. The Royal Society, 903-995.

Hwang, H.-J. & Choi, S.-I. 2019. Effects of Different Re-referencing Methods on Spontaneously Generated Ear-EEG. *Frontiers in neuroscience,* 13**,** 822.

Hyvärinen, A. & Oja, E. 2000. Independent component analysis: algorithms and applications. *Neural networks,* 13**,** 411-430.

Iacoviello, D., Petracca, A., Spezialetti, M. & Placidi, G. 2015. A real-time classification algorithm for EEG-based BCI driven by self-induced emotions. *Computer methods and programs in biomedicine,* 122**,** 293-303.

Iatsenko, D., Mcclintock, P. V. & Stefanovska, A. 2015. Nonlinear mode decomposition: a noise-robust, adaptive decomposition method. *Physical Review E,* 92**,** 032916.

Im, C. & Seo, J.-M. 2016. A review of electrodes for the electrical brain signal recording. *Biomedical Engineering Letters,* 6**,** 104-112.

Iniewski, K. 2008. *VLSI circuits for biomedical applications*, Artech House.

Isa, N. M., Amir, A., Ilyas, M. & Razalli, M. 2019. Motor imagery classification in Brain computer interface (BCI) based on EEG signal by using machine learning technique. *Bulletin of Electrical Engineering and Informatics,* 8**,** 269-275.

İşcan, Z. & Nikulin, V. V. 2018. Steady state visual evoked potential (SSVEP) based brain-computer interface (BCI) performance under different perturbations. *PloS one,* 13**,** e0191673.

Jacquemet, V., Kappenberger, L. & Henriquez, C. S. 2008. Modeling atrial arrhythmias: impact on clinical diagnosis and therapies. *IEEE reviews in biomedical engineering,* 1**,** 94-114.

Jasper, H. H. 1958. The ten twenty electrode system of the international federation. *Electroencephalography and Clinical Neurophysiology,* 10**,** 371-375.

Jia, C. A., Gao, X. R., Hong, B. & Gao, S. K. 2011. Frequency and Phase Mixed Coding in SSVEP-Based Brain-Computer Interface. *Ieee Transactions on Biomedical Engineering,* 58**,** 200-206.

Jin, J., Allison, B. Z., Sellers, E. W., Brunner, C., Horki, P., Wang, X. Y. & Neuper, C. 2011. An adaptive P300-based control system. *Journal of Neural Engineering,* 8.

Juhasz, Z. & Kozmann, G. 2016. A GPU-based soft real-time system for simultaneous EEG processing and visualization. *Scalable Computing,* 17**,** 61-78.

Jung, T. P., Makeig, S., Westerfield, M., Townsend, J., Courchesne, E. & Sejnowski, T. J. 2000. Removal of eye activity artifacts from visual event-related potentials in normal and clinical subjects. *Clin Neurophysiol,* 111**,** 1745-58.

Kang, J.-H., Lee, C. H. & Kim, S.-P. EEG feature selection and the use of Lyapunov exponents for EEG-based biometrics. 2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), 2016. IEEE, 228-231.

Kaur, A. 2021. Wheelchair control for disabled patients using EMG/EOG based human machine interface: a review. *Journal of Medical Engineering & Technology,* 45**,** 61-74.

Kawanabe, M., Krauledat, M. & Blankertz, B. 2006. *A bayesian approach for adaptive BCI classification*, Citeseer.

Kelly, S. P., Lalor, E. C., Finucane, C., Mcdarby, G. & Reilly, R. B. 2005a. Visual spatial attention control in an independent brain-computer interface. *IEEE Transactions on Biomedical Engineering,* 52**,** 1588-1596.

Kelly, S. P., Lalor, E. C., Reilly, R. B. & Foxe, J. J. 2005b. Visual spatial attention tracking using high-density SSVEP data for independent brain-computer communication. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 13**,** 172-178.

Keren, A. S., Yuval-Greenberg, S. & Deouell, L. Y. 2010. Saccadic spike potentials in gamma-band EEG: Characterization, detection and suppression. *NeuroImage,* 49**,** 2248-2263.

Kirk, D. B. & Hwu, W.-M. W. 2010. *Programming Massively Parallel Processors: A Hands-on Approach*, Morgan Kaufmann Publishers Inc.

Kitamura, K., Takizawa, Y., Nishitani, A., Futamata, H., Matsubara, F., Furuta, T., Jibiki, I. & Yamaguchi, N. 1991. EEG data filing system with personal computer and magneto-optical disc. *Rinsho byori. The Japanese journal of clinical pathology,* 39**,** 853-858.

Kleber, B. & Birbaumer, N. 2005. Direct brain communication: neuroelectric and metabolic approaches at Tübingen. *Cognitive Processing,* 6**,** 65-74.

Kluge, T. & Hartmann, M. Phase coherent detection of steady-state evoked potentials: experimental results and application to brain-computer interfaces. Neural Engineering, 2007. CNE'07. 3rd International IEEE/EMBS Conference on, 2007. IEEE, 425-429.

Koga, K., Nakayama, I. & Kobayashi, J. Portable biological signal measurement system for biofeedback and experiment for functional assessment. International Conference on Control, Automation and Systems, 2013. 412-416.

Krishna, C. M. 1999. *Real-Time Systems*, Wiley Online Library.

Krusienski, D. J., Mcfarland, D. J. & Wolpaw, J. R. 2006. An evaluation of autoregressive spectral estimation model order for brain-computer interface applications. *2006 28th Annual International Conference of the Ieee Engineering in Medicine and Biology Society, Vols 1-15***,** 2302-+.

Krusienski, D. J., Schalk, G., Mcfarland, D. J. & Wolpaw, J. R. 2007. A mu-rhythm matched filter for continuous control of a brain-computer interface. *IEEE Trans Biomed Eng,* 54**,** 273-80.

Kübler, A. 2020. The history of BCI: From a vision for the future to real support for personhood in people with locked-in syndrome. *Neuroethics,* 13**,** 163-180.

Kubler, A., Furdea, A., Halder, S., Hammer, E. M., Nijboer, F. & Kotchoubey, B. 2009. A brain-computer interface controlled auditory event-related potential (p300) spelling system for locked-in patients. *Ann N Y Acad Sci,* 1157**,** 90-100.

Kus, R., Valbuena, D., Zygierewicz, J., Malechka, T., Graeser, A. & Durka, P. 2012. Asynchronous BCI based on motor imagery with automated calibration and neurofeedback training. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 20**,** 823-835.

Lakany, H. & Conway, B. A. 2007. Understanding intention of movement from electroencephalograms. *Expert Systems,* 24**,** 295-304.

Laszlo, S., Ruiz-Blondet, M., Khalifian, N., Chu, F. & Jin, Z. 2014. A direct comparison of active and passive amplification electrodes in the same amplifier system. *Journal of Neuroscience Methods,* 235**,** 298-307.

Law, A. M. & Kelton, D. W. 2000. Simulation modeling and analysis.

Lazarou, I., Nikolopoulos, S., Petrantonakis, P. C., Kompatsiaris, I. & Tsolaki, M. 2018. EEG-based brain–computer interfaces for communication and rehabilitation of people with motor impairment: a novel approach of the 21st century. *Frontiers in human neuroscience,* 12**,** 14.

Leardi, R. 2000. Application of genetic algorithm–PLS for feature selection in spectral data sets. *Journal of Chemometrics: A Journal of the Chemometrics Society,* 14**,** 643-655.

Lecun, Y., Bengio, Y. & Hinton, G. 2015. Deep learning. *nature,* 521**,** 436-444.

Lee, B.-G., Lee, B.-L. & Chung, W.-Y. 2014. Mobile healthcare for automatic driving sleep-onset detection using wavelet-based EEG and respiration signals. *Sensors,* 14**,** 17915-17936.

Lee, P.-L., Hsieh, J.-C., Wu, C.-H., Shyu, K.-K. & Wu, Y.-T. 2008. Brain computer interface using flash onset and offset visual evoked potentials. *Clinical Neurophysiology,* 119**,** 605-616.

Lee, P. L., Hsieh, J. C., Wu, C. H., Shyu, K. K., Chen, S. S., Yeh, T. C. & Wu, Y. T. 2006. The brain computer interface using flash visual evoked potential and independent component analysis. *Ann Biomed Eng,* 34**,** 1641-54.

Lee, S.-J., Lee, J., Kim, H., Yun, S.-W., Lee, T. & Hong, J. 2021. Design for parallel computation of model-based signal processing in Thomson scattering diagnostic. *Fusion Engineering and Design,* 171**,** 112546.

Lemm, S., Blankertz, B., Curio, G. & Muller, K. R. 2005. Spatio-spectral filters for improving the classification of single trial EEG. *Ieee Transactions on Biomedical Engineering,* 52**,** 1541-1548.

Li, G., Lee, B.-L. & Chung, W.-Y. 2015. Smartwatch-based wearable EEG system for driver drowsiness detection. *IEEE Sensors Journal,* 15**,** 7169-7180.

Li, K., Sankar, R., Arbel, Y. & Donchin, E. Single trial independent component analysis for P300 BCI system. 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 3-6 Sept. 2009 2009. 4035-4038.

Li, P. Z. X., Kassiri, H. & Genov, R. A compact low-power VLSI architecture for real-time sleep stage classification. Proceedings - IEEE International Symposium on Circuits and Systems, 2016. 1314-1317.

Li, X., Song, D., Zhang, P., Zhang, Y., Hou, Y. & Hu, B. 2018a. Exploring EEG features in cross-subject emotion recognition. *Frontiers in neuroscience,* 12**,** 162.

Li, Z., Li, J., Zhao, S., Yuan, Y., Kang, Y. & Chen, C. P. 2018b. Adaptive neural control of a kinematically redundant exoskeleton robot using brain–machine interfaces. *IEEE transactions on neural networks and learning systems,* 30**,** 3558-3571.

Liang, S.-F., Kuo, C.-E., Hu, Y.-H., Pan, Y.-H. & Wang, Y.-H. 2012. Automatic stage scoring of single-channel sleep EEG by using multiscale entropy and autoregressive models. *IEEE Transactions on Instrumentation and Measurement,* 61**,** 1649-1657.

Liao, L.-D., Wang, I.-J., Chen, S.-F., Chang, J.-Y. & Lin, C.-T. 2011. Design, Fabrication and Experimental Validation of a Novel Dry-Contact Sensor for Measuring Electroencephalography Signals without Skin Preparation. *Sensors,* 11**,** 5819.

Liao, Y. & Vemuri, V. R. 2002. Use of k-nearest neighbor classifier for intrusion detection. *Computers & security,* 21**,** 439-448.

Lin, C.-J. & Hsieh, M.-H. 2009. Classification of mental task from EEG data using neural networks based on particle swarm optimization. *Neurocomputing,* 72**,** 1121-1130.

Lin, C. T., Liao, L. D., Liu, Y. H., Wang, I. J., Lin, B. S. & Chang, J. Y. 2011. Novel Dry Polymer Foam Electrodes for Long-Term EEG Measurement. *IEEE Transactions on Biomedical Engineering,* 58**,** 1200-1207.

Lin, Y. P., Wang, Y. & Jung, T. P. A mobile SSVEP-based brain-computer interface for freely moving humans: The robustness of canonical correlation analysis to motion artifacts. 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 3-7 July 2013 2013. 1350-1353.

Lins, O. G., Picton, T. W., Berg, P. & Scherg, M. 1993. Ocular artifacts in recording EEGs and event-related potentials. II: Source dipoles and source components. *Brain Topogr,* 6**,** 65-78.

Litwin, L. 2000. FIR and IIR digital filters. *IEEE potentials,* 19**,** 28-31.

Liu, Y., Zhou, Z. & Hu, D. 2011. Gaze independent brain–computer speller with covert visual search tasks. *Clinical Neurophysiology,* 122**,** 1127-1136.

Lopez-Gordo, A. M., Sanchez-Morillo, D. & Valle, P. F. 2014. Dry EEG Electrodes. *Sensors,* 14.

Lotte, F., Bougrain, L., Cichocki, A., Clerc, M., Congedo, M., Rakotomamonjy, A. & Yger, F. 2018. A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update. *Journal of neural engineering,* 15**,** 031005.

Lutzenberger, W., Elbert, T., Rockstroh, B. & Birbaumer, N. 1982. Biofeedback produced slow brain potentials and task performance. *Biological Psychology,* 14**,** 99-111.

Ma, Z., Tan, Z.-H. & Guo, J. 2016. Feature selection for neutral vector in EEG signal classification. *Neurocomputing,* 174**,** 937-945.

Madoš, B., Ádám, N., Hurtuk, J. & Čopjak, M. Brain-computer interface and Arduino microcontroller family software interconnection solution. SAMI 2016 - IEEE

14th International Symposium on Applied Machine Intelligence and Informatics - Proceedings, 2016. 217-221.

Mainsah, B., Collins, L., Colwell, K., Sellers, E., Ryan, D., Caves, K. & Throckmorton, C. 2015. Increasing BCI communication rates with dynamic stopping towards more practical use: an ALS study. *Journal of neural engineering,* 12**,** 016013.

Mason, S. G. & Birch, G. E. 2000. A brain-controlled switch for asynchronous control applications. *IEEE Trans Biomed Eng,* 47**,** 1297-307.

Mathewson, K. E., Harrison, T. J. L. & Kizuk, S. a. D. 2017. High and dry? Comparing active dry EEG electrodes to active and passive wet electrodes. *Psychophysiology,* 54**,** 74-82.

Mayaud, L., Congedo, M., Van Laghenhove, A., Orlikowski, D., Figère, M., Azabou, E. & Cheliout-Heraut, F. 2013. A comparison of recording modalities of P300 event-related potentials (ERP) for brain-computer interface (BCI) paradigm. *Neurophysiologie Clinique/Clinical Neurophysiology,* 43**,** 217-227.

Mcafee, S. S., Liu, Y., Sillitoe, R. V. & Heck, D. H. 2019. Cerebellar lobulus simplex and crus I differentially represent phase and phase difference of prefrontal cortical and hippocampal oscillations. *Cell reports,* 27**,** 2328-2334. e3.

Mcfarland, D. J., Miner, L. A., Vaughan, T. M. & Wolpaw, J. R. 2000. Mu and beta rhythm topographies during motor imagery and actual movements. *Brain Topography,* 12**,** 177-186.

Mcfarland, D. J., Sarnacki, W. A. & Wolpaw, J. R. 2010. Electroencephalographic (EEG) control of three-dimensional movement. *Journal of Neural Engineering,* 7.

Mcmenamin, B. W., Shackman, A. J., Greischar, L. L. & Davidson, R. J. 2011. Electromyogenic artifacts and electroencephalographic inferences revisited. *Neuroimage,* 54**,** 4-9.

Mcmenamin, B. W., Shackman, A. J., Maxwell, J. S., Bachhuber, D. R. W., Koppenhaver, A. M., Greischar, L. L. & Davidson, R. J. 2010. Validation of ICA-based myogenic artifact correction for scalp and source-localized EEG. *Neuroimage,* 49**,** 2416-2432.

Meng, J., Zhang, S., Bekyo, A., Olsoe, J., Baxter, B. & He, B. 2016. Noninvasive Electroencephalogram Based Control of a Robotic Arm for Reach and Grasp Tasks. 6**,** 38565.

Middendorf, M., Mcmillan, G., Calhoun, G. & Jones, K. S. 2000. Brain-computer interfaces based on the steady-state visual-evoked response. *Ieee Transactions on Rehabilitation Engineering,* 8**,** 211-214.

Miller, K. J., Schalk, G., Fetz, E. E., Den Nijs, M., Ojemann, J. G. & Rao, R. P. 2010. Cortical activity during motor execution, motor imagery, and imagery-based online feedback. *Proceedings of the National Academy of Sciences,* 107**,** 4430-4435.

Minguillon, J., Lopez-Gordo, M. A. & Pelayo, F. 2017. Trends in EEG-BCI for daily-life: Requirements for artifact removal. *Biomedical Signal Processing and Control,* 31**,** 407-418.

Mirza, I. A., Tripathy, A., Chopra, S., D'sa, M., Rajagopalan, K., D'souza, A. & Sharma, N. Mind-controlled wheelchair using an EEG headset and arduino

microcontroller. Proceedings - International Conference on Technologies for Sustainable Development, ICTSD 2015, 2015.

Mohammadpour, M., Hashemi, S. M. R. & Houshmand, N. Classification of EEG-based emotion for BCI applications. 2017 Artificial Intelligence and Robotics (IRANOPEN), 2017. IEEE, 127-131.

Momose, K. 2007. Evaluation of an eye gaze point detection method using VEP elicited by multi-pseudorandom stimulation for brain computer interface. *2007 Annual International Conference of the Ieee Engineering in Medicine and Biology Society, Vols 1-16*, 5063-5066.

Moritz, C. T., Perlmutter, S. I. & Fetz, E. E. 2008. Direct control of paralysed muscles by cortical neurons. *Nature,* 456, 639-U63.

Mousavi, E. A., Maller, J. J., Fitzgerald, P. B. & Lithgow, B. J. 2011. Wavelet Common Spatial Pattern in asynchronous offline brain computer interfaces. *Biomedical Signal Processing and Control,* 6, 121-128.

Muthukumaraswamy, S. 2013. High-frequency brain activity and muscle artifacts in MEG/EEG: A review and recommendations. *Frontiers in Human Neuroscience,* 7.

Nicolas-Alonso, L. F. & Gomez-Gil, J. 2012. Brain Computer Interfaces, a Review. *Sensors (Basel, Switzerland),* 12, 1211-1279.

Nieva, Peralta, E. G., Beltramone, M. F. & Antonio, D. Home automation by brain-computer interface. 3rd International Conference on Advances in New Technologies, Interactive Interfaces and Communicability, ADNTIIC 2012: Design, E-Commerce, E-Learning, E-Health, E-Tourism, Web 2.0 and Web 3.0, 2012.

Noergaard, T. 2005. *Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers*, Newnes.

Oehler, M., Neumann, P., Becker, M., Curio, G. & Schilling, M. Extraction of SSVEP signals of a capacitive EEG helmet for Human Machine Interface. 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 20-25 Aug. 2008 2008. 4495-4498.

Olkkonen, H., Pesola, P., Olkkonen, J., Valjakka, A. & Tuomisto, L. 2002. EEG noise cancellation by a subspace method based on wavelet decomposition. *Medical Science Monitor,* 8, MT199-MT204.

Oostenveld, R. & Praamstra, P. 2001. The five percent electrode system for high-resolution EEG and ERP measurements. *Clinical Neurophysiology,* 112, 713-719.

Ortner, R., Allison, B. Z., Korisek, G., Gaggl, H. & Pfurtscheller, G. 2011. An SSVEP BCI to Control a Hand Orthosis for Persons With Tetraplegia. *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 19, 1-5.

Oweiss, K. G. 2006. A systems approach for data compression and latency reduction in cortically controlled brain machine interfaces. *IEEE Transactions on Biomedical Engineering,* 53, 1364-1377.

Pan, J., Gao, X., Duan, F., Yan, Z. & Gao, S. 2011. Enhancing the classification accuracy of steady-state visual evoked potential-based brain–computer interfaces using phase constrained canonical correlation analysis. *Journal of neural engineering,* 8, 036027.

Pandarinath, C., Nuyujukian, P., Blabe, C. H., Sorice, B. L., Saab, J., Willett, F. R., Hochberg, L. R., Shenoy, K. V. & Henderson, J. M. 2017. High performance communication by people with paralysis using an intracortical brain-computer interface. *Elife,* 6**,** e18554.

Paszkiel, S., Hunek, W. & Shylenko, A. 2016. Project and simulation of a portable device for measuring bioelectrical signals from the brain for states consciousness verification with visualization on LEDs. *Advances in Intelligent Systems and Computing.*

Pauwels, K., Tomasi, M., Alonso, J. D., Ros, E. & Van Hulle, M. M. 2011. A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features. *IEEE Transactions on Computers,* 61**,** 999-1012.

Pavan, K. E. & Rajesh, G. N. A novalized VLSI design and implementation of EEG signal acquisition system. 2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings, 2017. 1955-1959.

Peker, M., Şen, B. & Gürüler, H. 2015. Rapid automated classification of anesthetic depth levels using GPU based parallelization of neural networks. *Journal of medical systems,* 39**,** 18.

Pfurtscheller, G. & Da Silva, F. H. L. 1999. Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clinical Neurophysiology,* 110**,** 1842-1857.

Pfurtscheller, G., Flotzinger, D. & Kalcher, J. 1993. Brain Computer-Interface - a New Communication Device for Handicapped Persons. *Journal of Microcomputer Applications,* 16**,** 293-299.

Pfurtscheller, G., Leeb, R., Keinrath, C., Friedman, D., Neuper, C., Guger, C. & Slaterc, M. 2006. Walking from thought. *Brain Research,* 1071**,** 145-152.

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

Pfurtscheller, G., Neuper, C., Muller, G., Obermaier, B., Krausz, G., Schlogl, A., Scherer, R., Graimann, B., Keinrath, C. & Skliris, D. 2003. Graz-BCI: state of the art and clinical applications. *IEEE Transactions on neural systems and rehabilitation engineering,* 11**,** 1-4.

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,* 19**,** 206-218.

Pham, T. T. H., Croft, R. J., Cadusch, P. J. & Barry, R. J. 2011. A test of four EOG correction methods using an improved validation technique. *International Journal of Psychophysiology,* 79**,** 203-210.

Phan, J. H., Quo, C. F., Cheng, C. & Wang, M. D. 2012. Multiscale integration of-omic, imaging, and clinical data in biomedical informatics. *IEEE reviews in biomedical engineering,* 5**,** 74-87.

Pieloth, C., Pizarro, J. M., Knosche, T., Maess, B. & Fuchs, M. An online system for neuroelectromagnetic source imaging. Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS 2013, 2013. 270-274.

Pires, G., Castelo-Branco, M. & Nunes, U. Visual P300-based BCI to steer a wheelchair: a Bayesian approach. 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2008. IEEE, 658-661.

Pisarenco, I., Caporro, M., Prosperetti, C. & Manconi, M. 2014. High-density electroencephalography as an innovative tool to explore sleep physiology and sleep related disorders. *International Journal of Psychophysiology,* 92**,** 8-15.

Posner, M. I. & Dehaene, S. 1994. Attentional networks. *Trends in neurosciences,* 17**,** 75-79.

Posner, M. I. & Petersen, S. E. 1990. The attention system of the human brain. *Annual review of neuroscience,* 13**,** 25-42.

Putra, A. E., Atmaji, C. & Utami, T. G. EEG-based microsleep detector using microcontroller. Proceedings of 2016 8th International Conference on Information Technology and Electrical Engineering: Empowering Technology for Better Future, ICITEE 2016, 2017.

Rahman, M. A., Haque, M. M., Anjum, A., Mollah, M. N. & Ahmad, M. Classification of motor imagery events from prefrontal hemodynamics for BCI application. Proceedings of International Joint Conference on Computational Intelligence, 2020. Springer, 11-23.

Rakotomamonjy, A. & Guigue, V. 2008. BCI competition III: dataset II-ensemble of SVMs for BCI P300 speller. *IEEE transactions on biomedical engineering,* 55**,** 1147-1154.

Ramoser, H., Muller-Gerking, J. & Pfurtscheller, G. 2000. Optimal spatial filtering of single trial EEG during imagined hand movement. *Ieee Transactions on Rehabilitation Engineering,* 8**,** 441-446.

Reeves, S. J. & Zhe, Z. 1999. Sequential algorithms for observation selection. *IEEE Transactions on Signal Processing,* 47**,** 123-132.

Rejer, I. Genetic algorithms in EEG feature selection for the classification of movements of the left and right hand. Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, 2013. Springer, 579-589.

Renaud, P., Joyal, C., Stoleru, S., Goyette, M., Weiskopf, N. & Birbaumer, N. 2011. Real-time functional magnetic imaging-brain-computer interface and virtual reality: promising tools for the treatment of pedophilia. *Enhancing Performance for Action and Perception: Multisensory Integration, Neuroplasticity and Neuroprosthetics, Pt Ii,* 192**,** 263-272.

Ring, H. & Rosenthal, N. 2005. Controlled study of neuroprosthetic functional electrical stimulation in sub-acute post-stroke rehabilitation. *Journal of Rehabilitation Medicine,* 37**,** 32-36.

Rockstroh, B., Elbert, T., Lutzenberger, W. & Birbaumer, N. 1982. The effects of slow cortical potentials on response speed. *Psychophysiology,* 19**,** 211-217.

Rodríguez-Andina, J. J., Valdes-Pena, M. D. & Moure, M. J. 2015. Advanced features and industrial applications of FPGAs—A review. *IEEE Transactions on Industrial Informatics,* 11**,** 853-864.

Romero, S., Mananas, M. A. & Barbanoj, M. J. 2008. A comparative study of automatic techniques for ocular artifact reduction in spontaneous EEG signals

based on clinical target variables: A simulation case. *Computers in Biology and Medicine,* 38**,** 348-360.

Rossini, P. M., Burke, D., Chen, R., Cohen, L., Daskalakis, Z., Di Iorio, R., Di Lazzaro, V., Ferreri, F., Fitzgerald, P. & George, M. 2015. Non-invasive electrical and magnetic stimulation of the brain, spinal cord, roots and peripheral nerves: basic principles and procedures for routine clinical and research application. An updated report from an IFCN Committee. *Clinical Neurophysiology,* 126**,** 1071-1107.

Rota, G., Sitaram, R., Veit, R., Erb, M., Weiskopf, N., Dogil, G. & Birbaumer, N. 2009. Self-Regulation of Regional Cortical Activity Using Real-Time fMRI: The Right Inferior Frontal Gyrus and Linguistic Processing. *Human Brain Mapping,* 30**,** 1605-1614.

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. *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 18**,** 581-589.

Royer, A. S., Rose, M. L. & He, B. 2011. Goal selection versus process control while learning to use a brain-computer interface. *Journal of Neural Engineering,* 8.

Rugg, M. D. & Curran, T. 2007. Event-related potentials and recognition memory. *Trends in cognitive sciences,* 11**,** 251-257.

Saa, J. F. D. & Gutierrez, M. S. EEG signal classification using power spectral features and linear discriminant analysis: A brain computer interface application. Eighth Latin American and Caribbean Conference for Engineering and Technology, 2010. LACCEI Arequipa, 1-7.

Safieddine, D., Kachenoura, A., Albera, L., Birot, G., Karfoul, A., Pasnicu, A., Biraben, A., Wendling, F., Senhadji, L. & Merlet, I. 2012. Removal of muscle artifact from EEG data: comparison between stochastic (ICA and CCA) and deterministic (EMD and wavelet-based) approaches. *Eurasip Journal on Advances in Signal Processing*.

Salvaris, M. & Sepulveda, F. 2009. Visual modifications on the P300 speller BCI paradigm. *Journal of neural engineering,* 6**,** 046011.

Saptono, D., Wahyudi, B. & Irawan, B. Design of EEG Signal Acquisition System Using Arduino MEGA1280 and EEGAnalyzer. MATEC Web of Conferences, 2016.

Sarac, M., Koyas, E., Erdogan, A., Cetin, M. & Patoglu, V. Brain computer interface based robotic rehabilitation with online modification of task speed. 2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR), 2013. IEEE, 1-7.

Sartoretto, F. & Ermani, M. 1999. Automatic detection of epileptiform activity by single-level wavelet analysis. *Clinical Neurophysiology,* 110**,** 239-249.

Schneider, F., Elbert, T., Heimann, H., Welker, A., Stetter, F., Mattes, R., Birbaumer, N. & Mann, K. 1993. Self-Regulation of Slow Cortical Potentials in Psychiatric-Patients - Alcohol Dependency. *Biofeedback and Self-Regulation,* 18**,** 23-32.

Schneider, F., Heimann, H., Mattes, R., Lutzenberger, W. & Birbaumer, N. 1992a. Self-Regulation of Slow Cortical Potentials in Psychiatric-Patients - Depression. *Biofeedback and Self-Regulation,* 17**,** 203-214.

Schneider, F., Rockstroh, B., Heimann, H., Lutzenberger, W., Mattes, R., Elbert, T., Birbaumer, N. & Bartels, M. 1992b. Self-Regulation of Slow Cortical Potentials in Psychiatric-Patients - Schizophrenia. *Biofeedback and Self-Regulation,* 17**,** 277-292.

Sellers, E. W. & Donchin, E. 2006. A P300-based brain–computer interface: initial tests by ALS patients. *Clinical neurophysiology,* 117**,** 538-548.

Sellers, E. W., Vaughan, T. M. & Wolpaw, J. R. 2010. A brain-computer interface for long-term independent home use. *Amyotrophic Lateral Sclerosis,* 11**,** 449-455.

Senkowski, D. & Herrmann, C. S. 2002. Effects of task difficulty on evoked gamma activity and ERPs in a visual discrimination task. *Clinical Neurophysiology,* 113**,** 1742-1753.

Senyo, P. K., Addae, E. & Boateng, R. 2018. Cloud computing research: A review of research themes, frameworks, methods and future research directions. *International Journal of Information Management,* 38**,** 128-139.

Shenoy, P., Krauledat, M., Blankertz, B., Rao, R. P. & Müller, K.-R. 2006. Towards adaptive classification for BCI. *Journal of neural engineering,* 3**,** R13.

Shih, W. Y., Liao, J. C., Huang, K. J., Fang, W. C., Cauwenberghs, G. & Jung, T. P. An efficient VLSI implementation of on-line recursive ICA processor for real-time multi-channel EEG signal separation. Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, 2013. 6808-6811.

Shon, D., Im, K., Park, J.-H., Lim, D.-S., Jang, B. & Kim, J.-M. 2018. Emotional stress state detection using genetic algorithm-based feature selection on EEG signals. *International Journal of environmental research and public health,* 15**,** 2461.

Simbolon, A. I., Turnip, A., Hutahaean, J., Siagian, Y. & Irawati, N. An experiment of lie detection based EEG-P300 classified by SVM algorithm. 2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), 2015. IEEE, 68-71.

Simões, H., Pires, G., Nunes, U. & Silva, V. Feature Extraction and Selection for Automatic Sleep Staging using EEG. ICINCO (3), 2010. 128-133.

Singla, R. & Haseena, B. 2014. Comparison of ssvep signal classification techniques using svm and ann models for bci applications. *International Journal of Information and Electronics Engineering,* 4**,** 6.

Sita, J. & Nair, G. Feature extraction and classification of eeg signals for mapping motor area of the brain. 2013 International Conference on Control Communication and Computing (ICCC), 2013. IEEE, 463-468.

Sitaram, R., Lee, S., Ruiz, S., Rana, M., Veit, R. & Birbaumer, N. 2011. Real-time support vector classification and feedback of multiple emotional brain states. *Neuroimage,* 56**,** 753-765.

Sivasankari, K. & Thanushkodi, K. 2013. Development of an epileptic seizure detection application based on parallel computing. *International Journal of Engineering and Technology,* 5**,** 4590-4597.

Sörnmo, L. & Laguna, P. 2005a. Chapter 5 - The Electromyogram. *Bioelectrical Signal Processing in Cardiac and Neurological Applications.* Burlington: Academic Press.

Sörnmo, L. & Laguna, P. 2005b. Chapter 7 - ECG Signal Processing. *Bioelectrical Signal Processing in Cardiac and Neurological Applications.* Burlington: Academic Press.

Spencer, K. M., Dien, J. & Donchin, E. 2001. Spatiotemporal analysis of the late ERP responses to deviant stimuli. *Psychophysiology,* 38**,** 343-358.

Sterman, M. B. & Egner, T. 2006. Foundation and practice of neurofeedback for the treatment of epilepsy. *Applied Psychophysiology and Biofeedback,* 31**,** 21-35.

Stone, J. E., Gohara, D. & Shi, G. 2010. OpenCL: A parallel programming standard for heterogeneous computing systems. *Computing in science & engineering,* 12**,** 66-73.

Strehl, U., Leins, U., Goth, G., Klinger, C., Hinterberger, T. & Birbaumer, N. 2006. Self-regulation of slow cortical potentials: A new treatment for children with attention-deficit/hyperactivity disorder. *Pediatrics,* 118**,** E1530-E1540.

Sugumaran, V., Muralidharan, V. & Ramachandran, K. 2007. Feature selection using decision tree and classification through proximal support vector machine for fault diagnostics of roller bearing. *Mechanical systems and signal processing,* 21**,** 930-942.

Sullivan, T. J., Deiss, S. R., Tzyy-Ping, J. & Cauwenberghs, G. A brain-machine interface using dry-contact, low-noise EEG sensors. 2008 IEEE International Symposium on Circuits and Systems, 18-21 May 2008 2008. 1986-1989.

Suthaharan, S. 2016. Support vector machine. *Machine learning models and algorithms for big data classification.* Springer.

Sutter, E. 1984. The Visual Evoked-Response as a Communication Channel. *Ieee Transactions on Biomedical Engineering,* 31**,** 583-583.

Sutter, E. E. 1992. The Brain Response Interface - Communication through Visually-Induced Electrical Brain Responses. *Journal of Microcomputer Applications,* 15**,** 31-45.

Sweeney, K. T., Ward, T. E. & Mcloone, S. F. 2012. Artifact Removal in Physiological Signals-Practices and Possibilities. *Ieee Transactions on Information Technology in Biomedicine,* 16**,** 488-500.

Tabar, Y. R. & Halici, U. 2016. A novel deep learning approach for classification of EEG motor imagery signals. *Journal of neural engineering,* 14**,** 016003.

Tam, W. K., Tong, K. Y., Meng, F. & Gao, S. K. 2011. A Minimal Set of Electrodes for Motor Imagery BCI to Control an Assistive Device in Chronic Stroke Subjects: A Multi-Session Study. *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 19**,** 617-627.

Taur, Y. & Ning, T. H. 1998. *Fundamentals of modern VLSI devices*, Cambridge University Press.

Tharwat, A. 2016. Linear vs. quadratic discriminant analysis classifier: a tutorial. *International Journal of Applied Pattern Recognition,* 3**,** 145-180.

Tomioka, R., Aihara, K. & Müller, K.-R. Logistic regression for single trial EEG classification. Advances in neural information processing systems, 2007. 1377-1384.

Townsend, G., Lapallo, B. K., Boulay, C. B., Krusienski, D. J., Frye, G. E., Hauser, C. K., Schwartz, N. E., Vaughan, T. M., Wolpaw, J. R. & Sellers, E. W. 2010. A novel P300-based brain-computer interface stimulus presentation paradigm:

Moving beyond rows and columns. *Clinical Neurophysiology,* 121**,** 1109-1120.

Treder, M. S. & Blankertz, B. 2010. (C) overt attention and visual speller design in an ERP-based brain-computer interface. *Behavioral and brain functions,* 6**,** 28.

Tsui, C. S. L. & Gan, J. Q. 2007. Asynchronous BCI Control of a Robot Simulator with Supervised Online Training. *In:* YIN, H., TINO, P., CORCHADO, E., BYRNE, W. & YAO, X. (eds.) *Intelligent Data Engineering and Automated Learning - IDEAL 2007: 8th International Conference, Birmingham, UK, December 16-19, 2007. Proceedings.* Berlin, Heidelberg: Springer Berlin Heidelberg.

Turnip, A. & Soetraprawata, D. 2013. The performance of EEG-P300 classification using backpropagation neural networks. *Journal of Mechatronics, Electrical Power, and Vehicular Technology,* 4**,** 81-88.

Unser, M. & Aldroubi, A. 1996. A review of wavelets in biomedical applications. *Proceedings of the Ieee,* 84**,** 626-638.

Vecchiato, G., Fallani, F. D. V., Astolfi, L., Toppi, J., Cincotti, F., Mattia, D., Salinari, S. & Babiloni, F. 2010. The issue of multiple univariate comparisons in the context of neuroelectric brain mapping: An application in a neuromarketing experiment. *Journal of Neuroscience Methods,* 191**,** 283-289.

Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S. & Schwartz, A. B. 2008. Cortical control of a prosthetic arm for self-feeding. *Nature,* 453**,** 1098-1101.

Vergara, J. R. & Estévez, P. A. 2014. A review of feature selection methods based on mutual information. *Neural computing and applications,* 24**,** 175-186.

Vigario, R. & Oja, E. 2008. BSS and ICA in neuroinformatics: from current practices to open challenges. *IEEE Rev Biomed Eng,* 1**,** 50-61.

Walker, J. E. & Kozlowski, G. P. 2005. Neurofeedback treatment of epilepsy. *Child and Adolescent Psychiatric Clinics of North America,* 14**,** 163-+.

Wallstrom, G. L., Kass, R. E., Miller, A., Cohn, J. F. & Fox, N. A. 2004. Automatic correction of ocular artifacts in the EEG: a comparison of regression-based and component-based methods. *International journal of psychophysiology,* 53**,** 105-119.

Wang, H. & Zhang, Y. 2016. Detection of motor imagery EEG signals employing Naïve Bayes based learning process. *Measurement,* 86**,** 148-158.

Wang, J., Xu, G. Z., Wang, L. & Zhang, H. Y. 2010. Feature Extraction of Brain-Computer Interface based on Improved Multivariate Adaptive Autoregressive Models. *2010 3rd International Conference on Biomedical Engineering and Informatics (Bmei 2010), Vols 1-7***,** 895-898.

Wang, T., Deng, H. & He, B. 2004. Classifying EEG-based motor imagery tasks by means of time-frequency synthesized spatial patterns. *Clinical Neurophysiology,* 115**,** 2744-2753.

Wang, T. & He, B. 2004. An efficient rhythmic component expression and weighting synthesis strategy for classifying motor imagery EEG in a brain-computer interface. *J Neural Eng,* 1**,** 1-7.

Wang, X.-W., Nie, D. & Lu, B.-L. 2014. Emotional state classification from EEG data using machine learning approach. *Neurocomputing,* 129**,** 94-106.

Wang, Y., Gao, S. & Gao, X. Common spatial pattern method for channel selelction in motor imagery based brain-computer interface. 2005 IEEE engineering in medicine and biology 27th annual conference, 2006a. IEEE, 5392-5395.

Wang, Y., Wang, R. P., Gao, X. R., Hong, B. & Gao, S. K. 2006b. A practical VEP-based brain-computer interface. *Ieee Transactions on Neural Systems and Rehabilitation Engineering,* 14**,** 234-239.

Wilson, J. A. Using general-purpose graphic processing units for BCI systems. 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2011. IEEE, 4625-4628.

Wilson, J. J. & Palaniappan, R. 2009. Augmenting a SSVEP BCI through single cycle analysis and phase weighting. *2009 4th International Ieee/Embs Conference on Neural Engineering***,** 364-+.

Wold, S., Esbensen, K. & Geladi, P. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems,* 2**,** 37-52.

Wolpaw, J. R. 2010. Brain-Computer Interface Research Comes of Age: Traditional Assumptions Meet Emerging Realities. *Journal of Motor Behavior,* 42**,** 351-353.

Wolpaw, J. R., Birbaumer, N., Mcfarland, D. J., Pfurtscheller, G. & Vaughan, T. M. 2002. Brain–computer interfaces for communication and control. *Clinical neurophysiology,* 113**,** 767-791.

Wolpaw, J. R. & Mcfarland, D. J. 2004. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. *Proceedings of the National Academy of Sciences of the United States of America,* 101**,** 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,* 78**,** 252-259.

Won, D.-O., Hwang, H.-J., Dähne, S., Müller, K.-R. & Lee, S.-W. 2015. Effect of higher frequency on the classification of steady-state visual evoked potentials. *Journal of neural engineering,* 13**,** 016014.

Wu, S.-L., Wu, C.-W., Pal, N. R., Chen, C.-Y., Chen, S.-A. & Lin, C.-T. Common spatial pattern and linear discriminant analysis for motor imagery classification. 2013 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2013. IEEE, 146-151.

Xu, J., Huang, E., Chen, C.-H. & Lee, L. H. 2015. Simulation optimization: A review and exploration in the new era of cloud computing and big data. *Asia-Pacific Journal of Operational Research,* 32**,** 1550019.

Xu, J., Mitra, S., Hoof, C. V., Yazicioglu, R. & Makinwa, K. a. A. 2017. Active Electrodes for Wearable EEG Acquisition: Review and Electronics Design Methodology. *IEEE Reviews in Biomedical Engineering,* PP**,** 1-1.

Xu, R., Jiang, N., Lin, C., Mrachacz-Kersting, N., Dremstrup, K. & Farina, D. 2013. Enhanced low-latency detection of motor intention from EEG for closed-loop brain-computer interface applications. *IEEE Transactions on Biomedical Engineering,* 61**,** 288-296.

Yacine, B., Amal, F. & Walter, B. 2014. Significant improvement in one-dimensional cursor control using Laplacian electroencephalography over electroencephalography. *Journal of Neural Engineering,* 11**,** 035014.

Yamawaki, N., Wilke, C., Liu, Z. & He, B. 2006. An enhanced time-frequency-spatial approach for motor imagery classification. *IEEE transactions on neural systems and rehabilitation engineering,* 14**,** 250-254.

Yao, D., Wang, L., Oostenveld, R., Nielsen, K. D., Arendt-Nielsen, L. & Chen, A. C. 2005. A comparative study of different references for EEG spectral mapping: the issue of the neutral reference and the use of the infinity reference. *Physiological measurement,* 26**,** 173.

Yegnanarayana, B. 2009. *Artificial neural networks*, PHI Learning Pvt. Ltd.

Yuan, H., Doud, A., Gururajan, A. & He, B. 2008. Cortical Imaging of Event-Related (de)Synchronization During Online Control of Brain-Computer Interface Using Minimum-Norm Estimates in Frequency Domain. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 16**,** 425-431.

Yuwono, H. A., Wijaya, S. K. & Prajitno, P. Feature selection with Lasso for classification of ischemic strokes based on EEG signals. Journal of Physics: Conference Series, 2020. IOP Publishing, 012029.

Zaghloul, Z. S. & Bayoumi, M. Implementable Spike Sorting techniques for VLSI wireless BCI/BMI implants: A survey. 5th International Conference on Energy Aware Computing Systems and Applications, ICEAC 2015, 2015.

Zhang, C., Liu, C., Zhang, X. & Almpanidis, G. 2017a. An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications,* 82**,** 128-150.

Zhang, D., Maye, A., Gao, X., Hong, B., Engel, A. K. & Gao, S. 2010. An independent brain–computer interface using covert non-spatial visual selective attention. *Journal of neural engineering,* 7**,** 016010.

Zhang, P., Wang, X., Li, X. & Dai, P. EEG feature selection based on weighted-normalized mutual information for mental fatigue classification. 2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings, 2016. IEEE, 1-6.

Zhang, T. & Chen, W. 2016. LMD based features for the automatic seizure detection of EEG signals using SVM. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 25**,** 1100-1108.

Zhang, Y., Liu, B., Ji, X. & Huang, D. 2017b. Classification of EEG signals based on autoregressive model and wavelet packet decomposition. *Neural Processing Letters,* 45**,** 365-378.

Zhang, Y., Wang, C., Wu, F., Huang, K., Yang, L. & Ji, L. 2020. Prediction of working memory ability based on EEG by functional data analysis. *Journal of Neuroscience Methods,* 333**,** 108552.

Zhong, Y. & Jianhua, Z. Subject-generic EEG feature selection for emotion classification via transfer recursive feature elimination. 2017 36th Chinese Control Conference (CCC), 2017. IEEE, 11005-11010.

Zhu, X. 2013. An EEMD signal processing method realized via GPU&CPU. *International Journal of Applied Mathematics and Statistics,* 48**,** 532-540.

CHAPTER 3

FEATURE EXTRACTION FOR REAL-TIME BCI

This chapter aims at investigating computational performance of feature extraction methods widely used in real-time BCI using the same data set and exploring the potential of using parallel computing to speed up the process. The study is mainly divided into two sections. First, an evaluation of a range of existing feature extraction methods was conducted to compare real-time performance using an archival data set. Second, a feasibility study of implementing a parallel computing concept for computing feature extraction methods is presented in the chapter. Experimental results and findings from both studies are also discussed in this chapter.

3.1 EEG-Based BCI Processing Pipeline

At present, EEG has several clinical uses ranging from neurodiagnostic to monitoring normal wakefulness to complex clinical situations involving seizure or coma. In addition to the use of EEG in a clinical application, many studies have reported on its use in functional applications, such as a communication device (based on steady-state visual evoked potential) (Martinez et al., 2007) or brain-controlled navigation using motor imagery (MI) (Huang et al., 2012). Most current EEG-based BCI applications can be operated offline or online (real-time), depending on different users or needs. However, to implement a BCI application in real time, there are still many significant challenges that need to be overcome, including accuracy of prediction, the latency of computation and command, consistency, flexibility, and adaptability. These concerns relate to many factors, including design limitations associated with hardware specification, algorithm complexity, and requirements of advanced signal processing techniques. Remarkably, when processing BCI, improving the computation time can

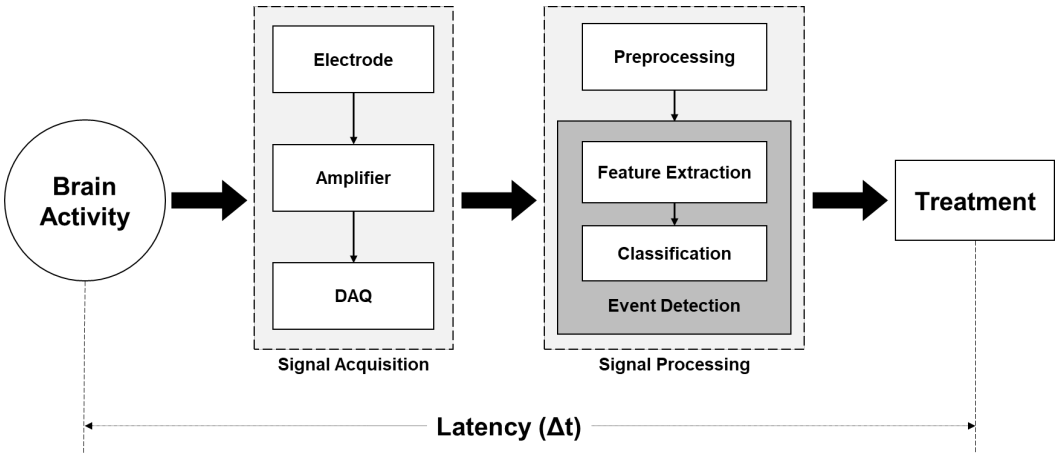

Figure 3.1. The standard BCI processing pipeline, which includes signal acquisition, signal pre-processing, feature extraction, classification, and application. Sequential processing in such way usually introduces an internal latency that can be a major issue when running in real time.

be a limiting feature as computing devices are generally not fast enough for processing in real time (Aluru and Jammula, 2013). With the rapid development of computing technology, there is a trend toward the use of an accelerating device for scientific computing applications requiring real-time solutions (Wang et al., 2015; Dohnálek et al., 2013). This technology can enhance BCI system performance and facilitating use of more sophisticated algorithms or multi-domain processing for high-speed BCI computation.

In a standard BCI processing pipeline, the most widely used approach comprises signal acquisition, signal pre-processing, feature extraction, classification, and output applications, as shown in Figure 3.1. However, this approach frequently encounters a slight delay referred to as internal latency, which leads to a significant issue when processing time is critical and when using a complex algorithm to detect brain activity that is time-sensitive from moment to moment control.

3.2 Feature Extraction Methods Used in Real-Time BCI

Famous feature extraction methods used in BCI applications can be found in Section 2.5.2.3. This section focuses on the existing methods that are widely used for real-time processing. According to the processing pipeline presented in Section 3.1, it can be seen that the feature extraction stage has a significant influence on the classification

result because it provides input to the classifier. For this reason, the range of feature extraction techniques in BCI has been explored with the goal of achieving the best classification accuracy but not necessarily an effective feature latency. In this study, potential feature extraction methods which are widely used in BCI applications, including template matching (Aarabi et al., 2009; Qu and Gotman, 1997), statistical moments (Soliman and Hsue, 1992; Alam and Bhuiyan, 2013), selective bandpower (Pfurtscheller et al., 2006; Palaniappan, 2005) and fast Fourier transform (FFT) power spectrum (Ko et al., 2009; Lehmann et al., 2007) have been compared to examine the classification success rate and the required computational performance when operating on different computing device types.

3.2.1 Template Matching

In digital image processing, template matching is a technique used for determining small areas of an image that match a template image. Also, template matching is used in time-domain signal processing, which measures the similarity between new data and initial data (a template) by computing a dot product of two signals. It can be presented in the form of the cross-correlation function as follows:

$$r_{xy}[\tau] = \frac{1}{N}\sum_{n=0}^{N-1} x[n] \cdot y[n-\tau]^{*} \tag{3.1}$$

where x[n] is an input signal and y[n] is a matching template.

To apply the template matching technique in EEG research, such as analysing cue-based motor imagery tasks, the template can be calculated by epoching the input EEG data around the cue point and applying the initial template to the epoch. The average of the cross-correlation function for each motor imagery task can be determined as an input feature for the classification stage. In this thesis, this similarity measurement is employed as a feature vector in a classifier.

3.2.2 Statistical Moments

Statistical moments are specific quantitative measurements in time-domain analysis, explaining the characteristics of the input signal's time courses. In signal processing,

the moments can also apply to many different aspects of image processing, ranging from invariant pattern recognition and image encoding to pose estimation. The frequently used statistical moments consist of mean, variance, skewness, and kurtosis. In addition, the general form or standardised form of the n-th order statistical moment of the random variable *X* can be described as

$$\frac{\mu_n}{\sigma^n} = \frac{\mathrm{E}[(X-\mu)^n]}{\sigma^n} \tag{3.2}$$

where μ_n and σ denotes the mean and variance of the signal, respectively.

In this study, the above equation forms the feature vector of the classification input, which comprises two entries extracted from only the first-order moment (mean) and second-order moment (variance). The classification accuracy may be improved by adding higher-order moments to the feature vector or using specific moments; however, this comes with more training samples and increased sensitivity to noise (Khan et al., 2019; da Silveira et al., 2017). Therefore, the selection of moments needs to be considered carefully.

3.2.3 Selective Bandpower

Selective bandpower represents the average power in a specific frequency range and is broadly used as a feature vector in many EEG-based binary classifications and multi-class problems. Since the non-deterministic characteristic of the time-varying EEG is unpredictably changed in different frequency bandwidths, the bandpower method is one of the most successful approaches to capturing changes. In this study, the average bandpower of time-domain EEG can be determined by convoluting an incoming EEG with a bandpass filter, squaring and averaging the samples over a time interval (e.g. the last second). The common form of the selective bandpower estimation can be represented by

$$BP_{selective} = \frac{1}{N}\sum_{n=0}^{N-1}(x[n] * h[n])^2 \tag{3.3}$$

where h[n] is an impulse response of the bandpass filter for convoluting with x[n].

3.2.4 FFT Power Spectrum

A fast Fourier transform (FFT) is one of the most commonly used tools for EEG spectral analysis, especially when dealing with a rapid change of signal components that are difficult to detect visually or mashed by noise or artefacts. Additionally, the FFT is a computationally efficient technique that can apply for examining signal characteristics in real time. The development of the FFT algorithm was originated from a discrete Fourier transform (DFT) theory, which are

$$X\left[e^{j\omega}\right] = \frac{1}{N}\sum_{n=0}^{N-1} x[n]e^{-j\omega n} \tag{3.4}$$

where x[n] is the input signal to be investigated. In this study, the power spectrum is obtained using the FFT method by taking the power of two to the output FFT coefficients. The feature vector can be then taken from the relative power spectrum of the signal, which can be described as

$$P_{relative} = \frac{\left|\int_{f_1}^{f_2} X\left[e^{j2\pi f}\right]df\right|^2}{\left|\int_{-\infty}^{\infty} X[e^{j2\pi f}]df\right|^2} \tag{3.5}$$

Note that the bandwidth frequency f_1 and f_2 are adjustable. To compare the outcome with the selective bandpower method, the numbers of f_1 and f_2 are set to match the previous feature extraction methods' parameters.

3.3 Evaluation of Potential Feature Extraction Methods for Real-Time BCI Processing

3.3.1 Introduction

Typically, a real-time BCI can be define as a system giving a feedback with a latency of less than a second (Xu et al., 2013; Li et al., 2011). Running signal processing algorithms those real-time environments is challenging and usually comes with high processing unit costs. Therefore, processing advanced algorithms in real time might not always be possible due to the complexity of the algorithms and the internal latency

in the system. Moreover, the affordable computational units' speed is not usually sufficient for applications requiring processing at a high-frequency rate. Regarding classification's training, it may be computationally expensive if the number of input samples is massively provided to a computing unit; however, this process is not required to complete in real time. Thus, training the classifier is usually performed offline and makes use of a pre-trained model for initialising the classification. Consequently, the most crucial part of real-time BCI is optimising the processing pipeline, particularly the feature extraction method, given that this represents the computational bottleneck leading to poor computational efficiency.

Due to the importance of optimising the processing pipeline, improving the internal latency or signal processing bottleneck can significantly benefit a real-time application to delivering a faster response, allowing increased data sampling frequency, and facilitating use of more sophisticated algorithms. This study evaluates the efficiency and performance of a range of time-domain and frequency-domain feature extraction approaches for their real-time application. A primary aim of this study is to provide a review of algorithm performance to aid in the identification of how to reduce pipeline bottlenecks. Furthermore, the study used an open-access data set (Leeb et al., 2008) to benchmark the performance of the selected feature extraction methods against other studies in terms of classification accuracy and computational efficiency. Note that the analysis was performed using MATLAB software.

3.3.2 Archival Data Set

3.3.2.1 Selection Criteria

At present, several EEG data sets are available for public access, which are commonly used as standard comparison studies when evaluating an algorithm's performance is needed. In this study, the COVID-19 pandemic precluded generation of new EEG data sets from the laboratory due to restrictions on face-to-face research studies. Accordingly, an archival EEG data set was used to investigate the performance of different feature extraction methods. To ensure that the quality and the specificity of the data used for evaluation are appropriate, the following selection criteria were used in choosing the data sets for this work.

Data Quality: The data sets must have a sufficient sampling rate (generally more than 250 Hz) and have a good signal-to-noise ratio with good compliance to a precise protocol.

Usability: The data sets should have adequate numbers of recruited participants, and the selected data should have achieved widespread public engagement with high impact and citation.

Familiarity: The data sets should relate to our laboratory's field of knowledge and practical expertise, primarily human neurophysiology and motor control.

Compatibility: The data format of the selected data set should be general and usable on multiple platforms. Also, the EEG channel montage used should be appropriate for motor imagery recordings.

By applying the above criteria, data set IV from Brain/Neural Computer Interaction: Horizon 2020 (Leeb et al., 2008) was identical as meeting the data specification outlined above. This data set is related to motor imagery tasks commonly used in our laboratory. In addition, the data set has been widely used in many BCI studies for benchmarking and evaluating of new classification/feature extraction algorithms.

3.3.2.2 Data Description

According to the experimental description of the selected data set, the study performed 2-class motor imagery (left and right) with a directional visual cue randomly presented on a screen. During the experiment with nine healthy participants, the EEG and EOG signals were collected. Regarding the experimental protocol, the subject had to perform five repeat sessions, each with 120 trials. The EEG and EOG signals were acquired with a sampling frequency of 250 Hz and ±250 microvolts of a dynamic range. To reduce the effect of interference, analogue bandpass filtering (0.5-100 Hz) and a notch filter at 50 Hz were applied. EEG electrodes were placed at C3, C4, and Cz according to the international 10-20 system. The original work followed a general signal processing pipeline to process the collected signals, including signal pre-processing, feature extraction and selection, binary classification, and error calculation. The experimental protocol for the motor imagery task of this data set is displayed in Figure 3.2. In this thesis, the proposed feature extraction and classification

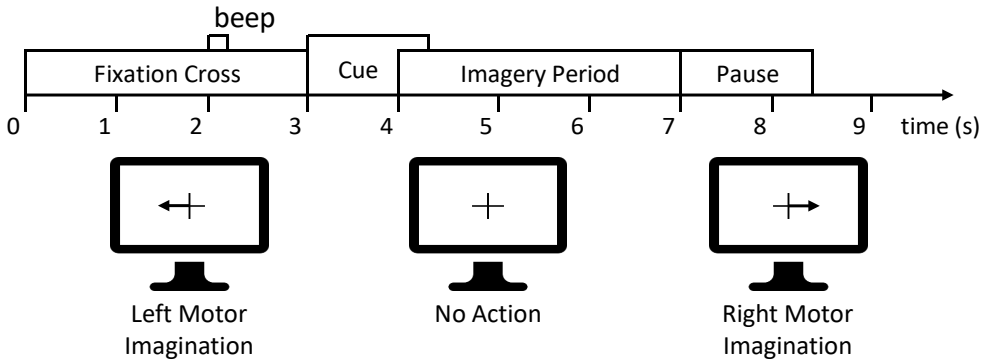

Figure 3.2. Diagram of cue-based motor imagery task from the selected data set (Leeb et al., 2008). Each single-trial lasted 7 seconds which the cue appeared at 3 seconds after starting. The figure is adapted from the original publication.

were applied to the data set to investigate the performance in terms of accuracy, sensitivity, and computation time.

3.3.2.3 Time-Frequency Data Visualisation

To analyse the frequency response of the EEG recorded from the motor imagery task, the EEG data is primarily visualised in both time and frequency domains using the continuous wavelet transform (CWT) based on the Morlet wavelet (Adeli et al., 2003). According to the EEG behaviour during a motor imagery task, a change in frequency magnitude, especially in mu rhythm, can be recognised in the time-frequency domain. Figure 3.3 demonstrates the time-frequency response of the EEG from C3 and C4 during the motor imagery task of the right hand and left hand, respectively. At the first stage, the visualisation was performed for all records to ensure the usability of the data set; if the plot showed negative signs of usability, such as containing too much noise or artefacts, it was excluded from the analysis. Regarding the results of this pre-screening session, nine data sets were preliminarily investigated, and there was no data set excluded from the study. In Figure 3.3, the plot shows enhanced activity in the 8-15 Hz range that occurs prior to and subsequent to the time of the movement cue.

3.3.3 Artefact Reduction

According to the other signal processing approach performed on this data set, R. Leeb and et al. (Leeb et al., 2007) used regression analysis for artefact removal. In order to apply the regression analysis, the input signal must have a linear relationship, which

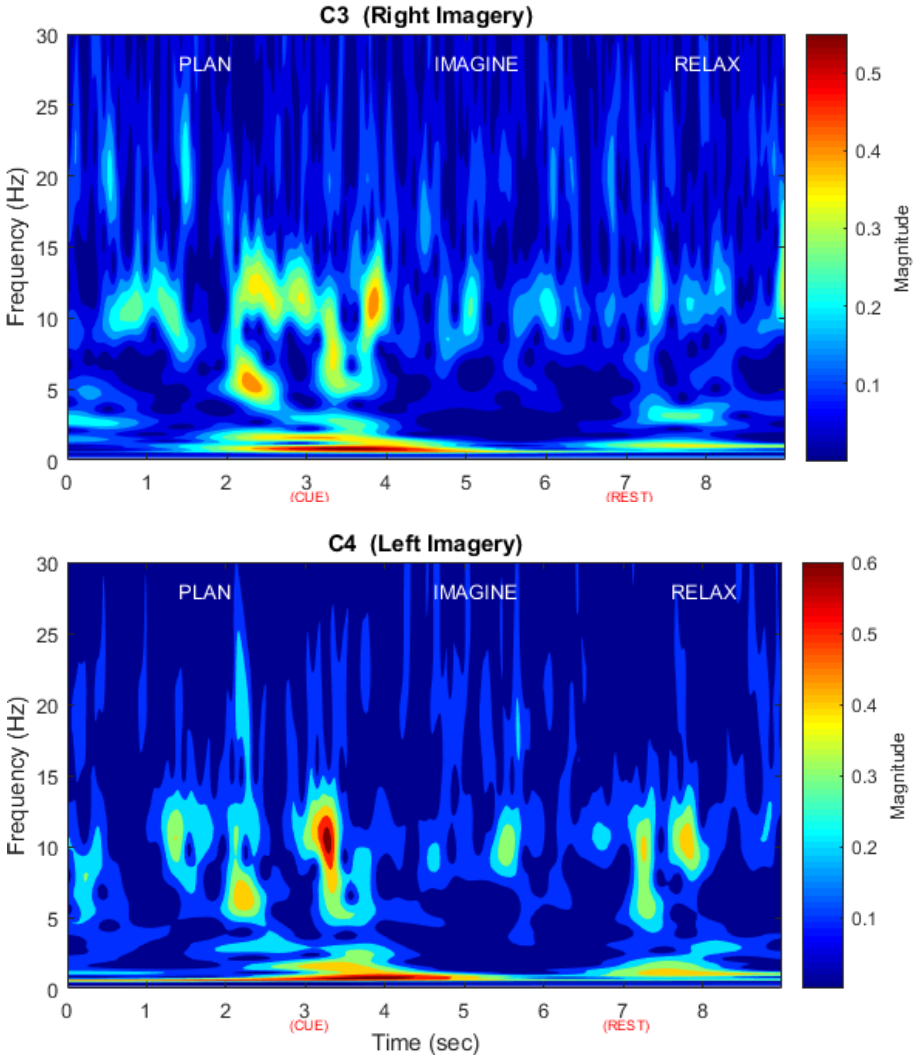

Figure 3.3. Data visualisation in time-frequency analysis using the CWT method applied to recordings from C3 and C4 during a motor imagery task, taken from Subject 1 (Leeb et al., 2008). The plot was generated from the average EEG of 120 trials using "cwt" function in MATLAB.

in this case, there are two components, i.e., the real EEG and the spatial EOG, showing their dependency. Here, independent component analysis (ICA) was applied to the 3 EEG channels and an EOG channel to decrease contamination by unwanted signals; this technique yield results better than the regression method (Jung et al., 2000). Theoretically, the EOG recorded at the frontal site can be used in the ICA step to remove its artefacts from the recorded EEG channels. Figure 3.4 shows the cleaned EEG signal after performing the ICA decomposition and shows the removal of the EOG artefact. Here, the ICA technique was applied to all data sets prior to post-processing feature extraction.

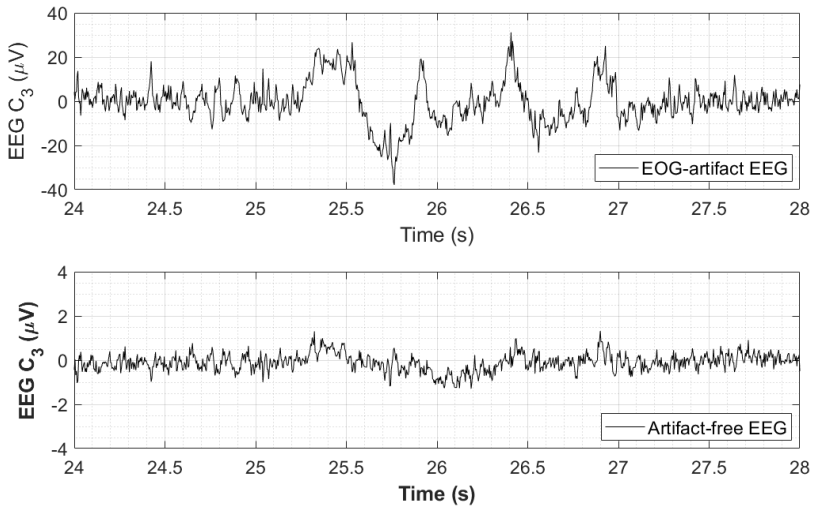

Figure 3.4. Presenting an example of an artifact-free EEG captured from C3 position using the ICA decomposition technique, taken from Subject 1.

3.3.4 Selected Feature Extraction Methods

As this study aims at examining the performance of the feature extraction methods mentioned in Section 3.2, four feature extraction methods were used to formulate a feature vector, including template matching, statistical moments, selective bandpower, and fast Fourier transform (FFT) power spectrum. Note that the original study (Leeb et al., 2008) utilised the selective bandpower feature vector as an input of their classification. In this study, the feature vector of each feature extraction method was initialised as follows:

Template Matching: The template was calculated from an average of EEG epochs, captured for 4 seconds from the cue onset. Then the cross-correlation value of each trial was obtained as a feature vector by using this template.

Statistical Moments: Two feature vectors, including the first and the second moments, were determined based on a 4-second window from the cue onset.

Selective Bandpower: The third-order Butterworth filter used for filtering was between 7 Hz to 22 Hz, which is identical to the previous study (Leeb et al., 2008). The bandpower was also calculated based on a 4-second window from the cue onset.

FFT Power Spectrum: The feature vector was initialised from the relative power spectrum of 4-second EEG epochs, in which the frequency bandwidth was considered within a range between 7 Hz to 22 Hz.

3.3.5 Classification Methods

To classify a simple BCI application such as a 2-class motor imagery problem, R. Leeb and et al. used linear discriminant analysis (LDA) for dimensionality reduction before their binomial classification, obtaining a 25% average classification error by offline analysis (Leeb et al., 2007). As an alternate approach and to utilise good generalisation properties and insensitivity to overtraining (Subasi and Gursoy, 2010), this thesis used a support vector machine (SVM) (Hearst et al., 1998) via a "fitcsvm" command in MATLAB to categorise the motor imagery task within this EEG data set. According to SVM, several kernel functions, including linear, quadratic, polynomial, and Gaussian, were tried and the best classification results were obtained from Gaussian kernel. The proposed feature extraction methods in Section 3.2 were initiated as a feature vector. In this study, the post-processed EEG data from C3 and C4 was used to create a two-dimensional feature vector; consequently, the single vector was the elements of the training set's obtained outcome which were concatenated from all trials. The obtained feature vector was classified using SVM by windowing for 4 seconds after the cue occurrence. The window length of each epoch is 1,000 milliseconds and the sliding window is moved a sample by a sample. At each sample point, the classification errors were calculated. In addition, a 10x10 cross-validation technique was applied to the classification, and the average maximum accuracy was described and its time point for comparison with other related works.

3.3.6 Experimental Results and Findings

This study examined the classification accuracy and the computational performance of four different feature extraction methods based on the above-mentioned data set (Leeb et al., 2008). By using SVM, the kernel was selected based on a trial and error basis. To verify the consistency of SVM classification, the 10x10 cross-validation technique was applied; the average accuracy across all subjects was obtained. The classification error and computation time of the proposed feature extraction methods, including template matching (Template), statistical moments (S-Moment), selective bandpower (Sub-BP), and FFT power spectrum (FFT Power), were evaluated based on the 9-subject data sets reported in Table 3.1 and Figure 3.5, respectively.

Table 3.1. Classification error of four different feature extraction methods.

Subject	Classification Error[a] (%)			
	Template	S-Moment	Sub-BP	FFT Power
S1	27.2	36.8	31.2	32.4
S2	32.4	34.8	39.9	35.4
S3	35.2	29.3	39.2	43.3
S4	39.5	36.3	6.7	9.8
S5	20.6	28.1	37.1	21.7
S6	37.4	30.0	24.9	26.2
S7	23.5	26.2	25.5	21.9
S8	38.0	36.0	22.0	22.4
S9	39.9	40.6	26.2	31.0
Mean	**32.6±7.2**	**33.1±4.8**	**28.1±10.4**	**27.1±9.7**

[a]There is no significant difference between groups (p-value>0.05).

Regarding the classification results, there was no significant difference observed among different features extraction methods (p-value>0.05). Furthermore, the one-way analysis of variance (ANOVA) revealed no best feature extraction method based on the classification prediction rate. In general, applying any feature extraction method for the classification of individual subject data requires fine-tuning intrinsic parameters to obtain the best results. This can be seen in Table 3.1, where each subject's data is appropriate to a different feature. However, for consistency, this study maintained the same parameters for all the subjects. As a result, the average classification error of all feature extraction methods across subjects in this study obtains 30.24±8.05% compared to 25.00±11.70% as reported in the original work (Leeb et al., 2007).

Along with the classification error, the decision period (the maximum classification accuracy after the cue occurrence) during subjects' imagination are reported in Table 3.2. The average event duration obtained from all feature extraction methods is approximately 2.52±0.51 seconds after the presence of the cue. This number closely matches the result reported in the original research (Leeb et al., 2007) of 2.88±0.51 seconds. Additionally, the results show no significant difference between the proposed feature extraction methods (with ANOVA on the obtained decision duration).

In addition to the classification performance, the computation time was also considered. The analysis and implementation in this study were deployed using a MATLAB program running on a 64-bit Windows 10 machine (16 GB of DDR3

Table 3.2. Decision duration of four different feature extraction methods.

Subject	Decision Duration[a] (sec)			
	Template	S-Moment	Sub-BP	FFT Power
S1	2.38	1.48	2.58	2.40
S2	1.73	1.74	2.63	2.33
S3	2.50	2.97	2.46	2.62
S4	1.92	3.26	3.12	2.73
S5	2.62	2.82	3.82	2.37
S6	2.13	1.72	2.70	2.87
S7	2.59	2.83	2.47	2.36
S8	2.08	1.50	2.98	2.70
S9	2.52	3.10	2.75	3.13
Mean	**2.28±0.32**	**2.38±0.75**	**2.83±0.43**	**2.61±0.28**

[a]There is no significant difference between groups (p-value>0.05).

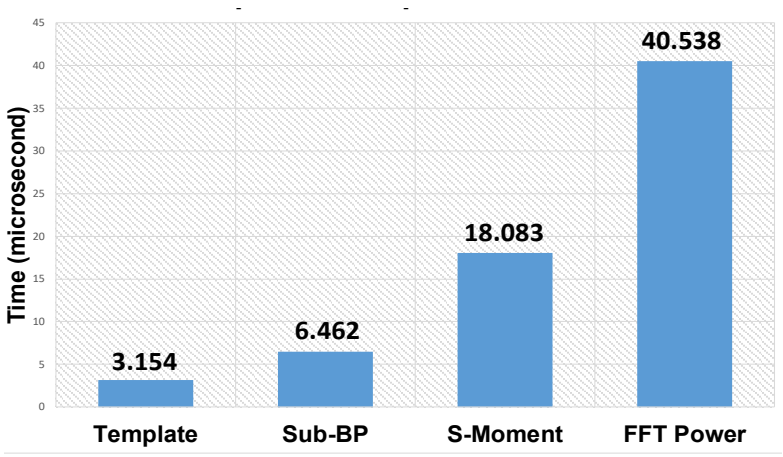

Figure 3.5. Results of compared computation time (in microsecond) of each feature operating on MATLAB programming using single-core processing.

memory and 2.6 GHz of Intel Core i7-6700 processors). Thus, the baseline computation was set to execute on a single-core processing unit without using any accelerating devices and system's optimisation.

According to the preliminary results attained from this study, the significant findings reveal that using a different feature extraction method in data set IV (Leeb et al., 2008) does not significantly differ in the classification accuracy but impacts on the computation time, as illustrated in Figure 3.5. For instance, using the FFT method consumes more computation time than other methods without accuracy gain. Furthermore, compared to the same category of the feature extraction technique used in this data sets, the classification error of the selective bandpower yielded approximately 28%, which is not significantly different from the result reported in the

original research (Leeb et al., 2007), whereas the average classification error is about 25%. Similarly, the average decision duration of the proposed selective bandpower is 2.83 seconds, while the original obtained 2.52 seconds; consequently, this emphasises that the results are not significantly different.

3.4 Enhanced Real-Time Feature Extraction Based on Parallel Computing Scheme

3.4.1 Introduction

Online or real-time BCI applications have become a key theme for BCI research. One major challenge in implementing such BCI applications is reducing the internal latency resulting from using complex signal processing algorithms. This issue becomes enhanced if processing at a high-frequency rate is also required. Typically, a trade-off between the input sample and the precision of the processing outcome needs to be considered carefully. Also, running feature extraction algorithms in real time is one of the most challenging BCI topics. The number of input channels used may be significantly greater than one or two. In addition, some complex features such as independent component analysis (ICA), autoregressive (AR) model, and discrete wavelet transform (DWT) are frequently applied to an offline BCI (Lotte et al., 2018) and are valuable in improving signal to noise ratio. However, these time-consuming methods, when used in real-time processing, required optimisation, which may itself limit the performance of the algorithms in some cases (Al-Fahoum and Al-Fraihat, 2014). By implementing a heterogeneous computing concept, these complex algorithms can be split into smaller parts. Each can be computed concurrently; accordingly, a complete computation can be gathered by a summation.

Here, a parallel computing architecture is implemented based on the feature extraction methods mentioned in the previous section to alleviate the computational complexity associated with real-time BCI processing. The system was developed based on an open-source parallel processing platform named OpenCL. Multi-domain feature extraction methods, including template matching, selective bandpower, statistical moments, and FFT bandpower, were selected to evaluate the computational

Table 3.3. Delay and versatility in signal processing.

Category	Delay	Versatility
Signal acquisition	Low	Low
Data transmission	Medium	Low
Types of applicaion	Medium	Low
Processing algorithms	High	High

performance when applying into a parallel processing pipeline using the archival EEG data set (Delorme et al., 2004).

3.4.2 Reducing Latency in Signal Processing

According to Figure 3.1, the signal processing illustrated usually generates a computational latency, leading to a significant problem when the runtime is a primary consideration. To analyse the cause of the computational latency in real-time processing, this study considers the most related aspects, which can be divided into four categories: signal acquisition hardware, data transmission, types of application, and processing algorithms. The analysis of the latency causes that is considered in terms of delay and versatility is shown in Table 3.3. The investigation is based on cost-effectiveness and current technology.

According to Table 3.3, most of the latency caused in a real-time BCI is the computational complexity that emerged from signal processing algorithms. In order to troubleshoot this issue, previous studies used a different approach, such as parallel processing instead of traditionally sequential method, including parallel-based video encoding (He et al., 2014), image reconstruction, and speech processing (Strope et al., 2013). The results showed that parallel computation yields significantly less runtime than using the sequential method. This achievement confirms the potential of applying a parallel distributed computing scheme in online BCI applications.

3.4.3 Accelerating the signal processing pipeline

As mentioned in the previous section, speeding up the processing runtime can be achieved through many approaches, depending on the latency category. The following techniques detail the existing solutions with and without the modification of an algorithm.

1. Signal Acquisition: replacing the acquisition hardware with better-quality specifications, reducing the sampling frequency, and reducing the number of input channels.
2. Data Transmission: reducing the number of samples, increasing the buffer size, encoding data before sending and decoding after receiving, and using shared memory between communicated devices.
3. Processing Algorithm: using a recursive/reduced form, using parallel computation with algorithm modification, increasing the number of computing devices or accelerating devices.

Based on the methods mentioned above for accelerating the processing pipeline, the use of a parallel computing scheme has been increasingly implemented in many scientific works. This is because the research and development in this area have grown exponentially, resulting in a wide variety of tools and state-of-the-art technologies in the market. Therefore, this brings us to the interest of applying parallel computation to real-time signal processing for BCI applications.

3.4.4 EEG Data Set

To evaluate the performance of the proposed parallel processing technique, open-access EEG data was used. Following the selection criteria proposed in the previous section, another motor-imagery-related EEG data set was selected. This public data set comprises data consisting of 32-channel EEG recorded from fourteen subjects (seven males, seven females), provided by Swartz Center for Computational Neuroscience (Delorme et al., 2004). Regarding the experimental protocol related to the data set, the subjects performed the following tasks: a go/no-go categorisation and a go/no-go recognition task by displaying natural photographs. Target and non-target images were randomly presented in both tasks. In each trial, subjects had to press a touch-sensitive button. Then, an image was flashed on the screen for 20 milliseconds. Participants gave their responses following a go/nogo paradigm. Each target required subjects to swiftly and precisely lift their finger off the button. Participants had 1000 milliseconds to respond, beyond which any response was deemed a no-go. Each participant was required to perform 2500 trials in total. Note that the EEG data was initially acquired

with a sampling frequency of 1 kHz; however, in this study, the archival data were regenerated and reamplified in order to match the specification of the acquisition device, including an input/output impedance and a voltage limit. The proposed pseudo-real-time simulation system also offers the ability to run other offline data set for analysing the computational performance of signal processing algorithms.

3.4.5 Simulation System

To simulate this study, the pseudo-real-time simulation system was developed using Qt (Qt 5.12 LTS), which is generally the main development platform for making a graphic user interface (GUI) and a back-end system. The proposed system used a C++ programming language for the deployment. One advantage of using the Qt platform is that it can easily integrate the OpenCL and related libraries. The proposed simulation system's hardware primarily includes a signal generator module and a signal acquisition module in this study. The signal generator is for continuously outputting the analogue signal of the archival data, which is then captured in the signal acquisition.

Regarding the reuse of the data set previously mentioned, the EEG data was regenerated through the 32-channel analogue output module (NI PCIe-6738) and then transmitted to the 32-channel analogue input module (NI PCIe-6343), which is connected using RG58 50-Ohm coaxial cables. Initially, the NI PCIe-6738 generated the output signal of each channel at the sampling frequency of 1 kHz, which is identical to the data set's characteristic. Figure 3.6 illustrates the overview of the proposed simulation system. The photo showing the system setup is also provided in Appendix A. Note that the output resolution of the NI PCIe-6738 is 16 bits with a voltage range

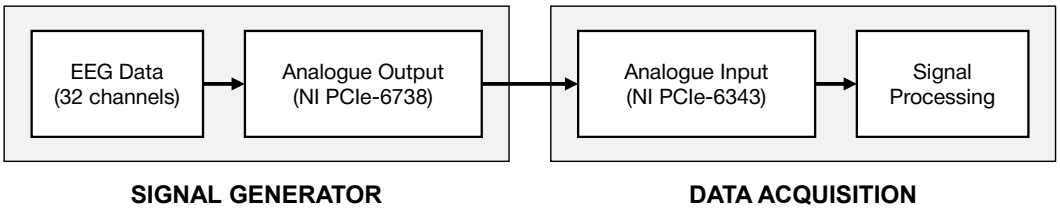

Figure 3.6. Demonstrating the simulation system used in this study. The arvhied EEG data set was used to regenerate the physical EEG signal via an anlogue outpu module. The analogue input module was used to acquire the generated EEG signal to be processed in the proposed pipeline.

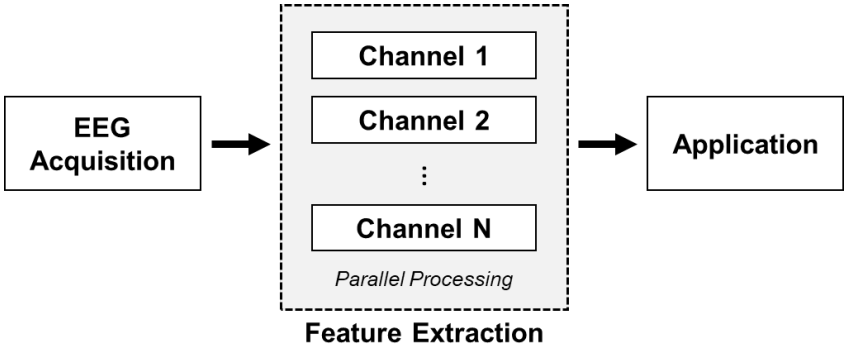

Figure 3.7. Overview of parallel feature extraction in real-time BCI system. Each EEG channel can be processed simultenouely using multiple compute units.

of ±10V. This study varies the sampling frequency for signal acquisition, including 128, 256 512, 1024, and 2048 Hz, to validate the effect of signal quality and processing performance.

3.4.6 Parallel Signal Processing

Processing digital signals, in general, is performed in a sequence by completing one processing block at a time. However, this approach frequently causes a small to considerable delay depending on loads of processing blocks required. Applying a parallel distributed computing concept to those processing sequences is a practical solution to minimise this internal latency. Moreover, the saved time gains arising from this optimisation can be used to enhance the system with additional processing algorithms. This study implements parallel processing architecture and performs feature extraction on each channel separately and concurrently by using multiple compute units. The main processing unit is responsible for synchronising the result when each compute unit completed its tasks. The proposed overall system for parallel feature extraction is demonstrated in Figure 3.7 and shows simultaneous processing of each channel.

3.4.7 OpenCL Initialisation

With the advantages of using an open-source parallel platform named OpenCL, the opportunity arises to take advantage of high-performance computing hardware, such as GPU and multi-core CPU that is not limited to a specific platform. The GPU is one

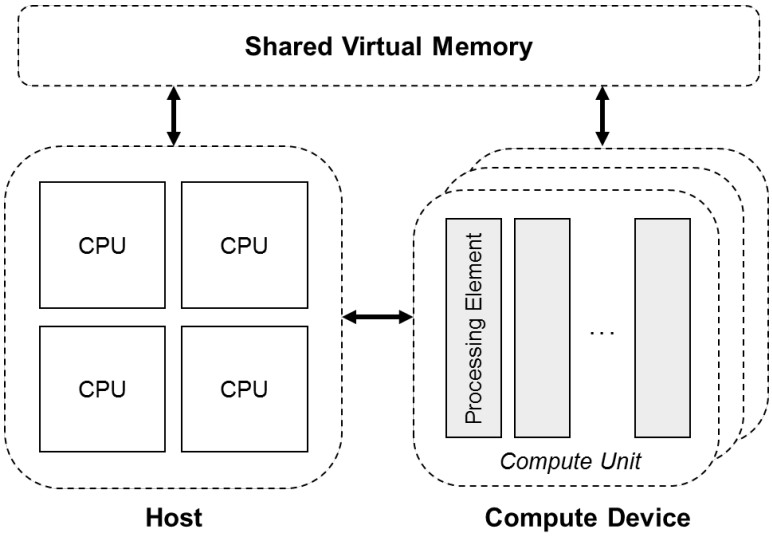

Figure 3.8. The overview of the structure of the OpenCL 2.0 platform: introducing a shared virtual memory to reduce the internal latency during data transmission. The memory is initialised by the host unit and will be immediately recognised by the compute device.

device that is seen as increasingly advantageous from the OpenCL platform as currently as several vendors compete in the market, providing high performance at low cost. In this study, the AMD graphic card (Radeon™ Pro WX 7100) was selected to deploy the parallel computation based on the OpenCL 2.0, allowing access to internally shared virtual memory. Using the shared virtual memory can dramatically decrease the internal latency caused by transferring data between a host and accelerating devices. Moreover, in order to achieve the best computing performance, initialising the number of work-items and workgroups is mandatory, which can be done manually; for example, by appropriately splitting the number of the input size into global and local memory. Figure 3.8 illustrates the architectural overview of the proposed OpenCL 2.0 platform used in the study.

3.4.8 Implemented Feature Extraction Techniques

For signal processing based on the frequency domain, several feature extraction methods have been increasingly developed, including discrete Fourier transform (DFT) or power spectral density (PDS) and wavelet transform (WT). These well-known methods have been considered one of the most efficient processing techniques, especially when dealing with time-varying bio-signals such as EEG and ECG.

Regarding signal processing in the time domain, the most widely used method for extracting features from EEG signals is using a selective bandpower, which calculates the average power of a signal of interest in a specific frequency range over time.

In this study, the bandpower method was applied to the feature extraction step for event detection. In addition to the bandpower, statistical moments (another time-domain method) are also applied for comparison. The statistical moments are specific quantitative measurements in the time-domain analysis. The general formula of the n-th order statistical moments described in Equation 3.2, including mean, variance, skewness, and kurtosis, are primarily used for feature extraction. However, here only the first moment (mean) and the second moment (variance) are used to demonstrate the potential of parallelising the statistical moments' algorithm. Each moment is calculated concurrently on each channel of the input signal. Furthermore, another two potential feature extraction methods, i.e. template matching and fast Fourier transform (FFT) power spectrum, are included in this study. The technique used to reduce these proposed feature extraction algorithms' computational complexity is presented in the next section.

3.4.9 Parallel Sum Reduction

Some of the proposed feature extraction algorithms are based on direct summation, such as the bandpower and the statistical moments. As a result, computational performance can be improved as some numerical calculations are redundant. However, with the use of parallel computing architecture, this issue can be accommodated by using a parallel sum reduction technique (Zaki et al., 1997), which offers improved flexibility and gives the opportunity to modify the existing algorithms to achieve the highest performance with the parallel distributed computing scheme. Here, all feature extraction methods mentioned in Section 3.2, including template matching, statistical moments, selective bandpower, and FFT bandpower, are modified using parallel sum reduction.

According to the parallel sun reduction algorithm, this technique maximises the performance of the computing unit by passing values from global memory to local memory of the same workgroup, resulting in a faster processing time. Each workgroup

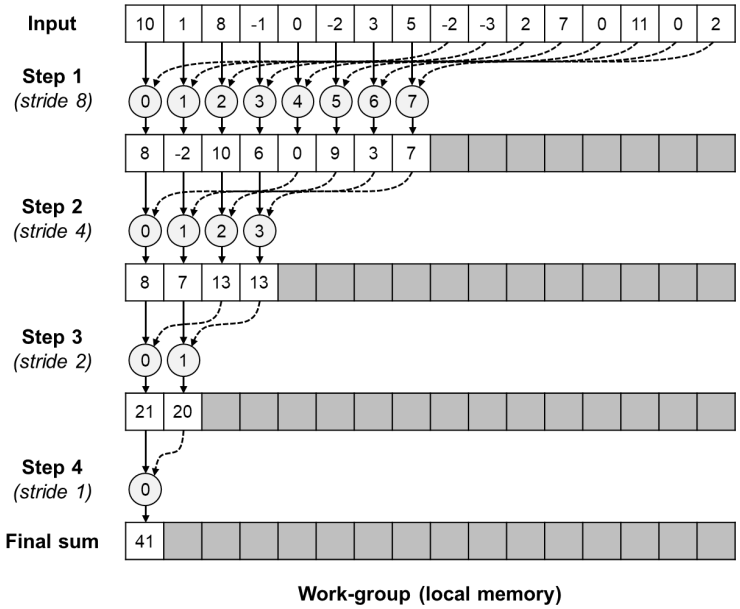

Figure 3.9. Representing a basic concept of a parallel sum reduction technique: each step reduces the number of numeric computation to half, resulting in a double speed up.

processes its local tasks called work-items concurrently, partitioning the whole summation into a small summation. As a consequence, the workgroups complete the final calculation when all work items finish their computation. The concept of the parallel sum reduction algorithm is displayed in Figure 3.9, which illustrates the use of local memory to store each element concurrently and then reduces the calculation to half by using a stride. Note that in OpenCL 2.0, the parallel sum reduction is integrated into a workgroup function; therefore, there is no additional requirement to generate a nested loop for calculating the summation.

3.4.10 Improvement of Performance

In this study, fine-tuning intrinsic parameters of parallel computation and optimising individual accelerating hardware are required to improve the runtime speed and the efficient use of computational resources. According to the nature of the parallel sum reduction algorithm, each step splits the whole summation into half. Accordingly, the most appropriate input length should be a power of two. Hence, this study set the local work-items to 16, 32, 64, 128, and 256, respectively, to support this computational behaviour. Of course, the maximum size of the local work items always depends on the specification of the hardware used. Most of the GPU cards in the market come with

parallel compute unit or core; for example, the compute unit is called “stream processors” for AMD product, while with NVIDIA devices the names “CUDA cores” is used. In this work, the AMD GPU card offers up to 256 units. In addition, in the OpenCL 2.0 framework, when using a sum reduction inside a workgroup function, it is not necessary to follow the power of the two rules as the built-in function can automatically handle the computation.

3.4.11 Benchmarking

To examine the performance of the proposed parallel computing scheme, this study compared performance in terms of execution time by using different settings of intrinsic parameters (number of local work-items). The average computation time of executing a complete command as each local work item deployed on a GPU was calculated from 100 executions. Moreover, parallel computing’s computational performance was also compared to a CPU’s performance using sequential computing. The result of each feature extraction method is reported in the next section, including the speedup ratio (t_{CPU}/t_{GPU}). Note that the complete execution is counted from the beginning of transferring input data from host to device, processing the kernel, and transferring computed results back to the host. This study was performed on a Windows 10 (64-bit) workstation with 32-GB DDR4 and Intel Xeon E5-1630 v4. The experimental protocol of the performance evaluation is illustrated in Figure 3.10.

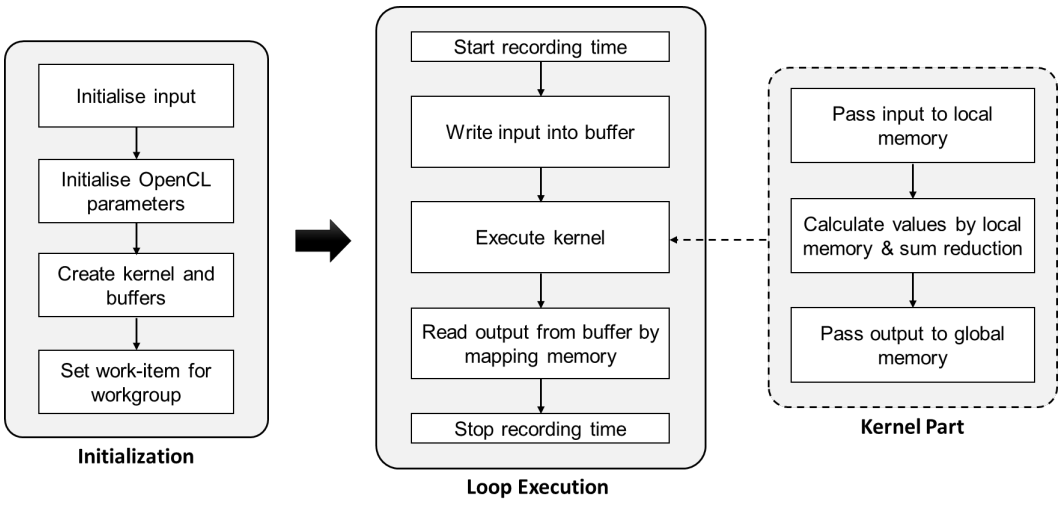

Figure 3.10. Programming flowchart of the experimental protocol.

3.4.12 Experimental Results and Findings

Potential feature extraction methods, including template matching, selective bandpower, statistical moments, and FFT bandpower, were examined in a real-time signal processing environment using a parallel distributed computing architecture. The computation time and comparison between using sequential processing and parallel processing approach are reported in this section. Figures 3.11 to 3.14 illustrate the execution time in milliseconds for each protocol (template matching, statistical moments, selective bandpower, and FFT bandpower, respectively).

The results show that the computation time is related to the number of input samples (see Figures 3.11a to 3.14a); the larger the input sample, the more computation time is necessary. Furthermore, modifying the initialisation of workgroup sizes has a large effect on the computational performance: the larger the workgroup size, the more performance improved. In addition to the computation time, the comparison between running sequential processing and parallel processing pipeline on the same computing machine is illustrated next to its computation time (Figures 3.11b to 3.14b). Finally, the speedup ratio was calculated from the different computation times of the sequential approach divided by the parallel approach (t_{CPU}/t_{GPU}).

According to the benchmarking result, increasing the input size yields a higher ratio (better performance) for all feature extraction methods (see Figures 3.11b to 3.14b). Specifically, using selective bandpower and FFT bandpower methods show a significantly increased ratio in Figure 3.13b and Figure 3.14b, respectively. This is a consequence of the reduced summations within the parallel computing scheme. Conversely, in template matching and statistical moment tests, the speedup ratios show the opposite behaviour, and the CPU performed the computation faster than the GPU. Remarkably, adjusting workgroup size can impact the speedup ratio, as shown in the results of all feature extraction methods.

According to the results of computation time, the latencies obtained from the proposed feature extraction methods are relatively small and can provide potential for use in real time applications. For example, at a sampling frequency of 1024, the template matching method takes approximately 0.35 milliseconds, the statistical

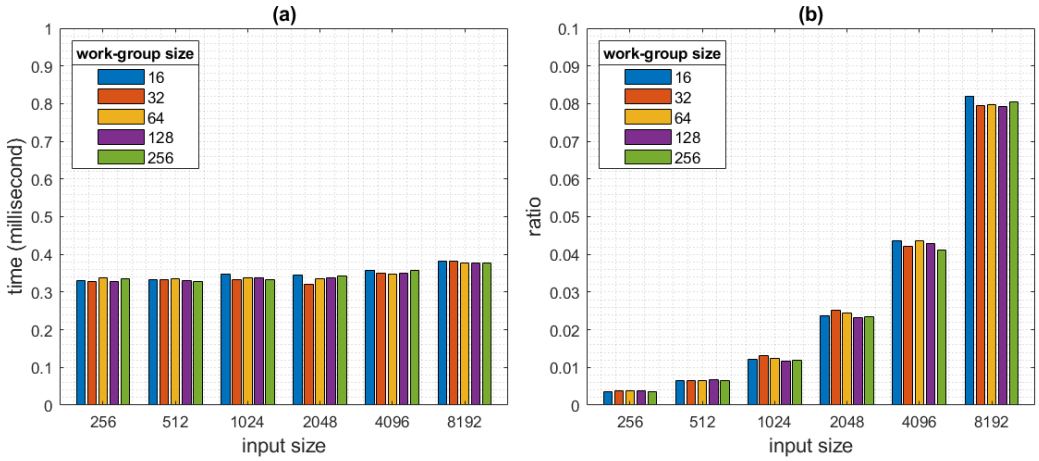

Figure 3.11. Experimental results of template matching method using different work-group sizes: (a) execution time (b) speedup ratio.

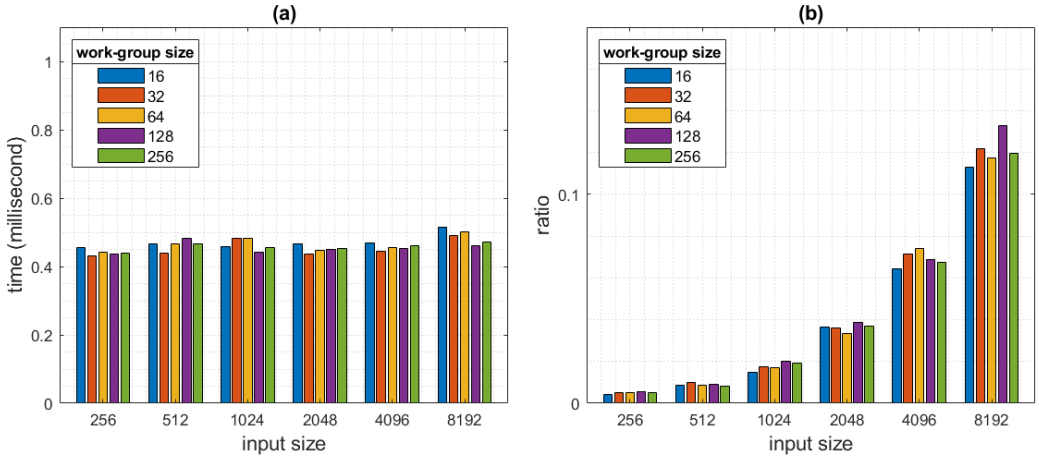

Figure 3.12. Experimental results of statistical moments method using different work-group sizes: (a) execution time (b) speedup ratio.

moments take roughly 0.45 milliseconds, and it requires 0.3 milliseconds in the selective bandpower feature. However, running the system using the FFT bandpower takes a longer computation time than others, approximately 10 milliseconds. In addition to the runtime difference, running data at nearly 2 kHz using the selective bandpower and FFT bandpower provides a runtime of approximately 0.4 milliseconds and 20 milliseconds, respectively. Interestingly, these numbers are significantly higher than those running at 1 kHz. While using template matching and statistical moments, the runtime results between running at 1 kHz and 2 kHz are almost similar. Note that simulating on a different processor, even having the same accelerating devices, usually provides different computation times.

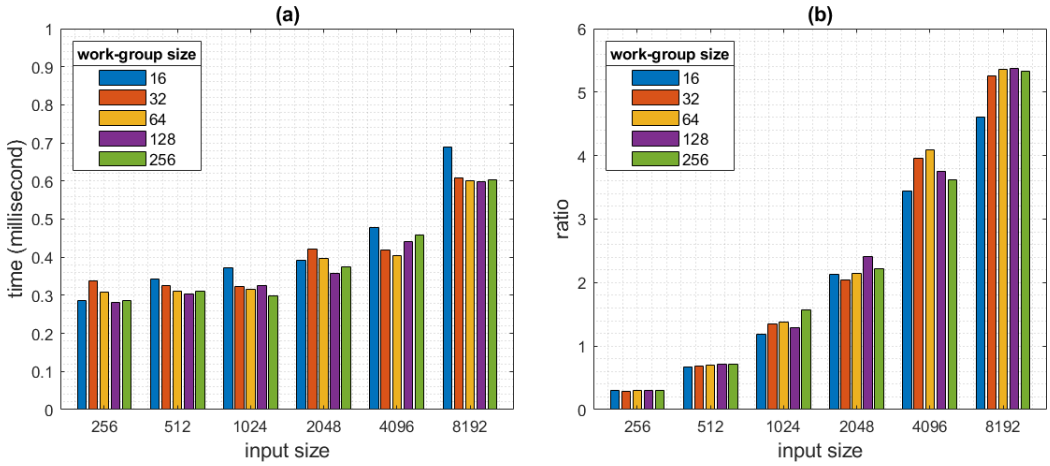

Figure 3.13. Experimental results of selective bandpower method using different work-group sizes: (a) execution time (b) speedup ratio.

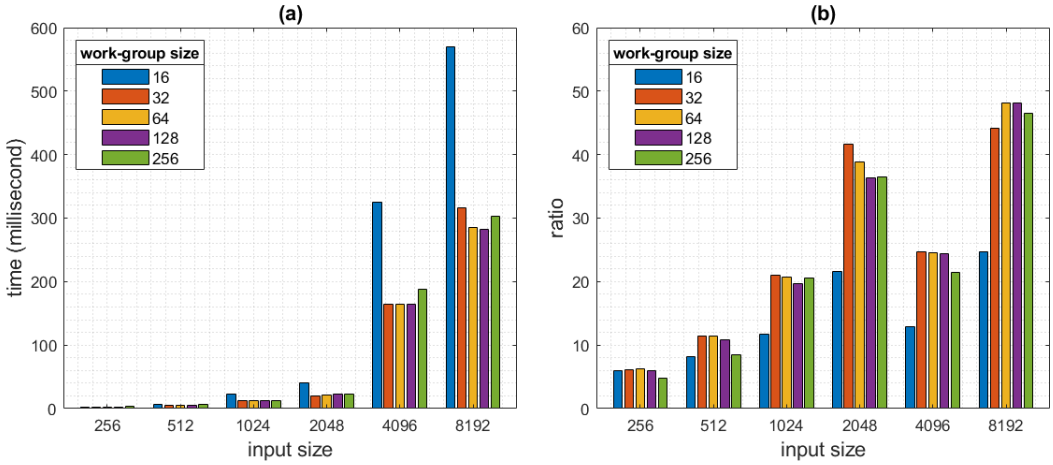

Figure 3.14. Experimental results of FFT bandpower method using different work-group sizes: (a) execution time (b) speedup ratio.

As illustrated in the benchmarking results, the parallel sum reduction technique can improve the speedup ratio when the input size grows. Additionally, increasing the number of workgroup sizes from 16 to 64 provides a higher speedup ratio for all features. On the other hand, while initialising a workgroup size to 128, the speedup ratio slightly decreases. This unpredictable behaviour occurs in many accelerating devices because it requires additional time for the compute unit to initialise intrinsic parameters (hardware-level) before processing incoming data continuously and efficiently, as discussed in the previous study (Fang et al., 2011). As a consequence, a trade-off between an input sample and acceptable internal latency should be

considered carefully. For the GPU used in the study, each processing dimension has a maximum workgroup size of 256 work items.

In addition to the advantage of the parallel sum reduction, it can be applied to other time-consuming signal processing algorithms, such as well-known frequency-domain analysis DFT and WT. These algorithms require multiple computing steps that can be separated into several kernels and directly execute on a device-side without sending any requests from the host side, as introduced in the new features of OpenCL 2.0. Moreover, this technique can be applied to higher statistical moments, i.e., skewness and kurtosis. Each kernel is responsible for calculating individual moments by running concurrently using shared virtual memory. Apart from applying to GPU devices, the project can be executed on other accelerating devices such as FPGAs and DSPs using the same OpenCL program. This implementation is slightly expeditious and uses a small effort to complete, which only minor parameter adjustment is required.

Concerning the latency analysis, the major limitation caused by the algorithm complexity can be addressed by implementing a parallel computing scheme to a processing pipeline that a computational load is required, e.g., massive summation, matrix multiplication. However, a trade-off between time- and cost-effectiveness or the system's accuracy must be considered. Furthermore, an external OpenCL library, mainly designed for signal processing, could help researchers benefit from the parallel distributed computing architecture. However, developing all tools and libraries from scratch is a very time-consuming task without a commercial or strong user group community. Furthermore, programming an accelerating device requires intensive background and knowledge of hardware-level architecture. Therefore, this study has developed all the steps from the beginning, including setting complicated intrinsic parameters, initialising a platform and a context, and testing the simulation system.

This study investigated the possibility of executing feature extraction methods on GPU devices and parallelising them using various optimisation techniques in this study. However, due to the system's existing limitations, such as computational efficiency and memory use, the system is still unsatisfactorily robust and dependable when input samples/channels numbers increase. Additionally, due to the OpenCL

protocol's limitations when used in conjunction with the parallel sum reduction technique, the algorithm must adhere to the available mathematical functions: addition, multiplication, and atomic operations. This complicates the task of incorporating it into modern signal processing methods. As a result, the study addressed these challenges by developing a novel processing method that can be applied to both sequential and parallel computing architectures.

In conclusion, this section presents the advantage of implementing a parallel computing technique into signal processing for real-time BCI applications. Accordingly, speeding up the computation time using a parallel processing scheme is theoretically possible and adaptable for processing in real time. However, in order to reduce the system latency further, fully optimising both hardware and software is required for use in an actual real-time environment. Furthermore, a signal processing OpenCL library could minimise the time to develop parallel-architecture-based BCI prototypes.

3.5 Chapter Summary

This chapter focused on the study of potential feature extraction methods that are of interest in realising real-time BCI applications. First, the most widely used methods were investigated using archival data sets in terms of classification accuracy and internal latency. In addition, the parallel distributed computing technique was applied to the selected feature extraction methods, demonstrating the speedup improvement. From the results, the conclusion is reached that the parallel scheme's potential is significant with feasibility to improve the existing signal processing algorithms' computational performance. Moreover, the highlights and future work of this study will be discussed in Chapter 7.

In the next chapter, the present thesis proposes implementing the distributed computing concept to the most commonly used spectral analysis method, named fast Fourier transform, which emphasises the advantages of using parallel processing in real time. The derivation of the newly proposed algorithm and the analytical calculation of computational complexity are provided.

3.6 References

Aarabi, A., Kazemi, K., Grebe, R., Moghaddam, H. A. & Wallois, F. 2009. Detection of EEG transients in neonates and older children using a system based on dynamic time-warping template matching and spatial dipole clustering. *Neuroimage,* 48**,** 50-62.

Adeli, H., Zhou, Z. & Dadmehr, N. 2003. Analysis of EEG records in an epileptic patient using wavelet transform. *Journal of neuroscience methods,* 123**,** 69-87.

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,* 2014.

Alam, S. S. & Bhuiyan, M. I. H. 2013. Detection of seizure and epilepsy using higher order statistics in the EMD domain. *IEEE journal of biomedical and health informatics,* 17**,** 312-318.

Aluru, S. & Jammula, N. 2013. A review of hardware acceleration for computational genomics. *IEEE Design & Test,* 31**,** 19-30.

Da Silveira, T. L., Kozakevicius, A. J. & Rodrigues, C. R. 2017. Single-channel EEG sleep stage classification based on a streamlined set of statistical features in wavelet domain. *Medical & biological engineering & computing,* 55**,** 343-352.

Delorme, A., Rousselet, G. A., Macé, M. J.-M. & Fabre-Thorpe, M. 2004. Interaction of top-down and bottom-up processing in the fast visual analysis of natural scenes. *Cognitive Brain Research,* 19**,** 103-113.

Dohnálek, P., Gajdoš, P., Peterek, T. & Penhaker, M. Pattern recognition in EEG cognitive signals accelerated by GPU. International Joint Conference CISIS'12-ICEUTE 12-SOCO 12 Special Sessions, 2013. Springer, 477-485.

Fang, J., Varbanescu, A. L. & Sips, H. A comprehensive performance comparison of CUDA and OpenCL. 2011 International Conference on Parallel Processing, 2011. IEEE, 216-225.

He, Y., Folta, F., Chan, C.-W., Spears, S. & Gu, C. 2014. Parallel multiple bitrate video encoding to reduce latency and dependences between groups of pictures. Google Patents.

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J. & Scholkopf, B. 1998. Support vector machines. *IEEE Intelligent Systems and their applications,* 13**,** 18-28.

Huang, D., Qian, K., Fei, D.-Y., Jia, W., Chen, X. & Bai, O. 2012. Electroencephalography (EEG)-based brain–computer interface (BCI): A 2-D virtual wheelchair control based on event-related desynchronization/synchronization and state control. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 20**,** 379-388.

Jung, T. P., Makeig, S., Humphries, C., Lee, T. W., Mckeown, M. J., Iragui, V. & Sejnowski, T. J. 2000. Removing electroencephalographic artifacts by blind source separation. *Psychophysiology,* 37**,** 163-178.

Khan, Y. D., Jamil, M., Hussain, W., Rasool, N., Khan, S. A. & Chou, K.-C. 2019. pSSbond-PseAAC: Prediction of disulfide bonding sites by integration of PseAAC and statistical moments. *Journal of Theoretical Biology,* 463**,** 47-55.

Ko, K.-E., Yang, H.-C. & Sim, K.-B. 2009. Emotion recognition using EEG signals with relative power values and Bayesian network. *International Journal of Control, Automation and Systems,* 7**,** 865.

Leeb, R., Brunner, C., Müller-Putz, G., Schlögl, A. & Pfurtscheller, G. 2008. BCI Competition 2008–Graz data set B. *Graz University of Technology, Austria*, 1-6.

Leeb, R., Lee, F., Keinrath, C., Scherer, R., Bischof, H. & Pfurtscheller, G. 2007. Brain–computer communication: motivation, aim, and impact of exploring a virtual apartment. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 15**,** 473-482.

Lehmann, C., Koenig, T., Jelic, V., Prichep, L., John, R. E., Wahlund, L.-O., Dodge, Y. & Dierks, T. 2007. Application and comparison of classification algorithms for recognition of Alzheimer's disease in electrical brain activity (EEG). *Journal of neuroscience methods,* 161**,** 342-350.

Li, Y., Li, X., Ratcliffe, M., Liu, L., Qi, Y. & Liu, Q. A real-time EEG-based BCI system for attention recognition in ubiquitous environment. Proceedings of 2011 international workshop on Ubiquitous affective awareness and intelligent interaction, 2011. 33-40.

Lotte, F., Bougrain, L., Cichocki, A., Clerc, M., Congedo, M., Rakotomamonjy, A. & Yger, F. 2018. A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update. *Journal of neural engineering,* 15**,** 031005.

Martinez, P., Bakardjian, H. & Cichocki, A. 2007. Fully Online Multicommand Brain-Computer Interface with Visual Neurofeedback Using SSVEP Paradigm. *Computational Intelligence and Neuroscience,* 2007**,** 094561.

Palaniappan, R. Brain computer interface design using band powers extracted during mental tasks. Neural Engineering, 2005. Conference Proceedings. 2nd International IEEE EMBS Conference on, 2005. IEEE, 321-324.

Pfurtscheller, G., Brunner, C., Schlögl, A. & Da Silva, F. L. 2006. Mu rhythm (de) synchronization and EEG single-trial classification of different motor imagery tasks. *NeuroImage,* 31**,** 153-159.

Qu, H. & Gotman, J. 1997. A patient-specific algorithm for the detection of seizure onset in long-term EEG monitoring: possible use as a warning device. *IEEE transactions on biomedical engineering,* 44**,** 115-122.

Soliman, S. S. & Hsue, S.-Z. 1992. Signal classification using statistical moments. *IEEE Transactions on Communications,* 40**,** 908-916.

Strope, B., Beaufays, F. & Siohan, O. 2013. Speech recognition with parallel recognition tasks. Google Patents.

Subasi, A. & Gursoy, M. I. 2010. EEG signal classification using PCA, ICA, LDA and support vector machines. *Expert systems with applications,* 37**,** 8659-8666.

Wang, N., Lu, P., Zhang, L., Li, S. & Hu, H. Accelerated Rendering and Fast Reconstruction of EEG Data in Real-Time BCI. Proceedings of the 2015 Chinese Intelligent Automation Conference, 2015. Springer, 61-75.

Xu, R., Jiang, N., Lin, C., Mrachacz-Kersting, N., Dremstrup, K. & Farina, D. 2013. Enhanced low-latency detection of motor intention from EEG for closed-loop brain-computer interface applications. *IEEE Transactions on Biomedical Engineering,* 61**,** 288-296.

Zaki, M. J., Parthasarathy, S., Ogihara, M. & Li, W. 1997. Parallel algorithms for discovery of association rules. *Data mining and knowledge discovery,* 1**,** 343-373.

CHAPTER 4

ENHANCED SLIDING DFT ALGORITHM

An aim for this thesis is to enhance computing performance in BCI applications using parallel processing techniques. Previous chapters have demonstrated promising results with respect to speed enhancement. However, drawbacks include being very hardware and platform dependent, challenging to expand when adding more input channels, and being insensitive to recording frequency changes. These are common difficulties associated with real-time BCI applications since it is often necessary to adjust sampling frequency and channel count to accommodate specific applications. Accordingly, to improve performance and computational complexity, this chapter of the thesis explores some of the algorithms used to create a more robust system for real-time feature extraction.

Here, the improvement of frequency-domain feature extraction based on a sliding discrete Fourier transform algorithm for real-time signal processing is determined. Furthermore, the momentary matrix transformation, an essential tool for developing a new algorithm named "enhanced sliding DFT (e-SDFT)", has been explained and derived in this chapter. Moreover, presented here is a performance evaluation of the proposed e-SDFT algorithm and a comparison with existing methods using numerical and analytical methods. Finally, findings and potential applications attained from this study are also discussed in this chapter.

4.1 Introduction

Discrete Fourier transform (DFT) is a long-standing approach used in analysing the spectral strength of signals. In medical applications relying on analysing physiological signals, the DFT is a standard toolbox in the signal processing pipeline. In addition,

frequency analysis and the DFT can be used as a model feature in classical signal processing algorithms (Samiee et al., 2014; Vidyaratne and Iftekharuddin, 2017). Used for calculating a spectrum of a finite sequence of discrete-time signals resulting in an output frequency sequence, the DFT can be applied to one-dimension or multi-dimension for both real-valued input and complex-valued input.

Due to the high computational complexity of the DFT algorithm, a fast Fourier transform (FFT) method was proposed to reduce those complexities (Cochran et al., 1967). The FFT is a powerful method in signal processing and has become a standard approach for frequency analysis. With the advantages of the traditional FFT, the computational complexity of calculating Fourier coefficients can decrease up to $\log N$ times compared to the direct approach. In addition to the FFT algorithms, many studies (Ma et al., 2015; Ayinala et al., 2013; Li et al., 2014) proposed a modified FFT that is more flexible and offers less complexity than the conventional FFT. In addition to the major restriction of the classic FFT, the input length has to be 2^n samples that may not suit some of the current digital signal processing (DSP) applications that need a specific input length, such as biosignal applications (Samiee et al., 2014; Vidyaratne and Iftekharuddin, 2017).

When calculating DFT coefficients in real-time processing, the approach to computing the DFT needs to be improved or carefully modified as all parameters can affect the performance. Previous research on improving the computational issues and many studies have explored reducing the number of complexities of the DFT scheme and resolving the 2^nissue; for example, prime factor algorithm (Guo et al., 2011), shifted DFT interpolation (Serbes, 2018), Winograd algorithm (Liang et al., 2019), a scaled DFT (Bi and Chen, 1998), and multidimensional DFT (Yu and Swartzlander, 2001; Wang et al., 2018). While algorithms can be optimised to achieve speedup, the optimisation always comes with the potential loss of accuracy or other perspectives. For instance, using different variable structures on the same algorithm provides different estimation errors (Hu, 1992). This issue has to be carefully examined as tradeoff risks associated with speedup.

In order to ultimately gain DFT capability in real-time environments (such as calculating the DFT coefficients on a per-sample basis called “sliding-window DFT”),

some DFT calculations can be discarded or estimated as they contain several duplicated input samples. The sliding DFT (SDFT) method has been proposed previously (Jacobsen and Lyons, 2003; Jacobsen and Lyons, 2004) as a way of reducing these repetitive calculations. According to the major advantage of the SDFT algorithm, the conventional SDFT algorithm is computationally efficient. However, its recursive formula typically suffers from the accumulation of round-off error when floating-point arithmetic is used, which cause potential instabilities or inaccurate output if running a system for an extended period.

Presently, most modern processing units are developed based on multi-core architectures, offering parallelism and high-performance computing. As a result, various computing devices have used the concept that parallel processing in hardware acceleration can improve computational speed. Such devices offer great potential for real-time applications such as signal and image processing or encoding and rendering. A multi-core processor (CPU) and a graphics processing unit (GPU) are currently the most widely-used devices in such real-time applications, including medical imaging (Cheng et al., 2019; Hosny et al., 2018) and biological signal processing (Yu et al., 2019; Bui et al., 2020). Implementing a parallel computing concept into multiple compute units is the key that makes the multi-core CPU and the GPU powerful. In recent times, many studies have promoted the advantages of multi-core CPU and GPU in general signal processing (Liavas et al., 2017) and image processing (Choe et al., 2013; Garcia-Rial et al., 2017). However, those applications have been required to amend their approaches and techniques in order to work flawlessly with the multi-core architecture.

This study presents an alternative approach called “enhanced sliding discrete Fourier transform (eSDFT)” that satisfies sliding-window applications such as those used in processing time-varying bio-signals, e.g. electroencephalogram (EEG) and electromyogram (EMG); or any signals that require spectrum estimation. The principle behind eSDFT is based on the momentary Fourier transform, which is identical to the sliding DFT algorithm, by reducing excessively repetitive calculations required to compute the DFT coefficients. Moreover, the proposed method can offer more stability of DFT calculation than the conventional SDFT as a cumulative error caused by the

rounding effect in floating-point arithmetic can be controlled within range by using the support of a multi-core processor. The proposed method can be applied to real-time signal processing when DFT coefficients of each incoming sample are needed.

4.2 Background

4.2.1 Discrete Fourier Transform (DFT)

Discrete Fourier transform (DFT) is designed to transform time-domain samples to frequency-domain components, in general, by determining Fourier coefficients of discrete-time signals. Usually, discrete-time signals can be categorised as periodic and aperiodic, while most digitised physiological signals are real and aperiodic. The Fourier transform of aperiodic discrete-time signals $x[n]$ is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}} \tag{4.1}$$

where $X[k]$ is the DFT coefficients of the N-sample input signal $x[n]$ where $n \in [0, N-1]$. The transformation produces N complex Fourier coefficients, and k represents the frequency index in $X[k]$, where $k \in [0, N-1]$. Equation 4.1 can be simplified for compactness by defining $W_N = e^{-j2\pi/N}$, commonly called the "twiddle factor", therefore

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk}. \tag{4.2}$$

On the other hand, the inverse of the DFT can be express by

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-nk}. \tag{4.3}$$

Given $x[n]$ is a complex input signal with N samples inside the domain $n \in [0, N-1]$, the computational complexity of the DFT $X[k]$ are N complex multiplications and $N-1$ complex additions, yielding the total complexity of $\mathcal{O}(N^2)$.

4.2.2 Matrix-Vector Form of DFT

Alternatively, DFT in Equation 4.2 can be converted into the operation of matrix and vector, thus determining the DFT coefficients can be more conveniently viewed as matrix-vector multiplication that is a general linear transformation. Considering the input sequence $\mathbf{x}$ with a length of N, the Fourier coefficients sequence $\mathbf{y}$ with a length of N of the input $\mathbf{x}$ can be defined by

$$\mathbf{y} = \mathbf{F}\mathbf{x} \tag{4.4}$$

where $\mathbf{F}$ is the N-by-N linear transformation matrix named DFT matrix that has many remarkable properties, particularly its invertibility. According to the DFT matrix, the "$\mathbf{F}$" term is a combined matrix of the twiddle factors defined as such:

$$\mathbf{F} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}. \tag{4.5}$$

It is known that $\mathbf{F}$ can be a unitary matrix and symmetric, which is always invertible. Hence, the inverted form of Equation 4.4 can be obtained as follows:

$$\mathbf{x} = \mathbf{F}^{-1}\mathbf{y}. \tag{4.6}$$

With the above-mentioned unitary property of the DFT matrix, $\mathbf{F}$ can achieve unitarity with an appropriate selection of scaling that is $1/\sqrt{N}$. This means that its inverse will be equal to its conjugate transpose therefore $\mathbf{F}^{-1} = \mathbf{F}^{*}$.

4.2.3 Fast Fourier Transform (FFT)

Regarding signal processing systems, DFT plays an essential role in designing, analysing, implementing, and operating. However, the major challenge when applying DFT in real time is the computational complexity that increases exponentially by the input data size. Due to its intensive computational requirements, a fast Fourier transform (FFT) is proposed to overcome the existing complexity in DFT. There are

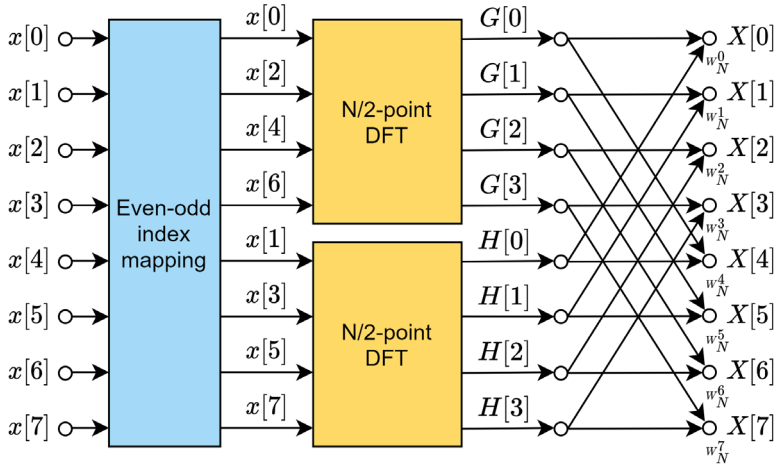

Figure 4.1. Illustrating the architecture of radix-2 DIT FFT algorithm for eight-point DFT. The input sequence is spitted into even index and odd index then 2-point DFT is performed stage-by-stage using butterfly operation.

many popular FFT architectures widely used in the field of modern digital signal processing, for example, the Cooley-Tukey algorithm (Nussbaumer, 1981), Winograd Fourier transform algorithm (WFTA) (Silverman, 1977), and prime factor algorithm (PFA) (Kolba and Parks, 1977). The most straightforward algorithm of computing FFT is called radix-2 FFT, including the decimation-in-time (radix-2 DIT) and decimation-in-frequency (radix-2 DIF).

According to the radix-2 FFT algorithm, a requirement is that the input length has to be a power of two, leading to the very efficient computation that yields the computational complexity of $\mathcal{O}(N \log N)$ to provide a significant saving over the direct computation of the DFT. This study selected the radix-2 DIT FFT to compare the computational performance among the other DFT algorithms due to its popularity. The concept of this FFT algorithm is to split the input into cascading groups of two, including even index and odd index, then perform the DFT for each group. The results can be achieved from the combination of the groups. Figure 4.1 illustrates the complete architecture of the radix-2 DIT FFT for computing the eight-point DFT.

4.2.4 Recursive Momentary Matrix Transformation

For a general matrix transformation that requires computing the output sequence in incremental steps, it can be completed in an efficient recursive form by reducing repetitive arithmetic operations, for instance, calculating the DFT coefficients on

blocks of data every incoming input sample named “momentary Fourier transform” (MFT) proposed by (Dudas, 1986). The concept of the momentary sequence is that one sample enters into the last element of the sequence while another sample leaves the first element of the sequence, called first-in, first-out (FIFO). Moreover, the current output will be updated from the previously calculated and stored value using for the next iteration.

For a sample of an arbitrary complex-valued sequence $\mathbf{x}_i$ and $\mathbf{x}_{i-1}$ with a length of N and the last element ending at the sample i and $i-1$, respectively, the sequences can be represented by the following vectors

$$\mathbf{x}_i = \begin{bmatrix} x_{i-(N-1)} \\ \vdots \\ x_{i-1} \\ x_i \end{bmatrix}, \quad \mathbf{x}_{i-1} = \begin{bmatrix} x_{i-N} \\ \vdots \\ x_{i-2} \\ x_{i-1} \end{bmatrix} \tag{4.7}$$

where $X[k]$ is the DFT coefficients of the N-sample input

Let $\mathbf{T}$ be a $N \times N$ non-singular transformation matrix with its inverse $\mathbf{T}^{-1}$, the sequence $\mathbf{y}_i$ and $\mathbf{y}_{i-1}$ of a linear transformation of $\mathbf{x}_i$ and $\mathbf{x}_{i-1}$ can be defined by

$$\mathbf{y}_i = \mathbf{T}\mathbf{x}_i, \quad \mathbf{y}_{i-1} = \mathbf{T}\mathbf{x}_{i-1}. \tag{4.8}$$

Let $\mathbf{P}$ be a $N \times N$ permutation matrix that results in a one-element circular shift of a sequence. The visualisation of the permutation matrix is as follow:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & 0 & 0 & \ddots & \vdots \\ 0 & & \vdots & \ddots & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{4.9}$$

By multiplying $\mathbf{P}$ with the vector $\mathbf{x}_{i-1}$ in Equation 4.7, all elements are shifted with one sample as follows:

$$\mathbf{P}\mathbf{x}_{i-1} = [x_{i-(N-1)} \quad \cdots \quad x_{i-1} \quad x_{i-N}]^{\mathrm{T}}. \tag{4.10}$$

As most elements in $\mathbf{x}_i$ and $\mathbf{x}_{i-1}$ are identical but different indices, this can be expressed in another form by using a modification vector $\Delta\mathbf{x}_i$

$$\begin{aligned}\mathbf{x}_i &= \begin{bmatrix} x_{i-(N-1)} \\ \vdots \\ x_{i-1} \\ x_{i-N} \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ x_i - x_{i-N} \end{bmatrix} \\ &= \mathbf{P}\mathbf{x}_{i-1} + \Delta\mathbf{x}_i. \end{aligned} \tag{4.11}$$

By substituting Equation 4.11 into Equation 4.8 with using its inverse form that is $\mathbf{x}_{i-1} = \mathbf{T}^{-1}\mathbf{y}_{i-1}$, the unique formula of the momentary matrix transform can be obtained:

$$\begin{aligned}\mathbf{y}_i &= \mathbf{T}\mathbf{x}_i \\ &= \mathbf{T}[\mathbf{P}\mathbf{x}_{i-1} + \Delta\mathbf{x}_i] \\ &= \mathbf{T}\mathbf{P}\mathbf{T}^{-1}\mathbf{y}_{i-1} + \mathbf{T}\Delta\mathbf{x}_i. \end{aligned} \tag{4.12}$$

The above equation is known as a recursive relationship that the sequence $\mathbf{y}_i$ can be obtained by the previous sequence $\mathbf{y}_{i-1}$, the difference between the entering sample and the leaving sample of the sequence $\mathbf{x}_i$, and the invertible matrix $\mathbf{T}$ (Dudas, 1986).

The best efficiency of using the momentary matrix transform can be achieved only if the term $\mathbf{T}\mathbf{P}\mathbf{T}^{-1}$ in Equation 4.8 is diagonal. This means that each element of $\mathbf{y}_i$ can be determined independently. The proof of the diagonal property is clearly stated in (Albrecht and Cumming, 1999). Let $\boldsymbol{\alpha}$ be a diagonal matrix of $\mathbf{T}\mathbf{P}\mathbf{T}^{-1}$ with N distinct elements inside. Equation 4.12 can be rearranged to the new form by

$$\mathbf{y}_i = \boldsymbol{\alpha}\mathbf{y}_{i-1} + \mathbf{T}_{N-1}(x_i - x_{i-N}) \tag{4.13}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_0 & 0 & \cdots & 0 \\ 0 & \alpha_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \alpha_{N-1} \end{bmatrix} \tag{4.14}$$

where $\mathbf{T}_{N-1}$ is the last column of the transformation matrix $\mathbf{T}$.

4.2.5 Momentary Fourier Transform (MFT)

Regarding the recursive momentary matrix transformation, it can be applied to calculate DFT coefficients. The DFT matrix produces the diagonal matrix of $\mathbf{TPT}^{-1}$ as the eigenvectors of the matrix P are the columns of the DFT matrix, a property discussed in (Albrecht and Cumming, 1999). By substituting the DFT matrix Equation 4.5 into Equation 4.12, the momentary Fourier transform (MFT) is obtained

$$\mathbf{y}_i = \mathbf{FPF}^{-1}\mathbf{y}_{i-1} + \mathbf{F}\Delta\mathbf{x}_i. \tag{4.15}$$

Equation 4.15 can be rearranged by using the property of the complex Nth root of unity ($W_N^{-k} = W_N^{N-k}$) therefore

$$\mathbf{y}_i = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & W_N^{-1} & 0 & \cdots & 0 \\ 0 & 0 & W_N^{-2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & W_N^{-(N-1)} \end{bmatrix} \mathbf{y}_i + \begin{bmatrix} 1 \\ W_N^{-1} \\ W_N^{-2} \\ \vdots \\ W_N^{-(N-1)} \end{bmatrix} (x_i - x_{i-N}). \tag{4.16}$$

The above recursive expression represents the MFT formula and each Fourier coefficient of $\mathbf{y}_i$ at index k can be obtained by

$$y_{i,k} = W_N^{-k}\left(y_{i-1,k} + x_i - x_{i-N}\right). \tag{4.17}$$

Equation 4.17 is identical to the expression of the sliding discrete Fourier transform (SDFT) proposed in the previous studies (Jacobsen and Lyons, 2003; Jacobsen and Lyons, 2004) and is a popular algorithm used in non-parametric spectrum estimation. Hence, the term "MFT algorithm" and "SDFT algorithm" are interchangeable. Accordingly, the MFT/SDFT algorithm can be more efficient when only a few Fourier coefficients are of interest. When calculating for N coefficients, $x_i - x_{i-N}$ can be calculated once and used for determining other coefficients. This, therefore, results in the total complexity of N complex multiplications and $N+1$ complex additions. For a real-valued input, the computation using MFT can be reduced to half the calculation because the Fourier transform of a real signal is guaranteed to be Hermitian, in which half of the computations are redundant.

4.2.6 Error Accumulation in MFT/SDFT Algorithm

Accumulation of round-off errors in recursive formulas can occur when solving problems in which the solution results from a large number of consecutively performed arithmetic operations. The round-off error can be magnified by some unstable algorithms, such as recursive algorithms. The following general recursive model can demonstrate the accumulation error.

$$y_n = a y_{n-1}. \tag{4.18}$$

Suppose y_0 is an initial value and has a small round-off error ε, this means the initial input to the above equation is $y_0+\varepsilon$ instead of y_0. The calculation of the output value at time n in Equation 4.18 can be replaced by

$$y_n = a^n y_0 + a^n \varepsilon. \tag{4.19}$$

It can be seen that in Equation 4.19, if a >1, the small round-off error can be amplified with the factor of a^n, resulting in the inaccuracy of output when n is relatively large. According to the MFT/SDFT expression in Equation 4.17, the term W_N can lead to the same error accumulation issue when a long sequence of DFT calculations is applied on an initial input with round-off error, such as the use of floating-point arithmetic. It consequently makes the algorithm unstable and produces inaccurate output.

4.3 Related Works

Several studies have investigated the sliding-window DFT based on recursive perspectives (Jacobsen and Lyons, 2003; Albrecht and Cumming, 1999; Varkonyi-Koczy, 1995; Chen et al., 1993). With the advantages of recursiveness, the complexity of the original DFT calculation, especially for real-world applications, can be proportionally or exponentially reduced. However, although the recursive method provides less complexity than the non-recursive approach, the memory requirement is one of the most challenging issues when implementing it in real-time applications. Hence, system optimisation is necessary for practical use.

In addition to a computational improvement of the DFT, several studies have considered non-recursive methods (Murakami, 1994; Johnson and Frigo, 2006; Sorensen et al., 1987). Other methods for sliding window FFT (Macias and Exposito, 1998; Exposito and Macfas, 2000) were proposed that offer excellent performance when only specific individual harmonics are needed. The use of non-recursive algorithms can be implemented into a single field-programmable gate array (FPGA) and directly mapped to the hardware specification. However, for some small processing units that lack computational power, such as a microcontroller unit (MCU), applying those algorithms may exhaust hardware limitations quickly if system management, i.e. memory and data transmission, is not well arranged.

Dudas proposed the momentary Fourier transform (MFT) in 1986 that presented the recursive calculation for sliding DFT using momentary matrix transformation (Dudas, 1986). Later, this concept had been of interest and was renamed a "sliding-window DFT" (SDFT). The original SDFT algorithm proposed in (Jacobsen and Lyons, 2003; Jacobsen and Lyons, 2004) provides high computational efficiency; however, the recursive structure can introduce a cumulative error due to the accumulation of round-off errors resulting in imprecise output. Several studies have proposed alternative SDFT algorithms to overcome the stability issue by using signal modulation (Duda, 2010), modified filtering (Gudovskiy and Chu, 2017), optimisation (Park, 2017), and CORDIC algorithm (Kulshreshtha and Dhar, 2018). However, those proposed methods are based on a single processing unit.

Regarding the use of MFT/SDFT, the algorithm was applied to real-time signal processing such as laser polarimeter by Harvie (Harvie et al., 2020) and power-line interference (PLI) removal in biosignals (Mishra et al., 2014). The computational performance of the MFT/SDFT showed enhanced efficiency when limited coefficients of the spectrum are required, compared to the conventional FFT. However, if multiple shifts in samples occur, more computations are required, resulting in inefficient performance. Based on a search of the literature, no study to date has presented the DFT calculation for multiple samples shifted based on the momentary matrix transformation, and no previous research has addressed the cumulative error issue that occurs in the original SDFT algorithm based on multiple shifts between samples.

When performing recursive floating-point operations such as the MFT/SDFT, the output estimate is often affected, resulting in a cumulative inaccuracy. To address this problem, the error may be kept within a limited range by periodically updating the actual Fourier coefficients in order to minimise the cumulative error. This concept may be realised with the help of multi-core operating systems since it is feasible to do this by working in parallel without significantly increasing computational overhead. However, to functionalise this, we need to alter the original MFT/SDFT method in real-time systems where the coefficient is supplied by a shift (delay). To do this, this study addresses two main problems, including (a) calculating the SDFT based on multiple shifts between samples and (b) minimising the error accumulation caused by the rounding effect in the original SDFT algorithm using multiple shifts SDFT.

4.4 Materials and Methodology

4.4.1 Recursive Momentary Matrix Transform of *m*-Sample Shift

As mentioned previously, no real-time biosignal processing to date has proposed implementing DFT calculation using the MFT algorithm based on m-sample shift. This study introduces another recursive DFT formula using the shifts between samples.

According to Equation 4.12, the recursive expression can be extended to determine the relationship between the current transformed sequence ($\mathbf{y}_i$) and the previous m-sample shifted sequence ($\mathbf{y}_{i-m}$), where $m \in [1, N-2]$ as follows:

$$\begin{aligned}
\mathbf{y}_i &= \mathbf{TPT}^{-1}\mathbf{y}_{i-1} + \mathbf{T}\Delta\mathbf{x}_i \\
&= \mathbf{TPT}^{-1}(\mathbf{TPT}^{-1}\mathbf{y}_{i-2} + \mathbf{T}\Delta\mathbf{x}_{i-1}) + \mathbf{T}\Delta\mathbf{x}_i \\
&= \mathbf{TP}^2\mathbf{T}^{-1}\mathbf{y}_{i-2} + \mathbf{TP}\Delta\mathbf{x}_{i-1} + \mathbf{T}\Delta\mathbf{x}_i \\
&= \quad\quad\quad\quad \vdots \\
&= \mathbf{TP}^m\mathbf{T}^{-1}\mathbf{y}_{i-m} + \mathbf{TP}^{m-1}\Delta\mathbf{x}_{i-(m-1)} + \cdots + \mathbf{TP}\Delta\mathbf{x}_{i-1} + \mathbf{T}\Delta\mathbf{x}_i \\
&= \mathbf{TP}^m\mathbf{T}^{-1}\mathbf{y}_{i-m} + \mathbf{T}\left(\mathbf{P}^{m-1}\Delta\mathbf{x}_{i-(m-1)} + \cdots + \mathbf{P}\Delta\mathbf{x}_{i-1} + \Delta\mathbf{x}_i\right) \\
&= \mathbf{TP}^m\mathbf{T}^{-1}\mathbf{y}_{i-m} + \mathbf{T}\sum_{s=0}^{m-1}\mathbf{P}^s\,\Delta\mathbf{x}_{i-s}.
\end{aligned} \tag{4.20}$$

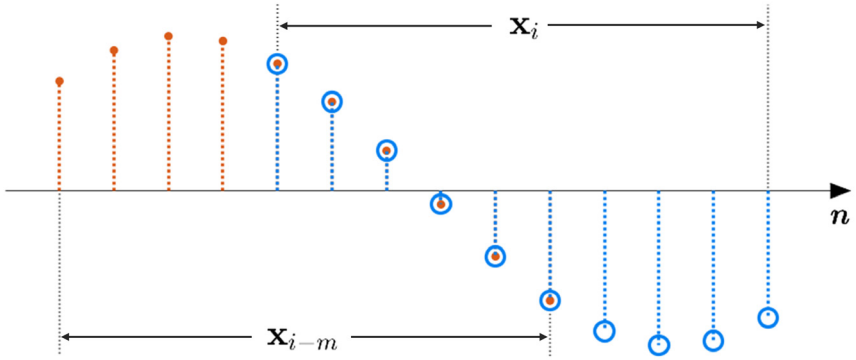

Figure 4.2. Demonstrating the differences of input sequence $\Delta_{\boldsymbol{m}}\mathbf{x}_{\boldsymbol{i}}$ for the sample $\boldsymbol{N = 10}$ and the shift $\boldsymbol{m = 4}$. The current analysed window is $\mathbf{x}_{\boldsymbol{i}}$ and the previously analysed window is $\mathbf{x}_{\boldsymbol{i-4}}$, resulting in 6 overlapping samples.

The latter term in Equation 4.20 can be simplified more by using a m-sample shifted modification sequence $\Delta_m\mathbf{x}_i$

$$\mathbf{y}_i = \mathbf{T}\mathbf{P}^m\mathbf{T}^{-1}\mathbf{y}_{i-m} + \mathbf{T}\Delta_m\mathbf{x}_i \tag{4.21}$$

$$\Delta_m\mathbf{x}_i = \sum_{s=0}^{m-1}\mathbf{P}^s\,\Delta\mathbf{x}_{i-s} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ x_{i-(m-1)} - x_{i-N-(m-1)} \\ \vdots \\ x_{i-1} - x_{i-N-1} \\ x_i - x_{i-N} \end{bmatrix}. \tag{4.22}$$

Equation 4.21 is proposed as a general form of the momentary matrix transform for m-sample shift. To understand the concept of the modification sequence $\Delta_m\mathbf{x}_i$, Figure 4.2 demonstrates two different sequences ($\mathbf{y}_i$ and $\mathbf{y}_{i-m}$) that are analysed through the window size of 10 samples, and the current sequence $\mathbf{y}_i$ is shifted from the previous sequence $\mathbf{y}_{i-m}$ for 4 samples. The window moves one sample at a time, by one sample entering and one sample leaving the sequence. Accordingly, there are 6 overlapping samples between the sequences.

4.4.2 Reformulated Recursive Momentary Matrix Transformation for Multiple Shifts

Following the general form of the recursive momentary matrix transformation, the modification vector will introduce many zero elements that can be discarded from the practical calculation if a few samples are shifted. The following derivation represents the reduction of the multiplication, which focuses on non-zero elements only. Let a sequence $\boldsymbol{\beta}$ be the differences between the samples of Equation 4.22, in which the first non-zero element is the first index of $\boldsymbol{\beta}$.

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{bmatrix} = \begin{bmatrix} x_{i-(m-1)} - x_{i-N-(m-1)} \\ x_{i-(m-2)} - x_{i-N-(m-2)} \\ \vdots \\ x_i - x_{i-N} \end{bmatrix}. \tag{4.23}$$

By replacing Equation 4.22 with Equation 4.23 and then substituting into Equation 4.21, an alternative momentary matrix transformation for m-sample shift is obtained

$$\mathbf{y}_i = \mathbf{T}\mathbf{P}^m\mathbf{T}^{-1}\mathbf{y}_{i-m} + \mathbf{T}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{bmatrix}. \tag{4.24}$$

It can be seen that the second term in Equation 4.24 can be expressed more by separating it into two terms as

$$\mathbf{T}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{m-1} \end{bmatrix} = \mathbf{T}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ \beta_0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \mathbf{T}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{m-1} \end{bmatrix}. \tag{4.25}$$

To understand the concept of separation, the above equation can be visualised in Figure 4.3 that only some elements of the matrix $\mathbf{T}$ and the sequence $\Delta_m \mathbf{x}_i$ require a particular multiplication. Hence, the equation can be focused on only non-zero elements, which can reduce computational complexity more.

As most of the elements in Equation 4.25 are zero that are discarded when performing multiplication, hence Equation 4.25 can be reduced to

$$\mathbf{T}\begin{bmatrix}0\\ \vdots\\ 0\\ \beta_0\\ 0\\ 0\\ \vdots\\ 0\end{bmatrix} = \mathbf{T}_{N-m}\beta_0, \quad \mathbf{T}\begin{bmatrix}0\\ \vdots\\ 0\\ 0\\ \beta_1\\ \beta_2\\ \vdots\\ \beta_{m-1}\end{bmatrix} = \sum_{s=1}^{m-1} T_{k,N-m+s}\beta_s \tag{4.26}$$

where $\mathbf{T}_{N-m}$ is the $(N-m)^{\text{th}}$ column of the transformation matrix ($\mathbf{T}$). As Equation 4.26 shows that some indices of $\mathbf{T}$ require multiplication with $\boldsymbol{\beta}$, the original transformation matrix ($\mathbf{T}$) can be replaced by the modified transformation matrix ($\boldsymbol{\tau}$):

$$\sum_{s=1}^{m-1} T_{k,N-m+s}\beta_s = \boldsymbol{\tau}\boldsymbol{\beta}_{\boldsymbol{\tau}} \tag{4.27}$$

where $\boldsymbol{\tau}(k,s) = \mathbf{T}(k, N-m+s)$ are within $s \in [1, m-1]$ and $k \in [0, N-1]$.

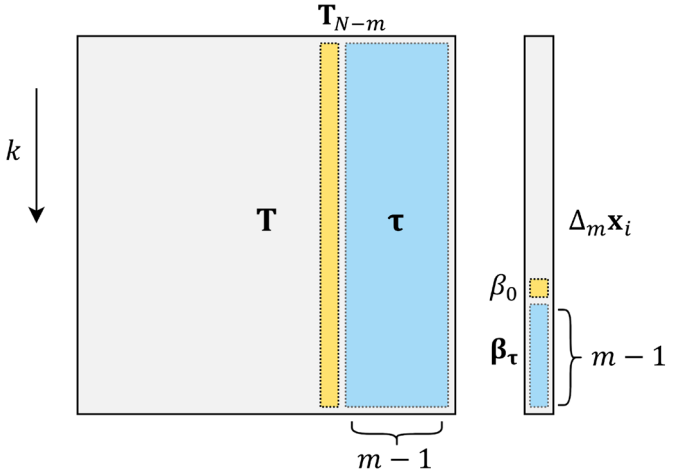

Figure 4.3. Demonstrating some multiplications in Equation 4.25 are performed. As most of the element in the sequence $\Delta_m \mathbf{x}_i$ are zeros except $\boldsymbol{\beta}$, the actual multiplication can be reduced to Equation 4.26.

4.4.3 *m*-Sample Shift Momentary Fourier Transform (m-MFT)

According to Equation 4.21, the equation can be implemented to the Fourier transform as the matrix (**T**) has some interesting properties and can be rearranged to improve the practical computation. For m-sample shifted momentary Fourier transform (m-MFT), substituting **F** into Equation 4.21 hence

$$\mathbf{y}_i = \mathbf{F}\mathbf{P}^m\mathbf{F}^{-1}\mathbf{y}_{i-m} + \mathbf{F}\Delta_m\mathbf{x}_i. \tag{4.28}$$

As demonstrated in Equation 4.15, the DFT matrix was applied to the momentary matrix transformation. The term $\mathbf{F}\mathbf{P}\mathbf{F}^{-1}$ produces a diagonal matrix, which can be represented by $\boldsymbol{\alpha}$ as shown in Equation 4.13. Because **P** has N linearly independent eigenvectors, which are related to **F**, **P** is a diagonalisable matrix. Similarly, the term $\mathbf{F}\mathbf{P}^m\mathbf{F}^{-1}$ can be extended to determine the relationship with $\mathbf{F}\mathbf{P}\mathbf{F}^{-1}$ by using matrix diagonalisation properties as follows:

$$\mathbf{F}\mathbf{P}^m\mathbf{F}^{-1} = (\mathbf{F}\mathbf{P}\mathbf{F}^{-1})^m. \tag{4.29}$$

Since $\mathbf{F}\mathbf{P}\mathbf{F}^{-1}$ is diagonal, $\mathbf{F}\mathbf{P}^m\mathbf{F}^{-1}$ also holds the same diagonal property following Equation 4.29. Besides, it is evident in (Albrecht and Cumming, 1999) that the main diagonal of the matrix α is identical to the last column of the DFT matrix (**F**). Correspondingly, the main diagonal of $\mathbf{F}\mathbf{P}^m\mathbf{F}^{-1}$ in Equation 4.29 is equal to the last column of the DFT matrix to the power of m. Interestingly, the result is the same as the $(N-m)^{\text{th}}$ column of the DFT matrix ($\mathbf{F}_{N-m}$). Hence there is no need to calculate this value if the whole transformation matrix (**F**) is stored beforehand. The relationship of those mentioned terms can be expressed by

$$\mathbf{F}\mathbf{P}^m\mathbf{F}^{-1} = \boldsymbol{\alpha}^m = diag(\mathbf{F}_{N-m}). \tag{4.30}$$

Equation 4.28 can be reformulated following Equations 4.24, 4.25, and 4.26 by using the above relationship and the property of the modified transformation matrix ($\boldsymbol{\tau}$) in Equation 4.27, hence

$$\mathbf{y}_i = \mathbf{F}\mathbf{P}^m\mathbf{F}^{-1}\mathbf{y}_{i-m} + \mathbf{F}\Delta_m\mathbf{x}_i \tag{4.31}$$

$$= \boldsymbol{\alpha}^m \mathbf{y}_{i-m} + \boldsymbol{\alpha}^m \beta_0 + \sum_{s=1}^{m-1} F_{k,N-m+s} \beta_s$$
$$= \boldsymbol{\alpha}^m (\mathbf{y}_{i-m} + \beta_0) + \boldsymbol{\tau}\boldsymbol{\beta}_{\boldsymbol{\tau}}.$$

The above recursive equation denotes the reduced m-MFT formula, in which the latter term requires less multiplication than its initial formula. In Equation 4.31, each coefficient of $\mathbf{y}_i$ can be obtained by

$$y_{i,k} = \alpha_k^m \left(y_{i-m,k} + \beta_0\right) + \boldsymbol{\tau}_k \cdot \boldsymbol{\beta}_{\boldsymbol{\tau}} \tag{4.32}$$

where $\boldsymbol{\tau}_k$ is the row vector of the matrix $\boldsymbol{\tau}$. Presenting Equation 4.32 in this form provides less computational complexity than Equation 4.28 because dense matrix multiplication is avoided. The complexity of Equation 4.32 is discussed in detail in Results and Discussion section.

4.4.4 Enhanced Sliding Discrete Fourier Transform (eSDFT)

With the computational advantages of the MFT/SDFT algorithm, it can be applied to real-time signal processing when hardware limitations and computational complexity are essential considerations. However, using floating-point operations in the recursive form will affect the output estimation resulting in a cumulative error. The effect depends on the number of operations and the length of the iterations. Likewise, such errors can introduce in the system that uses the MFT/SDFT. This means that the Fourier coefficients will be far from the correct values if the MFT/SDFT is performed multiple times. To resolve this issue, the error can be controlled within a bounded range if there is a way to update the Fourier coefficients to keep the cumulative error as low as possible. This idea can be accomplished within multi-core operating systems, as it is possible to achieve this by working in parallel without adding excessive computational overhead. Hence, multi-core systems can be instructed to simultaneously work with the original MFT/SDFT algorithm and the proposed m-MFT formula. This concept is called “enhanced sliding discrete Fourier transform (eSDFT)”, as shown in Figure 4.4.

For calculating the spectrum coefficients at every incoming sample, the MFT/SDFT is performed. Simultaneously, another compute unit calculates the DFT

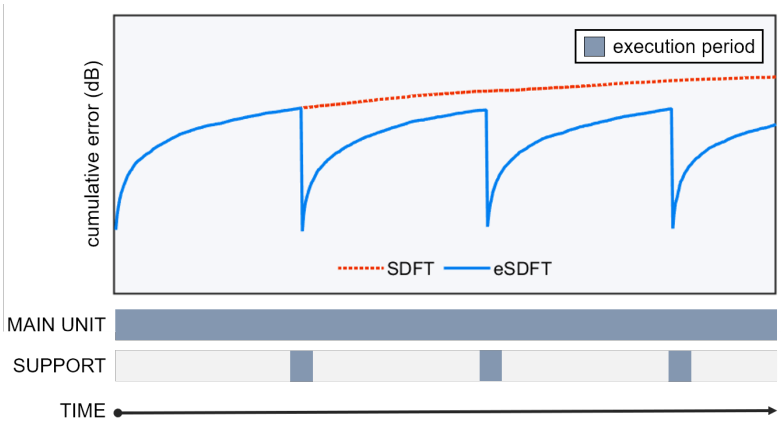

Figure 4.4. Demonstrating the cumulative error per time step when running the eSDFT algorithm compared to the original SDFT algorithm, which the infinitely increasing error occurred in the traditional SDFT method can be extremely reduced when the supporting unit provides the correct DFT coefficients to the main compute unit.

using non-recursive DFT algorithms such as the FFT method. When the correct coefficients are available, the coefficients are updated by using the m-MFT approach. This will control the cumulative error problem faced in the usual recursive method within an acceptable range. Figure 4.5 illustrates the cumulative error during execution on the main compute unit, in which the error bound is controlled within range when the supporting compute unit provides the correct DFT results to the system. Additionally, the error accumulation that occurred in the original SDFT algorithm is also represented, which tends to increase infinitely.

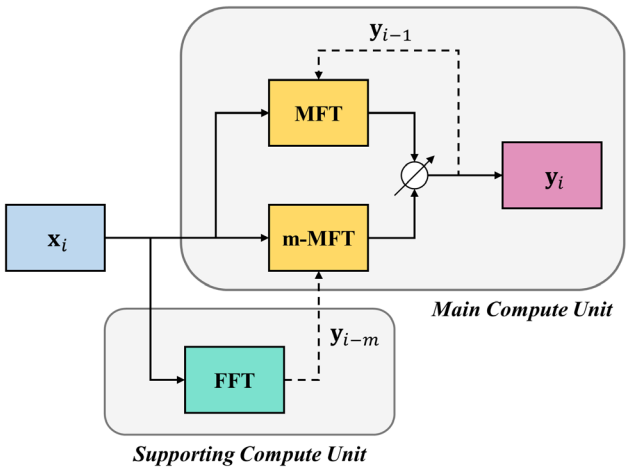

Figure 4.5. Explaining the concept of enhanced sliding discrete Fourier transform (eSDFT) that is concurrently operated on two compute units: the main compute unit performs fast calculation using the eSDFT algorithm, while the supporting unit runs using other DFT algorithms such as FFT for providing update periodically to minimize cumulative error.

The following pseudocode represents the computer coding of the general eSDFT algorithm, which combines both MFT and m-MFT approaches, assuming the twiddle factors of the MFT and m-MFT have been precomputed.

Algorithm 1 Algorithm for computing eSDFT

Input: time-series sequence $\mathbf{x}_i$, DFT sequences $\mathbf{y}_{i-1}$
Output: DFT sequence $\mathbf{y}_i$
Initialisation: twiddle factors W_{MFT} and W_{m-MFT}
LOOP Process
1: calculate $(x_i - x_{i-N})$
2: **if** supporting core is available **then**
3: calculate $\mathbf{y}_{i-m}$ using FFT algorithm
4: **else if** supporting core completes calculation **then**
5: calculate $\mathbf{y}_i$ using Equation 4.32 with $\mathbf{y}_{i-m}$
6: **else**
7: calculate $\mathbf{y}_i$ using Equation 4.17 with $\mathbf{y}_{i-1}$
8: **end if**

The source code demonstrating how the eSDFT algorithm works developed on MATLAB is provided in Appendix B.

4.4.5 Applying Windowing Function

In order to reduce the effect of aperiodicity when computing the Fourier coefficients of a real-world time-varying signal such as biological signals, applying windowing functions is one of the techniques that is commonly used to reduce the discontinuities at the edges of an input signal. In general, using eSDFT with well-known windowing functions such as Hanning, Hamming, and Blackman is possible. To demonstrate, the following equation explains how the windowing functions can be implemented into the eSDFT. Let $w(n)$ be the Hamming window that is to apply to N input samples that can be expressed by

$$w(n) = 0.54 - 0.46\cos\left(2\pi\frac{n}{N}\right). \tag{4.33}$$

Multiplying the window function in Equation 4.33 to the given sequence $\mathbf{x}_i$ provides the windowed input sequence as

$$\mathbf{x}_{wi} = 0.54\mathbf{x}_i - 0.23\left(e^{\frac{2j\pi n}{N}} + e^{-\frac{2j\pi n}{N}}\right)\mathbf{x}_i. \tag{4.34}$$

By taking the DFT of each part of the above equation, the Fourier coefficients of the windowed input can be obtained individually as

$$y_{wi,k} = 0.54y_{i,k} - 0.23\left(y_{i,k+1} + y_{i,k-1}\right). \quad (4.35)$$

Equation 4.35 expresses the recursive formula of the eSDFT using a windowing function that can be applied to the other available windows straightforwardly. Although applying the windows can reduce the effect of aperiodicity, it is a tradeoff consideration as the efficiency drops while additional computation is required. The performance of using the windowing function in the eSDFT is discussed in the next section.

4.4.6 Benchmarking and Performance Evaluation

As previously stated, the eSDFT can improve the cumulative error in a long-running system; in order to show its advantage, this study evaluated and compared the eSDFT to the state-of-the-art FFT library named FFTW (Frigo and Johnson, 2005) that uses the Cooley-Tukey FFT algorithm and is widely implemented in commercial numerical-analysis software such as MATLAB. Based on a literature search, the FFTW is the most optimised open-source library for computing the DFT using the FFT algorithm, which outperforms other current FFT libraries (Frigo and Johnson, 2005). In this study, the performance comparison was developed based on Microsoft Visual C++ (MSVC2017) and performed on the same computing machine (64-bit Window 10 OS, with 16-GB DDR3 and Intel Core i7 6th Gen), where the eSDFT used a 2-core CPU. In contrast, the FFTW used 1-core CPU to calculate Fourier coefficients. The window size (input length) was varied from 128, 256, 512, 1024, 2048, 4096, and 8192. Also, different input types (complex numbers and real numbers) and diverse floating-point operations (double-precision and single-precision) were considered. The input data was generated based on a normal distribution. In each run, output Fourier coefficients and runtime (execution time) on each system were recorded, with the total records of 10,000. The evaluation mainly focuses on the cumulative error, the speedup ratio, and the maximum number of shifts, as discussed in the next section.

4.5 Results and Discussion

4.5.1 Reduction of Accumulation of Round-Off Error

Due to the occurrence of the cumulative error in the original SDFT algorithm, the proposed eSDFT method was designed to minimise this issue. Furthermore, the error can be controlled within range by using the advantage of multiple compute units such as multi-core CPU to calculate the DFT coefficients and update the accurate results with the proposed m-MFT equation.

As stated in the previous section, the eSDFT was performed on a 2-core CPU, and the output coefficients were recorded. Figures 4.6 and 4.7 demonstrate the cumulative error when using the eSDFT method with double-precision floating-point and single-precision floating-point, respectively. Each iteration, the root-mean-square error (RMSE) was calculated by comparing the error between the eSDFT results $\mathbf{y}_i^{eSDFT}$ and its actual DFT results $\mathbf{y}_i^{DFT}$ (direct calculation). The RMSE (in dB) at time index i is obtained by

$$\mathrm{RMSE}_i = 20\log_{10}\left(\sqrt{\frac{1}{N}\sum_{k=0}^{N-1}\left(y_{i,k}^{DFT} - y_{i,k}^{SDFT}\right)^2}\right). \tag{4.36}$$

Additionally, the cumulative error of the original SDFT was determined in order to compare with the eSDFT. Figures 4.6 and 4.7 show that the RMSE of using single-precision is rising much higher than using double-precision. This is because the error propagation of the traditional SDFT method tends to increase gradually in double-precision, whereas it is rising significantly in single-precision.

The proposed eSDFT was implemented on two compute units, including the main compute and supporting compute units. The main unit was responsible for computing DFT coefficients using MFT and m-MFT algorithms, while the supporting unit was responsible for calculating the coefficients using the FFTW library. When the output coefficients from the supporting unit are ready, it passes to the main unit to update the recursive parameter using the m-sample shifted MFT formula. This scheme is

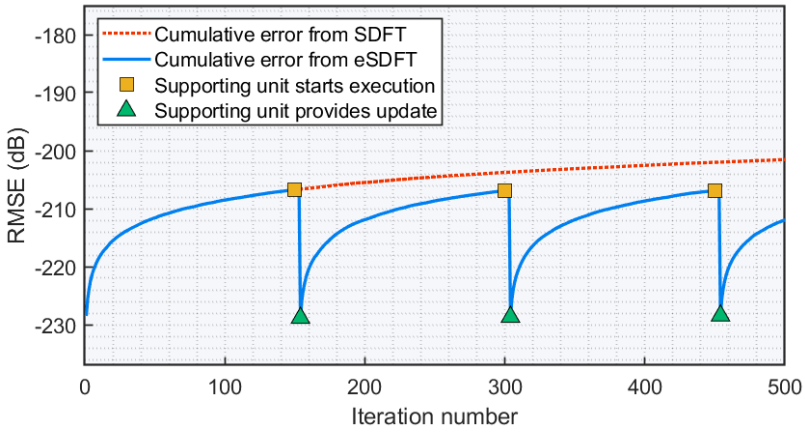

Figure 4.6. Demonstrating the cumulative error (point-by-point update) of the proposed eSDFT method compared to the original SDFT method using double-precision floating-point, which the non-recursive DFT algorithm was deployed on the supporting compute unit while the main compute unit performed the eSDFT.

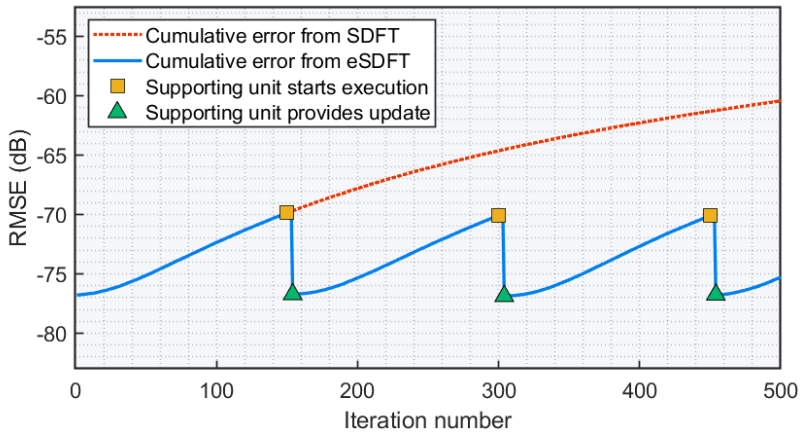

Figure 4.7. Demonstrating the cumulative error (point-by-point update) of the proposed eSDFT method compared to the original SDFT method using single-precision floating-point, which the non-recursive DFT algorithm was deployed on the supporting compute unit while the main compute unit performed the eSDFT.

performed periodically along with the real-time system. Running the system using the proposed method showed that the supporting compute unit does not need to be fast but should not exceed the maximum lag allowed (m). This topic will be discussed in-depth in the next part.

According to Figures 4.6 and 4.7, when starting calculation of the Fourier coefficients as the first sample arrived, the cumulative error will be increasing as it uses the previous coefficients to compute the results, which introduces the cumulative error as round-off, cancellation, and other traits of floating-point arithmetic can affect computations (Hu, 1992). However, the error decreases immediately when the

supporting compute unit starts execution and provides the correct coefficients to the system. Thus, by keeping the system running the eSDFT, the upper boundary of the cumulative error can be manageable, and the error depends on the different types of floating-point arithmetic.

4.5.2 Comparison of Complexity: multiple MFTs vs m-MFT

As mentioned in the previous study (Albrecht and Cumming, 1999), the original MFT operations are much less than the radix-2 FFT method if a small number of Fourier coefficients are considered a quarter or a half of the spectrum. In this study, the input was separated into two types, including real-valued inputs and complex-valued inputs. Equations 4.17 and 4.32 were selected to compare the computational complexity in terms of the number of real operations. Note that adding or subtracting two complex numbers requires two real additions or subtractions while multiplying two complex numbers is associated with four real multiplications and two real additions.

Referring to Equation 4.17, the term $x_i - x_{i-N}$ can be pre-calculated and used for the rest of the calculation. Similarly, most elements of $\boldsymbol{\beta}$ in Equation 4.23 that is used in Equation 4.32 can be achieved from the previous iteration of the MFT except for the last element ($\beta_{m-1} = x_i - x_{i-N}$); therefore, the computational cost of the previous elements can be excluded from the current computational cost. Subsequently, the calculation of the last element requires only two real subtractions for complex-valued input or one real subtraction for real-valued input. This number was included in the total number of operations at the end.

Table 4.1 shows a comparison of the total number of operations of different input and output types between the MFT method, the m-MFT method, and the radix-2 DIT FFT method used by the FFTW. The input types include real numbers and complex numbers, whereas the output types contain single-harmonic coefficient, half-spectrum coefficients, and full-spectrum coefficients. In several practical applications, the input data are purely real numbers, in which case the output DFT coefficients satisfy the Hermitian redundancy, resulting in only half of the spectrum calculation required. Note that the computational complexity was analysed based on an input length of N

Table 4.1. Computational complexity of different input/output types.

Input Type	Output Type	MFT (N-input)	FFT[a] (N-input)	m-MFT (m-shift, N-input)
Real	Single-harmonic	8	-	$4m$+4
	Half-spectrum	7(N/2)+1	(5N/2)($\log_2 N$)	(N/2)(4m+3)+1
Complex	Single-harmonic	10	-	8m+2
	Half-spectrum	8N+2	(5N)($\log_2 N$)	(N)(8m)+2

[a]From the Cooley-Tukey FFT (or radix-2 DIT FFT) algorithm which is used in the FTTW library (Mishra et al., 2014).

and the complexity of the FFT method for a single harmonic value is discarded from this study.

To compare the number of operations between the use in the original MFT algorithm for m samples (performing m-times MFT) versus using the m-MFT algorithm, the underlying assumption is that using the m-MFT approach offers fewer numbers of operations than the MFT at any samples shifted.

Let Q_{MFT} be the number of real operations of the MFT method, Q_{m-MFT} be the number of real operations of the m-MFT method, N_c be the input length, and m be the number of shifts. For the complex-valued input, it can be derived by

$$\begin{aligned} mQ_{MFT(complex)} &> Q_{m-MFT(complex)} \\ m(8N_c + 2) &> N_c(8m) + 2 \\ m &> 1. \end{aligned} \tag{4.37}$$

Correspondingly, for the real-valued input, half of the spectrum calculation is required, it can be derived as follows:

$$\begin{aligned} mQ_{MFT(real)} &> Q_{m-MFT(real)} \\ m\left(\frac{7N_c}{2} + 1\right) &> \frac{N_c}{2}(4m + 3) + 1 \\ m &> 1. \end{aligned} \tag{4.38}$$

Equations 4.37 and 4.38 are valid for all integers $m > 1$. Therefore, it can be concluded that the total number of operations when using the m-MFT method is always less than using the MFT method when some shifts in DFT samples are required, regardless of the input type.

4.5.3 Comparison of Complexity: multiple FFTs vs m-MFT

The previous section shows that using m-MFT leads to better performance than running multiple MFTs in any scenario. Presented here is an analytically computational comparison between the m-MFT and multiple FFTs. Using the radix-2 DIT FFT method, any N input samples require $\frac{N}{2}\log_2 N$ complex multiplication and $N\log_2 N$ complex addition for calculating full-spectrum coefficients. However, the radix-2 DIT FFT always requires that N must be a number to the power of 2. To compare the complexity between using the m-MFT method and the radix-2 DIT FFT method for m-sample shift when the input is complex numbers, the comparison of the arithmetic complexity can be determined by

$$\begin{aligned} mQ_{FFT(complex)} &> Q_{m-MFT(complex)} \\ m(5N_c\log_2 N_c) &> N_c(8m)+2 \\ m &> 0. \end{aligned} \tag{4.39}$$

Equation 4.39 proves that using the m-MFT method required less complexity than the FFT method for all integers $m > 0$.

For real input, the comparison can be obtained as follows:

$$\begin{aligned} mQ_{FFT(real)} &> Q_{m-MFT(real)} \\ m\left(\frac{5N_c}{2}\log_2 N_c\right) &> \frac{N_c}{2}(4m+3)+1 \\ m &> \frac{3N_c+2}{N_c(5\log_2 N_c-8)}. \end{aligned} \tag{4.40}$$

According to Equation 4.40, the equation exists for all integers $m > 0$ when the input length is more than eight samples. Hence, for the real-valued input, using the m-MFT method is more efficient than using the FFT method m times.

4.5.4 eSDFT Requirements and Its Limitations

As the complexity of using the MFT/SDFT method is always less than the FFT method in all scenarios (Jacobsen and Lyons, 2003; Albrecht and Cumming, 1999), there are no complexity issues if implementing only the MFT in the system. However, using the

eSDFT when including both the MFT method and the m-MFT method may introduce some limitations because of the number of shifts. In order to implement the eSDFT into real-world application, we need to consider the computational complexity in the case that the m-MFT method is less than a single computation of the FFT method. This means that another unit's DFT calculation should be complete within m samples to update the main unit using the m-MFT. Note that comparing with the FFT method requires the input length to the power of two.

For complex-valued input, the maximum number of the shifted samples can be determined by

$$\begin{aligned} Q_{FFT(complex)} &> Q_{m-MFT(complex)} \\ 5N_c \log_2 N_c &> N_c(8m) + 2 \\ m &< \frac{5N_c \log_2 N_c - 2}{8N_c}. \end{aligned} \tag{4.41}$$

In the same way, for real-valued input, the condition can be derived by

$$\begin{aligned} Q_{FFT(real)} &> Q_{m-MFT(real)} \\ \frac{5N_c}{2} \log_2 N_c &> \frac{N_c}{2}(4m + 3) + 1 \\ m &< \frac{5N_c \log_2 N_c - 3N_c - 2}{4N_c}. \end{aligned} \tag{4.42}$$

The maximum integer number of m that satisfies Equations 4.41 and 4.42 are visualised in Figures 4.10 and 4.11, respectively.

Implementing the eSDFT in real-time applications not only considers the computing performance as a key achievement but also should focus on memory usage. Table 4.2 provides the memory requirements of the eSDFT algorithm for different input types, i.e. complex numbers and real numbers. The memory needed is depended on the number of output coefficients. As stated in the previous study (Albrecht and Cumming, 1999), the MFT algorithm requires 8N arrays and N arrays when the whole spectrum and the half spectrum are computed, respectively. In the eSDFT system, a combination of the memory usage of the MFT formula and the m-MFT formula is

Table 4.2. Memory requirement (words) of different input types.

Parameter	Description	Size (Complex)	Size (Real)
W_{MFT}	Twiddle factors of Equation 4.17	$2N$	N
W_{m-MFT}	Twiddle factors of Equation 4.32	$2N$	N
$\boldsymbol{\tau}$	Modified Fourier matrix	$2N(m-1)$	$N(m-1)$
$\Delta\mathbf{x}_i$	Sample difference (MFT)	2	1
$\boldsymbol{\beta}$	Sample difference (m-MFT)	$2m$	m
$\mathbf{y}_i$	DFT coefficients at i	$2N$	N
$\mathbf{y}_{i-1}$	DFT coefficients at $i-1$	$2N$	N
$\mathbf{y}_{i-m}$	DFT coefficients at $i-m$	$2N$	N

required. The additional memory needed consist of the twiddle factors (W_{m-MFT}), the modified Fourier matrix ($\boldsymbol{\tau}$), the sample difference ($\boldsymbol{\beta}$), and the update output coefficients ($\boldsymbol{y}_{i-m}$). If the input is real numbers, the memory requirements can be reduced to the half of the complex-valued input.

4.5.5 Benchmarking Results

This section presents the results of computational experiments comparing the proposed eSDFT method (when m= 2, offering the maximum speedup ratio) and the FFTW method in terms of average runtime (in microsecond) and speedup ratio. The speedup ratio was calculated from the average execution time of running the FFTW algorithm divided by the average execution time of running the eSDFT algorithm. The comparison was performed using different types of floating-point operations, including double-precision and single-precision. Figures 4.8 and 4.9 illustrate the execution time and the speedup ratio when input is complex numbers and real numbers, respectively. The theoretical ratio (Q_{FFT}/Q_{m-MFT}) was determined to evaluate the CPU performance. Note that the input data of each execution was randomly generated based on a normal distribution.

Regarding the results, single-precision offers a higher speedup ratio when the window size is relatively large in both input types. Also, real-valued input shows reduced runtime and a higher speedup ratio than complex-valued input. Determining the coefficients of the real input provides the speedup ratio relatively close to the analytical ratio. However, a ratio that is lower than its theoretical value does not mean that the proposed method does not work; as in practice, the runtime depends on several factors, such as optimising the code, memory management, and computing platform.

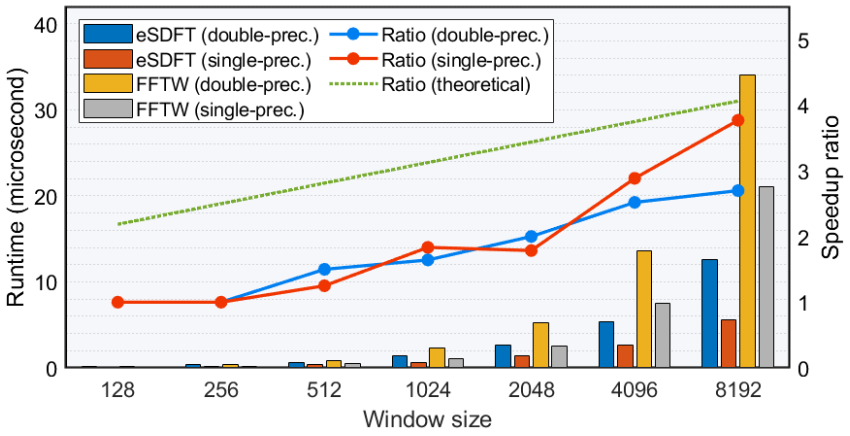

Figure 4.8. Showing the average execution time $\boldsymbol{t}$ (in microsecond) of running the eSDFT algorithm compared to the FFTW algorithm when an input is complex value. The input was analysed based on double-precision and single-precision floating-point arithmetic. Additionally, the speedup ratio ($\boldsymbol{t_{FFT}/t_{m-MFT}}$) was calculated comparing to the theoretical calculation ($\boldsymbol{Q_{FFT}/Q_{m-MFT}}$).

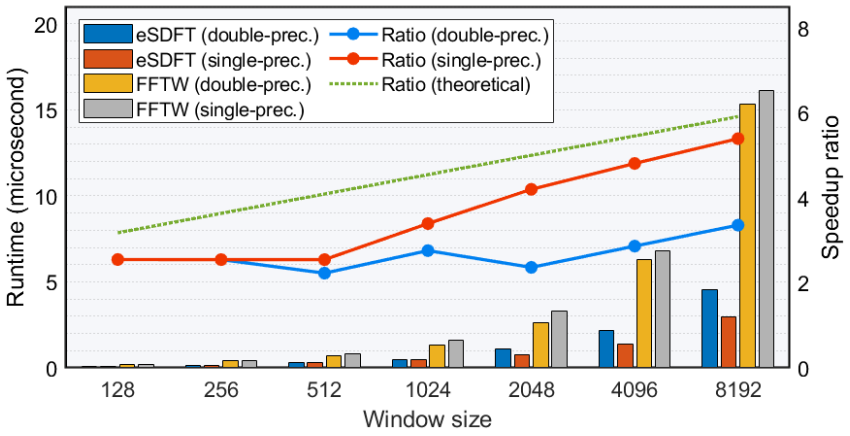

Figure 4.9. Showing the average execution time $\boldsymbol{t}$ (in microsecond) of running the eSDFT algorithm compared to the FFTW algorithm when an input is real value. The input was analysed based on double-precision and single-precision floating-point arithmetic. Additionally, the speedup ratio ($\boldsymbol{t_{FFT}/t_{m-MFT}}$) was calculated comparing to the theoretical calculation ($\boldsymbol{Q_{FFT}/Q_{m-MFT}}$).

In addition to the computational performance, the evaluation test of the eSDFT was conducted with a wide variety of m to find the maximum number of shifts compared to the FFTW method. Figures 4.10 and 4.11 demonstrate the maximum shift number when inputs are complex numbers and real numbers, respectively. The numerical evaluation was performed using double-precision and single-precision floating arithmetic. Moreover, the analytical calculation in Equations 4.41 and 4.42 is also shown in the figures. Regarding the results, it can be seen that applying the eSDFT

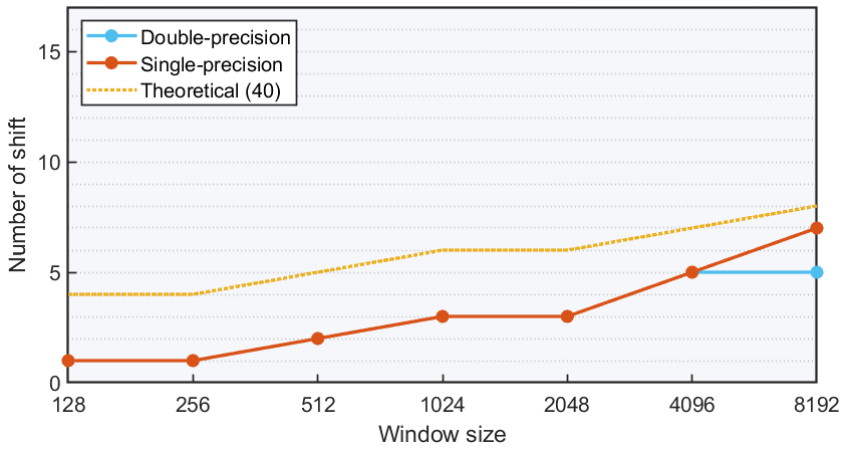

Figure 4.10. Plotting the maximum number of shift when an input is complex numbers, comparing between double-precision floating-point, single-precision floating-point, and the shift number that satisfies Equation 4.41. The window size is a number to the power of two. Note that m must be an integer.

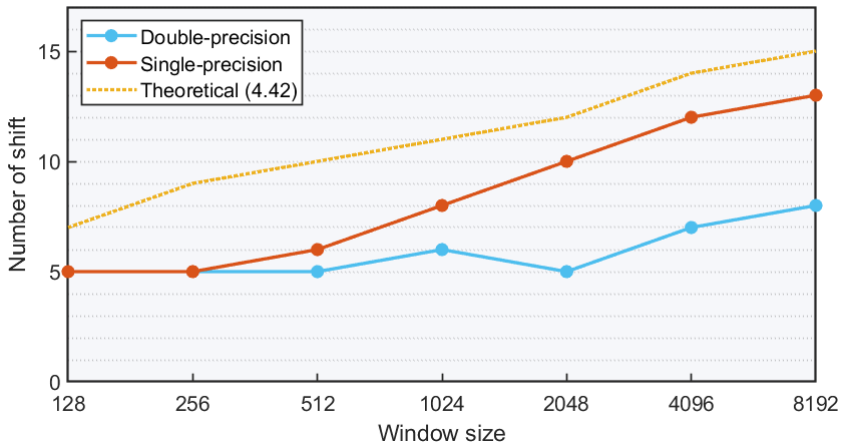

Figure 4.11. Plotting the maximum number of shift when an input is real numbers, comparing between double-precision floating-point, single-precision floating-point, and the shift number that satisfies Equation 4.42. The window size is a number to the power of two. Note that m must be an integer.

method to real-valued input has a less restricted number of shifted samples than applying to the complex-valued input at the same window size. This is because of fewer complex multiplications in the eSDFT.

On the other hand, using single-precision arithmetic provides the maximum number closer to its theoretical value than using double-precision, especially for real numbers. Additionally, allowing for a wider window size in real-valued input can increase the number of shifts extremely. Therefore, if fewer coefficients are required, the complexity of the eSDFT is much more efficient than the FFT method.

4.5.6 eSDFT Performance When Applying Windowing Function

Applying the windowing function when calculating the DFT coefficients can decrease the discontinuities at the edges of an input signal. However, in the eSDFT system, it is allowed to add additional complexity. Equation 4.35 expresses the eSDFT with Hamming window when the window function can be performed following the eSDFT method. This section shows the use of the windowing function in Equation 4.35 with different input types.

To determine the maximum number of shifted samples allowed when comparing the radix-2 DIT FFT with the eSDFT, Equations 4.41 and 4.42 can be applied by adding the additional complexity of applying the windowing function. In general, the FFT method requires N operation for real-valued input and $2N$ operation for complex-valued input to multiply before the DFT calculation. On the other hand, when calculating full-spectrum coefficients for complex-valued input, Equation 4.35 requires $8N$ additional operation; however, it requires only $4N$ additional operations for half of the spectrum of real-valued input, which is a significant advantage when using the eSDFT. As a result, the equation expressing the condition of using eSDFT with the windowing comparing to the FFT can be obtained as follows:

For complex input, the maximum shift can be obtained by

$$\begin{aligned} Q_{FFT(cx)} + Q_{windowing(cx)} &> Q_{m-MFT(cx)} + Q_{windowing(cx)} \\ N_c(5\log_2 N_c + 2) &> 8N_c(m+1) + 2 \\ m &< \frac{5N_c \log_2 N_c - 6N_c - 2}{8N_c}. \end{aligned} \tag{4.43}$$

Alternatively, for real input, the condition can be attained by

$$\begin{aligned} Q_{FFT(re)} + Q_{windowing(re)} &> Q_{m-MFT(re)} + Q_{windowing(re)} \\ N_c\left(\frac{5}{2}\log_2 N_c + 1\right) &> \frac{N_c}{2}(4m + 11) + 1 \\ m &< \frac{5N_c \log_2 N_c - 9N_c - 2}{4N_c}. \end{aligned} \tag{4.44}$$

Figure 4.12 illustrates the analytical results attained from Equations 4.43 and 4.44 compared with Equations 4.41 and 4.42. Regarding the results, applying the

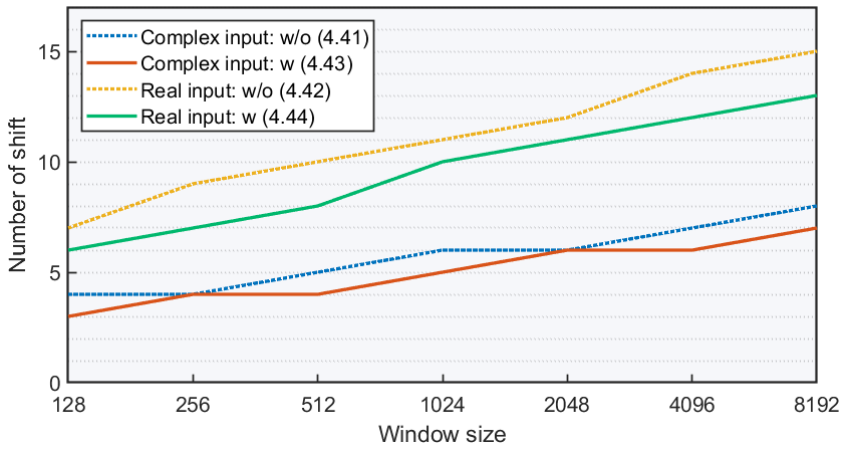

Figure 4.12. Plotting the maximum integer number of m on different input types that satisfies the m-MFT method compared to the radix-2 DIT FFT method, when determining Fourier coefficients with (w) and without (w/o) applying windowing function. The window size is a number to the power of two.

windowing function to the eSDFT can reduce the maximum shift numbers compared to the non-window eSDFT. The numbers further drop when the input is real numbers, whereas the complex-valued input shows a lesser decrease. Note that using different windowing functions in the eSDFT provides different computational performance. Thus, applying a windowing function in the eSDFT system needs to be considered carefully as a tradeoff with its improvement of output coefficients.

4.5.7 Speedup Ratio Versus Additional Overhead

In practice, applying the eSDFT to the system always comes with additional computational overhead as it uses two cores rather than one core. Hence, the system may consume more CPU/GPU resources because the supporting unit needs to compute the update coefficients. The additional overhead is defined as an extra numerical operation that is used to compute the Fourier coefficients in the eSDFT algorithm comparing to the FFT method, which can be calculated from

$$\text{Additional overhead (\%)} = \frac{Q_{eSDFT}}{Q_{FFT}} \times 100 \tag{4.45}$$

where Q_{eSDFT} can be attained from the combination of Q_{m-MFT} and Q_{FFT} as they represent the total number of numerical computations in the eSDFT system. Therefore,

a tradeoff between the speedup ratio achieved and the percentage of the additional overhead has to be considered. In this study, the analytical calculation of the additional overhead and the speedup ratio was obtained from the computational complexity of the eSDFT algorithm and the radix-2 DIT FFT algorithm. The speedup ratio is directly calculated from the ratio of the FFT complexity divided by the m-MFT complexity (Q_{FFT}/Q_{m-MFT}) while the percentage of the additional overhead was calculated using Equation 4.45. Figures 4.13 and 4.14 represents the comparison when input is complex numbers and real numbers, respectively. The speedup ratio and the additional overhead were determined with a different number of shifts and window size. Note that the maximum speedup ratio exists when $m = 2$.

With regard to the results, the speedup ratio tends to rise infinitely with an increase of the window size for both real-valued and complex-valued input. For example, when $m = 2$ and the window size of 128, the speedup ratio is approximately 2 times and 3 times for complex numbers and real numbers, respectively. However, when increasing the window size to 8192, the ratio can rise to 4 times for complex-valued input and nearly 6 times for real-valued input.

In addition to the speedup ratio, the additional overhead tends to decrease with an increase in the window size for all input types. For instance, the percentage of additional overhead at $m = 2$ and the window size of 128 shows that if we increase the speedup ratio to 2 times for complex input and 3 times for real input, approximately 46% of additional overhead for complex input and 31% of additional overhead for real input are needed, respectively. While at $m = 2$ and the window size of 8192, the percentages can reduce to 25% in complex-valued input and 17% in real-valued input. These additional overheads show that the eSDFT algorithm requires few extra operations to gain speedup. According to Figures 4.13 and 4.14, the plots confirm that at $m = 2$ the eSDFT provides the maximum speedup ratio with the minimum additional overhead, compared to other shifts.

Running the eSDFT algorithm using complex input requires more overhead than running with real input in all scenarios. For instance, when $m = 5$ with window size of 128, over a hundred per cent of additional overhead is needed when the input is complex numbers, whereas the real-valued input requires approximately 60%. On the

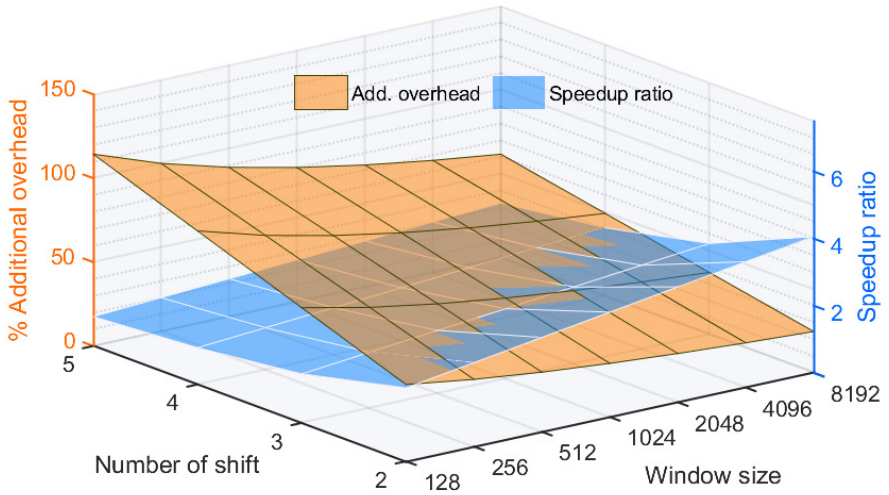

Figure 4.13. Illustrating the percentage of additionally computational overhead and the speedup ratio achieved when an input is complex numbers, comparing between the radix-2 DIT FFT method and the eSDFT method.

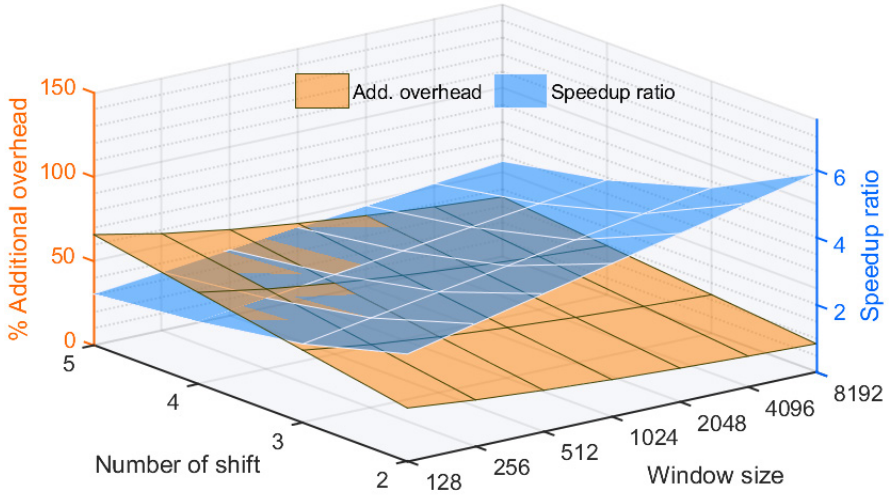

Figure 4.14. Illustrating the percentage of additionally computational overhead and the speedup ratio achieved when an input is real numbers, comparing between the radix-2 DIT FFT method and the eSDFT method.

other hand, the real input uses lower computational resources and offers a higher speedup ratio than the FFT algorithm. Consequently, in most real-world applications that use real numbers, the proposed eSDFT provides further efficiency in terms of the overhead and the speedup.

Compared with the use of parallel architecture applied to the traditional FFT algorithm, assuming the FFT can be parallelised for 2-core operation, the speedup can be doubled and the eSDFT can show better utilisation of computational resources and

provide more speedup ratio when the input length is relatively large. For example, using real input at the window size of 512 and $m = 2$, the eSDFT can gain the speedup ratio up to 4 times. However, in order to achieve the same ratio in the parallel FFT architecture, the system may need more than two cores to run, which can be an extra cost for the system. In contrast, the eSDFT consumes only 25% more computational resources. With this advantage, the eSDFT can be implemented in any system with limited computational resources, such as a small microprocessor.

4.6 Chapter Summary

In multichannel EEG analysis, there many changeable factors including the number of recording locations, the recording frequencies, and the number of feature extraction parameters. These factors significantly impact the real-time system performance and may result in signal processing system instability. To address this issue, this work developed a new algorithm capable of operating in a real-time environment comprising a method to compute the sliding-window DFT for real-time signal processing using a multi-core approach called “enhanced sliding discrete Fourier transform”. The eSDFT method is most suitable for processing time-varying physiological signals when there is a need for real-time or short-time processing performance where results may be fed-back to the user or used to command extrinsic devices.

The proposed method was developed based on reformulated momentary matrix transformation for m-sample shift. The computational complexity analysis showed that the hybrid approach provided much better performance than the conventional FFT method for both real-valued and complex-valued input, depending on the number of the input length. Moreover, the eSDFT always provides shorter runtime when half of the spectrum or specific numbers of coefficients are considered. For example, using real-valued input can gain more speedup and require less overhead than complex numbers. Besides, applying a windowing function to the eSDFT method to reduce the effect of aperiodicity is possible with additional computation. Regardless, future research could continue to explore the iteration bound, including the maximum number of iteration numbers for the eSDFT algorithm under the given precision constraint. In summary, with the advantages of the proposed method, it can be applied to large-scale,

real-time signal processing that utilises multiple compute units for distributed computing. The highlights and future work will be recapped in Chapter 7.

The next chapter will examine the possibility of applying the parallel computing idea to a different feature extraction method, the discrete wavelet transform. The computational performance will be evaluated extensively using an accelerating device. Additionally, modifications to the algorithm for interacting with the device will be discussed in detail.

4.7 References

Albrecht, S. & Cumming, I. 1999. Application of momentary Fourier transform to SAR processing. *IEE Proceedings-Radar, Sonar and Navigation,* 146**,** 285-297.

Ayinala, M., Lao, Y. & Parhi, K. K. 2013. An in-place FFT architecture for real-valued signals. *IEEE Transactions on Circuits and Systems II: Express Briefs,* 60**,** 652-656.

Bi, G. & Chen, Y. Q. 1998. Fast DFT algorithms for length N= q* 2/sup m. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* 45**,** 685-690.

Bui, N. T., Phan, D. T., Nguyen, T. P., Hoang, G., Choi, J., Bui, Q. C. & Oh, J. 2020. Real-Time Filtering and ECG Signal Processing Based on Dual-Core Digital Signal Controller System. *IEEE Sensors Journal,* 20**,** 6492-6503.

Chen, W., Kehtarnavaz, N. & Spencer, T. 1993. An efficient recursive algorithm for time-varying Fourier transform. *IEEE transactions on signal processing,* 41**,** 2488-2490.

Cheng, W., Lu, J., Zhu, X., Hong, J., Liu, X., Li, M. & Li, P. 2019. Dilated Residual Learning With Skip Connections for Real-Time Denoising of Laser Speckle Imaging of Blood Flow in a Log-Transformed Domain. *IEEE transactions on medical imaging,* 39**,** 1582-1593.

Choe, J. W., Nikoozadeh, A., Oralkan, Ö. & Khuri-Yakub, B. T. 2013. GPU-based real-time volumetric ultrasound image reconstruction for a ring array. *IEEE transactions on medical imaging,* 32**,** 1258-1264.

Cochran, W. T., Cooley, J. W., Favin, D. L., Helms, H. D., Kaenel, R. A., Lang, W. W., Maling, G. C., Nelson, D. E., Rader, C. M. & Welch, P. D. 1967. What is the fast Fourier transform? *Proceedings of the IEEE,* 55**,** 1664-1674.

Duda, K. 2010. Accurate, guaranteed stable, sliding discrete Fourier transform [DSP tips & tricks]. *IEEE Signal Processing Magazine,* 27**,** 124-127.

Dudas, J. 1986. *The momentary Fourier transform.* Ph.D. dissertation, Technical University of Budapest, Hungary.

Exposito, A. G. & Macfas, J. 2000. Fast non-recursive computation of individual running harmonics. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* 47**,** 779-782.

Frigo, M. & Johnson, S. G. 2005. The design and implementation of FFTW3. *Proceedings of the IEEE,* 93**,** 216-231.

Garcia-Rial, F., Ubeda-Medina, L. & Grajal, J. 2017. Real-time GPU-based image processing for a 3-D THz radar. *IEEE Transactions on Parallel and Distributed Systems,* 28**,** 2953-2964.

Gudovskiy, D. A. & Chu, L. 2017. An accurate and stable sliding DFT computed by a modified CIC filter [tips & tricks]. *IEEE Signal Processing Magazine,* 34**,** 89-93.

Guo, Y., Mao, Y., Park, D. S. & Lee, M. H. 2011. Fast DFT matrices transform based on generalised prime factor algorithm. *Journal of Communications and Networks,* 13**,** 449-455.

Harvie, A. J., Phillips, T. & Demello, J. C. 2020. A high-resolution polarimeter formed from inexpensive optical parts. *Scientific reports,* 10**,** 1-12.

Hosny, K. M., Darwish, M. M., Li, K. & Salah, A. 2018. Parallel multi-core CPU and GPU for fast and robust medical image watermarking. *IEEE Access,* 6**,** 77212-77225.

Hu, Y. H. 1992. The quantisation effects of the CORDIC algorithm. *IEEE Transactions on signal processing,* 40**,** 834-844.

Jacobsen, E. & Lyons, R. 2003. The sliding DFT. *IEEE Signal Processing Magazine,* 20**,** 74-80.

Jacobsen, E. & Lyons, R. 2004. An update to the sliding DFT. *IEEE Signal Processing Magazine,* 21**,** 110-111.

Johnson, S. G. & Frigo, M. 2006. A modified split-radix FFT with fewer arithmetic operations. *IEEE Transactions on Signal Processing,* 55**,** 111-119.

Kolba, D. & Parks, T. 1977. A prime factor FFT algorithm using high-speed convolution. *IEEE Transactions on Acoustics, Speech, and Signal Processing,* 25**,** 281-294.

Kulshreshtha, T. & Dhar, A. S. 2018. CORDIC-based high throughput sliding DFT architecture with reduced error-accumulation. *Circuits, Systems, and Signal Processing,* 37**,** 5101-5126.

Li, K., Zheng, W. & Li, K. 2014. A Fast Algorithm With Less Operations for Length-$ N= q\times 2^{m} $ DFTs. *IEEE Transactions on Signal Processing,* 63**,** 673-683.

Liang, Y., Lu, L., Xiao, Q. & Yan, S. 2019. Evaluating fast algorithms for convolutional neural networks on FPGAs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* 39**,** 857-870.

Liavas, A. P., Kostoulas, G., Lourakis, G., Huang, K. & Sidiropoulos, N. D. 2017. Nesterov-based alternating optimisation for nonnegative tensor factorisation: Algorithm and parallel implementation. *IEEE Transactions on Signal Processing,* 66**,** 944-953.

Ma, Z.-G., Yin, X.-B. & Yu, F. 2015. A novel memory-based FFT architecture for real-valued signals based on a radix-2 decimation-in-frequency algorithm. *IEEE Transactions on circuits and systems II: Express Briefs,* 62**,** 876-880.

Macias, J. R. & Exposito, A. G. 1998. Efficient moving-window DFT algorithms. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* 45**,** 256-260.

Mishra, S., Das, D., Kumar, R. & Sumathi, P. 2014. A power-line interference canceler based on sliding DFT phase locking scheme for ECG signals. *IEEE transactions on Instrumentation and Measurement,* 64**,** 132-142.

Murakami, H. 1994. Real-valued decimation-in-time and decimation-in-frequency algorithms. *IEEE Transactions on circuits and systems II: Analog and Digital Signal Processing,* 41**,** 808-816.

Nussbaumer, H. J. 1981. The fast Fourier transform. *Fast Fourier Transform and Convolution Algorithms.* Springer.

Park, C.-S. 2017. Guaranteed-stable sliding DFT algorithm with minimal computational requirements. *IEEE Transactions on Signal Processing,* 65**,** 5281-5288.

Samiee, K., Kovacs, P. & Gabbouj, M. 2014. Epileptic seizure classification of EEG time-series using rational discrete short-time Fourier transform. *IEEE transactions on Biomedical Engineering,* 62**,** 541-552.

Serbes, A. 2018. Fast and efficient sinusoidal frequency estimation by using the DFT coefficients. *IEEE Transactions on Communications,* 67**,** 2333-2342.

Silverman, H. 1977. An introduction to programming the Winograd Fourier transform algorithm (WFTA). *IEEE Transactions on Acoustics, Speech, and Signal Processing,* 25**,** 152-165.

Sorensen, H. V., Jones, D., Heideman, M. & Burrus, C. 1987. Real-valued fast Fourier transform algorithms. *IEEE Transactions on acoustics, speech, and signal processing,* 35**,** 849-863.

Varkonyi-Koczy, A. R. 1995. A recursive fast Fourier transformation algorithm. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* 42**,** 614-616.

Vidyaratne, L. S. & Iftekharuddin, K. M. 2017. Real-time epileptic seizure detection using EEG. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 25**,** 2146-2156.

Wang, S., Patel, V. M. & Petropulu, A. 2018. Multidimensional sparse Fourier transform based on the Fourier projection-slice theorem. *IEEE Transactions on Signal Processing,* 67**,** 54-69.

Yu, S. & Swartzlander, E. E. 2001. A pipelined architecture for the multidimensional DFT. *IEEE transactions on signal processing,* 49**,** 2096-2102.

Yu, T., Akhmadeev, K., Le Carpentier, E., Aoustin, Y. & Farina, D. 2019. On-line recursive decomposition of intramuscular EMG signals using GPU-implemented Bayesian filtering. *IEEE Transactions on Biomedical Engineering,* 67**,** 1806-1818.

CHAPTER 5

MODIFIED DWT ALGORITHM

Many BCI research studies have used the Fourier transform because of its simplicity and versatility in processing bio-signals in the frequency domain. However, owing to the high temporal resolution of EEG signals, a straightforward Fourier transform cannot take advantage of this feature, frequently resulting in the loss of time information. As a result, this investigation explored another feature extraction technique capable of providing both time and frequency domains, namely the discrete wavelet transform. The previous chapter demonstrated the enhanced version of the FFT algorithm using recursive modification, which showed promising results. This chapter presents modifications of the widely used discrete wavelet transform (DWT) based on parallel computation. The DWT algorithm is modified in a different perspective, which can support operating with modern hardware acceleration. Furthermore, an alternate matrix-vector form of the DWT, which is the primary basis for the parallel computing scheme, is also presented. Moreover, signal extension modes applied to the DWT modification is also proposed to reduce the border distortion effect. Finally, the computational performance of the proposed method is evaluated via a pseudo-real-time system, in which the results and findings in terms of computation time and speed up ratio are discussed thoroughly.

5.1 Introduction

As discussed earlier in Section 2.6.1, real-time signal processing requires fast speed, system reliability and stability. Therefore, the processing system within a BCI has to be optimised to achieve the best computing performance, such as minimising the processing pipeline's latency and maximising the classification accuracy as explained

in Section 3.4.2. However, a major issue that arises in real-time BCIs is the latency introduced by signal acquisition hardware, data transmission between recording devices and processors, application types, and the necessary signal processing algorithm computations that underpin performance. As signal processing is one of the significant contributors to the total latency time, it is important to simplify or optimise the computational complexity of the signal-processing pipeline.

Regarding the history and the advantages of the OpenCL platform, these have been discussed in Section 2.6.3. In order to rapidly develop an OpenCL-based system, there are many libraries available for mathematical functions. These can help a user reduce development time as building a new function or operation in OpenCL frequently involves low-level management. However, this can be problematic as achieving the best computing performance can be compromised if one of these managements is missing. Currently, clFFT is a widely used library (Vázquez et al., 2019). This library was explicitly developed for reducing the computation time for Fourier analysis. The clFFT claims that the speedup performance, when implemented in OpenCL, outperformed the regular FFT operation. Another helpful library is clBAS (Nugteren, 2018), which performs common linear algebra operations (e.g., vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication. This useful library helps in solving many computational problems and computationally demanding big data scenarios. In addition to solving matrix equations, a clMAGMA library (Bernabé et al., 2017) has been developed to alleviate complex matrix operations; however, the library does not provide any function related to wavelet computation.

Presently, there is widespread time-frequency analysis in both offline and online BCI applications (Kinoshita et al., 2020; Tian et al., 2019; Durongbhan et al., 2019). One well-known technique is the Discrete Wavelet Transform (DWT), used for feature extraction within clinical and BCI applications, including epilepsy detection (Li et al., 2017) and movement intention detection (Ma et al., 2020). There have been several practical techniques to perform the DWT, for example, the filter bank method (Strang and Nguyen, 1996), lifting-based schemes (Sweldens, 1996) and CORDIC-based

architectures (Simon et al., 1996). The straightforward filter bank method has gained popularity within commercial software because of its ease of implementation.

Regarding the lifting-based technique, this leads to an integer wavelet transform, which is faster than the original floating-point DWT. Hence, it is well suited for lossless coding and compression. CORDIC is another appropriate method for Very-Large-Scale Integration (VLSI) implementation, as only addition and shifts are involved with the least possible adders required. Interestingly, based on our literature search, no study to date has examined DWT methods based on a parallel approach despite the clear potential to use the power of heterogeneous computing to improve runtime. This subject will be discussed in detail in Section 5.3 of this study.

This study investigates the power of heterogeneous computing in performing multi-channel filter bank DWT for real-time applications using parallel computing techniques. A primary aim of this work was to propose a general wavelet transformation as matrix-vector operations based on the parallel computing approach. In this context, a Central Processing Unit (CPU) and multiple GPU devices were deployed as a host and a device, respectively. The proposed system offers 32-channel real-time parallel computing in which each channel executes the DWT concurrently. To evaluate the system performance, the proposed system was examined under a pseudo-real-time condition, in which the input signal was repetitively reproduced as analogues from an archived data set. This testing data set was generated with different resolutions of sampling frequency. In comparison with the sequential processing technique, the runtime of the parallel computing scheme was compared to the runtime of the sequential scheme.

5.2 Background

5.2.1 OpenCL in Action

As mentioned previously, the OpenCL platform is widely used in HPC where the heterogeneous computing concept is used for execution acceleration. In this study, the design of the system architecture of the parallel computing is similar to the standard OpenCL architecture, as previously illustrated in Figure 2.12 in Chapter 2.

5.2.2 Discrete Wavelet Transform

Discrete Wavelet Transform (DWT) is a recognised technique for the time-frequency analysis of nonstationary biosignals (e.g. the Electroencephalography (EEG) or Elecotrocorticography (ECoG)). Employing a large window to low frequencies and a short window for higher frequencies in the time domain results in excellent time-frequency resolution. The DWT is a procedure for decomposing a discrete-time signal $x[n]$ into a set of signals called wavelet coefficients by scaling and shifting the mother wavelet function. Accordingly, the appropriate number of the decomposition levels (L_m) can be preferred from the initial input length; the principal frequency components restrict the maximum level in the given signal (Wu et al., 2000). In general, the DWT is implemented as a filter bank by splitting a signal into two chunks. For every single level of the decomposition, the signal $x[n]$ is filtered by both the high-pass filter $h[n]$ and the low-pass filter $g[n]$, followed by the process of downsampling by 2. After the decomposition of each level, the approximation coefficients (A_L) and the detailed coefficients (D_L) are denoted as follows:

$$D_L[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[2n-k] \tag{5.1}$$

$$A_L[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot g[2n-k]. \tag{5.2}$$

Equations 5.1 and 5.2 are a general filter-bank-based form of determining the DWT coefficients. The approximation coefficients (A_L) is set as $x[n]$ in the decomposition of the next level (L+1). The procedure repeats as long as the defined number of decomposition levels (L_m) is attained, as shown in Figure 5.1.

5.3 Related Works

Regarding the implementation of DWT using a parallel processing concept, Wong and Heng proposed a fast 2-D DWT that ran on consumer-level graphics hardware using a parallel processing technique (Wong et al., 2007). Applied to image processing and

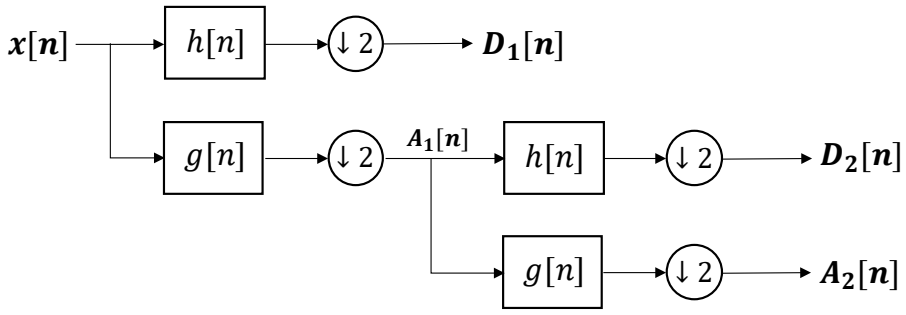

Figure 5.1. Explaining the cascaded DWT structure for decomposition level of 2.

encoding problems using Open Graphics Library (OpenGL) extensions on a GPU (which is Single Instruction Multiple Data (SIMD) processor), computing the convolution-based DWT outperformed a standard PC. Another study of parallel processing on DWT using OpenGL and C for Graphics (Cg) programming language was proposed by Tenllado et al. (Tenllado et al., 2008). They compared the DWT performance between the Filter Bank Scheme (FBS) and Lifting Scheme (LS), reporting that the speedup of the FBS was better than the LS. However, they mentioned that the data transfer between the CPU and the GPU was a significant bottleneck.

Later, Sharma and Vydyanathan (Sharma and Vydyanathan, 2010) introduced the first usage of the OpenCL platform for DWT calculation using a lifting-based approach for JPEG200 image compression with the intention to show improved portability and the capabilities of using the OpenCL for cross-platform programming between NVIDIA's and AMD's hardware. Interestingly, although excellent computing performance was achieved, comparisons with native GPU programming like Compute Unified Device Architecture (CUDA, NVIDIA) showed that the OpenCL kernel performance was slower than the CUDA kernel. According to the different approaches of computing DWT based on a parallel scheme, many studies focused on optimisation (van der Laan et al., 2010; Matela, 2009), reduction of memory usage (Ikuzawa et al., 2016), and a hybrid approach (Quan and Jeong, 2016) that the lifting-based scheme was preferred. For example, Tenllado et al. (Tenllado et al., 2004) revealed 10-20% processing improvement in the performance of the lifting-based DWT over convolution methods. However, another study concluded that the lifting scheme required more rendering steps because of increased data dependencies (Tenllado et al., 2008). With shorter wavelets, the convolution approach achieved a gain of 50-100%

compared to the lifting scheme, whilst for larger wavelets, the lifting method was 10-20% faster.

While the development of the lifting method is more popular than the filter bank scheme, with few numbers of input, the speed-up improvement is still not great (Tenllado et al., 2008). Recently, Hence Zhao (Zhao, 2015) implemented the real-time convolutional 3-D wavelet transform on mobile GPU computing using a shared-memory technique.

Concerning the type of computations and memory access, all earlier approaches are restricted by the necessity to map the algorithms to graphics operations. On the other hand, with the advancement of GPU programming and intermediate fast-shared memories, the user can achieve in-place transforms in a single pass. In addition to the latest OpenCL version (OpenCL 2.0) (Kaeli et al., 2015), virtual shared memory (VSM) also allows users to write code with extensive use of pointer-linked data structures. This enables sharing the programming memory between the host (CPU) and a device-side (GPU). This new feature has been developed to overcome the data-transferring problem, which assists developers in reducing memory management problems.

Although there are many heterogeneous programming platforms available for parallel computing, the trends of GPU-based DWT research primarily focused on specified CUDA technology due to its ease and continued vendor support, which encourages many developers to use it. Conversely, due to limited research interest in OpenCL, a lack of example and a practical tutorial on its use exists with most applications having been developed for image (Wang et al., 2013; Simmross-Wattenberg et al., 2018) and large-scale data processing (Suda et al., 2016; Sun et al., 2019). Taking a signal processing perspective, a significant challenge is to reduce the computational complexity and the latency in the processing pipeline, especially for the use of DWT in real time.

EEG signals are often acquired in multi-channel configurations. Using the DWT for real-time time-frequency analysis of this kind of data requires intensive computation. In clinical research, many studies employ real-time DWT to detect physiological changes such as epileptic seizures (Chen et al., 2017; Vidyaratne and

Iftekharuddin, 2017) and emotion recognition (Mohammadi et al., 2017). Those studies mention the computational cost obtained from computing the DWT when used in multi-channel applications. Consistently, this difficulty becomes an obstacle for a high number of the decomposition level of the DWT (Alickovic et al., 2018; Amin et al., 2015). In light of the essentials mentioned above, the DWT plays a vital role in signal processing for real-time applications.

5.4 Materials and Methodology

5.4.1 A Real-Time Multi-Channel BCI System for Parallel DWT

In this chapter, the real-time heterogeneous technique for determining the discrete wavelet transform is proposed. In summary, multi-channel EEG data acquired from the signal acquisition module are simultaneously processed in real time via a modified DWT algorithm executed using the parallel compute units on the hardware acceleration (GPU). Hence, the output DWT coefficients attained from each compute unit can be further used in the feature extraction step. The overall system mentioned above is illustrated in Figure 3.6 in Chapter 3.

5.4.2 Border Distortion Effect Reduction in DWT

In most cases, when calculating the wavelet coefficients of each decomposition level, the input signals are on compact intervals. These intervals have finite support and are usually defined over a rectangle window. To perform the DWT outside the correlation window, the signal needs to be extended at its borders. This often leads to distortions (Montanari et al., 2015), resulting in different coefficient values that do not reflect the original signal appearing from the DWT. In general, signal extension mode techniques illustrated in Figure 5.2 are used to handle this problem:

1) Zero-padding (ZPD): by assuming that the value outside the input window is zero. The disadvantage of this technique is that discontinuities are artificially generated at the boundary.

2) Symmetrisation (SYM): by assuming that the border can be extended by symmetric boundary value duplication. The disadvantage is that discontinuities of the

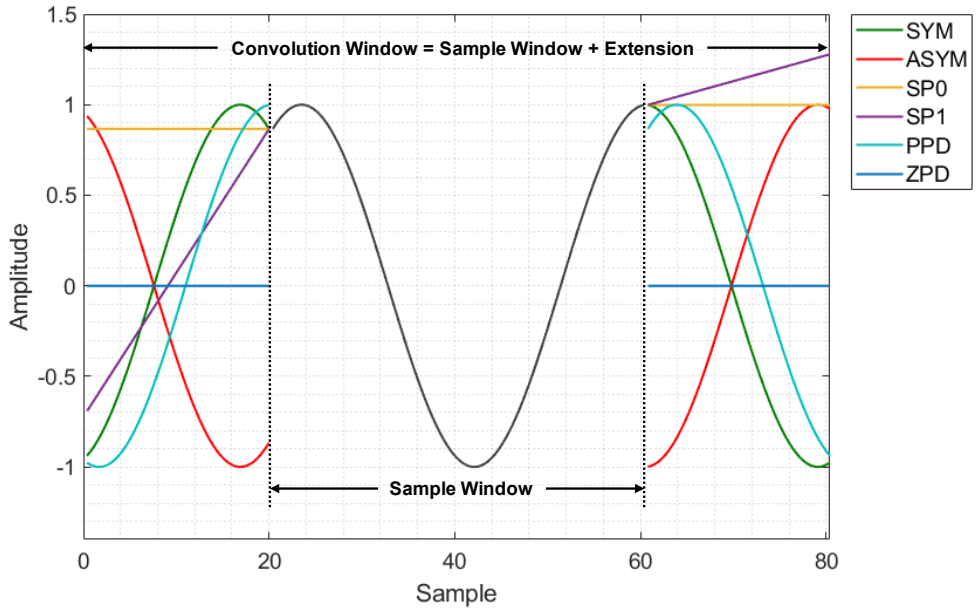

Figure 5.2. Showing the common techniques used for border interpretation when performing the DWT by shifting the pattern window over the signal vector. Hence an extra sample is added to the given sample window.

first derivative are artificially created at the border. However, this technique works well for images.

3) Antisymmetric Padding (ASYM): by assuming that the border can be extended by antisymmetric boundary value duplication.

4) Smooth Padding of Order 1 (SP1): by assuming that the border can be extended by a simple first-order derivative extrapolation, which is padding using a linear extension fit to the first two and last two values. This technique works perfectly for smooth functions.

5) Smooth Padding of Order 0 (SP0): by assuming that the border can be extended by a simple constant extrapolation that is the duplication of the first value on the left and the last value on the right.

6) Periodic Padding (PPD): by assuming that the border can be extended by periodic extension. This technique has the disadvantage of artificially creating discontinuities at the border.

7) Wavelet Periodisation (PER): this extension works in the same procedure as by the PPD. When the signals have odd samples, an additional sample is appended at the right side, which the sample added is equal to the last sample value.

In this study, the signal extension modes mentioned above were applied to the DWT procedure, computed by the accelerating device (GPU). The computing performance was compared in terms of execution time.

5.4.3 Wavelet Families and Mother Wavelets

In practice, when performing wavelet decomposition, selecting an appropriate mother wavelet is necessary and sensitive (Chen et al., 2017). The choice of wavelet depends on the signal characteristics and the nature of the application. Generally, wavelet families differ in terms of various essential properties, including support of the wavelet in time and frequency and rate of decay, symmetry or antisymmetric of the wavelet, number of vanishing moments, the regularity of the wavelet, and the existence of a scaling function. Although a mother wavelet is an effective tool for denoising contaminated signals, selecting a proper mother wavelet for specific bio-signal such as EEG remains challenging because of its non-deterministic nature (Chen et al., 2017). Table 5.1 shows the mother wavelets in accordance with the wavelet families used in this study, which were performed as the filter bank function.

The mother wavelet can be represented as wavelet coefficients with different lengths, depending on its filtering property. The use of the mother wavelet is intensely explained in the next section. In this study, the wavelet families listed in Table 5.1 are employed as a filter bank basis, including Haar (haar), Daubechies (db), Biorthogonal (bior), Reverse Biorthogonal (rbio), Symlets (sym), and Coiflets (coif).

Table 5.1. Mother wavelet used in this study.

Wavelet Family	Mother Wavelet
Haar (haar)	haar
Daubechies (db)	db1, db2, db3, db4, db5, db6, db7, db8, db9, db10
Biorthogonal (bior)	bior1.1, bior1.3, bior1.5, bior2.2, bior2.4, bior2.6, bior2.8, bior3.1, bior3.3, bior3.7, bior3.9, bior4.4, bior5.5, bior6.8
Reverse Biorthogonal (rbio)	rbio1.1, rbio1.3, rbio1.5, rbio2.2, rbio2.4, rbio2.6, rbio2.8, rbio3.1, rbio3.3, rbio3.7, rbio3.9, rbio4.4, rbio5.5, rbio6.8
Symlets (sym)	sym2, sym3, sym4, sym5, sym6, sym7, sym8
Coiflets (coif)	coif1, coif2, coif3, coif4, coif5

5.4.4 Discrete Wavelet Transform: Matrix-Vector Operation

In order to best utilise the power of heterogeneous computing, the DWT algorithm needs to be modified to suit the architecture of the OpenCL devices. Formulating the whole process as a series of matrix multiplication is one of the most commonly used techniques when implementing the algorithm using OpenCL.

Daubechies Wavelet Transform (Daubechies, 2009) is an alternative approach to the DWT that applies the scaling function directly to the input. Referring to the use of Daubechies Wavelet Transform, the computational structure was clearly explained in the context of linear algebra operation, e.g. a matrix form (Yan, 2009). However, this transformation addresses the edge effects by modifying the wavelet function matrix so that it may limit merely to symmetrisation or periodic padding method. In this study, a modified DWT matrix transformation is proposed and consists of the wavelet matrix ($\mathbf{W}$), the modified input sequence ($\mathbf{U}$), and output coefficients ($\mathbf{S}$). The N-by-P wavelet matrix can be formed by the mother wavelet, which is the combination of L-element filtering coefficients, including scaling function ($\mathbf{g}$) and wavelet function ($\mathbf{h}$). The P-element modified input sequence is the combination of the N-element input signal ($\mathbf{x}$) and R-element extension mode ($\boldsymbol{\sigma}$). The N-element output coefficient combines approximation coefficients ($\mathbf{a}$) and detail coefficients ($\mathbf{c}$). The proposed matrix is denoted as follows:

$$\mathbf{S} = \mathbf{W} \cdot \mathbf{U} \tag{5.3}$$

$$\mathbf{W} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{L-1} & 0 & 0 & & & \cdots & 0 & 0 \\ g_0 & g_1 & \cdots & g_{L-1} & 0 & 0 & & & \cdots & 0 & 0 \\ 0 & 0 & h_0 & h_1 & \cdots & h_{L-1} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & g_0 & g_1 & \cdots & g_{L-1} & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & & & \ddots & & & & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & h_0 & h_1 & \cdots & h_{L-1} & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & g_0 & g_1 & \cdots & g_{L-1} & 0 & 0 \\ 0 & 0 & \cdots & & & 0 & 0 & h_0 & h_1 & \cdots & h_{L-1} \\ 0 & 0 & \cdots & & & 0 & 0 & g_0 & g_1 & \cdots & g_{L-1} \end{bmatrix}_{N \times P} \tag{5.4}$$

$$\mathbf{U} = [x_0 \quad x_1 \quad \cdots \quad x_{N-1} \mid \sigma_0 \quad \sigma_1 \quad \cdots \quad \sigma_{R-1}]^{\mathrm{T}} \tag{5.5}$$

$$\mathbf{S} = \left[c_0 \quad a_0 \quad c_1 \quad a_1 \quad \cdots \quad c_{\frac{N}{2}-1} \quad a_{\frac{N}{2}-1} \right]^{\mathrm{T}}. \tag{5.6}$$

Equation 5.3 is a matrix-vector form for calculating the DWT, in which the input requires a wavelet matrix and a modified input vector as represented in Equations 5.4 and 5.5, respectively. The size of the wavelet matrix (W) is N-by-P which $P = N + R$ and $R = L - 2$. If the number of filtering coefficients is two, the extension mode is not required. However, if the number is more than two, the extension mode plays an essential role in this step because distortion effects arise at the edge. Hence, the selection of the extension method needs to be considered. The combined output DWT coefficients (detail and approximation) are represented by a vector form as shown in Equation 5.6. The example of the transformation matrix with 4-element filtering coefficients and the 8-element input signal is demonstrated in Figure 5.3. The extension mode is required to handle the edge problem. The output wavelet coefficients are half of the input vector.

Note that the edge problem arises at the last iteration since x_N and x_{N+1} do not exist. To handle this problem, signal extension such as periodic padding or symmetrisation was used to replace the missing elements.

$$\begin{bmatrix} c_0 \\ a_0 \\ c_1 \\ a_1 \\ c_2 \\ a_1 \\ c_3 \\ a_3 \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \end{bmatrix} \bullet \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ \hline \sigma_0 \\ \sigma_1 \end{bmatrix}$$

Figure 5.3. Showing the transformation matrix for an 8-element signal with the 4-element wavelet coefficients. The edge problem arising at the last iteration can be resolved by using signal extension technique.

5.4.5 Multi-Channel DWT Based on Matrix Multiplication

According to Equation 5.3, the proposed matrix formula is presented only for one input channel; thus, the 1-D wavelet coefficients are generated. In addition, this study considered the DWT calculation based on multiple channels so that the transformation matrix can be rearranged as follows:

$$\begin{bmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,M-1} \\ a_{0,0} & a_{0,1} & \cdots & a_{0,M-1} \\ c_{1,0} & c_{1,1} & \cdots & c_{1,M-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,M-1} \\ \vdots & \vdots & & \vdots \\ c_{\frac{N}{2}-1,0} & c_{\frac{N}{2}-1,1} & \cdots & c_{\frac{N}{2}-1,M-11} \\ a_{\frac{N}{2}-1,0} & a_{\frac{N}{2}-1,1} & \cdots & a_{\frac{N}{2}-1,M-1} \end{bmatrix} = \mathbf{W} \cdot \begin{bmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,M-1} \\ x_{1,0} & x_{1,1} & \cdots & x_{1,M-1} \\ \vdots & \vdots & & \vdots \\ x_{N-1,0} & x_{N-1,1} & \cdots & x_{N-1,M-1} \\ \sigma_{0,0} & \sigma_{0,1} & \cdots & \sigma_{0,M-1} \\ \sigma_{1,0} & \sigma_{1,1} & \cdots & \sigma_{1,M-1} \\ \vdots & \vdots & & \vdots \\ \sigma_{R-1,0} & \sigma_{R-1,1} & & \sigma_{R-1,M-1} \end{bmatrix}. \quad (5.7)$$

Referring to Equation 5.7, the N-by-M input matrix (M channels, N samples) forms the matrix multiplication that the input is a dense matrix while the wavelet matrix is almost sparse.

In practice, the matrix is not appropriate for calculating the wavelet transform because of its computational inefficiency and memory consumption. For example, each step of wavelet transform using a matrix algorithm costs the multiplication of the signal vector by a transform matrix, which has a computational complexity of $O(N^2)$ (where N is the data size for each transform step). On the other hand, a step of the standard transform is an $O(N)$ operation. Nevertheless, with the advance of HPC technology, multiplying an extensive matrix on the GPU is possible and can improve the computation speed.

5.4.6 Practical DWT Using Parallel Computing Scheme

In order to complete the DWT based on parallel implementation, the multiplication of the input matrix by the transform matrix is split into multiple sub-routines applied to multiple compute units. In this case, the multiplication with zero elements in the wavelet matrix can be discarded. A group of multiplications inside a kernel can be formed by setting up the number of work items. Most OpenCL devices offer vector operations, which mean addition, subtraction, and dot product are performable through

a single instruction. To take benefit from this, Equations 5.8 and 5.9 illustrate the approximation coefficients (**a**) and detail coefficients (**c**) of one level with a single-channel input sequence, which can be represented by a vector operation:

$$\mathbf{c}_i = \mathbf{h} \cdot \mathbf{x}_i \tag{5.8}$$

$$\mathbf{a}_i = \mathbf{g} \cdot \mathbf{x}_i \tag{5.9}$$

where **x**, **g**, and **h** are the input sequence, scaling function and wavelet function, respectively, illustrated as follows:

$$\mathbf{x}_i = \begin{bmatrix} x_{2i} \\ x_{2i+1} \\ x_{2i+2} \\ \vdots \\ x_{2i+L-1} \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ \vdots \\ h_{L-1} \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{L-1} \end{bmatrix} \tag{5.10}$$

where $i = 0,1,2,\dots,N/2-1$.

According to Equations 5.8 and 5.9, the detail coefficients can be formed as a dot product of the wavelet function sequence (**h**) and the modified input sequence ($\mathbf{x}_i$). Correspondingly, the approximation coefficients are represented by a dot product of the scaling function sequence (**g**) and the same modified input sequence. All the sequences have a length of L according to the length of filtering coefficients. Note that for every output coefficient, the index of the modified input sequence will be shifted by a factor of two due to the downsampling process. Figure 5.4 demonstrates the example of sub-multiplication inside the kernel for one-channel input with signal extension, in which each group is shifted every two elements.

5.4.7 Multi-Level DWT Algorithm Based on OpenCL

Computing the DWT for multiresolution analysis when used in multi-channel applications requires K levels of DWT execution. To demonstrate the DWT calculation via the proposed technique, given X is a 2D input array with N samples by M channels, the K-level DWT is required. Based on the proposed approach using the OpenCL devices, the execution workflow can be presented as follows:

$$\begin{bmatrix} c_0 \\ a_0 \\ c_1 \\ a_1 \\ c_2 \\ a_1 \\ c_3 \\ a_3 \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \end{bmatrix} \bullet \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ \hline \sigma_0 \\ \sigma_1 \end{bmatrix}$$

Figure 5.4. Demonstrating handling the multiplication inside the kernel. A group is formed by setting up the number of work-item to the length of wavelet coefficients so each work-item returns two coefficients. With this setting, each compute unit is responsible to calculate the DWT of each channel in parallel.

Algorithm 1 Pseudocode for DWT based on OpenCL

Input: time-series sequence X
Output: DWT coefficients
Initialise processing sample: $n = N$
1: **for** $i = 1{:}K$ **do**
2: Downsample the input by 2: $n = n/2$
3: Set global work-item: $X(M, n)$
4: Set local work-item: $X(1, n)$
5: Execute multiplication kernel with the above settings
8: **end for**

Note that the maximum number of local work-item allowed depends on different hardware acceleration devices.

5.4.8 Optimisation for Parallel Computing Scheme

To improve the performance of OpenCL kernels, some practical optimising techniques were used in this study. Though many vendors deliver the built-in optimisation inside the devices, fine-tuning optimising parameters are still needed in practice. The techniques include; avoiding global memory access, vectorising data, using pipes protocol, and specifying the workgroup size. The recommended techniques are explained in depth as follows:

1) Avoid Multiple Access on Global Memory: accessing global memory, which is stored on the host side, significantly decreases the performance of the OpenCL kernel due to the architecture of the device (Dao et al., 2014; Wang et al., 2014). Multiple reading and writing from/to the global memory can even take longer. To counter the effect of the memory complexity, the vendor suggests copying the data from global memory to local memory and processing the data under the local memory even using atomic operation. The local memory shared by all the work items in a single workgroup is faster than the global memory shared by all the workgroups on the device.

2) Use Vector Data Structure: vector operation is a powerful function when working with a massive array data structure as it significantly speeds up basic mathematical operations. Most OpenCL devices fully support vector operations. The data can be stored in a width of 2, 4, 8, and 16, allowing the use of built-in functions such as addition and subtraction operations between data. Thus, using vectorisation delivers extreme performance on a kernel (Fang et al., 2018).

3) Specify Workgroup Size: in the case of using local memory in the kernel, the actual number of workgroups that can run concurrently on the OpenCL device is limited by the maximum size of the workgroup. Therefore, larger sizes are required to keep the device utilisation high within the restricted number of the workgroup. The recommended technique (Shata et al., 2019) is using power-of-two workgroup sizes between 64 and 256.

5.4.9 Programming Workflow

Practically, implementing the OpenCL for mathematical computation is the management of kernel and data transfer. Frequently, initialising the OpenCL device with improper parameters causes lower performance and data transfer bottlenecks. This issue can be primarily prevented by considering the amount of incoming data and the availability of the device's memory. Figure 5.5 explains a kernel workflow and shows how the data transfers inside the context, in which each workgroup processes the kernel on a single compute unit.

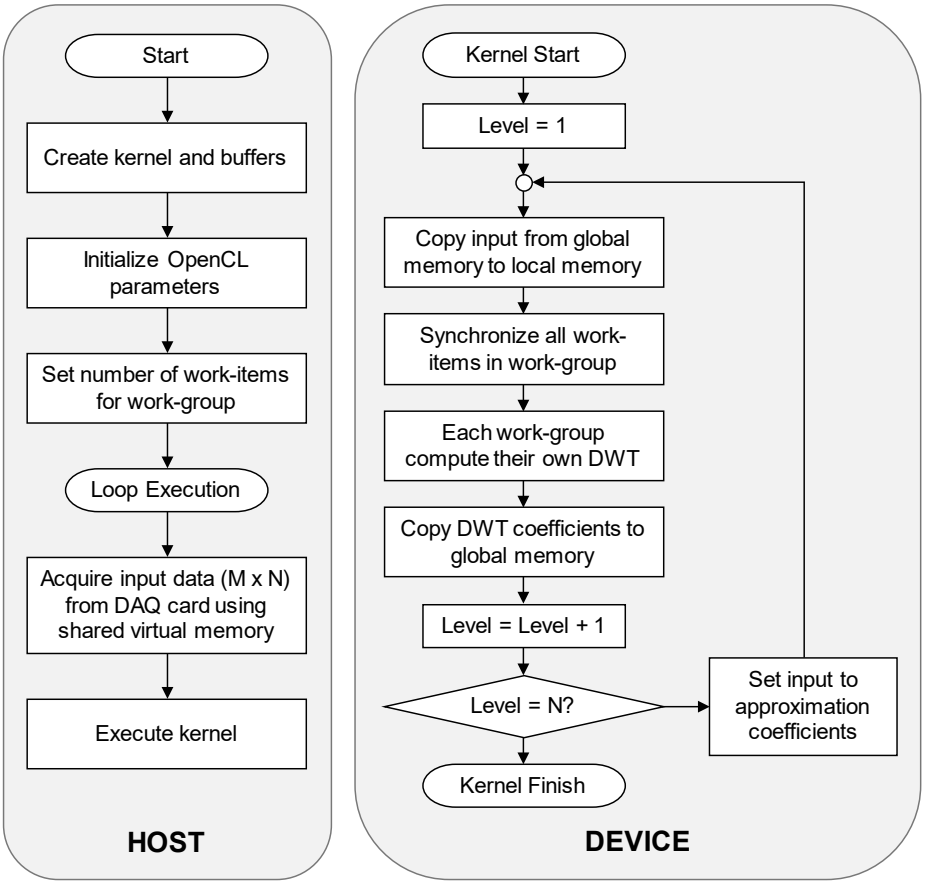

Figure 5.5. Illustrating the OpenCL workflow of computing multichannel DWT.

The diagram displayed in Figure 5.5 shows that the input data are attained from the data acquisition module and the kernel. It shows the real-time processing can be executed immediately on data arrival. The optimisation techniques discussed previously were used in this study, including using local memory, data vectorisation, and a specific workgroup size.

5.4.10 Real-Time Multi-Channel Implementation

To evaluate the computational performance of the modified DWT algorithm under the parallel architecture, the proposed real-time system was developed using the Qt environment (Qt 5.12 LTS), including signal acquisition, OpenCL implementation and graphical user interface (GUI). To acquire the EEG signal, the NI PCIe-6343, which is a 32-channel analogue input device with a resolution of 16 bits and a maximum sample rate of 500 kS/s, was used in this study with a variety of sampling rates,

including 128, 256, 512, 1024, and 2048 Hz. Regarding hardware acceleration, the AMD graphic cards (NVIDIA Quadro P4000 with the maximum compute units of 14 and the maximum workgroup size of 1024) were selected to deploy the wavelet computation based on the OpenCL 2.0 version. Note that the wavelet coefficients were computed only for one level in this study.

5.4.11 System Evaluation Using Pseudo Real-Time Simulation

The simulation system produced a pseudo-real-time environment for evaluating the performance of the proposed system. An archived EEG data set from our research group was used to regenerate an input signal through the NI PCIe-6738 card, which is a 32-channel analogue output device with an output resolution of 16 bits and the voltage range of ±10 Volts. The archive signal was directly sent to the signal acquisition module using RG58 50-Ohm coaxial cables. To attain the highest signal quality, each channel was regenerated at 2 kHz following the sampling frequency of the data set.

5.4.12 Data Description for System Evaluation

As stated previously, this evaluation used another EEG data set (Syam bin Ahmad Jamil, 2017) that has the same functionality and experimental protocol as the previous public data sets to verify the system performance. The data was acquired from 10 healthy volunteers performing motor imagery tasks in an ethically approved study. The experimental task was to imagine moving the subject's dominant hand toward different directions in response to a visual cue. The cues instructed imagined motion of the wrist in 4 different directions (flexion, extension, radial deviation and ulnar deviation). The diagram of the task is shown in Figure 5.6, in which the cue appeared at the second 3.

The original recordings comprised 32-channel EEG using the 10-20 international system and sampled at a 2 kHz per channel using a Neuroscan SynAmps 2 system. In addition to computing real-time wavelet decomposition significantly, all EEG channels were formed as the initial input 32-by-N matrix, which N is denoted as the sampling rate. Hence, the output coefficients from each level were 32-by-K, which K is denoted as the length of the input vector.

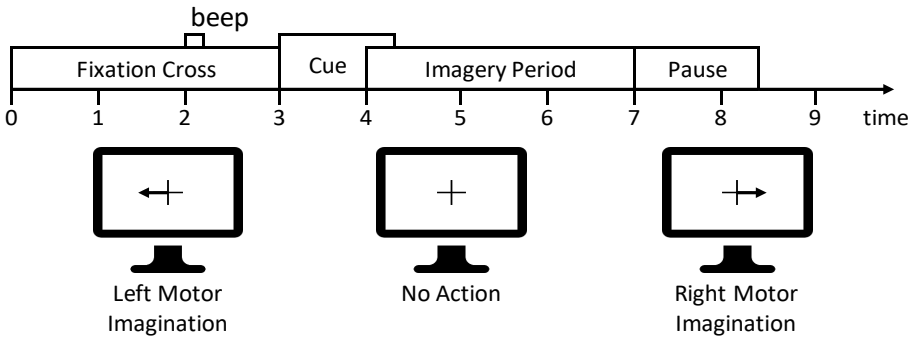

Figure 5.6. Showing the diagram of cue-based motor imagery task from the archive data set that the cue presented in the third second.

5.4.13 Computational Performance: Parallel Versus Sequential

Benchmarking the computational performance was primarily investigated, and the runtime of complete command execution on the compute unit was averaged 100 times. The execution involved transferring data from host to device, processing the kernel, and transferring data back to the host. The number of the workgroup was evaluated inside the DWT kernel, including 32, 64, 128, and 256. Furthermore, to compare the performance of different OpenCL devices, the DWT algorithm was deployed on both the CPU (sequential) and the GPU (parallel). In order to benchmark the computing performance between the proposed system and the current solution, MATLAB software commonly used for real-time signal processing (i.e., MATLAB toolboxes, including Signal Processing Toolbox and Wavelet Toolbox) were selected to compare the computing performance using the same testing parameters. The estimation errors of the DWT coefficients were also reported. This study ran on 64-bit Window 10 OS, with 32-GB DDR4 and Intel Xeon E5-1630 v4.

5.5 Results and Discussion

5.5.1 Estimation Error on Wavelet Coefficient Output

Validation of output wavelet coefficients is essential before processing in further steps as the existence of error in the coefficients increases in the possibility of misclassification. In this work, the coefficients calculated from the proposed matrix multiplication were compared to the DWT result from MATLAB function (*dwt*,

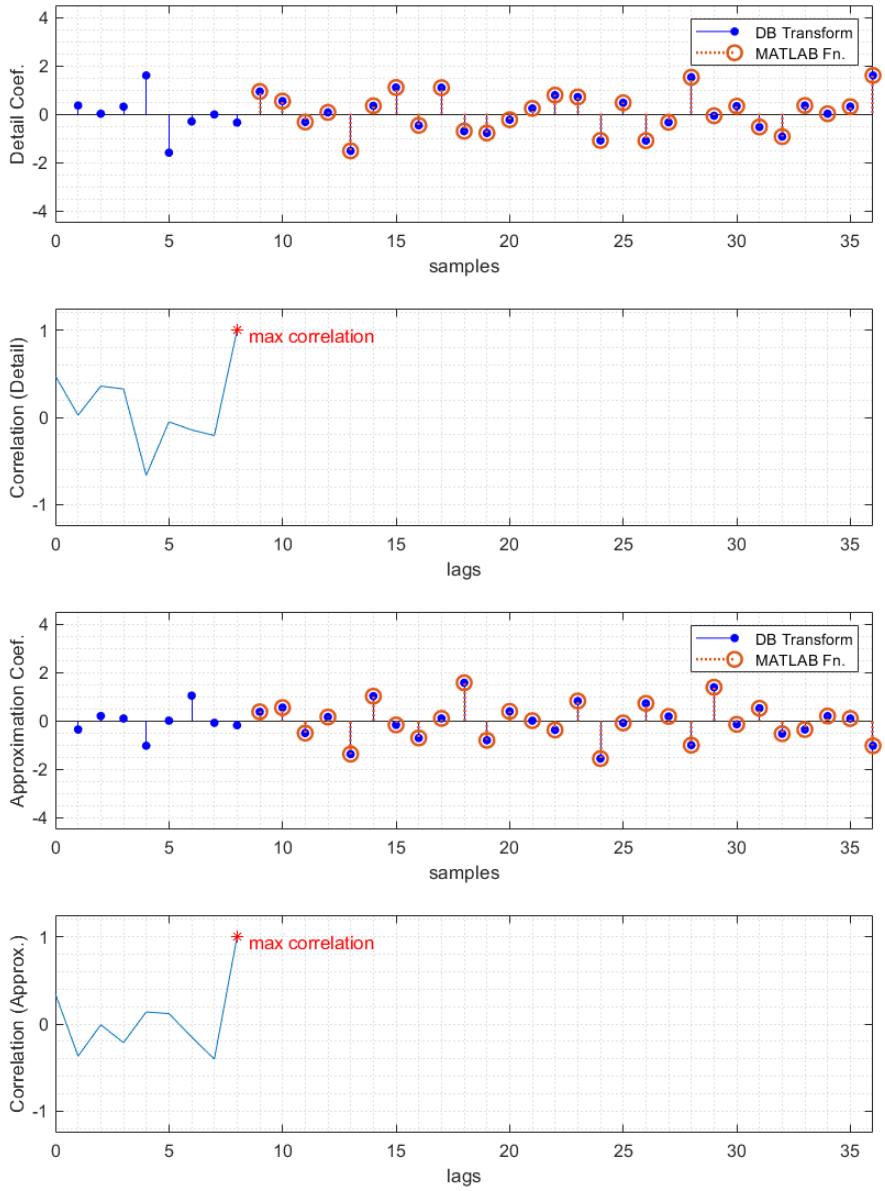

Figure 5.7. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using Daubechies wavelet transform based on the bior6.8 wavelet and the signal extension mode of "periodic padding (ppd)". In addition to each coefficient output, the normalised cross-correlation between the coefficients attained from the proposed technique and the MATLAB DWT function was calculated to locate the lag point at the maximum correlation value.

single-level 1-D discrete wavelet transform) with the same parameter settings (for example, type of wavelet, length of the input, and signal extension mode). Figures 5.7 and 5.8 illustrate a comparison of different DWT conditions where all the coefficient outputs from matrix multiplication are identical to the results from the MATLAB function. Note that the input data was randomly sampled from the EEG data set.

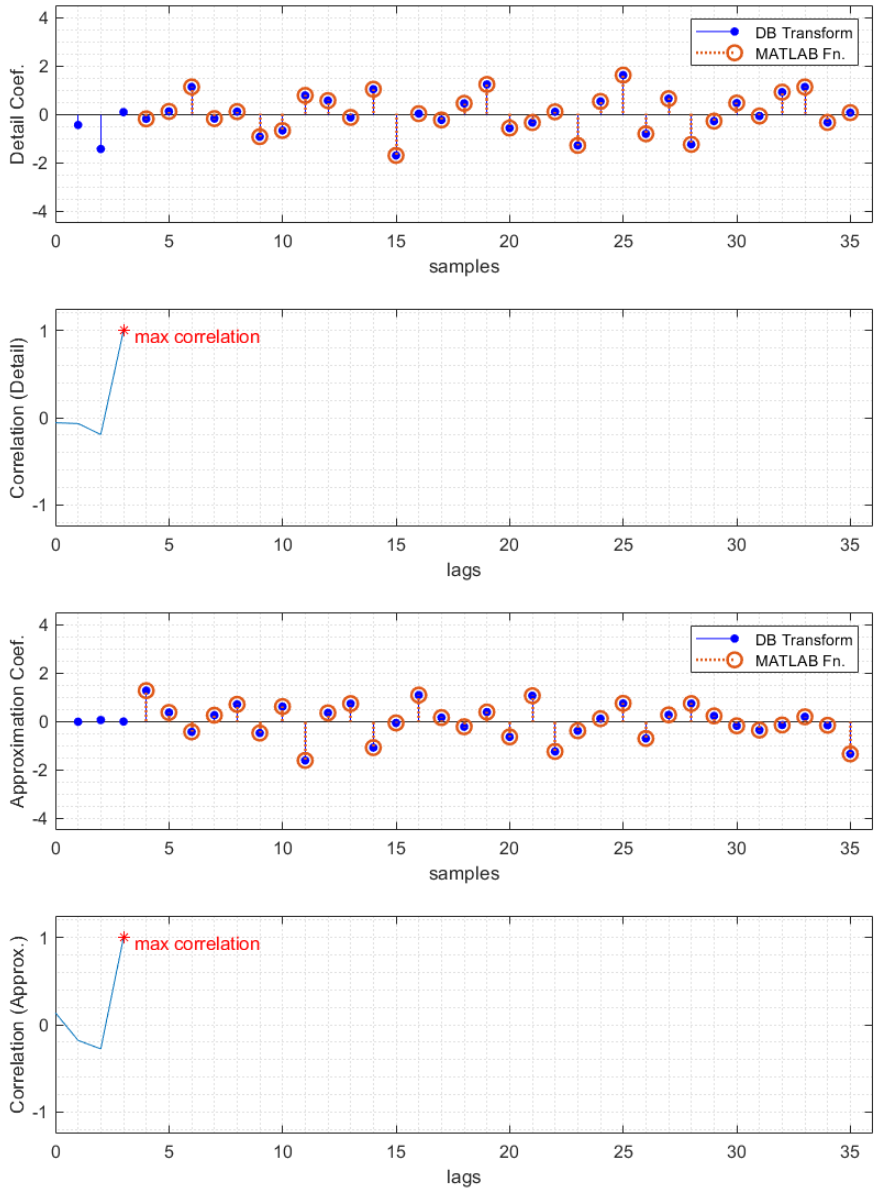

Figure 5.8. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using Daubechies wavelet transform based on the db4 wavelet and the signal extension mode of "zero padding (zpd)". In addition to each coefficient output, the normalized cross-correlation between the coefficients attained from the proposed technique and the MATLAB DWT function was calculated to locate the lag point at the maximum correlation value.

According to the results, the output coefficients attained from both functions show similarity in detail and approximation coefficients, yielding the maximum correlation of 1 with a small number of lags. The lag issue occurred because the DWT function in MATLAB automatically performed the full convolution when calculating the DWT coefficients, resulting in the zero-padded edges. In contrast, using the proposed DWT

algorithm on a parallel architecture not only achieves the same coefficients but also reduces the computational complexity associated with this full convolution. More results from different wavelets are provided in Appendix C.

5.5.2 Computational Performance

Table 5.2 shows the runtime comparison when performing the DWT with different window lengths and workgroup sizes, starting from the beginning of data acquisition throughout the processing pipeline. The speedup ratio is calculated from the CPU runtime divided by the GPU runtime. The results indicate that running the DWT on the GPU yielded an approximate 90 times faster performance compared to the CPU in all circumstances. In general, calculating single-level coefficients with N-input, L-filter, and M-channels requires a computational complexity of $O(NLM)$. Whereas using a parallel scheme can perform the computation of each channel concurrently, yielding a complexity of $O(NL)$ only. Considering the results, setting up the parameters such as the number of workgroups and work-items had an impact on the speedup ratio. This is because of the different infrastructure of the GPU, i.e. memory model in data transmission (Rafique et al., 2014). Therefore, it is important to

Table 5.2. Computation time used in this study.

Input Length	CPU Runtime	Workgroup Size	GPU Runtime[a]	Speedup Ratio[b]
256	17283	32	188	92.15
		64	192	90.16
		128	194	89.05
		256	190	90.96
512	17489	32	199	87.70
		64	196	89.22
		128	200	87.38
		256	195	89.81
1024	18154	32	215	84.27
		64	211	86.12
		128	205	88.58
		256	206	87.94
2048	22439	32	250	89.63
		64	242	92.65
		128	242	92.67
		256	244	91.88

[a]Execution time was observed in microsecond (μsec).
[b]Ratio was calculated from the runtime of the CPU divided by the GPU.

understand the system compatibility and the accelerating device's specification, including memory size, memory bandwidth, number of compute units, and number of stream processors before implementing any algorithms or optimisation techniques to avoid data transmission bottlenecks when performing repetitive tasks.

5.6 Chapter Summary

This chapter presents the possibility of implementing a parallel computing scheme in processing a time-frequency-domain feature extraction method based on the standard open-source platform called OpenCL. In addition to the heterogeneous computing concept, discrete wavelet transform was selected in order to demonstrate the use of a parallel computing scheme by modifying the original transformation based on a matrix multiplication approach. Furthermore, a signal extension mode was added in the proposed matrix form to minimise the border distortion effect. Evaluating the system under a pseudo-real-time environment showed identical output wavelet coefficients compared to the standard function software. Moreover, the results show the effectiveness of using hardware acceleration in completing massive computational tasks in terms of computation performance. In addition, running the DWT using the proposed parallel scheme outperformed the sequential methods up to 90 times in all scenarios. The proposed parallel approach can be adapted to any signal processing algorithms by representing the calculation based on matrix operation. Furthermore, the highlights and future work will be recapitulated in Chapter 7.

The next chapter presents the implementation of the proposed parallel computing technique used in this chapter in combination with the proposed recursive technique used in the previous chapter, which shows the advantage and the flexibility of using a matrix-vector form in signal processing arithmetic.

5.7 References

Alickovic, E., Kevric, J. & Subasi, A. 2018. Performance evaluation of empirical mode decomposition, discrete wavelet transform, and wavelet packed decomposition for automated epileptic seizure detection and prediction. *Biomedical signal processing and control,* 39**,** 94-102.

Amin, H. U., Malik, A. S., Ahmad, R. F., Badruddin, N., Kamel, N., Hussain, M. & Chooi, W.-T. 2015. Feature extraction and classification for EEG signals using wavelet transform and machine learning techniques. *Australasian physical & engineering sciences in medicine,* 38**,** 139-149.

Bernabé, S., Botella, G., Martin, G., Prieto-Matias, M. & Plaza, A. 2017. Parallel implementation of a full hyperspectral unmixing chain using opencl. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* 10**,** 2452-2461.

Chen, D., Wan, S., Xiang, J. & Bao, F. S. 2017. A high-performance seizure detection algorithm based on Discrete Wavelet Transform (DWT) and EEG. *PloS one,* 12**,** e0173138.

Dao, T. T., Kim, J., Seo, S., Egger, B. & Lee, J. 2014. A performance model for gpus with caches. *IEEE Transactions on Parallel and Distributed Systems,* 26**,** 1800-1813.

Daubechies, I. 2009. *The wavelet transform, time-frequency localization and signal analysis*, Princeton University Press.

Durongbhan, P., Zhao, Y., Chen, L., Zis, P., De Marco, M., Unwin, Z. C., Venneri, A., He, X., Li, S. & Zhao, Y. 2019. A dementia classification framework using frequency and time-frequency features based on EEG signals. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 27**,** 826-835.

Fang, M., Fang, J., Zhang, W., Zhou, H., Liao, J. & Wang, Y. 2018. Benchmarking the GPU memory at the warp level. *Parallel Computing,* 71**,** 23-41.

Ikuzawa, T., Ino, F. & Hagihara, K. 2016. Reducing memory usage by the lifting-based discrete wavelet transform with a unified buffer on a GPU. *Journal of Parallel and Distributed Computing,* 93**,** 44-55.

Kaeli, D. R., Mistry, P., Schaa, D. & Zhang, D. P. 2015. *Heterogeneous computing with OpenCL 2.0*, Morgan Kaufmann.

Kinoshita, T., Fujiwara, K., Kano, M., Ogawa, K., Sumi, Y., Matsuo, M. & Kadotani, H. 2020. Sleep Spindle Detection Using RUSBoost and Synchrosqueezed Wavelet Transform. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 28**,** 390-398.

Li, M., Chen, W. & Zhang, T. 2017. Classification of epilepsy EEG signals using DWT-based envelope analysis and neural network ensemble. *Biomedical Signal Processing and Control,* 31**,** 357-365.

Ma, X., Wang, D., Liu, D. & Yang, J. 2020. DWT and CNN based multi-class motor imagery electroencephalographic signal recognition. *Journal of neural engineering,* 17**,** 016073.

Matela, J. GPU-based DWT acceleration for JPEG2000. Annual Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, 2009. Citeseer, 136-143.

Mohammadi, Z., Frounchi, J. & Amiri, M. 2017. Wavelet-based emotion recognition system using EEG signal. *Neural Computing and Applications,* 28**,** 1985-1990.

Montanari, L., Basu, B., Spagnoli, A. & Broderick, B. M. 2015. A padding method to reduce edge effects for enhanced damage identification using wavelet analysis. *Mechanical Systems and Signal Processing,* 52**,** 264-277.

Nugteren, C. CLBlast: A tuned OpenCL BLAS library. Proceedings of the International Workshop on OpenCL, 2018. 1-10.

Quan, T. M. & Jeong, W.-K. 2016. A fast discrete wavelet transform using hybrid parallelism on GPUs. *IEEE Transactions on Parallel and Distributed Systems,* 27**,** 3088-3100.

Rafique, A., Constantinides, G. A. & Kapre, N. 2014. Communication optimization of iterative sparse matrix-vector multiply on GPUs and FPGAs. *IEEE Transactions on Parallel and Distributed Systems,* 26**,** 24-34.

Sharma, B. & Vydyanathan, N. Parallel discrete wavelet transform using the Open Computing Language: a performance and portability study. 2010 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010. IEEE, 1-8.

Shata, K., Elteir, M. K. & El-Zoghabi, A. A. 2019. Optimized implementation of OpenCL kernels on FPGAs. *Journal of Systems Architecture,* 97**,** 491-505.

Simmross-Wattenberg, F., Rodríguez-Cayetano, M., Royuela-Del-Val, J., Martín-González, E., Moya-Sáez, E., Martín-Fernández, M. & Alberola-López, C. 2018. OpenCLIPER: an OpenCL-based C++ Framework for overhead-reduced medical image processing and reconstruction on heterogeneous devices. *IEEE journal of biomedical and health informatics,* 23**,** 1702-1709.

Simon, S., Rieder, P., Schimpfle, C. & Nossek, J. A. CORDIC-based architectures for the efficient implementation of discrete wavelet transforms. 1996 IEEE International Symposium on Circuits and Systems (ISCAS), 1996. IEEE, 77-80.

Strang, G. & Nguyen, T. 1996. *Wavelets and filter banks*, SIAM.

Suda, N., Chandra, V., Dasika, G., Mohanty, A., Ma, Y., Vrudhula, S., Seo, J.-S. & Cao, Y. Throughput-optimized OpenCL-based FPGA accelerator for large-scale convolutional neural networks. Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2016. 16-25.

Sun, Y., Liu, B. & Xu, X. An OpenCL-Based Hybrid CNN-RNN Inference Accelerator On FPGA. 2019 International Conference on Field-Programmable Technology (ICFPT), 9-13 Dec. 2019 2019. 283-286.

Sweldens, W. 1996. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and computational harmonic analysis,* 3**,** 186-200.

Syam Bin Ahmad Jamil, S. H.-F. 2017. *Developing multi degree of freedom control brain computer interface system for spinal cord injury patients.* Thesis [PhD] --University of Strathclyde, 2017.

Tenllado, C., Lario, R., Prieto, M. & Tirado, F. The 2D discrete wavelet transform on programmable graphics hardware. IASTED Visualization, Imaging and Image Processing Conference, 2004.

Tenllado, C., Setoain, J., Prieto, M., Pinuel, L. & Tirado, F. 2008. Parallel implementation of the 2D discrete wavelet transform on graphics processing units: Filter bank versus lifting. *IEEE Transactions on Parallel and Distributed Systems,* 19**,** 299-310.

Tian, X., Deng, Z., Ying, W., Choi, K.-S., Wu, D., Qin, B., Wang, J., Shen, H. & Wang, S. 2019. Deep multi-view feature learning for EEG-based epileptic seizure detection. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 27**,** 1962-1972.

Van Der Laan, W. J., Jalba, A. C. & Roerdink, J. B. 2010. Accelerating wavelet lifting on graphics hardware using CUDA. *IEEE Transactions on Parallel and Distributed Systems,* 22**,** 132-146.

Vázquez, S., Amor, M. & Fraguela, B. B. 2019. Portable and efficient FFT and DCT algorithms with the Heterogeneous Butterfly Processing Library. *Journal of Parallel and Distributed Computing,* 125**,** 135-146.

Vidyaratne, L. S. & Iftekharuddin, K. M. 2017. Real-time epileptic seizure detection using EEG. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 25**,** 2146-2156.

Wang, B., Alvarez-Mesa, M., Chi, C. C. & Juurlink, B. 2014. Parallel H. 264/AVC motion compensation for GPUs using OpenCL. *IEEE Transactions on Circuits and Systems for Video Technology,* 25**,** 525-531.

Wang, G., Xiong, Y., Yun, J. & Cavallaro, J. R. Accelerating computer vision algorithms using OpenCL framework on the mobile GPU-a case study. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013. IEEE, 2629-2633.

Wong, T.-T., Leung, C.-S., Heng, P.-A. & Wang, J. 2007. Discrete wavelet transform on consumer-level graphics hardware. *IEEE Transactions on Multimedia,* 9**,** 668-673.

Wu, Y.-L., Agrawal, D. & El Abbadi, A. A comparison of DFT and DWT based similarity search in time-series databases. Proceedings of the ninth international conference on Information and knowledge management, 2000. 488-495.

Yan, J. 2009. Wavelet matrix. *Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada*.

Zhao, D. 2015. Fast filter bank convolution for three-dimensional wavelet transform by shared memory on mobile GPU computing. *The Journal of Supercomputing,* 71**,** 3440-3455.

CHAPTER 6

DWT IMPLEMENTATION FOR REAL-TIME BCI

The preceding chapter described a modified technique for finding DWT coefficients that computationally outperforms traditional methods. Also, the sliding technique presented in Chapter 4 shows its advantages when performing Fourier analysis in real time. The techniques proposed in both chapters can be applied to address real-world BCI applications, which mostly require real-time and high-rate processing. This chapter shows how to use recursiveness and parallel architecture to solve similar real-time processing problems, particularly in sliding-window-based calculations. Combining the recursive structure described in Chapter 4 with the parallel architecture described in Chapter 5 makes it possible to significantly decrease the computing time of such real-time processing. Throughout this chapter, signal processing based on the sliding and parallel architecture is thoroughly explained.

6.1 Introduction

Recently, discrete wavelet transform (DWT) has been increasingly used in real-time applications, including object tracking (Cheng and Chen, 2006; Hsia and Guo, 2014), power measurement (Alves et al., 2017), biosignal feature extraction (Desai and Sankhe, 2012; Galatenko et al., 2012), and filtering (Freitas et al., 2019). Those studies relied on either the first-generation wavelet transform (FGWT) or the second-generation wavelet (SGWT), which showed promising results and good potential compared to their traditional feature extraction methods. However, many studies from the above reported their limitations on scaling up the system using DWT, for instance, increasing input size or sampling frequency, without affecting the computation time, resulting in the speed-accuracy tradeoff. Remarkably, the first-generation filter bank

approach has been primarily used for several classification-based applications, which require utilising the DWT coefficients to form a feature vector. According to the DWT architecture, the filter bank method usually has more computational complexity than the second-generation lifting scheme at the same wavelet family. Moreover, most lifting structures are intensively modified for executing on dedicated hardware such as DSP or FPGA, which fully supports real-time operation. In contrast, the modification of the first-generation DWT algorithms based on CPU-like hardware (which offers more flexibility of use) remains less developed due to its computationally expensive architecture.

For this reason, this issue has become a major interest for reducing the complexity of the FGWT method, i.e., the filter bank algorithm, in order to deliver the DWT coefficients more efficiently, especially when used in a sliding-window manner. Based on a search of the literature, no study to date has presented the real-time DWT calculation with modifying the algorithm based on a sliding window approach. Therefore, this chapter presents an alternative method for computing the incremental DWT coefficients using the modified filter bank algorithm.

6.2 Background and Related Works

The development of the DWT algorithm has become popular since the rapid growth in demand for time-frequency analysis, mainly driven by the imaging industry (Demirel and Anbarjafari, 2011; Ye and Hou, 2015; Zhou et al., 2020). Two approaches to computing the DWT coefficients, including FGWT and SGWT, have been increasingly employed. First-generation wavelets' basis functions are constructed using the Fourier transform, which requires subsampling after passing high-pass and low-pass filtering functions, as shown in Figure 6.1. Therefore, the functions are examined in the frequency domain. The analysis of stochastic signals using classical FGWT is carried out using memory-intensive DWT, which entails convolution and filtering.

Additionally, the computational cost of the method rises as the filter length increases. To circumvent these constraints, Sweldens (Sweldens, 1998) developed a novel method based on a lifting approach for constructing wavelet functions in the

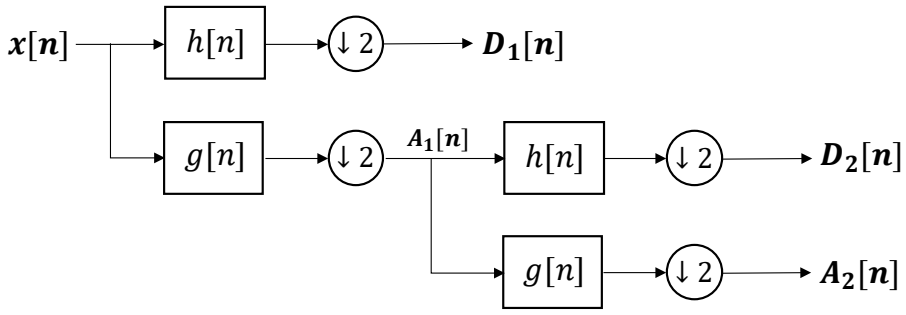

Figure 6.1. Presenting the first-generation filter bank architecture. Each level of DWT requires high-pass and low-pass filters, followed by a subsampling step.

time domain rather than the frequency domain. This idea has been influenced by Donoho and Harten, who separately proposed the concept of departing from the Fourier domain in the early 1990s (Donoho, 1993). Second-generation wavelets are used to refer to these basis functions. In contrast to FGWT, second-generation wavelet coefficients may be computed in place without the need for auxiliary memory. In addition, SGWT is memory-efficient because of its lifting method, which utilises fewer filter coefficients than first generation wavelets, making them computationally less costly. Despite its benefits, SGWT has been used exclusively for signal and image denoising and compression.

Although the FGWT structure such as filter bank DWT is comprehensible and can be implemented quickly, the algorithm may result in slow computation due to its convolutional structure, particularly when the number of inputs increases massively. This can be more challenging when used in real-time environments. The lifting-based DWT can be used to address the complexity issue, which uses roughly half of the time to compute the DWT coefficients compared to the traditional filter bank method (Liao et al., 2004). From the literature, there are plenty of available lifting schemes. Most of them are applied to modern FPGA hardware (Phadikar et al., 2019; Jana and Phadikar, 2020; Radhakrishnan and Themozhi, 2020) because the algorithm can be straightforwardly mapped to their hardware architecture. Each lifting scheme can be constructed based on a polyphase representation, described by the following basic operations: split, predict, and update, as illustrated in Figure 6.2. Hence, most of the wavelet families can be converted into those lifting steps.

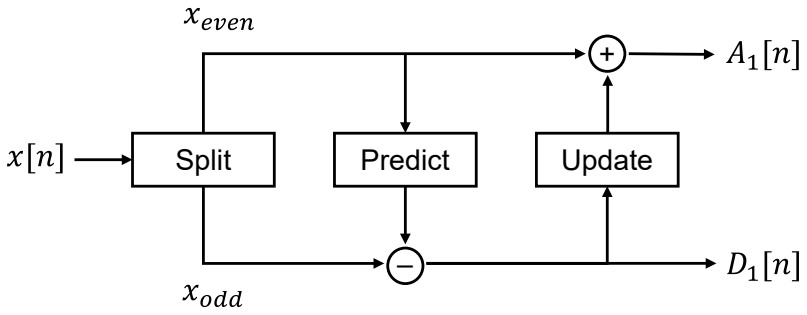

Figure 6.2. The general architecture of the second-generation lifting scheme. Each level comprises the following steps: split, predict, and update.

The performance comparison of DWT algorithms has been frequently proposed. Smoothing techniques based on SGWT and bivariate shrinkage have been shown to be more successful than the FGWT approach (Hatsuda, 2012). Also, lifting has been used to analyse EEG data for brain computer interfaces (Murugappan et al., 2008), with substantial improvement in classification results over the first generation wavelets. A modified lifting scheme (MLS) outperformed the first generation wavelet packet technique in resolving the recurrent issue of electromyographic (EMG) signal storage and transmission duration (Ele et al., 2009). Moreover, a symmetrical second-generation wavelet has been shown to perform better at image denoising than classical wavelets in terms of signal-to-noise ratio (Binglian and Qiusheng, 2009).

However, some DWT studies reported the opposite outcome. By comparing the performance of filter banks and liftings in image watermarking, the results show that FGWT-based watermarking produces more robustness than the SGWT method (Taha et al., 2020). In the context of genomic signal processing (Saini and Dewan, 2018), the SGWT method showed that it outperformed FGWT in terms of reconstruction errors but at the cost of increased calculation time. The actual calculation time for SGWT was longer than that for FGWT, even though SGWT has lower computational complexity because SGWT conducts in-place computing. Unlike FGWT, which allocates auxiliary memory to hold the calculated wavelet coefficients, SGWT computes and updates the coefficients in place each time. Based on the literature, there is no perfect solution to the DWT approach; consequently, it should be used in conjunction with the most appropriate numerical representations and wavelet functions for signal analysis with the goal of fully comprehending their underlying information.

6.3 Sliding DWT: A Case Study

According to the computationally expensive architecture of the filter bank algorithm, applying the method into real-time signal processing for large input sizes such as EEG signals is relatively challenging for a small computing machine. The parallel method described in Chapter 5 may significantly accelerate the computation of the filter-bank-based DWT. However, applying the proposed technique in real-time applications may be difficult, mainly if it is employed in a sliding-window manner that requires many input channels and a high sampling rate. Recalling DWT calculation from the previous chapter, given the wavelet matrix ($\mathbf{W}$), the modified input sequence ($\mathbf{U}$), and output coefficients ($\mathbf{S}$), the matrix-vector form can be derived by

$$\mathbf{S} = \mathbf{W} \cdot \mathbf{U}. \tag{6.1}$$

To demonstrate the redundant calculation in the above formula, given the input sequence $\mathbf{x}$ of 8 samples running incrementally with a 1-sample shift. By applying Equation 6.1 using the "db2" wavelet function with 4 coefficients and no extension mode applied, the DWT coefficients of the first four sequences can be obtained below.

$$\mathbf{S}_0 = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} \tag{6.2}$$

$$\mathbf{S}_1 = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} \tag{6.3}$$

$$\mathbf{S}_2 = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} \tag{6.4}$$

$$\mathbf{S}_3 = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \begin{bmatrix} x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix}. \tag{6.5}$$

6.4 Redundancy in Sliding-Window Calculation

Equation 6.1 shows that each sequence requires a total complexity of $\mathcal{O}(N^2)$ for matrix-vector multiplication. This complexity can be easily optimised with the help of parallel computing. However, performing this calculation with repeated input such as overlapping data results in ineffectiveness in managing computational resources. Following Equations 6.2 to 6.5, it can be seen that some elements of the input sequence perform duplicated multiplication at every two shifts. Given N-sample input, this means $N-2$ elements of the sequence $\mathbf{S}_0$ and $\mathbf{S}_2$ are duplicates, so the multiplication of these repeated elements causes N^2-2N operations, similar to the sequence $\mathbf{S}_1$ and $\mathbf{S}_3$. When the input size is increased, this may result in a considerable delay due to enormous duplication in the computation. Although parallelising matrix multiplication may speed up this calculation, a complex problem with the sliding window remains a data transmission bottleneck and a buffer overflow. Therefore, any repeated calculation should be discarded in order to improve the system performance. A recursive implementation is one of the most famous techniques used to deal with repeated calculations, such as this sliding window DWT.

6.5 Proposed Recursive Formula for Sliding DWT

As discussed previously, the output coefficients of the even index ($\mathbf{S}_0$ and $\mathbf{S}_2$) and the odd index ($\mathbf{S}_1$ and $\mathbf{S}_3$) show its duplication due to repeated multiplication. To address this issue, all repetitive calculations can be omitted by calculating only once and storing in a buffer. This step can be simply demonstrated in a recursive mathematical formula. According to Equations 6.2 to 6.5, it can be reformulated into two recursive consequences as follows:

$$\mathbf{S}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{S}_0 + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & 0 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \mathbf{x}_2 \tag{6.6}$$

$$\mathbf{S}_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{S}_1 + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & 0 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \cdot \mathbf{x}_3. \tag{6.7}$$

Equations 6.6 and 6.7 show the relationship among the current DWT coefficients $\mathbf{S}_n$, the previous coefficients $\mathbf{S}_{n-2}$, and the current input $\mathbf{x}_n$. Given the initial coefficients $\mathbf{S}_0$ and $\mathbf{S}_1$. The following sequence's DWT output can be easily calculated with less computation than the matrix-vector form in Equation 6.1. According to Equations 6.6 and 6.7, some repeated terms can be predefined. Let $\mathbf{P}$ be the $N \times N$ modified permutation matrix and $\mathbf{W}_\mathrm{m}$ be the $N \times N$ modified DWT matrix, the recursive formula of the above relationship can be obtained by

$$\mathbf{S}_n = \mathbf{P}\mathbf{S}_{n-2} + \mathbf{W}_\mathrm{m}\mathbf{x}_n \tag{6.8}$$

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \end{bmatrix}_{N \times N} \tag{6.9}$$

$$\mathbf{W}_{\mathrm{m}} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & & & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & h_{L-2} & h_{L-1} \\ 0 & 0 & \cdots & 0 & 0 & g_{L-2} & g_{L-1} \\ 0 & 0 & \cdots & 0 & 0 & h_0 & h_1 \\ 0 & 0 & \cdots & 0 & 0 & g_0 & g_1 \end{bmatrix}_{N \times N} . \tag{6.10}$$

According to Equation 6.8, the permutation matrix shown in Equation 6.9 moves the preceding sequence's output coefficients to two samples, a process that is analogous to array rotation, which requires only a circular buffer of N for rotating the output coefficients each iteration. As a result, the actual computation of the first term of Equation 6.8 can be ignored, so the component that requires the most multiplications is the latter term.

6.6 Reducing Unnecessary Calculation

Equation 6.8 contains many unnecessary calculations (e.g., operations involving zero). To eliminate needless multiplications of the second term of Equation 6.8, where many elements multiply to zero, such operations may be omitted when computing devices are used. Thus, the number of real arithmetic operations needed to calculate this latter term is eight multiplications and four additions. Consequently, Equation 6.8 has a total computational complexity of 8 multiplications and 6 additions when coupled with the first term since the last two components of the former term require no addition. This is a significantly smaller number than the conventional filter-back-based method, making Equation 6.8 more suited for real-time applications. However, suppose many

input channels are needed. In that case, the number of operations required is proportional to the size of the input channel, which may be an issue if large input channel counts are utilised.

Chapter 5 presents a parallel computing method that may accelerate any repeated arithmetic operation by utilising parallelism. This approach is also applicable to Equation 6.8, where the analytical operations in the latter term may be decreased via vectorisation or built-in functions such as the parallel dot product. Furthermore, the suggested method is adaptable to multiple channels, allowing for parallel calculation, resulting in faster computation.

6.7 Comparison of Computational Complexity

This section investigates the computational performance of running sliding-window DWT, comparing analytically and numerically the conventional filter-back-based algorithm, the second-generation lifting scheme, and the proposed processing technique. The comparison was evaluated based on a single-level DWT without using any extension mode (identical to zero paddings) and discarding any arithmetic addition and multiplication equal to zero.

To compare the number of operations for calculating one-level DWT coefficients of N input elements, given any mother wavelet which has L coefficients, the convolutional-based filter bank algorithm for decomposing 1D signals without extension mode (or zero-padding) requires $2N^2$ additions and $2N^2 - 2N$ multiplication. This expensive complexity comes from many multiplications of zero elements inside the convolution, which is, in practice, a waste of computational resources. While using the matrix-vector form as presented in Equation 6.1 may eliminate half of the calculation, this number is still high. With optimisation, the complexity of Equation 6.1 can be fully minimised to $LN - \frac{L^2}{2} + L$ additions and $LN - \frac{L^2}{4} + \frac{L}{2} - N$ multiplication by discarding any multiplication with zero elements. Nevertheless, the complexity might be problematic when applied to multiple channels with many input samples.

Table 6.1. Computational complexity per channel of a single-level DWT (no extension mode).

Method	Multiplication	Addition	Total Operation
Filter bank (convolution)	$2N^2$	$2N^2 - 2N$	$4N^2 - 2N$
Lifting scheme[a] (general)	-	-	$4N$
Equation 6.1 (matrix-vector)	N^2	$N^2 - N$	$2N^2 - N$
Equation 6.1 (optimised)	$LN - \frac{L^2}{2} + L$	$LN - \frac{L^2}{4} + \frac{L}{2} - N$	$2LN - \frac{3L^2}{4} + \frac{3L}{2} - N$
Equation 6.8 (matrix-vector)	$2N^2$	$2N^2 - 2N$	$4N^2 - 2N$
Equation 6.8 (optimised)	8	6	14

[a]From the 1D lifting-based DWT algorithm which in general each coefficient requires at least 4 operations for the predict and update steps (Liao et al., 2004).

Without optimisation, Equation 6.8 requires several operations similar to the filter bank method, which is $\mathcal{O}(N^2)$. By using fully optimised techniques, including predefining some constants and neglecting zero multiplication, the computational complexity of the proposed recursive DWT formula can be vastly reduced to only 8 multiplications and 6 additions for one level of DWT decomposition. In addition, the number of the input samples and the mother wavelet coefficients do not affect the total complexity at all. Compared to the state-of-the-art lifting method, which generally reduces the computational cost by more than half of the calculation, the proposed optimisation requires a minor operation than the lifting scheme. Table 6.1 summarises the computational complexity as discussed above, including the traditional filter-bank-based method, the lifting scheme, the matrix-vector approach, and the proposed recursive formula.

6.8 Memory Requirement

The recursive formula of the sliding DWT algorithm requires a small number of words to store the previous coefficients. Table 6.2 summarises the memory (words) required for calculating single-level DWT coefficients per channel. In general, Equation 6.8 requires $2N^2 - N$ words to store the previous coefficients and related matrices; however, only $N + 8$ words are required after optimisation. This shows another advantage of using recursiveness, which offers better computational speed and requires less storage than the traditional method.

Table 6.2. Memory (words) required per channel of a single-level DWT (no extension mode).

Parameter	Description	Size (Full)	Size (Optimised)
$\mathbf{S}_n$	Output DWT coefficients	N	N
$\mathbf{P}$	Modified permutation matrix	N^2	0
$\mathbf{W}_{\mathrm{m}}$	Modified DWT matrix	N^2	8

6.9 Experimental Design

To assess the proposed recursive formula's computational performance, the method was compared to various well-known DWT algorithms using pseudo-real-time simulations. The input signals used in the simulation were generated randomly from previously recorded EEG data from the laboratory's prior studies; most were related to motor imagery tasks. The filter-bank technique, the lifting method, the matrix-vector form (Equation 6.1), and the proposed recursive formula were all compared in terms of performance (Equation 6.8).

6.9.1 System Setup and Benchmarking

The simulation was developed using the MATLAB environment (MATLAB2021a), and all DWT algorithms were run on the same computing system (64-bit Windows 10 OS, 32 GB DDR3, and an Intel Xeon E5-1630 v4 processor). To evaluate the performance, the first-generation filter bank method and the second-generation lifting scheme were utilised for benchmarking based on the data set used in Chapter 5. These algorithms are available as MATLAB functions "dwt" and "lwt," respectively (MATLAB 2021a). The functions provide the output coefficients based on selected wavelet functions. However, for the sake of fairness, the customised DWT functions based on the filter bank and the lifting scheme were also included. Additionally, the matrix-vector formula in Equation 6.1 was executed directly, while the recursive form in Equation 6.8 was executed utilising full optimisation methods. There were a variety of different input sizes resampled, including 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, and 32768. Each simulation recorded the output DWT coefficients and the runtime of each algorithm. Each method was performed 10,000 times, and the average runtime was determined at the conclusion of the simulation. Note that the assessment is primarily concerned with runtime and speedup ratio.

6.9.2 Wavelet Families

A wavelet family is a collection of basis functions produced via dilations and translations of a single acceptable mother wavelet. In general, wavelet families differ from one essential property to another. There are several families of wavelets that have proven to be valid for decomposing DWT coefficients. The wavelet selection is determined by the signal or image properties and the application nature. Hence, it is critical to understand the characteristics of the analysis and synthesis wavelet in order to choose an optimal wavelet on an individual basis. For example, the symmetry or antisymmetry of the wavelet, the number of vanishing moments, regularity of the wavelet, and the existence of a scaling function (Ngui et al., 2013). In this study, the simulation was conducted based on several wavelet families. Five wavelets commonly used in BCI research, including Daubechies, Haar, Biorthogonal, Symlets, and Coiflets, were taken into account.

6.9.2.1 Daubechies (db)

The Daubechies family wavelets are the most well-known because they are compactly supported orthonormal wavelets (Daubechies and Sweldens, 1998). The Daubechies wavelets may be used to analyse epileptic electroencephalograms. For instance, db2 and db4 wavelets have been employed to identify seizures in real time (Zarei and Asl, 2021; Chen et al., 2017; Zandi et al., 2010). Additionally, the researchers used the db4 wavelet to generate a feature vector for categorising BCI motor imagery (Khalaf et al., 2019). The performance of db2 and db4 wavelets was evaluated in this research.

6.9.2.2 Haar (haar)

The Haar wavelet is the most straightforward wavelet, which is discontinuous and similar to a step function (Haar, 1910). Therefore, numerous time-frequency studies have made comparisons between this wavelet and other sophisticated wavelets. However, the Haar wavelet has the disadvantage of not being continuous and, therefore, not being differentiable on a technical level. Nevertheless, this wavelet has been extensively utilised in BCI research to eliminate ocular artefacts from real-time EEG processing (Khatun et al., 2016; Majmudar and Morshed, 2016). Similarly, the

Haar wavelet demonstrated excellent results when used to identify drowsiness on a single EEG channel (da Silveira et al., 2016).

6.9.2.3 Biorthogonal (bior)

This wavelet family displays the feature of a linear phase, which is required to reconstruct signals and images. The bior wavelet families were employed to detect focal EEG signals, and the findings indicated that bior 3.5 performed better than other wavelets (Sairamya et al., 2021). In addition, bior 3.3 had the highest classification accuracy for emotion detection using EEG data (Chen et al., 2019). Bior wavelets were utilised in conjunction with evolutionary algorithms to denoise EEG signals (Alyasseri et al., 2021). Furthermore, bior 3.1 and bior 3.3 wavelets have been applied to compress EEG data (Shinde et al., 2021). The bior 3.3 wavelet was utilised to compare performance in this study.

6.9.2.4 Symlets (sym)

Daubechies suggested the Symlets as modifications to the db family of wavelets. Both wavelet families have comparable characteristics. Additionally, it is referred to as the least asymmetric wavelet, defining a family of orthogonal wavelets. The sym4 wavelet has been used for feature extraction in the classification of motor-imagery tasks (Kevric and Subasi, 2017). Additionally, it has been used to categorise Alzheimer's patients using the sym3 wavelet (Fiscon et al., 2018), while the sym7 wavelet was successfully used to identify EEG sleep spindles. (Al-Salman et al., 2019). Moreover, the sym4 wavelet is capable of motor imagery classification using EEG signals (Xu et al., 2019). In this study, the sym4 wavelet was used for comparison.

6.9.2.5 Coiflets (coif)

The Coiflets are discrete wavelets developed by Ingrid Daubechies to have scaling functions with vanishing moments at the request of Ronald Coifman. The wavelet is almost symmetric and has been utilised in a wide variety of applications. For example, the coif4 wavelet was effectively used to decompose EEG signals and extract features from motor imagery tasks (Alomari et al., 2014b). Another study (Alomari et al., 2014a) implemented the same wavelet for real-time movement classification using EEG signals, and the result showed that the Coiflets wavelet provided the best

classification accuracy. Moreover, this wavelet family was used to detect alcoholism through EEG signal classification (Rodrigues et al., 2019). The computational performance of the coif2 wavelet was examined in the simulation.

6.10 Results and Findings

The purpose of this study was to examine the computational performance of DWT algorithms using various wavelet types. The filter bank technique, the lifting method, the matrix-vector method, and the proposed recursive method are among the algorithms that have been chosen. According to the simulation, the virtual input signal was generated using archival EEG data with varying sample sizes (identical to sampling rate). The input samples were updated on a first-in-first-out (FIFO) basis throughout each cycle. The simulation system showed a pseudo-real-time environment with this configuration.

6.10.1 Computation Time

From 10,000 executions, the average calculation time of the DWT algorithms was calculated. Each run was timed from the point at which the input was sent to the algorithms until the point at which the DWT coefficients were received. Tables 6.3 to 6.8 show the runtime results (in microseconds) versus different input lengths. Each table summarises the results of running six different DWT algorithms: MATLAB filter-bank DWT (dwt_func), MATLAB lifting-scheme DWT (lwt_func), customised filter-bank DWT (dwt_cust), customised lifting-scheme DWT (lwt_cust), and the proposed methods (Equations 6.1 and 6.8). The number of input lengths increases by the power of two. With regards to the findings, it can be observed that the computation times for MATLAB functions (dwt_func and lwt_func) are much longer than those for customised functions (dwt_cust and lwt_cust) and that the proposed method (Equation 6.8) is significantly faster. Similarly, the runtime results of Equation 6.1 are considerably greater than the rest. As a result, the remainder of the study will mainly focus on the customised DWT functions and Equation 6.8.

Table 6.3. Average computation time (in microsecond) of DWT using Haar wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	170.99	2957.21	2.23	1.00	0.20	1.65
32	167.50	2949.91	2.63	0.86	0.20	1.73
64	167.40	2987.84	2.78	0.93	0.44	1.84
128	167.75	3066.35	3.21	1.13	4.03	1.78
256	178.07	3070.29	4.25	1.25	12.94	1.83
512	184.07	3073.30	5.16	1.55	25.56	1.82
1024	186.16	3094.32	7.04	1.91	62.38	2.13
2048	273.35	3303.04	36.38	2.91	232.92	3.60
4096	315.30	3340.70	63.06	6.25	1953.35	5.25
8192	379.43	3397.15	89.95	11.69	8667.94	8.27
16384	476.52	3702.00	132.58	46.19	36872.22	14.77
32768	629.02	4105.80	222.35	67.72	139569.80	30.86

Table 6.4. Average computation time (in microsecond) of DWT using db2 wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	194.22	3203.88	2.20	0.95	0.23	1.64
32	192.21	3202.88	2.80	0.80	0.25	1.64
64	191.89	3244.73	2.82	0.92	0.54	1.82
128	191.78	3358.57	3.18	1.44	4.11	1.77
256	208.13	3344.37	4.48	2.14	15.46	1.80
512	208.36	3350.89	5.69	3.21	24.15	1.98
1024	211.98	3359.27	7.89	5.65	65.73	2.46
2048	311.18	3604.65	35.71	10.07	258.80	3.56
4096	356.85	3668.84	59.33	20.13	1862.59	5.82
8192	455.30	3920.45	87.08	40.02	8667.52	8.74
16384	558.99	4322.38	131.39	111.61	36793.78	15.50
32768	749.18	5004.81	211.75	206.05	139469.45	31.19

Table 6.5. Average computation time (in microsecond) of DWT using db4 wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	198.88	3628.00	2.57	1.39	0.19	1.77
32	195.35	3658.21	2.90	1.28	0.21	1.68
64	194.11	3727.24	3.06	1.73	0.44	1.51
128	195.86	3736.91	4.08	2.69	4.56	1.60
256	208.77	3747.88	4.73	4.52	12.78	1.62
512	213.80	3786.33	5.96	7.87	27.66	1.93
1024	215.27	3808.42	8.47	13.74	91.21	2.12
2048	314.93	4124.52	39.31	31.15	224.39	3.19
4096	368.51	4429.05	62.34	51.97	1876.14	5.42
8192	465.70	4990.53	100.51	103.59	8636.61	8.67
16384	597.20	5898.13	140.05	256.33	36603.93	15.05
32768	768.99	9119.09	224.73	478.67	138978.11	30.16

Table 6.6. Average computation time (in microsecond) of DWT using sym4 wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	194.71	3636.55	2.20	1.07	0.22	1.59
32	199.32	3659.25	2.86	0.92	0.20	1.68
64	194.41	3738.01	3.10	1.25	0.44	1.86
128	193.52	3748.21	3.95	2.05	5.70	1.70
256	209.09	3751.54	4.64	3.39	13.71	1.81
512	229.32	3770.92	6.02	5.46	18.53	1.95
1024	219.47	3802.29	8.36	10.02	86.73	2.50
2048	313.56	4032.07	39.70	19.04	274.88	3.50
4096	364.87	4226.69	65.19	38.18	1869.90	5.87
8192	453.86	4660.36	91.95	73.33	8620.78	8.36
16384	589.55	5439.54	137.22	185.54	36659.05	15.09
32768	765.70	6616.68	223.32	358.92	138918.69	31.34

Table 6.7. Average computation time (in microsecond) of DWT using bior 3.3 wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	244.71	3241.03	2.27	0.93	0.19	1.57
32	246.04	3269.44	2.98	0.82	0.20	1.65
64	243.81	3326.85	2.97	1.02	0.43	1.68
128	243.14	3356.12	3.34	1.63	4.33	1.73
256	260.97	3358.61	4.47	2.61	9.09	1.83
512	271.10	3380.36	5.81	4.60	25.89	2.08
1024	267.55	3399.02	7.79	8.07	74.50	2.36
2048	378.76	3669.56	40.30	15.23	225.31	3.50
4096	443.98	3753.53	64.76	30.61	1936.82	5.48
8192	524.36	4048.05	90.49	60.72	8677.26	9.24
16384	656.55	4608.22	134.51	145.42	36598.78	14.09
32768	845.28	7168.66	217.14	293.38	138506.40	30.82

Table 6.8. Average computation time (in microsecond) of DWT using coif2 wavelet.

Input Length	MATLAB Function		Customised Function		Proposed Method	
	dwt_func	lwt_func	dwt_cust	lwt_cust	Eq 6.1	Eq 6.8
16	193.73	4014.93	2.31	1.12	0.20	1.69
32	191.38	4038.54	3.15	1.06	0.22	1.68
64	196.55	4127.57	3.32	1.49	0.43	1.66
128	194.59	4109.62	4.36	3.08	2.91	1.71
256	208.33	4117.13	5.02	5.41	9.53	1.80
512	214.13	4161.94	6.43	9.95	17.25	1.94
1024	213.54	4165.19	9.24	18.44	73.48	2.27
2048	325.75	4510.85	41.73	36.30	231.50	3.43
4096	376.31	4815.83	64.42	72.34	1986.84	5.51
8192	473.18	5474.29	95.49	142.12	8634.64	8.84
16384	615.78	6549.33	140.29	336.58	36653.57	15.03
32768	789.77	8547.57	226.99	674.74	138323.36	30.48

Figures 6.3 to 6.5 illustrate the runtime of different wavelet algorithms from the previous tables: dwt_cust, lwt_cust, and Equation 6.8, respectively. Note that the input length ranges from 128 to 8192 samples, which corresponds to the commonly used sampling frequency and the typical bandwidth of EEG signals.

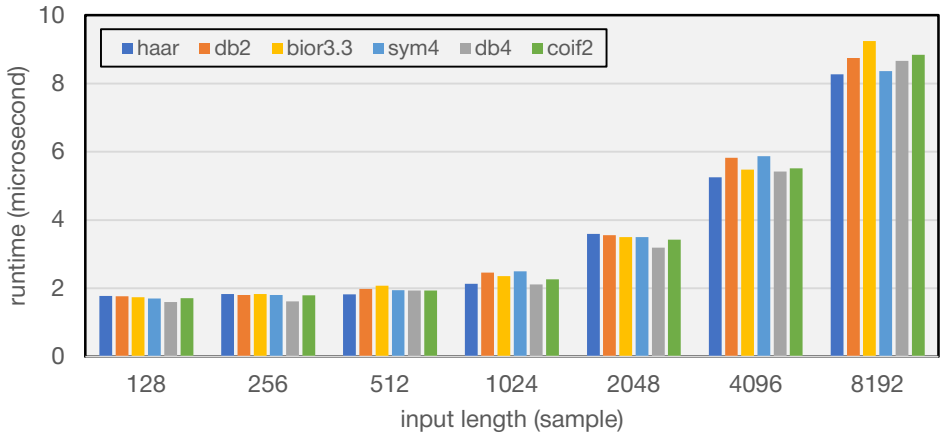

Figure 6.3. Runtime results of the proposed sliding dwt algorithm on different wavelets.

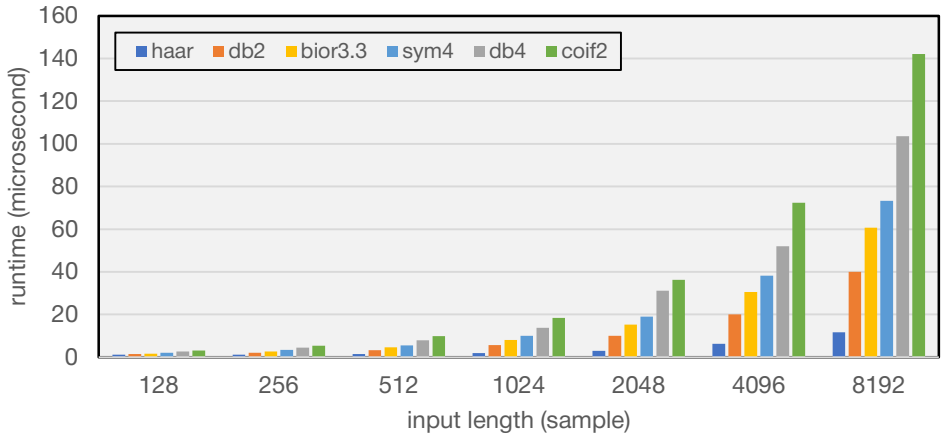

Figure 6.4. Runtime results of customised lwt algorithm on different wavelets.

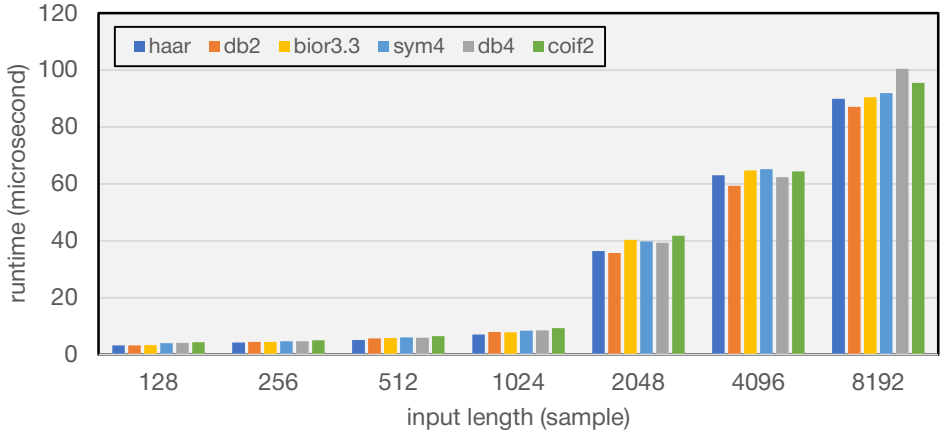

Figure 6.5. Runtime results of customised dwt algorithm on different wavelets.

6.10.2 Speedup Ratio

A speedup ratio is another index that can be used to evaluate the performance of an algorithm related to numerical calculation. According to the runtime results shown in Tables 6.3 to 6.8, due to the excessive results produced by MATLAB functions, we further analyse the computational efficiency of the proposed recursive method by comparing it to the runtimes of customised DWT functions only, in terms of the speedup ratio. The ratio was determined by dividing the time required to execute customised functions by the time required to run the recursive algorithm (Equation 6.8). The results of the customised filter bank and lifting scheme methods are shown in Figures 6.6 and 6.7, respectively. Additionally, each plot combines the results of six wavelet functions.

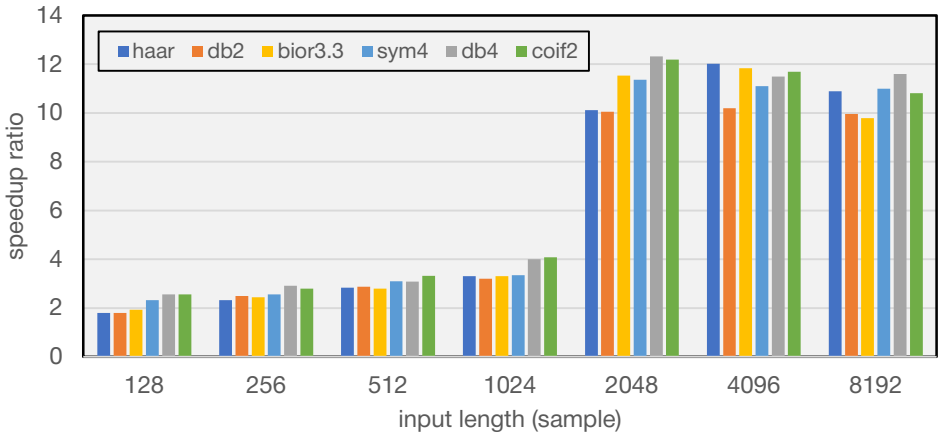

Figure 6.6. Illustrating the speedup ratio of different input lengths and wavelets calculated from the runtimes of the customised dwt function divided by the time of the proposed recursive algorithm.

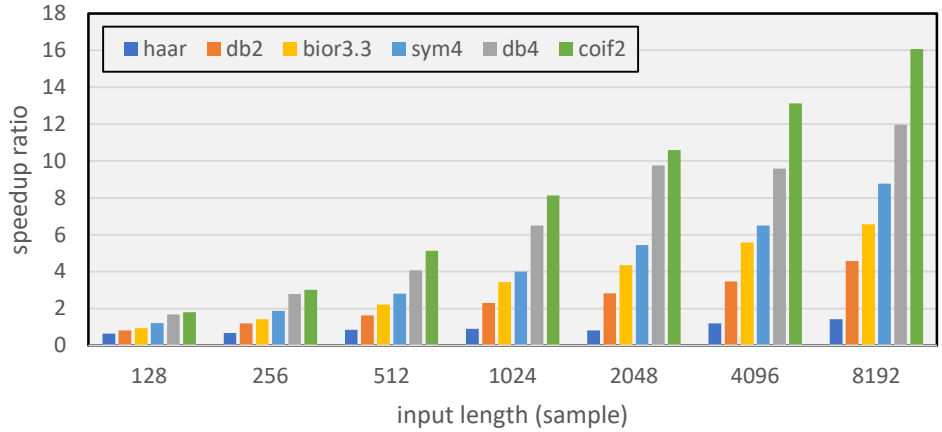

Figure 6.7. Illustrating the speedup ratio of different input lengths and wavelets calculated from the runtimes of the customised lwt function divided by the time of the proposed recursive algorithm.

6.11 Discussion

The proposed recursive method outperforms the previous DWT algorithms in runtime, as shown in Table 6.3 through Table 6.8. As stated earlier in Table 6.1, the recursive method computes two pairs of a new DWT coefficient rather than all coefficients. Due to the convolutional nature of the filer bank algorithm, the overall complexity costs the maximum operation $\mathcal{O}(N^2)$ resulting in the slowest runtime. In comparison, employing a lifting structure reduces the complexity of the computation by about half (Liao et al., 2004); as a result, it is frequently utilised for real-time processing. Additionally, when using built-in MATLAB functions, mainly the dwt function, it is apparent that the built-in function has a more significant runtime than the customised function. This is because MATLAB calculates the DWT coefficients using the wavelet toolbox, which is often not completely optimised, and most toolboxes need some time to initialise (Misiti et al., 1996). Note that the plots of the computation time and the speedup ratio of the MATLAB functions achieved are also provided in Appendix D.

As shown in Figures 6.3 and 6.5, the runtimes of the modified filter bank approach and the proposed recursive method are not substantially different among wavelets. This is due to the fact that both methods need just filtering coefficients for direct convolution/multiplication. While the customised lifting scheme clearly shows the difference, a more complicated lifting structure or a greater number of filtering coefficients results in a longer runtime. Surprisingly, the suggested method's runtime progressively rises as the input length grows, even though numerical analysis indicates an exact number of operations of 14. Therefore no changes in runtime results were anticipated. This occurred due to MATLAB's data transfer and initialisation, for which the system needs time in proportion to the input size for memory allocation. Using a low-level programming language such as C/C++, on the other hand, should result in improved performance (Fourment and Gillings, 2008). Along with the DWT algorithm's performance, the matrix-vector approach has the longest calculation time, as expected. However, the author advises against using this technique due to its inefficiency in multiplying unless applied to hardware acceleration such as GPUs.

Regarding the speedup findings, as shown in Figure 6.6, the ratio of utilising the customised dwt function tends to rise with input length, and there is no significant

variation among wavelets. Similarly, when the input length grows, the speedup ratio of the customised lwt function increases. The findings, however, vary significantly across wavelets. For instance, the haar wavelet produces the lowest ratio compared to the others, while the coif2 produces the highest ratio. This is because the coif wavelet and the haar wavelet have a simpler and more complicated lifting structure. As a result, the more complicated the structure, the greater the computational complexity.

When DWT algorithms are used for real-time processing, it is clear that the proposed recursive approach may be used to analyse EEG signals in this environment since the majority of real-time EEG applications utilise sampling rates less than 2 kHz (Galatenko et al., 2012; Majmudar and Morshed, 2016; Wang et al., 2015; Zandi et al., 2010). However, the method has a restriction in that the input must have a one-element-shift basis. When the processing window is moved by more than one sample, the algorithm must be modified to accommodate the calculation. Correspondingly, each moving step adds to the complexity of the algorithm. Additional study on the DWT method based on a sliding window mechanism is required to solve this issue. Modifications to such algorithms may increase performance.

6.12 Chapter Summary

This chapter shows how to handle real-time signal processing by using recursive and parallel computing techniques. The DWT algorithms, both the first and second generations, were chosen for investigation based on a sliding-window method. Compared to other DWT techniques, modifying the filter-bank algorithm through a recursive methodology needs a minimal amount of memory and is less complicated. The findings demonstrate that the recursive approach has the shortest runtime and the most outstanding speedup ratio for all input lengths. Additionally, the proposed technique may be used for a wide variety of wavelet functions without compromising performance. Nonetheless, additional research into the DWT algorithms for a multiple-shift system may aid in speeding up processing. The highlights and future work will be further discussed in the next chapter.

The next chapter summarises all contributions, findings, and recommendations made throughout this thesis.

6.13 References

Al-Salman, W., Li, Y. & Wen, P. 2019. Detecting sleep spindles in EEGs using wavelet fourier analysis and statistical features. *Biomedical Signal Processing and Control,* 48**,** 80-92.

Alomari, M. H., Abubaker, A., Turani, A., Baniyounes, A. M. & Manasreh, A. 2014a. EEG mouse: A machine learning-based brain computer interface. *Int. J. Adv. Comput. Sci. Appl,* 5**,** 193-198.

Alomari, M. H., Awada, E. A., Samaha, A. & Alkamha, K. 2014b. Wavelet-based feature extraction for the analysis of EEG signals associated with imagined fists and feet movements. *Computer and Information Science,* 7**,** 17.

Alves, D. K., Costa, F. B., Ribeiro, R. L. D. A., Neto, C. M. D. S. & Rocha, T. D. O. A. 2017. Real-Time Power Measurement Using the Maximal Overlap Discrete Wavelet-Packet Transform. *IEEE Transactions on Industrial Electronics,* 64**,** 3177-3187.

Alyasseri, Z. a. A., Khader, A. T., Al-Betar, M. A., Abasi, A. K. & Makhadmeh, S. N. EEG signal denoising using hybridizing method between wavelet transform with genetic algorithm. Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019, 2021. Springer, 449-469.

Binglian, X. & Qiusheng, Z. Image denoising based on a new symmetrical second-generation wavelet. 2009 International Conference on Image Analysis and Signal Processing, 11-12 April 2009 2009. 1-4.

Chen, D., Wan, S., Xiang, J. & Bao, F. S. 2017. A high-performance seizure detection algorithm based on Discrete Wavelet Transform (DWT) and EEG. *PloS one,* 12**,** e0173138.

Chen, J., Jiang, D. & Zhang, Y. 2019. A common spatial pattern and wavelet packet decomposition combined method for EEG-based emotion recognition. *Journal of Advanced Computational Intelligence and Intelligent Informatics,* 23**,** 274-281.

Cheng, F.-H. & Chen, Y.-L. 2006. Real time multiple objects tracking and identification based on discrete wavelet transform. *Pattern Recognition,* 39**,** 1126-1139.

Da Silveira, T. L. T., Kozakevicius, A. J. & Rodrigues, C. R. 2016. Automated drowsiness detection through wavelet packet analysis of a single EEG channel. *Expert Systems with Applications,* 55**,** 559-565.

Daubechies, I. & Sweldens, W. 1998. Factoring wavelet transforms into lifting steps. *Journal of Fourier analysis and applications,* 4**,** 247-269.

Demirel, H. & Anbarjafari, G. 2011. IMAGE Resolution Enhancement by Using Discrete and Stationary Wavelet Decomposition. *IEEE Transactions on Image Processing,* 20**,** 1458-1460.

Desai, K. D. & Sankhe, M. S. A real-time fetal ECG feature extraction using multiscale discrete wavelet transform. 2012 5th International Conference on BioMedical Engineering and Informatics, 16-18 Oct. 2012 2012. 407-412.

Donoho, D. L. Nonlinear wavelet methods for recovery of signals, densities, and spectra from indirect and noisy data. In Proceedings of Symposia in Applied Mathematics, 1993. Citeseer.

Ele, P., Ntsama, P. E. & Tonye, E. Modified Lifting Scheme Characterization for the Compression of EMG Signals. 2009 Fifth International Conference on Signal Image Technology and Internet Based Systems, 29 Nov.-4 Dec. 2009 2009. 73-80.

Fiscon, G., Weitschek, E., Cialini, A., Felici, G., Bertolazzi, P., De Salvo, S., Bramanti, A., Bramanti, P. & De Cola, M. C. 2018. Combining EEG signal processing with supervised methods for Alzheimer's patients classification. *BMC medical informatics and decision making,* 18**,** 1-10.

Fourment, M. & Gillings, M. R. 2008. A comparison of common programming languages used in bioinformatics. *BMC bioinformatics,* 9**,** 1-9.

Freitas, D. R., Inocêncio, A. V., Lins, L. T., Alves, G. J. & Benedetti, M. A. A parallel implementation of the discrete wavelet transform applied to real-time EEG signal filtering. XXVI Brazilian Congress on Biomedical Engineering, 2019. Springer, 17-23.

Galatenko, V. V., Livshitz, E. D., Podolskii, V. E., Chernorizov, A. M. & Zinchenko, Y. P. 2012. Automated real-time classification of psychological functional state based on discrete wavelet transform of EEG data. *International Journal of Applied Mathematics,* 25**,** 871-882.

Haar, A. 1910. Zur theorie der orthogonalen funktionensysteme. *Mathematische Annalen,* 69**,** 331-371.

Hatsuda, H. 2012. Robust smoothing of quantitative genomic data using second-generation wavelets and bivariate shrinkage. *IEEE transactions on biomedical engineering,* 59**,** 2099-2102.

Hsia, C.-H. & Guo, J.-M. 2014. Efficient modified directional lifting-based discrete wavelet transform for moving object detection. *signal Processing,* 96**,** 138-152.

Jana, P. & Phadikar, A. 2020. Low-Power FPGA-Based Hardware Implementation of Reversible Watermarking Scheme for Medical Image. *Computational Advancement in Communication Circuits and Systems.* Springer.

Kevric, J. & Subasi, A. 2017. Comparison of signal decomposition methods in classification of EEG signals for motor-imagery BCI system. *Biomedical Signal Processing and Control,* 31**,** 398-406.

Khalaf, A., Sejdic, E. & Akcakaya, M. 2019. Common spatial pattern and wavelet decomposition for motor imagery EEG- fTCD brain-computer interface. *Journal of Neuroscience Methods,* 320**,** 98-106.

Khatun, S., Mahajan, R. & Morshed, B. I. 2016. Comparative Study of Wavelet-Based Unsupervised Ocular Artifact Removal Techniques for Single-Channel EEG Data. *IEEE Journal of Translational Engineering in Health and Medicine,* 4**,** 1-8.

Liao, H., Mandal, M. K. & Cockburn, B. F. 2004. Efficient architectures for 1-D and 2-D lifting-based wavelet transforms. *IEEE Transactions on Signal Processing,* 52**,** 1315-1326.

Majmudar, C. A. & Morshed, B. I. 2016. Autonomous OA removal in real-time from single channel EEG data on a wearable device using a hybrid algebraic-wavelet algorithm. *ACM Transactions on Embedded Computing Systems (TECS),* 16**,** 1-16.

Misiti, M., Misiti, Y., Oppenheim, G. & Poggi, J.-M. 1996. Wavelet toolbox. *The MathWorks Inc., Natick, MA,* 15**,** 21.

Murugappan, M., Rizon, M., Nagarajan, R., Yaacob, S., Zunaidi, I. & Hazry, D. Lifting scheme for human emotion recognition using EEG. 2008 International Symposium on Information Technology, 26-28 Aug. 2008 2008. 1-7.

Ngui, W. K., Leong, M. S., Hee, L. M. & Abdelrhman, A. M. Wavelet analysis: mother wavelet selection methods. Applied mechanics and materials, 2013. Trans Tech Publ, 953-958.

Phadikar, A., Maity, G. K., Chiu, T.-L. & Mandal, H. 2019. FPGA implementation of lifting-based data hiding scheme for efficient quality access control of images. *Circuits, Systems, and Signal Processing,* 38**,** 847-873.

Radhakrishnan, P. & Themozhi, G. 2020. FPGA implementation of XOR-MUX full adder based DWT for signal processing applications. *Microprocessors and Microsystems,* 73**,** 102961.

Rodrigues, J. D. C., Filho, P. P. R., Peixoto, E., N, A. K. & De Albuquerque, V. H. C. 2019. Classification of EEG signals to detect alcoholism using machine learning techniques. *Pattern Recognition Letters,* 125**,** 140-149.

Saini, S. & Dewan, L. 2018. Performance Comparison of First Generation and Second Generation Wavelets in the Perspective of Genomic Sequence Analysis. *International Journal of Pure and Applied Mathematics,* 118**,** 417-442.

Sairamya, N. J., Subathra, M. S. P., Suviseshamuthu, E. S. & Thomas George, S. 2021. A new approach for automatic detection of focal EEG signals using wavelet packet decomposition and quad binary pattern method. *Biomedical Signal Processing and Control,* 63**,** 102096.

Shinde, A. N., Nalbalwar, S. L. & Nandgaonkar, A. B. 2021. Impact of optimal scaling coefficients in bi-orthogonal wavelet filters on compressed sensing. *International Journal of Pervasive Computing and Communications*.

Sweldens, W. 1998. The lifting scheme: A construction of second generation wavelets. *SIAM journal on mathematical analysis,* 29**,** 511-546.

Taha, D. B., Taha, T. B. & Al Dabagh, N. B. 2020. A comparison between the performance of DWT and LWT in image watermarking. *Bulletin of Electrical Engineering and Informatics,* 9**,** 1005-1014.

Wang, N., Lu, P., Zhang, L., Li, S. & Hu, H. Accelerated Rendering and Fast Reconstruction of EEG Data in Real-Time BCI. Proceedings of the 2015 Chinese Intelligent Automation Conference, 2015. Springer, 61-75.

Xu, B., Zhang, L., Song, A., Wu, C., Li, W., Zhang, D., Xu, G., Li, H. & Zeng, H. 2019. Wavelet Transform Time-Frequency Image and Convolutional Network-Based Motor Imagery EEG Classification. *IEEE Access,* 7**,** 6084-6093.

Ye, L. & Hou, Z. 2015. Memory Efficient Multilevel Discrete Wavelet Transform Schemes for JPEG2000. *IEEE Transactions on Circuits and Systems for Video Technology,* 25**,** 1773-1785.

Zandi, A. S., Javidan, M., Dumont, G. A. & Tafreshi, R. 2010. Automated real-time epileptic seizure detection in scalp EEG recordings using an algorithm based on wavelet packet transform. *IEEE Transactions on Biomedical Engineering,* 57**,** 1639-1651.

Zarei, A. & Asl, B. M. 2021. Automatic seizure detection using orthogonal matching pursuit, discrete wavelet transform, and entropy based features of EEG signals. *Computers in Biology and Medicine,* 131**,** 104250.

Zhou, C., Zhang, G., Yang, Z. & Zhou, J. 2020. A Novel Image Registration Algorithm Using Wavelet Transform and Matrix-Multiply Discrete Fourier Transform. *IEEE Geoscience and Remote Sensing Letters***,** 1-5.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

This chapter summarises the work accomplished throughout the thesis, including the research, highlights, fresh insights, and limits, and concludes with a future direction for the current study.

7.1 Executive Summary

7.1.1 Evaluation of Real-Time Feature Extraction Methods

The first section of Chapter 3 evaluated possible feature extraction techniques for real-time BCI applications. To summarise, the thesis shows the distinction between time and frequency domain methods for a two-class motor imaginary BCI. The preliminary results indicate that employing various feature extraction techniques provided no significant change in classification accuracy but resulted in a substantial difference in computation time. For instance, as illustrated in Table 3.1 and Figure 3.5, the FFT technique takes much more time than others without providing a benefit in terms of accuracy. On the other hand, the template matching technique is almost 10 times faster than the FFT technique. According to the results, the findings demonstrate the need to consider computing time when choosing algorithms for real-time applications and the usefulness and benefits of utilising open-access data archives for comparative BCI research. The work also highlights the benefit of groups providing high-quality data to such archives. It allows not just replication of prior studies but facilitates the continuation of research in laboratories when experimented programmes employing face-to-face research cannot process due to extended circumstances (e.g. 2020-2121 COVID-19 workplace restrictions).

7.1.2 Augmentation of Feature Extraction Methods Based on Parallel Computing Scheme

Parallel computing was later considered in the second half of Chapter 3 with the purpose of speeding up feature extraction processing. This section discusses the benefits of parallel computing in signal processing.

Increasing the computing time via a parallel processing method is theoretically feasible and flexible enough for the real-time processing of a high volume of biosignal data. However, in order to achieve decreased system latency, it is necessary to optimise both hardware and software when used in a real-time context. The route chosen to achieve this within a parallel-architecture based BCI prototype was with an OpenCL library for signal processing.

7.1.3 Modification of Feature Extraction Algorithms for Real-Time Signal Processing

As summarised above, the evaluation of feature extraction methods for real-time BCIs based on discrete Fourier transform provided similar classification results but so computationally expensive that it makes real time processing almost impossible for the majority of functional BCI applications. Chapter 4 addresses this issue by introducing an alternative named “enhanced sliding discrete Fourier transform (eSDFT).” The eSDFT technique is best suited for processing time-varying physiological data when real-time or near-real-time processing is required. The suggested technique for the multi-sample shift was created using a reformulated momentary matrix transformation. According to the computational complexity study, the hybrid technique outperformed the traditional FFT method.

Additionally, with extra calculation, a windowing function may be applied to the eSDFT technique to mitigate the impact of aperiodicity. Thus, the proposed method has the potential to be used for large-scale, real-time signal processing that makes use of numerous compute units for distributed computing. Note that although the proposed algorithm can significantly improve the computation time, there are potential dangers in using the method when the quality of the input data is relatively low (e.g. small signal-to-noise ratio and artefacts contained). To ensure the analysis is based on high-

quality data, pre-processing and signal conditioning such as resampling, smoothing, and aligning signals should be performed prior to using this approach.

7.1.4 Parallel Computing for Time-Frequency Feature Extraction

Numerous BCI research analyses bio-signals in the frequency domain using the Fourier transform. However, due to the great temporal resolution of EEG signals, a straightforward Fourier transform cannot utilise this feature, often resulting in the loss of time information. As a result, another technique for feature extraction, which offers both time and frequency domains, was investigated. Chapter 5 discusses the potential of developing a parallel computing strategy for processing a technique for extracting time-frequency-domain features. The discrete wavelet transform was chosen along with the heterogeneous computing idea to illustrate the usage of a parallel computing method by altering the original transformation based on matrix multiplication.

Furthermore, a signal extension mode was included in the proposed matrix form to mitigate the impact of border distortion. As illustrated in Table 5.2, the results show that using the proposed parallel computing technique can gain the speedup ratio up to 90 times. These findings demonstrate the efficiency of hardware acceleration in terms of performing large computing workloads. Moreover, by expressing the computation as a matrix operation, the proposed processing method may be used to accelerate any signal processing algorithm. However, the proposed algorithm remains room for improvement, primarily when used in real time, as it can duplicate calculations.

7.1.5 Real-Time Implementation Based on Feature Extraction in Time-Frequency Domain

Chapter 6 demonstrates how to use the novel recursive techniques presented in Chapter 4 and the parallel computing methods presented in Chapter 5 to handle real-time signal processing. The first and second generations of discrete wavelet transform algorithms were selected for study based on a sliding window basis. The modification of the traditional DWT algorithm has been proposed to reduce redundancy in the calculation of coefficients. Compared to other DWT methods, changing the filter-bank algorithm recursively requires less memory and is less complex. The findings show that, for all

input lengths, the recursive method provides the lowest runtime and the highest speedup ratio. Additionally, the suggested method is capable of being applied to a broad range of wavelet functions without sacrificing performance.

Moreover, based on the performance achieved, the proposed algorithm can be used in several parts of real-time BCI applications, such as for feature extraction. An example of this can be drawn associated with the results illustrated in Figures 6.3 to 6.5, in which the proposed DWT approach requires less runtime than other algorithms. This means, when using the recursive DWT formula, the BCI system can increase sampling frequency in order to acquire higher resolution of EEG data without affecting the computation time.

7.2 Highlights

As stated in the first chapter, the present thesis contributes to knowledge in the following ways:

1. Evaluation of time- and frequency-domain feature extraction techniques in multichannel EEG recordings using a pseudo-real-time system has been done.
2. The state-of-the-art method for real-time feature extraction has been modified to include a Fourier transform approach.
3. The current method for real-time feature extraction has been modified to support a parallel computing strategy based on a discrete wavelet transform approach.
4. A stimulation of real-time BCI system for neurorehabilitation has been developed to examine the proposed algorithms utilising high-performance computer technology.
5. Different feature extraction techniques have been accomplished by detecting signal amplitude and frequency variations with low latency based on multichannel recordings.

7.3 Future Work

7.3.1 Developing Real-Time Processing Algorithms

The majority of the findings in this thesis emphasise the capability of any feature extraction technique to be implemented or modified for real-time BCIs. Nevertheless, feature extraction is only one component of a signal processing pipeline. The primary techniques used in the thesis, including recursiveness and parallel distributed processing, provide the potential for further development of effective BCI systems that utilise multichannel EEG or other biosignals.

Further research on system latency in other areas of signal processing, such as signal acquisition, classification algorithms, and output feedback, may aid in developing real-time applications. Furthermore, additional study of the computing performance of the proposed feature extraction techniques based on the Fourier transform and discrete wavelet transform may be of interest. This may involve benchmarking against other current algorithm designs or a new structure of sliding patterns. Additional research areas may include further optimising the suggested algorithms' complexity and floating-point operations when applied to real-world situations. Additionally, the techniques utilised to alter the algorithms, such as recursion and parallelism, may be extended to different feature extraction and classification methods. Finally, it would be beneficial to modify the proposed algorithms in order to enable operation on additional hardware acceleration devices with a different design and architecture.

7.3.2 Integrating New Technology

Apart from algorithm development, future technology will also be needed to improve the real-time performance of BCI applications. To demonstrate this, cloud-based processing and edge computing via the next-generation communication networks (i.e., 5G), a distributed computing paradigm that brings computation and data storage closer to the sources of data, can help accelerate signal processing. This cutting-edge technology can bring real-time multidimensional BCI systems. Moreover, improving electrode systems, enhancing noise reduction conditioning, and extending error

detection mechanisms will be required, leading to autonomous and adaptable optimisation.

7.3.3 BCI Systems Based on User-Centred Design

The objective of user-centred design is to create something useful for people. In this case, user research is critical. It guarantees that consumers find the new item simple to use, that they find the technology acceptable to use at home and outside, and that it creates no adverse events. The major limitation of this thesis is the inability to apply the findings to actual use cases (with patients) due to COVID-19 workplace restrictions, which resulted in the inability to conduct face-to-face experiments and therefore explore the concept of more real-world applications. Therefore, the work is not able to judge its functionality, compatibility, and users' feelings. However, public participation could be a key to understanding where potential users needs lie and what types of communication and functions they want to use. This can systematically improve the direction of BCI development in order to deliver the best solution. To create this engagement, stakeholder invited workshops, intensive interviews with focused groups and user communities are needed. However, there are limited studies to date associated with BCI user communities, and there is a lack of official standards or regulations for BCI devices. The new BCI industry and larger cooperation with big companies may help accelerate the process of community development.

7.4 Data and Software Availability

According to the data sets used in this thesis, there are 2 public archives including a) the data set IV of BNCI Horizon 2020 (http://bnci-horizon-2020.eu/database/data-sets) used in Section 3.3 and b) the EEG data set provided by the Swartz Center for Computational Neuroscience (https://doi.org/10.18112/openneuro.ds002680.v1.2.0) used in Section 3.4. In addition, the EEG data set related to motor imagery tasks used in Chapters 5 and 6 belongs to the Centre of Excellence in Rehabilitation Engineering, Department of Biomedical Engineering, University of Strathclyde, which is available upon request.

Along with the data sets, a demo of e-SDFT algorithm can be download from https://pureportal.strath.ac.uk/en/datasets named “Demo of Enhanced Sliding DFT (eSDFT) Algorithm” for the interested reader.

APPENDIX A

PSEUDO-REAL-TIME SIMULATION SETUP

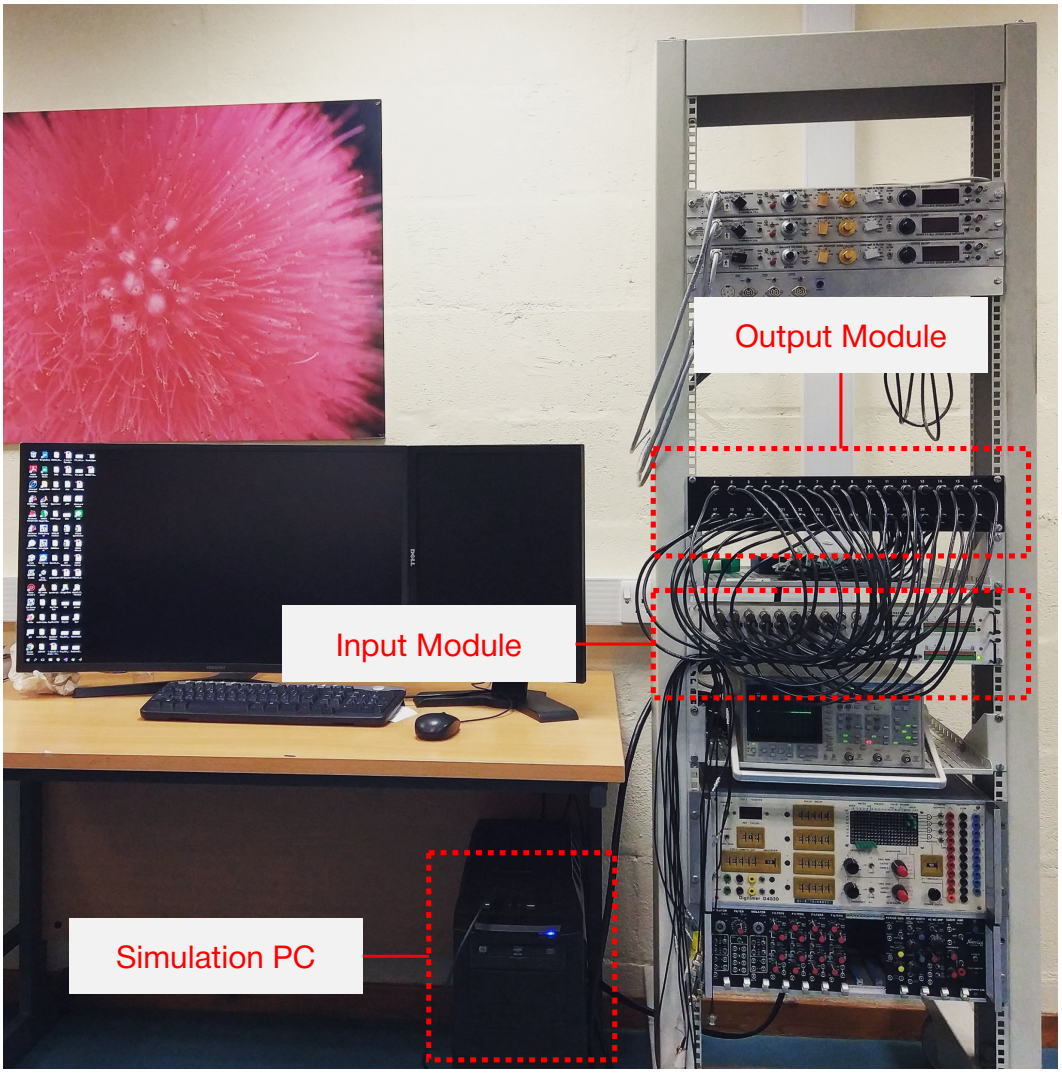

Figure A.1. Showing the simulation setup used in the study of feature extraction acceleration using parallel computing. The output module generates the 32-channel EEG signal to the input module which acquires and processes the signal based on parallel computing pipeline.

APPENDIX B

ALGORITHM CODE EXAMPLE

B.1 MATLAB Script for eSDFT Algorithm

```
% A MATLAB script demonstrating the implementation of the enhanced sliding
% discrete Fourier transform (eSDFT) algorithm, which is an improvement of
% the traditional sliding DFT (SDFT) algorithm. Based on a momentary matrix
% transformation, the eSDFT offers the speedup and more accuracy than the
% original SDFT.
%
% Copyright 2021
% Jetsada Arnin

clc; clear all; close all;
rng(7);

% Parameter setup for simulation of sliding DFT
n_DFT = 128;      % Number of DFT coefficients
n_channel = 2;    % Number of input channels
m = 4;           % Number of samples shifted between sequences
n_iter = 200;     % Number of total iterations
n_mMFT = 50;      % Iteration intervals to update the accurate coefficients

% Signal generation
x = zeros(n_DFT,n_channel);       % Generate a blank matrix of the input
signal
x = single(x);   % Convert the input to single-precision floating-point

% Initilization of DFT coefficients
Y_eSDFT = zeros(n_DFT,n_channel); % Initialise DFT coefficients of eSDFT
system
Y_SDFT = zeros(n_DFT,n_channel);  % Initialise DFT coefficients of SDFT
system

% Start simulation
for i = 1:n_iter

    % Update the current sequence xn with a random sample (FIFO basis)
    xn = [x(2:end,:); randn(1,n_channel)];

    % Calculate the actual DFT coefficients for comparing with the eSDFT
    Yn_actual = fft(xn);

    % Calculate the tradditional sliding DFT and its estimation error
    Yn_SDFT = eSDFT(xn,x,Y_SDFT,1);
    rmse_SDFT(i) = sqrt(immse(Yn_SDFT,Yn_actual));
```

```
    % Calculate the sliding DFT using eSDFT algorithm
    % If the supporting unit is available to compute the accurate DFT,
    % calculate the accurate DFT (FFT method) and keep the current input
    % for an update in the next m-iteration
    if (mod(i,n_mMFT)==0 && i>=n_mMFT)
        Ym = fft(xn);
        xm = xn;
    end

    % If the supporting unit is ready to update the accurate coefficients,
    % estimate the DFT coefficients using m-MFT method (m-sample shift)
    if (mod(i,n_mMFT)==m && i>=n_mMFT)
        Yn_eSDFT = eSDFT(xn,xm,Ym,m);
        rmse_eSDFT(i) = sqrt(immse(Yn_eSDFT,Yn_actual));

    % If the supporting unit is not ready to update or still running,
    % estimate the DFT coefficients using MFT method (1-sample shift)
    else
        Yn_eSDFT = eSDFT(xn,x,Y_eSDFT,1);
        rmse_eSDFT(i) = sqrt(immse(Yn_eSDFT,Yn_actual));
    end

    % Update the recursive parameters for the next iteration
    x = xn;
    Y_eSDFT = Yn_eSDFT;
    Y_SDFT = Yn_SDFT;
end

% Convert the cumulative error to log-scale
rmse_SDFT = 10*log10(rmse_SDFT);
rmse_eSDFT = 10*log10(rmse_eSDFT);

% Location of the supporting unit starting the execution
loc_start = n_mMFT:n_mMFT:n_iter;
loc_finish = n_mMFT+m:n_mMFT:n_iter;

% Plot the cumulative error over iteration
p(1) = plot(rmse_SDFT,':'); hold on;
p(2) = plot(rmse_eSDFT,'-');
p(3) = plot(loc_start,rmse_eSDFT(loc_start),'ks');
p(4) = plot(loc_finish,rmse_eSDFT(loc_finish),'k^');

xlabel('Iteration number');
ylabel('RMSE (dB)');
xlim([0 n_iter]);
legend(p,'Cumulative error from SDFT','Cumulative error from eSDFT', ...
        'Supporting unit starts execution','Supporting unit provides
update', ...
        'location','se');
title('Simulation of eSDFT Algorithm');
```

APPENDIX C

SUPPLEMENTARY RESULTS OF CHAPTER 5

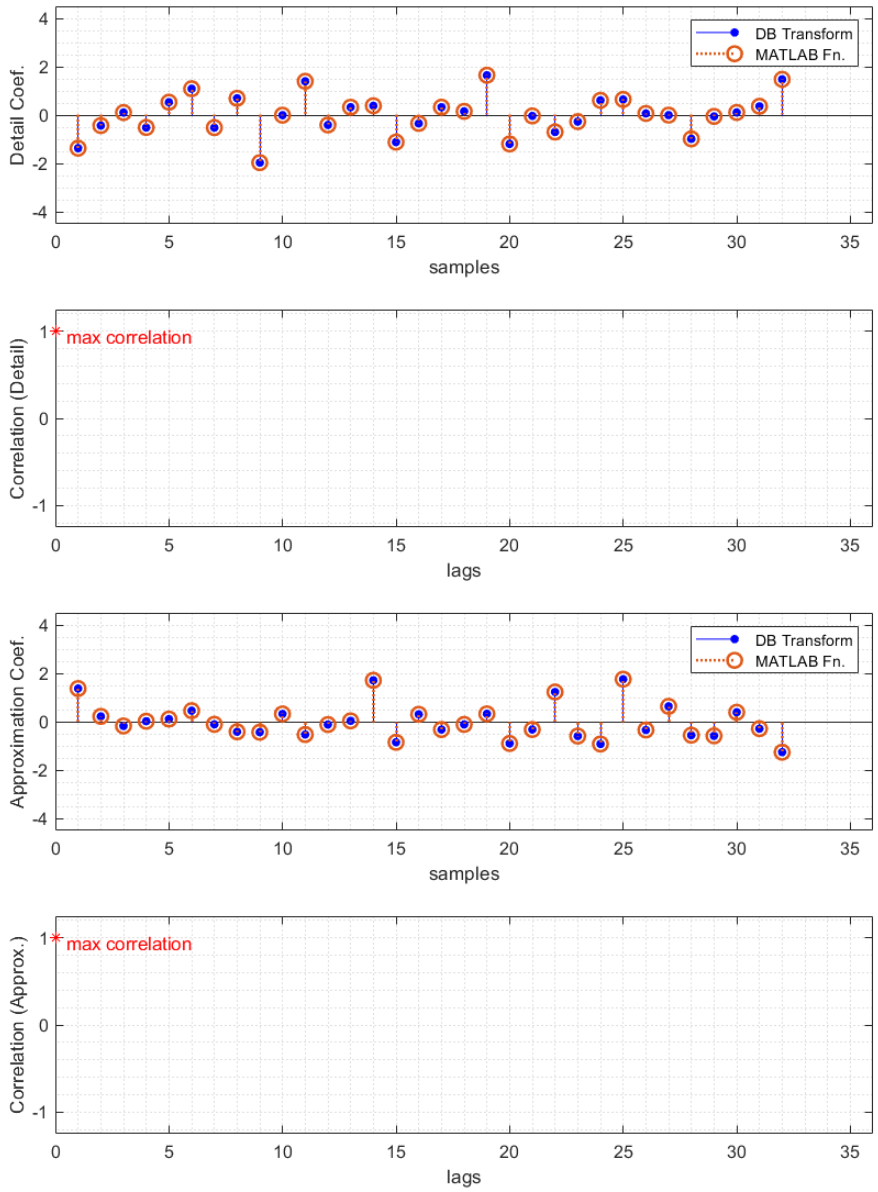

Figure C.1. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using DWT based on the haar wavelet and the signal extension mode of "periodic padding (ppd)". The cross-correlation values between the output coefficients are also provided.

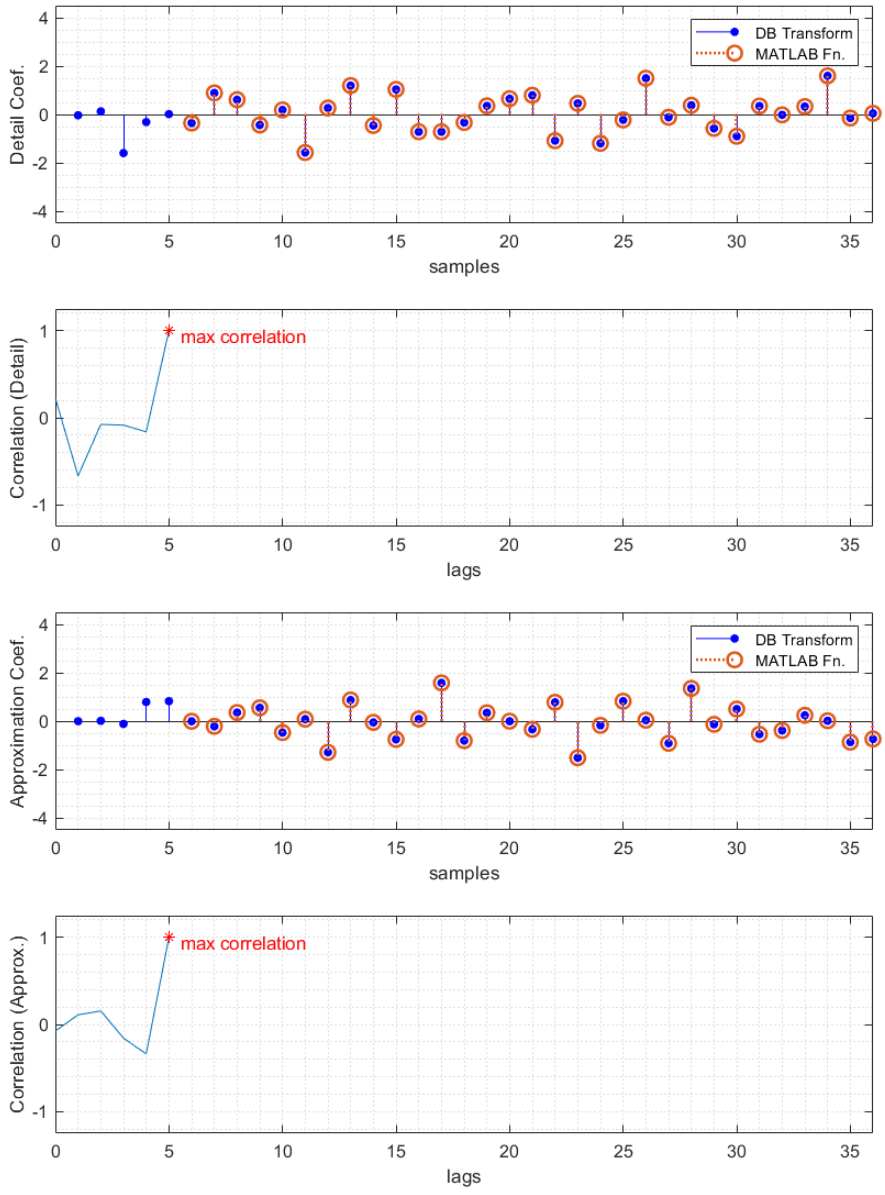

Figure C.2. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using Daubechies wavelet transform based on the coif2 wavelet and the signal extension mode of "zero padding (zpd)". In addition to each coefficient output, the normalised cross-correlation between the coefficients attained from the proposed technique and the MATLAB DWT function was calculated to locate the lag point at the maximum correlation value.

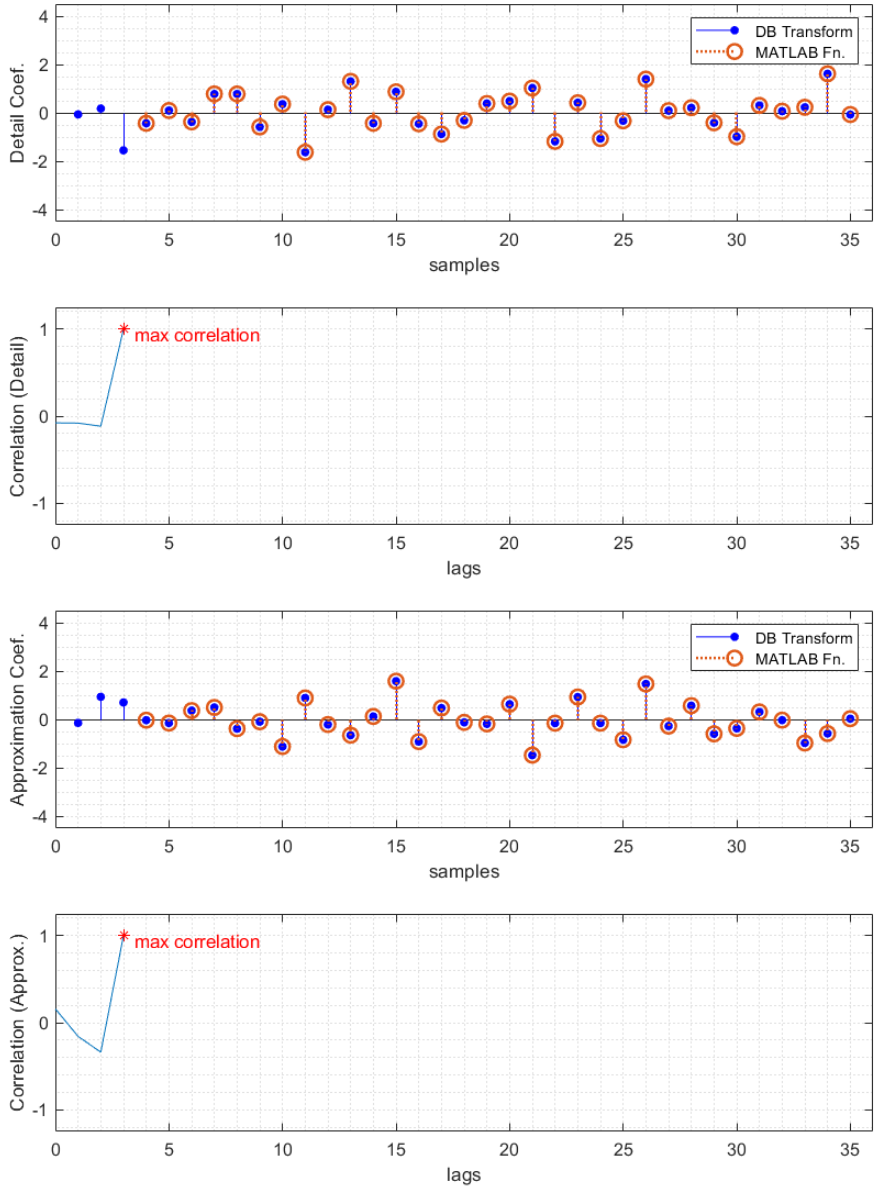

Figure C.3. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using Daubechies wavelet transform based on the sym4 wavelet and the signal extension mode of "zero padding (zpd)". In addition to each coefficient output, the normalised cross-correlation between the coefficients attained from the proposed technique and the MATLAB DWT function was calculated to locate the lag point at the maximum correlation value.

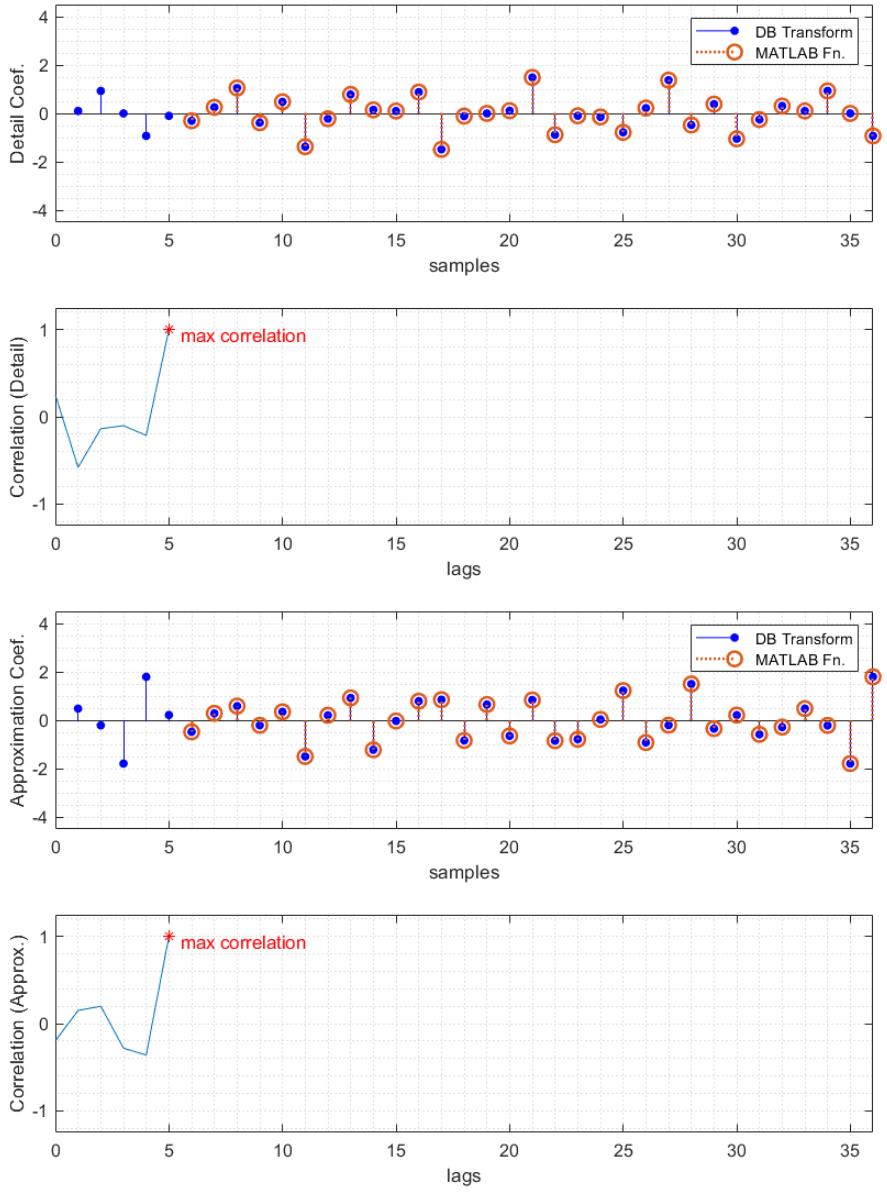

Figure C.4. Illustrating the approximation coefficients and the detail coefficients achieved from 64-sample input using Daubechies wavelet transform based on the rbio5.5 wavelet and the signal extension mode of "periodic padding (ppd)". In addition to each coefficient output, the normalised cross-correlation between the coefficients attained from the proposed technique and the MATLAB DWT function was calculated to locate the lag point at the maximum correlation value.

APPENDIX D

SUPPLEMENTARY RESULTS OF CHAPTER 6

D.1 Additional Results of Computation Time

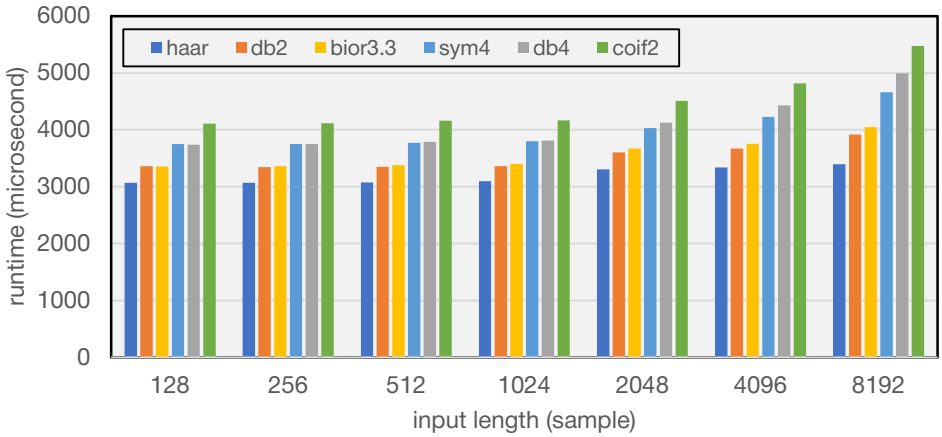

Figure D.1. Displaying runtime results of MATLAB built-in dwt function on different wavelts.

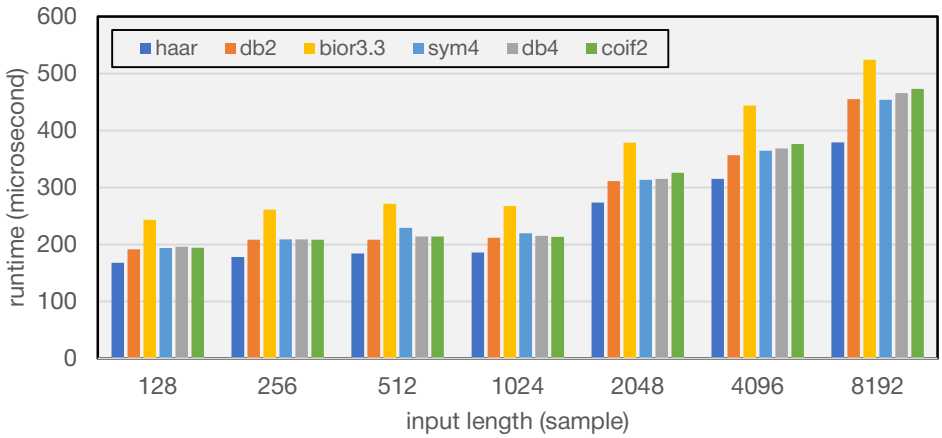

Figure D.2. Displaying runtime results of MATLAB built-in lwt function on different wavelts.

D.2 Additional Results of Speedup Ratio

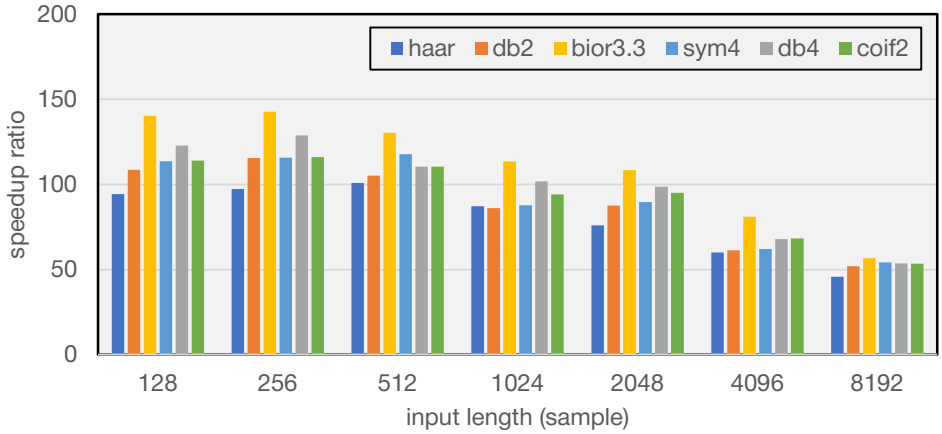

Figure D.4. Illustrating the speedup ratio of different input lengths and wavelets calculated from the runtimes of the MATLAB built-in dwt function divided by the time of the proposed recursive algorithm.

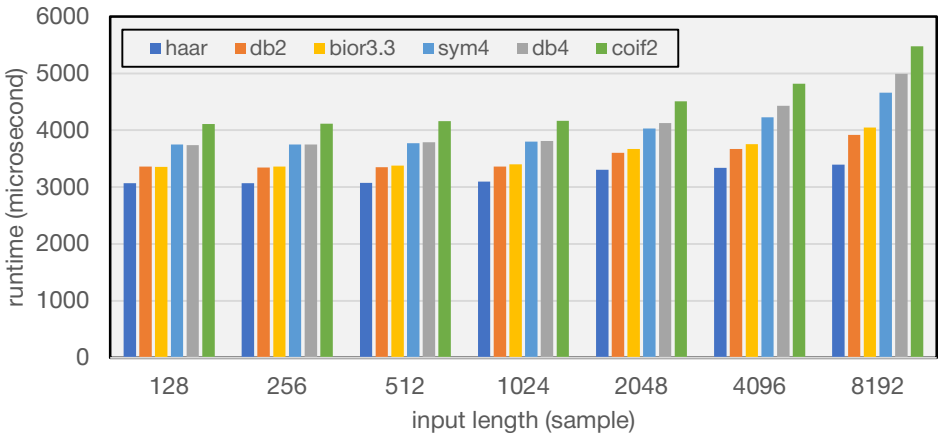

Figure D.3. Illustrating the speedup ratio of different input lengths and wavelets calculated from the runtimes of the MATLAB built-in lwt function divided by the time of the proposed recursive algorithm.