

STATIC HAND GESTURE RECOGNITION USING ARTIFICIAL
NEURAL NETWORK

HAITHAM SABAH HASAN

FACULTY OF COMPUTER SCIENCE AND INFORMATION
TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2014

STATIC HAND GESTURE RECOGNITION USING ARTIFICIAL
NEURAL NETWORK

HAITHAM SABAH HASAN

THESIS SUBMITTED IN FULFILMENT
OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF COMPUTER SCIENCE AND INFORMATION
TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2014

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: **Haitham Sabah Hasan** (I.C/Passport No: **A7524032**)

Registration/Matric No: **WHA100003**

Name of Degree: **PhD in Computer Science**

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):

STATIC HAND GESTURE RECOGNITION USING ARTIFICIAL NEURAL NETWORK

Field of Study: **Artificial Intelligence**

I do solemnly and sincerely declare that:

- (1) I am the sole author/writer of this Work;
- (2) This Work is original;
- (3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
- (4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
- (5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
- (6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature

Date

Subscribed and solemnly declared before,

Witness's Signature

Date

Name:

Designation:

ABSTRACT

The goal of static hand gesture recognition is to classify the given hand gesture data represented by some features into some predefined finite number of gesture classes.

The main objective of this effort is to explore the utility of two feature extraction methods, namely, hand contour and complex moments to solve the hand gesture recognition problem by identifying the primary advantages and disadvantages of each method. Artificial neural network is built for the purpose of classification by using the back- propagation learning algorithm.

The proposed system presents a recognition algorithm to recognize a set of six specific static hand gestures, namely: Open, Close, Cut, Paste, Maximize, and Minimize. The hand gesture image is passed through three stages, namely, pre-processing, feature extraction, and classification. In the pre-processing stage some operations are applied to extract the hand gesture from its background and prepare the hand gesture image for the feature extraction stage. In the first method, the hand contour is used as a feature which treats scaling and translation problems (in some cases). The complex moments algorithm is, however, used to describe the hand gesture and treat the rotation problem in addition to the scaling and translation. The back-propagation learning algorithm is employed in the multi-layer neural network classifier. The results show that hand contour method has a performance of 71.30% recognition, while complex moments has a better performance of 86.90% recognition rate.

ABSTRAK

Tujuan utama pencaman isyarat tangan statik ialah untuk mengkelas isyarat-isyarat tangan yang diwakili oleh beberapa ciri kepada sejumlah klas-klas isyarat. Objektif utama usaha ini adalah untuk mencuba dua kaedah pengekstrakan ciri, iaitu “*hand contour*” dan “*complex moments*” bagi tujuan menyelesaikan masalah pengiktirafan isyarat tangan dengan mengenal pasti kelebihan dan keburukan setiap daripada kaedah ini. Suatu rangkaian saraf buatan dibina untuk tujuan pengkelasan dengan mengguna algoritma pembelajaran “*back propagation*”. Sistem pencaman yang dicadangkan ini mengguna algoritma pencaman bagi mencam satu set yang terdiri dari enam isyarat tangan statik tertentu, iaitu: Buka, Tutup, Potong, Tampal, Maksimum, dan Minimum.

Proses pencaman isyarat tangan terdiri daripada tiga peringkat, iaitu, pra- pemprosesan, pengekstrakan ciri, dan klasifikasi.

Pada peringkat pra- pemprosesan beberapa operasi digunakan untuk mengasingkan imej isyarat tangan dari latar belakangnya dan menyediakan imej isyarat tangan untuk peringkat pengekstrakan ciri. Dalam kaedah pertama, “*hand contour*” digunakan sebagai ciri yang mengatasi masalah yang disebabkan oleh “*scaling*” dan “*translation*” (dalam beberapa kes).

Algoritma “*complex moments*” pula, digunakan untuk mewakili isyarat tangan dan menyelesaikan masalah putaran sebagai tambahan kepada “*scaling*” dan “*translation*”. Algoritma pembelajaran “*back propagation*” digunakan dalam rangkaian saraf berbilang lapisan yang berfungsi sebagai pengkelas. Keputusan yang diperolehi menunjukkan bahawa kaedah “*hand contour*” mempunyai prestasi pencaman sebanyak 71.30 %, manakala “*complex moments*” mempunyai prestasi yang lebih baik iaitu sebanyak 86.37 % pencama

ACKNOWLEDGEMENT

Alhamdulillah, all praises to Allah for giving me the strength to complete this thesis. I owe my deepest gratitude to my supervisor; Associate professor Datin Dr Sameem Abdul Kareem for their advice, guidance, encouragement, patience and support throughout this study. This research will never be accomplished without her supervision.

I am heartily thankful to the staffs of the Faculty of Computer Science and Information Technology (FCSIT), University Malaya (UM) for their help and support. I am also grateful to the University of Malaya for giving me the opportunity to further my study at FCSIT, UM.

I am indebted to many of my colleagues, for the words of courage and support in various possible ways.

I dedicated this thesis to my father and mother for all the prayers that help me to go through this lonely journey of research. My special thanks to my two brothers Laith and Haider, and my son, Sabah for their unconditional love and understanding.

TABLE OF CONTENTS

1.0	INTRODUCTION.....	1
1.1	General Introduction	1
1.2	Statement of the Problem.....	2
1.3	Research questions.....	4
1.1	Objective of Research.....	5
1.2	Scope of the study.....	5
1.3	Outline of the Thesis.....	8
2.0	LITERATURE REVIEW.....	9
2.1	Introduction.....	9
2.2	Gesture Definition.....	10
2.3	Hand Gestures.....	11
2.3.1	Static Gestures.....	12
2.3.2	Dynamic Gestures.....	12
2.4	The Basics of Gesture Recognition.....	12
2.5	Review of Hand Gesture Recognition systems.....	13
2.6	Applications of Hand Gesture Recognition	19
2.6.1	Virtual Reality.....	20
2.6.2	Sign Language.....	21
2.6.3	Hand Gesture-Based Graphical User Interface	22
2.6.4	Robotics	23
2.7	Gesture Recognition Techniques	24
2.7.1	Template Matching	24
2.7.2	Hidden Markov Models (HMMs).....	25
2.8	Summary	27
3.0	REVIEW OF IMAGE PROCESSING AND NEURAL NETWORKS	28
3.1	Introduction.....	28
3.2	Image Processing	28
3.2.1	Segmentation.....	29
3.2.2	Feature Extraction.....	35
3.3	Artificial Neural Networks.....	40
3.3.1	Artificial Neuron.....	40

3.3.2	Types of Activation Functions	41
3.3.3	Learning Paradigms	44
3.3.4	Neural Networks Architectures.....	45
3.3.5	Back-Propagation Learning Algorithm.....	46
3.3.6	Advantages of Neural Computing.....	49
3.4	Summary	50
4.0	PROPOSED STATIC HAND GESTURE RECOGNITION SYSTEM	51
4.1	Introduction.....	51
4.2	Hand gesture Image Capture.....	51
4.3	Pre-processing stage.....	54
4.3.1	Hand Segmentation	54
4.3.2	Noise Reduction.....	55
4.3.3	Edge Detection.....	56
4.4	Gesture Feature Extraction methods	57
4.4.1	Hand Contour.....	58
4.4.2	Complex Moments.....	64
4.5	Neural Network Based Classifier.....	64
4.5.1	ANN with Hand contour.....	65
4.5.2	ANN with complex moments	68
4.6	Summary	69
5.0	EXPERIMENTAL SETUPS	70
5.1	Introduction.....	70
5.2	Hand Contour with ANNs.....	70
5.2.1	Hand Gesture Segmentation.....	70
5.2.2	Noise Reduction.....	70
5.2.3	Edge Detection.....	70
5.2.4	Feature Extraction.....	70
5.2.5	Training Phase.....	71
5.2.6	Testing Phase	74
5.3	Complex Moment with ANNs	74
5.3.1	Image Trimming Effect.....	75
5.3.2	Coordinate Normalization.....	76
5.3.3	Complex Moments Calculation	76
5.3.4	Training Phase.....	78

5.3.5	Testing Phase	78
5.4	Preliminary results	79
5.4.1	Hand Contour with Neural Network	79
5.4.2	Complex Moments with Neural Network	83
5.5	Summary	86
6.0	RESULTS AND DISCUSSIONS	87
6.1	Introduction.....	87
6.2	Criteria for evaluation	87
6.3	Results of Testing Phase for Hand Contour with ANNs.....	88
6.3.1	Specificity and sensitivity for Hand Contour.....	96
6.3.2	Scaling and translation in Hand Contour	97
6.4	Results of Testing Phase Complex Moments with ANNs	100
6.4.1	Specificity and sensitivity for Complex Moments.....	108
6.4.2	Rotation, Scaling and Translation for Complex Moments.....	108
6.5	Comparison between the results of Hand Contour and Complex Moments.....	112
6.5.1	The Learning Speed	112
6.5.2	Recognition Accuracy	112
6.5.3	Overfitting of Neural networks	114
6.5.4	Comparison with previous works	114
6.6	Summary	116
7.0	CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK.....	118
7.1	Conclusions.....	118
7.2	Contribution of this study	120
7.3	Limitations and suggestions for future works.....	121
	REFERENCES	123

LIST OF FIGURES

Figure 1.1 Overview of the method used to develop our hand gesture recognition system	7
Figure 2.1 Vitarka Mudra (Rose, 1919)	10
Figure 2.2 The Cyborg Glove: Data Glove is Constructed with Stretch Fabric for Comfort and A Mesh Palm for Ventilation (Adapted from (Kevin, Ranganath, & Ghosh, 2004)	15
Figure 2.3 A Gestural Interface to Virtual Environments (O'Hagan, Zelinsky, & Rougeaux, 2002)	20
Figure 2.4 The ASL Gesture Set (Kulkarni & Lokhande, 2010)	22
Figure 2.5 Menu Items are displayed in a pie shape; the thumb is extended to switch from Draw Mode to Menu Mode (Mo, Lewis, & Neumann, 2005)	22
Figure 2.6 Human –Robot Interaction (Kosuge & Hirata, 2004)	23
Figure 2.7 Template Matching	26
Figure 3.1 Sobel Operation Mask	31
Figure 3.2 Prewitt Operation Mask	33
Figure 3.3 Laplacian Operation Mask	34
Figure 3.4 Simple Artificial Neuron (S. Haykin, 1998)	41
Figure 3.5 Identity Function	42
Figure 3.6 The Binary Step Function	42
Figure 3.7 Binary Sigmoid Function	43
Figure 3.8 Bipolar Sigmoid Function	43
Figure 3.9 Single Layer Neural Network	45
Figure 3.10 Three- Layer Neural Network	46
Figure 4.1 Six Static Hand Gestures: Open, Close, Cut, Paste, Maximize and Minimize	51
Figure 4.2 Hand gestures images under different conditions	53
Figure 4.3 Hand gesture images before and after segmentation	55
Figure 4.4 Median Filter Effect	55
Figure 4.5 An example illustrating the median filter using 3×3 neighbourhood	56
Figure 4.6 Sobel Edge detection for Open, Close and Cut	57
Figure 4.7 Sobel Operator Edge Detection	59
Figure 4.8 A 7 x7 Surround Mask	59
Figure 4.9 Image Scaling Effect 32×32	60
Figure 4.10 General features of the Cut gesture	61
Figure 4.11 Example of binary encoding of general features (6 × 6) matrix	62
Figure 4.12 Geometric and General Features as Input Vector to the Multilayer Neural Network	63
Figure 4.13 Complex Moments Feature Vector	64
Figure 4.14 Gesture Recognition Network Architecture	65
Figure 4.15 Flowchart for Back-Propagation Learning Algorithm	67
Figure 4.16 Detailed Design of a Neural Network	68

Figure 5.1 Feature Extraction Stage.....	71
Figure 5.2 Hand Gestures under different lightening conditions.....	73
Figure 5.3 Image Trimming Effects.....	75
Figure 5.4 Coordinate Normalization	76
Figure 5.5 Learning convergence algorithm for the first neural network.....	80
Figure 5.6 Learning phase with respect to number of epochs (first neural network)	80
Figure 5.7 Learning convergence algorithm for the second neural network	81
Figure 5.8 Learning phase with respect to number of epochs (second neural network)	81
Figure 5.9 Learning convergence algorithm for the third neural network.....	82
Figure 5.10 Learning phase with respect to number of epochs (third neural network) ..	82
Figure 5.11 Learning convergence algorithm for the first neural network.....	83
Figure 5.12 Learning phase with respect to number of epochs (first neural network) ...	83
Figure 5.13 Learning convergence algorithm for the second neural	84
Figure 5.14 Learning phase with respect to number of epochs (second neural network)	
.....	84
Figure 5.15 Learning convergence algorithm for the third neural network.....	85
Figure 5.16 Learning phase with respect to number of epochs (third neural network) ..	85
Figure 6.1 Percentages of Correct Recognition for each hand gesture class (Hand Contour)	89
Figure 6.2 Close gesture with translation and scaling	95
Figure 6.3 Percentages of Correct Recognition for each hand gesture class (complex moments).....	101

LIST OF TABLES

Table 4.1 Parameters for the Five Multilayer Neural Networks	66
Table 4.2 Parameters for the Five Multi-layer Neural Networks.....	69
Table 5.1 Parameters for the Five Neural Networks.....	72
Table 5.2 Complex Moments Values before Normalization	77
Table 5.3 Complex Moments Values after Normalization	77
Table 5.4 Parameters of Back-Propagation Neural Networks	Error! Bookmark not defined.
Table 6.1 Summary of Recognition Results (Hand Contour)	88
Table 6.2 Results for (Open) Hand Gesture with Scaling and Translation effects (Hand Contour)	90
Table 6.3 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Open) (Hand Contour).....	91
Table 6.4 Results for (Close) Hand Gesture with Scaling and Translation effects (Hand Contour)	91
Table 6.5 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Close) (Hand Contour)	91
Table 6.6 Results for (Cut) Hand Gesture with Scaling and Translation effects (Hand Contour)	92
Table 6.7 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Cut) (Hand Contour)	92
Table 6.8 Results for (Paste) Hand Gesture with Scaling and Translation effects (Hand Contour)	93
Table 6.9 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Paste) (Hand Contour).....	93
Table 6.10 Results for (Maximize) Hand Gesture with Scaling and Translation effects (Hand Contour)	94
Table 6.11 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Maximize) (Hand Contour)	94
Table 6.12 Results for (Minimize) Hand Gesture with Scaling and Translation effects (Hand Contour)	95
Table 6.13 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Minimize) (Hand Contour).....	95
Table 6.14 specificity and sensitivity values for hand gestures (Hand Contour)	96
Table 6.15 Recognition Errors of Hand Gesture with Scaling, Translation and Artificial Illumination Effects.....	97
Table 6.16 Recognition Rate of Various Hand Gestures with Scaling Effects for Hand Contour.....	98
Table 6.17 Recognition Rate of Various Hand Gesture with Translation for Hand Contour.....	98
Table 6.18 Confusion matrix for Hand Contour	99

Table 6.19 Summary of the Recognition Results and the Recognition Rates (Complex Moments)	100
Table 6.20 Results for (Open) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	102
Table 6.21 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Open) (Complex Moments).....	102
Table 6.22 Results for (Close) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	103
Table 6.23 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Close) (Complex Moments)	103
Table 6.24 Results for (Cut) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	104
Table 6.25 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Cut) (Complex Moments)	104
Table 6.26 Results for (Paste) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	105
Table 6.27 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Paste) (Complex Moments).....	105
Table 6.28 Results for (Maximize) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	106
Table 6.29 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Maximize) (Complex Moments)	106
Table 6.30 Result for (Minimize) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments).....	107
Table 6.31 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Minimize) (Complex Moments).....	107
Table 6.32 Specificity and sensitivity values for Complex Moments	108
Table 6.33 Recognition Error of Hand Gesture with Translation for Complex Moments	109
Table 6.34 Recognition of Hand Gesture with Rotation for Complex Moments	110
Table 6.35 Recognition of Hand Gesture with Scaling For Complex Moments	110
Table 6.36 Recognition of Hand Gesture with Translation for Complex Moments.....	111
Table 6.37 Confusion Matrix of the Results Achieved Using Complex Moments	111
Table 6.38 The Number of “Not Recognized” Cases for Hand Contour and Complex Moments.....	113
Table 6.39 The Number of “False” Cases for Hand Contour and Complex Moments.....	113
Table 6.40 Recognition rates of related hand gesture recognition methods	115

1.0 INTRODUCTION

1.1 General Introduction

Computers have become a key element of our society since their first appearance in the second half of the last century. Surfing the web, typing a letter, playing a video game, or storing and retrieving data are some of the examples of the tasks that involve the use of computers. Computers will increasingly influence our everyday life because of the constant decrease in the price and size of personal computers and the advancement of modern technology. Today, the widespread use of mobile devices such as smart phones and tablets either for work or communication, has enabled the people to easily access applications in different domains which include GPS navigation, language learning apps, etc. The efficient use of most current computer applications requires user interaction. Thus, human-computer interaction (HCI) has become an active field of research in the past few years (Just, 2006). On the other hand, input devices have not undergone significant changes since the introduction of the most common computer in the nineteen eighties probably because existing devices are adequate. Computers are tightly integrated with everyday life, and new applications and hardware are constantly introduced as answers to the needs of modern society (Symeonidis, 1996). The majority of existing HCI devices is based on mechanical devices, such as keyboards, mouse, joysticks, or game pads. However, a growing interest in a class of applications that use hand gestures has emerged, aiming at a natural interaction between the human and various computer-controlled displays (Pavlovic, Sharma, & Huang, 1997). The use of human movements, especially hand gestures, has become an important part of human computer intelligent interaction (HCII) in recent years, which serves as a motivating

force for research in modeling, analysis, and recognition of hand gestures (Wu & Huang, 1999). The various techniques developed in HCII can be extended to other areas, such as surveillance, robot control, and teleconferencing (Wu & Huang, 1999). The detection and understanding of hand and body gestures is becoming an important and challenging task in computer vision. The significance of the problem can be illustrated easily by the use of natural gestures that are applied with verbal and nonverbal communications (Dadgostar, Barczak, & Sarrafzadeh, 2005).

1.2 Statement of the Problem

Gesture recognition has been adapted for various research applications from facial gestures to complete bodily human action (Dong, Yan, & Xie, 1998). Several applications have emerged and created a stronger need for this type of recognition system (Dong et al., 1998) .

Static gesture recognition is a pattern recognition problem; as such, an essential part of the pattern recognition pre-processing stage, namely, feature extraction, should be conducted before any standard pattern recognition techniques can be applied. Features correspond to the most discriminative information about the image under certain lighting conditions. A fair amount of research has been performed on different aspects of feature extraction (Bretzner, Laptev, & Lindeberg, 2002; Gupta, Jaafar, & Ahmad, 2012; Parvini & Shahabi, 2007; Vieriu, Goras, & Goras, 2011; Wu & Huang, 1999; Yoon, Soh, Bae, & Seung Yang, 2001). Parvini and Shahabi (2007) proposed a method for recognizing static and dynamic hand gestures by analysing the raw streams generated by the sensors attached to human hands. This method achieved a recognition rate of more than 75% on the ASL signs. However, the user needs to use a glove-based interface to extract the features of the hand gestures which limits their usability in real-

world applications, as the user needs to use special gloves in order to interact with the system.

Another study (Vieriu et al., 2011) presented a real-time static isolated gesture recognition application using a hidden Markov model approach. The features of this application were extracted from gesture silhouettes. Nine different hand poses with various degrees of rotation were considered. The drawback of this feature extraction method is the use of skin-based segmentation method which does not work properly in the presence of skin-colored objects in the background.

Dong et al. (1998) described an approach of vision-based gesture recognition for human-vehicle interaction using the skin -colour method for hand segmentation. Similar to the problem in (Vieriu et al., 2011), the performance of the recognition system is dramatically affected when skin-coloured objects are present in the background.

Developing a hand gesture recognition system that is capable of working under various conditions is difficult but it is also more practical because these challenging conditions exist in real-world environment. These conditions include varying illumination and complex background as well as some effects of scaling, translation, and rotation by specific angles (Freeman & Roth, 1995; Li, 2005; Parvini & Shahabi, 2007; Symeonidis, 1996).

Another criteria that should be considered in the hand gesture recognition systems, that are employed in real-world applications is the computational cost. Some feature extraction methods have the disadvantage of being complicated and therefore consume more time, like Gabor filters with a combination of PCA (Gupta et al., 2012) and the combination of PCA and Fuzzy-C-Mean (Amin & Hong, 2007) which are computationally costly which may limit their use in real-world applications.

In fact, the trade-off between the accuracy and the computational cost in proposed hand gesture methods should be considered (Chen, Fu, & Huang, 2003). While, most hand gesture systems focus only on the accuracy for hand gesture system assessments (Francke, Ruiz-del-Solar, & Verschae, 2007), it is desirable, in the phase of results evaluation, to consider the two criteria, namely, accuracy and the computational cost in order to identify their strengths and weaknesses and to recommend their potential applications (Chen et al., 2003).

Some of the researches mentioned in this section are further discussed in chapter 2 together with their findings and limitations.

1.3 Research questions

The following research questions are used as guidance to conduct this research at various stages:

Q1. What is the effect of using the hand contour feature extraction method on the recognition of static hand gestures?

Q2. What is the effect of using the complex moments feature extraction method on the recognition of static hand gestures?

Q3. Do scaling, rotation and translation reduce the efficiency of recognising hand gestures?

Q4. Is the recognition capability of the hand contour and the complex moments feature extraction method affected by different lighting conditions ?

Q5. What is the performance of the artificial neural network in terms of accuracy and speed when used with different feature extraction methods?

Q6. What are the potential applications that can use hand gesture recognition system with limited number of gestures?

1.1 Objective of Research

1. Compare and contrast the performance of two popular feature selection approaches, namely, hand contour and complex moments in recognising static hand gestures.
2. Explore the performance of the feature selection methods in (1.) for recognising hand gestures under different conditions, such as scaling, rotation, and translation.
3. Investigate the suitability of artificial neural networks as a classification method for hand gesture recognition in terms of accuracy, convergence speed and overfitting.
4. Develop a static hand gesture recognition system that can be used for applications that involve a limited number of hand gestures.

The objective of the current research is to discuss and try to find answers for the questions posed in “Research questions” section. For objective 1, this study attempts to evaluate the effect of using feature extraction methods, namely, hand contour and complex moments for static hand gesture recognition problem (Q1 and Q2). In addition, we aim in objective 2 to evaluate effect of scaling, translation, rotation and lighting conditions on the performance of the recognition for both feature extraction methods (Q3 and Q4). In objective 3, we try to answer Q5 by investigating the suitability of Artificial Neural Network as a classifier for hand gesture recognition system using two criteria which are performance and speed. In addition, objective 3 seeks to answer Q6 by recommending the potential applications that can use the system proposed in this study.

1.2 Scope of the study

This study deals with the problem of developing a vision-based static hand gesture recognition algorithm to recognize the following six static hand gestures: Open, Close, Cut, Paste, Maximize, Minimize. These gestures are chosen because they are commonly used to communicate and can thus be used in various applications, such as, a virtual

mouse that can perform six tasks (Open, Close, Cut, Paste, Maximize, Minimize) for a given application. The proposed system consists mainly of three phases: the first phase (i.e.: pre-processing), the next phase (i.e.: feature extraction) and the final phase (i.e.: classification). The first phase includes hand segmentation that aims to isolate hand gesture from the background and removing the noises using special filters. This phase includes also edge detection to find the final shape of the hand. The next phase, which constitutes the main part of this research, is devoted to the feature extraction problem where two feature extraction methods, namely, hand contour and complex moments are employed. These two extraction methods were applied in this study because they used different approaches to extract the features, namely, a boundary-based for hand contour and region-based for complex moments.

The feature extraction algorithms deal with problems associated with hand gesture recognition such as scaling, translation and rotation. In the classification phase where neural networks are used to recognize the gesture image based on its extracted feature, we analyse some problems related to the recognition and convergence of the neural network algorithm. As a classification method, ANN has been widely employed especially for real-world applications because of its ability to work in parallel and online training (Rubaai, Kotaru, & Kankam, 1999). In addition, a comparison between the two feature extraction algorithms is carried out in terms of accuracy and processing time (computational cost). This comparison, using these two criteria, is important to identify the strengths and weaknesses of each feature extraction method and assess the potential application of each method. Figure 1.1 provides an overview of the method used to develop the hand gesture recognition system.

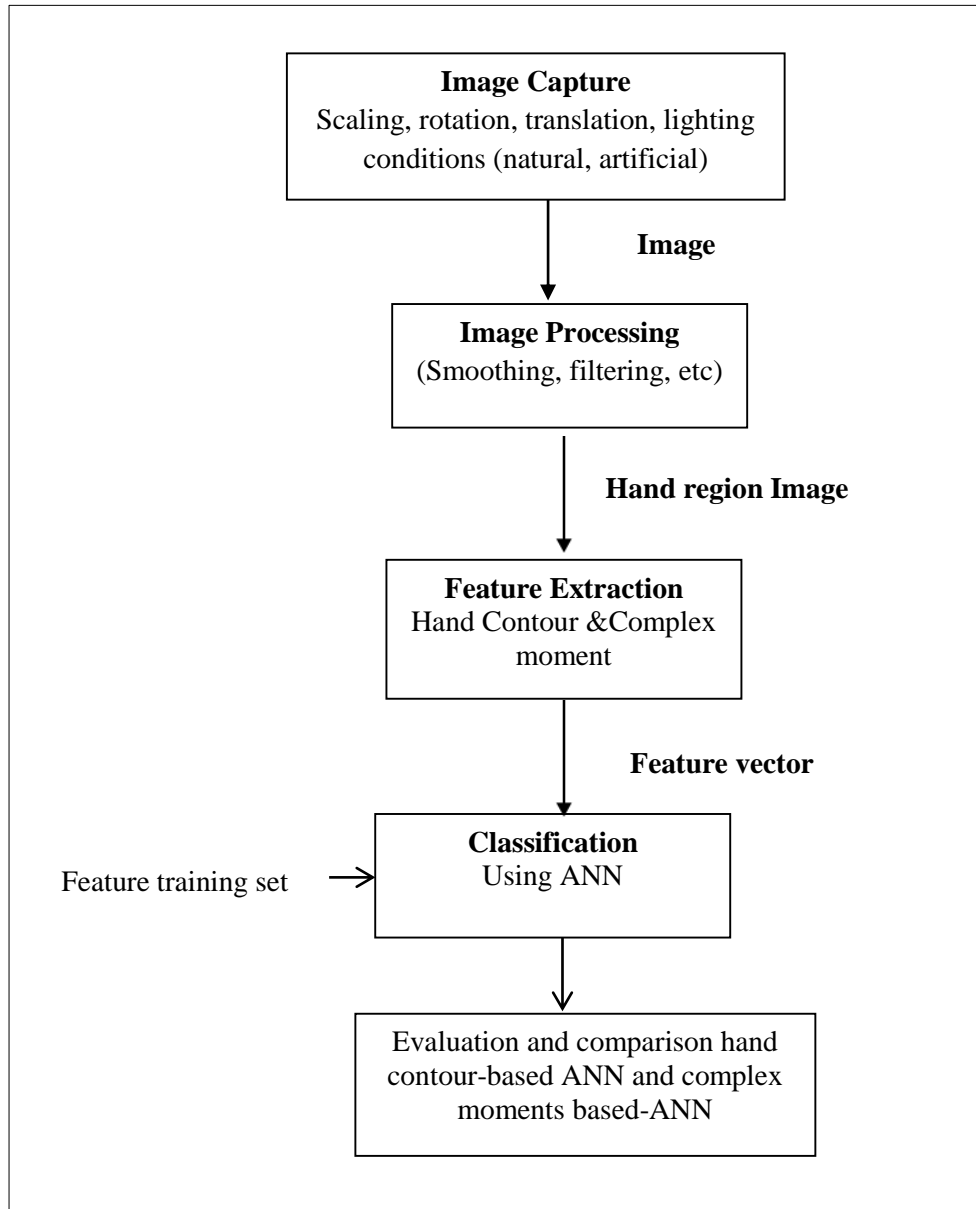


Figure 1.1 Overview of the method used to develop our hand gesture recognition system

1.3 Outline of the Thesis

This thesis is divided into seven chapters.

Chapter 1 provides an introduction. A brief overview of the problem is presented in this chapter. A fair amount of research in static hand gesture recognition that was conducted on different aspects is also discussed. The objectives of the research are presented as well.

Chapter 2 describes human gestures. This chapter also provides the definition of gesture, types, and the concept of gesture recognition with its applications. An overview of gesture recognition techniques and related works are also presented, which are used for static and dynamic hand gesture recognition.

Chapter 3 presents an overview of the general stages of the system, which includes background information about the image processing, their extraction, and neural networks in general.

Chapter 4 presents the proposed recognition algorithm based on neural network, which contains two features of hand contour and complex moments.

Chapter 5 discusses the experiment obtained from the presentation of the proposed gesture recognition technique.

Chapter 6 discusses the results obtained from the presentation of the proposed gesture recognition technique.

Chapter 7 highlights the conclusions and provides suggestions for future work.

2.0 LITERATURE REVIEW

2.1 Introduction

Humans naturally use gesture to communicate. Young children can readily learn to communicate with machines before they learn to talk (Kjeldsen, 1997). Gestures are frequently used as a means of communication. Gestures are used for everything, from pointing at a person to getting their attention to conveying information about space and characteristics. Gestures are not merely used as qualifiers for oral communication but actually form part of the main language generation process (Shet et al., 2004).

Gestures in modern societies are everything, from a smile to hand-and-arm movements. Most people add meaning to their words by drawing pictures with their hands, which is done subconsciously and is therefore hard to suppress. They do it even when they speak on the phone or when they talk to themselves. Deaf or mute people might use sign language as the sole means to communicate. Instructors in diverse fields such as military or aerobics use arm signals to give commands (Winnemöller, 1999).

Gestures play a major role in many aspects of human life. Gesturing is probably universal. A community that does not use gestures probably does not exist. Gestures are a crucial part of everyday conversation, such as in Greek paintings, Indian miniatures, or European paintings. Gestures play a role in religious or spiritual rituals, such as the Christian sign of the cross. A mudra (Sanskrit, which literally means “seal”) in Hinduism and Buddhism is a symbolic gesture made with the hand or fingers. Each mudra has a specific meaning and plays a central role in Hindu and Buddhist photography. An example is the *Vitarka mudra*, which is shown in Figure 2.1. Gestures are also provided in discussions and transmission of Buddhist teaching, which are

conducted by joining the tips of the thumb and the index together while keeping the fingers straight (Rose, 1919).



Figure 2.1 Vitarka Mudra
(Rose, 1919)

2.2 Gesture Definition

The following definition of gesture can be found in the Oxford Advanced Learner's Dictionary:

“Gesture - a movement of part of the body, especially, a hand or the head to express an idea or meaning” (Gesture).

The word gesture is used for various phenomena that involve human movement, especially of the hands and arms. However, some of these gestures are interactive or communicative (Nehaniv et al., 2005). Gestures differ from pure functional movements, which can be achieved with other actions. Movements that show or symbolize something and contain a message are called gestures. For example, steering a car is a pure functional movement without information. However, gestures that describe the size of a round object by circling the hand contains information about the size of the object. Writing down words on a sheet of paper is a pure functional movement. Words contain

the information, not the movement of the hand. Writing can be replaced by typing. Therefore, writing is not a gesture (Cadoz & Wanderley, 2000).

2.3 Hand Gestures

Hand gestures or gestures performed by one or two hands is the largest category of gestures because of the ability of the human hand to acquire a huge number of clearly discernible configurations, a fact of importance for sign languages. Hand gestures can be classified into several categories according to different application scenarios. These categories include conversational gestures, controlling gestures, manipulative gestures, and communicative gestures (Wu & Huang, 2001). Sign language is an important case of communicative gestures. Sign languages are suitable for acting as a test-bed for vision algorithms because this type of language is highly structural (Wu & Huang, 1999). Similarly, sign language can help the disabled interact with computers. Controlling gestures is the focus of current studies in vision-based interfaces (VBI) (Wu & Huang, 1999). Virtual objects can be located by analyzing pointing gestures. Some display-control applications demonstrate the potential of pointing gestures in HCI. Another controlling gesture is the navigating gesture. Instead of using wands, hand orientation can be captured as a 3D directional input to navigate the virtual environments (VEs). Manipulative gestures serve as a natural way to interact with virtual objects. Tele-operation and virtual assembly are good examples of applications. Communicative gestures are subtle in human interaction, which involves psychological studies. However, vision-based motion capturing techniques can help these studies (Wu & Huang, 1999). Generally, gestures can be classified into static gestures and dynamic gestures. Static gestures are described in terms of hand shapes, whereas dynamic gestures are generally described according to hand movements (Chang, Chen, Tai, & Han, 2006).

2.3.1 Static Gestures

Liang (Lamar, Bhuiyan, & Iwata, 2000) provided the following definition of static gesture or hand posture:

“Posture is a specific combination of hand position, orientation, and flexion observed at some time instance.”

Posture of static gestures is not a time-varying signal. Thus, they can be completely analyzed using a single image or a set of images of the hand taken at a specific time, the hand signs for "OK," or the "STOP" sign, in a simple picture are examples of hand postures since they convey enough meaning for complete understanding.

2.3.2 Dynamic Gestures

Liang (Lamar et al., 2000) provided the following definition of “gesture” to describe dynamic gestures:

“Gesture is a sequence of postures connected by motion over a short time span.”

A gesture can be regarded as a sequence of postures. The individual frames in a video signal define the postures, whereas the video sequence defines the gesture. The head movements for "No" and "Yes" and the hand "goodbye" or "come here" gestures that can only be recognized by taking the temporal context information, are good examples of dynamic gestures.

2.4 The Basics of Gesture Recognition

The general gesture recognition process in any kind of system can be broken down into the components shown in Figure 1.1 (Winnemöller, 1999).

The first stage of hand gesture recognition system, is primarily concerned with the hardware of the system and how the data for the recognition process is gathered (in the form of bitmaps or lists of vertices). The second stage is a pre-processor stage in which

edge-detection, smoothing, and other filtering processes occur. In this stage the data is prepared for the main computational stage, that is, feature extraction (3rd stage). The features of the input data are extracted and then evaluated in the fourth phase, the evaluation stage, by one or more of the several possible ways to decide which gesture corresponds to the extracted feature vector. All systems have a limited set of gestures, such as Open, Cut, Paste, etc., that they can recognize at any given time (Winnemöller, 1999).

2.5 Review of Hand Gesture Recognition systems

Gesture recognition is an important topic in computer vision because of its wide range of applications, such as HCI, sign language interpretation, and visual surveillance (Kim & Cipolla, 2007).

Krueger (1991) was the first who proposed Gesture recognition as a new form of interaction between human and computer in the mid-seventies. The author designed an interactive environment called *computer-controlled responsive environment*, a space within which everything the user saw or heard was in response to what he/she did. Rather than sitting down and moving only the user's fingers, he/she interacted with his/her body. In one of his applications, the projection screen becomes the wind-shield of a vehicle the participant uses to navigate a graphic world. By standing in front of the screen and holding out the user's hands and leaning in the direction in which he/she want to go, the user can fly through a graphic landscape. However, this research cannot be considered strictly as a hand gesture recognition system since the potential user does not only use the hand to interact with the system but also his/her body and fingers, we choose to cite this (Krueger, 1991) due to its importance and impact in the field of gesture recognition system for interaction purposes.

Gesture recognition has been adapted for various other research applications from facial gestures to complete bodily human action (Dong et al., 1998). Thus, several applications have emerged and created a stronger need for this type of recognition system (Dong et al., 1998). In their study, (Dong et al., 1998) described an approach of vision-based gesture recognition for human-vehicle interaction. The models of hand gestures were built by considering gesture differentiation and human tendency, and human skin colors were used for hand segmentation. A hand tracking mechanism was suggested to locate the hand based on rotation and zooming models. The method of hand-forearm separation was able to improve the quality of hand gesture recognition. The gesture recognition was implemented by template matching of multiple features. The main research was focused on the analysis of interaction modes between human and vehicle under various scenarios such as: calling-up vehicle, stopping the vehicle, and directing vehicle, etc. Some preliminary results were shown in order to demonstrate the possibility of making the vehicle detect and understand the human's intention and gestures. The limitation of this study was the use of the skin colors method for hand segmentation which may dramatically affect the performance of the recognition system in the presence of skin-colored objects in the background.

Hand gesture recognition studies started as early as 1992 when the first frame grabbers for colored video input became available, which enabled researchers to grab colored images in real time. This study signified the start of the development of gesture recognition because color information improves segmentation and real-time performance is a prerequisite for HCI (Shet et al., 2004).

Hand gesture analysis can be divided into two main approaches, namely, glove-based analysis, vision-based analysis (Ionescu, Coquin, Lambert, & Buzuloiu, 2005).

The glove-based approach employs sensors (mechanical or optical) attached to a glove that acts as transducer of finger flexion into electrical signals to determine hand posture, as shown in Figure 2.2.

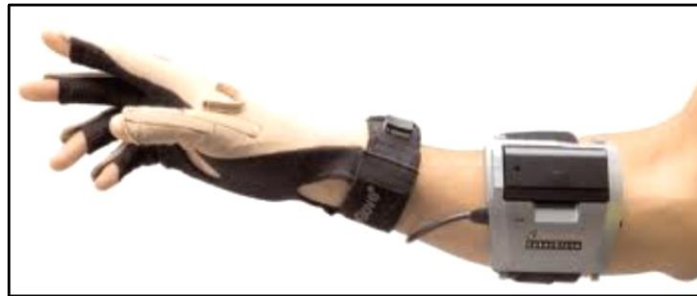


Figure 2.2 The Cyborg Glove: Data Glove is Constructed with Stretch Fabric for Comfort and A Mesh Palm for Ventilation (Adapted from (Kevin, Ranganath, & Ghosh, 2004)

The relative position of the hand is determined by an additional sensor. This sensor is normally a magnetic or an acoustic sensor attached to the glove. Look-up table software toolkits are provided with the glove for some data-glove applications for hand posture recognition. This approach was applied by (Parvini & Shahabi, 2007) to recognize the ASL signs. The recognition rate was 75%. The limitation of this approach is that the user is required to wear a cumbersome device, and generally carry a load of cables that connect the device to a computer (Pavlovic et al., 1997). Another hand gesture recognition system was proposed in (Swapna, Pravin, & Rajiv, 2011) to recognize the numbers from 0 to 10 where each number was represented by a specific hand gesture. This system has three main steps, namely, image capture, threshold application, and number recognition. It achieved a recognition rate of 89% but it has some limitations as it functioned only under a number of assumptions, such as wearing of colored hand gloves and using a black background.

The second approach, vision based analysis, is based on how humans perceive information about their surroundings (Ionescu et al., 2005). In this approach, several feature extraction techniques have been used to extract the features of the gesture images. These techniques include Orientation Histogram (Freeman & Roth, 1995; Symeonidis, 1996), Wavelet Transform (Triesch & von der Malsburg, 1996), Fourier Coefficients of Shape (Licsár & Szirányi, 2002), Zernic Moment (Chang et al., 2006), Gabor filter (Amin & Hong, 2007; Deng-Yuan, Wu-Chih, & Sung-Hsiang, 2009; Gupta et al., 2012), Vector Quantization (H. Meng, Furoo, & Jinxi, 2014), Edge Codes (Chao, Meng, Liu, & Xiang, 2003), Hu Moment (Liu & Zhang, 2009), Geometric feature (Bekir, 2012) and Finger-Earth Mover's Distance (FEMD) (Zhou, Junsong, Jingjing, & Zhengyou, 2013).

Most of these feature extraction methods have some limitations. In orientation histogram for example, which was developed by (McConnell, 1986), the algorithm employs the histogram of local orientation. This simple method works well if examples of the same gesture map to similar orientation histograms, and different gestures map to substantially different histograms (Freeman & Roth, 1995). Although this method is simple and offers robustness to scene illumination changes, its problem is that the same gestures might have different orientation histograms and different gestures could have similar orientation histograms which affects its effectiveness (Khan & Ibraheem, 2012). This method was used by (Freeman & Roth, 1995) to extract the features of 10 different hand gesture and used nearest neighbour for gesture recognition. The same feature extraction method was applied in another study (Symeonidis, 1996) for the problem of recognizing a subset of American Sign Language (ASL). In the classification phase, the author used a Single Layer Perceptron to recognize the gesture images. Using the same feature method, namely, orientation histogram, (Ionescu et al., 2005) proposed a gesture recognition method using both static signatures and an original dynamic signature. The

static signature uses the local orientation histograms in order to classify the hand gestures. Despite the limitations of orientation histogram, the system is fast due to the ease of the computing orientation histograms, which works in real time on a workstation and is also relatively robust to illumination changes. However, it suffers from the same fate associated with different gestures having the same histograms and the same gestures having different histograms as discussed earlier.

In (Amin & Hong, 2007), the authors used Gabor filter with PCA to extract the features and then fuzzy-c-means to perform the recognition of the 26 gestures of the ASL alphabets. Although the system achieved a fairly good recognition accuracy 93.32%, it was criticized for being computationally costly which may limit its deployment in real-world applications (Gupta et al., 2012).

Another method extracted the features from color images as in (R.-L. Vieriu, B. Goras, & L. Goras, 2011) where they presented a real-time static isolated gesture recognition application using a hidden Markov model approach. The features of this application were extracted from gesture silhouettes. Nine different hand poses with various degrees of rotation were considered. This simple and effective system used colored images of the hands. The recognition phase was performed in real-time using a camera video. The recognition system can process 23 frames per second on a Quad Core Intel Processor. This work presents a fast and easy-to-implement solution to the static one hand-gesture recognition problem. The proposed system achieved 96.2% recognition rate. However, the authors postulated that the presence of skin-colored objects in the background may dramatically affect the performance of the system because the system relied on a skin-based segmentation method. Thus, one of the main weaknesses of gesture recognition from color images is the low reliability of the segmentation process, if the background has color properties similar to the skin (Oprinescu, Rasche,

& Bochao, 2012).

The feature extraction step is usually followed by the classification method, which use the extracted feature vector to classify the gesture image into its respective class. Among the classification methods employed are: Nearest Neighbour (Chang et al., 2006; Freeman & Roth, 1995; Licsár & Szirányi, 2002), Artificial Neural Networks (Just, 2006; Parvini & Shahabi, 2007; Symeonidis, 1996), Support Vector Machines (SVMs)(Deng-Yuan et al., 2009; Gupta et al., 2012; Liu & Zhang, 2009), Hidden Markov Models (HMMs) (Vieriu et al., 2011).

As an example of classification methods, Nearest Neighbour classifier is used as hand recognition method in (Licsár & Szirányi, 2002) combined with modified Fourier descriptors (MFD) to extract features of the hand shape. The system involved two phases, namely, training and testing. The user in the training phase showed the system using one or more examples of hand gestures. The system stored the carrier coefficients of the hand shape, and in the running phase, the computer compared the current hand shape with each of the stored shapes through the coefficients. The best matched gesture was selected by the nearest-neighbor method using the MED distance metric. An interactive method was also employed to increase the efficiency of the system by providing feedback from the user during the recognition phase, which allowed the system to adjust its parameters in order to improve accuracy. This strategy successfully increased the recognition rate from 86% to 95%.

Nearest neighbour classifier was criticised for being weak in generalization and also for being sensitive to noisy data and the selection of distance measure (Athitsos & Sclaroff, 2005).

To conclude the related works, we can say that hand gesture recognition systems are generally divided into two main approaches, namely, glove-based analysis and vision-

based analysis. The first approach, which uses a special gloves in order to interact with the system, and was criticized because the user is required to wear a cumbersome device with cables that connect the device to the computer. In the second approach, namely, the vision-based approach, several methods have been employed to extract the features from the gesture images. Some of these methods were criticized because of their poor performance in some circumstances. For example, orientation histogram's performance is badly affected when different gestures have similar orientation histograms. Other methods such as Gabor filter with PCA suffer from the high computational cost which may limit their use in real-life applications. In addition, the efficiency of some methods that use skin-based segmentation is dramatically affected in the presence of skin-colored objects in the background.

Furthermore, hand gesture recognition systems that use feature extraction methods suffer from working under different lighting conditions as well as the scaling, translation, and rotation problems.

2.6 Applications of Hand Gesture Recognition

Some existing applications of hand gesture recognition are as follows: (1) interaction with virtual environment , for example, in one of the applications the user “painted” on a virtual wall with an extended finger, and erased what they had done with their spread open hand (Kjeldsen, 1997), (2) sign language understanding, and (3) as a part of more traditional computer interfaces such as the use of gesture as a direct mouse replacement (Kjeldsen, 1997). Although sign language is a very attractive application, it has a unique set of problems and potentially includes several of the subtleties of natural language understanding and speech recognition (Kjeldsen, 1997). A few examples of these applications are provided below.

2.6.1 Virtual Reality

The primary goal of virtual environments (VEs) is to support natural, efficient, powerful, and flexible interactions (Figure 2.3). The traditional two-dimensional keyboard and mouse-oriented graphical user interface (GUI) is not suitable for (VEs). Devices that can sense body position and orientation, direction of gaze, speech and sound, facial expressions, galvanic skin response, and other aspects of human behaviour or state can be used to mediate communication between the human and the environment (Turk, 1999).

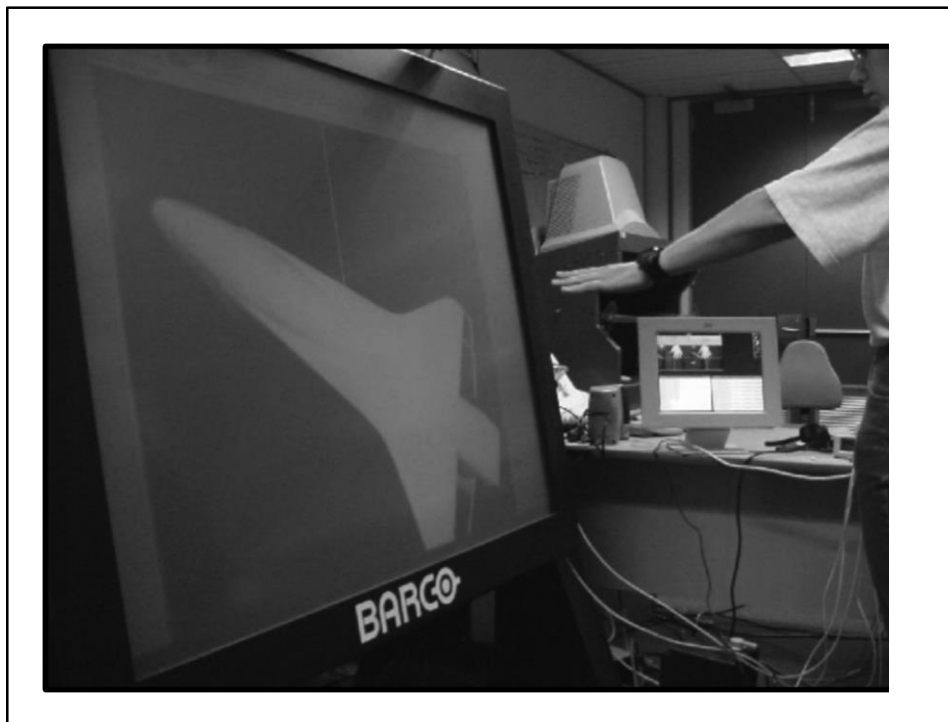


Figure 2.3 A Gestural Interface to Virtual Environments (O'Hagan, Zelinsky, & Rougeaux, 2002)

This interface is used to control navigation and manipulation of 3D objects. The user controls the direction of the object by tilting his hand. Forward and backward motion is controlled via the location of the user's hand in space.

Interactions with virtual reality applications are currently performed in a simple way. Sophisticated devices, such as space balls, 3D mice, or data gloves, are merely used for pointing and grabbing, i.e., the same I/O paradigm used in 2D mice (Winnemöller, 1999). Although traditional input devices (e.g., keyboards, mice, and joysticks) are still widely used in virtual environments and mobile applications, the virtual environments remain abstract and require physical contact with the devices. The presence of these devices is considered a barrier to interactions in virtual environments and mobile settings where gestures have been recognized and pursued as a more natural and more effective mechanism for human computer interaction. However, the difficulty of creating gesture interfaces impedes further development and application of this technology (Mo & Neumann, 2006).

2.6.2 Sign Language

Sign language, which is a type of structured gesture, is one of the most natural means of exchanging information for most hearing impaired individuals. This motivated the interest to develop systems that can accept sign language as one of the input modalities for human-computer interaction (HCI) to support the communication between the deaf and the hearing society. In fact, a new field of sign language engineering is emerging, in which advanced computer technology is being utilized to enhance the system capability, consequently serving society by creating a powerful and friendly human-computer interface (Gao, Ma, Wu, & Wang, 2000). Sign language is undoubtedly the most grammatically structured and complex set of human gestures. In American Sign Language (ASL) (Figure 2.4), the use of hand postures (static gestures) is very important in differentiating between numerous gestures (Binh, Enokida, & Ejima, 2006). Several hand gesture recognition systems for sign language recognition are developed (Gupta et al., 2012; Naidoo, Omlin, & Glaser, 1999; Symeonidis, 1996).

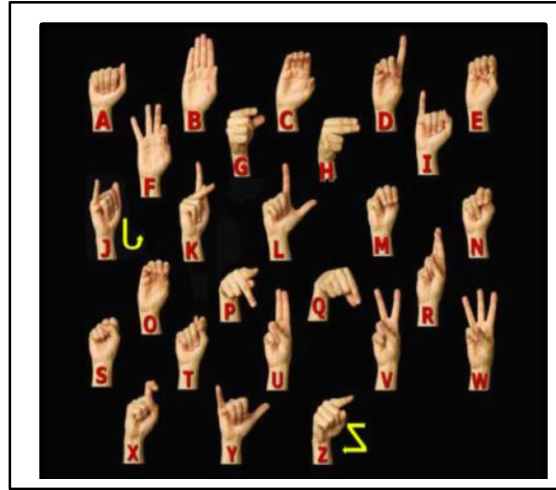


Figure 2.4 The ASL Gesture Set
(Kulkarni & Lokhande, 2010)

2.6.3 Hand Gesture-Based Graphical User Interface

For a more classic interaction, the hand gesture can be used to draw or replace the mouse. A draw-board is a drawing tool that uses hand motions to control visual drawing. The cursor on the screen is controlled by the position of the hand. Similarly, fingertip motion can also be used to draw. Hand postures are viewed as commands by the computer. The Smart Canvas system (Mo et al., 2005) is an intelligent desk system that allows a user to perform freehand drawing on a desk or any similar surface using gestures. Hand gestures can be applied to character drawing as well. For example in Figure 2.5, a user switches from the draw mode to menu mode by extending the thumb. Another main feature of hand gesture interaction is being able to replace the mouse as the “mouse clicking” event can be modelled in numerous different ways (Just, 2006). For example, in (Iannizzotto, Villari, & Vita, 2001), the contact between the thumb and index fingers correspond to a mouse-click event.

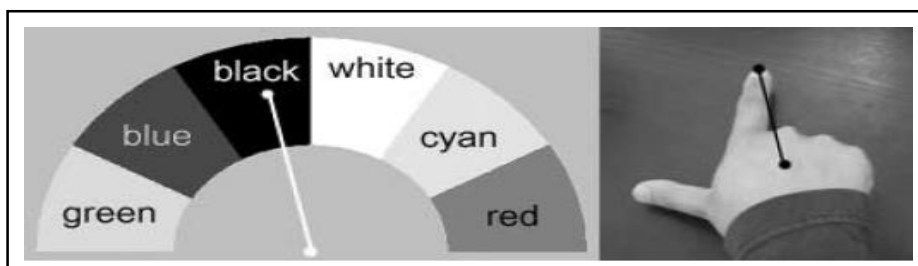


Figure 2.5 Menu Items are displayed in a pie shape; the thumb is extended to switch from Draw Mode to Menu Mode (Mo, Lewis, & Neumann, 2005)

2.6.4 Robotics

Using hand gestures is one of the few methods used in tele-robotic control (Figure 2.6). This type of communication provides an expressive, natural, and intuitive technique for humans to control robotic systems to perform specific tasks. One advantage of using hand gestures is that it is a natural means of sending geometrical information to the robot, such as left, right, up, and down hand gestures. The gestures may represent a single command, a sequence of commands, a single word, or a phrase (Wachs, Kartoun, Stern, & Edan, 2002).



Figure 2.6 Human –Robot Interaction (Kosuge & Hirata, 2004)

The use of hand gestures in human-robot interaction is a formidable challenge because the environment contains a complex background, dynamic lighting conditions, a deformable human hand shape, and a real-time execution requirement. In addition, the system is expected to be independent of the user and the device so that any user can use it without the need to wear a special device (Wachs et al., 2002).

2.7 Gesture Recognition Techniques

Gesture recognition can be subdivided into two main tasks: the recognition of gestures (dynamic gestures) and the recognition of postures (static gestures) (Just, 2006). In this thesis, however, we only consider the techniques that are used to recognize static hand gestures. Hand posture recognition (HPR) can be accomplished by using a template matching, geometric feature classification, neural network, or any other standard pattern recognition technique that classifies poses. Meanwhile, hand gesture recognition (HGR) requires the consideration of temporal events. HGR is a sequence processing problem that can be accomplished by using finite state machines, dynamic time warping (DTW), and hidden Markov models (HMM) (Just, 2006). These techniques are described below.

2.7.1 Template Matching

One of the simplest and earliest approaches to pattern recognition is based on template matching. Matching is a generic operation in pattern recognition that is used to determine the similarities between two entities (points, curves, or shapes) of the same type. In template matching, a template (typically a 2D shape) or a prototype of the pattern to be recognized is available. The pattern to be recognized is matched against the stored template while considering all allowable poses (translation and rotation) and scale changes (Jain, Duin, & Jianchang, 2000).

Consider the 3 x 3 template illustrated in Figure 2.7(a). This template represents the pattern to be detected within the total image array (Figure 2.7(b)). The template matching process commences with the template on the top left position (Figure 2.7(c)) when the correlation between the template and the array can be quantified by summing the products of the corresponding pixel values within the template and image array, respectively. The value 8 is then sorted in the correlation array (Figure 2.7(e)). This process is repeated until the template is scanned across the entire image array. The

resulting correlation array shows that the highest correlation value is 8. Therefore, the position of occurrence of the object as defined by the template is presumably at the first template position. A perfect match, which would have been signified by a correlation value of 9, was not achieved. The similarity measure, which is often a correlation, can be optimized based on the available training set. Template matching is a computationally demanding process, but the availability of faster hardware has now made this approach more acceptable. Although effective in certain application domains, the rigid template matching previously described has a number of disadvantages. For instance, it would fail if the patterns were distorted because of changes in the imaging process viewpoint or large intra-class variations among the patterns (Jain et al., 2000).

2.7.2 Hidden Markov Models (HMMs)

HMM is a powerful statistical tool for modeling generative sequences that can be characterized by an underlying process generating an observable sequence. HMMs have been applied in several areas of signal processing and speech processing. Moreover, HMMs have been applied with success to low-level natural language processing tasks such as part-of-speech tagging, phrase chunking, and extracting target information from documents. The mathematical theory of Markov processes were named after Markov during the early 20th century, but the theory of HMMs were developed by Baum and his colleagues in the 1960s (Blunsom, 2004).

HMM is widely used in speech recognition (Manabe & Zhang, 2004) . However, HMM has also been recently employed in human motion recognition because of the similarities between speech recognition and temporal (dynamic) gesture recognition. In addition, HMM has been used to model the state transition among a set of dynamic models (Wu & Huang, 2001).

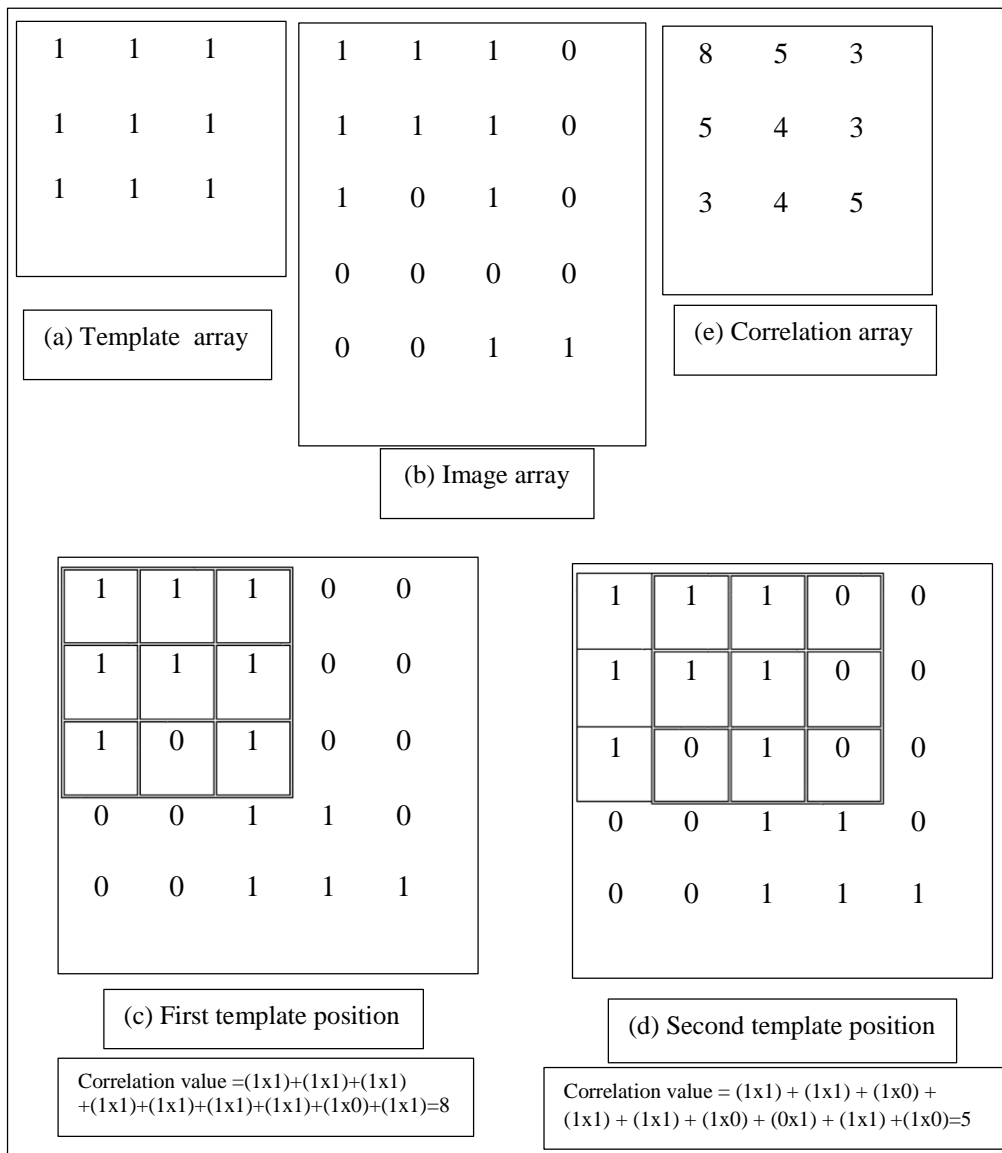


Figure 2.7 Template Matching

The matching process moves the template image to all possible positions in a larger source image and computes a correlation value that indicates how well the template matches the image in that position.

HMMs have been used extensively in gesture recognition. For instance, HMMs were used for ASL recognition by tracking the hands based on color. An HMM consists of a set (S) of n distinct states such that $S = \{s_1, s_2, s_3 \dots s_n\}$, which represents a Markov stochastic process. A stochastic process is considered a Markov process if the conditional probability of the current event given all past events depends only on the j

most recent events. In particular, if the current event only depends on the previous event, then this event is considered a first order Markov process, and the HMM is considered a first order HMM. HMM is considered hidden if the stochastic variable associated with the states is not observable and the observation is a probabilistic function of the state (Shet et al., 2004). Each state has a probability distribution over the possible output tokens. Therefore, the sequence of tokens generated by HMM provides information regarding the sequence of the states. In the context of gesture recognition, the observable parameters are estimated by recognizing the posture (tokens) in the images. Thus, HHMs are widely used for gesture recognition because gestures can be recognized as a sequence of postures (Zabulis, Baltzakis, & Argyros, 2009).

2.8 Summary

Hand gestures which are performed by one or two hands can be categorized according to their applications into different categories including conversational, controlling, manipulative and communicative gestures (Kjeldsen, 1997).

Generally, hand gesture recognition aims to identify specific human gestures and use them to convey information. The process of hand gesture recognition composes mainly of four stages: (1) hand gesture images collection, (2) gesture image preprocessing using some techniques including edge-detection, filtering and normalization, (3) capture the main characteristics of the gesture images using feature extraction algorithms, and (4) the evaluation (or classification) stage where the image is classified to its corresponding gesture class.

There are many methods that have been used in the classification stage of hand gesture recognition such as Artificial Neural Networks, Template Matching, Hidden Markov Models and Dynamic Time Warping (DTW) although DTW is predominantly used for dynamic gestures.

3.0 REVIEW OF IMAGE PROCESSING AND NEURAL NETWORKS

3.1 Introduction

The study of gesture recognition and gesture-based interaction is increasingly becoming an attractive research subject in HCI as more applications are using natural means, such as hands, fingers and voice, to interact with computers and smart devices. The implementation of a gesture-based HCI requires the capturing of necessary information to determine what gesture is being performed by the user. Recognizing gestures is a complex task that primarily involves two phases: first, extracting features, which characterises the gestures from the images, and second, using a suitable classifier (in this study, neural network) to assign each gesture to its respective class based on the extracted features. These two phases involve numerous techniques and methods that fall under these two following areas:

- Image processing
- Neural network

The purpose of this chapter is to introduce some basic concepts and methods related to these two areas that are employed in our methodology.

3.2 Image Processing

Computer imaging can be defined as the acquisition and processing of visual information by the computer (Umbaugh, 1997). Computer imaging can be divided into two main categories, namely, Computer vision and Image processing (Umbaugh, 1997). The processed (output) images in computer vision applications are for computer use, whereas the output images in image processing applications are for human consumption, that is, the images are to be examined and acted upon by people. The major topics in the field of image processing include image restoration, image enhancement, and image compression (Umbaugh, 1997).

3.2.1 Segmentation

Segmentation is the initial stage for any recognition process in which the acquired image is broken down into meaningful regions or segments. The segmentation process is only concerned with partitioning the image and not with what the regions represent. In the simplest case (binary images), only two regions exist: a foreground (object) region and a background region. In gray level images, several types of region or classes may exist within the image. For example, when a natural scene is segmented, regions of clouds, ground, buildings, and trees may exist (Awcock & Awcock, 1995).

Segmentation subdivides an image into its constituent parts, the level of which depends on the problem being solved. Segmentation should be stopped when the objects of interest in an application have been isolated (Gonzalez & Woods, 2002). The two main approaches to segmentation are as follows:

1. Pixel-based or local methods: is the process of partitioning a digital image into multiple segments or groups of pixels, also known as super pixels. This category includes edge detection and boundary detection (Basavaprasad & Ravi, 2014).
2. Region-based or global approaches: which may attempt to find the object boundaries and then locate the object itself by fulfilling them. This category includes region merging and splitting and threshold (Awcock & Awcock, 1995).

3.2.1.1 Thresholding

A simple image segmentation problem occurs when an image contains an object that has homogeneous intensity and a background with different intensity levels (Pitas, 2000). This problem can be overcome by employing thresholding techniques, such as partitioning the image histogram using a single threshold T . Segmentation is then

accomplished by scanning the image pixel by pixel and labelling each pixel as an object or a background depending on whether the gray level of that pixel is greater or less than the value of the threshold T (Gonzalez & Woods, 2002).

$$g(x, y) = \begin{cases} 1 & \text{if } (x, y) > T \\ 0 & \text{otherwise} \end{cases} \dots \dots \dots (3.1)$$

3.2.1.2 Noise Reduction

Spatial filters can be effectively used to remove various types of noise in digital images. These spatial filters typically operate on small neighbourhoods ranging from (3 x 3) to (11 x 11). Numerous spatial filters are implemented with convolution masks¹, because a convolution mask operation provides a result that is a weighted sum of the values of a pixel and its neighbours. This result is called a linear filter. The mean filters are essentially averaging filters; they operate on local groups of pixels called neighbourhoods and replace the centre pixel with the average of the pixels in this neighbourhood. This replacement is performed with a convolution mask (Umbaugh, 1997). The median filter is a nonlinear filter. A nonlinear filter gives a result that cannot be obtained by the weighted sum of the neighbourhood pixels as was performed with the convolution masks (Umbaugh, 1997). However, the median filter does operate on a local neighbourhood after the size of the local neighbourhood is defined. The centre pixel is replaced by the median or the centre value present among its neighbours, rather than by the average (Umbaugh, 1997). The median filter disregards extreme values (high or low) and does not allow such values to influence the selection of a pixel value that is truly representative of the neighbourhood. Therefore, the median filter is excellent in removing isolated extreme noise pixels (often known as “salt and pepper” noise) while substantially retaining spatial detail. However, its performance deteriorates

¹ Convolution mask is the application of a mask to an input image produces and output image on the same size as the input (Trucco & Verri, 1998).

when the number of noise pixels is more than half the number of pixels in the window (Awcock & Awcock, 1995).

3.2.1.3 Edge Detection

Edges are basic image features that carry useful information regarding the object boundaries. This information can be used for image analysis, object identification, and image filtering applications (Pitas, 2000). Edge detection methods are used as the first step in the line detection process. Edge detection methods are also used in finding complex object boundaries by marking the potential edge points that correspond to the places in an image where changes in brightness occur. After these edge points are marked, they are merged to form lines and object outlines. Edge detection operations are based on the idea that the edge information in an image can be found by examining the relationship between a pixel and its neighbours. If a pixel's gray level value is similar to those around it, then this pixel is probably not an edge point. By contrast, if a pixel has neighbours with widely varying gray levels, then this pixel may represent an edge point. Thus, an edge is defined by a discontinuity in gray level values. Ideally, an edge is caused by changes in colour or texture or by the specific lighting conditions present during the image acquisition process (Umbaugh, 1997).

A) Sobel Operator

The Sobel operator is recognized as one of the best "simple" edge operators that utilizes two (3 x 3) masks (Awcock & Awcock, 1995). The Sobel edge detection masks search for the horizontal and vertical directions and then combine this information into a single metric. The masks are given as follows (see Figure 3.1):

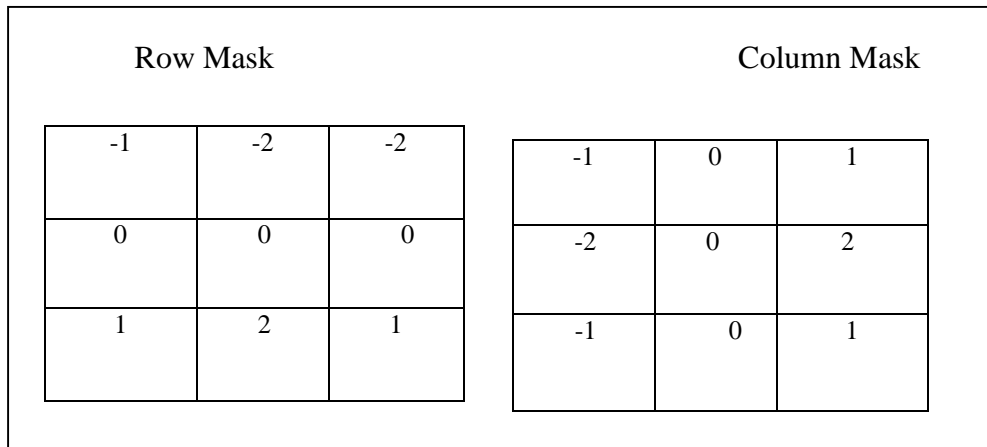


Figure 3.1 Sobel Operation Mask

Each mask is convolved with the image. Two numbers exist at each pixel location, namely, S_1 , which corresponds to the result from the row mask, and S_2 , which is from the column mask. These numbers are used to compute two metrics, namely, the edge magnitude and the direction, which are defined as follows (Umbaugh, 1997):

$$\text{Edge Magnitude} = \sqrt{S_1^2 + S_2^2} \dots \dots \dots (3.2)$$

$$\text{Edge Direction} = \tan^{-1} \left(\frac{S_1}{S_2} \right) \dots \dots \dots (3.3)$$

B) Prewitt Operator

The Prewitt operator is similar to the Sobel operator, but with different mask coefficients. The masks are defined as follows (see Figure 3.2):

Row Mask			Column Mask		
-1	-1	-1	-1	0	1
0	0	0	-1	0	1
1	1	1	-1	0	1

Figure 3.2 Prewitt Operation Mask

Each mask is convolved with the image. Two numbers exist at each pixel location, namely, P_1 and P_2 , which correspond to the row and the column masks, respectively. These numbers are used to compute two metrics, namely, the edge magnitude and the direction, which are defined as follows (Umabaugh, 1997):

$$\text{Edge Magnitude} = \sqrt{P_1^2 + P_2^2} \dots \dots \dots (3.4)$$

$$\text{Edge Direction} = \tan^{-1}\left(\frac{P_1}{P_2}\right) \dots \dots \dots (3.5)$$

C) Laplacian Operator

The Laplacian operator is a second order derivative operation that has zero crossing (i.e., transition from positive to negative and vice versa) (Gonzalez & Woods, 2002; Pitas, 2000). The Laplacian masks are rotationally symmetric, which means that the edges at all orientations contribute to the result. They are applied by selecting one mask and convolving it with the image. The sign of the result (positive or negative) from two adjacent pixel locations provides the directional information and indicates which side of the edge is brighter (Umabaugh, 1997) (see Figure 3.3).

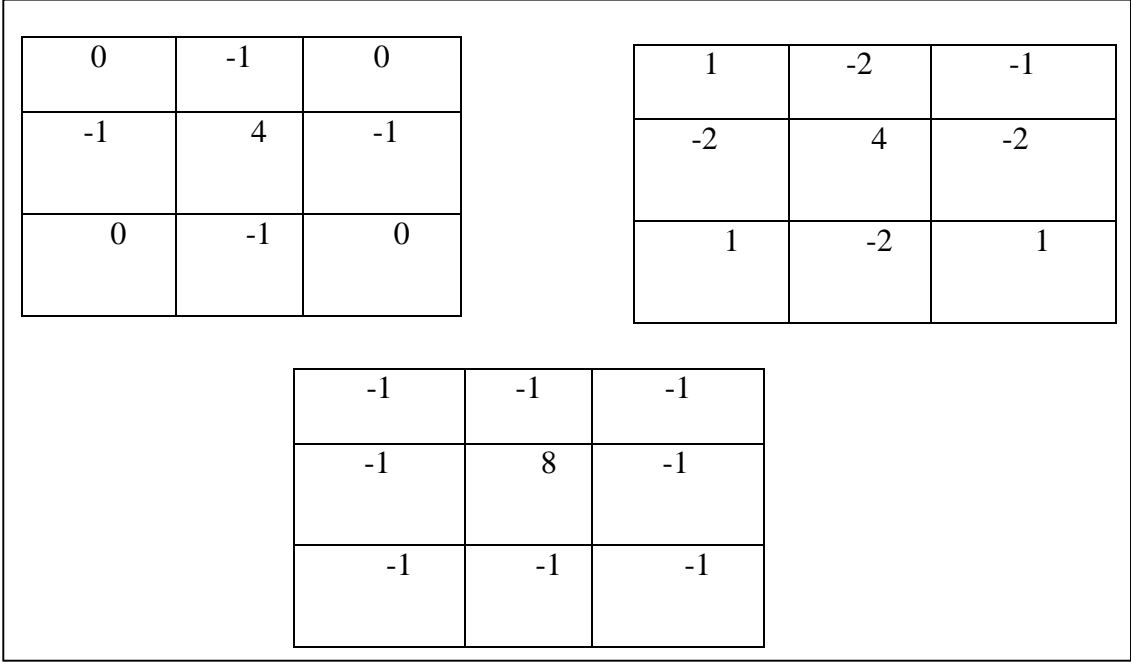


Figure 3.3 Laplacian Operation Mask

3.2.1.4 Coordinate Normalization

The idea of coordinate normalization is to map the scaled hand image coordinates to the standard size ranging between [-1, +1] (Musa, 1998). The purpose of this step is to keep the domain of the image coordinates fixed irrelevant to the original size. The condition of keeping the domain of the coordinates within the limited boundaries will effectively satisfy the convergence of higher ordered moments. Thus, the scaled image coordinates (X, Y) will be transformed into the normalized set (X_n, Y_n) which can be considered as the “standard” version of the original coordinate (X, Y). By using the center of the image, each pixel coordinates (X, Y) values are mapped to the domain [-1, +1] which can be performed using the following equations:

$$X_n = \left(\frac{2}{W - 1} * X \right) \dots \dots \dots (3.6)$$

$$Y_n = (2/(H - 1) * Y) - 1 \dots \dots \dots (3.7)$$

Where H and W are the height and width of the scaled image respectively (Musa, 1998).

3.2.2 Feature Extraction

Feature extraction is part of the data reduction process and is followed by feature analysis. One of the important aspects of feature analysis is determining exactly which features are important (Umbaugh, 1997). Feature extraction is a complex problem in which the whole image or the transformed image is often taken as the input. The goal of feature extraction is to find the most discriminating information in the recorded images. Feature extraction operates on two-dimensional image arrays but produces a list of descriptions or a “feature vector” (Awcock & Awcock, 1995; Huang, 1998).

Mathematically, a feature is an n -dimensional vector with its components computed by some image analysis. The most commonly used visual cues are colour, texture, shape, spatial information, and motion in video. For example, colour may represent the colour information in an image, such as colour histogram, colour binary sets, or colour coherent vectors. The n components of a feature may be derived from one visual cue or from composite cues, such as the combination of colour and texture (Huang, 1998).

The selection of good features is crucial to gesture recognition because hand gestures are rich in shape variation, motion, and textures. Although hand postures can be recognized by extracting some geometric features such as fingertips, finger directions, and hand contours, these features are not always available and reliable because of self-occlusion and lighting conditions. Moreover, although a number of other non-geometric features are available, such as colour, silhouette, and textures, these features are inadequate for recognition. Explicitly specifying features is not easy. Therefore, whole

images or transformed images are taken as input, and features are selected implicitly and automatically by the classifier (Wu & Huang, 1999).

3.2.2.1 Contour Detection

Although contour detection in real images is a fundamental problem in many computer vision tasks (Joshi & Sivaswamy, 2006), the usefulness of contour detection in various applications has been well-established and demonstrated, particularly in image analysis and scene understanding (Wang et al., 2006). Image edge detection is also becoming an important part of image segmentation. The difference between edges and contours is that edges are characterized by variations in the intensity level in a gray level image, whereas contours are salient coarse edges that belong to objects and region boundaries in the image (Joshi & Sivaswamy, 2006).

Numerous types of edge detectors are available, such as Sobel and Canny, and most of them are based on derivative operators that provide high response at the contour points and low response in homogeneous areas (Beghdadi & France, 1999). The digital gradient operator is the oldest and simplest edge detector (Beghdadi & France, 1999).

Binarisation

A binary contour map can be constructed by using a standard procedure such as non-maxima suppression followed by hysteresis thresholding (Canny, 1986). The gradient magnitude $M(x, y)$ and orientation map $\theta(x, y)$ specify the local strength and local edge direction, respectively. Non-maxima suppression is the process of thinning the regions at edge locations where $M(x, y)$ is non-zero by retaining only the local maxima in the gradient direction. To generate candidate contours, two virtual neighbors are defined at the intersections of the gradient direction with a 3×3 sampling grid. The gradient magnitude for these neighbors is then interpolated from adjacent pixels. The central pixel is retained for further processing only if its gradient magnitude is the largest of the

three values; otherwise, the magnitude of the central pixel is changed to zero. The final contour map is computed from the candidates by hysteresis thresholding, which involves threshold values t_i and t_h , $t_i < t_h$. All pixels with $M(x, y) \geq t_h$ are retained for the final contour map, whereas all pixels with $M(x, y) \leq t_i$ are discarded. The pixels with $t_i > M(x, y) > t_h$ are retained only if they already have at least one neighbor in the final contour map (Joshi & Sivaswamy, 2006).

3.2.2.2 Invariant Features

Features associated with images are called “invariant” if they are not affected by certain changes of the object view point. Invariant features are independent of modifiers such as translation, scaling, rotation, and light conditions. Ideally, invariant features should recognize objects whose geometry can change either because the object is moving in relation to the articulated camera or because different viewpoints cause different patterns in 2D images. These modifiers are usually not independent of each other; thus, they often happen simultaneously. Truly pure invariant features are lacking, and features that are more or less robust to one or more modifiers exist (Barczak & Dadgostar, 2005). The need for invariant features rises in many practical problems, such as speech recognition, speaker recognition and image recognition. For the last example (i.e. image recognition), features should be invariant under rotation, translation, scaling, and the illumination angle used in generating the image (Abd Alrazak, 2004).

3.2.2.3 Complex Moments (CMs)

The notation of CM was introduced by Abu-Mostafa and Psaltis (1985) as a simple and straightforward method of deriving moment invariants. The CM of order (m) is defined as (Abu-Mostafa & Psaltis, 1984):

$$C_m = \iint (x + iy)^m \mu(x, y) dx dy \dots \dots \dots (3.8)$$

Where $i = \sqrt{-1}$ and $\mu(x, y)$ is the real image intensity function.

A moment invariant is a set of moment values extracted from the image data in such a way that their values are invariant to the rotation of the image data. Moreover, the value of a CM could be considered a moment invariant if that value can be computed from a group of CMs for the same object at different resolutions (Musa, 1998). Thus, a moment invariant can be used as a feature for the classification and recognition of an object. In turn, CMs, which are simple and relatively powerful in providing the analytic characteristics of moment invariants, have been proposed as a solution to different pattern recognition problems. CMs have two parts, namely, real and imaginary parts. However, the computation of their values decomposes into two directions: the x -axis moment, which represents the direction of the real part, and the y -axis moment for the direction of the imaginary part (Musa, 1998). Moment sets can offer a powerful description of the geometrical distribution of the material within any region of interest. The low order of CMs has meanings that are significantly relevant to a number of well-known physical quantities (Musa, 1998).

1. Zero-order moments represent the total mass of the image.
2. First-order moments together with zero-order moments assign the center of the mass of the image.
3. Second-order moments represent moment of inertia.
4. Third-order and fourth-order moments are used for computing statistical quantities such as skews and kurtosis, respectively.

Although higher n th-order moments provide additional statistical and structural information about an image, these moments are computationally more expensive. The

computation of a CM should involve the calculation of its real and imaginary components. The n th-order CM (M_i), for the hand image of size $(n \times m)$ is then calculated according to the following equation (Musa, 1998):

$$M_i = \sum_{x=0}^{n-1} \sum_{y=0}^{m-1} Z_n^i a(x, y) \dots \dots \dots (3.9)$$

where (i) indicates the order of the moment;

$Z_n = X_n + iY_n$ is a complex number;

$a(x, y)$ represents a pixel value at the position (x, y) , which may be an ON (i.e., its value is 1) or OFF (its value is 0)) pixel.

The calculation of complex moments may require a long computation time. The following relation is used to reduce the computation time (Musa, 1998):

$$Z^n = Z^{n-1} \cdot Z \dots \dots \dots (3.10)$$

Where $Z = x + iy$ is the complex number.

When the real and imaginary parts of $(Z^n$ and $Z^{n-1})$ are assumed as complex numbers, they could be written as (Musa, 1998):

$$Z^n = R_{n+i} I_n \dots \dots \dots (3.11)$$

And

$$Z^{n-1} = R_{n-1} + iI_{n-1} \dots \dots \dots (3.12)$$

By considering the case Z^0 and Z^1 , we can obtain

$$R_0 = 1 ; I_0 = 0$$

$$R_1 = x; I_1 = y.$$

The values of Z_n , Z_{n-1} and Z_1 are then substituted in Equation (3.8):

$$R_1 = R_{n-1} X - I_{n-1} Y \dots \dots \dots (3.13)$$

$$I_1 = I_{n-1} X + R_{n-1} Y \dots \dots \dots (3.14)$$

These equations indicate that the knowing components (Z^{n-1}) will be directly used to compute the Z^n component (Musa, 1998).

3.3 Artificial Neural Networks

An artificial neural network is an information processing system that has certain performance characteristics similar to biological neural networks. Artificial neural networks have been developed as generalizations of mathematical models of human cognition or neural biology on the basis of the following assumptions (Fausett, 1994):

1. Information processing occurs at many simple elements called neurons;
2. Signals are passed between neurons over connection links;
3. Each connection link has an associated weight, which multiplies the signal transmitted in a typical neural network;
4. Each neuron applies an activation function (usually nonlinear) to its net input (sum of weighted input signals) (Fausett, 1994; Picton, 2000).

3.3.1 Artificial Neuron

A neuron is an information processing unit that is fundamental to the operation of the neural network. Figure 3.4 shows the model of a neuron consisting of three basic elements:

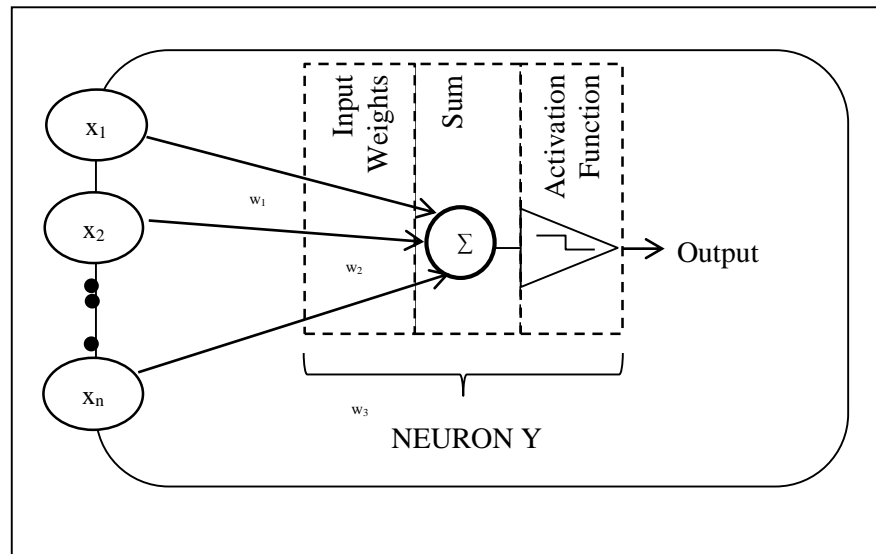


Figure 3.4 Simple Artificial Neuron (S. Haykin, 1998)

1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own; specifically, signal x_j at the input of synapse j connected to neuron k is multiplied by synaptic weight w_k ;
2. A linear combiner function for summing the input signals weighted by the respective synapses of the neuron; the operations described here constitute a linear combination;
3. An activation function for limiting the amplitude of the output of a neuron; the normalized amplitude range of the output of a neuron is typically written as the closed unit interval $[0,1]$ or alternatively as $[-1, 1]$ (S. Haykin, 1998).

3.3.2 Types of Activation Functions

3.3.2.1 Identity Function

The basic operation of an artificial neuron involves summing its weighted input signal and applying an output or activation function. This function is the identity function for the input unit (see Figure 3.5).

$$f(x) = x \text{ for all } x \dots\dots\dots (3.15)$$

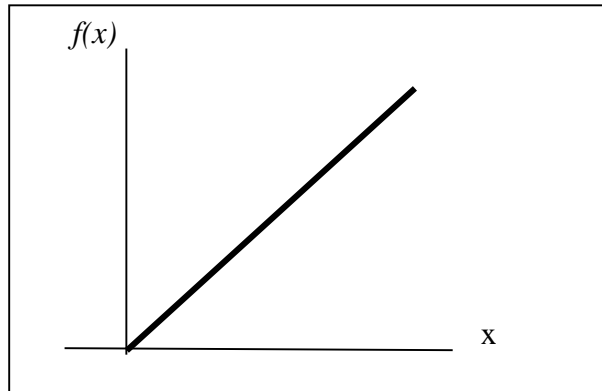


Figure 3.5 Identity Function

3.3.2.2 Binary Step Function (With Threshold θ):

Single layer networks often use a step function to convert the net input, which is a continuously valued variable, into an output unit, which is a binary (1 or 0) or a bipolar (1 or -1) signal, as shown in Figure 3.6. The binary step function is also known as the threshold function (Fausett, 1994):

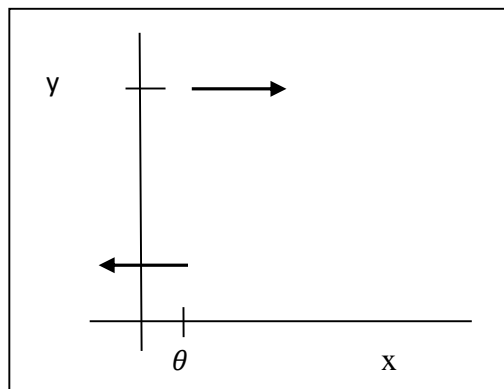


Figure 3.6 The Binary Step Function

$$f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases} \dots\dots\dots (3.16)$$

3.3.2.3 Binary Sigmoid

Sigmoid functions (S-shaped curves) are useful activation functions, and logistic and hyperbolic tangent functions are the most common activation functions. These functions are particularly useful in neural networks trained by back-propagation. If the range of the function is between 0 and 1, then this function is called a “binary sigmoid” (see Figure 3.7). However, if the range of the function is between -1 and 1 , then this function is called a “bipolar sigmoid” (see Figure 3.8) (Fausett, 1994).

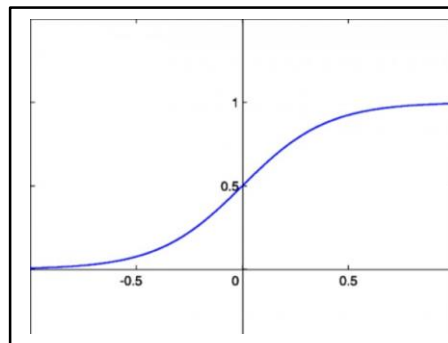


Figure 3.7 Binary Sigmoid Function

$$f(x) = \frac{1}{1 + \exp(-\partial x)} \dots \dots \dots (3.17)$$

3.3.2.4 Bipolar Sigmoid

$$g(x) = 2f(x) - 1 = \frac{2}{1 + \exp(-\partial x)} - 1 \dots \dots \dots (3.18)$$

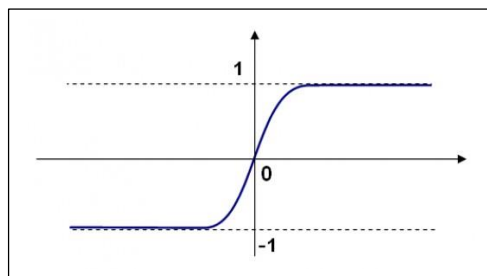


Figure 3.8 Bipolar Sigmoid Function

3.3.3 Learning Paradigms

Two modes of learning exist, namely, supervised and unsupervised learning. The following is a description of these modes of learning.

3.3.3.1 Supervised Learning

Supervised learning, which is a commonly used training method, is based on a system aiming to predict outcomes for known examples. Supervised learning compares its predictions to the target answer and “learns” from its mistakes. The data start as inputs to the input layer neurons. The neurons pass the inputs along the next nodes. Weights or connections are applied as inputs and are passed along. When the inputs reach the next node, the weights are summed and are either intensified or weakened. This process continues until the data reach the output layer where the model predicts an outcome. In a supervised learning system, the predicted output is compared with the actual output for that case. If the predicted output is equal to the actual output, then no change is made to the weights in the system. However, if the predicted output is higher or lower than the actual outcome in the data, the error is propagated back through the system and the weights are adjusted accordingly. This backward feeding of the error through the network is called “back-propagation.” Both the multi-layer perceptron and the radial basis function are supervised learning techniques. The multi-layer perceptron uses back-propagation, whereas the radial basis function uses a feed-forward approach that trains on a single pass of data (Symeonidis, 1996).

3.3.3.2 Unsupervised Learning

Neural networks, which use unsupervised learning, are most effective for describing rather than for predicting data. A neural network does not show any output or answers as part of the training process. In fact, no concept of output fields exists in this type of system. The Kohonen network (Kohonen & Maps, 1995), which is a primary

unsupervised technique, and other unsupervised neural systems are mainly used for cluster analysis to group “like” cases together. Cluster analysis benefits from this type of neural network because cluster analysis does not require initial assumptions on what constitutes a group or how many groups exist. This system starts with a clean slate and is not biased about which factors should be the most important (Symeonidis, 1996).

3.3.4 Neural Networks Architectures

The arrangement of neurons into layers and the connection patterns within and between layers is called the “network architecture.” Neural networks are often classified as either single layer or multilayer (Picton, 2000).

3.3.4.1 Single-Layer Neural Networks

A single-layer network has one layer of connection weights. The units can often be distinguished as input units, which receive signals from the outside world, and as output units, from which the response of the network can be read. In a typical single-layer network (see Figure 3.9), the input units are fully connected to the output units, but not to other input units. Similarly, the output units are not connected to other output units (Fausett, 1994).

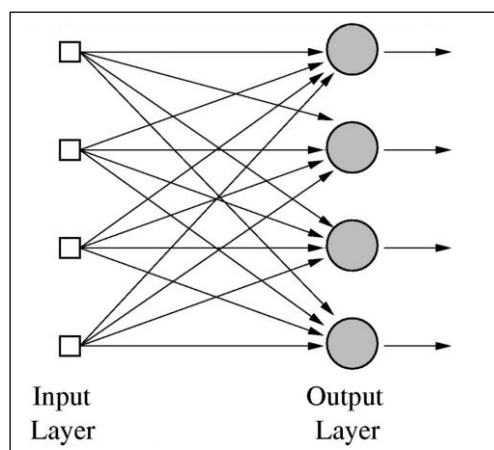


Figure 3.9 Single Layer Neural Network

3.3.4.2 Multilayer Network

A multilayer network is composed of one or more layers (or levels) for nodes (the so-called hidden units) between the input and output units (see Figure 3.10). Typically, a layer of weights exists between two adjacent levels of units (input, hidden, and output). Multilayer networks can solve more complicated problems than a single-layer network could. However, training the former may be more difficult than training the latter. Multilayer perceptron neural networks are useful for classification purposes (Fausett, 1994; Picton, 2000).

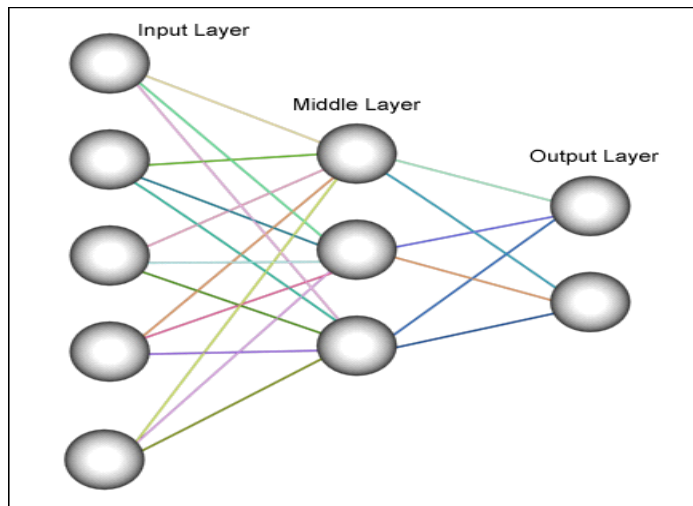


Figure 3.10 Three- Layer Neural Network

3.3.5 Back-Propagation Learning Algorithm

The error back-propagation process involves two passes through the different layers of the network: a forward pass and a backward pass (S. Haykin, 1998; Kinnebrock, 1995).

The algorithm is as follows (Fausett, 1994):

Step 0. Initialize weights (set to small random values);

Step 1. While stopping condition is false, do steps 2–9;

Step 2. For each training pair, do steps 3–8.

Feed-forward (Fausett, 1994):

Step 3. Each input unit ($X_i, i = 1, \dots, n$) receives input signal x_i and broadcasts this signal to all units in the layer above (the hidden units).

Step 4. Each hidden unit ($Z_j, j = 1, \dots, p$) sums its weighted input signals,

$$Z_{in} = V_{0j} + \sum_{i=1}^n X_i V_{ij} \dots \dots \dots (3.19)$$

v_{0j} : Bias on hidden unit j .

v_{ij} : Weight between input and hidden units; applies its activation function to compute its output signal,

$$Z_j = f(Z_{in_j}) \dots \dots \dots (3.20)$$

and send this signal to all units in the layer above (output units).

Steps5. Each output unit ($Y_k, k = 1, \dots, m$) sum its weighted input signals,

$$y_{in_k} = w_{0k} + \sum_{j=1}^p z_j w_{jk} \dots \dots \dots (3.21)$$

w_{0k} : Bias on output unit k .

w_{jk} : Weight between hidden unit and output unit

and applies its activation function to compute its output signal.

$$y_k = f(y_{in_k}) \dots \dots \dots (3.22)$$

Backpropagation of error:

Step6. Each output unit ($Y_k, k = 1, \dots, m$) receives a target pattern corresponding to the input training pattern, computes its error information term,

$$\delta_k = (t_k - y_k)f'(y_{in_k}) \dots \dots \dots (3.23)$$

Calculates its weight correction term (used to update w_{jk} later),

$$\Delta w_{jk} = \partial \delta_k z_j, \dots \dots \dots (3.24)$$

Calculates its bias correction term (used to update w_{0k} later),

$$\Delta w_{0k} = \partial \delta_k, \dots \dots \dots (3.25)$$

And sends δ_k to units in the layer below.

Step7. Each hidden unit ($Z_j, j = 1, \dots, p$) sums its delta inputs (from the unit in the Layer above),

$$\delta_{in_j} = \sum_{k=1}^m \delta_k w_{jk} \dots \dots \dots (3.26)$$

Multiplies itself to the derivative of its activation function to calculate its error information term,

$$\delta_j = \delta_{in_j} f'(z_{in_j}) \dots \dots \dots (3.27)$$

Calculates its weight correction term (used to update v_{ij} later),

$$\Delta v_{ij} = \partial \delta_j x_j \dots \dots \dots (3.28)$$

and calculates its bias correction term (used to update v_{oj} later),

$$\Delta v_{0j} = \partial \delta_j, \dots \dots \dots (3.29)$$

Update weights and biases:

Step8. Each output unit ($Y_k, k = 1, \dots, m$) updates its bias and weights ($j = 0, \dots, p$):

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \dots \dots \dots (3.30)$$

Each hidden unit ($Z_j, j = 1, \dots, p$) updates its bias and weights ($i = 0, \dots, n$):

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij} \dots \dots \dots (3.31)$$

Step9. Test stopping condition.

3.3.6 Advantages of Neural Computing

Neural networks offer analysts a variety of benefits (Symeonidis, 1996):

1. Artificial neural networks are a powerful technique for harnessing the information in the data and for making generalizations about this information. Neural networks learn to recognize patterns in the data set (S. S. Haykin, 2007).
2. The system is developed through learning rather than through programming. Programming is much more time consuming for the analyst and requires the analyst to specify the exact behavior of the model. Neural networks teach themselves the patterns in the data, freeing the analyst for more interesting work (Moghadassi, Parvizian, & Hosseini, 2009).
3. Neural networks are flexible in changing environments. Rule-based systems or programmed systems are limited to the situation for which they were designed; when conditions change, the rules are no longer valid. Although neural networks may take some time to learn a sudden drastic change, they excel at adapting to constantly changing information (Din, 2002).
4. Neural networks can build informative models where more conventional approaches fail. Neural networks can handle very complex interactions and can thus easily model data that are too difficult to model by using traditional approaches, such as inferential statistics or programming logic (Moghadassi et al., 2009). Neural networks perform at least as well as classical statistical modeling and outperform the latter on most problems. Neural networks build models that are more reflective of the structure of the data in significantly less time (Din, 2002).

3.4 Summary

Hand gesture recognition process involves several techniques and algorithms that fall under the areas of image processing and artificial neural networks.

The first phase deals with problems related to image processing, such as reducing noise by using filters, scaling, and break down the image into meaningful regions using segmentation techniques such as thresholding and edge detection methods. The next step is the application of feature extraction methods such as Hand Contour and Complex Moments to find the most discriminating information in the hand gesture images.

In the second phase, a powerful classification method called Artificial Neural Networks is selected to classify the images into their respective classes using the extracted feature vectors. This classification method is inspired by the characteristics of biological neural networks. ANN is generally divided based on the learning paradigm into two categories: supervised and unsupervised neural networks. The multi-layer perceptron which uses back-propagation learning method is one of the most used supervised neural networks.

4.0 PROPOSED STATIC HAND GESTURE RECOGNITION SYSTEM

4.1 Introduction

The overview of the hand gesture recognition system (as shown in Figure 1.1), consists of the following stages. The first stage is the hand gesture image capture stage where the images are taken using digital camera under different conditions such as scaling, translation and rotation.

The second stage is a pre-processor stage in which edge-detection, smoothing, and other filtering processes occur. In the next stage, the features of the images of hand gesture are extracted using two methods, namely, hand contour and complex moments.

The last stage is the classification using Artificial Neural Network (ANN), where the recognition rate is calculated for both hand contour-based ANN and complex moments-based ANN and comparison is carried out. The following is a description of these stages.

4.2 Hand gesture Image Capture

The construction of a database for hand gesture (i.e., the selection of specific hand gestures) generally depends on the intended application.

A vocabulary of six static hand gestures is made for HCI as shown in Figure 4.1.

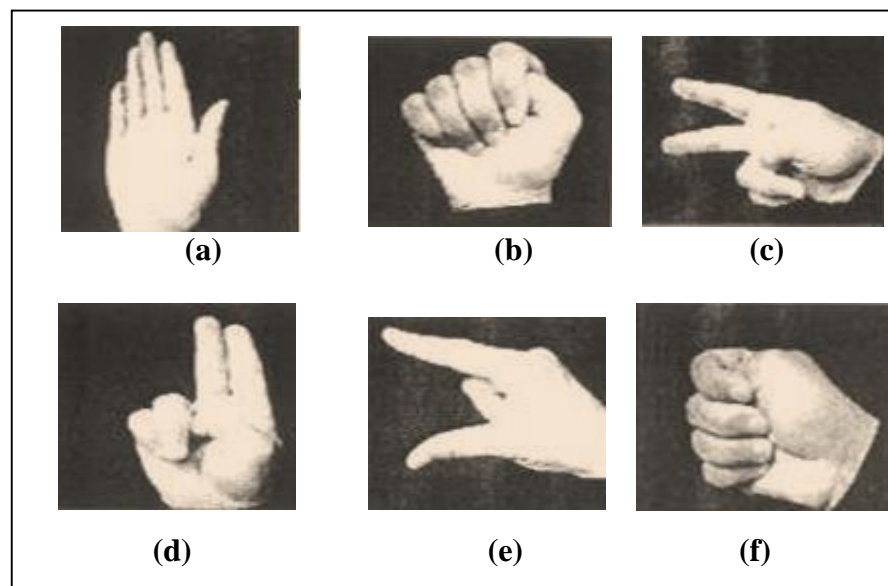


Figure 4.1 Six Static Hand Gestures: Open, Close, Cut, Paste, Maximize and Minimize

Each gesture represents a gesture command mode. These commands are commonly used to communicate and can thus be used in various applications such as a virtual mouse that can perform six tasks (Open, Close, Cut, Paste, Maximize, and Minimize) for a given application. The gesture images have different sizes.

In image capture stage, we used a digital camera Samsung L100 with 8.2MP and 3x optical zoom to capture the images and each gesture is performed at various scales, translations, rotations and illuminations as follows (see Figure 4.2 for some examples):

Translation: translation to the right and translation to the left.

Scaling: small scale (169×173), medium scale (220×222) and large scale (344×348).

Rotation: rotate 4 degree, rotate 2 degree and rotate -3 degree.

Original of lightning: original and artificial.

The database consists of 30 images for the training set (five samples for each gesture) and 56 images for testing with scaling, translation, and rotation effects. Employing relatively few training images facilitates the measurement of the robustness of the proposed methods, given that the use of algorithms that require relatively modest resources either in terms of training data or computational resources is desirable (Fei-Fei, Fergus, & Perona, 2007; Kanan & Cottrell, 2010). In addition, (Guodong & Dyer, 2005) considered that using a small data set to represent each class is of practical value especially in problems where it is difficult to get a lot of examples for each class.











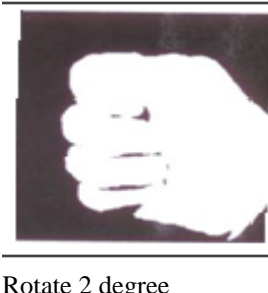

			
Original Open	Translation to the left	Translation to the right (artificial lightning)	Translation to the right
			
Original Maximize	Small Scale 169 * 173	Medium Scale 220 * 222	Large Scale 344 * 348
			
Original Minimize	Rotate 4 degree	Rotate 2 degree (artificial lightning)	Rotate -3 degree

Figure 4.2 Hand gestures images under different conditions

4.3 Pre-processing stage

The primary goal of the pre-processing stage is to ensure a uniform input to the classification network. This stage includes hand segmentation to isolate the foreground (hand gesture) from the background and the use of special filters to remove any noise caused by the segmentation process. This stage also includes edge detection to find the final shape of the hand.

4.3.1 Hand Segmentation

The hand image is segmented from the background. The segmentation process should be fast, reliable, consistent, and able to achieve optimal image quality suitable for the recognition of the hand gesture. Gesture recognition requires accurate segmentation. A thresholding algorithm is used in this study to segment the gesture image (see Figure 4.3). Segmentation is accomplished by scanning the image pixel by pixel and labelling each pixel as an object or a background depending on whether the gray level of that pixel is greater or less than the value of the threshold T .

This method can be mathematically expressed as follows:

$$g(x, y) = \begin{cases} 1 & \text{if } (x, y) > T \\ 0 & \text{otherwise} \end{cases} \dots \dots \dots (4.1)$$

where T is the threshold value, and (x, y) is coordinates of the threshold value point.

T is determined based on trial and error.

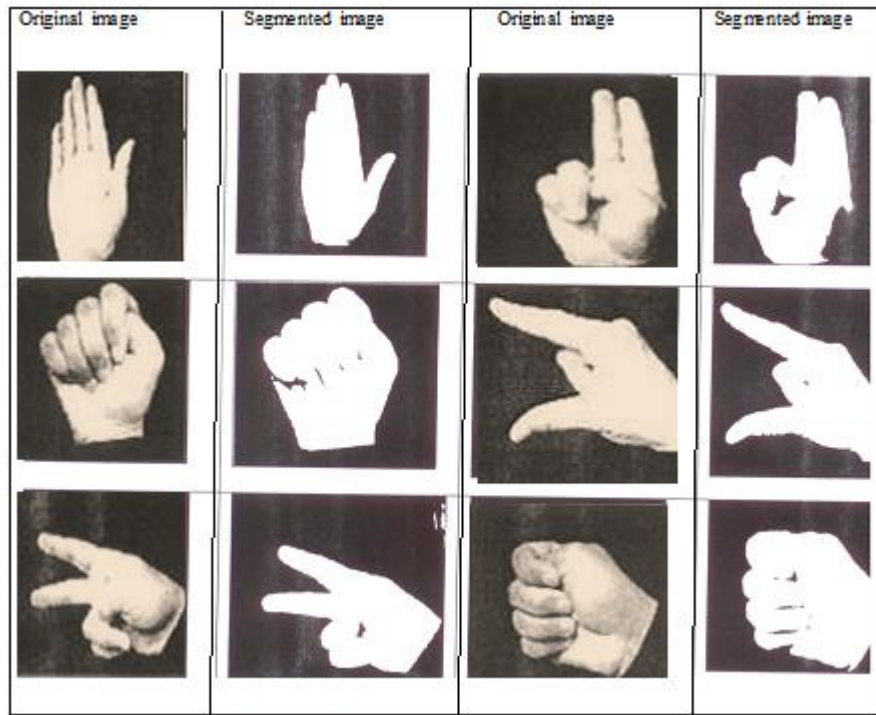


Figure 4.3 Hand gesture images before and after segmentation

4.3.2 Noise Reduction

Once the hand gesture image has been segmented, a special filter is applied to remove noise by eliminating all the single white pixels on a black background and all the single black pixels on a white foreground. To accomplish this goal, a median filter is applied to the segmented image as shown in Figure 4.4.

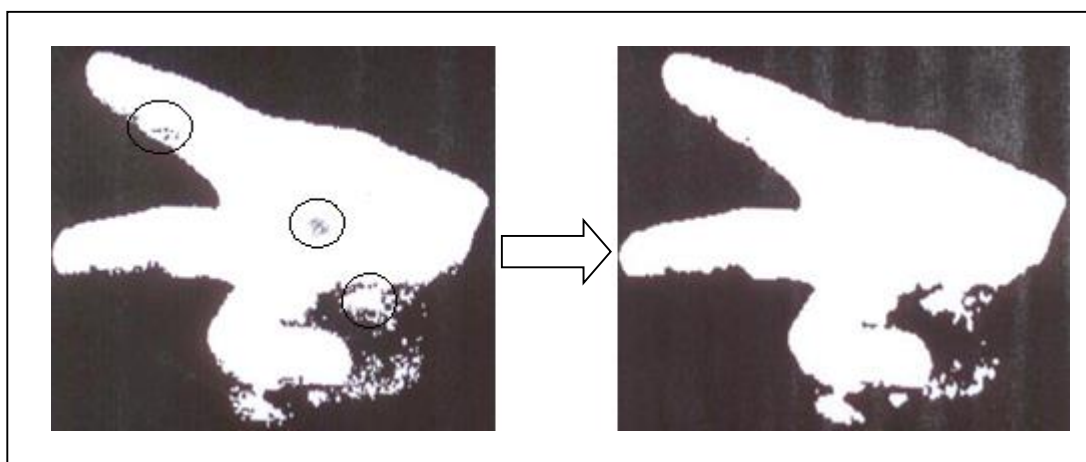


Figure 4.4 Median Filter Effect

The following example illustrates the way this algorithm works . For example, take a 3×3 window and compute the medium of the pixels in each window centered around (i, j) :

- 1- Sort the pixels into ascending order by gray level.
- 2- Select the value of the middle pixel as the new value for pixel (i, j) .

This process is illustrated in Figure 4.5.

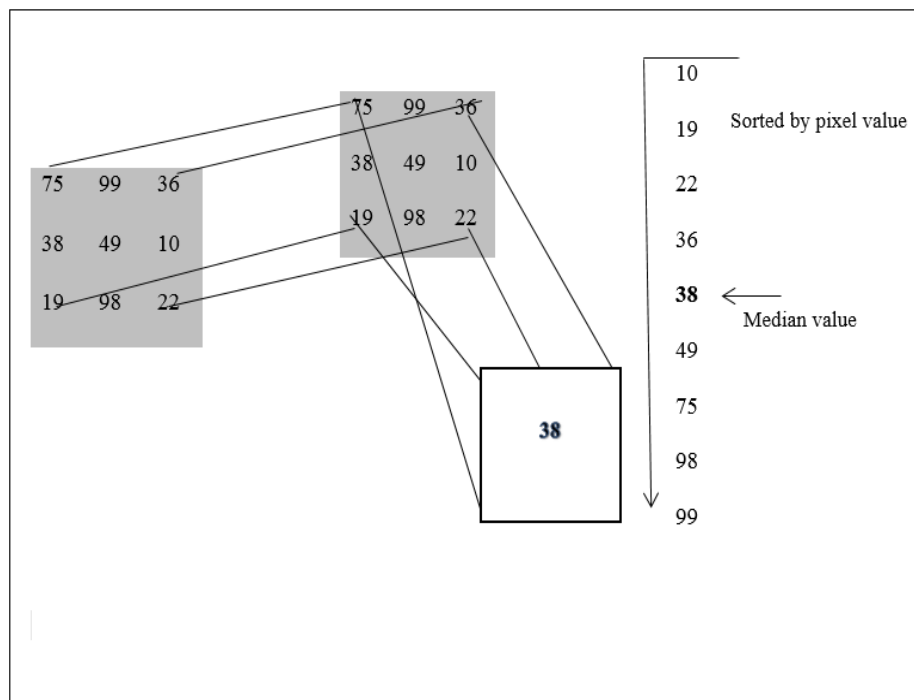


Figure 4.5 An example illustrating the medium filter using 3×3

In general, an odd-size neighbourhood is used for calculating the median. However, if the number of pixels is even, the median is taken as the average of the middle two pixels after sorting.

4.3.3 Edge Detection

To recognize static gestures, the model parameters derived from the description of the shape and the boundary of the hand are extracted for further processing. Sobel was

chosen for edge detection. Figure 4.6 shows some gesture images before and after edge detection operation using Sobel method. This method is explained in subsection 3.2.1.3 Edge detection.

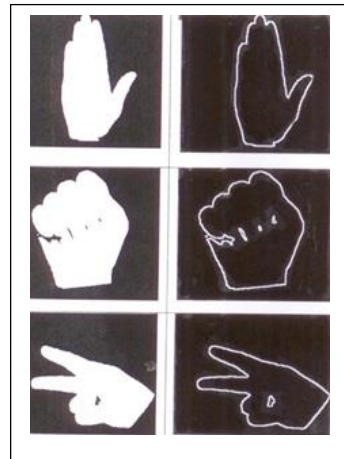


Figure 4.6 Sobel Edge detection for Open, Close and Cut

4.4 Gesture Feature Extraction methods

The objective of the feature extraction stage is to capture and distinguish the most relevant characteristics of the hand gesture image for recognition. The selection of good features can greatly affect the classification performance and reduce computational time. The features used must be suitable for the application and the applied classifier.

Two different feature extraction methods were used as part of the proposed hand gesture recognition algorithm:

1. Neural networks with hand contour;
2. Neural networks with hand complex moments.

These two extraction methods were applied in this study because they used different approaches to extract the features, namely, a boundary-based for hand contour and

region-based for complex moments. The use of different approaches may help us to identify strengths and weaknesses of each approach. Complex moments is adopted from Abu-Mostafa and Psaltis (1985) where the authors proposed a method for image recognition and we have applied this method specifically to the hand gesture recognition problem. The advantage of this method is its ability to extract invariant features that are independent of modifiers such as translation, scaling, rotation, and light conditions. There are other moments methods such as Hu moments (Ming-Kuei, 1962) but we chose to apply complex moments method on hand gesture recognition to investigate its suitability to solve this problem as, to our best of knowledge, no previous study has applied this method to hand gesture recognition systems.

In the Hand Contour method, we combined general and geometric features. The advantage of Boundary-based methods, which are commonly used for feature extraction, is that they are simple to implement and computationally fast (Du-Ming, 1997).

4.4.1 Hand Contour

4.4.1.1 Geometric Feature

Hand contour is one of the most commonly used geometric feature methods in static hand gesture recognition (Pavlovic et al., 1997). Contour detection can be implemented in a simple manner as follows:

1. Compute a gradient map; this gradient computation must be performed in two orthogonal directions by using the Sobel mask (see Figure 4.7).

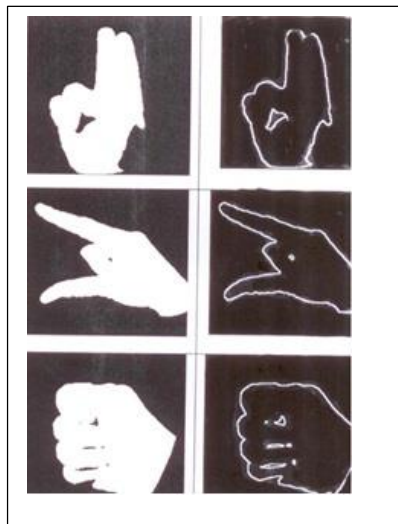


Figure 4.7 Sobel Operator Edge Detection

2. Incorporate the surrounding influence on the gradient map; the surrounding influence can be implemented as a convolution operation with an appropriate isotropic mask, as shown in Figure 4.8:

-5	-6	-5.5	-5	-5.5	-6	-5
-6	-5	-2	0.5	-2	-5	-6
-5.5	-2	0.4	0.4	0.4	-2	-5.5
-5	0.5	0.4	225	0.4	0.5	-5
-5.5	-2	0.4	0.4	0.4	-2	-5.5
-6	-5	-2	0.5	-2	-5	-6
-5	-6	-5.5	-5	-5.5	-6	-5

Figure 4.8 A 7 x7 Surround Mask

Convert the output of the second stage into binary by using a non maxima suppression followed by hysteresis thresholding (Canny, 1986).

Feature image scaling takes place once the contour map is produced. Feature image

scaling is a simple method that reduces the feature image size by selecting several rows and columns. Scaling is performed for all the images in the training set. The result is a feature image with 32 rows and 32 columns (see Figure 4.9). The coordinates (x_{nu}, y_{nu}) of the scaled image can be calculated as follows:

$$x_{nu} = \frac{x_p}{32} \dots \dots \dots (4.2)$$

$$y_{nu} = \frac{y_p}{H_p} \dots \dots \dots (4.3)$$

Where (x_p, y_p) is the coordinates of the pixel (i, j) in the original image, and W_p and H_p are the width and the height of the original image, respectively.

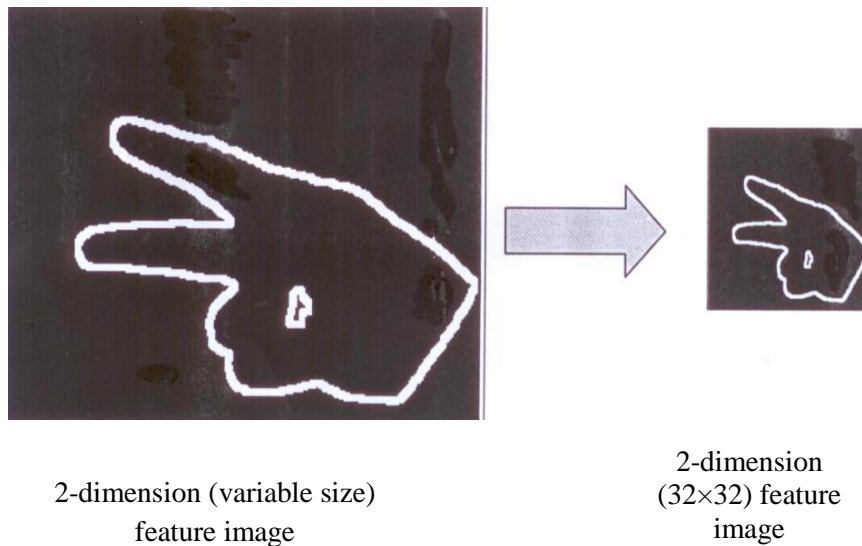


Figure 4.9 Image Scaling Effect 32×32

After feature image scaling, a feature gesture image must be prepared before entering the last stage, i.e., the classifier stage. In order to show the effects of translation purposes, the image of the hand gesture is shifted to the origin (0.0) by mapping the scaled image coordinates to the normalized coordinates by using the relevant equations (see sub-section 5.3.2 (Coordinate Normalization)).

4.4.1.2 General Features

General features describe the hand image offsets, such as the height and width of the hand image, that serve as additional information. To encode the values of the height and width of the hand (the general features, see Figure 4.10), we use binary coding with a 6×6 matrix whose indices are (2) to the power of (0, 1, 2, 3, 4, 5), (or 1, 2, 4, 8, 16 and 32), as shown in Figure 4.11. Before converting this matrix into a vector that will be used as an input for the ANNs, we use the matrix as a means of storing the value of the general features. In this case, the value of the features can take the following values: 1, 2, 4, 8, 16, and 32, where 32 is the maximum value that a general feature value may take in the resized 32×32 image.

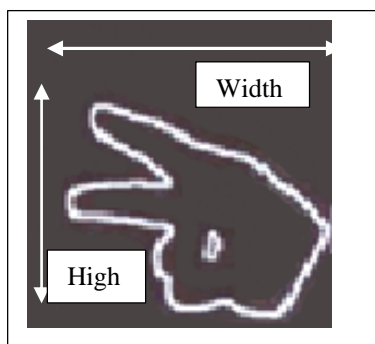


Figure 4.10 General features of the Cut gesture

This matrix consists of values of two 1s and the other remaining values of 0. The two 1s represent the height and width of the hand gesture. For example (as shown in the figure below), if the hand gesture has a width equal to 8 (width=8) or near this value (by using a threshold value) and a height equal to 32 (height=32) or near this value, then the position of (2^3) holds a value equal to 1. Similarly, the height position of (2^5) holds a value equal to 1. Another presentation can be suggested by using only one (1) rather than two (1s) to represent the height and width of the gesture. However, two positions (two 1s and remaining 0s) in a vector of 1024 (32×32) are more likely to be distinguishable by the ANNs than if using only one position. The position of 1s in the

matrix and in the converted vector (i.e., the matrix is converted into a vector) is used as a means to distinguish between the different values of the general features. This feature matrix is compounded with the contour feature vector as a composed feature to produce a new feature vector with (1060) elements and is then passed to the back-propagation neural network as its input vector, as shown in Figure 4.12.

	2^0	2^1	2^2	2^3	2^4	2^5
2^0	0	0	0	0	0	0
2^1	0	0	0	0	0	0
2^2	0	0	0	0	0	0
2^3	0	0	0	0	0	1
2^4	0	0	0	0	0	0
2^5	0	0	0	1	0	0

Figure 4.11 Example of binary encoding of general features (6×6) matrix

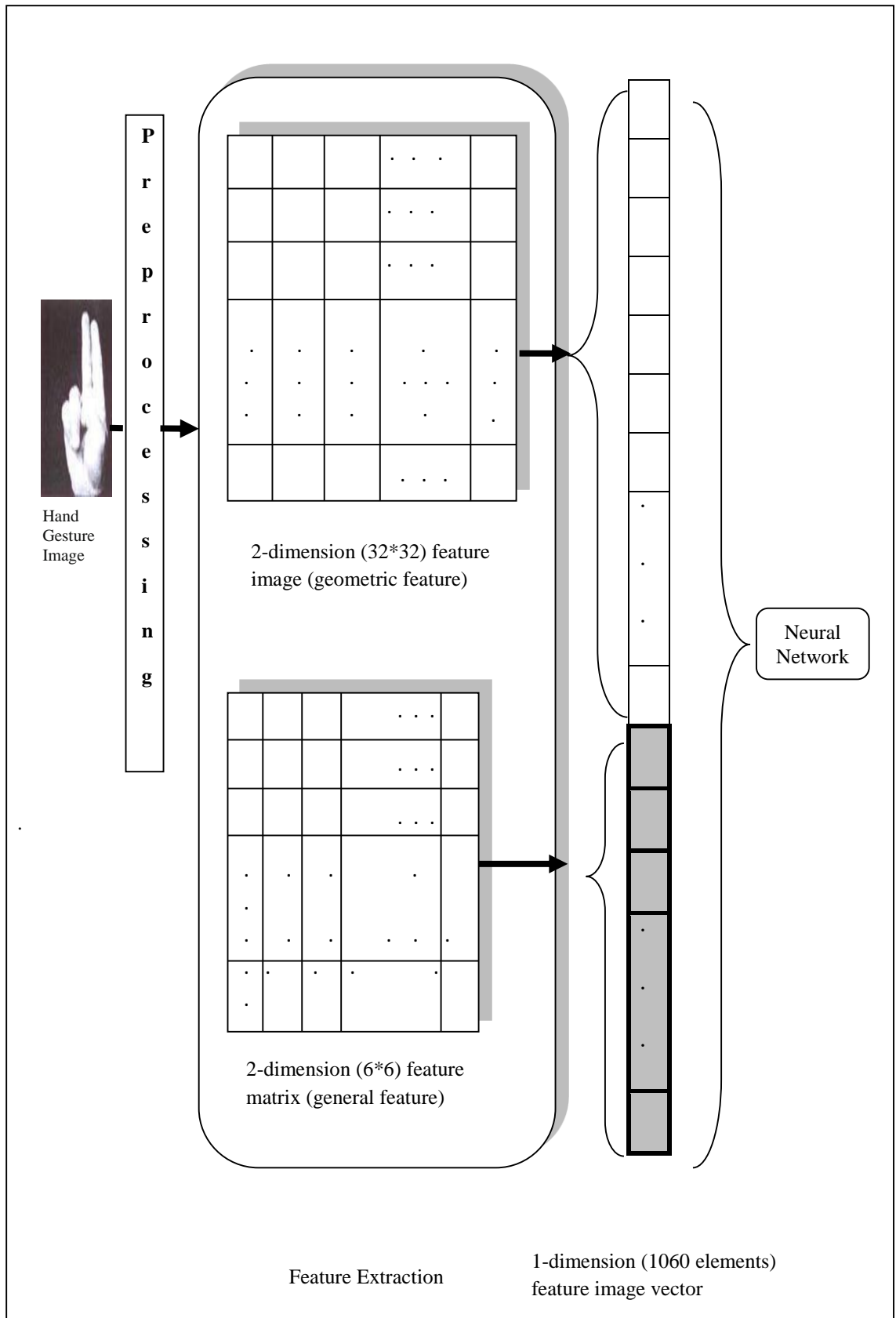


Figure 4.12 Geometric and General Features as Input Vector to the Multilayer Neural Network

4.4.2 Complex Moments

In the proposed method of gesture recognition, the complex moments introduced in sub-section 3.2.2.3 are used as a moment invariant, which is a set of moment values extracted from the image data such that their values are invariant to scaling, translation, and rotation of the image data (see sub-section 3.2.2.3). The CM feature extraction stage includes the calculation of the complex moments feature vector for each hand gesture in the database. This method of calculation is briefly explained in the next subsection.

4.4.2.1 Complex Moments Calculation

For each hand gesture image, the complex moments from (zero-order M0) to (ninth-order M9) are calculated by using the equations in Section 3.2.2.3, such that each feature vector has 10 values (see Figure 4.13).

M0	M1	M2	M3	M4	M5	M6	M7	M8	M9
----	----	----	----	----	----	----	----	----	----

Figure 4.13 Complex Moments Feature Vector

Normalization is performed on these values in the range of [0 1] prior to conducting the training phase with ANN.

4.5 Neural Network Based Classifier

The last stage of the proposed system is classification. The classification algorithm used in a specific gesture recognition system is highly dependent on the properties and the format of the features representing the gesture image. In this study, a standard back-propagation neural network is used to classify gestures. The network consists of three layers; the first layer consists of neurons responsible for inputting a hand gesture sample into the neural network. The second layer is a hidden layer that allows the neural network to perform the error reduction necessary to achieve the desired output.

The final layer is the output layer with one node per class. Typically, the number of neurons in this layer is determined by the size of the set of desired outputs, with each possible output represented by a separate neuron. The structure of this particular back-propagation neural network is illustrated in Figure 4.14.

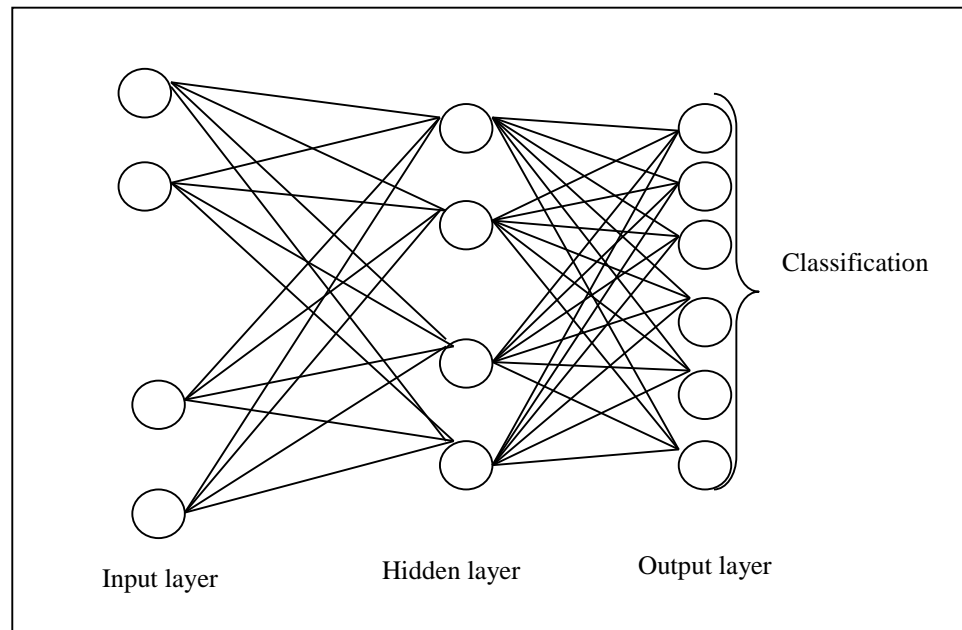


Figure 4.14 Gesture Recognition Network Architecture

Six outputs are present in the neural network. Each output represents the index for one of the six hand gesture images classes. The highest index value (in the testing phase) represents the recognized gesture image. Five neural networks with the same structure are used in the recognition process.

4.5.1 ANN with Hand contour

The parameters for the multilayer neural networks used with hand contour method are shown in Table 4.1

Table 4.1 Parameters for the Five Multilayer Neural Networks

Parameters	Values
Input Layer	1060 nodes
Hidden Layer	100 nodes
Output layer	6 nodes
Stop error	0.01
Learning rate	0.9

As discussed in Section 3.3.5, in the back-propagation network, a hand gesture sample is propagated through the multilayer neural network, producing an output. This output is compared with the desired output, giving an error rate for the output layer. Given that the error rate of a neuron is a function of the error rates of all the units that use its output, the error rates of the layer directly below the output layer can now be obtained. These error rate calculations continue to propagate through the network in a backward fashion until the error rates for all the neurons have been found. Each neuron then makes a slight weight adjustment to minimize error signal. This pattern is repeated until the error rate of the output layer reaches a minimum value. This process is then repeated for the next input value until all of the input values have been processed. Figure 4.15 shows the back-propagation flowchart.

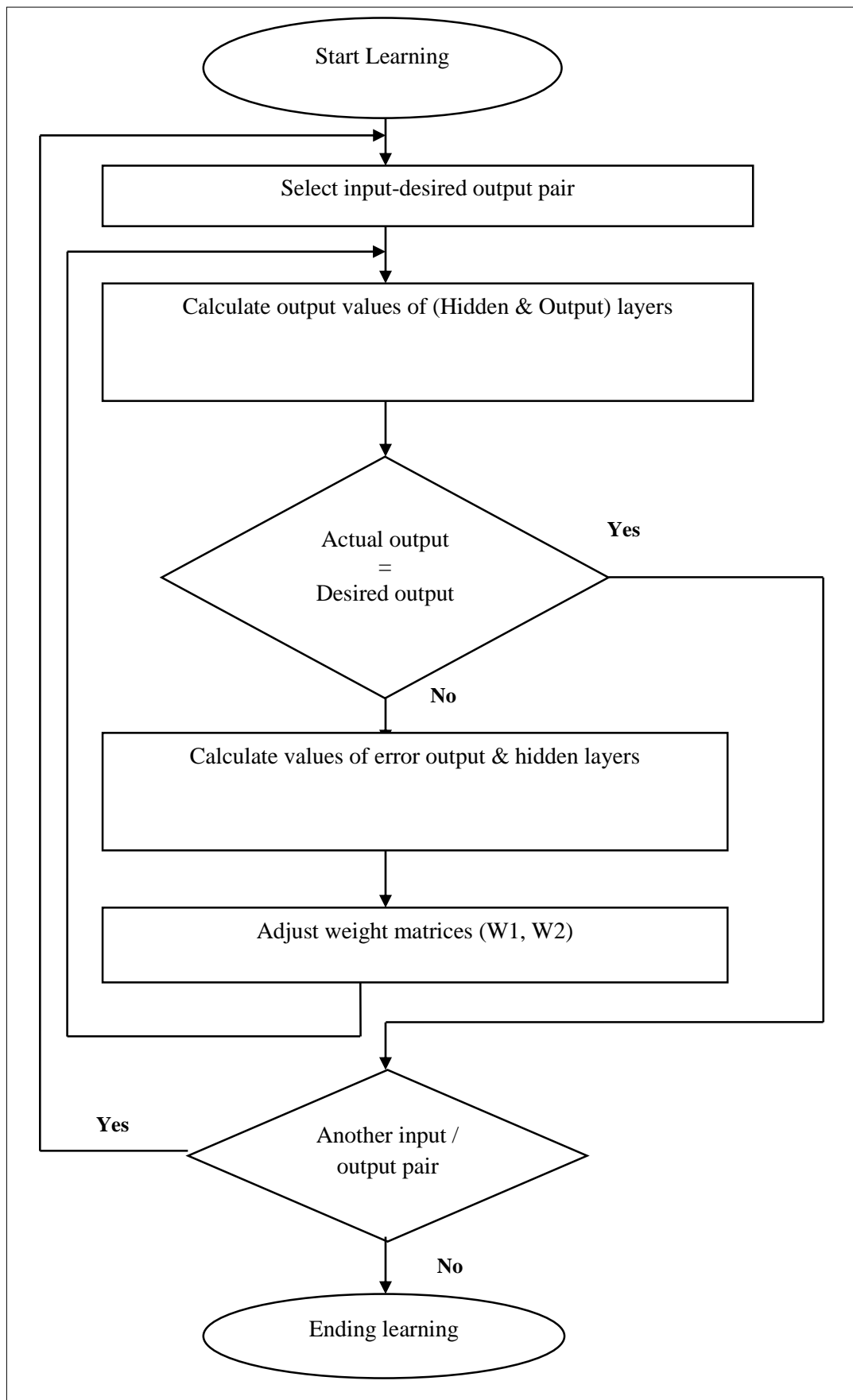


Figure 4.15 Flowchart for Back-Propagation Learning Algorithm

The detailed design of the classification stage is illustrated in Figure 4.16

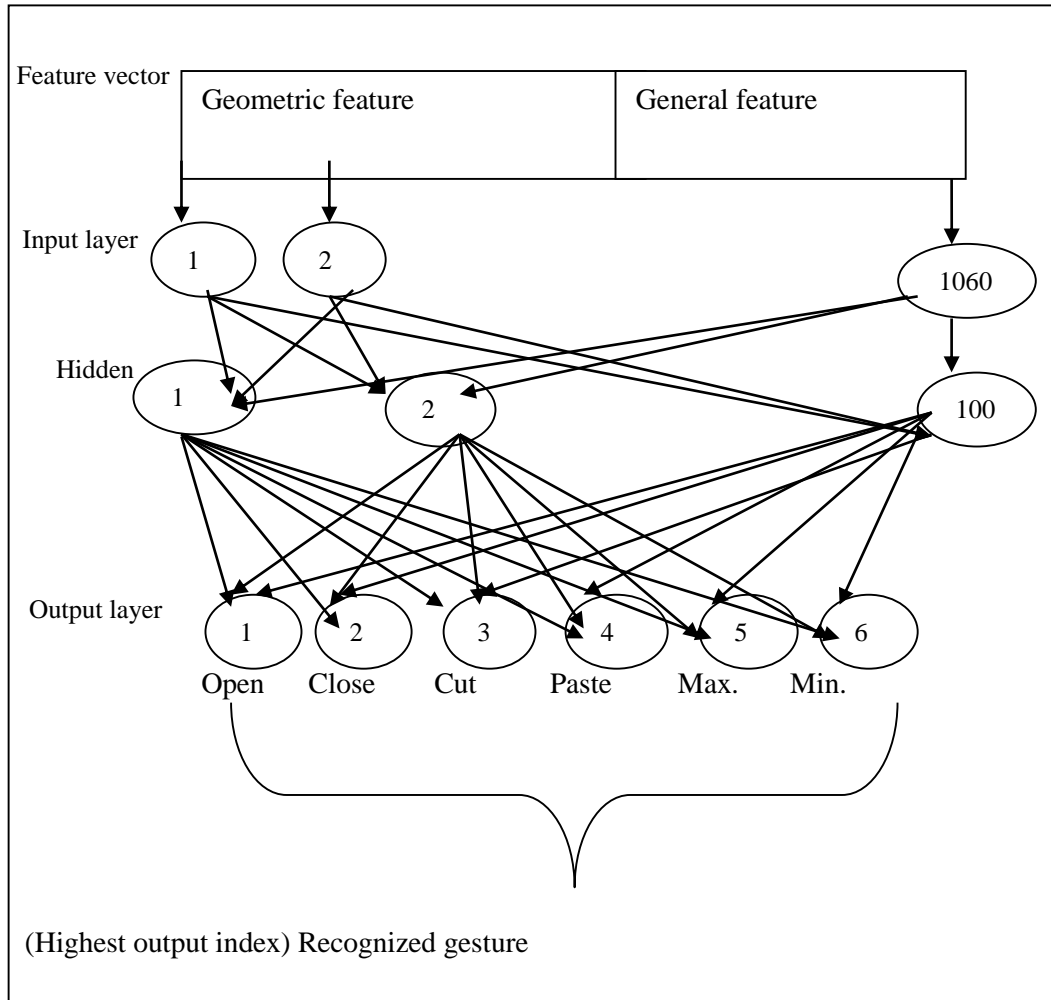


Figure 4.16 Detailed Design of a Neural Network

4.5.2 ANN with complex moments

As in the previous method, a multilayer neural network with the same structure is used for recognizing hand gestures. The input for this network is a feature vector with 10 values. Table 4.2 shows the parameters of this neural network. The remaining details on the recognition process are the same as that discussed in Section 4.5.1.

Table 4.2 Parameters for the Five Multi-layer Neural Networks

Parameters	Values
Input Layer	10 nodes
Hidden Layer	6 nodes
Output layer	6 nodes
Stop error	0.01
Learning rate	0.9

4.6 Summary

The proposed gesture recognition system consists of three basic stages: pre-processing, feature extraction, and classification. The main problems associated with 2D object recognition are scaling, translation of position, and rotation by angle from the principle axes (Torres-Mendez, Ruiz-Suarez, Sucar, & Gomez, 2000). The system was developed using algorithm encompasses two methods. In the first method, the hand contour was extracted as a geometric feature with the intention to address the problems of scaling and translation (in some cases). The second method intended to treat the problem associated with rotation in addition to the problems of scaling and translation by using the hand complex moments feature. In the pre-processing stage, a homogeneous background was used to facilitate hand segmentation. The extracted features were used in the last stage (by using neural networks and the supervised back-propagation learning algorithm), during which the neural network was responsible for recognizing and classifying the hand gesture.

In a classification system, choosing the features properly, as well as determining the correct means of presenting these features to the classifier, are necessary steps for building a successful pattern recognition system.

5.0 EXPERIMENTAL SETUPS

5.1 Introduction

In this chapter, we will briefly describe and illustrate the steps of the experimental work and then define the values of the parameters for both feature extraction methods, used together with the ANN implementation. In addition, a description of the database used for training and testing is presented.

5.2 Hand Contour with ANNs

In this stage, many processes were performed on the hand gesture image to prepare these images for the subsequent feature extraction stage. These processes were performed using some image processing operations as discussed in Chapter Four. The effect of these operations is explained below:

5.2.1 Hand Gesture Segmentation

All the techniques used in this research are based on hand shapes. The color images of hand gestures were segmented to isolate the foreground (hand) from the background.

5.2.2 Noise Reduction

The segmented images may contain some noises that will affect the results of the feature extraction stage. Thus, a median filter was used to reduce the noise as much as possible.

5.2.3 Edge Detection

The Edge detection process was performed by using a Sobel operator.

5.2.4 Feature Extraction

The result of the pre-processing phase was fed to the feature extraction stage to compute for the feature information of the gesture image, as shown in Figure 5.1.

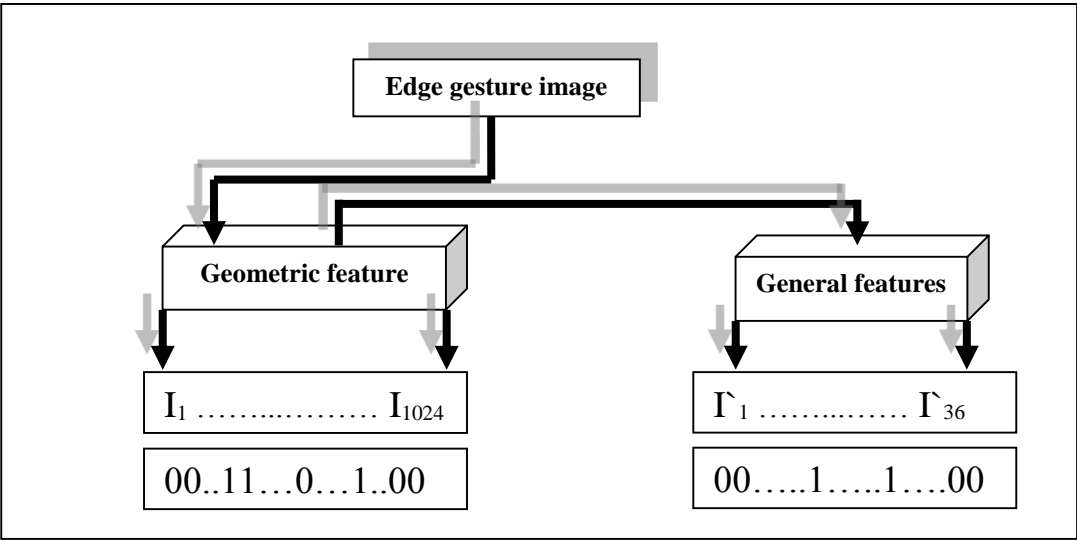


Figure 5.1 Feature Extraction Stage

As soon as the contour feature is computed, the image size is adjusted, such that each hand gesture image has the size of 32×32 . Image resizing accelerates the system and reduces the negative effects of size change by creating a standard size for all images. The general features (height offset and width offset) will be computed implicitly.

5.2.5 Training Phase

In this phase, the composite feature vectors computed earlier and stored in a feature image database are used as inputs to train the neural networks in the next stage, as shown in Figure 5.2.

The learning process for the five multilayer neural networks is accomplished by using the parameters shown in Table 5.1.

Table 5.1 Parameters for the Five Neural Networks

Parameters	Values
Input Layer	1060 nodes
Hidden Layer	100 nodes
Output layer	6
Stop error	0.01
Learning rate	0.9



Figure 5.2 Hand Gestures under different lightening conditions

5.2.6 Testing Phase

After training the five neural networks, the performance is evaluated by using a new set of inputs (test set) and then computing the classification error. The activation function used is the binary-sigmoid function, which always produces outputs between 0 and 1. In our case, the five neural networks are used in a sequential manner, i.e., the test gesture feature image will be entered to the first neural network; if the network successfully recognizes the gesture, the test operation stops. If this network does not recognize the gesture features, the second network will be activated, and so on. If all the five networks fail to identify the feature, a message “*gesture not recognized*” appears. Notably, the failure of the neural network to recognize the gesture rather than wrongly recognizing it is directly related to the output of the network, where the recognized image is the one that receives the highest value; in the case where two or more images receive the same highest output value, the network fails to recognize the gesture.

In the testing phase, 56 hand gesture images were used to test the system under different light conditions and with the effects of scaling and translation. The system is capable of recognizing and classifying any unknown gesture if such gesture is in the original database.

5.3 Complex Moment with ANNs

The processing stage in this method includes, in addition to segmentation and noise reduction processes as in the previous method, image trimming for eliminating the empty space and extracting only the region of interest, followed by the normalization process.

5.3.1 Image Trimming Effect

The filtered hand gesture image may contain unused space surrounding the hand gesture. Thus, image trimming process is used to extract the hand gesture from its background. The effect of this process is shown in Figure 5.3.













Filtered image	Trimmed image	Filtered image	Trimmed image
			
			
			

Figure 5.3 Image Trimming Effects

5.3.2 Coordinate Normalization

After scaling each image to a fixed size (250×250), the coordinates for the hand image are normalized between $[-1, +1]$. An example of this operation is shown in Figure 5.4. For the example below, the coordinate $(250, 0)$ becomes $(1,1)$ and the coordinate $(0, -250)$ becomes $(-1,-1)$.

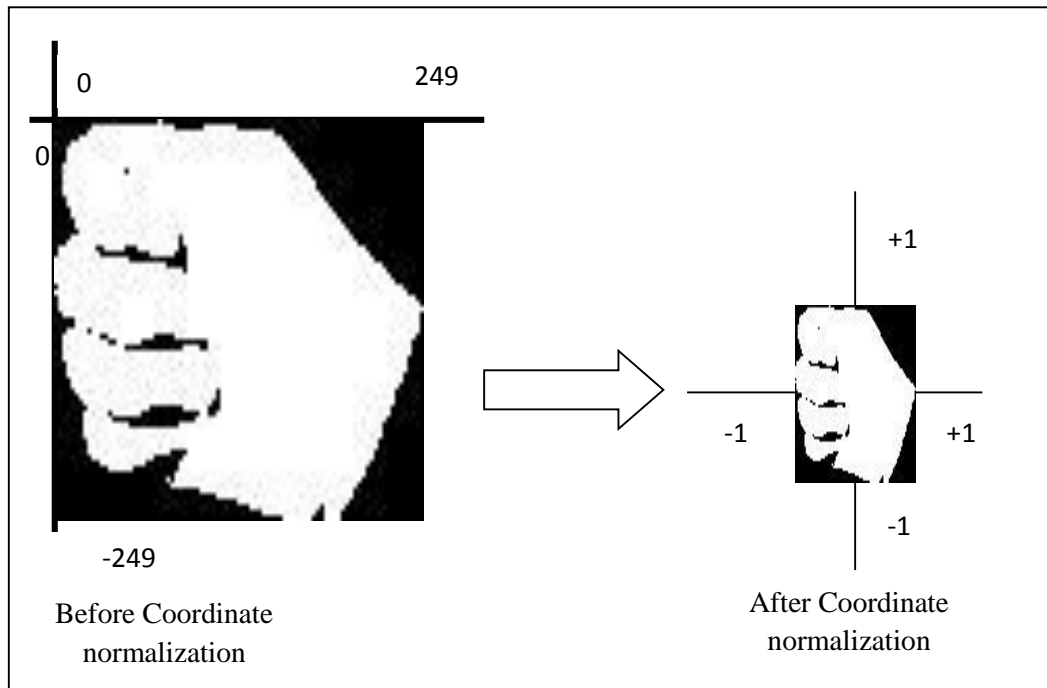


Figure 5.4 Coordinate Normalization

5.3.3 Complex Moments Calculation

Each hand gesture in the training set will have a feature vector with 10 values, which represent the complex moments starting with zero order up to nine order. Tables 5.2 and 5.3 show examples of the results of these computations before and after the normalization process. As can be seen in Table 5.3, all the feature values for complex moments are normalized between 0 and 1.

Table 5.2 Complex Moments Values before Normalization

Moment order ➔	M0	M1	M2	M3	M4	M5	M6	M7	M8	M9
Gesture name ⬇										
Open	45660	0.11310	2167.2	2330.2	538.9	1448.3	555.3	57.6	1691.2	8819
Close	46652	0.02355	953.7	1248.5	2342.8	1971.3	1261.6	731.1	1656.5	905.8
Cut	30916	0.16550	7553.2	1300.5	2793.3	1545.0	3230.4	1249.5	4125.3	1247.2
Paste	37186	0.14053	4680.8	1900.6	965.4	1161.9	1418.5	1190.7	408.5	616.5
Maximize	28965	0.17819	7142.2	3383.1	8998.0	752.4	7099.1	2109.6	10954.9	1894.8
Minimize	43866	0.08460	5710.4	1938.4	3020.8	2963.1	2904.2	2420.0	3076.2	3117.8

Table 5.3 Complex Moments Values after Normalization

Moment order ➔	M0	M1	M2	M3	M4	M5	M6	M7	M8	M9
Gesture name ⬇										
Open	1	0.63	0.28	0.68	0.05	0.48	0.07	0.02	0.15	0.28
Close	0.97	0.13	0.12	0.36	0.26	0.66	0.17	0.30	0.15	0.29
Cut	0.67	0.92	1	0.38	0.31	0.52	0.45	0.51	0.37	0.40
Paste	0.81	0.78	0.61	0.56	0.10	0.39	0.19	0.49	0.03	0.19
Maximize	0.63	1	0.94	1	1	0.25	1	0.87	1	0.60
Minimize	0.96	0.47	0.75	0.57	0.33	1	0.40	1	0.28	1

5.3.4 Training Phase

After the computation of feature vectors, each one (feature vector) contains 10 translation, scaling, and rotation-invariant elements characterizing the complex moments for the hand gestures. Five similar neural network classifiers are trained with a data set containing 30 feature vectors (training data set). These vectors were computed from the training set that includes five examples for each hand gesture performed by one subject. The training set of the gesture images are the same as that shown in Figure 5.8. The learning process for the back-propagation neural networks is accomplished by using the parameters shown in Table 5.4. The number of nodes in the input layer is equal to the length of the feature vector while the number of nodes in the output layer is equal to the number of hand gestures. In addition, the number of nodes in the hidden layer is selected based on a trial and error approach, that is, many trials are performed with different number of nodes and the number that gives the best result is selected.

Table 5.4 Parameters of Back-Propagation Neural Networks

Parameters	Values
Input Layer	10 nodes
Hidden Layer	6 nodes
Output layer	6
Stop error	0.01
Learning rate	0.9

5.3.5 Testing Phase

After training the five neural networks using the training data consisting of 30 images, the performance is evaluated by applying the testing set on the network inputs and then computing the classification error. The testing process is conducted in the same manner

as in the previous method, which is discussed in Section 5.3.4. In this phase, 84 hand gesture images are used to test the system. Each one of the six hand gestures has a number of samples under different light conditions and with effects of scaling, translation and rotation.

5.4 Preliminary results

This section provides the training results of Hand Contour with Neural Network and Complex Moments with Neural Networks as follows. We mention that all the algorithms used in this thesis are implemented and tested using Matlab R2013a.

5.4.1 Hand Contour with Neural Network

Five neural networks, initialized with different weight values, are trained through successive epochs (iterations) to create five NNs models with different parameters values. These ANNs models are used in the testing phase in a sequential way, as was described in section 5.3.4, to enhance the recognition ability of our system, whereby the second neural network will be invoked if the first ANN fails to uniquely identify the hand gesture. This process is repeated with the other ANNs until the gesture is recognized or the label “not recognized” is output if the last (fifth) ANN fails to recognize the gesture.

During the training phase and after each epoch, the square error over the validation set is computed. The training results are presented in Figures 5.5 to 5.10, which show the convergence of the learning algorithm for three back-propagation neural networks (as examples) and the learned samples with respect to the number of epochs for each network.

For example, in Figure 5.5, which represents the learning convergence of the first network, the recognition error for the first network was 0.5 for almost the first 100 epochs then it started converging, or the error start decreased, between 100 and 110

epochs until reaching the stop error (0.01). In the corresponding learning phase of the first network, in Figure 5.6, at the beginning the number of gestures that was successfully recognized by the network was 0 then, after several iterations, the network started learning how to recognize different gestures slowly until it successfully recognize all the gestures.

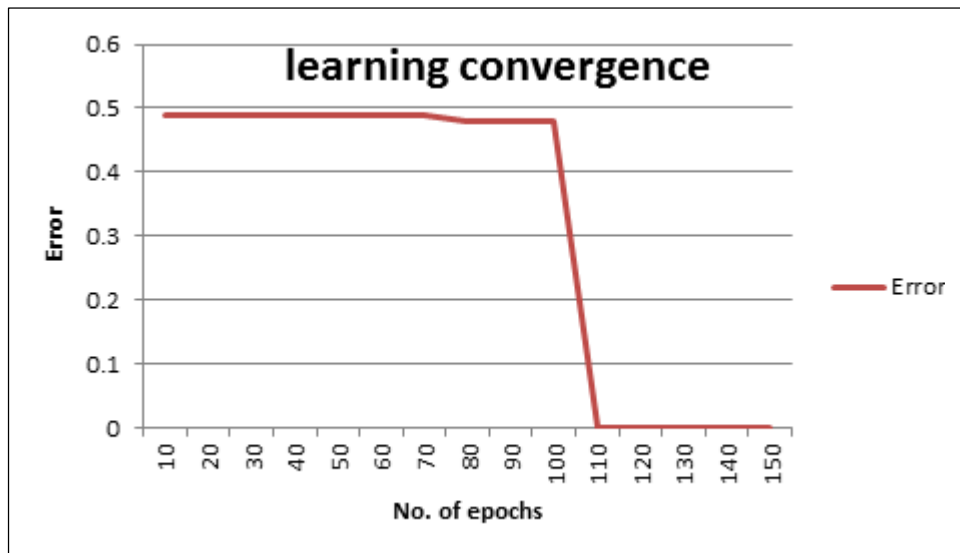


Figure 5.5 Learning convergence algorithm for the first neural network

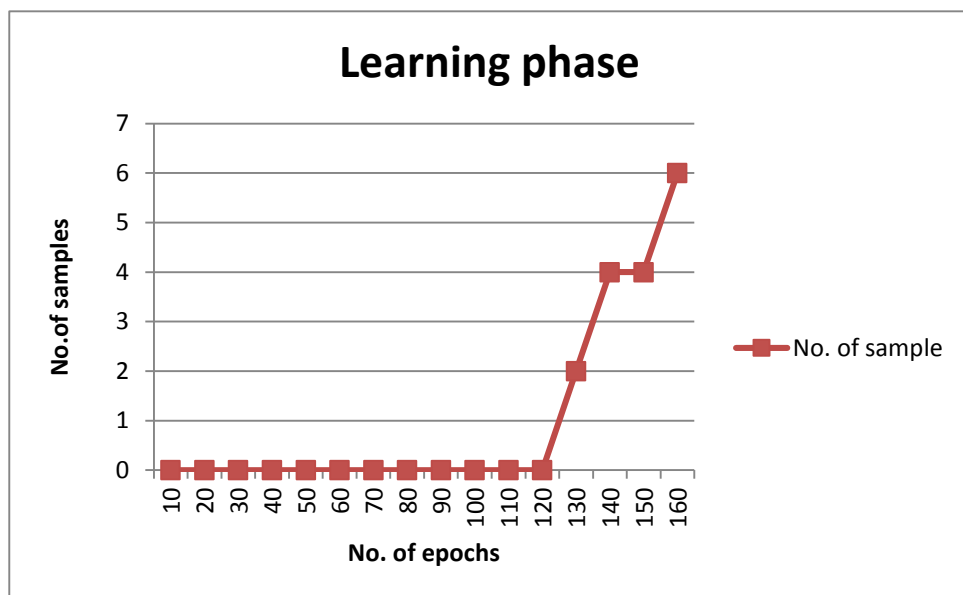


Figure 5.6 Learning phase with respect to number of epochs (first neural network)

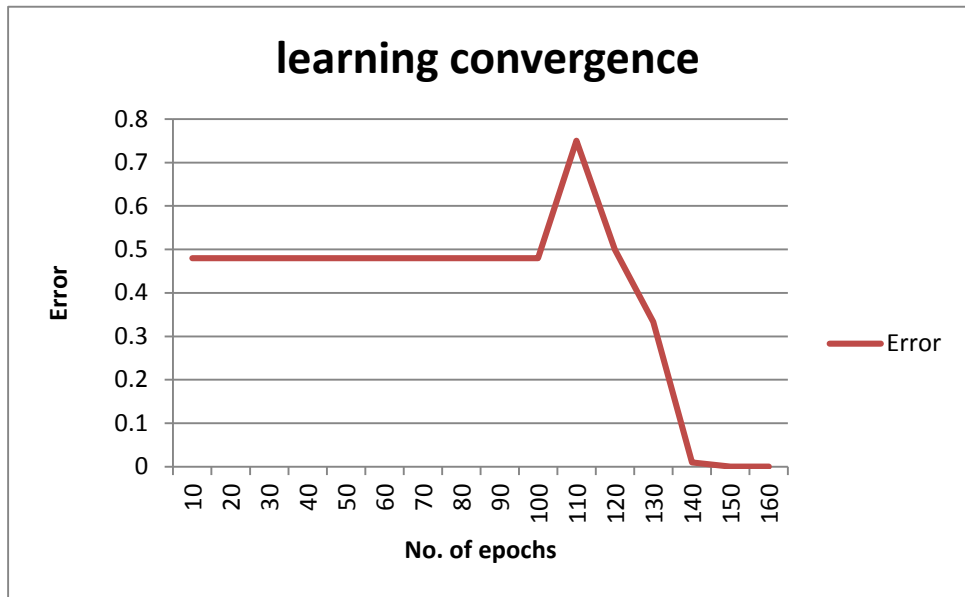


Figure 5.7 Learning convergence algorithm for the second neural network

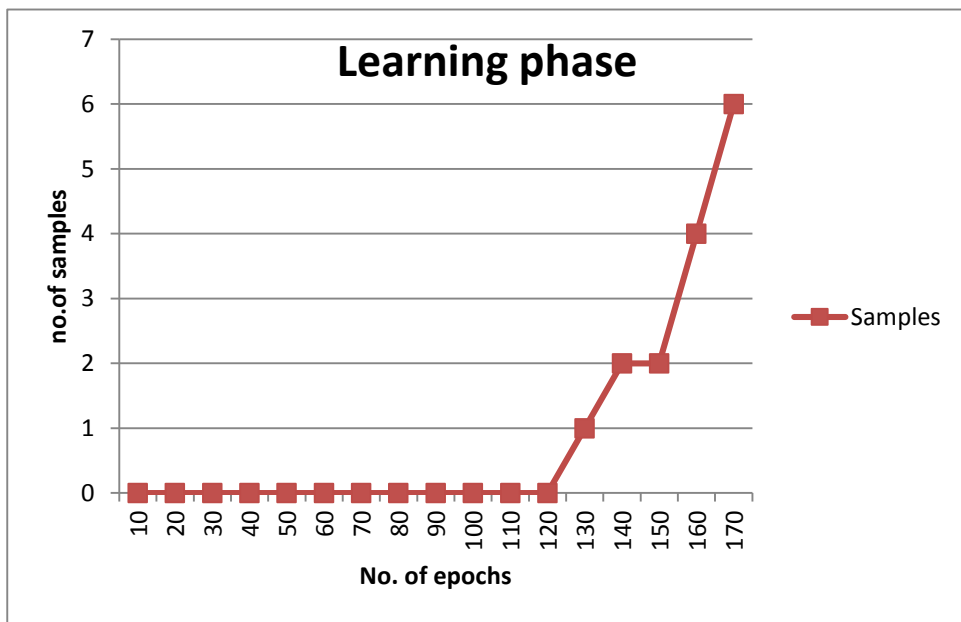


Figure 5.8 Learning phase with respect to number of epochs (second neural network)

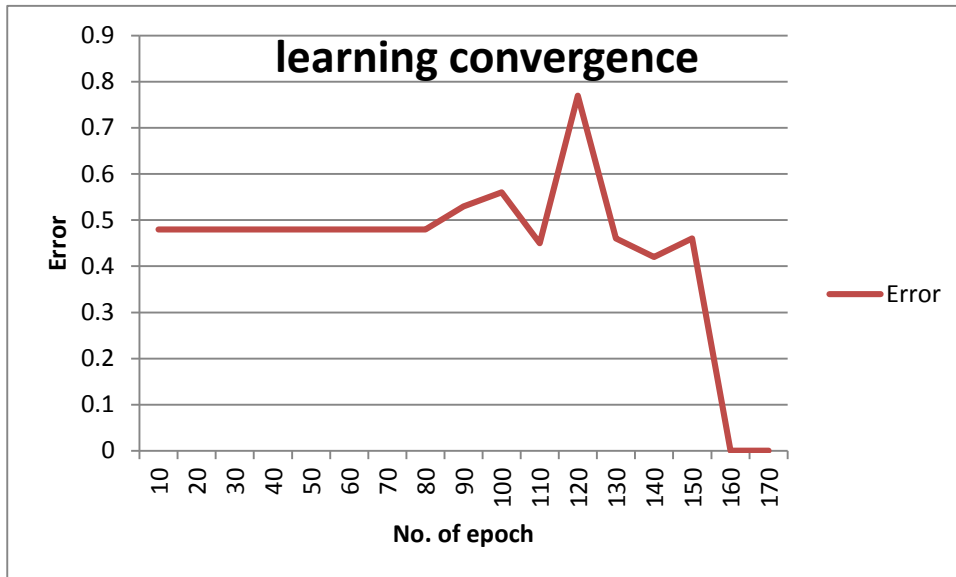


Figure 5.9 Learning convergence algorithm for the third neural network

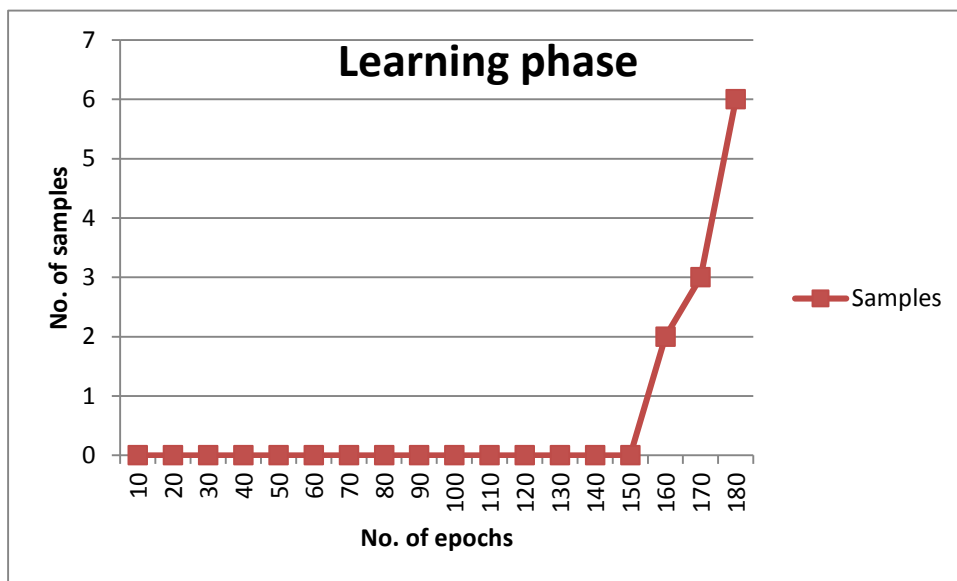


Figure 5.10 Learning phase with respect to number of epochs (third neural network)

5.4.2 Complex Moments with Neural Network

The training results in Figures 5.11 to 5.16 show the convergence of the learning algorithm for three back-propagation neural networks (as examples), as well as the learned samples with respect to the number of epochs for each network.

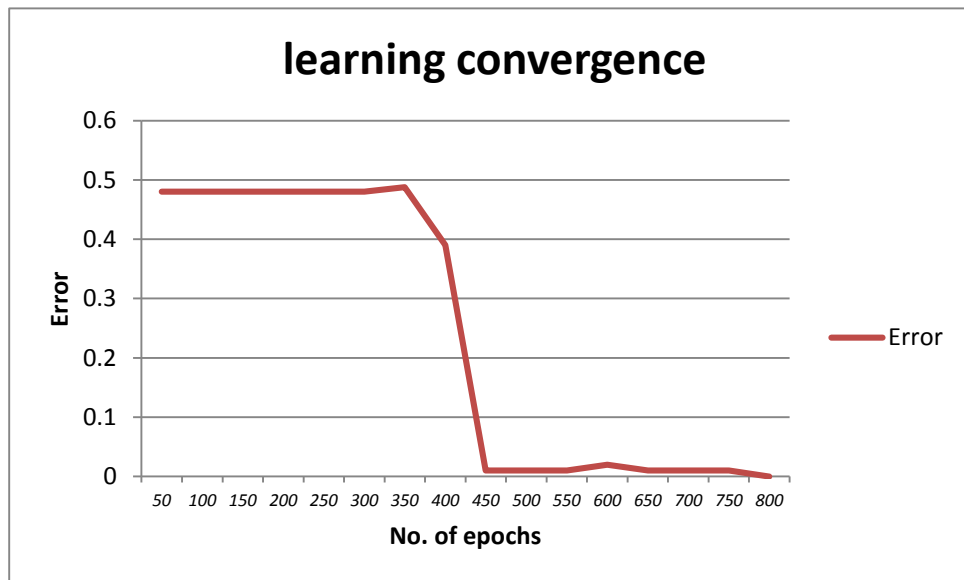


Figure 5.11 Learning convergence algorithm for the first neural network

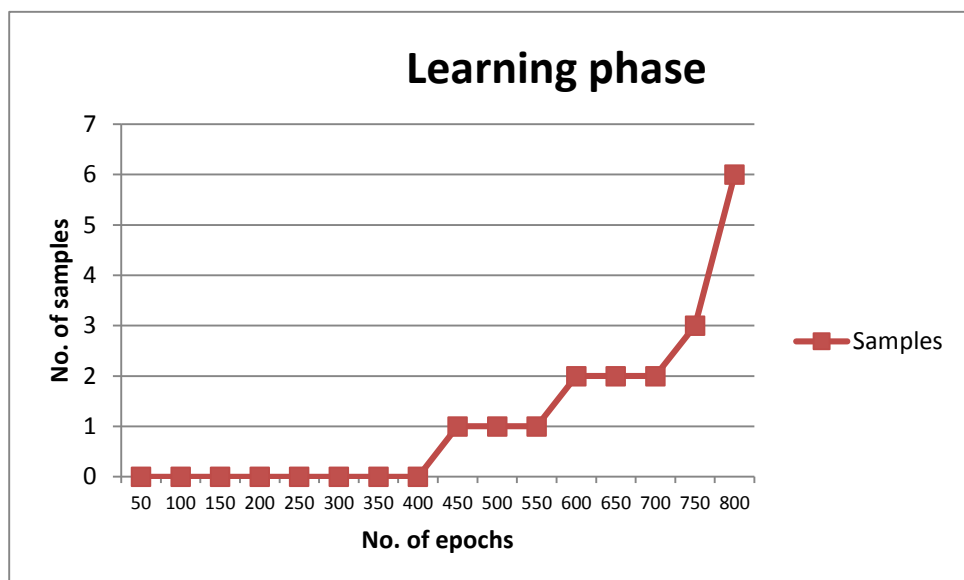


Figure 5.12 Learning phase with respect to number of epochs (first neural network)

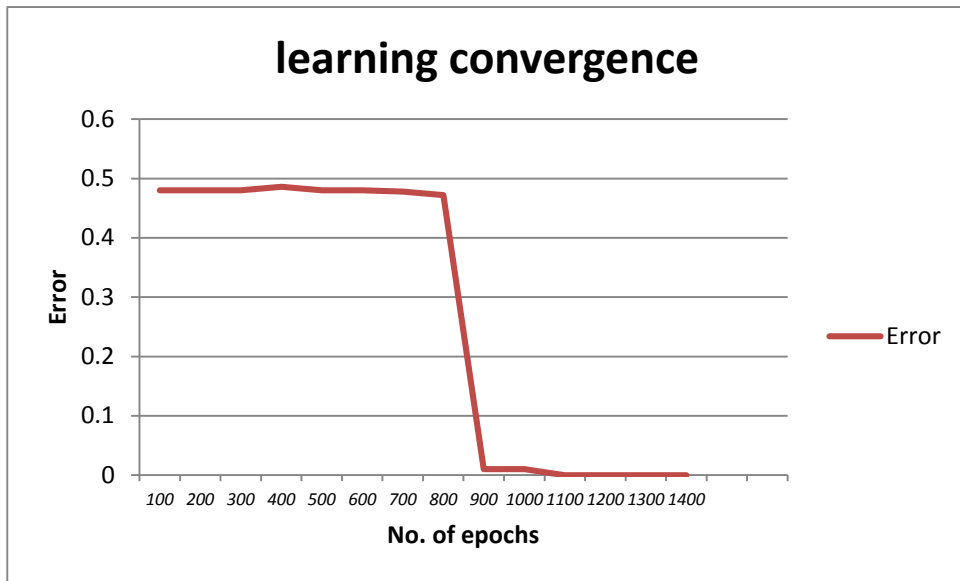


Figure 5.13 Learning convergence algorithm for the second neural

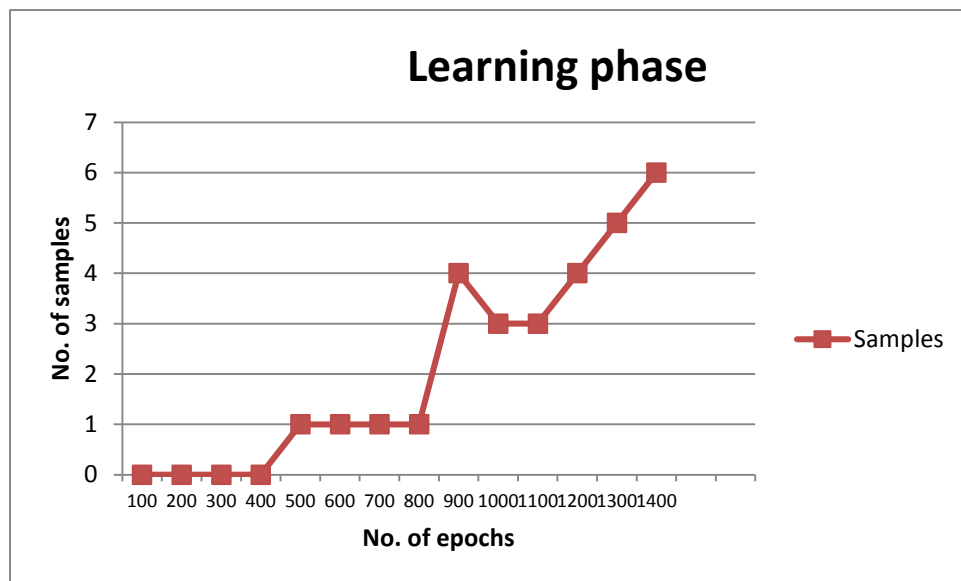


Figure 5.14 Learning phase with respect to number of epochs (second neural network)

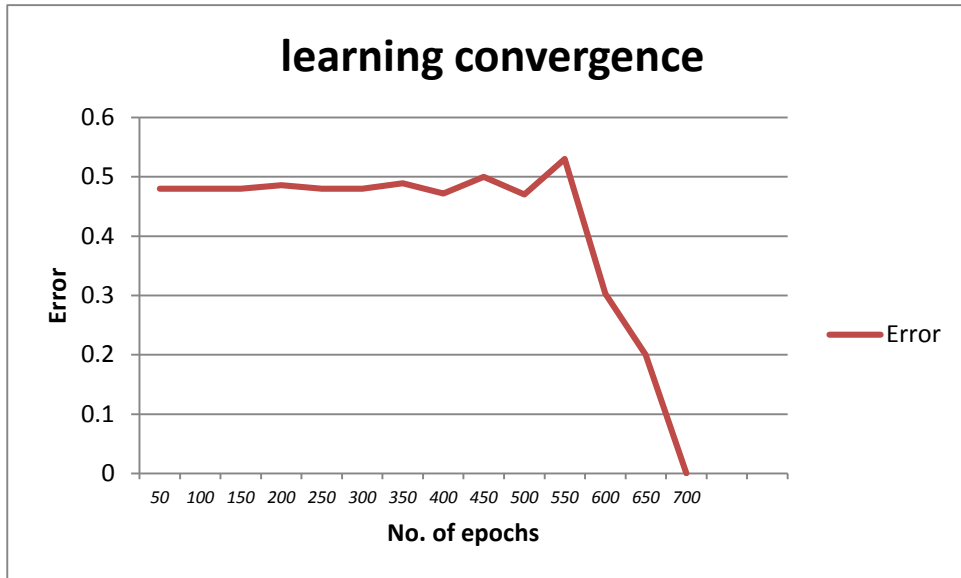


Figure 5.15 Learning convergence algorithm for the third neural network

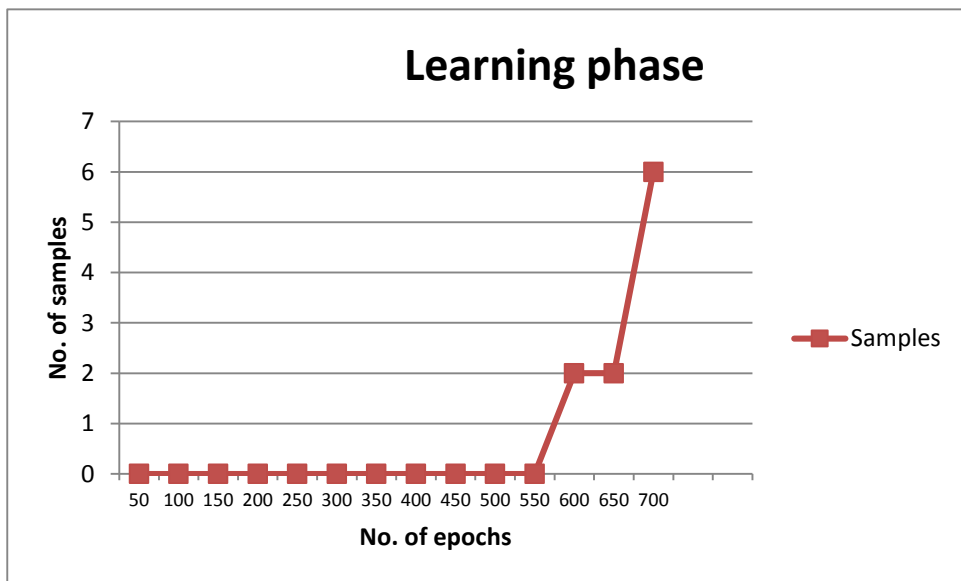


Figure 5.16 Learning phase with respect to number of epochs (third neural network)

5.5 Summary

The experimental work was carried out using the following setting regarding the data set and the parameters of neural network:

For hand contour method, the database consists of 30 images for the training set (five samples for each gesture) and 56 images for testing under different light conditions and with effects of scaling and translation. The parameters of the multi-layer perceptron neural networks for this method are the following: 1060 nodes for input layer, 100 nodes for hidden layer and 6 nodes for output layer.

For complex moments, the database consists of 30 images for the training set and 84 hand gesture images for testing under different light conditions and with effects of scaling translation and rotation. The parameters of the multi-layer perceptron neural networks for this method are the following: 10 nodes for input layer, 6 nodes for hidden layer and 6 for output layer.

6.0 RESULTS AND DISCUSSIONS

6.1 Introduction

In this chapter, the results obtained from the evaluation of the proposed hand gesture recognition system as mentioned in Chapter 5, are presented and discussed.

6.2 Criteria for evaluation

The performance of the proposed system is evaluated based on its capability to recognize gestures correctly based on the corresponding input gestures. The metric used to accomplish this task is called the recognition rate. The recognition rate is defined as the ratio of the number of correctly recognized gestures to the total number of input gesture samples, as shown in Equation (6.1).

$$\text{Recognition rate (\%)} = \frac{\text{No. of correctly recognized cases}}{\text{Total number of cases}} \times 100 \quad \dots \quad (6.1)$$

In addition to the recognition rate, *sensitivity* and *specificity* are also considered for each of the six gesture classes. Sensitivity measures the proportion of positive cases that are correctly identified while specificity calculates the proportion of negative cases that were correctly classified (Altman & Bland, 1994). These metrics, in the case of multi-class problem, are calculated for each class_{*i*} as follows:

$$\text{sensitivity}_{\text{of class}_i} = \frac{\text{No. of true positive cases of class}_i}{\text{No. of all positive cases of class}_i} \quad \dots \dots \dots \quad (6.2)$$

Where *true positive cases of class_{*i*}* are the cases of class_{*i*} which are correctly recognized, and all *positive cases of class_{*i*}* are all the cases that belong to the class_{*i*}.

$$\text{specificity}_{\text{of class}_i} = \frac{\text{No. of true negative cases of class}_i}{\text{No. of all negative cases of class}_i} \quad \dots \dots \dots \quad (6.3)$$

Where *true negative cases of class_{*i*}* are the cases that belong to other classes *class_{*j*}* (*j*=1,...,*c*) where *i*≠*j* and are correctly recognized, and *all negative cases of class_{*i*}* are the







all the cases that belong to other classes *class_j*; (j=1,...,c) where i≠j.

c is the number of classes (c=6 in our case).

6.3 Results of Testing Phase for Hand Contour with ANNs

A summary of all the recognition results and the recognition rates for each of the six static hand gestures is presented in Table 6.1, while the recognition rates for each class are shown in Figure 6.8.

Table 6.1 Summary of Recognition Results (Hand Contour)

Open	Close	Cut	Paste	Maximize	Minimize
					
Gesture Meaning	Number of test gesture	Successful recognition	Recognition rate (%)		
Open	8	4	50.00		
Close	9	7	77.77		
Cut	10	9	90.00		
Paste	10	6	60.00		
Maximize	10	8	80.00		
Minimize	10	7	70.00		
Average of recognition rate (%)				71.30	

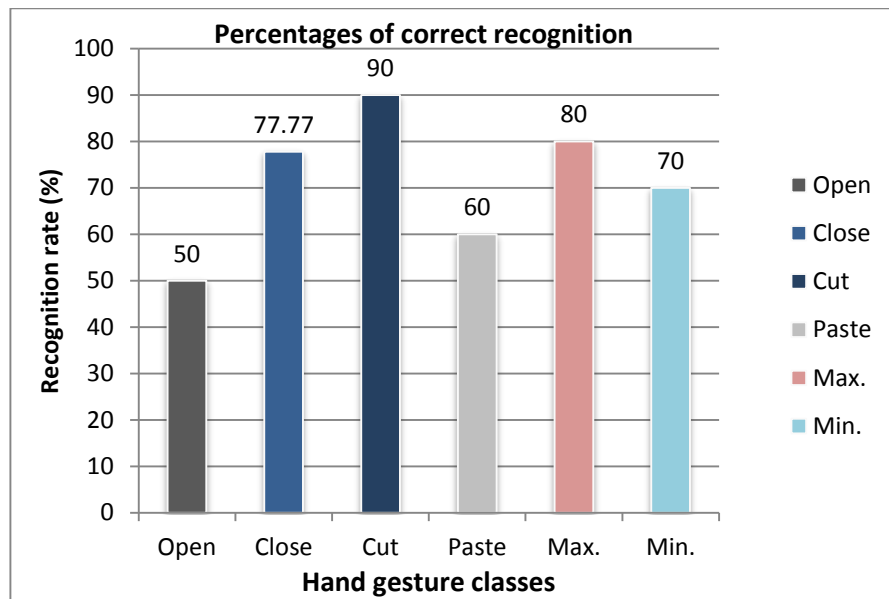


Figure 6.1 Percentages of Correct Recognition for each hand gesture class (Hand Contour)

The detailed recognition results for the hand contour method are presented in Tables 6.2 to 6.13. The results of hand gesture recognition with scaling and translation effects for open, close, cut, paste, maximize and minimize are shown in Tables 6.2, 6.4, 6.6, 6.8, 6.10 and 6.12, respectively. For illustration purposes, an example of the results of each of the six gestures using neural networks recognition are displayed in Tables 6.3, 6.5, 6.7, 6.9, 6.11 and 6.13. As can be seen, for example in Table 6.3 which displays the results of a testing gesture image (open), the neural network output layer which consists of six outputs (each represents one gesture) produces six values (where each value is between 0 and 1), one for each gesture. Each value can be seen as the probability or likelihood that the corresponding gesture is the recognized one. For example, in Table 6.3, the likelihood values that the recognized gesture is Open, Close, Cut, Paste, Max. and Min. are 0.505, 0.00185, 0.0349, 0.043, 0.00722 and 0.00392, respectively. Intuitively, the gesture with the highest likelihood value is the most likely to be the correct gesture, which is Open in this case (Table 6.3). Note that the results of Tables

6.3, 6.5, 6.7, 6.9, 6.11 and 6.13 are for illustration only and they display the result of one case for each gesture.

We can see from the comparison between the results of Open and Cut in Table 6.3 and 6.7, respectively that the output value of Open in Table 6.3 is 0.505 while the output value of Cut in Table 6.7 is 0.926. This means that it is easier for the NN to recognize Cut (i.e. more confident) than Open; these values are reflected in the recognition rate for each of the two gestures: 90% for Cut and 50% for Open (Table 6.1). Figure 6.7 shows examples of Close gestures with translation and scaling effects.

Table 6.2 Results for (Open) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Open	Original (Artificial)	True
2	Open	Scaling	True
3	Open	Scaling	True
4	Open	Scaling	True
5	Not recognized	Scaling	False
6	Not recognized	Translation	False
7	Not recognized	Translation	False
8	Not recognized	Translation	False

Table 6.3 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Open) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.505	0.00185	0.0349	0.043	0.00722	0.00392

Table 6.4 Results for (Close) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Close	Original (Natural)	True
2	Close	Scaling	True
3	Close	Scaling	True
4	Close	Scaling	True
5	Close	Scaling	True
6	Close	Translation	True
7	Close	Translation	True
8	Not recognized	Translation	False
9	Not recognized	Translation	False

Table 6.5 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Close) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.00129	0.90	0.0152	0.0245	0.0130	0.0865

Table 6.6 Results for (Cut) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Cut	Original (Natural)	True
2	Cut	Scaling	True
3	Cut	Scaling	True
4	Cut	Scaling	True
5	Cut	Scaling	True
6	Not Recognized	Scaling	False
7	Cut	Translation	True
8	Cut	Translation	True
9	Cut	Translation	True
10	Cut	Translation	True

Table 6.7 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Cut) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0211	0.00102	0.926	0.00329	0.0147	0.0368

Table 6.8 Results for (Paste) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Paste	Original (Natural)	True
2	Not recognized	Original (Artificial)	False
3	Paste	Scaling	True
4	Paste	Scaling	True
5	Paste	Scaling	True
6	Not Recognized	Scaling	False
7	Paste	Scaling	True
8	Not recognized	Translation	False
9	Not recognized	Translation	False
10	Paste	Translation	True

Table 6.9 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Paste) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0148	0.0209	0.00418	0.880	0.00151	0.0432

Table 6.10 Results for (Maximize) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Maximize	Original (Natural)	True
2	Maximize	Scaling	True
3	Maximize	Scaling	True
4	Maximize	Scaling	True
5	Maximize	Scaling	True
6	Maximize	Scaling	True
7	Not recognized	Translation	False
8	Maximize	Translation	True
9	Maximize	Translation	True
10	Not recognized	Translation	False

Table 6.11 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Maximize) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0204	0.0449	0.00223	0.00022	0.940	0.000557

Table 6.12 Results for (Minimize) Hand Gesture with Scaling and Translation effects (Hand Contour)

Image No.	Recognition Result	Type of effects	Match result
1	Minimize	Original (Natural)	True
2	Minimize	Scaling	True
3	Minimize	Scaling	True
4	Minimize	Scaling	True
5	Minimize	Scaling	True
6	Minimize	Scaling	True
7	Not recognized	Translation	False
8	Not recognized	Translation	False
9	Minimize	Translation	True
10	Not recognized	Translation	False

Table 6.13 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Minimize) (Hand Contour)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.000186	0.0443	0.0121	0.00899	0.00309	0.881

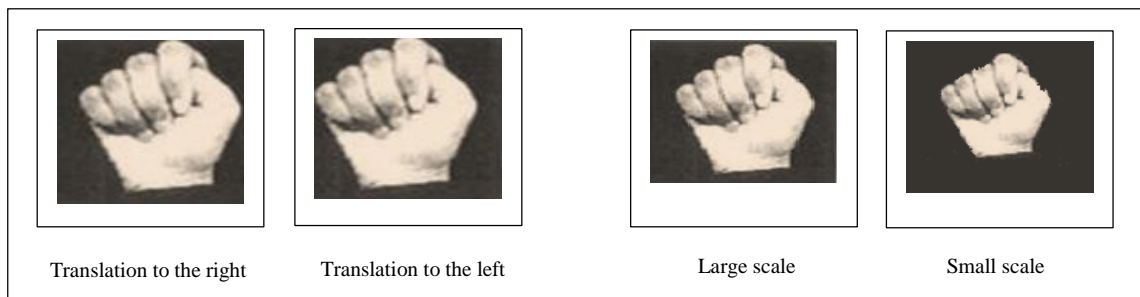


Figure 6.2 Close gesture with translation and scaling

6.3.1 Specificity and sensitivity for Hand Contour

As shown in Table 6.14 and Figure 6.8, the sensitivity and accuracy values for gesture classes are the same because they are calculated by the same method. For specificity values, we notice that Open and Cut achieved the highest and lowest value, respectively. For Open, the specificity value is as high as 0.7551, which means that the probability of any image taken from other classes (i.e. {Close, Cut, Paste, Max, Min}) to be correctly recognized is 0.7551, or simply the average recognition rate of the other classes is 75.51%, which means that Open has negatively contributed to the overall recognition rate.

For sensitivity, which reflects the recognition rate per class, we noticed that Cut/Open has the highest/lowest value 0.9/0.5 which means that Cut/Open has the best/worst recognition rate among the six gestures.

Table 6.14 specificity and sensitivity values for hand gestures (Hand Contour)

	specificity	Sensitivity
Open	0.7551	0.5
Close	0.7083	0.78
Cut	0.6808	0.90
Paste	0.7447	0.60
Max	0.7021	0.80
Min	0.7234	0.70

6.3.2 Scaling and translation in Hand Contour

From Table 6.15, we notice that most of the recognition errors are caused by images with Translation (75%) while a small portion of errors comes from images with Scaling (18.75%) and less than that from artificial illumination (6.25%). To evaluate the recognition rate of images with scaling and translation in hand contour², we can see from Tables 6.16 and 6.17 which show the results of recognition on images with scaling and translation effects, respectively, that hand contour was able to handle the cases of scaling relatively well (25 correct cases out of 28, or 89.29%) but the problem was related to the translation cases especially for some gestures such as Open, Paste and Min at 0%, 33.33% and 25% correctly recognized gestures, hence significantly decreasing the translation recognition rate to 45.45%. In Table 6.18 which represents the confusion matrix of the gesture recognition, we notice that all the errors are due to the “not recognized” cases which means that the classifier could not uniquely identify the gesture because the highest likelihood value is shared by two or more gestures.

Table 6.15 Recognition Errors of Hand Gesture with Scaling, Translation and Artificial Illumination Effects.

	Scaling	Translation	Artificial	Total
Open	1 (25%)	3 (75%)	0 (0%)	4
Close	0 (0%)	2 (100%)	0 (0%)	2
Cut	1 (100%)	0 (0%)	0 (0%)	1
Paste	1 (25%)	2 (50%)	1 (25%)	4
Max	0 (0%)	2 (100%)	0 (0%)	2
Min	0 (0%)	3 (100%)	0 (0%)	3
total	3	12	1	16
Recognition rate (%)	18.75%	75%	6.25%	100

² Unlike Complex Moments, Hand Contour method does not handle rotation cases, so we did not use them with this method.

Table 6.16 Recognition Rate of Various Hand Gestures with Scaling Effects for Hand Contour

	Correct recognition	Wrong recognition	Not recognized	Total cases of each gesture	Percentage of correct cases for each gesture (%)
Open	3	0	1	4	75
Close	4	0	0	4	100
Cut	4	0	1	5	80
Paste	4	0	1	5	80
Max	5	0	0	5	100
Min	5	0	0	5	100
total	25	0	3	28	
Recognition rate (%)	89.29	0	10.71	100	

Table 6.17 Recognition Rate of Various Hand Gesture with Translation for Hand Contour

	Correct recognition	Wrong recognition	Not recognized	Total cases of each gesture	Percentage of correct cases for each gesture (%)
Open	0	0	3	3	0
Close	2	0	2	2	100
Cut	4	0	0	4	100
Paste	1	0	2	3	33.33
Max	2	0	2	4	50
Min	1	0	3	4	25
total	10	0	12	22	
Recognition rate (%)	45.45	0	54.55	100	







Table 6.18 Confusion matrix for Hand Contour

	Open	Close	Cut	Paste	Max	Min	Not recognized	TOTAL
Open	4	0	0	0	0	0	4	8
Close	0	7	0	0	0	0	2	9
Cut	0	0	9	0	0	0	1	10
Paste	0	0	0	6	0	0	4	10
Max	0	0	0	0	8	0	2	10
Min	0	0	0	0	0	7	3	10

6.4 Results of Testing Phase Complex Moments with ANNs

A summary of all the recognition results and recognition rates for each of the six static hand gestures is presented in Table 6.19. These results are obtained by using Equation (6.1), and the recognition rates for each class are shown in Figure 6.15.

Table 6.19 Summary of the Recognition Results and the Recognition Rates (Complex Moments)

Open	Close	Cut	Paste	Maximize	Minimize
					
Gesture Meaning	Number of test gesture	Successful recognition	Recognition rate (%)		
Open	15	15	100.00		
Close	15	12	80.00		
Cut	12	10	83.33		
Paste	12	9	75.00		
Maximize	15	14	93.33		
Minimize	15	13	86.66		
Average of recognition rate (%)			86.37		

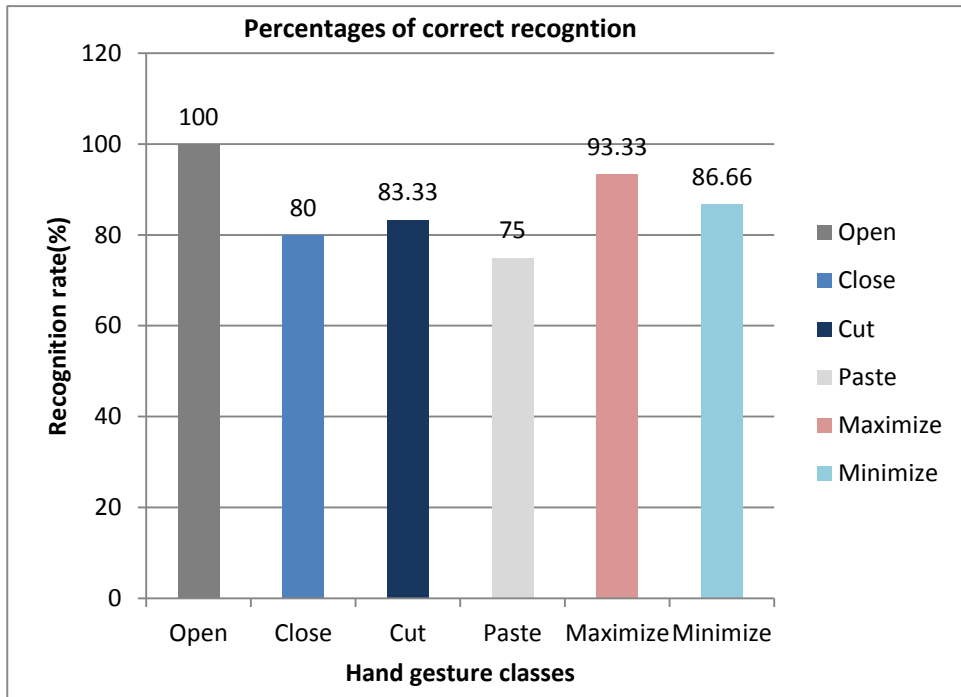


Figure 6.3 Percentages of Correct Recognition for each hand gesture class (complex moments)

The detailed recognition results are presented in Tables 6.20 to 6.31. Each gesture has a table of recognition results with the neural network outputs for one gesture image as an example. In addition, the testing images have some effects such rotation, scaling and translation.

Table 6.20 Results for (Open) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image No.	Recognition Result	Type of effects	Match result
1	Open	Original (Natural)	True
2	Open	Original (Artificial)	True
3	Open	Rotation	True
4	Open	Rotation	True
5	Open	Rotation	True
6	Open	Rotation	True
7	Open	Scaling	True
8	Open	Scaling	True
9	Open	Scaling	True
10	Open	Scaling	True
11	Open	Scaling	True
12	Open	Translation	True
13	Open	Translation	True
14	Open	Translation	True
15	Open	Translation	True

Table 6.21 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Open) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.99	0.00208	0.00387	0.249	0.0542	0.0000000707

Table 6.22 Results for (Close) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image No.	Recognition Result	Type of effects	Match result
1	Close	Original (Natural)	True
2	Close	Original (Artificial)	True
3	Close	Rotation	True
4	Close	Rotation	True
5	Close	Rotation	True
6	Open	Rotation	False
7	Paste	Rotation	False
8	Close	Scaling	True
9	Close	Scaling	True
10	Close	Scaling	True
11	Close	Scaling	True
12	Not Recognized	Scaling	False
13	Close	Translation	True
14	Close	Translation	True
15	Close	Translation	True

Table 6.23 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Close) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0663	0.505	0.00398	0.00032	0.00121	0.0232

Table 6.24 Results for (Cut) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image No.	Recognition Result	Type of effects	Match result
1	Cut	Original (Natural)	True
2	Cut	Rotation	True
3	Close	Rotation	False
4	Cut	Rotation	True
5	Close	Rotation	False
6	Cut	Scaling	True
7	Cut	Scaling	True
8	Cut	Scaling	True
9	Cut	Translation	True
10	Cut	Translation	True
11	Cut	Translation	True
12	Cut	Translation	True

Table 6.25 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Cut) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.00000179	0.00545	0.708	0.0136	0.00188	0.320

Table 6.26 Results for (Paste) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image No.	Recognition Result	Type of effects	Match result
1	Paste	Original (Natural)	True
2	Open	Rotation	False
3	Paste	Rotation	True
4	Open	Rotation	False
5	Open	Rotation	False
6	Paste	Scaling	True
7	Paste	Scaling	True
8	Paste	Scaling	True
9	Paste	Translation	True
10	Paste	Translation	True
11	Paste	Translation	True
12	Paste	Translation	True

Table 6.27 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Paste) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.001	0.032	0.145	0.589	0.000354	0.00135

Table 6.28 Results for (Maximize) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image no.	Recognition result	Type of effects	Match result
1	Maximize	Original (Natural)	True
2	Maximize	Original (Artificial)	True
3	Maximize	Rotation	True
4	Maximize	Rotation	True
5	Maximize	Rotation	True
6	Minimize	Rotation	False
7	Maximize	Rotation	True
8	Maximize	Scaling	True
9	Maximize	Scaling	True
10	Maximize	Scaling	True
11	Maximize	Scaling	True
12	Maximize	Scaling	True
13	Maximize	Translation	True
14	Maximize	Translation	True
15	Maximize	Translation	True

Table 6.29 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Maximize) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0000748	0.000352	0.304	0.00153	0.829	0.000356

Table 6.30 Result for (Minimize) Hand Gesture with Rotation, Scaling and Translation effects (Complex Moments)

Image No.	Recognition result	Type of effects	Match result
1	Minimize	Original (Natural)	True
2	Minimize	Original (Artificial)	True
3	Minimize	Rotation	True
4	Minimize	Rotation	True
5	Minimize	Rotation	True
6	Paste	Rotation	False
7	Paste	Scaling	False
8	Minimize	Scaling	True
9	Minimize	Scaling	True
10	Minimize	Scaling	True
11	Minimize	Scaling	True
12	Minimize	Translation	True
13	Minimize	Translation	True
14	Minimize	Translation	True
15	Minimize	Translation	True

Table 6.31 The Likelihood Value from the NN for a specific case of the Testing Hand Gesture Image (Minimize) (Complex Moments)

Class No.	1	2	3	4	5	6
Class Name	Open	Close	Cut	Paste	Max.	Min.
Neural network output	0.0374	0.0255	0.0000478	0.841	0.0000515	0.937

6.4.1 Specificity and sensitivity for Complex Moments

As shown in Table 6.32, the sensitivity and accuracy values for gesture classes are the same because they are calculated by the same method.

Table 6.32 shows that the highest and lowest for specificity is achieved by Paste (0.8889) and Open, respectively. For Open, the specificity value is 0.8405, the lowest value, while sensitivity is 1.00, the highest value, means that the probability of an image taken from Open gesture class to be correctly recognized is higher (probability is 1) than the average probability of other classes (probability is 0.8405). It also means that Open class positively contributes to the overall recognition rate.

Table 6.32 Specificity and sensitivity values for Complex Moments

	specificity	Sensitivity
Open	0.8405	1.00
Close	0.884	0.8
Cut	0.875	0.833
Paste	0.8889	0.75
Max	0.855	0.933
Min	0.8696	0.8667

6.4.2 Rotation, Scaling and Translation for Complex Moments

Table 6.33 shows clearly that most of the recognition errors for Complex Moments are attributed to the cases with rotation (81.82%) while small portion of errors (18.18%) is caused by cases with scaling effect. Translation was perfectly recognized for all the gestures (error = 0%).

Tables 6.34, 6.35, and 6.36 display the recognition rate of hand gestures with rotation, scaling, and translation, respectively. The complex moment method is efficient in

handling scaling (96.15%) and translation (100%) but is less robust in rotation (65.38%). If we consider the recognition errors in the rotation cases, some of these errors can be attributed to the similarity of hand gestures. For example, as the confusion matrix in Table 6.37 displays, three out of 12 hand gesture images of *paste*, or 25%, are wrongly recognized as *open* because of their similarities, especially in the case of rotation with specific angles. The same reason can be given for cases of hand gesture *close*, where two out of 15 images, or 13.33%, are wrongly recognized as *open* and *paste*. Thus, the complex moments method can be considered invariant to rotation for some hand gestures (for example, *open* and *max* with 100% and 80% correct recognition in rotation cases, respectively (see Table 6.34)), but not invariant to others with specific angles especially for *paste* where it successfully recognized only 1 case out of 4, or 25% of the rotation cases.

Table 6.33 Recognition Error of Hand Gesture with Translation for Complex Moments

	Scaling	Translation	Rotation	Total
Open	0 (0%)	0 (0%)	0 (0%)	0
Close	1 (33.33%)	0 (0%)	2 (66.67%)	3
Cut	0 (0%)	0 (0%)	2 (100%)	2
Paste	0 (0%)	0 (0%)	3 (100%)	3
Max	0 (0%)	0 (0%)	1 (100%)	1
Min	1 (50%)	0 (0%)	1 (50%)	2
Total	2	0	9	11
Recognition rate (%)	18.18%	0%	81.82%	100

Table 6.34 Recognition of Hand Gesture with Rotation for Complex Moments

	Correct recognition	Wrong recognition	Not recognized	Total cases of each gesture	Percentage of correct cases for each gesture (%)
Open	4	0	0	4	100
Close	3	2	0	5	60
Cut	2	2	0	4	50
Paste	1	3	0	4	25
Max	4	1	0	5	80
Min	3	1	0	4	75
total	17	9	0	26	
Recognition rate (%)	65.38%	34.62%	0%		

Table 6.35 Recognition of Hand Gesture with Scaling For Complex Moments

	Correct recognition	Wrong recognition	Not recognized	Total cases of each gesture	Percentage of correct cases for each gesture (%)
Open	5	0	0	5	100
Close	4	0	1	5	80
Cut	3	0	0	3	100
Paste	3	0	0	3	100
Max	5	0	0	5	100
Min	5	0	0	5	100
total	25	0	1	26	
Recognition rate (%)	96.15%	0%	3.85%	100	

Table 6.36 Recognition of Hand Gesture with Translation for Complex Moments

	Correct recognition	Wrong recognition	Not recognized	Total cases of each gesture	Percentage of cases for each gesture (%)
Open	4	0	0	4	100
Close	3	0	0	3	100
Cut	4	0	0	4	100
Paste	4	0	0	4	100
Max	3	0	0	3	100
Min	4	0	0	4	100
total	22	0	0	22	
Recognition rate (%)	100	0	0		

Table 6.37 Confusion Matrix of the Results Achieved Using Complex Moments

	open	close	cut	paste	max	Min	Not recognized
Open	13	0	0	0	0	0	0
Close	1	12	0	1 (Rotation)	0	0	1
Cut	0	2 (Rotation)	10	0	0	0	0
Paste	3 (Rotation)	0	0	9	0	0	0
Max	0	0	0	0	14	1 (Rotation)	0
Min	0	0	0	2 (1Rotation +1scaling)	0	13	0

6.5 Comparison between the results of Hand Contour and Complex Moments

6.5.1 The Learning Speed

Figures 6.1 to 6.6 and Figures 6.8 to 6.13, respectively, present the convergence of the learning and learned samples with respect to the number of epochs for three neural networks trained by features extracted using hand contour and complex moments. As clearly shown by these figures, the three neural networks trained using hand contour features take roughly between 110 and 160 epochs to converge, whereas the three neural networks trained using complex moment features require at least between 450 and 900 epochs to convergence. The same observation can be made for the learned samples with respect to number of epochs for the two methods. The conclusion that can be drawn from this comparison is that the hand contour method is significantly faster (at least 4 times faster in the examples shown) than complex moments, which suggests its advantage in real-world applications where the speed of learning is a prime concern.

6.5.2 Recognition Accuracy

Tables 6.1 to 6.13 and 6.19 to 6.31 respectively show the recognition accuracy achieved for hand contour and complex moments. The results show that hand contour achieved recognition accuracy as high as 71.30%, whereas complex moments achieved 86.90% as shown in Tables 6.1 and 6.19 respectively. These results indicate that complex moments is more accurate than hand contours.

In addition, Table 6.38 shows that the number of “not recognized” cases for hand contour is significantly higher than complex moments (16 cases for hand contour and 1 case for complex moments), which suggests that hand contour faced some difficulties in uniquely distinguishing the hand gestures as it gives equal certainty (probability) values to more than one hand gesture, thus preventing the ANN from making the decision.

By contrast, complex moment features are more “decisive” in recognizing the hand gesture, having only one “*not recognized*” case out of 84 cases compared to 16 cases out

of 56 for hand contour method. Table 6.39 illustrates this behavior. While all the error cases of complex moments except one are attributed to wrong predictions, all the errors for hand contour are attributed to the incapability of the network to make a decision (see Table 6.38). This behavior is also known in fuzzy logic in a case where the predicted class cannot be uniquely defined because two or more classes have the same highest certainty (or probability) value (Ishibuchi, Nakashima, & Murata, 1999).

Table 6.38 The Number of “Not Recognized” Cases for Hand Contour and Complex Moments

	Hand contour	Complex moments
Open	4	0
Close	2	1
Cut	1	0
Paste	4	0
Maximize	2	0
Minimize	3	0
Total	16	1
Percentage of “ Not Recognized ” cases in the testing data (%)	28.57	1.20

Table 6.39 The Number of “False” Cases for Hand Contour and Complex Moments

	Hand contour	Complex moments
Open	0	0
Close	0	2
Cut	0	2
Paste	0	3
Maximize	0	1
Minimize	0	2
Total	0	10
Percentage of “ False ” cases in the testing data (%)	0	11.90

6.5.3 Overfitting of Neural networks

As can be seen from the figures that represent the learning convergence, the neural networks learning process has converged and reached the lower target error at 0.01 for both methods in the training phase. In the testing phase, however, hand contour for example achieved only 71.30% as testing accuracy. This reflects the overfitting behaviour of neural networks which occurs when the testing accuracy is considerably lower as compared with the training accuracy. One conventional solution to reduce the overfitting problem in ANN is the use of an extra data as a validation set and check after every epoch whether the accuracy on the validation data is increasing or not. The training is stopped when the accuracy is increasing on the training set while it is decreasing in the validation set (Heskes, 1997). The drawback of this method is the need of a lot of data set and it is computationally expensive especially for real-time applications.

So, we can see that the overfitting of neural networks is more visible in the hand contour method which is considered as one of the disadvantage of hand contour method. In addition, neural networks seem more effective when the dimension of the feature vector, which is equal to the number of nodes in the input layer, is moderate (10 nodes for complex moments and 1060 nodes for hand contour). This may be due to the difficulty that neural networks usually face when approximating complicated problems that involve high dimensionality as in the case of the hand contour method which combines both geometric (vector with 1024 elements) and general (vector with 36 elements) features using binary coding (J. E. Meng, Shiqian, Lu, & Hock, 2002).

6.5.4 Comparison with previous works

In this subsection, our work is compared with three previous related works, namely, (Symeonidis, 1996), (Just, 2006) and (Parvini & Shahabi, 2007). These works applied

different feature extraction methods but similar classification method. The recognition accuracy of the methods proposed in these works along with our work are shown in the Table 6.40.

Table 6.40 Recognition rates of related hand gesture recognition methods

	Feature extraction method	Data set	Classification method	Recognition accuracy
(Symeonidis, 1996)	Orientation histogram	American Sign Language (ASL)	Single Layer Perceptron (ANN)	54.76%
(Just, 2006)	Modified Census Transform (MCT)	Data set with 10 hand gestures	Neural Networks	Uniform background: 92.79% Complex background: 81.25%
(Parvini & Shahabi, 2007)	Range of motion	American Sign Language (ASL)	ANNs	75%
Hand contour (our method)	Hand contour	Our data set	ANNs	71.30%
Complex moments (our method)	Complex moments	Our data set	ANNs	86.37%

Since the methods listed in Table 6.40 are not applied on the same data set, the comparison is just to get a general idea about the performance of other similar works for benchmarking purposes. Data set used by (Parvini & Shahabi, 2007) and (Symeonidis, 1996) is called American Sign Language (ASL) which has 26 gestures that represent the alphabets but without effects. (Just, 2006) used a data set that has 10 hand gestures images; representing 10 selected alphabets. These images were tested under two

different conditions: the first one tests images with uniform background while the second is with complex background.

As can be seen from Table 6.40, our methods and specifically Complex Moments compare favourably with other feature extraction methods and clearly better than some methods such as Orientation Histogram proposed in (Symeonidis, 1996). The results as obtained by (Parvini & Shahabi, 2007) shows that more challenging images with complex background achieved a lower recognition rate of (81.25%) as compared to images with a uniform background (92.79%). This finding supports the results of with our own research where images with more challenging effects such as rotation achieved as low as 65.38% recognition rate. One limitation of the work proposed in (Parvini & Shahabi, 2007) is that the user has to wear gloves in order to get the features of the hand gesture. In addition, it seems that the complexity of the data set or the number of gestures in data set makes the recognition task more challenging and this can be seen in the results achieved by the studies that used ASL ((Symeonidis, 1996) and (Parvini & Shahabi, 2007)) compared with our work and (Just, 2006).

6.6 Summary

We can summarize the main results of this chapter as follows:

- 1- The overall recognition accuracy of ANN with complex moments is 86.37% while it is only 71.30% for ANN with hand contour which clearly indicates that complex moments is better than hand contour in term of accuracy.
- 2- Regarding the convergence speed, hand contour is at least four times faster than complex moments (hand contour-based neural network took roughly between 110 and 160 epochs to converge, whereas complex moment-based neural network required at least between 450 and 900 epochs to convergence). This suggests that the hand contour method is more suitable than the complex moments method for real-world applications

that need faster training, such as online training systems.

3- The complex moment method is efficient in handling scaling (96.15%) and translation (100%) but is less robust in rotation (65.38%).

4- Hand contour is able to handle the cases of scaling relatively well (89.29%) but the problem was related to the translation cases especially for some gestures such as Open, Paste and Min which significantly decrease the translation recognition rate to 45.45%.

In addition, it cannot handle rotation cases.

7.0 CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

This research addresses the problem of static hand gesture recognition, and specifically, as one of the objectives of this study stated, develops a neural-based recognition system to recognize six selected static hand gestures (Open, Close, Cut, Paste, Maximize, and Minimize). The primary idea is to employ two feature extraction methods, namely, the hand contour method and the complex moments method, in the extraction of the features that characterize these hand gestures. These features are then used in training the neural networks to classify each gesture image into its respective class. In this chapter, we summarize the primary conclusions drawn from this research in the next section, while the main contributions of this study and some suggestions for future works are provided in section 7.2 and 7.3, respectively.

7.1 Conclusions

The primary conclusions can be summarized in the following points:

1. Hand contour-based neural networks training is evidently faster than complex moments-based neural networks training (at least 4 times faster as shown in Chapter 6 where hand contour-based neural network took roughly between 110 and 160 epochs to converge, whereas complex moment-based neural network required at least between 450 and 900 epochs to convergence). This suggests that the hand contour method is more suitable than the complex moments method in real-world applications that need faster training, such as online training systems.
2. On the other hand, complex moments-based neural networks (86.37%) proved to be more accurate than the hand contour-based neural networks (71.30%). In addition, the complex moments-based neural networks are shown to be resistant to scaling (96.15%) and translation (100%), and to some extent to rotation (65.38%) in some gestures (for example: open (100%), Maximum(80%)). The results indicate that the

complex moments method is preferred to the hand contour method because of its superiority in terms of accuracy especially for applications where training speed is not very crucial, such as off-line training applications and desktop applications.

3. Hand contour features are less “distinguishable” compared to complex moments features. The high number of “not recognized” cases predicted via the hand contour method makes this evident (11.90% of the testing cases for hand contour against 1.20% for complex moments). The recognized class cannot be uniquely defined because there are two or more classes (gestures) that have the same high certainty (or probability) value.
4. Neural networks are powerful classifier systems, but they suffer from the problem of overfitting (as discussed in 6.5.3), a problem which was more visible with hand contour method. Less overfitting was observed with the complex methods method, which is considered as an advantage for this method as the learning techniques which avoid the overfitting problem can provide a more realistic evaluation about their future performance based on the training results. In addition, neural networks appear to be more efficient when the number of features or the dimension of the feature vector, which is equal to the number of nodes in the input layer, is moderate (e.g., the complex moments method with 10 nodes is more accurate than the hand contour method with 1060 nodes).
5. The current research aims to provide a generic system that can be customized according to the needs of the user by using each of the six gestures as a specific command. For example, a direct application of the current system is to use it as virtual mouse that has six basic functions, namely, Open, Close, Cut, Paste, Maximize, and Minimize.

6. In addition, the proposed system is flexible; it can be expanded by adding new gestures or reduced by deleting some gestures. For example, you can use four gestures for TV control application, with each gesture being translated into one TV command: “Open”: to turn on the TV; Close: to turn off the TV; Min: to reduce the sound volume; and Max: to increase the sound volume, and so on.

7.2 Contribution of this study

Current researches in the field are limited to the use of glove-based, non-skin color background and orientation histogram with their associated problems.

In this research we have shown that the use of feature extraction method together with a neural network classifier is able to recognise a limited number of gestures accurately. We have also shown that the recognition system developed is non-costly with respect to time as compared to systems using the gabor filter (Gupta et al., 2012).

Furthermore, hand recognition system based on the feature extraction method works well under different lighting conditions while most feature extraction methods employed by other researchers (Freeman & Roth, 1995) failed.

We have also shown that the method we employed are able to overcome the limitations of scaling, translation and rotation associated with most feature extraction methods (Abu-Mostafa & Psaltis, 1984).

This research contributes, in general, to the use of a “natural” mean, namely, hand gesture that humans employ to communicate with each other into Human Computer Interaction (HCI) technology which has become an increasingly important part of our daily lives. Some of the contributions of this research can be summarized as follows:

(1) This study develops a static hand gesture recognition system that can be used for different applications that involve a limited number of hand gestures such as virtual mouse.

(2)- This study investigates the suitability of two different feature extraction approaches to solve the hand gesture recognition problem by identifying the primary advantages and disadvantages of each method. This comparison may shed some lights on the challenges and opportunities that have to be considered if anyone would like to further improve the proposed methods or chooses to employ them in any real-life application. Complex moments method, due to its higher accuracy, is preferred for hand gesture-based desktop applications where the time cost is not of a prime concern while hand contour is better used in hand gesture-based online applications as it is faster in training compared to complex moments.

(3)- Complex moments method apparently overcomes the challenges other previous methods could not handle, namely, working under different conditions such as scaling, translation and rotation. Hand contour method, however, does not handle rotation cases which limits its use in hand gesture-based applications.

(4)- Neural network classifier, which is used as a recognizer for hand gesture images based on the features extracted by the two methods, is also evaluated in terms of accuracy, convergence speed and overfitting.

(5)- The two proposed feature extraction methods can be used by other researchers to develop hand gesture recognition systems or enhance the recognition rate by using new classification methods.

7.3 Limitations and suggestions for future works

The current study used data set which has six different hand gestures. Although the methods used are also applicable for more than six gestures but the results obtained in this study cannot be generalized for any other number of gestures.

Possible ways to extend and improve this work are suggested below:

- 1- Although the neural networks methods have been widely recognized as powerful classifier methods, other classifiers, such as the Hidden Markov Model (HMM) or the Support Vector Machine (SVM), may also be used for this problem along with the two feature extraction methods. In other words, researchers can benefit from our study by using our feature extraction methods along with new or existing classification methods.
- 2- One possible way to reduce the “not recognized” cases in the gesture recognition process is to employ ensemble classifiers, where the members of the ensemble are various types of classifiers such as decision trees, fuzzy systems, SVMs, etc. The recognized gesture, in this case, is the one which receives the highest number of votes from the ensemble members.
- 3- The two feature extraction methods employed in this study can be applied by other researchers or developers for other hand gesture-based applications. In this case, the current system should be adjusted to fit the new application by, for example, changing the number of neurons in the output layer of ANNs to correspond to the number of gestures to that the system needs to recognize.

REFERENCES

- Abd Alrazak, R. (2004). *Signature Recognition Using Wavelet Transform*. (PhD Thesis), University of Technology/ Department of Computer Science.
- Abu-Mostafa, & Psaltis. (1984). Recognitive Aspects of Moment Invariants. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-6(6)*, 698-706. doi: 10.1109/TPAMI.1984.4767594
- Abu-Mostafa, & Psaltis. (1985). Image normalization by complex moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on(1)*, 46-55.
- Altman, D. G., & Bland, J. M. (1994). Statistics Notes: Diagnostic tests 1: sensitivity and specificity. *BMJ, 308(6943)*, 1552. doi: 10.1136/bmj.308.6943.1552
- Amin, M. A., & Hong, Y. (2007, 19-22 Aug. 2007). *Sign Language Finger Alphabet Recognition from Gabor-PCA Representation of Hand Gestures*. Paper presented at the Machine Learning and Cybernetics, 2007 International Conference on.
- Athitsos, V., & Sclaroff, S. (2005). *Boosting nearest neighbor classifiers for multiclass recognition*. Paper presented at the Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on.
- Awcock, G. J., & Awcock, T. (1995). *Applied Image Processing*: McGraw-Hill Companies.
- Barczak, A., & Dadgostar, F. (2005). Real-time hand tracking using a set of co-operative classifiers based on Haar-like features. *Res. Lett. Inf. Math. Sci., 7*, 29 -42.
- Basavaprasad, B., & Ravi, M. (2014). A comparative study on classification of image segmentation methods with a focus on graph based techniques. *International Journal of Research in Engineering and Technology, 03(03)*, 310-314.
- Beghdadi, K. B., & France, M. A. (1999, 19-21 May, 1999). *A contour detection method based on some knowledge of the visual system mechanisms*. Paper presented at the Vision Interface, Trois-Rivières, Canada.
- Bekir, C. (2012). *HAND GESTURE RECOGNITION*. (Master thesis), Dokuz Eylül University
- Binh, N. D., Enokida, S., & Ejima, T. (2006). *A New Approach Dedicated To Real-Time Hand Gesture Recognition*. Paper presented at the IPCV.
- Blunsom, P. (2004). Hidden markov models. [<http://www.cs.mu.oz.au/460/2004/materials/hmm-tutorial.pdf>].
- Bretzner, L., Laptev, I., & Lindeberg, T. (2002, 21-21 May 2002). *Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering*. Paper presented at the Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on.

- Cadoz, C., & Wanderley, M. M. (2000). Gesture-music. *Trends in gestural control of music*, 12, 101.
- Canny, J. (1986). A Computational Approach to Edge Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-8*(6), 679-698. doi: 10.1109/TPAMI.1986.4767851
- Chang, C. C., Chen, J. J., Tai, W. K., & Han, C. C. (2006). New approach for static gesture recognition. *Journal of Information Science and Engineering*, 22(5), 1047-1057.
- Chao, H., Meng, M. Q., Liu, P. X., & Xiang, W. (2003, 27-31 Oct. 2003). *Visual gesture recognition for human-machine interface of robot teleoperation*. Paper presented at the Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on.
- Chen, F.-S., Fu, C.-M., & Huang, C.-L. (2003). Hand gesture recognition using a real-time tracking method and hidden Markov models. *Image and Vision Computing*, 21(8), 745-758. doi: [http://dx.doi.org/10.1016/S0262-8856\(03\)00070-2](http://dx.doi.org/10.1016/S0262-8856(03)00070-2)
- Dadgostar, F., Barczak, A. L., & Sarrafzadeh, A. (2005). A color hand gesture database for evaluating and improving algorithms on hand gesture and posture recognition. *Research letters in the information and mathematical sciences*, 7, 127-134.
- Deng-Yuan, H., Wu-Chih, H., & Sung-Hsiang, C. (2009, 12-14 Sept. 2009). *Vision-Based Hand Gesture Recognition Using PCA+Gabor Filters and SVM*. Paper presented at the Intelligent Information Hiding and Multimedia Signal Processing, 2009. IHH-MSP '09. Fifth International Conference on.
- Din, A. (2002). *Optimization and forecasting with financial time series*. Paper presented at the Note from Seminar at CERN.
- Dong, G., Yan, Y., & Xie, M. (1998). *Vision-based hand gesture recognition for human-vehicle interaction*. Paper presented at the Proc. of the International conference on Control, Automation and Computer Vision.
- Du-Ming, T. (1997). Boundary-based corner detection using neural networks. *Pattern Recognition*, 30(1), 85-97. doi: [http://dx.doi.org/10.1016/S0031-3203\(96\)00057-X](http://dx.doi.org/10.1016/S0031-3203(96)00057-X)
- Fausett, L. V. (1994). *Fundamentals of neural networks: architectures, algorithms, and applications*: Prentice-Hall Englewood Cliffs.
- Fei-Fei, L., Fergus, R., & Perona, P. (2007). Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1), 59-70.
- Francke, H., Ruiz-del-Solar, J., & Verschae, R. (2007). Real-time hand gesture detection and recognition using boosted classifiers and active learning *Advances in Image and Video Technology* (pp. 533-547): Springer.
- Freeman, W. T., & Roth, M. (1995). *Orientation histograms for hand gesture recognition*. Paper presented at the International Workshop on Automatic Face and Gesture Recognition.

- Gao, W., Ma, J., Wu, J., & Wang, C. (2000). SIGN LANGUAGE RECOGNITION BASED ON HMM/ANN/DP. *International Journal of Pattern Recognition and Artificial Intelligence*, 14(05), 587-602. doi: doi:10.1142/S0218001400000386
- Gesture. (2014). Oxford Advanced Learner. *Oxford University Press*.
- Gonzalez, R. C., & Woods, R. E. (2002). *Digital Image Processing*: Prentice Hall; 2nd edition.
- Guodong, G., & Dyer, C. R. (2005). Learning from examples in the small sample case: face expression recognition. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(3), 477-488. doi: 10.1109/TSMCB.2005.846658
- Gupta, S., Jaafar, J., & Ahmad, W. F. W. (2012). Static Hand Gesture Recognition Using Local Gabor Filter. *Procedia Engineering*, 41(0), 827-832. doi: <http://dx.doi.org/10.1016/j.proeng.2012.07.250>
- Haykin, S. (1998). *Neural networks: a comprehensive foundation*. NJ: Prentice Hall Englewood Cliffs.
- Haykin, S. S. (2007). *Neural networks: a comprehensive foundation*: Prentice Hall Englewood Cliffs, NJ.
- Heskes, T. (1997). Balancing between bagging and bumping. *Advances in Neural Information Processing Systems*, 466-472.
- Huang, J. (1998). *Color-spatial image indexing and applications*. (Doctoral dissertation), Cornell University.
- Iannizzotto, G., Villari, M., & Vita, L. (2001). *Hand tracking for human-computer interaction with Graylevel VisualGlove: turning back to the simple way*. Paper presented at the Proceedings of the 2001 workshop on Perceptive user interfaces, Orlando, Florida.
- Ionescu, B., Coquin, D., Lambert, P., & Buzuloiu, V. (2005). Dynamic hand gesture recognition using the skeleton of the hand. *EURASIP Journal on Applied Signal Processing*, 2005, 2101-2109.
- Ishibuchi, H., Nakashima, T., & Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *Ieee Transactions on Systems Man and Cybernetics Part B-Cybernetics*, 29(5), 601-618. doi: Doi 10.1109/3477.790443
- Jain, A. K., Duin, R. P. W., & Jianchang, M. (2000). Statistical pattern recognition: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1), 4-37. doi: 10.1109/34.824819
- Joshi, G. D., & Sivaswamy, J. (2006). *A simple scheme for contour detection*. Paper presented at the Conference on Computer Vision Theory and Applications.
- Just, A. (2006). Two-Handed Gestures for Human-Computer Interaction. *IDIAP-RR*, 6(73), 94.

Kanan, C., & Cottrell, G. (2010, 13-18 June 2010). *Robust classification of objects, faces, and flowers using natural image statistics*. Paper presented at the Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.

Kevin, N. Y. Y., Ranganath, S., & Ghosh, D. (2004). *Trajectory modeling in gesture recognition using CyberGloves[®] and magnetic trackers*. Paper presented at the TENCON 2004. 2004 IEEE Region 10 Conference.

Khan, R. Z., & Ibraheem, N. A. (2012). Hand gesture recognition : a Literature. *International Journal of Artificial Intelligence & Applications (IJAIA)*, 03(04).

Kim, T. K., & Cipolla, R. (2007). Gesture recognition under small sample size *Computer Vision–ACCV 2007* (pp. 335-344): Springer.

Kinnebrock, W. (1995). *Neural Network, Fundamentals, Applications, Examples*: Galotia publications.

Kjeldsen, F. C. M. (1997). *Visual interpretation of hand gestures as a practical interface modality*. (Doctoral dissertation), Columbia University.

Kohonen, T., & Maps, S.-O. (1995). Springer series in information sciences. *Self-organizing maps*, 30.

Kosuge, K., & Hirata, Y. (2004, 22-26 Aug. 2004). *Human-Robot Interaction*. Paper presented at the Robotics and Biomimetics, 2004. ROBIO 2004. IEEE International Conference on.

Krueger, M. W. (1991). *Artificial reality II* (Vol. 10): Addison-Wesley Reading (Ma).

Kulkarni, V. S., & Lokhande, S. (2010). Appearance based recognition of american sign language using gesture segmentation. *International Journal on Computer Science and Engineering*, 2(03), 560-565.

Lamar, M. V., Bhuiyan, M. S., & Iwata, A. (2000). T-CombNET - A Neural Network Dedicated to Hand Gesture Recognition. In S.-W. Lee, H. Bülthoff & T. Poggio (Eds.), *Biologically Motivated Computer Vision* (Vol. 1811, pp. 613-622): Springer Berlin Heidelberg.

Li, X. (2005). Vision Based Gesture Recognition System with High Accuracy. *Department of Computer Science, The University of Tennessee, Knoxville, TN, 37996-33450*.

Licsár, A., & Szirányi, T. (2002). *Hand-Gesture Based Film Restoration*. Paper presented at the PRIS.

Liu, Y., & Zhang, P. (2009, 28-30 Oct. 2009). *An Automatic Hand Gesture Recognition System Based on Viola-Jones Method and SVMs*. Paper presented at the Computer Science and Engineering, 2009. WCSE '09. Second International Workshop on.

Manabe, H., & Zhang, Z. (2004, 1-5 Sept. 2004). *Multi-stream HMM for EMG-based speech recognition*. Paper presented at the Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE.

McConnell, R. K. (1986). US Patent, No. 4567610.

Meng, H., Furao, S., & Jinxi, Z. (2014, 6-11 July 2014). *Hidden Markov models based dynamic hand gesture recognition with incremental learning method*. Paper presented at the Neural Networks (IJCNN), 2014 International Joint Conference on.

Meng, J. E., Shiqian, W., Lu, J., & Hock, L. T. (2002). Face recognition with radial basis function (RBF) neural networks. *Neural Networks, IEEE Transactions on*, 13(3), 697-710. doi: 10.1109/TNN.2002.1000134

Ming-Kuei, H. (1962). Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2), 179-187. doi: 10.1109/TIT.1962.1057692

Mo, Z., Lewis, J. P., & Neumann, U. (2005). *SmartCanvas: a gesture-driven intelligent drawing desk system*. Paper presented at the Proceedings of the 10th international conference on Intelligent user interfaces.

Mo, Z., & Neumann, U. (2006, 04-07 Jan. 2006). *Lexical Gesture Interface*. Paper presented at the Computer Vision Systems, 2006 ICVS '06. IEEE International Conference on.

Moghadassi, A., Parvizian, F., & Hosseini, S. (2009). A new approach based on artificial neural networks for prediction of high pressure vapor-liquid equilibrium. *Australian Journal of Basic and Applied Sciences*, 3(3), 1851-1862.

Musa, A. K. (1998). *Signature recognition and verification by using complex-moments characteristics* (Masters thesis), University of Baghdad.

Naidoo, S., Omlin, C., & Glaser, M. (1999). Vision-based static hand gesture recognition using support vector machines. *Department of Computer Science, University of the Western Cape, South Africa*.

Nehaniv, C. L., Dautenhahn, K., Kubacki, J., Haegele, M., Parlitz, C., & Alami, R. (2005, 13-15 Aug. 2005). *A methodological approach relating the classification of gesture to identification of human intent in the context of human-robot interaction*. Paper presented at the Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on.

O'Hagan, R. G., Zelinsky, A., & Rougeaux, S. (2002). Visual gesture interfaces for virtual environments. *Interacting with Computers*, 14(3), 231-250. doi: [http://dx.doi.org/10.1016/S0953-5438\(01\)00050-9](http://dx.doi.org/10.1016/S0953-5438(01)00050-9)

Oprinescu, S., Rasche, C., & Bochao, S. (2012, 27-31 Aug. 2012). *Automatic static hand gesture recognition using ToF cameras*. Paper presented at the Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European.

Parvini, F., & Shahabi, C. (2007). An algorithmic approach for static and dynamic gesture recognition utilising mechanical and biomechanical characteristics. *International Journal of Bioinformatics Research and Applications*, 3(1), 4-23.

Pavlovic, V. I., Sharma, R., & Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7), 677-695. doi: 10.1109/34.598226

Picton, P. (2000). *Neural Networks*. Palgrave Macmillan; 2nd edition.

Pitas, I. (2000). *Digital image processing algorithms and applications*: Wiley-interscience.

Rose, H. A. (1919). The Language of Gesture. *Folklore*, 30(4), 312-315. doi: 10.1080/0015587X.1919.9719112

Rubaai, A., Kotaru, R., & Kankam, M. D. (1999, 1999). *Parallel computation of continually on-line trained neural networks for identification and control of induction motors*. Paper presented at the Industry Applications Conference, 1999. Thirty-Fourth IAS Annual Meeting. Conference Record of the 1999 IEEE.

Shet, V. D., Shiv, V., Prasad, N., Elgammal, A., Yacoob, Y., & Davis, L. S. (2004). *Multi-Cue Exemplar-Based Nonparametric Model for Gesture Recognition*. Paper presented at the ICVGIP.

Swapna, B., Pravin, F., & Rajiv, V. D. (2011). Hand Gesture Recognition System for Numbers Using Thresholding. *Computational Intelligence and Information Technology*, 250, 782-786.

Symeonidis, K. (1996). Hand Gesture Recognition using neural networks. *Neural Networks*, 13, 5.1.

Torres-Mendez, L. A., Ruiz-Suarez, J. C., Sucar, L. E., & Gomez, G. (2000). Translation, rotation, and scale-invariant object recognition. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 30(1), 125-130.

Triesch, J., & von der Malsburg, C. (1996, 14-16 Oct 1996). *Robust classification of hand postures against complex backgrounds*. Paper presented at the Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on.

Trucco, E., & Verri, A. (1998). *Introductory techniques for 3-D computer vision* (Vol. 201): Prentice Hall Englewood Cliffs.

Turk, M. (1999). Gesture Recognition. *Microsoft Research*, Retrieved from <http://www.ilab.cs.ucsb.edu/projects/turk/TurkVEChapter.pdf>.

Umbaugh, S. E. (1997). *Computer Vision and Image Processing: A Practical Approach Using Cviptools with Cdrom*: Prentice Hall PTR.

Vieriu, Goras, & Goras. (2011, June 30 2011-July 1 2011). *On HMM static hand gesture recognition*. Paper presented at the 10th International Symposium on Signals, Circuits and Systems (ISSCS), 2011.

Vieriu, R.-L., Goras, B., & Goras, L. (2011). *On HMM static hand gesture recognition*. Paper presented at the Signals, Circuits and Systems (ISSCS), 2011 10th International Symposium on.

- Wachs, J., Kartoun, U., Stern, H., & Edan, Y. (2002, 2002). *Real-time hand gesture telerobotic system using fuzzy c-means clustering*. Paper presented at the Automation Congress, 2002 Proceedings of the 5th Biannual World.
- Wang, J., Kunieda, K., Iwata, M., Koizumi, H., Shimazu, H., Ikenaga, T., & Goto, S. (2006, 12-15 Dec. 2006). *Multi-resolution Analysis based Salient Contour Extraction*. Paper presented at the Intelligent Signal Processing and Communications, 2006. ISPACS '06. International Symposium on.
- Winnemöller, H. (1999). " Practical Gesture Recognition for controlling Virtual Environments. *Project for Bachelor of Science (Honours) of Rhodes University*.
- Wu, Y., & Huang, T. S. (1999). Vision-Based Gesture Recognition: A Review. In A. Braffort, R. Gherbi, S. Gibet, D. Teil & J. Richardson (Eds.), *Gesture-Based Communication in Human-Computer Interaction* (Vol. 1739, pp. 103-115): Springer Berlin Heidelberg.
- Wu, Y., & Huang, T. S. (2001). Hand modeling, analysis, and recognition - For vision-based human computer interaction. *Ieee Signal Processing Magazine*, 18(3), 51-60.
- Yoon, H.-S., Soh, J., Bae, Y. J., & Seung Yang, H. (2001). Hand gesture recognition using combined features of location, angle and velocity. *Pattern Recognition*, 34(7), 1491-1501. doi: [http://dx.doi.org/10.1016/S0031-3203\(00\)00096-0](http://dx.doi.org/10.1016/S0031-3203(00)00096-0)
- Zabulis, X., Baltzakis, H., & Argyros, A. (2009). Vision-based hand gesture recognition for human-computer interaction. *The Universal Access Handbook*. LEA.
- Zhou, R., Junsong, Y., Jingjing, M., & Zhengyou, Z. (2013). Robust Part-Based Hand Gesture Recognition Using Kinect Sensor. *Multimedia, IEEE Transactions on*, 15(5), 1110-1120. doi: 10.1109/TMM.2013.2246148