ARABIC SIGN LANGUAGE RECOGNITION USING AN INSTRUMENTED GLOVE

SALAH M. S. AL-BURAIKY

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the

Requirements for the degree of

MASTER OF SCIENCE

In

ELECTRICAL ENGINEERING

September 2004

KING FAHD UNIVERSITY OF PETROLIUM & MINERALS

DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis written by Salah Mohammed Saeed Al-Buraiky under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE in Electrical Engineering**.

THESIS COMMITTEE

Dr. Mohamed Mohandes

Prof. Talal Halawani

Prof. Mahmoud Dawoud

Dr. Hussain Al-Duwaish

Dr. Ahmed Yamani

Dr. Jamil Bakhashwain
(Department Chairman)

Prof. Osama Al-Jannadi
(Dean of Graduate Studies)

Date

The research of this thesis has is supported by Prince Salman Center for Disability Research (PSCDR).

PSCDR Project No.: 02-R-0023-NE-02-AT-1.

مركز الأمير سلمان لأبحاث الاعاقة

Prince Salman Center for Disability Research

Dedicated to:

My Parents

ACKNOWLEDGMENTS

Acknowledgment is due to the **King Fahd University of Petroleum & Minerals** and to **Prince Salman Center for Disability Research** for supporting this research. I wish also to express my deep appreciation for **Dr. Mohamed Mohandes** who served as my major advisor and whose persistent encouragement and continuous support were vital to the completion of this work. I also wish to thank the other members of my thesis committee **Dr. Talal Halawani**, **Dr. Mahmoud Dawoud, Dr. Hussain Al-Duwaish**, and **Dr. Ahmed Yamani**. I 'd like to also express my deep appreciation to **Dr. Waleed Kadous** who welcomed my questions through e-mail and who has kindly provided me with the PGSI, a major component of the system proposed by this thesis. Thanks are also due to **Mr. Jafar Al-Okakah**, who is a specialist in Arabic Sign Language and in teaching students with disabilities, for his great help in understanding Arabic Sign Language and in sample collection. I'd like also to thank **Mr. Mahdi Al-Faraj**, **Mr. Abduljaleel Al-Shaikh** and **Mr. Faris Al-Mughasil**, the fluent Arabic Sign Language performers who have patiently provided the samples needed for the development of the system proposed by the thesis. Thanks are also due to my brother, **Salam Al-Buraiky,** for recording, in his voice, the Arabic words used for providing the sound output of the system.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

THESIS ABSTRACT

Name:	SALAH M. S. AL-BURAIKY
Title:	ARABIC SIGN LANGUAGE RECOGNITION USING AN INSTRUMENTED GLOVE
Degree:	MASTER OF SCIENCE
Major Field:	ELECTRICAL ENGINEERING
Date of Degree:	SEPTEMBER 2004

Automated recognition of sign language greatly facilitates communication with deaf and non-vocal people. It is also important for the development of human-machine interface. This thesis proposes a system for automatically recognizing Arabic Sign Language using an instrumented glove as an interfacing device and the support vector machine algorithm as a classification algorithm. The instrumented glove used is the PowerGlove, which is chosen for its availability and low cost. The support vector machine algorithm is chosen because it is a relatively new approach to machine learning and because it has many attractive features compared to competing approaches. Promising results were obtained using the approach presented by this thesis.

Keywords: Arabic Sign Language, Support Vector Machine, Instrumented Glove.

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS, DHAHRAN.

SEPTEMBER 2004

خلاصة الرسالة

اسم الطالب :	صلاح محمد سعيد البريكي
عنوان الرسالة :	التعرف على لغة الإشارة العربية باستخدام قفاز إلكتروني
الدرجة:	الماجستير في العلوم
التخصص:	هندسة كهربائية
تاريخ الشهادة:	1425 هـ

إنَ إيجاد وسيلةٍ إ لكترونيةٍ لترجمة لغة الإشارة إلى لغة مقروءةٍ أو منطوقة ييسر إلى حدٍ كبيرٍ التواصل بين الصَم وبقية أفراد المجتمع و يساعد على تقدَم وسائل إدخال البيانات إلى الآلات الإلكترونية. تقدم هذه الرسالة تصميماً لنظامٍ يمكنه تمييز لغة الإشارة العربية باستخدام قفَازٍ إلكتروني و خوارزم المتجهات الداعمة حيث تم استخدام قفَاز ناينتندو لسهولة الحصول عليه ولكلفته المنخفضة و تم استخدام خوارزم المتجهات الداعمة لحداثته و لما له من مميزات تفوق تلك التي للخوارزمات المشابهة. و قد تم تحقيق نتائج واعدة باستخدام النظام المقترح.

درجة الماجستير في العلوم

جامعة الملك فهد للبترول والمعادن

أيلول 2004

CHAPTER 1

INTRODUCTION

1.1 Overview

A sign language maps the letters, words or expressions of a certain language to a set of hand gestures enabling an individual to communicate by using hands and gestures rather than by speaking. Automated recognition of sign languages is important in a world that is showing a continuously increasing interest in removing barriers faced by physically challenged individuals in communicating and contributing to the community and the workforce. Systems capable of recognizing sign-language symbols can be used as means of communication between hearing-impaired and normal people. They can be used to implement dictionaries for the various sign-languages and can be used as substitutes for voice recognition either for the use of the deaf community or for the use in environments where voice recognition is needed but not possible. In addition, virtual reality systems can largely benefit from the advances and successes in sign language computerized recognition.

Similar to the situation in the fields of voice and hand writing recognition, many difficulties are encountered in designing systems for sign-language interpretation. One difficulty is that the use of traditional programming paradigms makes the system overwhelmingly complex and hence impractical. This dictates resorting to machine-learning methods. Another difficulty encountered is the interface issue. Ideally, the interface should deliver accurate measurements to the processing machine, have low cost and provide input in a form that requires low pre-processing overhead. Building a system that satisfies these three requirements is not easy and design compromises must be accepted to build a practical system.

1.2 Arabic Sign Language

Until the recent publication of the unified Arabic Signs dictionary, only numbers and Arabic letters (finger spelling) were standardized among the users of the Arabic Sign Language. In fact, differences in signs might be found among speakers in the same country and sometimes even among speakers in neighboring cities. This is because signs are mostly created by the hearing-impaired individuals themselves and are highly influenced by the local environment. An example of this trend is referring to the white color by pointing to one's chest in Saudi Arabia, which originates from the fact that most males wear a white ``thoub''. The Arabic Sign Language accent used in this thesis is basically the Saudi Arabian accent, except for numbers and letters, which are the standard signs adopted by all Arabic speaking countries.

Compared to other sign languages, not much has been done in the automation of Arabic Sign Language. However, attempts to build systems for automated Arabic Sign Language have been found in the literature. One system used moment invariants as features and the Support Vector Machine Algorithm for automating the recognition of the letters of the Arabic alphabet signs [1]. Another system attempted to recognize the 30 letters of the Arabic alphabet signs using a video camera as a system input device and neuro-fuzzy systems for recognition [2].

1.3 The Human-Machine Interface

Interfaces in sign language systems can be categorized as being based on one of two major approaches. These approaches are the vision-based approach and the direct-device approach. The direct-device approach uses measurement devices that are in direct contact with the hand such as instrumented gloves, flexion sensors, styli and position-tracking devices. On the other hand, the vision-based approach captures the movement of the singer's hand using a camera that is sometimes aided by making the signer wear a glove that has painted areas indicating the positions of the fingers or knuckles.

Vision-based systems include, as an example, the system proposed by Davis and Shah in which a camera and a glove with bright points painted on the edges of the fingers are used [3]. A more complex system in which each knuckle has a color code is proposed by Dorner [4]. Another system, designed by Starner, uses two differently colored simple gloves [5]. Isibuchi and his colleagues and Krueger proposed vision-based systems that use 3D predictive modeling and a hand color probability transformation with no special gloves [6] [7]. The main advantage of vision-based systems is that the user isn't encumbered by any complex devices. Their main disadvantage, however, is that they require a large amount of computation just to extract the hand position before performing any analysis on the images.

The direct-device approach, on the other hand, requires much less computing power. The direct-device approach has been used by James Kramer, Peter Vamplew and others [8] [9]. Systems based on the direct-device approach typically use instrumented gloves that could be assisted by other devices. The first widely known instrumented glove was the Digital Data Entry Glove designed by Dr. Grims at AT&T Bell Labs and patented in 1983 [10]. It was originally proposed as an alternative input device to the keyboard and worked by generating ASCII characters according to finger positions. The gloves had finger flex sensors, tactile sensors at their tips, orientation sensors and wrist-positioning sensors. The VPL DataGlove, designed by Thomas Zimmerman, followed. It used optical flex sensors that had fiber optic strands with a light at one end and a photodiode at the other. A simplified version, the Z-glove, was also built by Zimmerman. It used fiber optic devices to measure the angles of each of the first two knuckles of the fingers and was usually combined with a Polhemus tracking device. The Z-glove was the first commercially available instrumented glove. The Exon Dextrous Hand Master was developed afterwards and introduced a high level of accuracy. It had 8-bits of accuracy, 20 degrees of freedom and a measurement frequency of 200 Hz [11].

The CyberGlove, originally developed by James Kramer at Stanford University, is another patented instrumented glove [11]. It was designed specifically for the Talking Glove Project and was patented in 1991. It comes in two models: a model that has 18 degrees of freedom and can measure the bend of the first two knuckles on each finger and a model that has 22 degrees of freedom and can measure the bend of the three knuckles on each finger. Both models use a strain gauge to measure the abduction between fingers and a number of additional measures around the thumb. CyberGlove is now a commercial product of Immersion Corporation [12].

The PowerGlove manufactured by Mattel/Nintendo and based on VPL's glove is highly attractive price-wise, although it has less accuracy and fewer sensors compared to other gloves. In addition to its relatively low price, the PowerGlove can be easily connected to an RS-232 serial port by using an interface box. The PowerGlove offers, in conclusion, a less accurate but a highly cost effective alternative to other instrumented gloves [11].

1.4 Literature Review

1.4.1 The Image-based Approach

Many researchers proposed or constructed image-based sign language recognition systems. A variety of pattern recognition schemes were employed by different researchers including artificial neural networks, instance-based learning, Hidden Markov Models (HMM) and an approach based on simple finite-state machines. Charayaphan and Marble proposed a system that attempted to recognize American Sign Language (ASL) from images by using three movement related features: The position at which the hand stopped, the course in space the hand followed and the shape of the hand when it stopped. A camera was used to capture the images and a method inspired by instance-based learning techniques was adopted for learning. The locations of the training set were averaged and used as typical examples. The system was able to successfully recognize 27 out of 31 ASL symbols [13].

Davis and Shah proposed a system for gesture recognition comprised of a black-and-white camera and a black glove with white finger tips. A finite-state machine with four states was used for recognition. The four states defined were: “keep hand in start position”, “slowly move hand to gesture position”, “keep hand in position for the desired duration” and “move fingers back to start position”. The finite-state machine was used to recognize a limited set of seven gestures (left, right, up, down, rotate, grab and stop). The major attractive feature of this system is its simplicity. However, this system has serious disadvantages when used for sign language recognition such as requiring stop and start signals, low frequency of operation (4 Hz) and its inability to monitor hand position. In addition, the finite state machine was found susceptible to going into the wrong state in some experiments [3].

The system developed by Starner for recognizing continuous American Sign Language (ASL) consisted of a color camera and two gloves; a yellow glove for the right hand and an orange one for the left hand. An SGI Indigo workstation was used for processing the images. The images were taken at a rate of five shots per second and features such as x-y position and the bounding ellipse were extracted. The pattern recognition mechanism relied on an extension of Hidden Markov Models (HMM) that is able to handle distributions of multiple variables and not just simple output symbols. This extension enabled the system to use features extracted directly from the data and eliminated the need for preprocessing. The system was designed to recognize signs from a set of 40 signs under the assumption that signs had one grammatical class (i.e. there are no signs that could be used both as verbs and as nouns). In addition, the order of words was restricted to ``pronoun, verb, noun, adjective, pronoun" with the possibility of leaving out pronouns and adjectives. With strict grammar, the system was able to achieve accuracy rates exceeding 99%. The major shortcoming for this system is that it doesn't consider the movements of fingers which are essential for finger-spelling and large-lexicon systems [5].

Dorner devised a system that uses a cotton glove that has colored bands each corresponding to a specific joint on a specific finger. That is, each finger joint is marked by three rings: a central one that indicates the finger the joint belongs to and two bands on both sides of the central one indicating a specific joint on the finger. This design attempts to construct a model of the hand in time and 3D space [14]. Hagen, on the other hand, designed a deductive database for American Sign Language that includes features that are above the lexical level such as the movement of eyebrows to indicate the type of the sentence signed (indicating that the sentence is a question for example). This deductive database was shown to be able to successfully translate from a standardized form of ASL to spoken English. A complete sign language recognition system is realized by integrating Hagen's and Dorner's systems together [14].

Another image-based system for recognizing the American Sign Language (ASL) alphabet (finger spelling) was proposed by Isaac and Foo. It uses a camera for capturing 2D hand images and uses neural networks for recognition. The features used were based on wavelet decomposition methods. Two types of features were tested: features derived from energy and entropy and features derived from the lowest reasonable level of decomposition of each letter. The system was able to achieve a recognition rate of 99.9% on 24 letters of the alphabet (J and Z were not included)[15].

Bauer and Kraiss used the image-based approach for recognizing the German Sign Language (GSL). A video camera was used as an input device and Hidden Markov Models (HMM) were used for recognition. In order to reduce the number of required patterns needed for initial and future training when new signs are added to the system, they used sign subunits rather than whole signs for training the Hidden Markov Model (HMM). Unlike most other systems based on dividing signs into subunits, this system uses self-organized subunits that are derived from the data itself rather than phonemes. A total number of 150 subunits were used and a recognition accuracy of 92.5% was achieved for a set of 100 signs that were used for training, while a recognition accuracy of 81% was achieved for a set of 50 signs that were not used in the training for the subunits [16].

Brasher, Starner, Lukowicz, Junker and Troster designed a wearable mobile sign language recognition system that is primarily image-based but utilizes a number of miniature accelerometers attached to the wrists and the torso of the signer as well. The system consists of a hat-mounted camera, a wearable computer and the three miniature acceleration sensors. Hidden Markov Model (HMM) was used as a recognition algorithm. The system performance was tested using three types of data: accelerometer data, vision data and combined accelerometer and vision data. It was able to achieve an accuracy of 52.38% on testing with vision data, 65.87% on testing with accelerometer data and an accuracy of 90.48% using both [17].

Yoon and Jo designed a vision-based system for recognizing the Korean sign language alphabet, which consists of 16 consonants and 14 vowels. The system consisted of a video camera connected to a computer with an image capturing board. To aid the system in extracting the hand shape form images, the user of the system wore gloves with different colors for each hand. Moment invariance was used for recognition and experiments with one person showed a recognition rate of 97% [18].

1.4.2 The Direct-device Approach

The GloveTalk project of Hinton and Fles used the VPL DataGlove Mark II as an interface, neural networks to process the input and a speech synthesizer to generate vocal words from the outputs of the neural networks. The speech synthesizer was a DecTalk module that executed on a SGI 4D/240S workstation. The glove was enhanced by a Polhemus tracker for position and orientation tracking. In GloveTalk, hand shape determined the root word while movement's direction determined the ending of the word. A separate neural network was used for each of these parameters and an additional network was used for classifying the signs. The problem with GloveTalk is the high complexity of the neural networks used and that its operation is computationally intensive [19].

GloveTalk-II was a refinement of GloveTalk with the objective of adapting the system to general-purpose use. This simplified version of GloveTalk had three neural networks instead of five. One network was used to decide whether the current letter was a vowel or a consonant; one was for individual vowel selection and one for individual consonant selection. A foot-pedal was used for controlling the volume of the generated sound and a keyboard was added to the system for generating stop sounds (e.g. B, D, P) because they are difficult to recognize. A CyberGlove replaced the DataGlove used in

Glove-Talk in order to provide more information to the system since the CyberGlove was equipped with more sensors compared to the DataGlove [19] [20].

Another system utilizing neural networks is the system proposed by Murakami and Taguchi for recognizing the Japanese finger alphabet using the VPL DataGlove and a recurrent neural network. In their work, they first trained the recurrent network to recognize 42 hand shapes form the Japanese finger alphabet and achieved a success rate of 98%. They then successfully attempted to make the system recognize continuous finger-spelling. After that, they attempted to recognize signs corresponding to whole words and chose a set of ten distinct signs to experiment with. They were able to achieve a success rate of 96% by using a recurrent network with 93 nodes in the input layer, 150 in the hidden and 150 in the context layer. The input consisted of the three past frames, where each frame provided information about finger position, orientation information, relative and absolute position [21].

Peter Vamplew also proposed a system for recognizing the Australian sign language (Auslan) using neural networks. The system, called SLARTI, used the CyberGlove as an input device and a Polhemus tracking device which provided six additional degrees of freedom. It used four feature extraction networks each trained for recognizing one feature. The feature vector produced by the four networks was used to perform the overall classification. A maximum recognition rate of 96.6 % was achieved over a test set supplied by the registered users (users who supplied the training signs), while a maximum recognition rate of 89.9 % was achieved over a test set supplied by unregistered users (users who didn’t supply any training samples) [9].

Roman Erenshteyn, Pavel Laskov, Richard Foulds, Lynn Messing and Garland Stern employed neural networks to address the problem of recognizing dynamic signing (signing at natural speed) of the American Sign Language. The proposed system uses the CyberGlove for interfacing and neural network systems for recognition. Two neural network systems were used: one for recognizing manual alphabet shapes (finger spelling) and one for recognizing hand shapes (words). The system used for finger spelling recognition consisted of a multi-class recognition neural network with 18 inputs and 26 outputs. Satisfactory results were obtained when using more than 100 neurons in the hidden layers and more than 800 samples for training. Backpropagation with an adaptive learning rate and momentum was used for training. The second system (used for recognizing hand shapes) consisted of a hierarchy of neural networks in order to reduce the amount of time needed for training which is expected to be relatively large due to the large number of classes involved (77 classes). The bottom level of the hierarchy consisted of 15 neural networks each with 3 to 7 outputs, while the top level had a network that decided the most competent network below. Recognition accuracy exceeding 93% was achieved [22].

Weissmann and Salmon investigated the use of neural networks for gesture recognition. Two types of neural networks were investigated: Backpropagation networks and radial basis function networks. Three 3-layer backpropagation networks with different interconnection schemes were explored:

1. Network BP_{full} : In this network, all hidden nodes were fully connected to all input nodes (30 hidden nodes).

2. Network BP_{pair}: In this network, each hidden node was connected to input units pertaining to the measurements coming from a pair of fingers (15 hidden nodes as the measurements of thumb rotation, palm arch, wrist pitch and wrist yaw were treated as measurements from a sixth finger).

3. Network BP_{triple}: In this network, each hidden node was connected to input units pertaining to the measurements coming from finger triples (15 hidden nodes).

To determine the gesture corresponding to the input, the system outputs the gesture that corresponds to the network output of the highest value (among outputs exceeding an experimentally determined threshold of .8). The networks with partial interconnections were experimented with in order to exploit a possible correlation between pairs (or triples) of fingers and gestures. It was found that the BP_{full} network doesn't perform well, while the two other networks achieved recognition rates of 99.5 % and 92.0 %, respectively. The radial basis function network showed good performance only when the training set size was increased. The main advantage of radial basis networks is that they can be retrained at run time because they require a small training time when compared to multi-layer perceptron backpropagation networks [23].

Jiangqin and his colleagues proposed a Chinese Sign Language recognition system that is based on both neural networks and Hidden Markov Models (HMM). The system works on the word level and uses the CyberGlove as its input device. Experiments performed using simulation produced a recognition rate over 90% [24].

Takahashi and Kishino proposed a system for interpreting the Japanese Kana manual alphabet, which consists of 46 signs, using a VPL DataGlove and principal component analysis (PCA). They built a table that maps the positions of individual fingers and joints to hand shapes which were then classified. The system was able to successfully recognize 30 out of the 46 signs [25].

Vogler and Metaxes addressed the problem of recognition systems scaling with the increase of the vocabulary set size. They approached the problem of scalability using two key ideas. The first was breaking the signs to phonemes (the elementary constituents of signs) and the second was modeling the phonemes using a parallel extension of Hidden Markov Models (HMM). The first key idea is based on the fact that the number of phonemes is much less than the number of signs and the second idea is based on the fact that phonemes can occur in parallel. The proposed system was tested using a MotionStar 3D tracking system and a database consisting of 400 training sentences, 99 testing sentences. The system achieved 84.85 % accuracy on the sentence level and 94.23 % on the sign level [26].

The system designed by Kadous (GRASP) used a PowerGlove as an input device and explored the use of instance-based and symbolic-based learning for classifying signs. The data set used for constructing the classifier consisted of 6650 examples collected from 5 people. An SGI workstation was used to acquire data and extract features from it. Many different features were investigated. This included energy, distance, time, bounding boxes and the use of simple time division. Simple time division, which basically averages the raw quantities measured by the glove and uses them as features, was found to achieve good results. The system was able to recognize 95 signs of the Auslan language at an accuracy of about 80% [27].

The Talking Glove Project carried out by Kramer and Liefer is an attempt to integrate several technologies together to build a system that can be used for communication between deaf, deaf-blind and non-vocal individuals. The CyberGlove mentioned earlier was originally designed and built as part of this work. Kramer used a prototyping approach for gesture recognition. In his work, each letter was associated with a prototype (called a beacon) that represented a point in the hand state vector space. Each of these points was surrounded by a hypersphere (called a recognition ball) and a letter was recognized when the hand entered its recognition ball. Each recognition ball was surrounded by another hypersphere (called a hysteresis ball) and it was required that the hand gets out of the hysteresis ball before repeating a sign. The system was found to be practical [8].

Another glove-based system was proposed by Mehdi and Khan. It used a seven-sensor glove form 5DT and a multi-layer perceptron neural network to recognize the alphabet of the American Sign Language (ASL). The glove used provided only measurements for the bend of each finger and measured the tilt and the rotation of the hand. The neural network used had 7 neurons in the input layer, 54 neurons in the hidden layer and 26 neurons in the output layer. The system was able to achieve an accuracy of around 88% [28].

Gao, Fang and Ma proposed a system for recognizing the Chinese Sign Language (CSL) using a pair of CyberGloves and three Polhemus tracker for input and Hidden Markov Models (HMM) for recognition. The HMM used was augmented with self-organizing feature maps (SOFM) to increase its ability to discriminate between patterns. In addition, a self-adjustment algorithm was employed to further enhance recognition. The system was found to achieve a recognition rate of 90.7% with the HMM alone, a recognition rate of 95.3% with HMM and SOFM and a recognition rate of 96.6% with self-adjustment [29][30]. With the same input devices, Gao, Wang and Shan investigated the advantage of using phonemes as basic recognition units rather than using whole words in scaling the system in terms of the number of words it can recognize. An HMM was built for each phoneme (around 2400 phonemes) and each sign in the word set (5119 words, almost all words in CSL) was decomposed to its constituent phonemes. The system was tested over the 5119 words and was found to achieve a recognition rate over 90% [31]. SOFM-enhanced HMM and the input system consisting of two CyberGloves and three trackers was used by Gao, Chen, Fang and Yang to build The Chinese Sign Language Dialog System (CSLDS), which is a complete system for translating sign language sentences to voice and to graphical facial animations. The CSLDS can provide continuous, real-time recognition with an accuracy of 91.6 % over a data set of 5113 words [32].

CHAPTER 2

THE PROPOSED RECOGNITION SYSTEM

2.1 Primary Goal and Objectives

The primary goal of this research work is to design and build a system for recognizing a subset of the Arabic sign language using a low cost instrumented glove as an interface and the support vector machine algorithm as a classifier. The main design objectives to be realized by the systems are the following:

1. To contribute to research directed towards facilitating the communication with physically-challenged people (the hearing-impaired).
2. To investigate an unexplored application (sign language recognition) of the support vector machine algorithm which has shown highly promising results in other applications.
3. To provide research work on the automated recognition of Arabic Sign Language which has little research done in its automated recognition.
4. To aid the unification and standardization of Arabic Sign Language accents through automating the recognition of Arabic Sign Language.

2.2 System Structure

The proposed system basically consists of an instrumented glove that is connected through the serial port to a workstation running the support vector algorithm. The major hardware and software components of the system are presented below.

2.3 System Hardware

The three major hardware components of the proposed system are:

1. An Instrumented glove: This is the direct interface between the system and the individual performing the sign. It takes measurements indicating the location and orientation of the hand at each instance of time. The instrumented glove used for this work is the Nintendo/Mattel PowerGlove.
2. A computing device: The basic role of the computing device is to execute the machine learning code (the classification machine). It receives frames generated by the glove as a result of performing a certain sign, extracts features to be used as an input to the classification machine. In this work, a standard PC is used as the processing device while the classification algorithm used is the Support Vector Machine (SVM).
3. An interface: The interface providing the link between the glove and the processing machine (the PC). Its role is to receive the frames containing location and orientation measurements form the glove, perform some processing and deglitching on them, and then re-send them to the PC in a specific, well defined format. It also interprets the commands coming from the PC and configures the glove accordingly. The interface used is in this research work is the PGSI (PowerGlove Serial Interface) unit designed

and built by the Students Chapter of the Association for Computing Machinery at the University of Illinois, Urbana-Champaign campus (ACM at UIUC).

2.4 System Software

The software running on the computer includes:

1. Data acquisition software for acquiring data from the glove. It reads the structured measurements frames sent through the serial port and stores them in a human readable format (ASCII text).
2. Data processing software for processing the raw data and extracting features. It cleans the received data and extracts features from it.
3. Machine learning software that implements the support vector machine algorithm for both training and testing (software that actually implements the classifier).
4. Sound player software that is used to play one of the pre-recorded words depending on the output of the classification machine.

The Support Vector Machine package used in this work is the SVMTorch II package [33]. The feature extraction and data processing software uses the publicly available glove library, Perl scripts written by Kadous for GRASP and UNIX shell scripts I wrote.

2.5 Operation Modes

The system has two modes of operation: training mode (offline mode) and recognition mode (online mode). In training mode, the support vector machine is taught to recognize the signs by presenting it with labeled examples. Each example consists of a vector of features extracted from the raw measurements generated by the instrumented glove and a label that corresponds to a specific sign. The purpose of operating the system in this mode is to generate support vector machines that are adapted to recognize a performed sign form the features extracted from that sign's glove measurements. In recognition mode, the adapted support vector machines are used to actually recognize the performed signs.

The features extracted from the performed sign are used as an input to the adapted (trained) support vector machines, which classifies it and then the word corresponding to the sign is either written on the screen or spoken out from the speakers. Figure 2.1 shows a flow diagram of the two modes of the system.

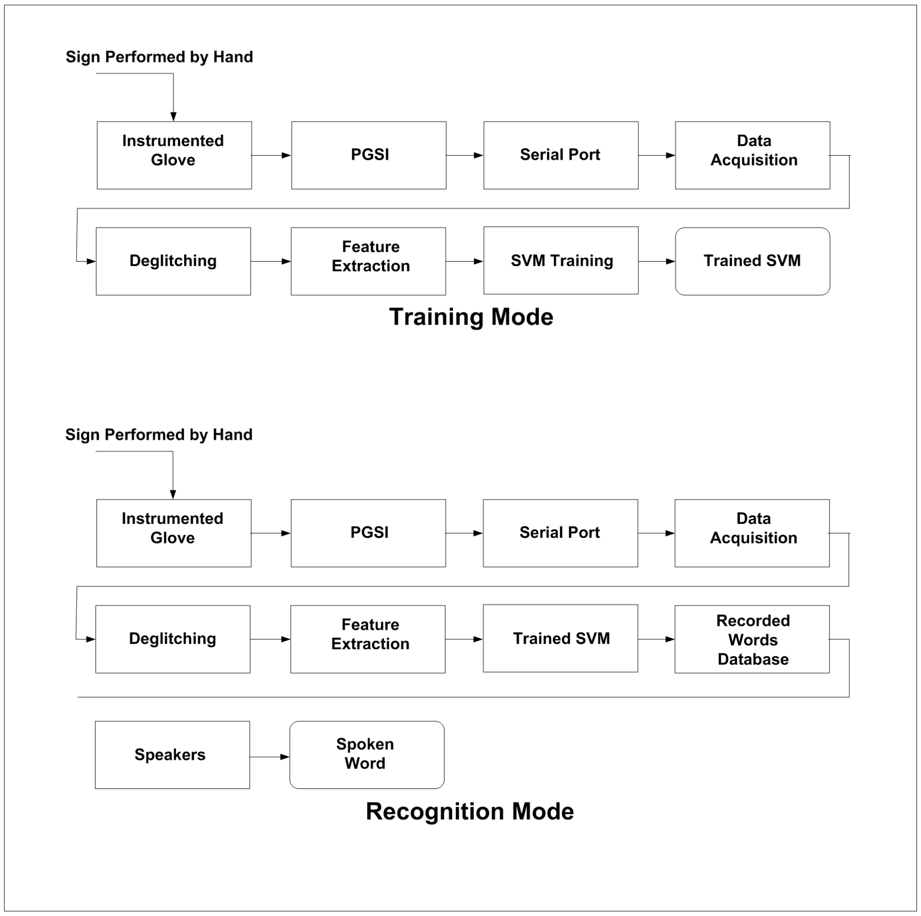

Figure 2.1: Flow diagrams for the system operation

CHAPTER 3

THE POWERGLOVE

3.1 The PowerGlove Hand-Machine Interface

The PowerGlove is a low-cost instrumented glove that can measure the position of the hand in space, its rotation angle about the wrist and the bends of the fingers. It was developed by Abrams-Gentile Entertainment Incorporation (AGE Incorporation) which cooperated with the Mattel toy company to manufacture it for the use with the Nintendo Entertainment Systems (NES) video gaming machines. It is basically a downgraded version of the VPL DataGlove with a lower cost and lower measurement accuracy. AGE and Mattel performed many design modifications aimed towards reducing the cost of the glove (compared to the VPL DataGlove) and making it more suitable for the use with video games.

Examples of such modifications include the use of an ultrasonic tracking mechanism rather than using the expensive Polhemus tracker, having sensors on four fingers only rather than all fingers, using resistive-ink flex sensors rather than the expensive optical fibers to measure the bend of fingers and having a flex measurement resolution of two bits per finger rather than the eight bits of resolution provided by the VPL DataGlove [27].

The PowerGlove has a molded plastic body with two ultrasonic transmitters fitted on the part covering the hand and control buttons fitted on the part partially covering the arm. An ultrasonic tracker is used to locate the glove in space with respect to a companion unit with an accuracy of one fourth of an inch. Those two transmitters send pulses to three receivers mounted to an L-shaped bar that is usually put on the screen, and the time for these pulses to reach the receiver is used for tracking the position of the hand .The rotation of the wrist is also measured by the trackers. The bend of the fingers is measured using resistive-ink flex sensors that are embedded into the plastic body of the glove. The flex sensors measure the bend of the thumb, index, middle and ring fingers (the bend of the little finger is not measured). They work by measuring the electrical resistance of the conductive ink embedded in the plastics covering the fingers. Structured frames containing the measured information are sent to the device connected to the glove [27]. Figure 3.1 shows the PowerGlove.

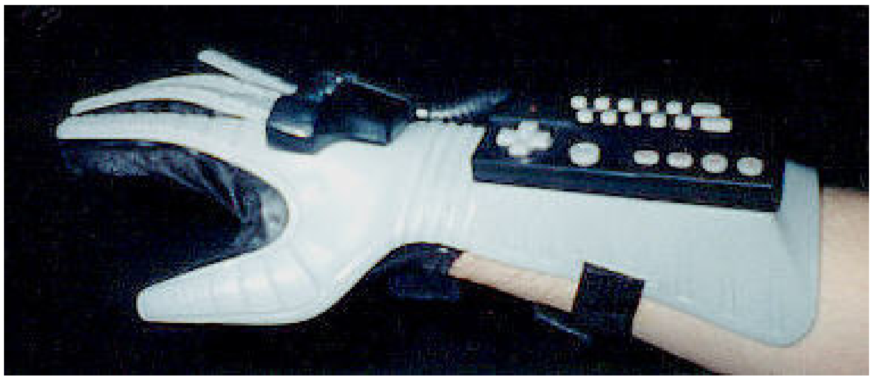

Figure 3.1: The PowerGlove

(Courtesy of Waleed Kadous)

3.2 PowerGlove Interfacing

The first standard device for interfacing the PowerGlove with the serial port of a computer was the PGSA (PowerGlove Serial Adaptor), which was designed by the original designers of the Nintendo PowerGlove, namely the Abrams-Gentile Entertainment Incorporation (AGE Inc.,). Another popular interface that appeared after the PGSA is the ``Menelli box", which is named after its designer Ron Menelli. Both interface protocols (AGE and Menelli) can operate either in continuous or request mode. In continuous mode, a new frame of information is sent from the PowerGlove to the host computer as soon as the information is received, while in request mode, the frame isn't sent unless the host computer sends a ``send frame" command. The PGSI (PowerGlove Serial Interface) is the PowerGlove interface designed by a special computer architecture group (SIGArch) in the student chapter of the Association for Computing Machinery (ACM) at the University of Illinois, Urbana-Champaign campus (UIUC). It was designed after the PGSA and the Menelli box and is capable of transparently emulating either one of them. The PGSI can automatically switch to the proper mode of emulation depending on the mode used by the software running on the host computer. The AGE protocol provides more complete information and provides more control commands. Continuous mode is more suited for real-time applications like the continuous tracking of the hand.

3.2.1 PGSA

The PowerGlove Serial Adapter is the interface box designed the Abrams-Gentile Incorporation (the designer of the PowerGlove). The AGE interface protocol uses the same frame format for both continuous and request mode. The AGE frame format is shown in table 3.1.

TABLE 3.1: Detailed Frame format for the AGE protocol.

Byte Position	**Name**	**Description**
00	Header	Frame starts
01	Header	Frame starts
02	X	X dimension reading
03	Y	Y dimension reading
04	Z	Z dimension reading
05	Rotation	
06	Flex	
7:6		Thumb flex
5:4		Forefinger flex
		Middle finger flex
3:2		Ring finger flex
07	Keys	Keys being pressed
08	GSTAT1	General status 1
09	GSTAT2	Unused
10	RECVALS	Receiver values
5		Left top receiver from left transmitter
4		Right bottom receiver from left transmitter
3		Right top receiver from left transmitter
2		Left top receiver from right transmitter
1		Right bottom receiver from right transmitter
0		Right top receiver from right transmitter

3.2.2 The Menelli Box

This interface is named after its designer Ron Menelli, who made it publicly available online. At the core of the design is a Motorola microcontroller that provides most of the interfacing functionality. The Menelli box sends six-byte frames in request mode and seven-byte frames in continuous mode [34]. Tables 3.2 and 3.2 respectively show the frame format of the Menelli protocol in continuous mode and in request mode.

TABLE 3.2: Frame format for the Menelli protocol in continuous mode.

Position in Frame	Name	Description
00	Header	Signals start of frame
01	X	x-direction reading
02	Y	y-direction reading
03	Z	z-direction reading
04	Rotation	Rotation value in 30 degree increments
05	Flex	Position of thumb and first three fingers
06	Switch	Key codes

TABLE 3.3: Frame format for the Menelli protocol in request mode.

Position in Frame	Name	Description
00	X	x-direction reading
01	Y	y-direction reading
02	Z	z-direction reading
03	Rotation	Rotation value in 30 degrees increments
04	Flex	Position of thumb and first three fingers
05	Switch	Key code

3.2.3 PowerGlove Serial Interface (PGSI)

The PGSI is the interface used for this thesis project. Just like the Menelli box, it is based on the Motorola microcontroller. It can provide 23 frames per second on average and is backward-compatible with the earlier interfacing standards (namely the AGE protocol and the Menelli box protocol). It has an AGE emulation mode and a Menelli emulation mode so that it appears as a PGSA for host software using the AGE protocol and as a Menelli box for host software using the Menelli protocol. It can be supplied by any source with a voltage ranging between 9 and 20 volts and current between .3 and 1 ampere (AC or DC as the unit has its own rectification circuitry). The frame format sent by the interface to the host computer will assume one of the above formats depending on the emulation and transmission modes used. The unit accepts a total of 35 commands including the AGE control commands and the Menelli commands and some commands that are specific to PGSI. Examples of the operations requested by the commands include switching the glove to the appropriate mode and turning various filters and A/D channels on or off.

At the heart of the PGSI is an MC68HC11E2FN Motorola microcontroller, an E-series member of the HC11 family of microcontrollers. The HC11 microcontroller used has 2048 (2k) of EEPROM memory and 256 bytes of RAM. The control program is stored in the EEPROM. The microcontroller has a total of 40 I/O pins that are used for communicating with the PowerGlove and the serial port of the host computer. The host computer sends special one byte commands to the PGSI and the firmware running on the microcontroller parses the commands, reads the values of the appropriate pins at the appropriate times and sends back a frame containing the information available from the glove. The communication speed is at 9600 bps with no parity and 1 stop bit [34]. Figure 3.2 shows a picture of the PGSI.

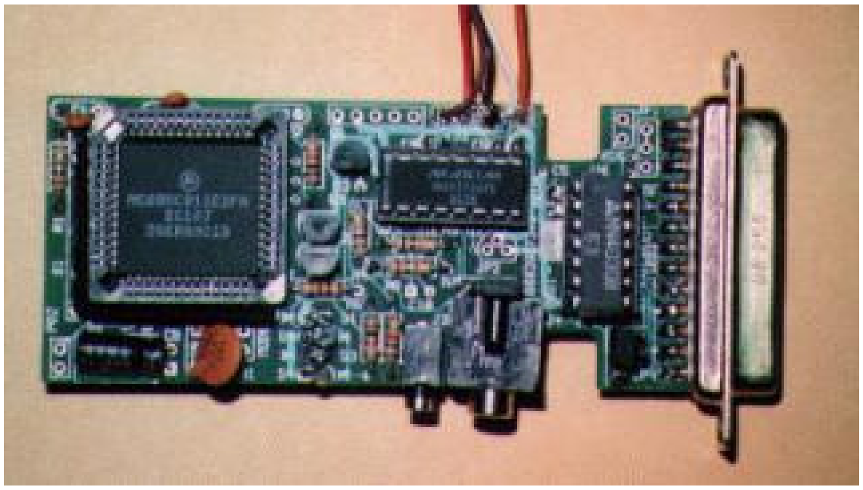

Figure 3.2: The PGSI.

(Courtesy of Joel Jordan of the ACM at UIUC)

3.3 The CyberGlove

The CyberGlove, manufactured by Immersion Technologies, is a more sophisticated alternative to the PowerGlove. It is designed from the outset as a general-purpose instrumented glove and has an accompanying comprehensive suite of interfacing software. The PowerGlove, in contrast, doesn’t have a standard set of application programming interface functions which could hinder rapid development. The CyberGlove is equipped with high accuracy, high repeatability 22 sensors for measuring the flex and abduction of fingers. Unlike, the PowerGlove, the basic system doesn’t provide motion tracking. However, it has provisions for connecting highly accurate tracking systems such as those provided by Polhemus and Ascension. Compared to the PowerGlove, the CyberGlove has more sensors and its sensors are more accurate and less prone to hysteresis and noise. The tracking systems used with the CyberGlove are more accurate and aren’t susceptible to ultrasonic noise and reflections as is the case with the PowerGlove. The CyberGlove can readily interface with a standard serial port (RS-232) and can send measurement information at a rate of 150 frames per second. This is in contrast with the PowerGlove which needs an external interfacing device (the PGSI for example) and can only provide measurements at a rate of 23 frames per second. Compared to the PowerGlove’s plastic body, the lightweight fabric body of the CyberGlove makes it more convenient to use and allows for more movement flexibility. The major disadvantage of the CyberGlove is its high cost, which is around $10,000 [12].

CHAPTER 4

THE SUPPORT VECTOR MACHINE

4.1 Introduction to the Support Vector Machine

4.1.1 Supervised Learning Model

Support vector learning belongs to the broad category of supervised learning methods, that is, learning by example. Radial basis functions and the multi-layer perceptron are examples of supervised learning as well. The general learning model of the supervised learning theory comprises three conceptual components:

1. A generator of random vectors x, drawn independently from a fixed but unknown distribution $P(x)$. A random vector x corresponds in this thesis to the vector of features extracted from the data frames generated by the glove for a certain sign.

2. A supervisor that returns an output vector y for every input vector x, according to a conditional distribution function P(y|x), that is fixed but unknown. The output y corresponds, in the context of this thesis, to the word represented by the sign performed.

3. A learning machine capable of implementing a set of functions $f(x,\alpha)$, called the hypothesis functions where x is a vector of sign features while α is an adjustable

parameter of the learning machine. α could be, in general, a scalar variable or a vector. The objective of a learning algorithm (training algorithm), is to choose the value of α that best approximates the response of the supervisor. α corresponds for example, to the vector of weights in an MLP problem (Multilayer Perceptron Neural Network) [35]. The relationship between the components of the learning model is illustrated in figure 4.1.

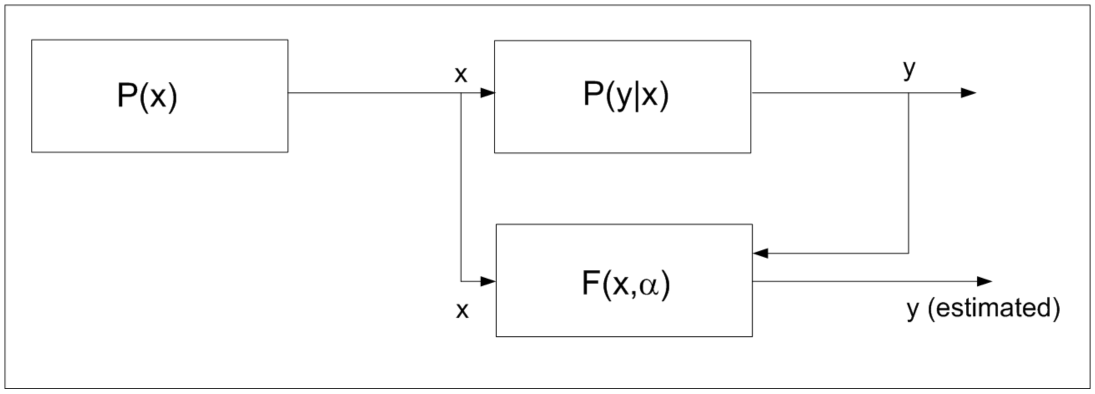

Figure 4.1: The supervised learning model

Training a learning machine is the process of finding a particular value of α. The design objective of a learning algorithm is to find the best value of α, in the sense that it leads to a learning machine that gives the best approximation of the underlying process.

Two points are worth noting here regarding the above abstract learning model:

1- The presented generic model for supervised learning encompasses many specific supervised learning problems. Three main ones are the pattern recognition problem, the regression estimation problem and the density estimation problem. The learning problem that is relevant to this thesis in particular is the pattern recognition problem

2- Supervised learning has an optimization problem at its core and hence optimization theory and numerical algorithms are essential to machine learning [35].

4.1.2 Overview of the Support Vector Machine Algorithm

The support vector machine learning (SVM) methodology was first introduced in 1992 at the Computational Learning Theory Conference (COLT) by Vapnik and his co-workers at Bell Laboratories. Since then, extensive research has been carried out in its theoretical analysis, implementation and applications [36].

The support vector machine has many attractive features some of which are unique compared to other competing methods such as multi-layer perceptron neural networks and radial basis function neural networks. Those advantages collectively constitute the main reason for selecting the SVM as the recognition algorithm used in this thesis. The advantageous features of the support vector machine learning include:

1. The optimization problem associated with the SVM machine has a unique solution. Other machine learning paradigms such as neural networks might have many local minima, which means that training might get stuck at one of them unable to reach the global minimum [35].

2. Training an SVM is relatively fast. Other learning paradigms such as the multi-layer perceptron can have relatively slow training due to the high dimensionality of the weight space [35].

3. The SVM algorithm has a simple geometric interpretation, which facilitates implementing and enhancing the algorithm [37].

4. Statistical learning theory a shows that the SVM either matches or significantly outperforms competing methods in generalization performance [37]. This is confirmed by empirical experiments.

5. The SVM solution is sparse, which facilitates the development of efficient numerical solutions for it [38].

6. The SVM lends itself to parallelization [37].

7. The SVM overcomes the curse of dimensionality (the exponential increase in complexity of a learning machine with the dimension of input space) [37].

The key idea used for the development of the SVM algorithm is the establishment of an optimal separating hyperplane between points belonging to two different classes. The hyperplane is described as optimal in this context, if it has the largest margin (the largest distance to the closest data point). The formulation of the SVM is presented in the following subsection beginning from the simplest case, which is finding the optimal hyperplane separating two linearly separable classes. The formulation is then generalized to handle less restrictive cases, were the classes are not linearly separable and the separation isn't perfect (i.e. some controllable error is allowed). Figure 4.2 illustrates the concept of optimal hyperplane separating two linearly separable classes.

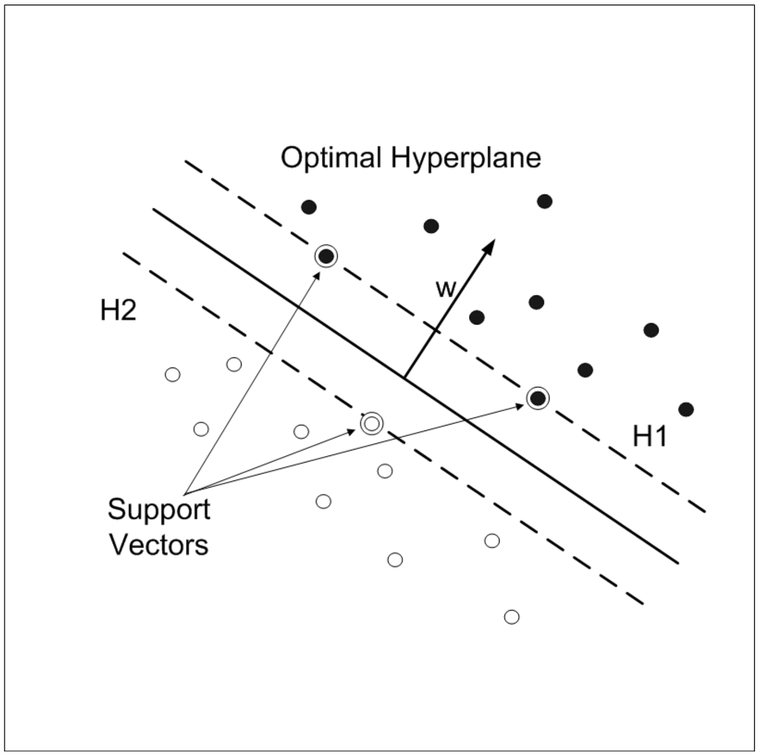

Figure 4.2: The optimal hyperplane.

4.2 Support Vector Machine Formulation

4.2.1 Classification Support Vector Machine for Linearly Separable Data

The basic idea behind the support vector machine algorithm is based, in its simplest form, on constructing a hyperplane that separates two linearly separable classes with the maximum margin. Assume that there is a set of vectors x_i each having a label y_i and that the separating hyperplane is $w.x + b = 0$ where w is a weight vector. Call the shortest distance between the separating hyperplane and the closest positive example d_+ and the shortest distance between the separating hyperplane and the closest negative example d_-. The margin is defined as the shortest of d_+ and d_-. Assume, also, that the separating hyperplane is scaled such that data points satisfy the following constraints:

$$x_i.w + b \geq +1 \text{ for } y_i = +1 \qquad (4\text{-}1)$$

$$x_i.w + b \leq -1 \text{ for } y_i = -1 \qquad (4\text{-}2)$$

Combining the two equations:

$$y_i(w.x_i + b) \geq 1 \text{ for } i = 1,2,...,l \qquad (4\text{-}3)$$

The above two constraints define two parallel hyperplanes with no data points between them. These hyperplanes are called the canonical hyperplanes. The optimal separating hyperplane is parallel to the canonical hyperplanes and has an equal distance from each of them. Normalizing the two planes yields:

$$\frac{w}{\|w\|}x + b = \frac{+1}{\|w\|} \qquad (4\text{-}4)$$

$$\frac{w}{\|w\|}x + b = \frac{-1}{\|w\|} \qquad (4\text{-}5)$$

The geometric distance between the two planes is then:

$$\frac{2}{\|w\|} \qquad (4\text{-}6)$$

and the margin is half of that:

$$\frac{1}{\|w\|} \qquad (4\text{-}7)$$

The optimum value for the weight vector is the values which maximizes the margin. This value can be obtained by minimizing

$$\frac{1}{2}\|w\|^2 \quad (4\text{-}8)$$

The cost function ($\frac{1}{2}\|w\|^2$) and the constraints ($y_i(w.x+b) \geq 1$) are used to construct the Lagrangian function L_P :

$$L_P \equiv \frac{1}{2}\|w\|^2 - \sum_{i=1}^{l} \alpha_i y_i (x_i.w + b) + \sum_{i=1}^{l} \alpha_i \qquad (4\text{-}9)$$

(where $\alpha_i \;\; i = 1,2,\ldots,l$ are Lagrange Multipliers)

Minimizing L_P is a convex quadratic programming problem. The dual of this problem can be obtained by using the stationarity conditions:

$$\frac{\partial L_p}{w} = w - \sum_{i=1}^{l} y_i \alpha_i x_i = 0 \qquad (4\text{-}10)$$

$$\frac{\partial L_p}{b} = \sum_{i=1}^{l} y_i \alpha_i = 0 \qquad (4\text{-}11)$$

Substituting in L_P, we get:

$$L_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j}^{l} y_i y_j \alpha_i \alpha_j x_i . x_j \qquad (4\text{-}12)$$

The Lagrange multipliers that maximize the dual Lagrangian L_D correspond to the weight vector that minimizes the primal Lagrangian L_P.Computing the weight vector from the Lagrange multipliers can be done through the first stationarity conditions. The training data points having non-zero Lagrange multipliers are the points laying on the canonical hyperplanes and are called *support vectors*. Training an SVM machine is essentially solving the associated optimization problem in order to identify the support vectors and compute the Lagrange multipliers associated with them. The decision function (hypothesis) that is based on the optimal hyperplane is:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i x_i . x + b \ (4\text{-}13).$$

or

$$f(x) = \sum_{i=1}^{N} \alpha_i s_i . x + b \qquad (4\text{-}14)$$

(where N is the number of support vectors and s_i with $i = 1,2,\ldots,N$ are the support vectors).

Examples that make the value of the decision function (the hypothesis) negative are put in one class and those making the value of the function positive are put in the other class [37][38].

The trained SVM machine classifies an input pattern (an input vector) by:

- Performing the dot product operation between the input pattern and each support vector.
- Multiplying the result of the dot product by the Lagrange multiplier associated with the support vector with which the dot product was performed.
- Adding the results of the dot products multiplied by the Lagrangian multipliers.
- Finding the sign of the summation.

This is depicted by figure 4.3

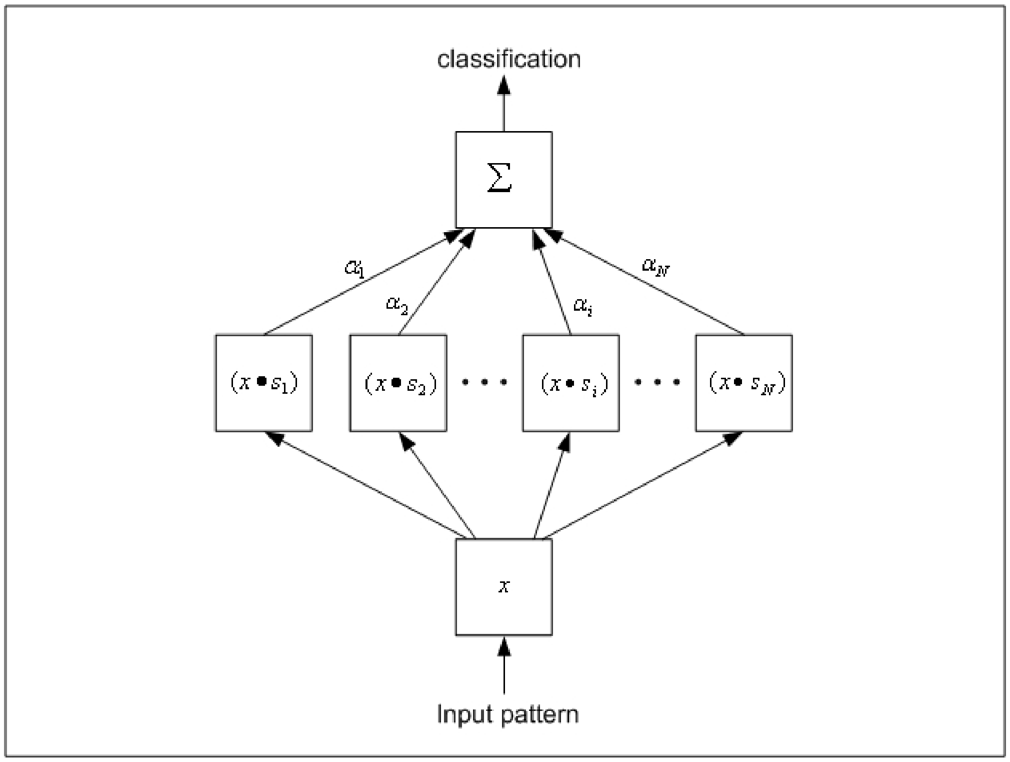

Figure 4.3: The structure of the basic support vector machine

Notice the following about the basic SVM:

- A trained SVM machine is completely specified by identifying the support vectors and the Lagrange multipliers associated with them.
- The basic SVM can only be used to separate linearly separable data.
- The basic SVM is a brittle recognition machine that can't tolerate errors.

The next sections will present extensions to the basic SVM formulation that will expand its applicability to a larger sphere of real-world problems.

4.2.2 Soft Margin Support Vector Machine

The above formulations lead to a brittle estimator in the sense that no training errors are allowed. This can be a problem with real-world systems because of the existence of noise in the data and the existence of classification problems that are not linearly separable. The following introduces some slackness in the classification support vector machine developed above so that some error is tolerated. This is done by introducing slack variables that represent a violation of the margin constraints and adding them to the cost function so that they are optimized as part of the SVM optimization process. With this modification the optimization problem becomes:

Minimize $\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\zeta_i$ (4-15)

subject to $y_i((w.x_i)+b) \geq 1-\xi_i$ (4-16)

where ξ_i, $i = 1,2,\ldots,l$ are non-negative slack variables and C is a constant coefficient whose best value is determined in practice by experimenting (a regularization parameter). Obtaining the dual of the problem can be performed in a manner similar to what was done for the basic SVM formulation [37][38].

4.2.3 Kernel Functions

The above formulation can be extended for the use in classifying non-linearly separable data by projecting data into a high dimensional feature space. A support vector machine for separating non-linearly separable data can be built by first using a non-linear mapping that transforms data from input space to feature space and then using a linear machine to separate classes in feature space. A function that represents the dot product in the feature space is called a kernel function $K(x, y) = \phi(x).\phi(y)$ and since data points appear in the dual formulation exclusively as dot products, mapping to feature space can be achieved by simply replacing the dot products with kernel functions. A formal definition of the kernel function can be stated as: A kernel function K is a mapping such that for all x, y belonging to an n-dimensional input space Z, $K(x, y) = \phi(x).\phi(y)$ where ϕ is mapping between the input space Z and a feature space F ($\phi : Z \rightarrow F$). By replacing the dot product with the kernel function), the dual Lagrangian and the decision function (the hypothesis) become, respectively:

$$L_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i . x_j) \qquad (4\text{-}17)$$

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x_i . x) + b \qquad (4\text{-}18)$$

Figure 4.4 shows the structure of the SVM with the introduction of the kernel function

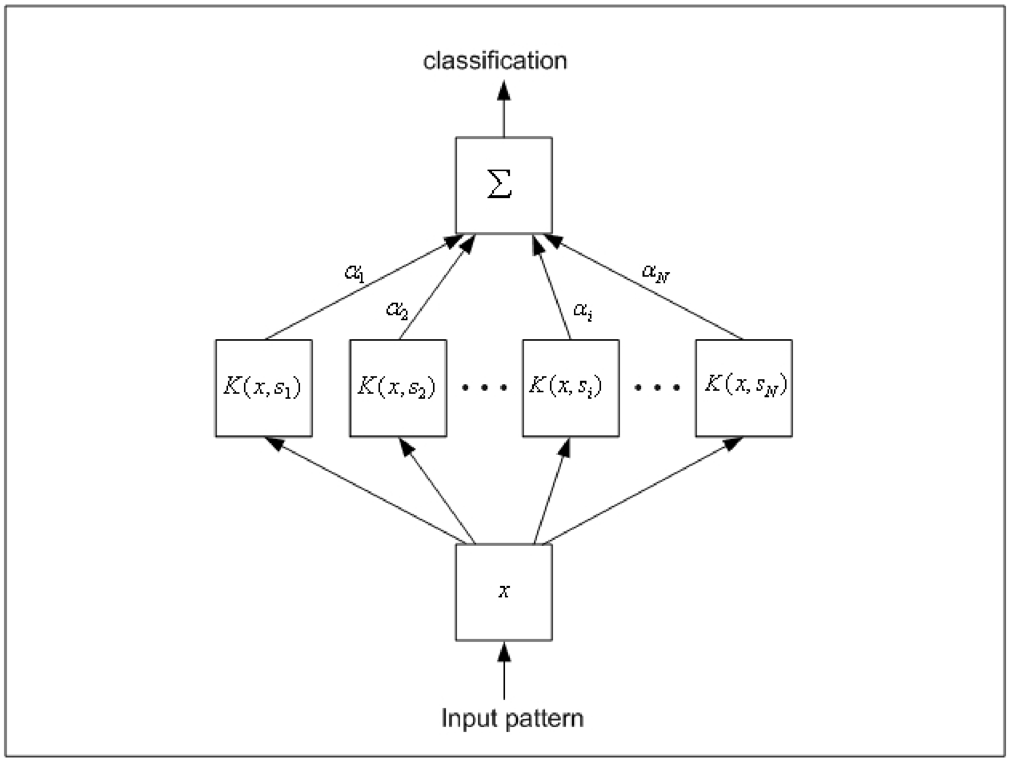

Figure 4.4: The structure of the support vector machine with kernel functions.

A number of different kernel functions can be found in the literature. Examples of kernel functions are:

The polynomial kernel $K(x,x_i) = (sx.x_i + r)^p$

The sigmoidal kernel $K(x,x_i) = \tanh(sx.x_i + r)$

The radial basis function kernel $K(x,x_i) = e^{-\|x-x_i\|^2/2\sigma^2}$

p (polynomial degree), σ (standard deviation), s and r are parameters whose best values for a particular problem are to be determined empirically. Using the polynomial kernel produces a decision function that is a polynomial in data points:

$$f(x) = \sum_{i=1}^{N_s} \alpha_i y_i (sx_i.x + r)^p + b \qquad (4\text{-}19)$$

Using the sigmoidal kernel produces a decision function that is equivalent to a two layer sigmoidal neural network:

$$f(x) = \sum_{i=1}^{N_s} \alpha_i y_i \tanh(sx_i.x + r) + b \qquad (4\text{-}20)$$

Using the Gaussian Radial Basis Function kernel produces a decision function that is equivalent to an RBF network:

$$f(x) = \sum_{i=1}^{N_s} \alpha_i y_i e^{\|x_i - x\|^2/2\sigma^2} + b \qquad (4\text{-}21)$$

Notice that the weights of this SVM-created RBF network are the Lagrange multipliers and that the centers of the RBF are the support vectors. Since the SVM algorithm determines the support vectors, the Lagrange multipliers and the threshold b automatically, it has an advantage over traditional RBF networks. This automatic selection of parameters is optimal since it is based on maximizing the margin of separation. An RBF SVM is depicted in figure 4.5.

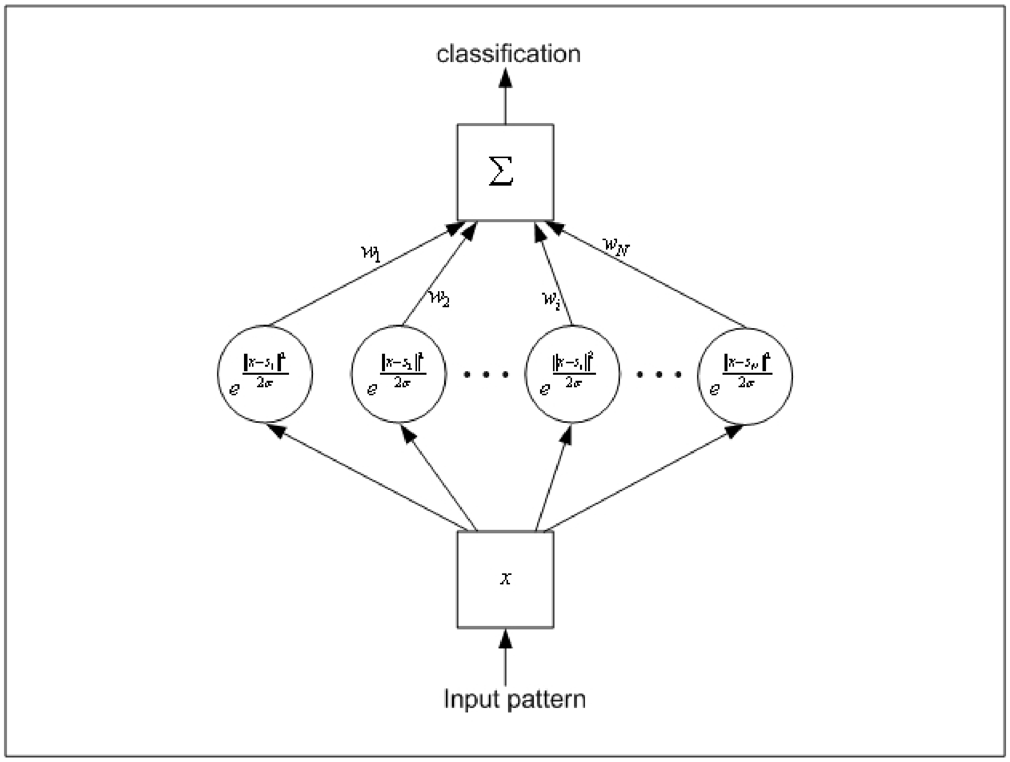
classification
Σ
w_1
w_2
w_i
w_N
$e^{\frac{\|x-s_1\|^2}{2\sigma}}$
$e^{\frac{\|x-s_2\|^2}{2\sigma}}$
$e^{\frac{\|x-s_i\|^2}{2\sigma}}$
$e^{\frac{\|x-s_N\|^2}{2\sigma}}$
x
Input pattern

Figure 4.5: The structure of the radial basis support vector machine

Different kernel functions with different parameters where explored in this research work [37][38][39].

4.2.4 Support Vector Machine for Multi-Class Classification

The above formulations handled cases where two classes exist. The two-class classification can be generalized to construct n-class classifiers by one of the following methods:

1. Build n classifiers, each capable of separating patterns belonging to one class from all other patterns.
2. Build the n-class classifier by feeding input to each of the two-class classifiers and choosing the class corresponding to the maximum $f_k(x), k = 1,2,...,n$.

The multi-class problem can be solved in a direct manner as well by generalizing the procedure used for the two-class case [40].

4.3 Numerical Solutions

Numerical methods for solving optimization problems have been developed in the last decades with substantial work in the 1950's and 1960's. Although those can be readily applied to the SVM optimization problem and has been actually applied, large scale SVM problems pose computational difficulties to the standard optimization methods in terms of computational speed and efficiency. Fortunately, the SVM optimization problem possesses a number of desirable features that can be leveraged to reduce time and space complexity (convergence time and main memory space).

Convexity, sparseness and the implicit mapping to feature space, are features that can be exploited in designing faster and more computationally efficient algorithms for SVM problems. This section discusses the numerical algorithms used for solving SVM problems and shows how the mathematical properties of the SVM problem were utilized to improve the algorithms used.

4.3.1 Standard Quadratic Programming

Early on during the development of the support vector machine, generic Quadratic Programming (QP) optimization engines were used at the core of the SVM solving packages. Several freeware and commercial optimization engines based on standard algorithms are available. Examples include Newton methods, the conjugate gradient method and the primal dual interior point method [38]. The advantage of using such approach is that leveraging existing packages makes developing an SVM package much faster. Although, those packages work well for moderately sized to small problems they don't scale well for large size problems. The reason is that they need to handle the kernel matrix which grows quadratically with the training set size. This means that, assuming that each element is stored in an eight byte double precision number, almost three gigabytes of RAM are needed for a training set of 20,000 points. Even with sufficient memory, convergence speed suffers adversely as problems of larger scale are approached (a matrix where element are formed by applying the kernel operation between every possible pair of training points) [41].

4.3.2 Chunking

In order to solve the scaling problem presented above, a number of methods have been devised. All of them are based on breaking the QP problem into smaller QP problems. Chucking was the first algorithm devised based on this approach. Chunking starts by selecting a partial set of the data and then limits training to the selected set. The support vectors found by training over the subset are used to form a hypothesis which is used to test all other data points. A number of points chosen according to a heuristic are added to the support vectors found so far and the whole working set is then used for training in the next iteration. The working set typically grows with each iteration until it eventually contains the support vectors only. Although chunking still relies on a generic optimization engine, the memory space requirements are reduced as the sub-problems solved at each iteration need less memory compared to the original problem [41].

4.3.3 Decomposition

Decomposition is another method for breaking up the SVM problem to smaller problems and is conceptually similar to chunking. The working set performs training on a fixed size set of points in each iteration. At any stage, adding a number of points to the working set must be accompanied by removing an equivalent number of points. The advantage of decomposition is that the fixed size of the working set can be chosen so that it fits in memory which enables the algorithm to handle problems with arbitrary size [41].

4.3.4 Sequential Minimal Optimization (SMO)

Sequential minimal optimization (SMO) is a variation of decomposition that has a working set size of only two points. Limiting the working set to two points has several performance advantages. First, the memory requirements are minimal and data sets of arbitrarily large sizes can be handled. Second, with only two variables considered in the optimization process at every step, an analytical solution can be hard-coded into the inner loop and invoking a QP optimization engine can be avoided completely. Avoiding the processor intensive QP inner loop significantly improves the convergence speed. In addition, the training becomes less susceptible to numerical precision problems.

The convergence speed of SMO is greater than the preceding methods by orders of magnitude. Two extra algorithm enhancements can be used to speed the convergence of the basic algorithm further. First, kernel evaluations can be cached. Second, selection of the two points that are to be jointly optimized could be based on heuristics rather than making it arbitrary. The selection heuristic gives higher priority for the points which when optimized, contribute more to the progress towards the solution [41]. Figure 4.6 provides a visual comparison between the different numerical approaches for training the SVM. It represents how many points of the original problem are retained after each algorithm iteration.

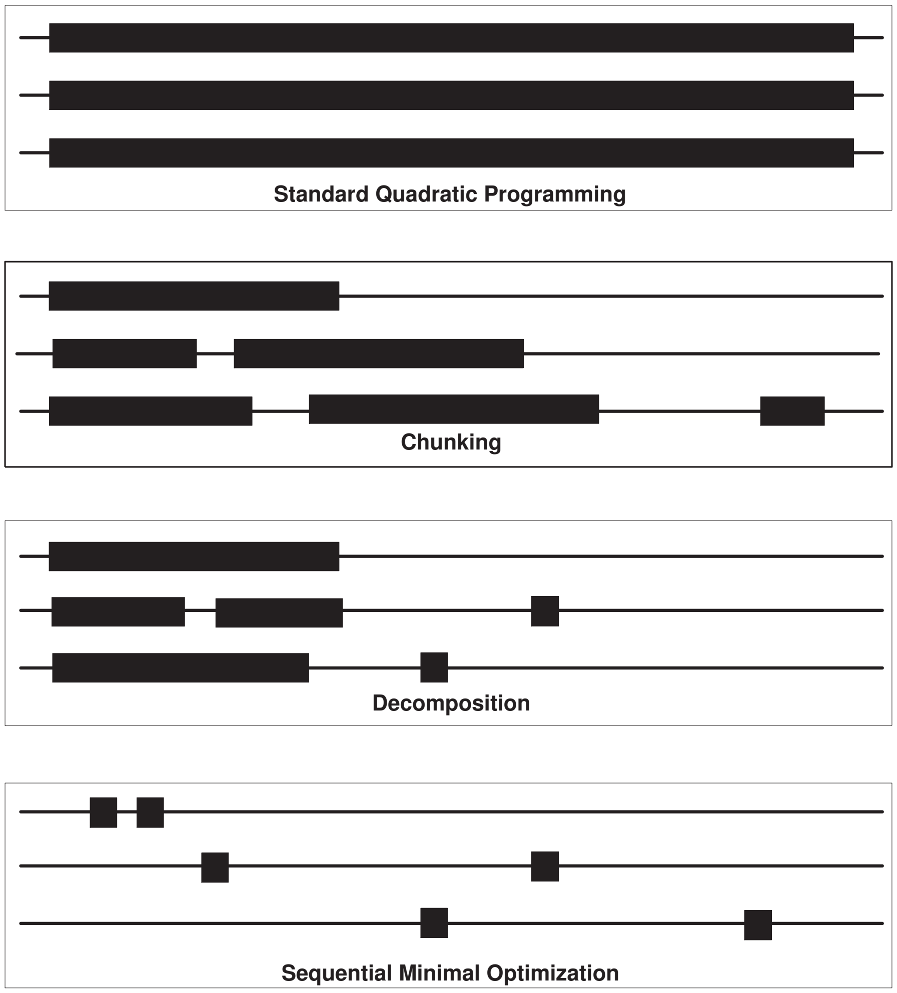
Standard Quadratic Programming
Chunking
Decomposition
Sequential Minimal Optimization

Figure 4.6: Numerical solutions for the SVM

4.4.5 SVMTorch

The SVM software package used for this work is the SVMTorch II [33]. SVMTorch II is a package developed in C++ by Ronan Collobert and Samy Bengio and is free for academic use. For classification, SVMTorch II implements an enhanced version of SMO based on the work of Keerthi and Gilbert [42].

CHAPTER 5

METHODOLOGY AND RESULTS

5.1 Feature Extraction

Features are extracted from the raw data to be used as inputs to the learning machine. Each feature vector is then given a numerical label to identify the sign it represents. Time division is considered a low-level feature extraction approach, yet it is an effective one. With time division, the time over which the sign is performed is divided into segments and the average of each primitive attribute (x,y,z, rotational and flex values) is computed over each segment. This average over the ith segment can be mathematically expressed as:

$$s_i = \sum_{j=l_i}^{u_i} \frac{x_j}{u_i - l_i + 1} \qquad \forall i, \quad 0 \le i \le d-1 \qquad (5\text{-}1)$$

where:

d: The number of time segments.

i: The segment index.

j: The frame index (starts from *j=1*) *n:* The total number of frames received during the signing time.

l_i: The index of the first frame in segment i. $l_i = \left\lfloor i\frac{n}{d} \right\rfloor + 1$

u_i: The index of the last frame in segment i. $u_i = \left\lfloor (i+1)\frac{n}{d} \right\rfloor$

Assume, for example, that during the signing time eight frames were received, then for a two-segment time division the time average for x over the first segment (s_0) is calculated as:

$l_0 = \left\lfloor 0\frac{8}{2} \right\rfloor + 1 = 1$ $u_0 = \left\lfloor (0+1)\frac{8}{2} \right\rfloor = 4$

$$s_0 = \sum_{j=1}^{4} \frac{x_j}{4-1+1} = \frac{1}{4}\sum_{j=1}^{4} x_j$$

Thus, a continuous stream of frames is generated by the glove as the hand performs the sign. Each frame bears an instantaneous measurement of each of the following eight raw measurements: the x position of the hand, the y position, the z position, the roll of the hand, the flex of the thumb, the flex of the index finger, the flex of the fore finger and the flex of the ring finger. The features used as inputs to the pattern recognition algorithm are time division averages of the raw measurements. Time division averages are computed by dividing the entire time during which the sign was performed into a number of equal segments and then finding the average of each of the eight raw attributes over each segment. Within a segment, the average value of a certain raw measurement is taken over all frames generated during that segment [27]. The length of the feature vector will be eight times the number of time segments. Different numbers of segments were explored in this research work. Figure 5.1 illustrates the feature extraction process.

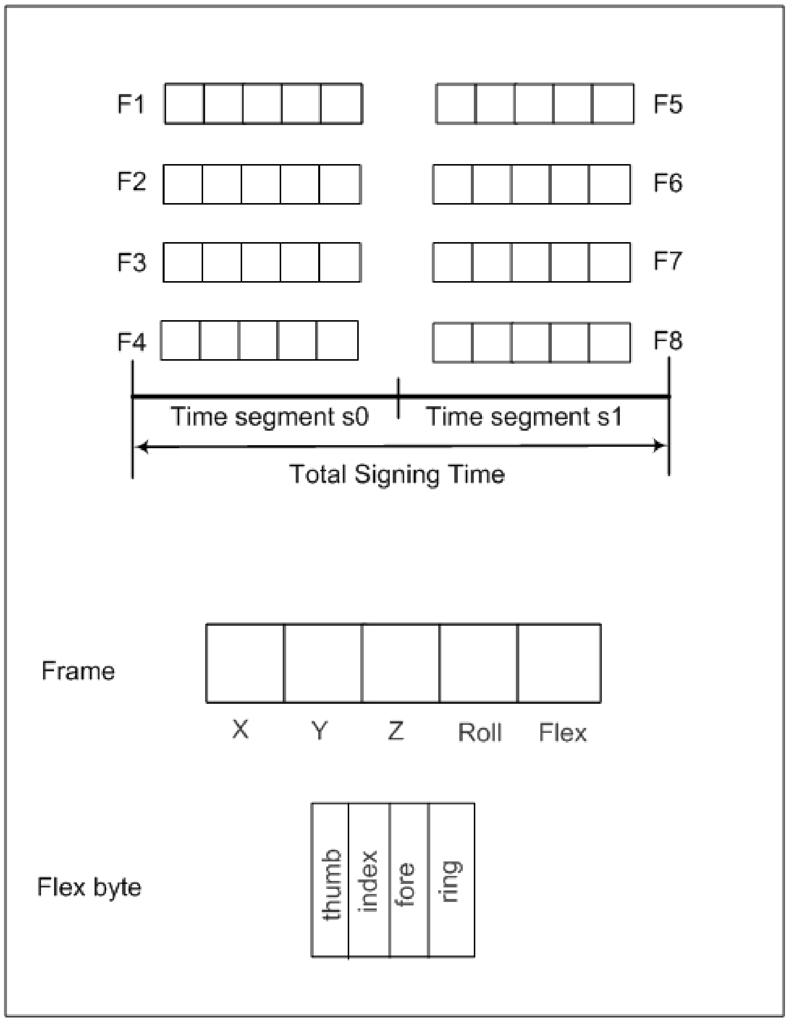

Figure 5.1: Extracting time division averaging features.

The fact that this approach for feature extraction reduces the data to fixed sized features that are based on averaging provides the learning machine with smoother data with less noise. This approach has also the advantage of being relatively invariant with respect to the total time taken by a signer to complete a sign. This is because sign language performers generally maintain the relative timing between the hand movements composing the sign, regardless of the signing speed. In that regard, sign language speakers are just like vocal speakers who generally keep the same relative speaking speed while pronouncing a word regardless of the speed with which they talk [27]. The feature vector extracted through time division segmentation can be seen as an approximate trajectory of the sign (A rough approximation of the path the hand takes while performing the sign). A three dimensional plot for this trajectory for the sign "Allah" (God) is shown in figure 5.2.

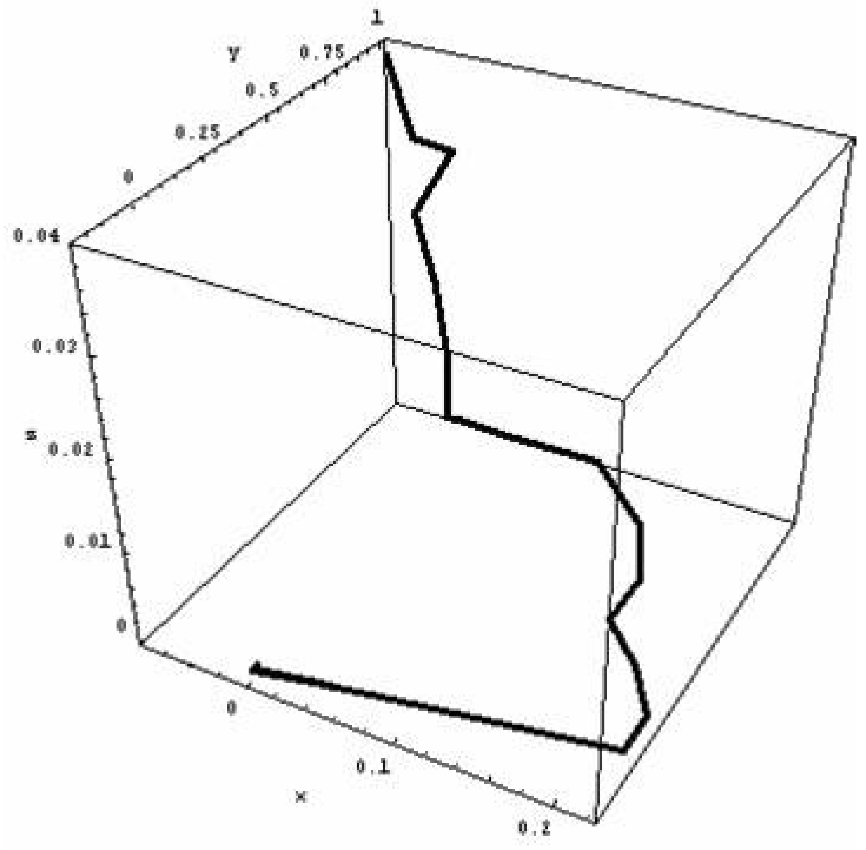

Figure 5.2: The trajectory of the sign "Allah" in space.

The sign for “Allah” is performed by raising the hand and pointing upward. Notice in the figure how the hand tends to move towards the right before it is raised. The following plots show each of the eight basic measurements plotted over all frames received from the glove during the performance of the sign “Allah”. Points representing the measured value at each frame are plotted for X,Y and Z, while roll and finger flexes are represented using a step graph. This is done as the roll and flex can only take one of few discrete values. Measurement points for X, Y and Z are connected to enable the visualization of an approximate trajectory of the hand with respect to the measured dimension. The units used don’t correspond to any standard units and while the usefulness of the measurements would be more with the usage of standard units, neither the original functionality of the PowerGlove as intended by the manufacturer nor its functionality as used in this research work require it as long as the measurements are consistent.

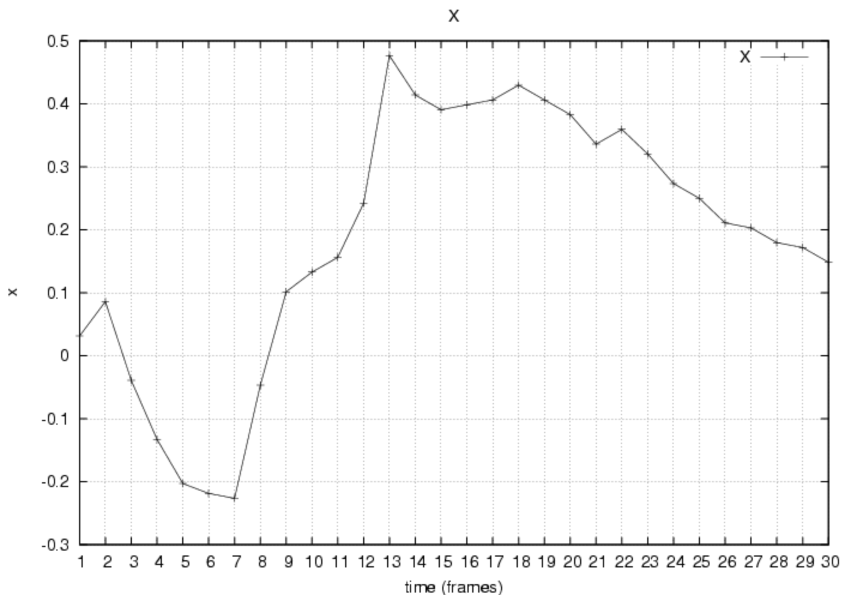

Figure 5.3: The x values for the sign “Allah”

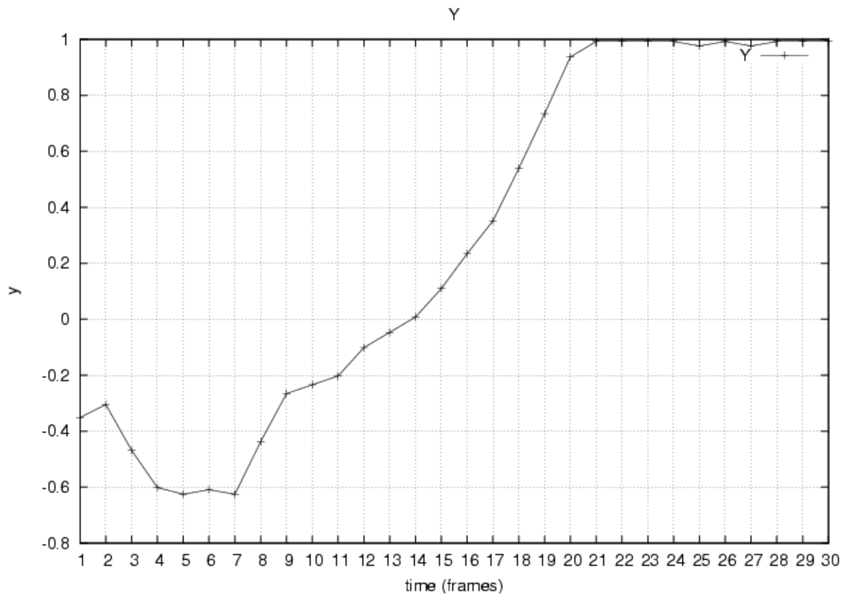

Figure 5.4: The y values for the sign “Allah”

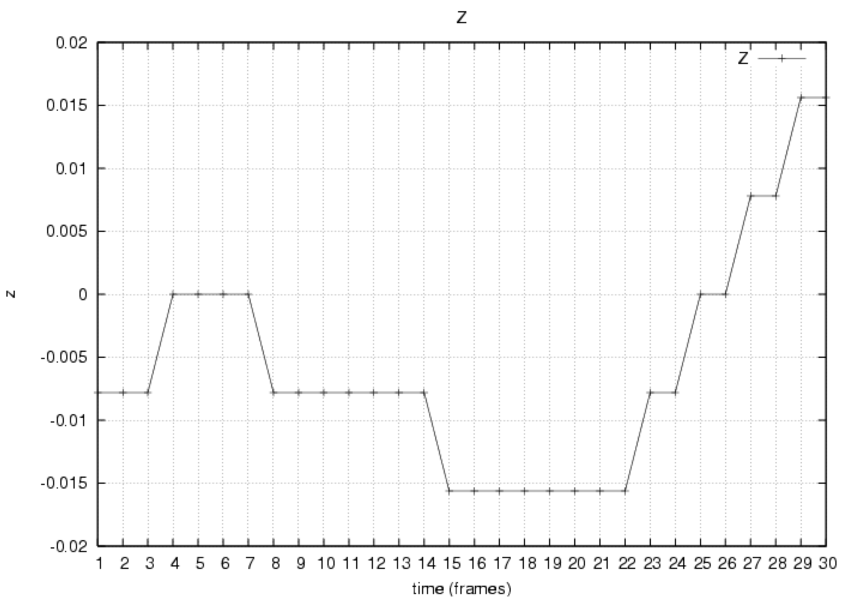

Figure 5.5: The z values for the sign “Allah”

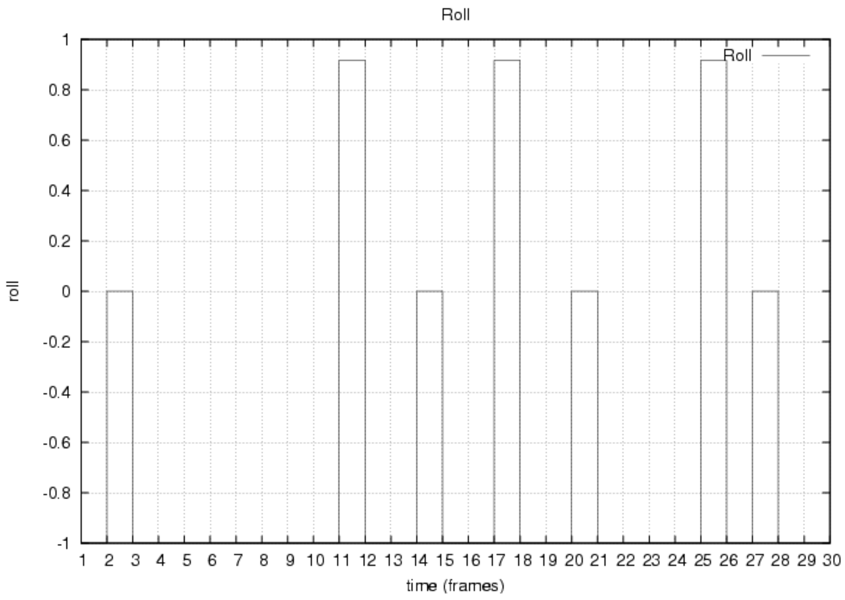

Figure 5.6: The roll values for the sign “Allah”

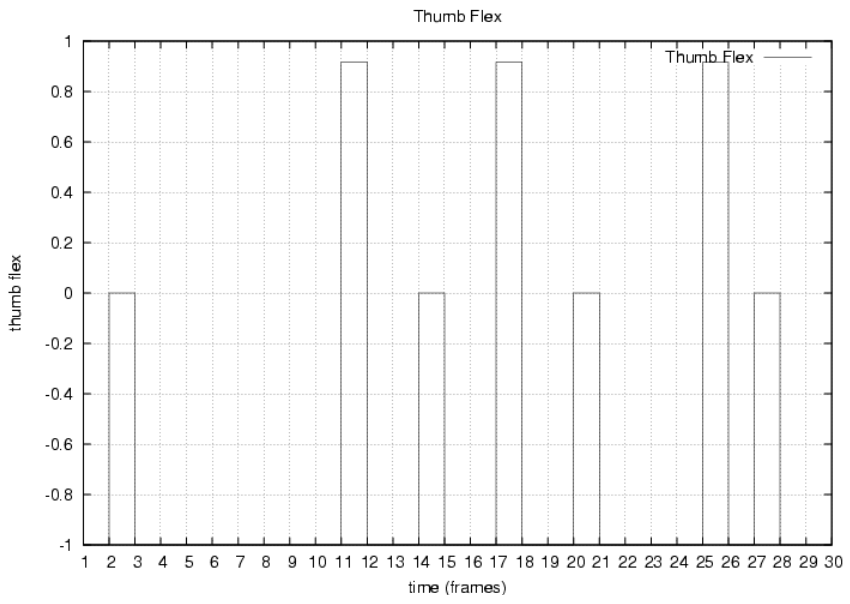

Figure 5.7: The thumb flex values for the sign "Allah"

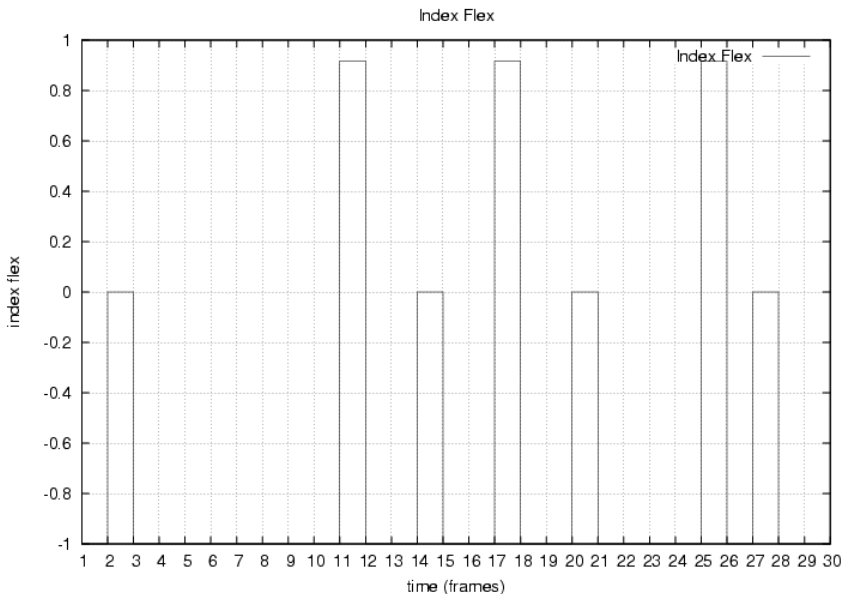

Figure 5.8: The index flex values for the sign "Allah"

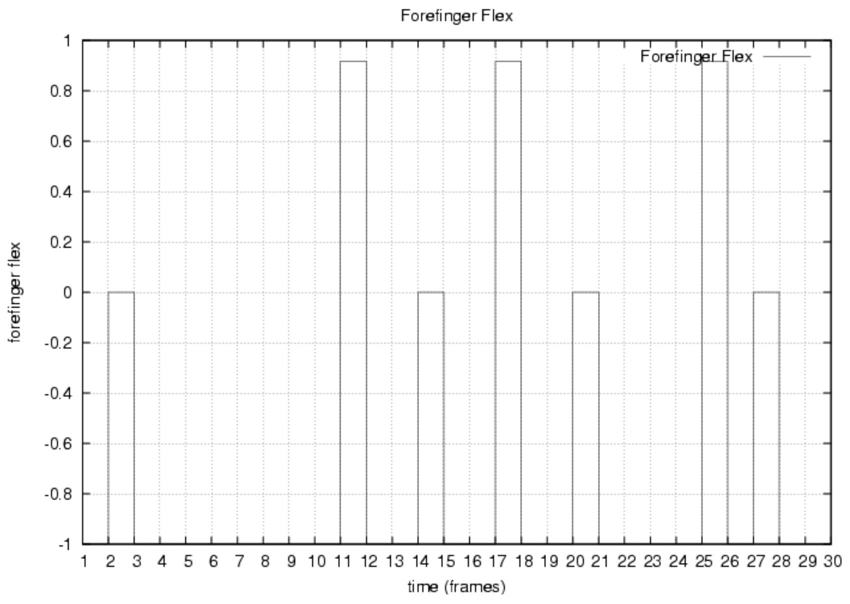

Figure 5.9: The forefinger flex values for the sign "Allah"

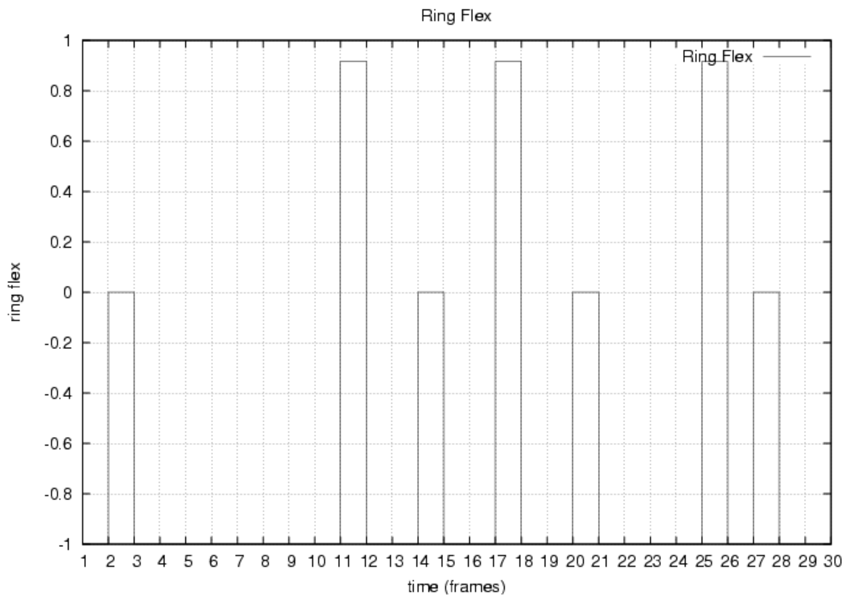

Figure 5.10: The ring flex values for the sign "Allah"

The following plots the time segmentation features extracted from the sign “Allah” with 10 time segments (80 features).

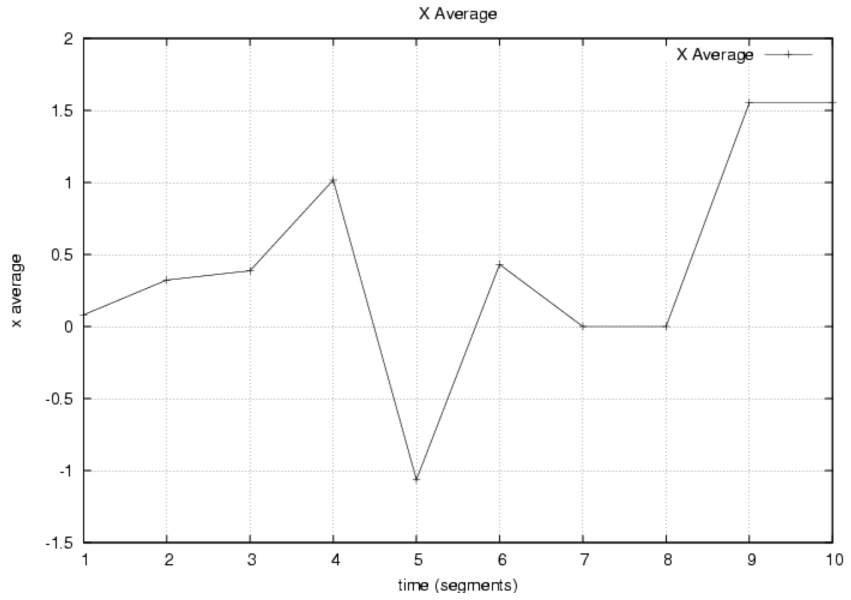

Fig 5.11: X time average for the sign "Allah"

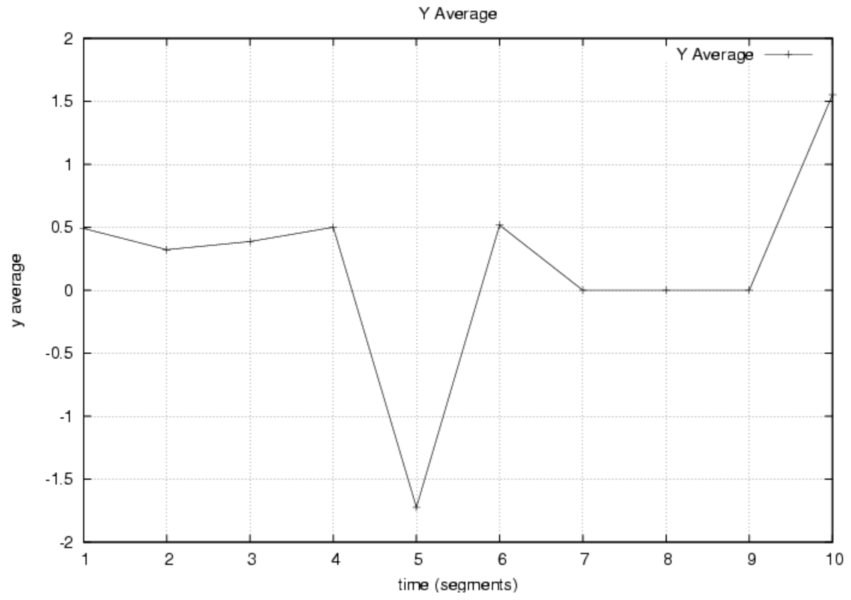

Fig 5.12: Y time average for the sign "Allah"

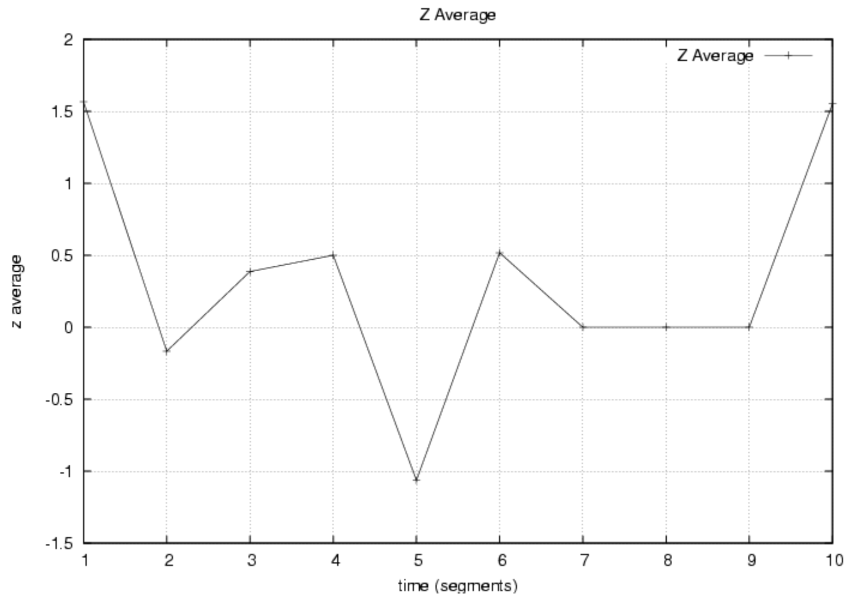

Fig 5.13: Z time average for the sign “Allah”

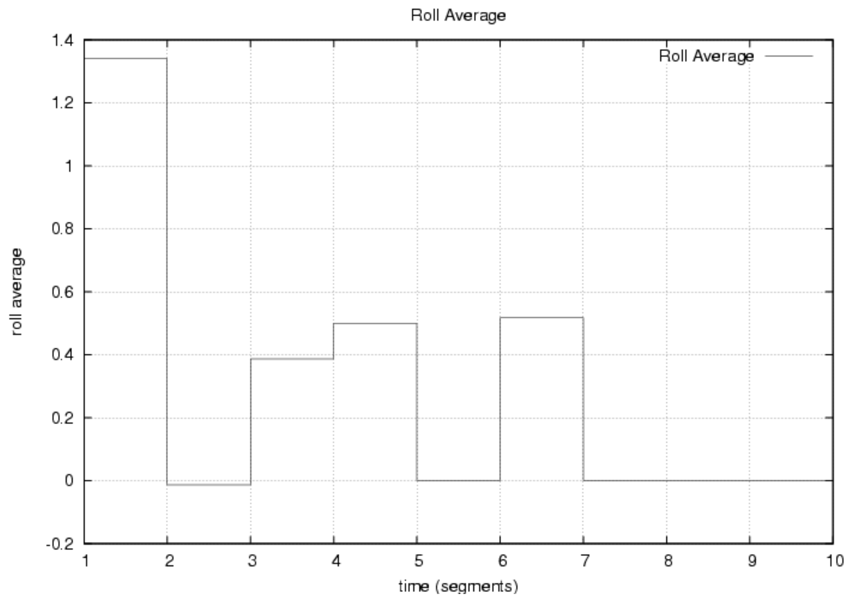

Fig 5.14: Roll time average for the sign “Allah”

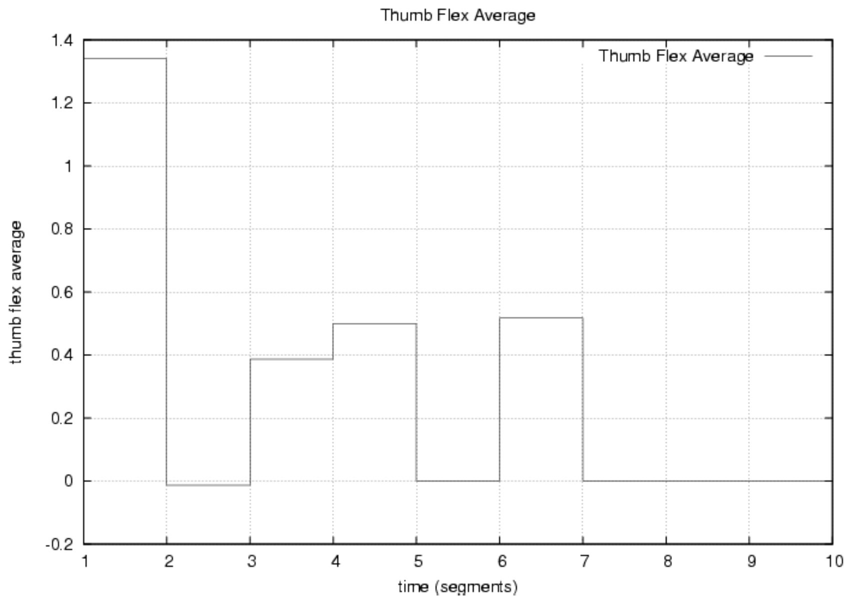

Fig 5.15: Thumb flex time average for the sign “Allah”

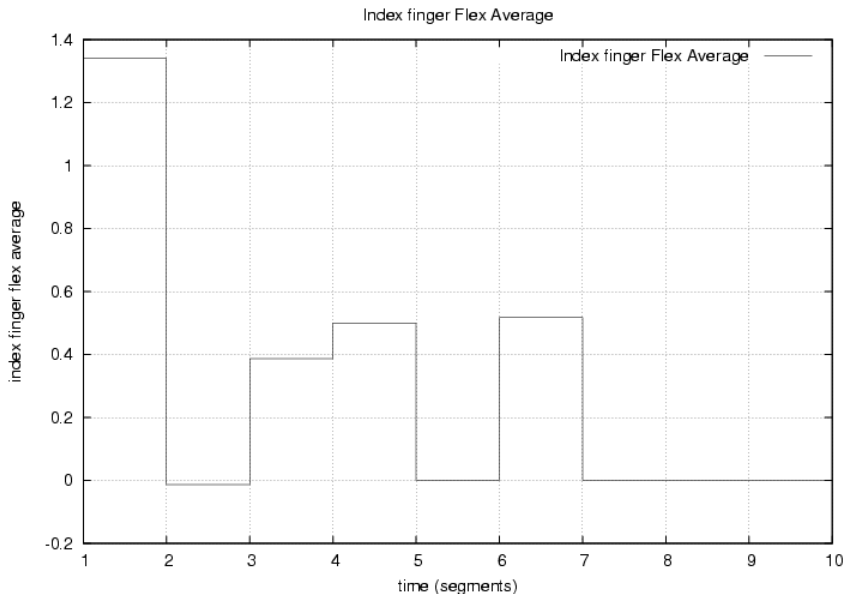

Fig 5.16: Index finger flex time average for the sign “Allah”

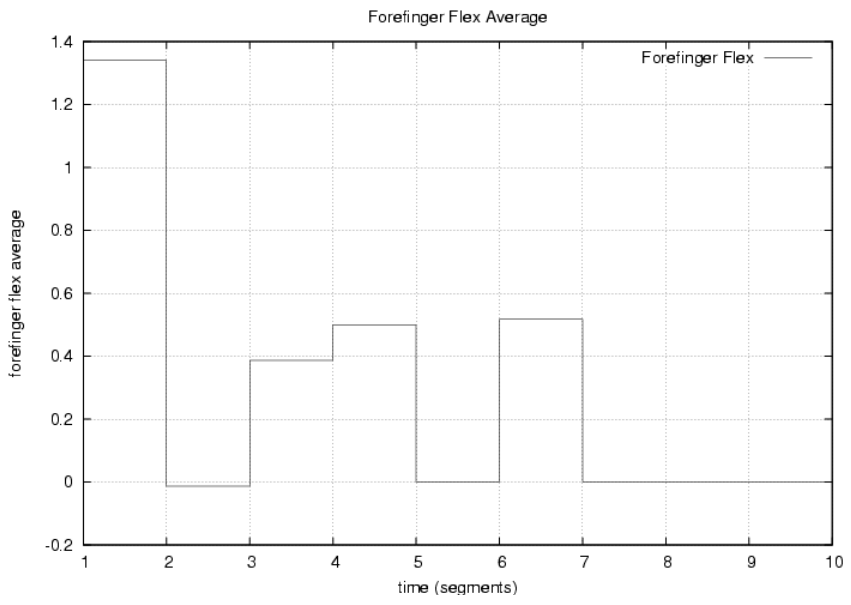

Fig 5.17: Forefinger flex time average for the sign “Allah”

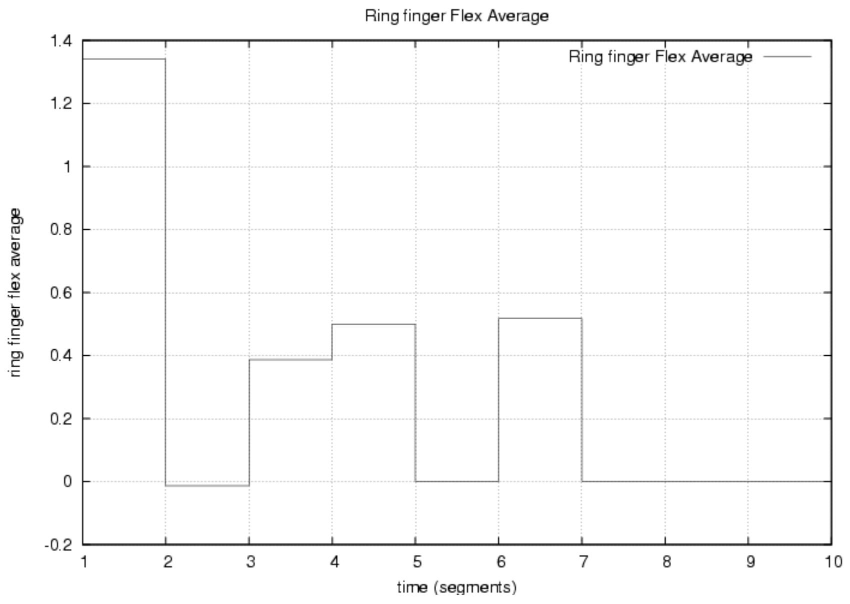

Fig 5.18: Ring finger flex time average for the sign “Allah”

5.2 Data Collection

Three individual adults from the deaf community volunteered to perform the signs to generate samples for the learning machine. The individuals were chosen among adults to insure their fluency in sign language and the accuracy of their signs. More than twenty samples were collected per sign for each person (a total of around 10,000 samples). The signer starts with his right hand resting on the table on which the screen is positioned and then presses the ``center button" on the glove to signal the start of the sign. As soon as the sign is completed, the ``A" button is pressed. This is repeated about 20 times for each sign. Raw measurements are continuously sent by the glove to the host machine at an average rate of 23 frames per second [33]. Frames assume the AGE format illustrated previously.

5.3 System Performance

The training data for each sign consisted of 36 samples, while testing data for each sign consisted of 18 samples. For each word, 12 samples were taken for training per person, while 6 samples per person were taken for testing. Some preliminary results has been reported in [43]. The highest accuracy was realized by using a radial basis function kernel with a time segmentation of ten (80 features). This was achieved when the standard deviation was 10 and C was 10. With those parameters, an accuracy of about 70% was achieved for a word set of 120 words, while accuracy above 90 % was achieved for a word set of 10 words. The parameter selection process is explained in the following paragraphs.

As illustrated earlier, the SVM has a number of adjustable parameters whose values must be determined through trails in order to find the combination of parameter values leading to the least generalization error (least error on the testing set of samples). The kernel to be used, the values of kernel-specific parameters not automatically computed by the SVM and the value of the regularization parameter C are all to be empirically through iterative trial. Aside form the SVM-related parameters, the length of the feature vector (the number of features extracted per sample) that leads to the least generalization error is also to be determined empirically. The strategy adopted for parameter selection is to start with running a number of simulations with different values to get a sense about the range within which reasonable values lie and then narrow down the range through several iterations of simulations to fine-tune the values of the parameters for better generalization. Initially, a set of ten words is considered with a time segmentation of ten. After fine-tuning the SVM parameters, larger word sets (up to a maximum of 120 words) and different time segmentations (number of features per sample) are considered. The following will explain how the SVM parameters were chosen using this strategy.

Several kernel functions were tried. Linear SVM and SVM with the sigmoidal kernel didn't produce any useful results. In both cases, the SVM mostly didn't converge. Better results were obtained by using polynomial kernel of the second degree. The best value of the s parameter was found to be .0002 (while fixing r at 10 and C at 10). Fixing the value of s at .0002 and trying different values for r over the range from 1 to 10 with an increment of 1 we find that the best value is 10. Since the value is at the end of the considered range, different values of r over the range of 10 to 100 with an increment of

10 were tried. In that range the best value was found to be 90. The range of values of 100 to 1000 with an increment of 100 was then tried to insure that the best value for r lies between 10 and 100. Accuracy was found to be less in that range, so the focus was narrowed down to the range of 85 to 95. It was found that the best value for r was 89. With $s = .0002, r = 89$, different values of C were tried. The best value of C was found to be 10. With $s = .0002, r = 89, C = 10,$ the system achieved accuracy just below 90% for a set of 10 words. Figures 5.19 to 5.24 show recognition accuracy using the polynomial kernel with different parameters.

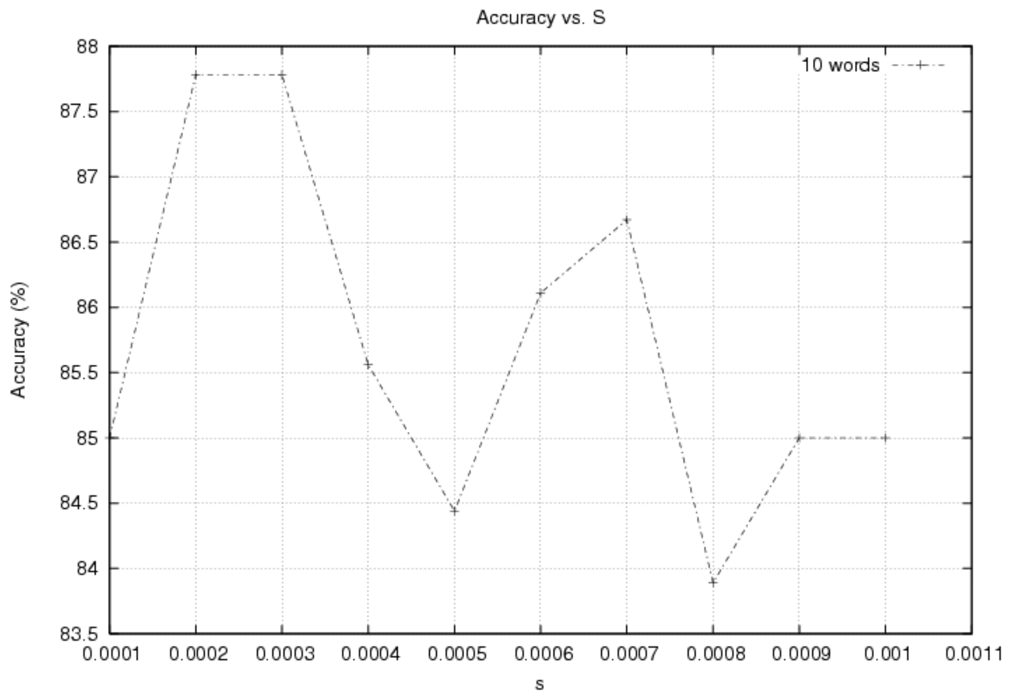

Figure 5.19: Accuracy for different S values for the polynomial kernel (.0001-.0011)

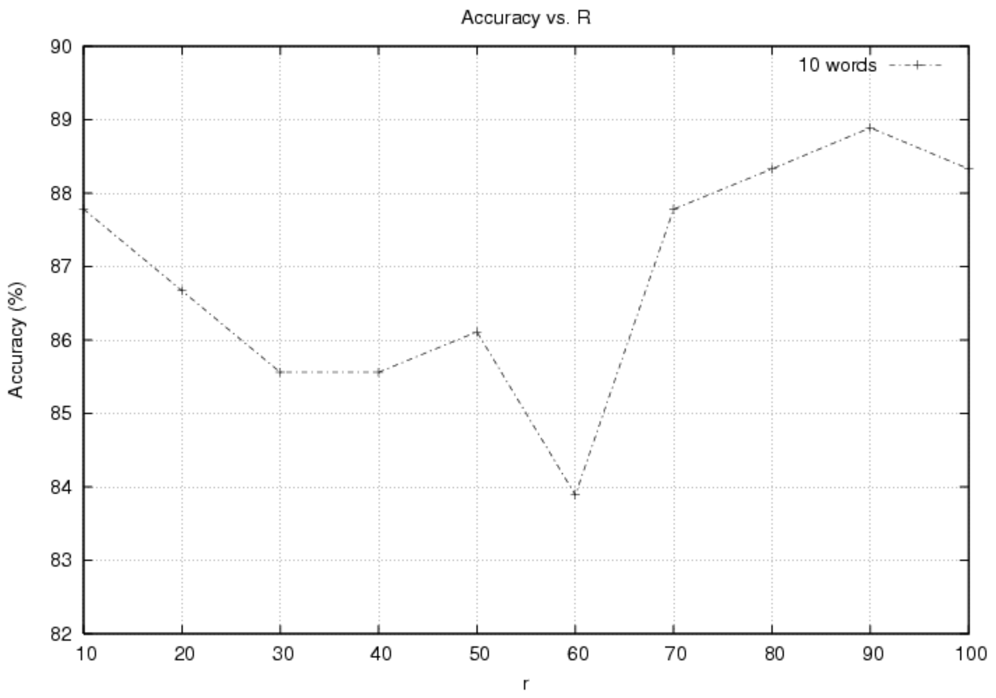

Figure 5.20: Accuracy for different R values for the polynomial kernel (10-100)

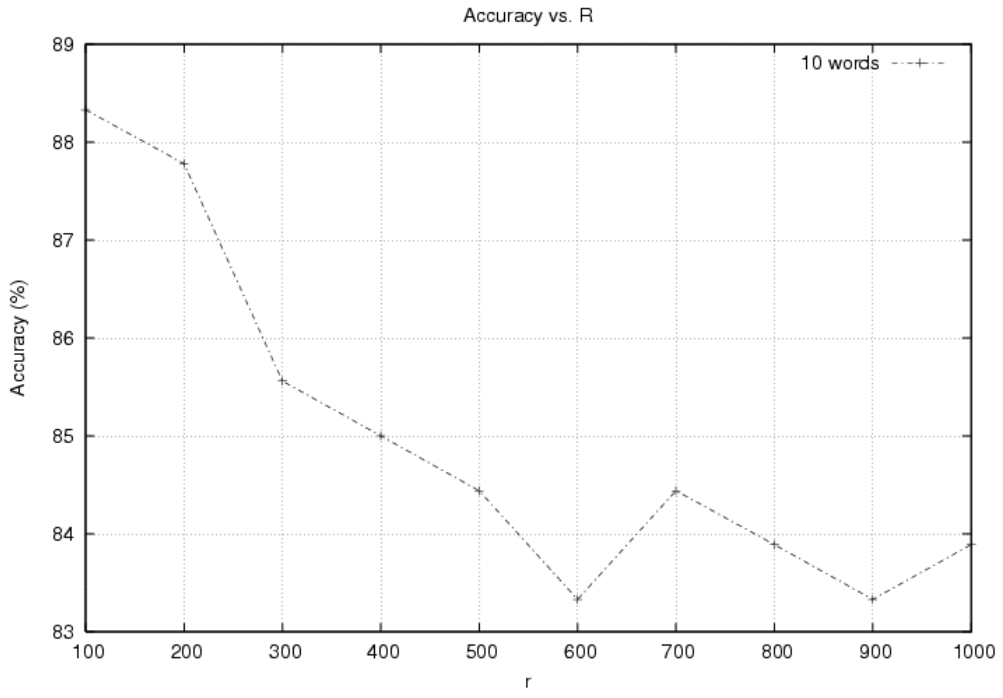

Figure 5.21: Accuracy for different R values for the polynomial kernel (100-1000).

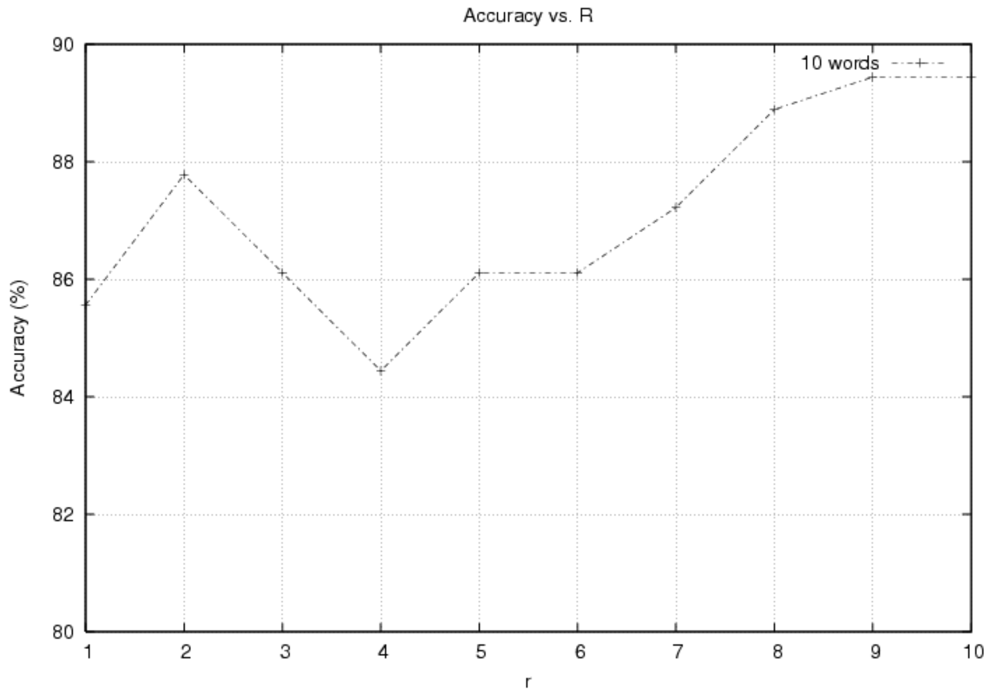

Figure 5.22: Accuracy for different R values for the polynomial kernel (1-10)

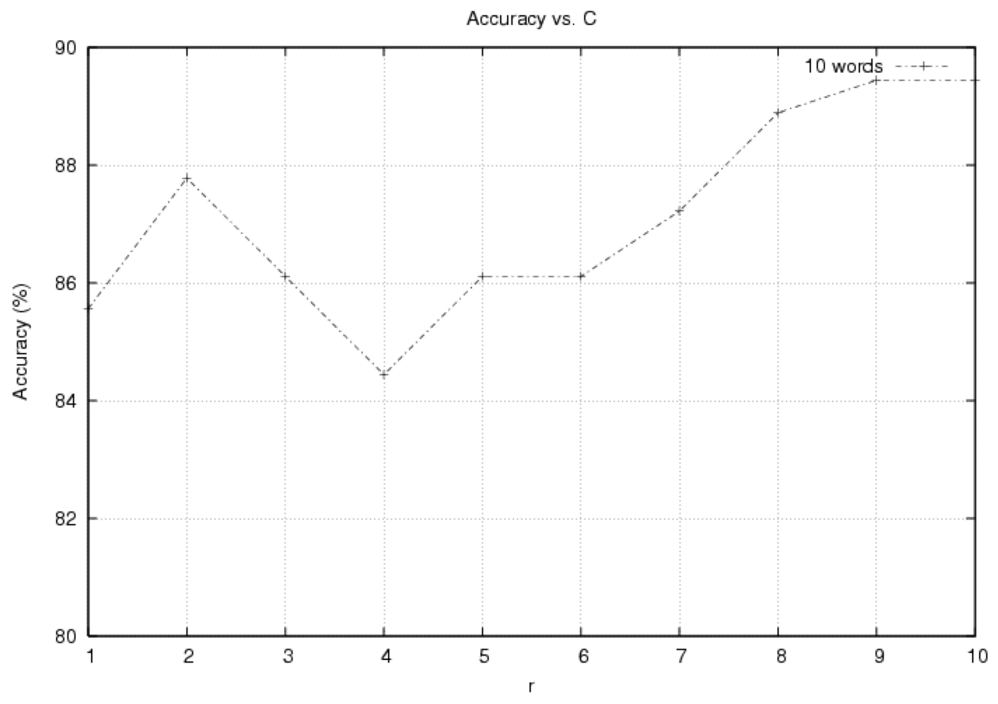

Figure 5.23: Accuracy for different C values for the polynomial kernel (1-10)

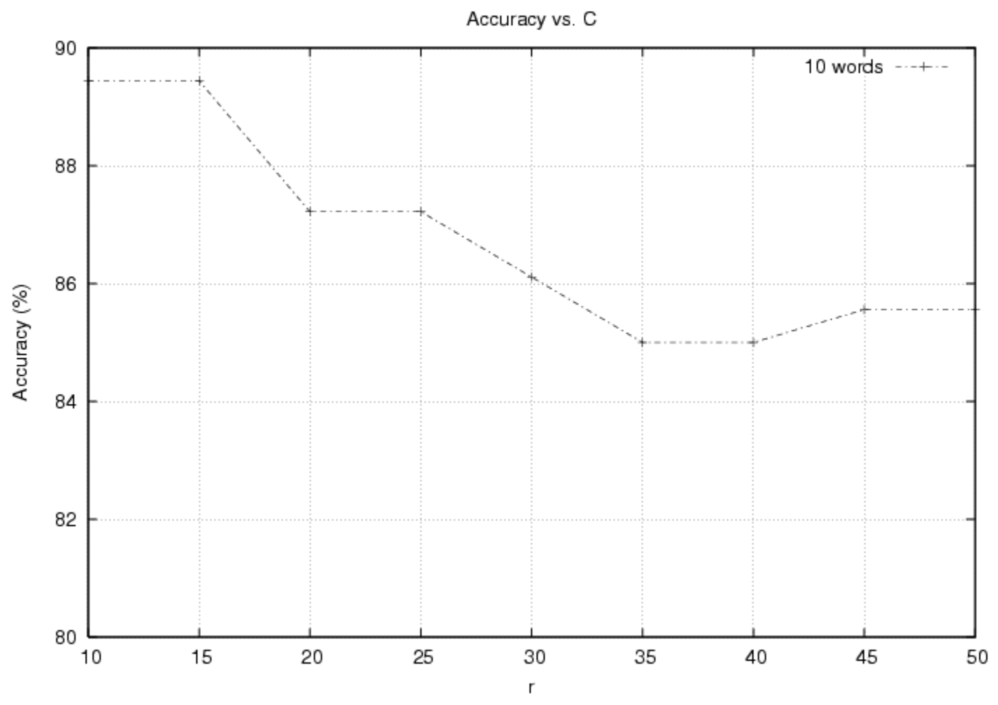

Figure 5.24: Accuracy for different C values for the polynomial kernel (10-50)

An accuracy of above 90 % was achieved using a radial basis function kernel. The standard deviation is the only adjustable kernel-specific parameter of the SVM constructed using RBF kernels as weights and function centers are automatically determined by the SVM algorithm. As mentioned earlier 10 was found to be the best value for both the standard deviation and for C. Since the RBF kernel was found to outperform other kernels, accuracy was computed for different time segmentations (number of features) and for different word sizes. As expected, the accuracy decreased with the number of words considered. Going beyond 10 segments (more than 80 features) doesn't yield better system performance. System performance for an SVM with an RBF kernel for a variety of parameters is illustrated in figures 5.25 to 5.31.

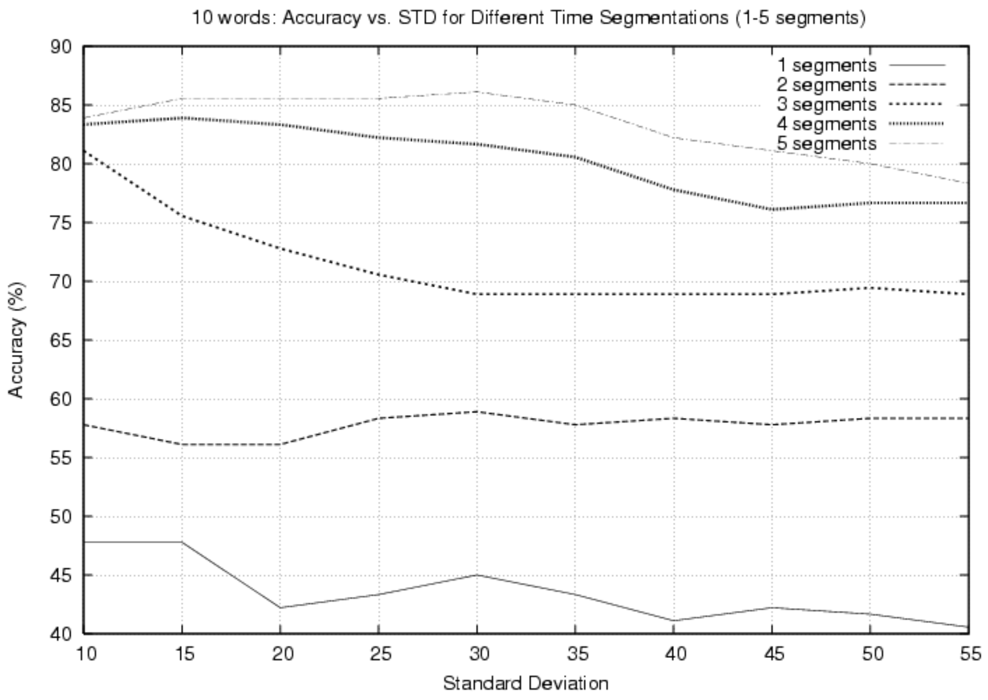

Figure 5.25: Accuracy for different time segmentations for the RBF kernel (1-5) for ten words

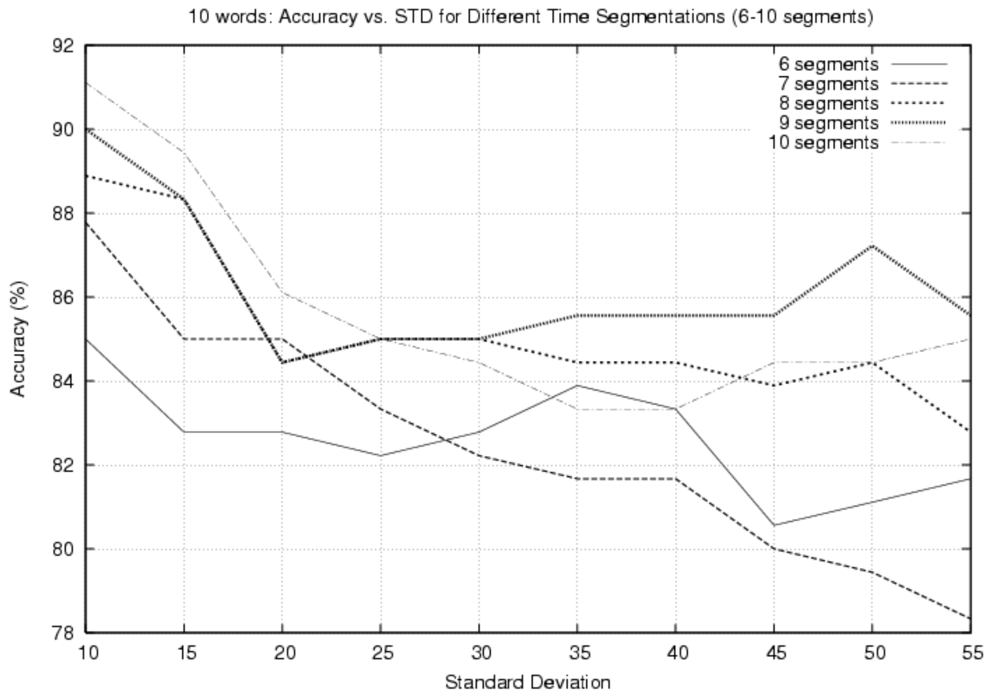

Figure 5.26: Accuracy for different time segmentations for the RBF kernel (6-10) for ten words

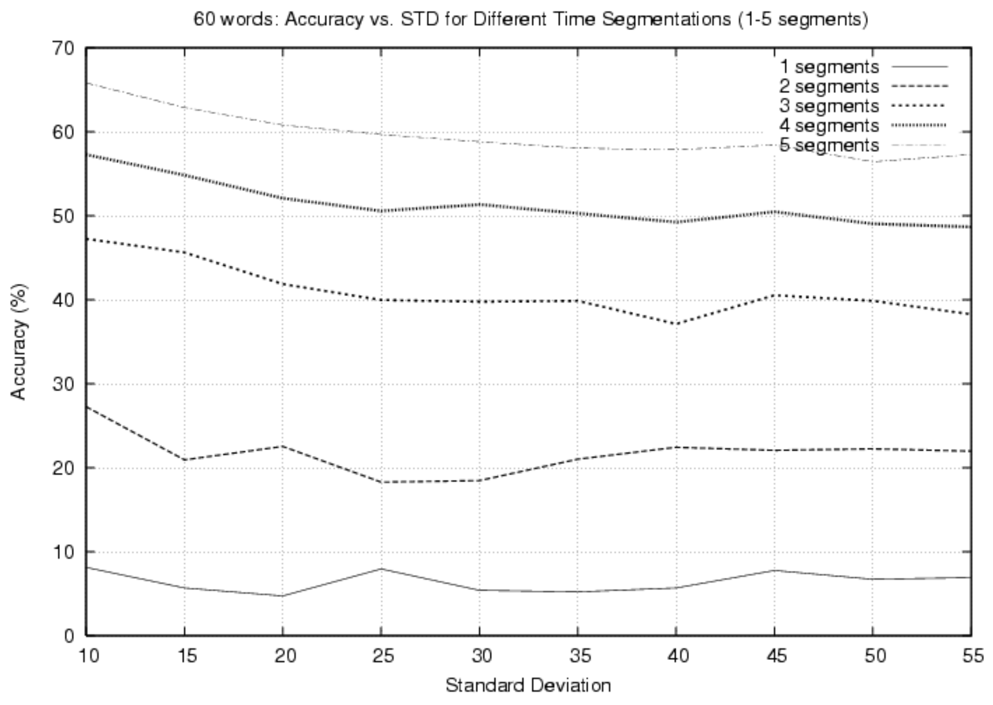

Figure 5.27: Accuracy for different time segmentations for the RBF kernel (1-5) for ten words

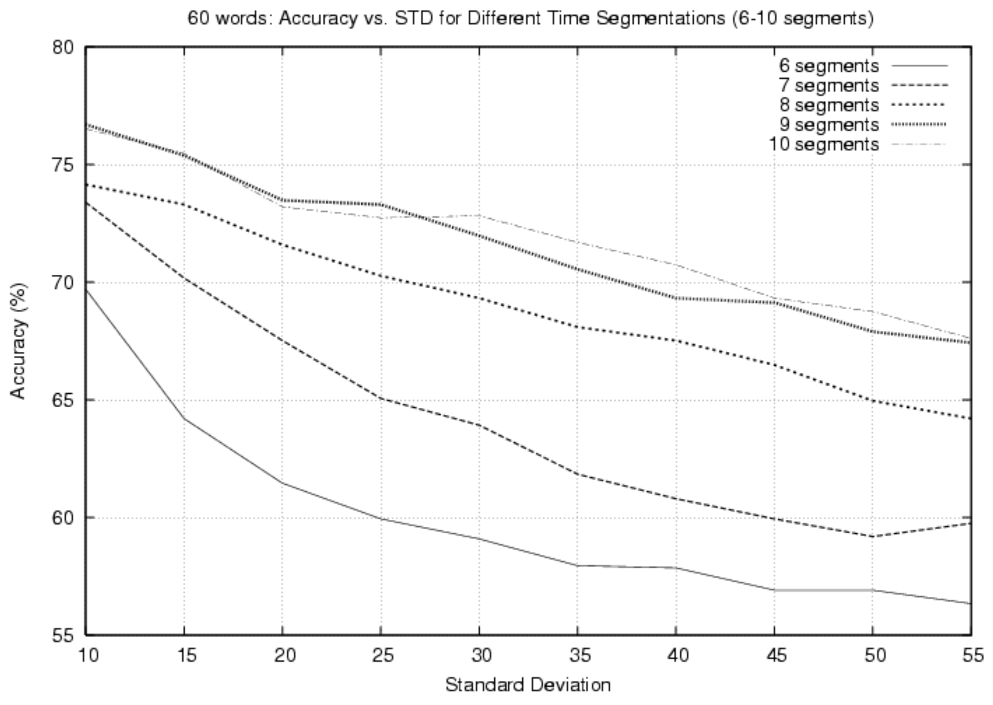

Figure 5.28: Accuracy for different time segmentations for the RBF kernel (6-10) for sixty words.

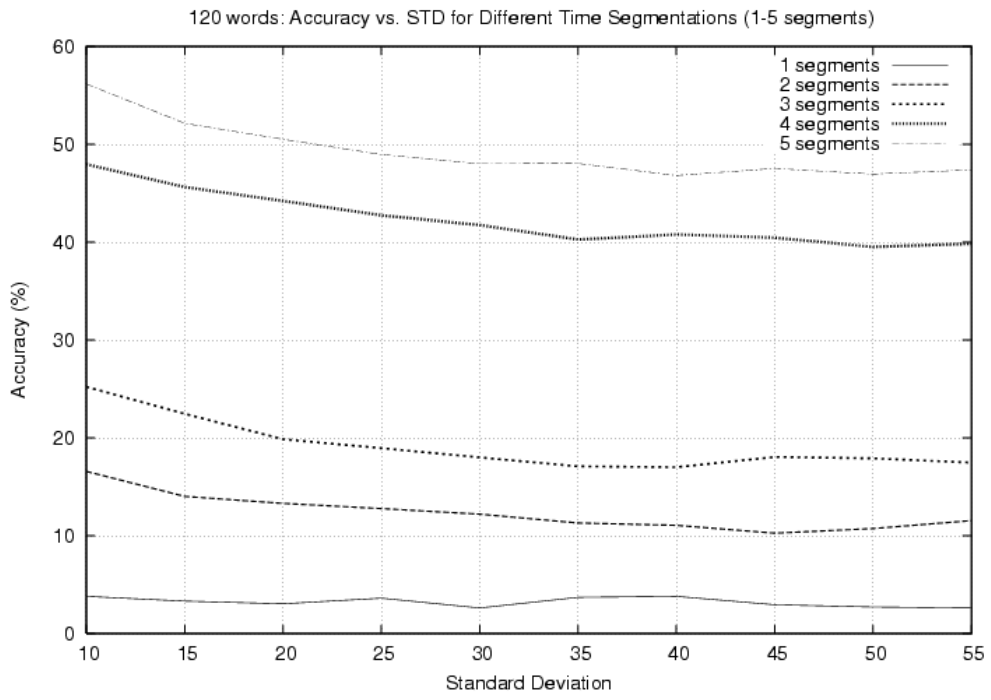

Figure 5.29: Accuracy for different time segmentations for the RBF kernel (1-5) for one hundred and twenty words

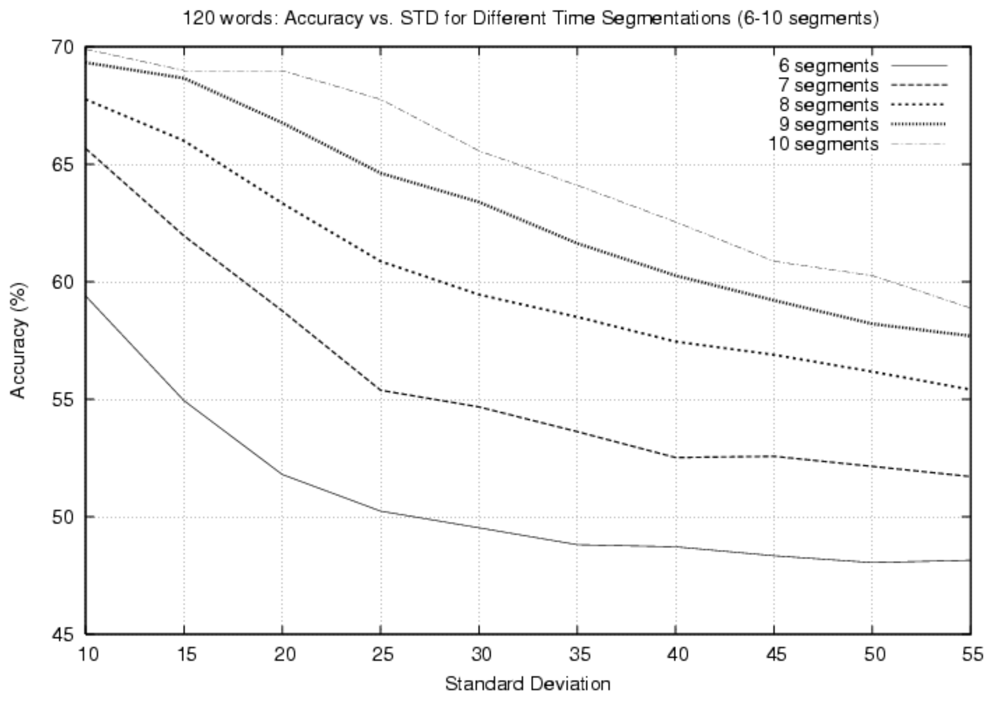

Figure 5.30: Accuracy for different time segmentations for the RBF kernel (6-10) for one hundred and twenty words

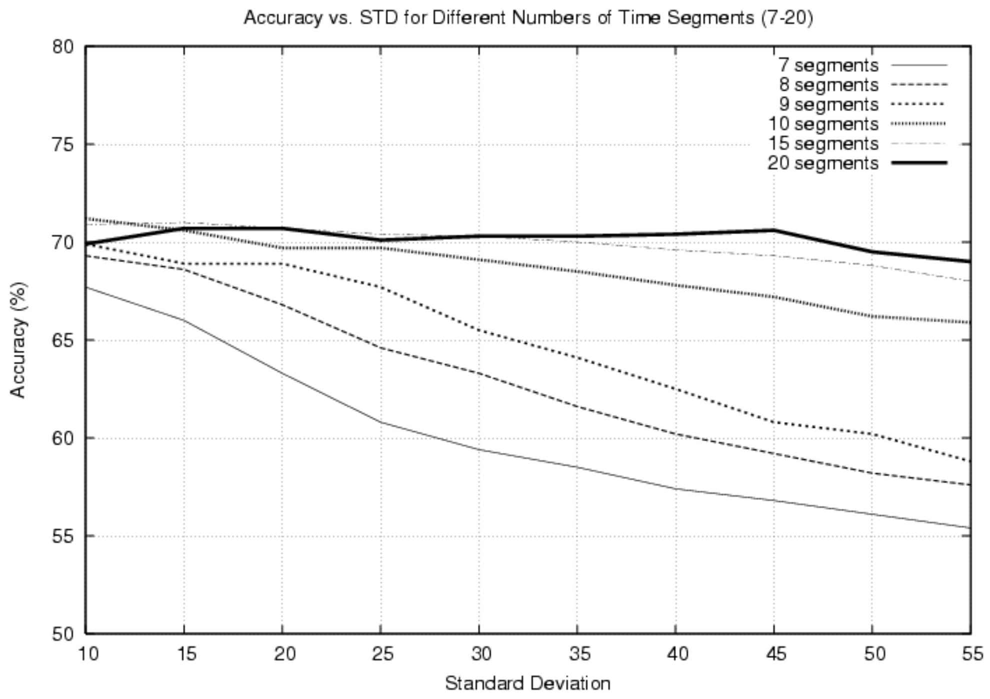

Figure 5.31: Accuracy for different time segmentations for the RBF kernel (7-20) for one hundred and twenty words

The generalization performance of an SVM is inversely proportional to the number of data points identified as support vectors. The percentage of data points that are support vectors is a bound on the expected generalization error [38]. This relationship between the number of support vectors and the generalization error is evident in the results obtained in this research work. Tables 5.1 and 5.2 show the error and the average number of support vectors for various values of C when using the radial basis function kernel with a standard deviation of 10 with 10 words and 10 time segments.

TABLE 5.1: Error Percentages for Different Numbers of Support Vectors for 10 words

C	Average Number of Support Vectors	Percentage of Support Vectors	Training Accuracy	Testing Accuracy
10	67	18.61	99.44	90.00
20	71	19.72	96.94	86.67
30	82	22.78	95.00	82.78
40	88	24.44	94.17	84.44
50	90	25.00	90.83	81.67

TABLE 5.2: Error Percentages for Different Numbers of Support Vectors for 120 words

C	Average Number of Support Vectors	Percentage of Support Vectors	Training Accuracy	Testing Accuracy
10	123	2.92	95.67	69.90
20	122	2.89	87.54	62.49
30	132	3.13	83.83	59.07
40	140	3.32	81.7	57.26
50	143	3.40	80.39	56.41

5.4 Limitations

The deficiencies of the PowerGlove are the main factors limiting the accuracy of the system. In addition to being single-handed, the sensors of the PowerGlove are susceptible to hysteresis. The glove sometimes sends data that is correlated to some extent with the previous signal rather than data that is completely independent of preceding signs.

Another factor affecting the glove's accuracy is that the ultrasonic acoustic mechanism used for position tracking is susceptible to random ultrasonic noise resulting from fluorescent light bulbs and signal reflections on rigid surfaces. It is also susceptible to the loss of line of sight during some signals. The limited accuracy of the sensors is also a major disadvantage of the PowerGlove. Flex sensors are only available for four fingers. Yet, only two bits of resolution are available for storing finger bend measurements and the abduction between fingers isn't measured. Roll is the only rotational measurement available (yaw and pitch aren't measured).

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

A main objective of this research work is to contribute to the community of physically challenged people by offering a simple interface device that aids communication. Another main objective is to investigate an unexplored area of application of the SVM algorithm which is a relatively new algorithm that has shown promising results in many real-world applications. The thesis is also a contribution to the research in Arabic Sign Language which has no research done in its automated recognition. Achieving cost effectiveness, reasonable performance and the use of off-the-shelf components are main design objectives. Although the system's performance is limited by the PowerGlove's deficiencies, the system has demonstrated reasonable performance and designing it can be seen as stepping stone for the development of more sophisticated systems with better generalization performance and better response.

6.2 Future Work

The performance of the system can be greatly enhanced by utilizing a glove with higher accuracy (such as the CyberGlove). Following the general trend in electronic devices, the availability and of such an option at a reasonable price is expected. Optimizing the code implementing pattern recognition engine and running it on a dedicated processing machine can add to the performance of the system as well. Using two hands rather than one is an option worth investigating in future works. Using more sophisticated features could be also tested in future works.

NOMENCLATURE

x_i	An input vector
s_i	A support vector
y_i	A classification label
w	Weight vector b
b	Threshold
d_+	Positive example margin
d_-	Negative example margin
l	Number of data examples
N	Number of support vectors
L_P	Lagrangian
L_D	Dual Lagrangian
α_i	A Lagrange Multiplier
$f(x)$	The decision function (hypothesis)
$K(x, y)$	A kernel function
ϕ	Mapping from input to feature space
Z	Input space
F	Feature space

BIBLIOGRAPHY

[1] M. Mohandes, “Arabic Sign Language Recognition”, in the *Proceedings of the International Conference of Imaging Science, Systems and Technology (CISST’2001)*, Las Vegas, Nevada, USA, 2001, pp. 25-28.

[2] O. Jarrah and A. Halawani, “Recognition of gestures in Arabic Sign Language using neuro-fuzzy systems”, *Artificial Intelligence,* pp. 117-138, 2001.

[3] J. Davis and M. Shah, “Gesture recognition,” University of Central Florida, Tech. Rep. CS-TR-93-11, 1993.

[4] B. Dorner, “Chasing the color glove: visual hand tracking,” M. S. Thesis, Simon Fraser University, 1994.

[5] T. Starner, “Visual recognition of American Sign Language using hidden Markov models,” M. S. Thesis, MIT Media Lab, July 1995.

[6] K. Isibuchi, H. Takemura, and K. Kishino, “Real-time hand shape recognition using a pipe-line image processor,” in *Proceedings of the IEEE International Workshop on Robot and Human Communications*, 1992, pp. 111-116.

[7] M. Krueger, *Artificial Reality*. Addison-Wesley, Reading Mass., second edition, 1990.

[8] J. Kramer and L. Leifer, "The Talking Glove for non-verbal deaf individuals," Center For Design Research, Stanford University, Tech. Rep. CDR TR 1990 0312, 1990.

[9] P. Vamplew, "Recognition of sign language gestures using neural networks," in *Proceedings of the First European Conference on Disabilities, Virtual Reality and Associated Technologies (ECDVRAT)*, Maidenhead, England, 1996, pp. 27-33.

[10] G. Grims, "Digital Data Entry Glove Interface Device," U. S. Patent 4,414,537, AT&T Bell Labs, November, 1983.

[11] D. J. Sturman and D. Zeltzer, "A Survey of Glove-Based Input," *IEEE Computer Graphics and Applications*, vol. 14, issue 1, pp. 30-39, January 1994.

[12] The CyberGlove Datasheet, Immersion Corporation, http://www.immersion.com/3d/docs/cyberglove_datasheet.pdf, 2002.

[13] C. Charayaphan and A. Marble, "Image processing system for interpreting motion in American Sign Language," *Journal of Biomedical Engineering*, vol. 14, pp. 419-425, September 1992.

[14] B. Dorner and E. Hagan, "Towards an American Sign Language Interface,"*Artificial Intellegince Review*, vol. 8, pp. 235-253,1994.

[15] J. Isaac and S. Foo, "Hand pose estimation for American sign language recognition," in *Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory,* 2004, pp. 132-136

[16] B. Bauer and K. Kraiss, "Video-based recognition using self-organizations subunits," in *Proceedings of the 16th International Conference on Pattern Recognition*, 2002, pp. 434-437.

[17] H. Brashear, T. Starner, P. Lukowicz, H. Junker and G. Troster, "Using multiple sensors for mobile Sign Language Recognition," in *Proceedings of the Seventh IEEE International Symposium on Wearable Computers (ISWC),* 2003, pp. 45-52.

[18] Y. Yoon and K. Jo, "Hand Shape Recognition using moment invariant for the Korean sign language recognition," in *Proceedings of the 7th Korea-Russia International Symposium, KORUS,* 2003, pp. 308-313

[19] S. S. Fels and Hinton, “GloveTalk: A neural network interface between a DataGlove and a speech synthesizer,” *IEEE Transactions on Neural Networks*, vol. 4, pp. 2-8, 1993.

[20] S. Sidney Fels. “Glove-TalkII: mapping hand gestures to speech using neural networks - An approach to building adaptive interfaces,” PhD dissertation, Computer Science Department, University of Toronto, 1994.

[21] K. Murakami and H. Taguchi, “Gesture recognition using recurrent neural networks,” in *Special Interest Group on Computer-Human Interaction (SIGCHI),* 1991, pp. 237-242.

[22] R. Erenshteyn, P. Laskov, R. Foulds, L. Messing and G. Stern, “Recognition approach at gesture language understanding,” in *Proceeding of International Conference on pattern recognition (ICPR)*, 1996, pp. 431-435.

[23] J. Weissmann and R. Salmon, “Gesture recognition for virtual reality applications using data gloves and neural networks,” in *International Joint Conference on Neural Networks (IJCNN),* 1999, pp. 2043-2046.

[24] W. Jiangqin, G. Wen, S. Yibo, L. Wei and P. Bo, “A Simple language recognition system based on data glove” in *Proceedings of International Conference on Signal Processing (ICSP)*, 1998, pp. 1257 – 1260.

[25] T. Takahashi and F. Kishino, “Gesture coding based on experiments with a hand gesture interface device,” *Special Interest Group on Computer-Human Interaction (SIGCHI) Bulletin*, vol. 23, issue 2, pp. 67-73, April 1991.

[26] C. Vogler and D. Metaxes, “Parallel Hidden Markov Models For American Sign Language Recognition,” in *Proceedings of International Conference on Computer Vision*, September 1999, pp. 22-25.

[27] W. Kadous, “GRASP: recognition of Australian Sign Language using instrumented gloves,” B. S. Thesis, The University of New South Wales, 1995.

[28] S. Mehdi and Y. Khan, “Sign language recognition using sensor gloves,” in *Proceedings of the 9th International Conference in Neural Information Processing* (ICONIP’02), 2002, pp. 2204-22-6.

[29] G. Fang, W. Gao and J. Ma, “Signer-independent sign language recognition based on SOFM/HMM,” in *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, 2001. pp. 90 -95.

[30] G. Fang and W. Gao, “A SRN/HMM for signer-independent continuous sign language recognition,” in *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR’02)*, 2002. pp. 297-302.

[31] C. Wang, W. Gao and S. Shan, “An approach based on phonemes to large vocabulary Chinese sign language recognition,” in *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR’05),* 2002, pp. 393-398

[32] Y. Chen, W. Gao, G. Fang, C. Yang and Z. Wang, “CSLDS: Chinese sign language dialog system,” in *Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures (AMFG’03)*, 2003, pp. 236-237.

[33] R. Collobert and S. Bengio, “Support vector machines for large scale regression problems”, *IDIAP-RR-00-17*, 2000, Available at ftp://www.idiap.ch/pub/reports/2000/rr00-17.ps.gz

[34] Student Chapter of the Association for Computing Machinery at the university of Illinois, *PowerGlove Serial Interface Version 2*, Urbana-Champaign Campus, 1994.

[35] V. Vapnik, “Structure of statistical learning theory,” in *Computational Learning and Probabilistic Reasoning*, A. Gammerman, UNICOM, John Wiley and Sons, 1996, pp. 3-31.

[36] B. Boser, I.Guyon and V. Vapnik, “A training algorithm for optimal margin classifier,” in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT 1992)*, 1992, pp. 144-152.

[37] C. Burgs “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, issue 2, pp. 121-167, 1998.

[38] N. Cristianini and J, Shawe-Taylor, *Support Vector Machine and other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[39] B. Scholkopf & A. Smola. *Learning with Kernels, Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA, 2002.

[40] V. Vapnik. *Statistical Learning Theory*, John Wiley & Sons, Inc., 1998.

[41] J. C. Platt. “Fast training of support vector machines using sequential minimal optimization,” in *Advances in Kernel Methods - Support Vector Learning,* MIT Press, Cambridge, MA, 1998.

[42] S.S. Keerthi, S. K. Shevade, C. Bhattacharyya & K.R.K. Murthy, “Improvements to Platt's SMO algorithm for SVM classifier design,” *Neural Computation*, vol. 13, pp. 637-649, March 2001.

[43] M. Mohandes, S. Al-Buraiky, T. Halawani and S. Al-Baiyat, "Automation of the Arabic Sign Language Recognition," in *Proceedings of the 2004 International Conference on Information and Communication Technologies: From Theory to Applications*, 2004, pp. 479-480.

Vita

- Salah M. S. Al-Buraiky
- Born in Illinois, USA in 1977.
- Received a Bachelor of Science in Mechanical Engineering, with first honors, from King Fahd University of Petroleum and Minerals in August 1998.
- Working in the Data Networking Engineering Division of Saudi Aramco.
- Completed Master's degree requirements in Electrical Engineering at King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia in 2004.
- Email: salah.buraiky@aramco.com