UNIVERSITY OF THE WESTERN CAPE

An Integrated Sign Language Recognition System

by

Warren Nel

A thesis submitted in fulfillment for the
degree of Master of Science

in the
Faculty of Science
Department of Computer Science

Supervisor: Mehrdad Ghaziasgar
Co-supervisor: James Connan

February 2014

Declaration

I, Warren Nel, declare that this thesis titled, 'An Integrated Sign Language Recognition System' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ________________________

Date: ________________________

"A good head and good heart are always a formidable combination. But when you add to that a literate tongue or pen, then you have something very special."

Nelson Mandela

Abstract

Research has shown that five parameters are required to recognize any sign language gesture: hand shape, location, orientation and motion, as well as facial expressions. The South African Sign Language (SASL) research group at the University of the Western Cape has created systems to recognize Sign Language gestures using single parameters. Using a single parameter can cause ambiguities in the recognition of signs that are similarly signed resulting in a restriction of the possible vocabulary size. This research pioneers work at the group towards combining multiple parameters to achieve a larger recognition vocabulary set. The proposed methodology combines hand location and hand shape recognition into one combined recognition system. The system is shown to be able to recognize a very large vocabulary of 50 signs at a high average accuracy of 74.1%. This vocabulary size is much larger than existing SASL recognition systems, and achieves a higher accuracy than these systems in spite of the large vocabulary. It is also shown that the system is highly robust to variations in test subjects such as skin colour, gender and body dimension. Furthermore, the group pioneers research towards continuously recognizing signs from a video stream, whereas existing systems recognized a single sign at a time. To this end, a highly accurate continuous gesture segmentation strategy is proposed and shown to be able to accurately recognize sentences consisting of five isolated SASL gestures.

Acknowledgements

First and foremost, I would like thank God for blessing me each and everyday. I would like to thank Him for the strength, wisdom, and faith to make it through life's everyday challenges.

I would like to thank my parents for the opportunity to study. There are so many things that I want to thank them for. They are the reason behind all of my success and I am forever indebted to them. I would like to thank my brother Edi, his wife Sharney, my sister Cheslyn and her husband Gavin, and my close cousins Ozzy and Elroy for all of their support.

I would especially like to thank my supervisor Mr Mehrdad Ghaziasgar for his outstanding guidance, inspiration and encouragement – "merci". Thank you for nurturing me into a hard-working student and for all of your patience and guidance. You were more than just a supervisor to me, you were and still are a true friend. I would also like to thank my co-supervisor Mr James Connan for his guidance and support from afar.

I would like to thank Telkom for sponsoring me throughout this degree. This sponsorship made this degree that much more possible.

To my labmates, Mr Imran Achmed, Mr Dane Brown, Mr Ibraheem Frieslaar, Mr Diego Mushfieldt, Mr Kenzo Abrahams, and Mr Roland Foster – thank you for creating the ideal learning environment in our lab. The lab was place where fun met quality research. Imran and Dane thank you very much for your generous advice and assistance. I wish the SASL group at UWC all the prosperity for the future.

I would like to thank the University of the Western Cape for the opportunity to study here. I thank my lecturers for educating me and helping me acquire knowledge in the field of computer science and other fields in life. I would especially like to thank Mr Reg Dodds – his passion for Computer Science was extremely motivational.

I would like to thank all of my train and test subjects for their participation in the testing phase of the project.

I would also like to thank my close friends Lindy, Carla-louise, Gonven, Chad, Adrian, Marco, and Tashwell for their friendship and contributions towards my thesis. I would like to extend a sincere gratitude to Carla for her patience, sense-of-humor, support, and understanding as it was extremely helpful throughout this project.

I am very thankful to all the above for being the solid support system in my life.

Contents

List of Figures

List of Tables

Abbreviations

2D	**2** **D**imensional
3D	**3** **D**imensional
CCA	**C**onnected **C**omponent **A**nalysis
CPU	**C**entral **P**rocessing **U**nit
DT	**D**istance **T**ransform
FPS	**F**rames **P**er **S**econd
GHz	**G**iga**h**ertz
GMM	**G**aussian **M**ixture **M**odels
HCDM	**H**ierarchical **C**hamfer **D**istance **M**atching
HMM	**H**idden **M**arkov **M**odels
HSV	**H**ue **S**aturation **V**alue
LibSVM	**Lib**rary of **S**upport **V**ector **M**achines
PF	**P**article **F**ilters
RAM	**R**andom **A**ccess **M**emory
SASL	**S**outh **A**frican **S**ign **L**anguage
SL	**S**ign **L**anguage
SVM	**S**upport **V**ector **M**achines

Chapter 1

Introduction

1.1 Background and Motivation

Effective communication is an essential skill required to facilitate effective interactions amongst human beings. Communication is the process by which information is exchanged between individuals. For effective communication between two parties, both parties should have the ability to understand each other by understanding and acknowledging the information conveyed. It is estimated that there are between 300 thousand and 1.5 million Deaf people in South Africa, and 70 million Deaf people worldwide [2, 18, 73]. The majority of these people use sign languages as their primary and even sole means of communication. A communication barrier exists when a Deaf person and a hearing person attempt to communicate [18].

Contrary to common belief, sign languages are not the signed equivalent of spoken languages. Sign languages are fully fledged languages with grammatical and syntactic structures that are distinct to those of spoken languages. Sign languages were developed within Deaf communities arising from the need to communicate with each other with little or no influence from spoken languages.

It is also misconceived that sign language is a single universal language that the Deaf communicate in globally. Most countries have a distinct sign language of their own. For example, in the United States of America the Deaf communicate in American Sign Language and in Britain the Deaf communicate in British Sign Language. These languages are, for the most part, distinct. Examples of other sign languages that are similarly distinct include Chinese Sign Language, Japanese Sign Language, Australian Sign Language, South African Sign Language, among others, each unique to a specific country, and generally distinct from other sign languages. Moreover, it is commonly incorrectly assumed that the Deaf can read and write spoken languages. Statistics indicate that 66% of the South African Deaf are illiterate [63].

The primary language amongst the Deaf in South Africa is South African Sign Language (SASL). The ability for the Deaf to read and write in spoken languages requires special training, which may be expensive and unavailable to most of the South African Deaf. Statistics indicate that only 40% of the South African Deaf attend school and only 12% of the Deaf have education that is higher than grade 8. It has also been found that only 14% of teachers providing specialized training to the Deaf are able to sign [63]. This limits the effectiveness of the education. Holt found that Deaf and hard of hearing students that were taught to read and write in spoken languages achieved a grade rating of 4.5 at age 17, a vast under-achievement as compared to hearing students at the age of 15 that achieved a grade rating of 10 [51]. According to a newsletter [63], South Africa has only graduated only 32 Deaf tertiary students.

It is clear that there are limited educational resources and job opportunities for the South African Deaf as a result of their inability to communicate in spoken languages. 70% of the South African Deaf are unemployed. As a result, many of the Deaf are victims of poverty, with 68% of the Deaf living in informal settlements. The use of skilled interpreters may be considered a solution to breach the communication barrier. However, skilled interpreters are very expensive and scarce and their use and has been limited to official applications rather than everyday common use.

The South African Sign Language (SASL) research group at the University of the Western Cape is currently in the process of developing a machine translation system that can automatically translate phrases between SASL and English. The eventual system aims to complement the use of interpreters and provide a cheaper and more accessible everyday solution to the problem of Deaf communication. The SASL group has made significant contributions towards the innovation in technologies to assist in the translation between SASL and English. The current focus of the group is in the field of gesture recognition of SASL.

One of the main aims of the project is to make use of inexpensive commodity hardware such as monocular cameras, making use of computer vision to extract sign language semantic information from 2D video. This is in contrast to the use of expensive hardware placed over the body to track the user's hands and body. It also aims to avoid solutions that require the user to wear specialized markers over the hands or body, which would undoubtedly simplify the problem, but place constraints on the use of the system. Since South Africa is home to a diverse mix of ethnicities and skin colours, the project also aims to produce recognition systems that are robust to skin colour and body dimensions.

According to Stokoe [111], there are five parameters that uniquely characterize each sign language gesture. These parameters are hand location, shape, motion and orientation, and non-manual gestures in the form of facial expressions. Several systems have been developed by the group each focusing on the recognition of single sign language gesture parameters. To date, the group has produced three facial expression recognition systems [77, 105, 125], two hand shape recognition systems [68, 101], two hand location recognition systems [2, 18], and two hand motion recognition systems [79, 88], each achieving high recognition accuracies.

Some of these recognition strategies were applied to the task of gesture recognition of SASL signs as proofs-of-concept, with only a single parameter used to characterize gestures. While these systems yielded plausible single parameter recognition accuracies, the use of a single parameter meant that only a limited number of SASL gestures could be recognized by these systems, and the signs needed to be distinct to be recognized. Achmed [2] and Brown's [18] recognition systems had a vocabulary size of 15 signs and used hand location to achieve a gesture recognition accuracy of 88% and 92.95% respectively. Naidoo [79] and Rajah's [88] recognition systems had the highest SASL vocabulary size of 20 and 23 signs, respectively, and used only hand motion to recognize gestures at an accuracy of 69% and 71% respectively.

All of these gesture recognition strategies take a video containing only a single isolated gesture as input and produce the English phrase corresponding to the SASL gesture as output. None of these systems have a phrase segmentation strategy towards continuous recognition of entire SASL phrases from a continuous video stream.

This research pioneers efforts by the SASL group to use and integrate multiple parameters to recognize a larger number of SASL gestures. The current research aims to use two parameters, namely hand shape and hand location, to recognize a larger SASL vocabulary. Furthermore, this research pioneers and lays the foundation for a continuous SASL gesture recognition strategy at the group. This will serve as the framework into which other parameter recognition strategies will ultimately be incorporated in future. This will ultimately provide the base on which to recognize entire SASL phrases in future.

Research has been conducted into continuous multi-parameter sign language gesture recognition. The majority of such research makes use of specialized hardware worn by the signer – such as Data Gloves, Power Gloves or specially marked colour-coded gloves – or complex camera configurations – such as stereo or Kinect cameras – to retrieve sign language parameters. None of these strategies are able to recognize gestures based on monocular video input in the absence of specialised hardware, robust to abitrary image background configurations and signer skin colour. This is another unique aspect of this research in the field of cotinuous multi-parameter recognition.

With the availability of extensive corpora in other sign languages such as American Sign Language, British Sign Language and Chinese Sign Language, it has been possible to extend continuous gesture recognition research towards the recognition of entire phrases of various length [20, 121, 123, 128]. In most cases, Hidden Markov Models (HMMs) are used to recognize and infer the meaning of phrases. Digital phrase recognition requires knowledge about the unique structure and syntax of a language in the form of a digitial corpus which is currently unavailable in the case of SASL [42, 45, 48]. Therefore, the current research is limited to the continuous recognition of gestures in a SASL video in isolation of each other.

The majority of two-parameter sign language gesture recognition research involves the use of the hand shape and hand motion parameters to recognize gestures. Another unique aspect of this research is an attempt at using hand shape and hand location to characterise and recognise SASL gestures.

1.2 Research Scope

The use of hand location to characterize gestures implies that the scope of the research is limited to recognizing gestures that do not involve motions, commonly referred to as static gestures. Examples of SASL signs that do not involve any motion and can be represented by a single image are "Go" performed as a flat hand placed in front of the chest and "Goose" performed by forming an 'O' shape with the hand in front of the right shoulder. As such, this research is limited to the recognition of SASL signs that do not involve any motion – static gestures – and can be represented by a single image.

The implication of this is that it is not possible to directly compare this work to any systems that use the motion parameter to characterize gestures, such as Naidoo and Rajah's systems. The data sets used in such systems include SASL signs that do not necessarily include motions, as this is the basis of such systems. Signs that involve motions are commonly referred to as dynamic gestures and include "Hello" performed by placing the hand on the forehead with a flat hand and moving upwards or rightwards and "Stress" performed by forming fists with both hands placed together in the middle of the chest and moving outwards and inwards continuously three or four times. Dynamic gestures can not be represented by a single image.

Therefore, a direct comparison between motion-based systems and the proposed system are not possible. However, a comparison in vocabulary size and overall accuracy may be considered.

Based on this, continuous recognition of static gestures can be achieved by assuming any motions of the hands as transitions between static gestures and poses of the hands as a single static gesture to be recognized.

1.3 Research Question

Considering the challenges posed in the previous section, the following research question is formulated: "Can static SASL gestures be continuously recognized at a high accuracy for a person of any skin tone from a monocular video feed using a combination of two sign language parameters – hand shape and hand location?".

1.4 Research Objectives

1. Implement a robust hand tracking strategy and a hand shape recognition strategy.
2. Use these features to implement a gesture recognition component that recognizes a larger number of SASL gestures.
3. Demonstrate that the strategy can recognize a larger vocabulary than the currently existing SASL gesture recognition strategies.
4. Demonstrate that the strategy is robust to skin colour variations.

1.5 Premises

The research is based on the following assumptions:

- It is assumed that the signer will stand in front of an arbitrary background and not require any special equipment such as data gloves or coloured markers. This is a requirement of the SASL project to attempt to provide the most natural feel to the system.
- It is assumed that only a single signer will be in the view of the camera at any time. In the eventual system, the Deaf user of the system can re-locate to a quiet area in which to use the system. The background, however, should not concern the signer. Therefore, this assumption is also justified.

1.6 Thesis Outline

The remainder of the thesis is arranged as follows:

Chapter 2: *Related Work*: This chapter reviews existing literature in the fields of articulated object tracking and recognition, and gesture recognition. It explores the various approaches and algorithms used to attain varying accuracies. The strengths and weaknesses of the various approaches are discussed in order to select an appropriate hand tracking and hand shape recognition strategy, as well as justify this research.

Chapter 3: *Image Processing For Gesture Recognition*: This chapter details the components that form part of the proposed gesture recognition system. This discussion forms the basis of understanding for the implementation chapter that follows.

Chapter 4: *Design and Implementation of the Integrated Sign Language Recognition System*: This chapter details the proposed gesture recognition system. It details the isolated gesture segmentation strategy used to segment gestures in a continuous image sequence. It also describes the method used to extract hand shape and hand location features from a segmented gesture frame. Subsequently, a description of the gesture classifier used and its optimization is described.

Chapter 5: *Experimental Setup and Analysis of Results*: This chapter discusses the experiments carried out to assess the accuracy of components of the proposed gesture recognition system, as well as the accuracy of the continuous gesture recognition strategy as a whole, in order to answer the research question posed in this chapter.

Chapter 6: *Conclusion*: This chapter concludes the thesis by highlighting the contributions made towards research in the field. It also provides possible directions for future work.

Chapter 2

Related Work

The system proposed in this research has two components: the feature extraction component and the gesture recognition component. The feature extraction component aims to extract features pertaining to the two sign language parameters set out in the previous chapter, namely, hand shape and hand location. In order to extract these features, the first task that has to be carried out is accurately tracking the hand across the frames of the image sequence. The fact that the hand is an articulated object makes this task challenging and complex. There are various methods that can be used to achieve this goal. Section 2.1 details the related work in the field of articulated object tracking and justifies a choice of tracking technique to this end.

Once the hand is accurately tracked, features need to be extracted to infer the location and hand shape. Accurate tracking automatically provides the location of the hand. What remains is to determine the shape of the hand. Once again, the fact that the hand is an articulated object makes the task of shape recognition challenging. Section 2.2 details the related work in the field of articulated object recognition and justifies the choice of a technique towards hand shape recognition.

Finally, the extracted features are used to characterize and recognize a set of gestures. Section 2.3 details the related work in the field of gesture recognition, with the majority of focus on studies that focused on sign language gesture recognition. It presents information pertaining to a variety of aspects of each study where possible, including the input, the size of the recognized vocabulary, the pre-processing procedure, the sign language phrase segmentation strategy where applicable, the choice of classification technique, the training and testing sets and the recognition accuracy. This is used to demonstrate the novel aspect of the proposed system and further motivate the need for this research.

The chapter is then concluded.

2.1 Articulated Object Tracking

A common problem that arises in vision-based systems is tracking an object through a sequence of images. It can be a challenging task to segment a desired object from the background, especially when the background is complex. Rigid objects, or those that are slightly articulated may be easier to track considering the object takes on a relatively small number of known shapes. Depending on the object's orientation and location with respect to the camera, the object may appear differently.

This task becomes even more complex when the desired target object is highly articulated and/or deformable. Articulated objects are deformable and can appear in and take on numerous varied forms. This becomes even more complex when considering that these forms may appear differently as projected in the camera imaging plane. Variations arise from the pose of the object due to its motion, deformation, and articulation such as a changing hand shape, as well as from the potentially complex ambient environment [74]. A good example of such a highly articulated and deformable object is the hand which consists of many articulations and, as such, has numerous degrees of freedom. Taking into consideration all the factors that contribute to the complexity of tracking articulated objects, it can be seen that a robust tracking technique is required to locate the hand before it is possible to extract relevant features from it to recognize the hand shape.

The methods that can be used to track articulated objects can broadly be divided into those are model-based and those that are not, henceforth referred to as "model-free". Model-based methods are methods that assume specific shapes on the part of the target object and use these shapes to locate and track the object across frames. Model-based methods are unsuitable for the task of hand tracking because the hand is a highly deformable object with several degrees of freedom. It is not possible to assume a constant shape on the part of the hand. One solution is to make use of a comprehensive training set that consists of all possible deformative variations of the hand. This solution is, however, not practical since it is very difficult to construct a converged model with all possible deformations of the hand, taking into account that it has x degrees of freedom. In any case, the performance of such a tracking system is expected to be computationally expensive and less than real-time. As a result, these methods are not considered in this research.

Model-free tracking methods have been proven to track hand motions effectively [57, 67, 116]. Model-free tracking methods make use of various cues to track objects. The three most popular tracking methods include Mean-Shift, particle filters and optical flow. These methods may make use of chrominance, luminance, edge and/or motion information to track objects. In principle, they are able to accurately track objects regardless of their shape, even in complex environments. As such, they are suitable to the task of tracking articulated objects. Two factors that can affect the performance of such methods are the chrominance uniformity and luminance uniformity of the object. In the presence of constant lighting, model-free methods can yield highly accurate tracking results. As a result, these methods are considered in this research.

This chapter focuses on model-free articulated object tracking techniques. Although there are many tracking techniques, this chapter focuses on Mean-shift, particle filters, a combination of Mean-shift and particle filters, and optical flow. Sections 2.1.1 through 2.1.4 discuss four popular model-free tracking techniques, namely: Mean-Shift, particle filters, Mean-shift-particle filters hybrid methods, and optical flow. We conclude our analysis in Section 2.1.5 where we motivate the use of a suitable hand tracking technique.

2.1.1 Mean-Shift

Mean-Shift is a colour-based tracking algorithm. The Mean-Shift algorithm was initially proposed by Fukunaga and Hostetler [47] for use in pattern recognition but was subsequently applied to other fields like Computer Vision, specifically cluster analysis. Cluster analysis is a procedure which attempts to group a set of objects according to some criterion so as to ensure that grouped objects – termed as "clusters" – are more similar to each other in terms of the grouping criterion than the objects in other clusters. Applied to cluster analysis, the Mean-Shift algorithm is computationally inexpensive and has a non-parametric clustering procedure which does not require prior knowledge of the number of clusters or nodes, nor does it constrain the shape of the clusters. Contrary to the k-means clustering approach [24, 43, 89], there are no embedded assumptions on the shape and distribution, the number of nodes or clusters. The Mean-Shift algorithm works well on static probability distributions but not as well as dynamic probability distributions such as movies [27].

The Mean-Shift algorithm initially requires that the target object be specified, usually by manual selection. A search window of known size, orientation and shape is positioned around the target object. Within an image processing context, a histogram of the region of the image within the search window is computed. This represents the colour distribution of the target object. Thereafter, the algorithm continuously carries out the following steps on each frame [27]:

1. The probability histogram is back-projected onto the current frame resulting in a probability distribution I. I summarizes the probability with which each pixel in the current frame matches the colour distribution of the target object.
2. The mean of the probability distribution I – better known as the center of mass – is computed. The center of mass represents the region with the highest number of high probability pixels which, in essence, is the highest focus of the target object in the search window.
3. The center of the search window, referred to as the centroid, is repositioned at the center of mass, thus re-positioning it ever closer to the target object.

The centroid of the search window is computed by means of the zeroth moment M_{00} and first-order moments M_{10} and M_{01} within the search window, given by:

$$M_{00} = \sum_x \sum_y I(x, y) \tag{2.1a}$$

$$M_{10} = \sum_x \sum_y xI(x, y) \tag{2.1b}$$

$$M_{01} = \sum_x \sum_y yI(x, y) \tag{2.1c}$$

where x and y range over the search window and $I(x, y)$ is the value of the probability distribution I of the pixel at (x, y). The centroid (X_c, Y_c) of the search window is then given by:

$$X_c = \frac{M_{10}}{M_{00}} \tag{2.2a}$$

$$Y_c = \frac{M_{01}}{M_{00}} \tag{2.2b}$$

The Mean-Shift algorithm has the disadvantage of assuming a static size and orientation on the part of the target object and search window. This may not be the case for the hand. While the hand moves, it may change in orientation a continuously and quickly, and may also change in size, as viewed by a monocular camera. A closed fist may appear smaller than an open hand.

Bradski extended the Mean-Shift algorithm, incorporating the ability for the search window to dynamically and adaptively change in size and orientation [17]. The new algorithm was named CamShift, which is short for Continuously Adaptive Mean-Shift. The algorithm addressed the static nature of the original Mean-Shift algorithm.

CamShift is robust to temporary occlusions of the target object. In such cases, the search window appears to first absorb the occlusion and then revert to the dominant distribution model when the occlusion passes. However, this is only the case if the colour distribution of the target and occluding objects are different. For example, the algorithm may lose track of the hand when the hand passes over the face region, due to the high skin pixel probability distribution that is present in the face region. In such a case, the search window may bind to the face region, which is a larger skin region.

Like Mean-Shift, CamShift requires that the target object be initially specified. This results in the creation of an initial search window positioned around the object. The algorithm applies Mean-Shift at every time step, but additionally computes the new orientation and size of the search window. The algorithm additionally uses the second-order moments to determine the new orientation and size of the search window at each time step [70]. The first-order moment M_{11} and second-order moments M_{20} and M_{02} are given by:

$$M_{20} = \sum_x \sum_y x^2 I(x,y) \tag{2.3a}$$

$$M_{02} = \sum_x \sum_y y^2 I(x,y) \tag{2.3b}$$

$$M_{11} = \sum_x \sum_y xyI(x,y) \tag{2.3c}$$

The new orientation of the search window is then given by:

$$\theta = \frac{arctan(\frac{2(\frac{M_{11}}{M_{00}} - x_c y_c)}{(\frac{M_{20}}{M_{00}} - x_c^2) - (\frac{M_{02}}{M_{00}} - y_c^2)})}{2} \tag{2.4}$$

The length and width of the search window may be determined by first letting:

$$a = \frac{M_{20}}{M_{00}} - x_c^2 \tag{2.5a}$$

$$b = 2(\frac{M_{11}}{M_{00}} - x_c y_c) \tag{2.5b}$$

$$c = \frac{M_{02}}{M_{00}} - y_c^2. \tag{2.5c}$$

Then the length l and width w are given by:

$$l = \sqrt{\frac{(a+c) + \sqrt{(b^2 + (a-c)^2)}}{2}} \tag{2.6a}$$

$$w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c}^2}{2}} \tag{2.6b}$$

Bradski showed that the algorithm was able to carry out tracking at a real-time frame rate of 30 frames per second (FPS) on a computer with a Pentium II processor running at 300 MHz with 256 MB RAM. Several researchers [7, 9, 27, 112, 127] have used CamShift, coupled with strategic optimizations to make the tracking technique more effective. Two of these are discussed below.

Araki *et al.* developed a hand tracking algorithm that uses CamShift, coupled with a motion mask to combine colour and movement probability distributions to achieve a "probability reduction", that is, to de-emphasize areas that may match the target colour distribution, but may not be part of the target object itself [7]. The algorithm can track the hands on multi-coloured backgrounds without the search window binding to noise regions such as the face or skin-coloured background objects. It is visually demonstrated that the tracking algorithm is accurate and more effective than the conventional CamShift algorithm. The algorithm was shown to achieve real-time tracking on a Intel Core 2 Duo 2.53 GHz CPU with 4 GB memory and a camera at a resolution of 640×480.

Askar *et al.* developed a vision-based skin-colour segmentation system which tracks the hands in real-time. The system makes use of a static skin detection method to detect and highlight skin pixels in the original image. Since only skin pixels are considered, three skin blobs comprised of the face and the two hands of the user are obtained. The hands and face are located by a combination of horizontal-vertical projection and neighborhood analysis within resulting image. The position of the face is determined by considering the face to be the largest blob. It is eliminated from the image. The CamShift algorithm is initialized by setting its search window over each of the hand blobs.

This approach was shown to track in real-time at 25 FPS on a 2 Ghz Pentium IV PC working on a video stream at a resolution of 576×720 pixels. It was demonstrated that the algorithm can perform well provided the background does not contain any pixels that resemble the skin. In such a case, more than three skin blobs may be detected resulting in a failure of the algorithm.

2.1.2 Particle Filters

Particle filters (PFs), also known as Sequential Monte Carlo methods, are online posterior probability estimation algorithms within a Bayesian recursion framework. They were first introduced in the famous condensation paper of Isard and Blake [56]. They have been shown to be efficient and effective tracking methods in a variety of contexts [14, 85, 93] such as human, vehicle and sports tracking. They have been shown to be especially robust to complex environments.

A posterior probability distribution is represented by a set of samples, each of which has a weight. The samples are also referred to as "particles", hence the name particle filters. The aim of the procedure is to sequentially and continuously determine and update the weights of all particles in time.

Particle filters may take a variety of forms, each of which makes use of a unique sampling method. Examples of particle filters include sequential importance sampling PFs, auxiliary sampling importance re-sampling PFs, sampling importance re-sampling PFs etc. The interested reader is referred to [8, 93, 109] for an extended reading.

Particle filters are based on three essential models: the observation model which weighs particles according to the associated extracted measurements; the reference model which is a reference representation of the tracked object; and the motion model according to which particles are propagated.

In general, particle filters aim to estimate filtered estimates of X_k given an entire set of measurements Z_t from time $t = 1$ upto time $t = k$.

Let $\{X^i_{0:k}, w^i_k\}^{N_s}_{i=1}$ represent an arbitrary measure used to characterize the posterior distribution function $p(X_{0:k}|Z_{1:k})$, where $\{X^i_{0:k}, i = 0, \ldots, N_s\}$ is a set of support points with associated weights $\{w^i_k, i, \ldots, N_s\}$ and $X_{0:k} = \{X_j, j = 0, \ldots, k\}$ is the set of all states up to time k. The weights are normalised such that $\sum_i w^i_k = 1$.

N samples X^i_k are drawn from the proposal distribution $q : X^i_k \sim q(X_k|X^i_{k-1}, Z_k)$. A weight w^i_k is associated with each particle as per:

$$w^i_k \propto w^i_{k-1} \frac{p(Z_k|X^i_k)p(X^i_k|X^i_{k-1})}{q(X^i_k|X^i_{k-1})Z_k} \tag{2.7}$$

and an approximation of the posterior filtered density $p(X_k|Z_{1:k})$ is given by:

$$p(X_k|Z_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(X_k - X^i_k) \tag{2.8}$$

An estimation of an effective sample size $\hat{N}_eff$ can be determined as:

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s}(w_k^i)^2} \tag{2.9}$$

Particle filters have been applied, with success, to object tracking. They have been found to be especially robust in complex environments. The main shortcomings of this algorithm are algorithm complexity, and sample impoverishment and degeneracy. The algorithm is complex and may be computationally expensive. Sample impoverishment is a situation in which an increasingly smaller number of samples reflect the true probability distribution. The extreme case is one which only a single particle remains. This results in high estimation error rates. Sample degeneracy refers to a situation in which the majority of samples end up with weights that are very close zero. The extreme case is where all but one of the weights are zero. This can take place quickly in complex environments. In such cases, particle filters become inefficient and inaccurate.

In order to counter the sample degeneracy problem, sequential importance re-sampling (SIR) particle filters were created [49]. The particles in these particle filters are propagated over time using a combination of sequential importance sampling and, very importantly, re-sampling. Particles are statistically multiplied and/or discarded at each iteration during the re-sampling phase to dynamically re-position particles in high posterior probability regions. This can help prevent sample degeneracy.

SIR particle filters were used by Spruyt *et al.* for hand tracking [109]. A combination of motion and skin cues were used as measures used in the particle filters. The skin model used was a static non-adaptive one. Gaussian Mixture Models (GMMs) were used to reduce sources of noise in images. This method ensures that only moving skin regions are tracked by the particle filters.

The suitability of particle filters to the problem of hand tracking is limited by the complexity of the algorithm and the sampling inefficiencies that can occur. These in efficiencies are further explained in the next section which discusses methods that are hybrids of Mean-Shift and particle filters. SIR Particle filters and such hybrid methods share the same drawbacks.

Rincon *et al* [93]. used particle filters to track objects in complex scenes. A novel particle filter algorithm was proposed that makes use of two sampling methods. The efficiency of the resulting particle filter algorithm is significantly enhanced. Measures from the current scene and a priori probability are used to to track the object. The method extracts pixels from the target object to generate the target object's colour distribution. The location of the target object in the next image is predicted using the sample likelihood of all particles and the probability density image.

2.1.3 Mean-Shift–Particle Filters Hybrid Methods

A number of research projects have proposed methods that combine features from both Mean-Shift and particle filters, in a bid to maximize the benefits of both methods and minimize their drawbacks. Examples include the work done by Koichiro *et al.* [33], Maggio and Cavallaro [71], Cai *et al.* [19] and Shan *et al.* [104].

Shan *et al.* proposed the Mean-Shift embedded particle filter (MSEPF), an algorithm that is a hybrid between Mean-Shift and particle filters. The algorithm uses the Mean-Shift algorithm to re-direct particles in a particle filter to nearby high probability areas after they have been propagated. In essence, Mean-Shift acts as a particle guide. It has been shown to perform well even with a small number of particles. This reduces the computational cost of the algorithm. This procedure enhances the estimation accuracy. Other studies have used a similar principle to achieve comparable results.

The advantage of the approach is that it helps prevent sample degeneracy by guiding particles to high probability areas. However, the drawback of the algorithm is the same as its advantage: Mean-Shift begins to guide all particles to a single point which is the highest probability point, causing the other particle filter problem: sample impoverishment. For particle filters to function effectively, there must be a number of sufficiently scattered particles. The algorithm begins to behave like Mean-Shift focused at a single particle. This especially becomes a problem when the target object is moving swiftly.

2.1.4 Optical Flow

Optical flow is a tracking method that assigns a motion vector to each pixel in a current image given the previous image as input. Assuming a sufficiently high capture rate of the camera and/or relatively slow motion in the scene, optical flow determines the direction and magnitude of the motion of objects within the scene. Motion of the objects in the scene can be caused by motion on the part of the objects in the scene relative to the camera, the camera relative to the scene, or both. This research assumes a stationary camera.

The method makes use of the intensity of pixels to track them. It is assumed that the intensity of the pixels in the image are static, with only their location varying. This is expressed mathematically as [13]:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \tag{2.10}$$

This can be expressed as:

$$\frac{dI}{dt} = 0 \tag{2.11}$$

Differentiating this equation by applying the chain rule yields:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{2.12}$$

If we let:

$$V_x = \frac{dx}{dt} \text{ and } V_y = \frac{dy}{dt} \tag{2.13}$$

we arrive at a single linear equation in two unknowns, V_x and V_y:

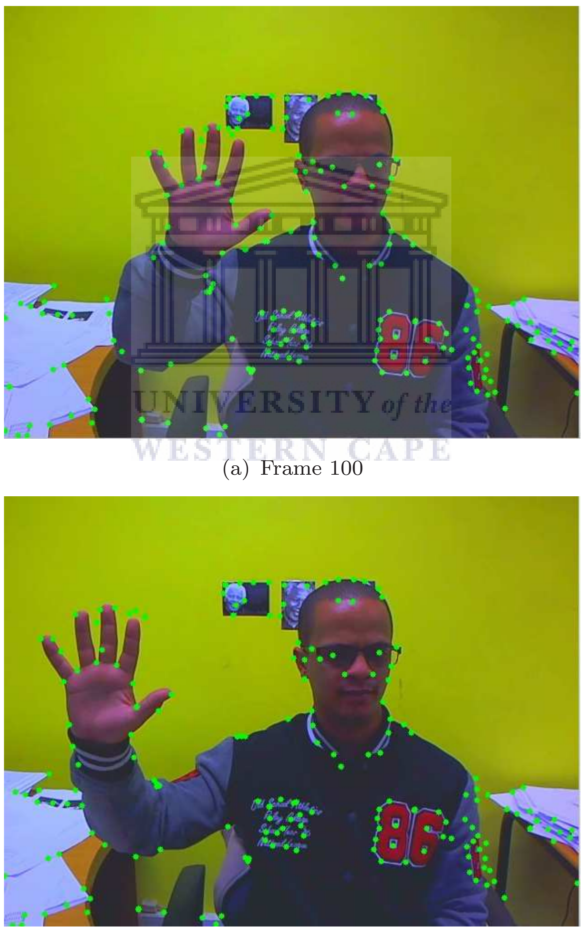
$$I_x V_x + I_y V_y + I_t = 0 \tag{2.14}$$

where I_x, I_y and I_t represent, respectively, the change in intensity with respect to x, y and t. V_x and V_y here represent, in essence, the velocity of optical flow in the x and y directions, respectively.

This equation cannot be solved without a further set of equations derived from an additional constraint assumed on the part of the scene. Various methods have been proposed to solve these equations by introducing additional constraints including phase correlation techniques, block-based methods and, most popularly, differential methods. The differential methods make specific assumptions about the flow of motion within the scene and, in turn, take various forms including the Horn-Schunck, Buxton-Buxton, Black-Jepson and, most popularly, the Lucas-Kanade method.

In all these cases, the intensity value of pixels is key to determining the flow of motion and tracking target objects in consecutive frames. Figure 2.1 illustrates a tracking sequence using the technique. The serious drawback of the technique is that it is very sensitive to noise. The assumption that pixels have a constant intensity requires for the lighting of the scene to be static. As such, even small changes in lighting during the frame sequence can severely disrupt tracking. In addition, depending on the assumption of motion, pixels with the same intensity value as the tracked pixel may be confused with the tracked pixel, resulting in tracking losses.

The most popular optical flow technique, the Lucas-Kanade method, is discussed as a base of understanding. The Lucas-Kanade method assumes small changes in motion in the scene between consecutive frames [13]. This assumption makes it possible to introduce a search window of known size around each pixel in the image, the bounds of which are assumed to be an upper limit to the vector of motion of the pixel within the scene. The size of the window is optimizable according to the specific context of application. Assuming a window size that is too large is equivalent to assuming none at all, while a window size that is too small places a constraint of very small motion within the scene between consecutive frames.



(a) Frame 100

(b) Frame 150

FIGURE 2.1: Tracking sequence using Optical Flow.

In order to provide the advantages of both a large window size and a small window size, an enhanced Lucas-Kanade optical flow method called the pyramidal Lucas-Kanade optical flow method was proposed. The algorithm constructs a pyramidal cascade, the base of which is the original image, and each ascending layer, a copy of the original at a resolution that is reduced by a constant factor p. This is illustrated in Figure 2.2.

In essence, the top of the pyramid is a summarized version of large scale features in the image, with each subsequent layer containing increasingly fine-grained information. The algorithm then applies the Lucas-Kanade optical flow algorithm to the cascade starting at the top working its way down. This makes it possible to get a global estimate of the flow of motion and use this at the next layer as the base point in which to conduct a search. Ultimately, the procedure pin-points the new location of each pixel at the lowest layer – the original image. This procedure makes it possible to assume a local neighborhood of motion but still track large motions as well.

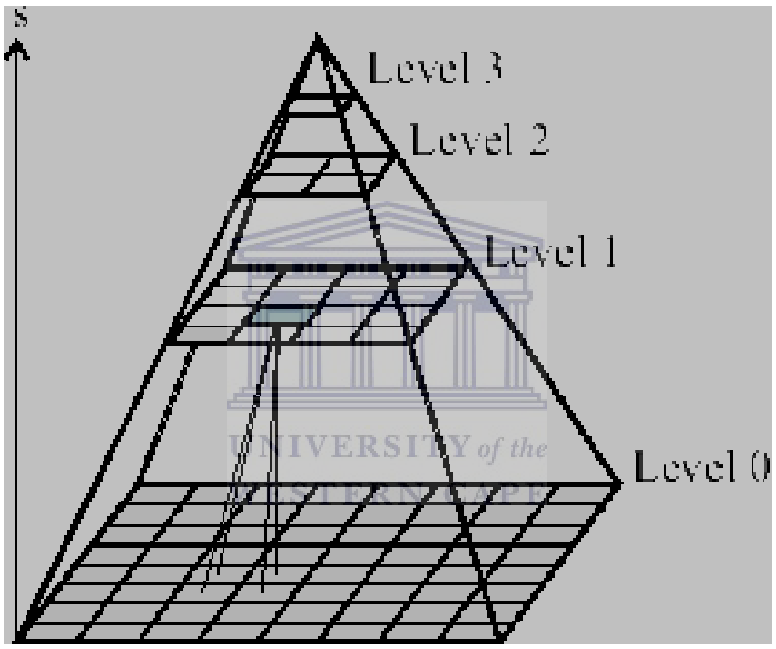


FIGURE 2.2: An illustration of the pyramidal Lucas-Kanade optical flow.

The algorithm is, however, not effective in tracking highly deformable objects such as the hands [86]. In order to address this problem and make it possible to track objects regardless of rapid and vast deformations, Kolsh and Turk enhanced the algorithm with a strategic assumption that they call “flocks of features” [65]. The main idea employed in this method is that the motion of a highly deformable object is generally similar to that of a flock of birds in flight. The birds in the flock do not wander too far off from the flock but always maintain a minimum distance from each other as well.

The method initially places a set of optical flow points around the hand to be tracked. These points represent the features in the flock. Optical flow is applied at each of the points to track the motion of the object in time with a continuous evaluation of the two aforementioned constraints: each point cannot wander off too far from the the flock; each point cannot roam too close to any other feature point.

If either of these constraints are violated, the point is placed in a position of high skin probability that does not violate either constraint [65]. In general, the position of the hand at any time is taken to be the median distance of all features in the flock. The method was shown to be able to track the hand on a computer with a 3 GHz CPU at an average near-real-time performance of 12.7 FPS on a video stream of resolution 720×480 pixels.

The method has several drawbacks. It is not robust to rapid motions. It assumes a constant and slow motion on the part of the hand. It is also not robust to complex environments. If the hand passes over a region that can be considered a higher skin probability region than the hand such as the face, the algorithm will bind to the latter region. An illustration of the tracking method is provided in Figure 2.3.The figure also illustrates that the method easily loses track when the hand passes over the cupboard – which may be considered a larger high skin probability region than the hand – and when it passes over the user's neck.

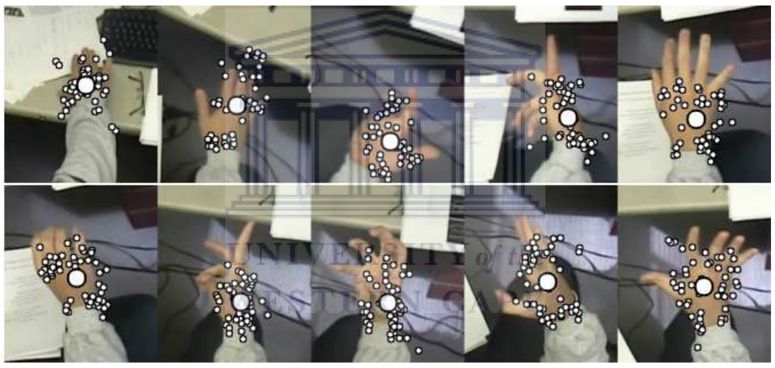


FIGURE 2.3: An illustration of “flocks of features” [65].

2.1.5 Summary of Articulated Object Tracking

Model-based methods are useful for tracking object with a static shape on an arbitrary background. However, these methods become impractical when the shape of the target object changes severely in time, which is characteristic of articulated objects such as the hands.

In such cases, model-free methods are selected as alternative tracking methods of choice. Such approaches make use of, for example, the colour or intensity of the target object to track it. Four popular tracking methods – Mean-Shift, particle filters and optical flow – were discussed. Methods that combine Mean-Shift and particle filters were also discussed. The strengths and weaknesses of each method were enumerated and illustrated. All things considered, all three methods are generally on par with each other.

However, CamShift, which is an extension of Mean-Shift, can be considered a highly efficient and robust method. It is more robust than optical flow and more computationally efficient than particle filters. Combined with an effective pre-processing procedure, it is easily considered as a good trade off between the two tracking approaches. It is selected as the tracking approach for this research.

2.2 Articulated Object Recognition

Articulated object recognition attempts to identify the shape of articulated objects. The two most common methods for articulated shape recognition are techniques that use machine learning and those that use template matching.

Machine learning methods use a trained classifier to determine the shape of the object. The classifier is trained on salient features extracted from a large database of example shape images, along with the shape class to which each image belongs. Given the same features from a query image, the classifier determines the shape class to which the query image belongs. There are a variety of machine learning techniques, each with strengths and weaknesses. Two popular techniques are Support Vector Machines (SVMs) and Neural Networks (NNs). The salient features that can be used to classify shapes are numerous and varied.

Template matching methods match a query image of the articulated object with a database of pre-determined example shape images. A popular method used to carry out the match is determining the distance to the edges of the object, which is also referred to as the segmented silhouettes method [76].

In the case of both methods, a robust pre-processing procedure is required to normalize the extracted object for variations in scale and rotation.

This section is sub-divided into two subsections discussing related research that has made use of each object recognition method in the context of hand shape recognition. Section 2.2.1 discusses machine learning methods and Section 2.2.2 deals with template matching methods. The findings from this section are summarized in Section 2.2.3.

2.2.1 Machine Learning Methods

Sato *et al.* [97] used neural networks (NNs) to recognize six distinct hand shapes. A normalization procedure was used to reduce the effects of variations in rotation and scale of hand images. This procedure follows.

After the hand region is located using hand tracking techniques, the translation is applied to the hand image such that the center of mass of the image aligns with the center of the image. A rotation is applied to the resulting image such that the major axis of the image is aligned with either the horizontal or vertical axis. The image is scaled down to a resolution of 12×12 pixels. Down-sampling the image reduces the effects of small amounts of noise and the computation time associated with larger feature sets.

The normalized image resulting from this procedure is taken as a feature vector. It was used to train the NN classifier on six hand shapes. Figure 2.4 illustrates the extraction and normalization procedure described. The system was shown to recognize the six hand shapes at an accuracy of 85% at a real-time speed of 20 FPS on a Pentium 2 with a 450 MHz processor.

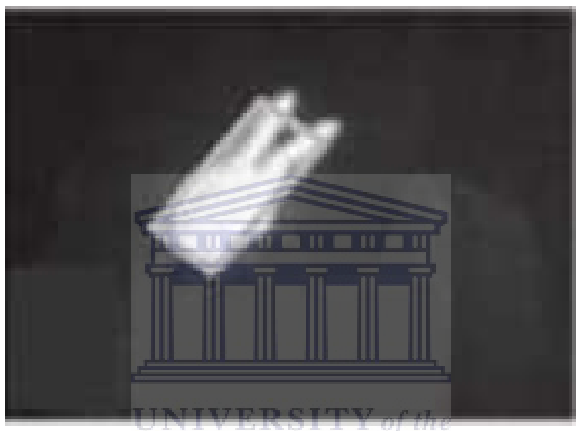
Another approach that also uses NNs, although in a different manner, is proposed in [23]. A novel technique is proposed that uses the NN to simulate the spreading out of a gas within the extracted hand region, thereby identifying its shape. The technique uses a static skin filter in the YCbCr colour space to identify and segment the hands. No further image normalization procedure is applied.

The technique is called the Self-Growing Self-Organizing Neural Network Gas. The technique operates by initially placing a few neurons – typically one or two – within the hand region. The NN then grows outwards to occupy the entire hand region. This procedure is illustrated in Figure 2.5 which illustrates the initial,intermediate and final configurations of the neural gas on a hand shape image. Finally, a likelihood function classifier is used to determine the hand shape based on the neural gas configuration.

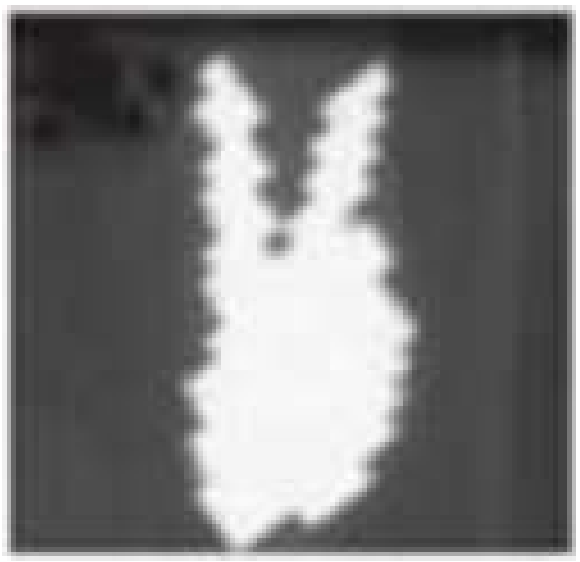
The system was trained to recognize 31 hand shapes. The system was tested on 180 hand shape images and was shown to achieve a 90.4% accuracy. However, the system was found to operate at less than 1 FPS, far less than real-time, on a PC with a 3GHz CPU. It is also not robust to varied skin colours or lighting conditions.

Another approach to hand shape recognition is proposed in [68]. This method makes use of SVMs to recognize 10 hand shapes. A robust normalization procedure is proposed and applied. The morphological operations Close, Open, Dilate and Erode, in that order, are applied once to the the hand image obtained from the hand tracking procedure. This procedure eliminates small isolated clusters of noise from the image. Connected Component Analysis (CCA) is used to identify the connected contours within the image. The hand is taken to be the largest contour. All other contours are eliminated. A computational geometry algorithm [46] is used to compute an oriented minimum bounding box around the contour of the hand. The principal axis of the box is aligned with the vertical axis of the image, thus ensuring that all hand images are upright and rotation

(a) Extracted hand region

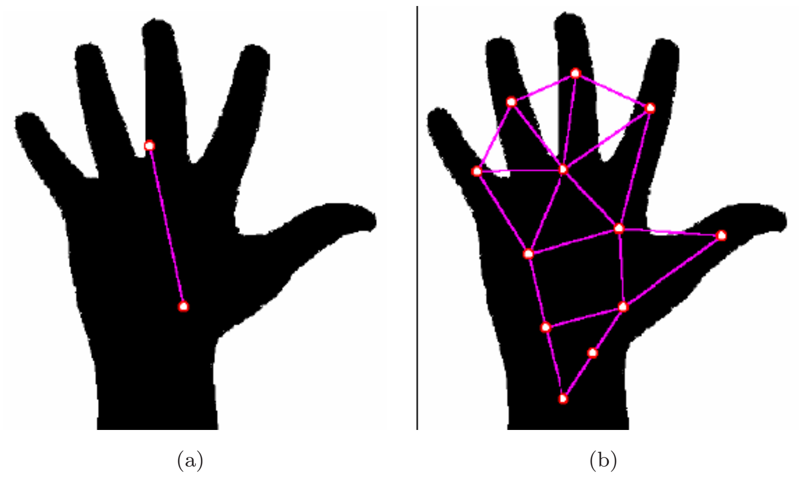


(b) Approximation with rectangle



(c) Normalization

FIGURE 2.4: An illustration of the extraction and normalization procedure of Sato *et al.* [97].



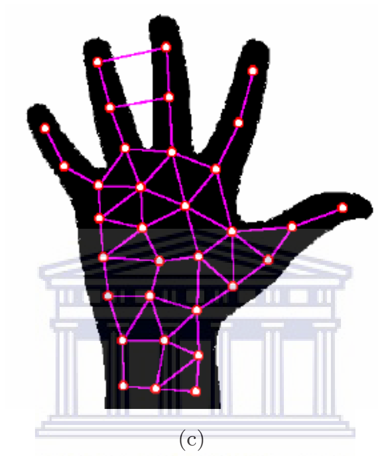


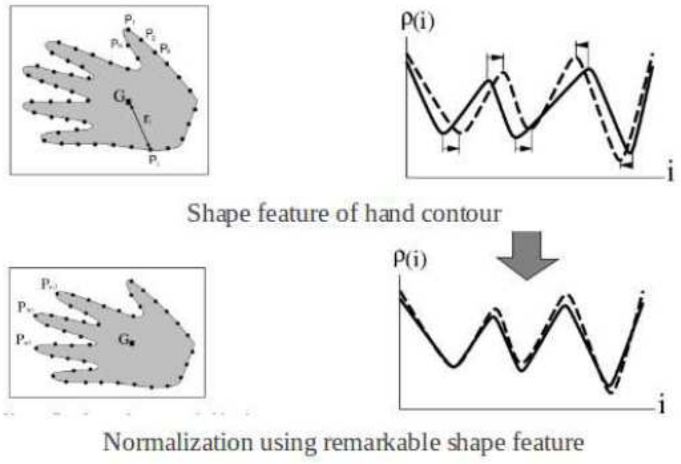
FIGURE 2.5: An illustration of the Self-Growing Self-Organizing Neural Network Gas technique [23].

invariant. The image is scaled down to a resolution of 20 × 30 pixels. The resulting image is taken as a feature vector.

A SVM was trained to recognize 10 hand shapes. It was shown to achieve an accuracy of 88.5% and operate at a real-time speed of 25 FPS on a MacBook Pro with a 2.5 GHz Intel Core 2 Duo processor and 4 GB memory. It was also shown to be invariant to skin colour and robust to complex backgrounds.

2.2.2 Template Matching Methods

Shimada *et al.* created a hand shape recognition system that used the segmented silhouettes method to characterize hand shapes [106]. 256 nodes are placed on the contours of the binary silhouette of the hand. The center of mass of the hand region is computed. The distance of the center of mass to each point along the hand contour is plotted on a graph illustrated in Figure 2.6. A unique and distinct distribution is obtained for each hand shape and is used to determine the hand shape.



P(i)
i
Shape feature of hand contour
P(i)
i
G
Normalization using remarkable shape feature

Figure 2.6: An illustration of the hand contour that is plotted on a graph [106].

It is still necessary to normalize the resulting distribution for variations in rotation and scale. Variations in rotation are controlled by shifting the resulting distribution such that it starts at the highest peak. Variations in scale are controlled by applying a normalization function to all distances. It is expected that generating the distribution in a clockwise or anti-clockwise manner will not affect the performance of the algorithm provided one or the other is done consistently. The exact recognition accuracy of the method is unclear from the literature but it is claimed that the method is robust and accurate.

A similar, yet enhanced, method is proposed in [100]. The method uses an alternative distance computation to the previous approach. As with the previous approach, a predetermined number of nodes are placed on the hand contour. For each node, the normal to the tangent of the hand contour at that point is computed. A line is drawn from the point within the hand region, in the direction of the normal to the hand contour on the other side of the point. It is this distance that is used to compute the characterizing distribution. The system was shown to be able to recognize 13 hand shapes at a real-time speed of 25 FPS on a video sequence at 640×480 pixels. The exact accuracy of the technique is unclear.

2.2.3 Summary of Articulated Object Recognition

Template matching methods are purely data-driven approaches. This reduces the complexity and computational cost of these approaches. It was demonstrated that these approaches generally perform at real-time speeds. However, it is expected that such approaches either require special hardware configurations or specific constraints on the part of hand images to perform at a high accuracy [100, 106].

Machine learning methods, on the other hand, provide a good balance between computational cost and complexity, and accuracy. In general, they are more robust than template matching methods. By providing a large training set of images in varied conditions, it is possible to cater for numerous conditions. Some approaches, such as the neural gas (SGONG) mentioned earlier, can be complex and may not perform at real-time. However, this is case-specific rather than a general rule with machine learning methods. It was mentioned that Li's [68] and Sato *et al.*'s [97] methodologies operate at real-time with a very high accuracy.

As such, machine learning methods are selected for use in the hand shape recognition strategy of this research. Specifically, Li's work is most suitable in terms of complexity, robustness, and computational speed. Li's methodology will be used to recognize hand shapes, to be applied to gesture recognition.

2.3 Sign Language Gesture Recognition

A variety of research projects have focused on gesture recognition. Of these, some are in the context of generic gesture recognition, as applied, for example, to Human-Computer Interaction, and some in the context of sign language recognition. This section discusses such research, with the greatest focus on those studies that have applied gesture recognition in the context of sign language recognition.

The research can broadly be divided into two groups according to the scope of recognition and the nature of the input expected: those that carry out continuous gesture recognition and those that recognize only a single gesture at a time – single-word gesture recognition systems.

Continuous gesture recognition systems expect a sequence of gestures as input and require a strategy, implicit or explicit, to segment sequences of gestures into single gestures. Such systems then optionally use a grammatical model to infer the meaning of the entire gesture sequence. Continuous gesture recognition systems can further be sub-divided into two groups based on the aim of recognition and the gesture sequence segmentation strategy employed: isolated gesture recognition systems and grammar-based gesture recognition systems.

Isolated gesture recognition systems attempt to explicitly segment the input into individual gestures, or isolated gestures, and subsequently recognize each gesture individually. Once the isolated gestures have been determined, a language model may be used to infer the meaning of the entire gesture sequence. Grammar-based gesture recognition systems, on the other hand, generally make use of HMMs coupled with a language model to implicitly and automatically segment a gesture sequence by characterizing the sequence in terms of specific features, with the aim of recognizing the meaning of the entire gesture sequence. Therefore, the entire basis of grammar-based models is a grammatical model of the underlying sign language with the aim of recognition to infer the meaning of the gesture sequence as a whole. The recognition of isolated gestures is implicit. Grammar-based gesture recognition is outside the focus of this research since a grammatical model of SASL is currently unavailable. Only isolated gesture recognition systems are discussed.

Single-word gesture recognition systems assume that the input consists of a single gesture and attempt to infer the meaning of that gesture. Such systems do not have and do not need a gesture sequence segmentation strategy. Such systems can be further subdivided into two groups, according to the use of either a single image or a sequence of images of the hands from which to extract features: static gesture recognition systems and dynamic gesture recognition systems.

Static gesture recognition systems use only a single image containing the hand or hands to recognize a set of gestures. Common amongst such systems is the use of the shape of the hands, possibly coupled with other features, to characterize gestures. Dynamic gesture recognition systems extract features from an entire image sequence of a gesture, usually motion coupled with other features, to characterize and recognize a set of gestures.

The following subsections discuss a number of prominent research studies in these groups. Section 2.3.1 discusses isolated gesture recognition systems. Section 2.3.2 details static gesture recognition systems. Finally, Section 2.3.3 explains the dynamic gesture recognition systems. Section 2.3.4 summarizes these systems and draws a conclusion towards the system proposed in this research.

2.3.1 Isolated Gesture Recognition Systems

Liang and Ouhyoung [69] developed a large-vocabulary Taiwanese Sign Language (TSL) recognition system using the a motion thresholding model to segment gesture sequences into isolated gestures and Hidden Markov Models to recognize isolated gestures and group isolated gestures into sentences. The system uses a DataGlove worn by the user of the system to seamlessly capture four sign language parameters: hand location, orientation, motion and shape.

A motion thresholding model is used to segment gesture sequences into isolated gestures by considering hand motion that is below an empirically determined threshold to be a hold and subsequent motion above a threshold taken to be a movement. A hold in between two movements is considered as an isolated gesture characterized by the five SL parameters.

For each isolated gesture, a feature vector is constructed consisting of: the flexion of 10 finger joints of the hand; the roll, azimuth and elevation of the palm; and information pertaining to the motion trajectory leading up to the hold in the gesture. The feature vector is used as input to Hidden Markov Models (HMMs), although no further details on the HMMs are provided. A TSL model is used to determine the sentence which matches the sequence of recognized gestures with the highest probability. The system achieved an isolated gesture recognition accuracy of 89.5% on a vocabulary of 250 gestures and short sentence recognition accuracy of 70.4% on 108 short sentences consisting of, on average, 3 gestures per sentence. It is stated that the system is signer-dependent.

The gesture segmentation strategy used in this study – the motion thresholding model – is significant since it is adapted for use in the system proposed in this research.

Chen *et al.* [48] developed a hand gesture recognition system by using Haar-like features and Adaboost to recognize gestures. The system recognizes four hand gestures using only hand shape as a gesture descriptor. The system requires input in the form of an image containing the hand of the signer in an upright orientation. The background can be complex. The Viola-Jones object detection algorithm, that uses a combination of Haar-like features and Adaboost, is used to recognize four hand gestures. Four separate classifiers were trained on 480, 412, 400 and 420 image samples at different scales of the four hand gestures “two fingers”, “palm”, “fist” and “little finger”, respectively. An additional 500 random images not containing the hand were used as negative examples to each of the classifiers.

The system was tested on 100 unseen images of each hand gesture under similar conditions to the training set. The system achieved an average accuracy of 87% across the four gestures. The use of Haar-like features coupled with Adaboost can achieve fast computation times but training such a system on a large vocabulary can prove challenging due to the large number of images required and the complexity of the training process.

A small grammar model was used to group gestures into “sentence” units, with only three sentences defined in the model and only two specific consecutive gestures forming a sentence. These sentences included: “Grasp” consisting of a “Palm” and “Fist” sequence; “Quote” consisting of a “Two fingers” and “Fist” sequence; and “J” consisting of “little finger” and “Fist” sequence. A stochastic context-free grammar was used to segment gesture sequences into sentence units by determining the highest probability sentence unit that matched the observed gesture sequence. The sentence recognition accuracy was not tested.

2.3.2 Static Gesture Recognition Systems

Mapari and Kharat [72] developed a system which recognizes American Sign Language (ASL) focusing 10 ASL finger spelling gestures. The input to the system is carefully controlled and it is stated that the camera is positioned such that the hand, and only the hand, is in the center of the frame. The background is simple and static. Figure 2.7 depicts an example hand image used as input to the system. As such, the system only makes use of a static skin model to highlight the hand in the frame. In order to facilitate this, the input image is converted from the RGB colour space to the L*a*b* colour space.

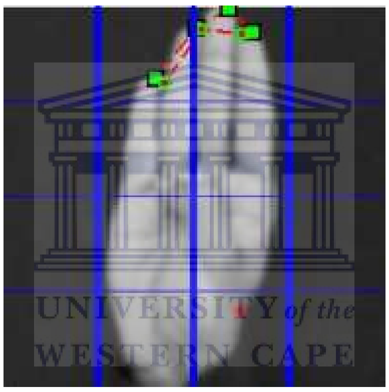


FIGURE 2.7: The hand image that serves as input to Mapari and Kharat's system [72].

Thereafter, the image is cropped and resized. A 4×4 grid is overlayed onto the image, with each grid cell of size 16×16 pixels. Peaks and valleys in the hand contour of each grid cell are detected and combined into a feature vector. The system was trained and tested on images collected from 20 subjects. It is stated that training was carried out using all of the data, and testing was carried out on a 20% subset of the data. A varied number of images was used to test the recognition accuracy of each hand shape, ranging from 17 to 19 images. The system achieved a recognition accuracy of 100%.

Kulkarni and Lokhande [66] presented a similar system, with an approach to the recognition of 26 ASL finger spelling letters. The approach also sets severe limitations on the input. The input consists of images containing an upright hand performing a finger spelling hand gesture on a simple and static background. The images were manually resized to achieve normalized results. Example input images are illustrated in Figure 2.8.

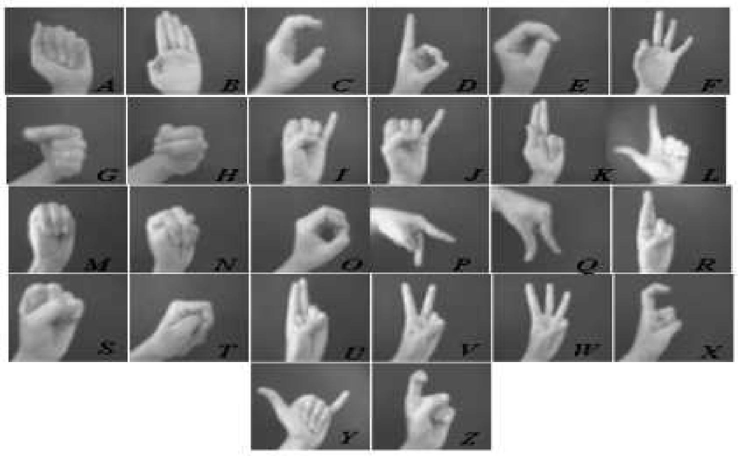


FIGURE 2.8: An example of an input image to Kulkarni's system [66].

The feature extraction process starts by converting the input image from RGB to grayscale. Canny edge detection is carried out to mark the points in the image where the intensity varies by a large amount. Thereafter, three feature vector types are produced by applying three operations on the edge detected image: computation of a histogram of the image; applying a Hough transformation; and Otsu thresholding. The exact construct of the three feature vector types is not detailed in the literature. The resulting feature vectors are fed to a feed-forward back propagation neural network with supervised learning, consisting of 256 instances as input vector and 214 output neurons in the output layer. Eight sample images were captured for each of the 26 ASL finger spelling gestures using eight subjects. Of these, five samples per subject were used for training and three samples, for testing. The researchers state that the Otsu thresholding feature extraction method achieves the highest recognition rate and only results of this method are provided. The method achieved an average recognition rate of 92.78% across all signs and subjects.

Grobel and Assan [10] developed an isolated Netherlands Sign Language (NSL) recognition system which uses coloured gloves to simplify the feature extraction process. Additionally, constraints on the background and clothing of the signer are used to further simplify the segmentation problem. This made it possible to recognize a large vocabulary of 262 NSL gestures at a high accuracy. The user is required to wear two colour-coded gloves. It is stated that the right hand is the dominant hand used to express NSL gestures. Therefore, the glove worn by the right hand has seven colours marking seven regions: the fingers, the palm and the back of the hand. The glove worn by the left hand is a single colour.

A camera captures gestures. Features are extracted from the images to indirectly represent the location, orientation and shape of the hands. A greater number of features are extracted from the dominant hand than the non-dominant hand. The features extracted from the dominant hand are: the x-coordinate of the hand relative to the spine and y-coordinate relative to the right shoulder; the distance of centers-of-gravity (COG) of all colour regions relative to each other; the size of all colour regions; and the angles of each finger. The features extracted from the non-dominant hand are: the x-coordinate of the hand relative to the spine and y-coordinate relative to the right shoulder; and the size of all colour regions. A final feature extracted is the distance between the COG of the dominant and non-dominant hand.

The resulting feature vector is used as input to a HMM which predicts the NSL gesture. The system was trained and tested on two signers and achieved a recognition accuracy of 91.1%. The system was trained 15 samples of each of the 262 NSL gestures performed by 2 signers on a simple and static background wearing plain clothing. The same signers performed the 262 NSL gestures 9 more times to produce a test set. The system was shown to achieve an accuracy of 91.3% across all gestures.

Bauer *et al.* [13] proposed a system that uses this same methodology to recognize 97 German Sign Language signs at an accuracy of 91.7%.

Phu and Tay [82] proposed a static gesture recognition system that recognizes a set of static hand gestures. The hand segmentation strategy is based on skin detection using a static range of the hue and saturation values of the human skin. As such, specific and limited skin colours and a simple and static background is assumed. Edge detection followed by connected component analysis (CCA) is applied to the resulting skin detected image to obtain the contours representing skin regions.

A multilayer perceptron (MLP) is used for classification. An experiment was carried out to compare two types of activation functions, namely, the tangent hyperbolic and sigmoid activation function. It was found that the tangent hyperbolic activation function obtained better results and was used in the MLP backward propagation network. In a second experiment, the gesture recognition accuracy of the MLP was evaluated with a set of web cam captured samples and a set of grey scaled samples. The Triesch's dataset which contains 10 classes of static gestures was used. Six sessions of training and nine sessions of evaluation were conducted. The system obtained an accuracy of 92.97% using web cam captured samples and an accuracy of 50.82% for the gray-scaled Triesch's dataset.

2.3.3 Dynamic Gesture Recognition Systems

Elzemain *et al.* [40] developed a dynamic Arabic number hand gesture recognition system. The system recognizes alphabet characters (A–Z) and numbers (0–9) performed in real-time, captured by a stereo-colour camera. Only hand motion is considered. The stereo camera is used to obtain a colour image and depth map of the scene containing the signer, possibly on a complex background. A static skin range is coupled with a depth constraint – the hands will always be closer to the camera than the face – to locate the hand blobs in the first frame. From this point on, a combination of Mean-Shift and Kalman filters is used to track the hands in skin detected images with the depth constraint applied. An accurate motion trajectory of the hand is thus obtained.

The motion trajectory is quantized by splitting the trajectory into segments according to the direction of motion. 18 direction classes are defined in 20 degree increments. The motion trajectory is described in terms of segments in each of the 18 directions. HMMs are used to classify the motion trajectory as being one of the alphabet characters and numbers. The Baum-Welch algorithm was used to train the HMM. 20 videos samples were used for training of each gesture and 10 were used for testing of each gesture. The system achieved a recognition rate of 98.33% across all gestures.

Hong *et al.* [52] proposed a gesture modeling and recognition approach using finite state machines. The approach recognizes, as a base-case, four hand gestures "left hand wave", "right hand wave", "drawing circle" and "drawing figure eight". A single camera is used to capture an image sequence. The face and hands are obtained by means of a dynamic skin model that uses a histogram of the face to model skin pixels of the signer, applied to each frame of the sequence. It is stated that the result of this procedure is "noisy".

Each gesture is defined as an ordered sequence of states in spatial-temporal space. Therefore, gestures are modeled as a sequence of states in spatial-temporal space. The gesture recognition model is represented by a finite state machine. The system learns the distribution of data without temporal information through dynamic k-means clustering. The result of this phase provides support for data segmentation and alignment. The temporal information is then learned from the aligned data segments. The system updates the spatial information of the gesture. In the finite state machine, each state sequence recognizes a single gesture. No experimental results are provided in the literature, but it is stated that a satisfactory gesture recognition accuracy is obtained.

Vafadar and Behrad [119] also proposed a hand gesture recognition system using the spatio-temporal volumes to characterize gestures. The system places strict constraints of the input data similar to those of Kulkarni and Lokhande [66] and others previously mentioned to simplify the feature extraction process. A k-means clustering algorithm is used to learn and model skin pixels and highlight these pixels in images. Connected Component Analysis is used to highlight the largest blob which is the hand and segment it for further processing.

Motion and shape features are extracted for gesture recognition. Motion is represented as a sequence of coordinates representing the center of the hand in each frame in the sequence. A normalization procedure is applied to avoid length discrepancies associated with varied velocity motions of each gesture. Shape features are represented by dividing the hand contour into 8 segments, centered at the center of mass and segmenting in a clockwise fashion. The points in each segment are used to represent the shape.

Three different classifiers, k-nearest neighbors (kNN), learning vector quantization and back propagation neural networks were used for feature vector classification for gesture recognition. The training and testing sets were not detailed. It was found that k-NN classifier achieved the highest recognition accuracy of 99.98% for noiseless, and 92.08% for noisy data.

2.3.4 Summary of Gesture Recognition Systems

It is clear from the literature that sign language gesture recognition is a complex problem. One significant problem that has been the theme across all of the studies is the difficulty in feature extraction. The majority of studies have either used complex and expensive equipment such as DataGloves to accurately extract sign language parameter features or set severe constraints on the scene and signer to simplify the feature extraction process. Such constraints include a simple and static background, a single skin colour, normalized position and orientation of the hand(s) in the frame, expecting only a hand in the frame, the use of custom-made colour-coded gloves, signer-dependence, and several other restricting constraints. In all cases, the user's freedom is restricted and the user is not able to interact with the system naturally. As an addition, there is no other research, to our knowledge, that focuses on SASL recognition.

Isolated gesture recognition systems were shown to be encouraging as a basis of a system that can eventually recognize entire sentences. Liang and Ouhyoung's research was of special note. The gesture sequence segmentation strategy employed was a motion thresholding model in which motion below a pre-determined threshold is considered a hold and subsequent motion above a related threshold is considered a movement. The hold in between two movement segments is considered as an isolated gesture characterized by the five SL parameters. This segmentation model is employed in the proposed system. The results of Liang and Ouhyoung's research are also encouraging and indicate that, once a SASL corpus is made available, the proposed isolated gesture recognition system can be extended to recognize entire SASL sentences with a high accuracy. Unlike this research, however, the proposed system uses SVMs to recognize individual gestures.

The majority of static gesture recognition systems only carry out SL finger spelling recognition. Finger spelling recognition only requires hand shape as a parameter and such systems generally extract only this parameter. In SASL, gestures other than finger spelling gestures are performed in specific locations. Therefore, it is typical to find two gestures performed with the same hand shape in different locations or vice versa. Such systems, therefore, do not scale to general gesture recognition. Dynamic gesture recognition systems were shown to be more flexible than static gesture recognition system, making use of a variety of features to recognize gestures with complex motions and shapes. However, such systems, once again, set constraints on the input and suffer from a limited vocabulary set.

The proposed system aims to provide a much greater degree of freedom to the user, with a small set of constraints used to keep the problem within the scope of the dissertation. The system is signer-independent, meaning that signers can have varied skin colours and body dimensions. Only a simple web camera is used to capture SASL sequences and the user can stand with his entire body in the frame i.e. it is not required that only the hands appear in the input. The user can also be located on a complex background.

Another significant advantage of the proposed system over those in the literature is the use of SVMs to recognize individual gestures. The majority of research studies have focused on the use of HMMs to recognize gestures given a set of feature observations, with a few making use of artificial neural networks. Compared to HMMs, SVMs are much easier to train [28, 60, 90], and the vocabulary size can be extended easily, given suitable features. Thus, the proposed approach is novel.

2.4 Conclusion

This chapter discussed the related work in the fields of articulated object tracking, articulated object recognition and gesture recognition.

The prominent methods used to track articulated objects were discussed as well as studies that implemented them. Two types of tracking techniques were discussed, namely, model-based and model-free methods. It was found that model-based methods were unsuitable for hand tracking. A look at model-free methods revealed that, all-in-all, such methods were on equal footing, each with a set of strengths and weaknesses. However, CamShift was found to be more applicable to this research since it is very computationally efficient and accurate.

The prominent articulated object recognition methods were also discussed as well as research that applied them. Template matching methods were found to be computationally efficient and with reduced complexity, but required special hardware or restrictive constraints on the input to perform accurately. Machine learning methods were found to provide a very good balance between computational cost, complexity and accuracy. While more complex, they can achieve much a higher accuracy. Li's method was of special note and was shown to be able to recognize hand shapes at a high accuracy in real-time. It was selected for use as the hand shape recognition strategy in the proposed system.

Finally, a detailed survey of gesture recognition systems was presented, with a special focus on sign language gesture recognition systems. It was concluded that the majority of such systems either make use of specialized hardware or a comprehensive set of stringent constraints on input, scene and signer, to achieve accurate feature extraction. In all such cases, the freedom of the signer is severely restricted. It was also demonstrated that the majority of such systems make use of HMMs to achieve gesture recognition. The proposed system makes use of SVMs which are easier to train and the vocabulary can be easily extended. Of special note was Liang and Ouhyoung's research which makes use of a motion thresholding model to segment gesture sequences into isolated gestures. This is used as a gesture sequence segmentation strategy in the proposed system.

Chapter 3

Image Processing For Gesture Recognition

This chapter explains, in detail, the image processing techniques used in the proposed gesture recognition system. This serves as a foundation of understanding for the next chapter which details the proposed gesture recognition system. The chapter is divided into three broad sections. Section 3.1 details the various feature extraction techniques used in the proposed system. Section 3.2 provides a detailed background into classification using Support Vector Machines. The section that follows concludes the chapter.

3.1 Feature Extraction Techniques

3.1.1 Face Detection

Face detection is a specific class of object detection. The task of the algorithm is to find the locations and sizes of all objects in an image that appear to be faces and to extract the face region from a human body. Depending on the specific application, constraints can be placed on the size and number of faces to be located and extracted. The proposed gesture recognition system requires the detection of only one human being in the frame. As such, only the largest true positive face is detected and located.

Many modern image processing applications [18, 68, 95, 121, 122] use face detection as the foundation of the system to:

1. Detect the presence of a human.

2. Normalize the position of the user(s) in an image sequence relative to the position of the face.

3. Serve as a relative measure of the positions of other parts of the human body in the image.

There are currently a wide variety of face detection techniques [58, 103]. The most popular of these techniques are those that are knowledge-based [5, 74], feature-invariant-based [1, 19, 64], template matching-based [41, 78, 124], appearance-based [36, 50, 102] and part-based methods [81, 110]. Of all these techniques, the appearance-based Viola-Jones face detector is the most popular [32, 34]. The technique has been shown to be highly robust, accurate and computationally efficient [18].

The Viola-Jones face detection framework is capable of processing images rapidly while achieving high detection accuracy rates [122]. The ability of the framework to rapidly detect faces clearly distinguishes it from and places it above other face detection techniques. The technique uses a boosted rejection cascade of Haar classifiers to attain a high accuracy face detection framework. The framework consists of four key features [18]:

1. Integral image computation which is used to speed up the Haar-like wavelet detection procedure.

2. The Detection of Haar-like wavelet features in the image.

3. The use of a statistical boosting algorithm based on AdaBoost that characterizes nodes.

4. The organization of weak classifier nodes as a rejection cascade.

These four stages are discussed in the following subsections.

3.1.1.1 Computation of Haar-like Features

Figure 3.1 depicts four haar-like features used in face detection. It can be seen that they are rectangles that contain high and low intervals which are vertically or horizontally aligned. In each case, the dark and light regions of the feature are of the same size. It can also be seen that they are of three types, according to the number of sub-rectangles that they contain: two, three and four. The actual rectangle combinations used for visual object detection are not true Haar wavelets. Instead, they contain rectangle

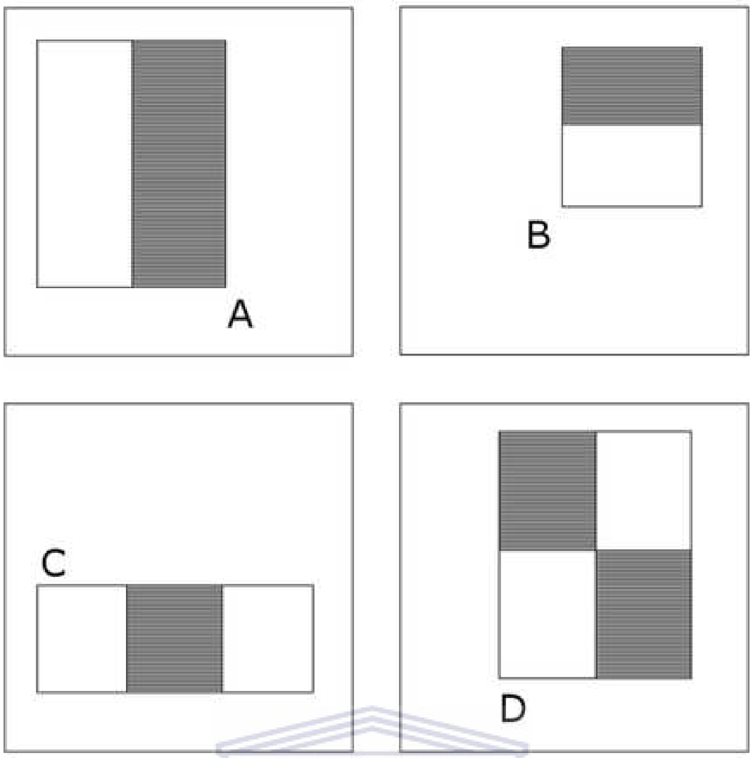


FIGURE 3.1: Four types of haar features used in the Viola-Jones detection framework [122].

combinations better suited to visual recognition tasks. Because of that difference, these features are called Haar features, or Haar-like features, rather than Haar wavelets [122].

Each haar-like feature, at various scales, is scanned over the image. The difference between the sum of all pixels in the light region and those in the dark region is computed. If this value is higher than a pre-determined threshold, then it is said that that specific haar-like feature is present in the image. It is obvious that carrying out this computation over multiple scales and locations can be computationally expensive.

Viola and Jones proposed a technique called the Integral Image representation which provides a means of effectively speeding up this procedure and detecting the presence of haar-like features at numerous locations and scales. The next sub-section explains the Integral Image representation.

3.1.1.2 Integral Image Representation

The integral image at location (x, y) contains the sum of all pixels above and to the left of (x, y). In general, the value of the Integral Image Representation I' at each (x, y), given by $I'(x, y)$, of an original image I is given by the following equation [122]:

$$I'(x,y) = \sum_{x' \leq x, y' \leq y} I(x', y') \tag{3.1}$$

Alternatively, this can be expressed as a pair of recurrences given by:

$$\begin{aligned} S(x,y) &= S(x, y-1) + I(x,y) \\ I(x,y) &= I(x-1, y) + S(x,y) \end{aligned} \tag{3.2}$$

where $S(x,y)$ is the cumulative row sum at pixel (x,y). The integral image can be computed in one pass over the original image. At this stage, it is possible to compute all three types of haar-like features in constant time. Figure 3.2 taken from [122] illustrates that the sum of pixels within rectangle D can be computed by subtracting the value of the integral image at locations 2 and 3 from the value at location 4, but then adding back the value at location 1 to compensate for the overlap between rectangles $(A + B)$ summarized at point 2 and rectangles $(A + C)$ summarized at point 3.

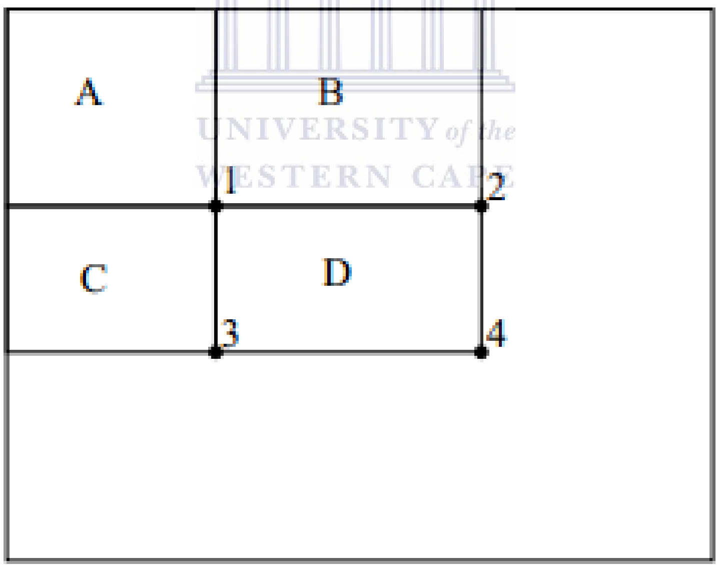


FIGURE 3.2: Computation of the integral image [122].

Using the Integral Image Representation makes it possible to compute two-rectangle haar-like features using six references to the integral image, three-rectangle features in eight references and four-rectangle features in nine references to the image.

3.1.1.3 Using AdaBoost to Generate a Rejection Cascade of Weak Classifiers

The Viola-Jones face detection method uses the Adaboost training algorithm to construct a rejection cascade. The algorithm selects uses a set of optimal features and trains a series of weak classifiers based on those features. Weak classifiers are classifiers that are only slightly more accurate than random guessing and so easy to create. They more accurately recognize true negative samples than true positive samples.

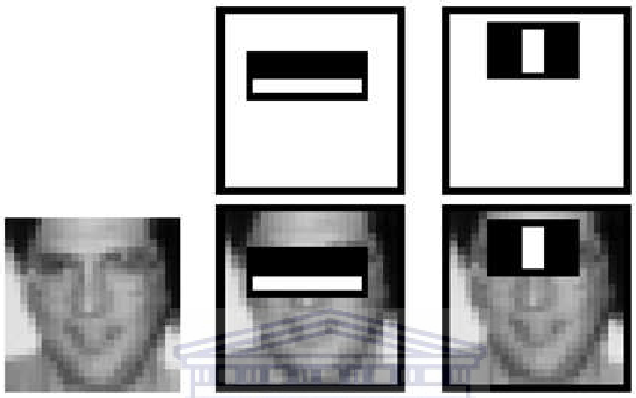


FIGURE 3.3: Features selected by AdaBoost for face detection [122].

Adaboost generates a rejection cascade that places the most relevant classifiers in a degenerate tree structure in which each classifier is assigned a weight according to its accuracy and – with exception to the first node in the cascade – only works on the result from the previous classifier in the cascade. Figure 3.3 is an example of possible features that may be selected by algorithm. It is only said that a face has been detected when the input image has passed through the entire cascade without being rejected. This structure, illustrated in Figure 3.4, has been shown to produce a result which is equivalent to a strong classifier [98]. It is also very computationally efficient since input that is rejected at any node is not processed any further.

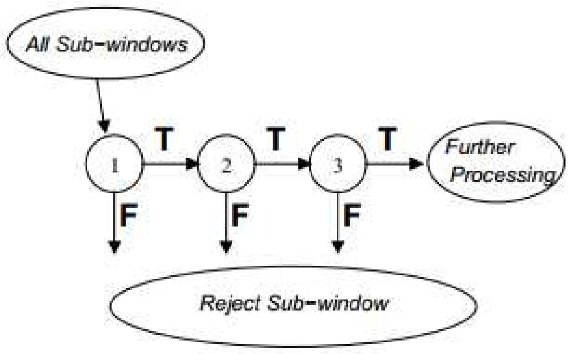


FIGURE 3.4: The detection process for rejecting regions in the image [122].

3.1.1.4 Viola-Jones Face Detection Performance

Viola and Jones tested the speed and accuracy of the Viola-Jones face detector on the MIT+CMU frontal face test set [94] which consists of 507 labeled faces contained in 130 frames. It was shown that the algorithm can detect faces at an accuracy of 93.9%, at a false detection rate of 167. This result is more accurate than any other face detection technique [122]. It was also shown that the algorithm can process about 15 FPS on a 700 MHz Pentium III PC, which is real-time and much faster than other known techniques.

3.1.2 Skin Detection

Skin detection is the process of highlighting regions in an image which have a high probability of being skin pixels. All other regions are de-emphasized. This procedure is fundamental to numerous applications that perform hand detection, tracking and recognition [3, 18, 96].

The technique is effective in detecting specific body parts since the colour of the skin is distinct from most objects. It can thus be used to distinctively detect and track these body parts. Skin detection can be a complicated problem. Various races have varied skin tones and this may be more so in diverse regions of the world than regions that are ethnically uniform. Various other variables such as lighting, skin exposure to the camera etc, can all add further complexity to the problem. In general, skin filter is created in the following steps [61, 80, 120]:

1. Selection of a suitable colour space that can, among other things, help control illumination changes.

2. Using a robust and adaptive skin model to classify pixels in the image according to a skin selection criterion, possibly into a binary representation.

There are a wide variety of colour spaces. However, many of these colour spaces have common attributes or are equivalent to each other in essence and are not all discussed. Only the most frequently used colour spaces are discussed below. The colour spaces that are discussed in this section, from Section 3.1.2.1 to Section 3.1.2.5 are: the RGB colour space, the normalized RGB colour space, the HSV colour space, the YCbCr colour space, and the TSL colour space. A suitable colour space for skin detection is selected in Section 3.1.2.6. The skin model is discussed in section 3.1.2.7 which discusses the construction of the skin model used for our system.

3.1.2.1 RGB Colour Space

The RGB (Red-Green-Blue) colour space is defined by three chromaticities which are those of the red, green, and blue additive primaries in colour. A particular colour is produced by super-imposing three light beams of specific shades of each colour. The combinations of these colours – also referred to as "channels" – can produce any colour.

The colour representation in this colour space may be thought of as a cube in which the three axes that are perpendicular to each other represent the amount of red, green, and blue in the particular colour. Each pixel can be represented as an (R, G, B) triplet.

Linear and non-linear transformations are applied to this colour space to derive other colour spaces and, as such, it may be considered as a fundamental colour space.

A significant drawback of this colour space is that it is not perceptually uniform. Any given (R, G, B) triplet may be reproduced or detected differently by different devices [120, 126], depending on the physical hardware used to generate the light beams. Additionally, the luminance and chrominance of the colour representation are tightly connected. As such, changes in illumination of a pixel directly affect the detected colour of the pixel.

3.1.2.2 Normalized RGB Colour Space

The following normalization formula is applied to the an (R, G, B) triplet in the default RGB colour space to obtain a Normalized RGB triplet (r, g, b):

$$r = \frac{R}{R+G+B} \tag{3.3a}$$

$$g = \frac{R}{R+G+B} \tag{3.3b}$$

$$b = \frac{R}{R+G+B} \tag{3.3c}$$

The sum of the normalized pixel values is 1, expressed mathematically as:

$$r + g + b = 1 \tag{3.4}$$

Seeing as the sum is constant and known, two colour components are sufficient in the representation of any colour in this colour space. The third component can be obtained, if required, by substituting the known values into and solving Equation 3.4. As a result, the space dimensionality of this colour space is reduced [120], which has implications on the size of images.

The normalized RGB colour space is less sensitive to illumination changes than the RGB colour space.

3.1.2.3 HSV Colour Space

The HSV (Hue-Saturation-Value) colour space is obtained by applying a non-linear transformation to the normalized RGB colour space. It is also referred to as the HSI (Hue, Saturation, Intensity) and the HSL (Hue, Saturation, Lightness) colour space. Using the normalized RGB components of a colour r, g and b, respectively, the Hue-Saturation-Value triplet (H, S, V) can be obtained by applying the following non-linear transformation [120]:

$$V = max_{r,g,b} \tag{3.5a}$$

$$S = \frac{max_{r,g,b} - min_{r,g,b}}{V} \tag{3.5b}$$

$$H = \begin{cases} \frac{g-b}{6(max_{r,g,b} - min_{r,g,b})}, & \text{if } V = r \\ \frac{2-r+b}{6(max_{r,g,b} - min_{r,g,b})}, & \text{if } V = g \\ \frac{4-g+r}{6(max_{r,g,b} - min_{r,g,b})}, & \text{if } V = b \end{cases} \tag{3.5c}$$

where $max_{r,g,b}$ and $min_{r,g,b}$ are the maximum and minimum of the normalized red, green and blue RGB values, respectively.

A significant feature of this transformation is that it separates the previously infused chrominance and luminance characteristics of a colour. The H and S components define the chrominance of a colour, while the V component specifies the brightness or intensity. It is an ideal colour space for applications that require robustness to changes in illumination of the target environment.

3.1.2.4 YCbCr Colour Space

The YCbCr (Luminance-Blue Chrominance-Red Chrominance) colour space is obtained by applying a linear transformation to the default RGB colour space. It is a colour space that is commonly used in the representation of digital video [55, 113]. The YCbCr colour space is obtained by applying the following formula to a (R, G, B) triplet:

$$Y = 0.299R + 0.587G + 0.114B \tag{3.6a}$$

$$Cr = R - Y \tag{3.6b}$$

$$Cb = B - Y \tag{3.6c}$$

where Y is the luminance component, and Cb and Cr are the chrominance components. The luminance information is stored as a single component Y, and the chrominance information is stored as two colour-difference components Cb and Cr. The Cb and Cr components represent the difference between the blue and red components, respectively, and the intensity.

This transformation also achieves a separation of the luminance and chrominance components of colour and has similar advantages to the HSV colour space in this respect.

3.1.2.5 TSL Colour Space

The TSL (Tint-Saturation-Lightness) colour space is obtained by applying a transformation to the normalized RGB colour space. It is commonly used in face detection [114, 115]. Using the normalized red, green and blue RGB components of a colour r, g and b, respectively, the Tint-Saturation-Lightness triplet (T, S, L) can be obtained by applying the following non-linear transformation [120]:

$$T = \begin{cases} \frac{\arctan(\frac{r'}{g'})}{2\pi} + \frac{1}{4}, & \text{if } g' > 0 \\ \frac{\arctan(\frac{r'}{g'})}{2\pi} + \frac{3}{4}, & \text{if } g' < 0 \\ 0, & \text{if } g' = 0 \end{cases} \tag{3.7a}$$

$$S = \sqrt{\frac{9(r'^2 + g'^2}{5})} \tag{3.7b}$$

$$L = 0.299R + 0.587G + 0.114B \tag{3.7c}$$

where r' and g' are the variants of the normalized red and green pixels expressed as:

$$r' = r - \frac{1}{3} \tag{3.8a}$$

$$g' = g - \frac{1}{3} \tag{3.8b}$$

3.1.2.6 Selection of an Optimal Colour Space for Skin Detection

Several studies have aimed to determine the suitability of various colour spaces to the task of skin detection [30, 31, 38, 61, 107, 108, 120, 126]. The questions that were specifically posed were:

1. Does the separation of luminance and chrominance aid skin detection?
2. Is there a significant difference in skin detection accuracy using different colour spaces?

These studies can be categorized into two large groups. One category of studies indicate that separating luminance and chrominance has no effect on skin detection, and that no colour space provides any significant advantage over any other colour space in this regard [61, 107].

The second category of studies contrasts this view and indicates that colour space selection is key to an effective skin detection strategy, at the heart of which is the separation of the luminance and chrominance of colour, and this appears to be the majority view [30, 31, 38, 107, 108, 120, 126]. These researchers have a specific inclination towards the Hue channel of the HSV colour space which they indicate can represent the skin very well. In all cases, however, the researchers indicate that the HSV colour space must be coupled with a robust pre-processing procedure to achieve effective skin detection results.

As such, the use of the HSV colour space in the proposed implementation is justified.

3.1.2.7 Dynamic Skin Segmentation Using Histogram Back Projection

The use of a static skin filter comprising of a constant pixel thresholding technique to detect skin pixels in an image is a popular technique. The drawback to this technique is that it is non-adaptive and may not be as effective in applications which are targeted at a racially diverse user group.

Achmed proposed an adaptive skin segmentation strategy that uses the colour distribution of regions of the face to determine a dynamic skin model. The model can be continuously updated with the skin colour model of the specific user in time to compensate for potential illumination changes [2]. Details of the approach follow.

A 10×10 pixel area in the centre of the facial frame is usually located around the nose. This region represents the skin very well since it is free from obstructions such as facial hair, shadows etc. A histogram of the Hue component of this region is generated. This histogram is, in essence, a colour distribution of the user's skin.

The bin width of this histogram was optimized by Brown [18]. Achmed originally proposed a bin width of 16, but Brown found that a bin width of 8 provides the optimal skin detection result. This improved both the accuracy and speed of the algorithm.

A back-projection of the histogram onto the original image results in a grayscale image that is a skin likelihood distribution of the original image. The likelihood ranges from white (255) to black (0), where white and black are the highest and lowest likelihoods, respectively, that a specific pixel is a skin pixel. Given the colour C of a pixel and the probability $P(C|F)$ of drawing colour C when the pixel is indeed a skin pixel F, the probability $P(F|C)$ that the the pixel is skin coloured is given by:

$$P(F|C) = \frac{P(F)}{P(C)} P(C|F) \tag{3.9}$$

The likelihood distribution is thresholded to obtain a binary image of only skin and non-skin pixel classes. Achmed and Brown [2, 18] determined that applying a threshold of 60 produced optimal skin detection results. The result of applying this skin detection method to the image depicted in Figure 3.5 is depicted in Figure 3.6. Pixels that are displayed in the image which are not of true skin pixels can be eliminated using noise reduction techniques and by combining the strategy with a motion cue explained in subsequent sections.

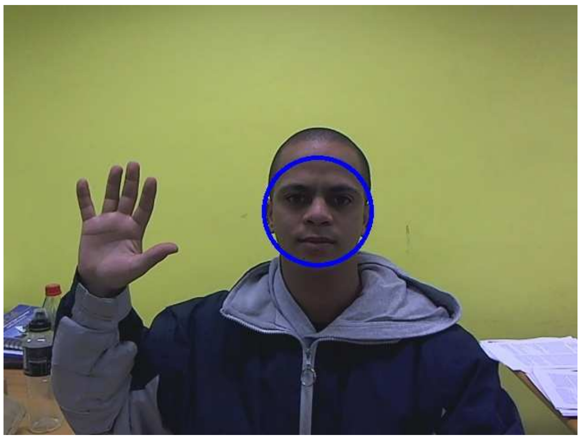


FIGURE 3.5: An image of a face.

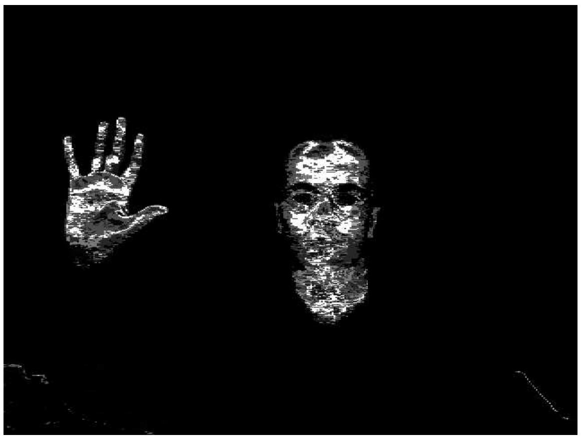


Figure 3.6: The resulting skin image of the face.

3.1.3 Background Subtraction

Background subtraction aims to segment a desired "foreground" from an unwanted "background" in a sequence of images [11, 83, 129]. As such, the background is effectively subtracted from the image, leaving the foreground, hence the name "background subtraction". The foreground in this research is the hands of the user performing sign language gestures.

There is a variety of background subtraction techniques, each with respective strengths and weaknesses. The following requirements may be used to guage the suitability of each background subtraction technique to the needs of this research [25]:

1. The method should be robust to lighting conditions.
2. It should successfully classify moving objects other than the hands as part of the background.

The following subsections, Sections 3.1.3.1 – 3.1.3.3, discuss three common background subtraction techniques. These are: static background subtraction; frame differencing; and Gaussian Mixture Models (GMMs). A comparison of these background subtraction techniques is presented in Section 3.1.3.4.

3.1.3.1 Static Background Subtraction

In static background subtraction, the first frame in a frame sequence is used as a static reference which is assumed to accurately represent the background. The following frames in the sequence are then subtracted from the reference image. Thresholding is carried out to binarize the resulting image into two classes: background pixels and foreground pixels. As such, the technique is a binary classification technique [18].

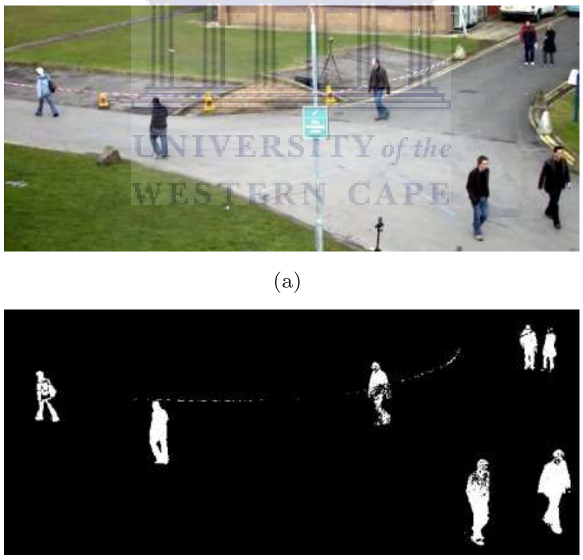
For each pixel $I(i, j)$ at coordinates (i, j) in the current image I, a label l is assigned to a pixel where $l \in 0, 1$, and 0 and 1 represent the background and foreground, respectively. Taking $R(i, j)$ to be the pixel at (i, j) in the reference image R and the threshold to be T_h, a pixel is assigned the label 1 if the following equation is satisfied:

$$|I(i, j) - R(i, j)| > T_h \tag{3.10}$$

The threshold T_h can be determined and optimized empirically [16]. Conversely, the pixel is assigned the label 0 if the following equation is satisfied:

$$|I(i, j) - R(i, j)| \leq T_h \tag{3.11}$$

The resulting binary image will have the foreground in white and the background in black. An illustration of the result of this technique can be seen in Figure 3.7.


(a)

(b)

FIGURE 3.7: An illustration of a static background subtraction technique.

3.1.3.2 Frame Differencing

Frame differencing is an optimized version of the previous technique. Unlike the previous technique, frame differencing dynamically updates its reference image throughout the image sequence. The preceding image to the current image in the sequence is commonly used as the reference image. Other schemes may be proposed as extensions to this basic method, such as using every second frame, average of R frames etc, which are not discussed.

For any pixel $I(i,j)$ at position (i,j) in the current image I, and letting $I_1(i,j)$ be the same pixel in the previous frame, the pixel is labeled as foreground if following equation is satisfied:

$$|I(i,j) - I_1(i,j)| > T_h \tag{3.12}$$

where, again, T_h is an empirically determined and optimized threshold [84]. A pixel is labeled as background if:

$$|I(i,j) - I_1(i,j)| \leq T_h \tag{3.13}$$

3.1.3.3 Gaussian Mixture Models

Gaussian Mixture Models (GMMs) represent background pixels as a mixture of adaptive Gaussians [91, 92, 99]. It is a far more complex technique than the previous two techniques that provides a rich source of location and motion information of moving objects in the frame. A detailed mathematical specification of the modelling of pixels using a mixture of Gaussians, adapted from [18] follows.

Consider an image sequence F formulated as:

$$F = \{I_1, \ldots, I_t\} \tag{3.14}$$

where I_p is the current image at time p and t is the current time. Letting $I(i,j,p)$ be the value of pixel (i,j) at time p, the history of the pixel at (i,j) at current time t across the image sequence can be formulated as:

$$\{I_1(i,j), \ldots, I_t(i,j)\} = I(i,j,p) : 1 \leq p \leq t \tag{3.15}$$

The pixel can be modelled as a mixture of k Gaussian distributions. The probability that a pixel may have a value at I_t at time t can be evaluated using the following formula:

$$P(I_t) = \sum_{x=1}^{k} W_{x,t} \times \eta(I_t, \mu_{x,t}, \sigma_{x,t}) \tag{3.16}$$

where $W_{x,t}$ is the estimated weight of the x-th Gaussian and $\eta(I_t, \mu_{x,t}, \sum_{x,t})$ is the normal distribution of the x-th Gaussian given by:

$$\theta(I_t, \mu_{x,t}, \sigma_{x,t}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\sum_{x,t}|^{\frac{1}{2}}} e^{\frac{-1}{2}(I_t - \mu_{x,t})^T \sum_{x,t}^{-1}(I_t - \mu_{x,t})} \tag{3.17}$$

where $\mu_{x,t}$ is the mean and $\sum_{k,t} = \sigma_{k,t}^2 I$ is the covariance to the k-th Gaussian component and I is the identity matrix. The k distributions are ordered based on the fitness value $\frac{W_{x,t}}{\sigma_{x,t}}$ while the background of the scene is modelled using the first M distributions, where M is estimated as:

$$M = \operatorname{argmin}_m (\sum_x^m W_{x,t} > T_h) \tag{3.18}$$

where T_h is the threshold, which is the minimum segment to the background model. After the background in the image is updated, any pixel found to be more than 2.5 standard deviations away from any of the M distributions the is labeled as foreground. If a match is found between the x-th Gaussian component and the test value, it is updated as:

$$\begin{aligned} W_{x,t} &= W_{x,t-1} \\ \mu_{x,t} &= (1-\rho)\mu_x + \rho I_t \\ \sigma_{x,t}^2 &= (1-\rho)\sigma_{x,t-1}^2 + \rho(I_t - \mu_{x,t})^T(I_t - \mu_{x,t}) \\ \rho &= \alpha\eta(I_t|\mu_k, \sigma_k) \end{aligned} \tag{3.19}$$

where $W_{x,t}$ is the x-th Gaussian component and $\frac{1}{\alpha}$ is defined as the time constant that determines change. If no match is found between the Gaussian component and the test value, it is updated as:

$$\begin{aligned} W_{x,t} &= (1-\alpha)W_{x,t-1} \\ \mu_{x,t} &= \mu_{x,t-1} \\ \sigma_{x,t}^2 &= \sigma_{x,t-1}^2 \end{aligned} \tag{3.20}$$

If no match is found between the test value and any of the test components, the component with the lowest probability is replaced by a new one with a low weight parameter, a high variance, and the current value as its mean. When the Gaussian distributions are evaluated, pixels that do not match are classified as foreground and are grouped using 2D connected component analysis. Connected component analysis is discussed in a subsequent section.

3.1.3.4 Summary and Discussion on Background Subtraction Techniques

The aforementioned background subtraction techniques each have their respective advantages and disadvantages. The selection of a suitable technique depends on the context of application.

Static background subtraction operates at a high processing speed, but is not robust to changes in environment. It is not dynamic or adaptive. This limits its use in contexts in which the environment can change.

Frame differencing is also a simple and computationally efficient technique that is a dynamic and adaptive version of static background subtraction. It is more robust to changing and dynamic environments. It is far more advantageous than static background subtraction.

GMMs are a more complex technique. They are more computationally expensive than the previous two techniques. However, they are more dynamic and adaptive than both previous methods and provide a rich source of information about the motion history of each pixel, that can be more useful than both previous techniques. Therefore, GMMs are selected as a motion cue in the proposed implementation.

3.1.4 Hierarchical Chamfer Matching for Hand Detection

Chamfer matching is an edge-based image matching algorithm, which matches the spatial extensive features on the basis of shape [39]. Chamfer matching is the convolution of a binary edge template with the distance transform. Hierarchical chamfer matching is the extension of hamfer matching into a multi-resolution pyramid. This matching technique utilizes the edge information to match images robustly and the use of a multi-resolution pyramid accelerates the matching process [127].

Subsection 3.1.4.1 provides a description of the chamfer distance transform method which converts a binary edge image to a chamfer distance image, or better known as the distance image.

3.1.4.1 Chamfer Distance Transformation

In the pre-distance image, each non-edge pixel is assigned a value that is a measure of the distance nearest to the edge pixel. The edge pixels are assigned the value 0. The true Euclidean distance is resource demanding (time, memory) to compute, therefore approximation is used. The operation that converts a binary image to an appropriate distance image is called the distance transform (DT). It is important that the DT used in the matching algorithm is a reasonable approximation of the Euclidean distance, otherwise the discriminating ability of the matching measure, computed from the distance values, becomes poor.

The DT uses iterated local operations. The basic idea is that global distances in the image are approximated by propagating local distances, or distances between neighboring pixels, over the image. The propagation of local distances can be done either in parallel or sequentially. Sequential DT's are known as "chamfer" distances.

A 3×3 pixel neighborhood is used. The two local distances in a 3×3 neighborhood are the distance between horizontal/vertical neighbors and between diagonal neighbors. In the original algorithm, the values 2 and 3 were used, respectively. With these values, the maximum difference from the Euclidean distance becomes about 13%. If the values 3 and 4 are used instead, the maximum difference is reduced to 8%. The well-known city block distance has a maximum difference of 59%.

In the binary edge image, each pixel is first set to zero and each non-edge pixel is set to infinity. If the DT is computed by parallel propagation of local distances, then a pixel $v_{i,j}^k$ in position (i,j) at iteration k obtains a new value given by:

$$\begin{aligned} v_{i,j}^k = \min(&v_{i-1,j-1}^{k-1} + 4, v_{i-1,j}^{k-1} + 3, v_{i-1,j+1}^{k-1} + 4, v_{i,j-1}^{k-1} + 3, v_{i,j+1}^{k-1} + 3, \\ &v_{i+1,j-1}^{k-1} + 4, v_{i+1,j}^{k-1} + 3, v_{i+1,j+1}^{k-1} + 4) \end{aligned} \tag{3.21}$$

where i is iterated over all $\{1, \ldots, w\}$ and w is the image width, and j is iterated over all $\{1, \ldots, h\}$ and h is the image height. The iterations continue until no value changes. The number of iterations is proportional to the longest distance occurring in the image. The sequential DT algorithm also starts from the zero/infinity image. Two passes are made over the image, first "forward" from left to right and from top to bottom; and then "backward" from right to left and from bottom to top. The pixel $v_{i,j}$ in position (i,j) obtains a new value in the forward scan given by:

$$v_{i,j} = \min(v_{i-1,j-1} + 4, v_{i-1,j} + 3, v_{i-1,j+1} + 4, v_{i,j-1} + 3, v_{i,j}). \tag{3.22}$$

where i is iterated over all $\{1, \ldots, w\}$ and j is iterated over all $\{1, \ldots, h\}$. In the backward scan, the value of the pixel is set as:

$$v_{i,j} = \min(v_{i,j}, v_{i,j+1} + 3, v_{i+1,j-1} + 4, v_{i+1,j} + 3, v_{i+1,j+1} + 4). \tag{3.23}$$

where, once again, i is iterated over all $\{1, \ldots, w\}$ and j is iterated over all $\{1, \ldots, h\}$.

3.1.4.2 Object Detection Using Chamfer Distance Matching

Once the distance transformed image is obtained, it can be used to locate the position of a template object by means of Chamfer Distance Matching. The template, which may be a binary image containing edges that mark the shape of the target object, is scanned over the distance transformed image. At every position, a sum of the values of the distance transformed image corresponding to the pixels under the edges in the template are summed. This sum provides a measure of the dissimilarity between the template and that region of the image. The region which provides the minimum sum distance is taken to be the region best matching the template.

3.1.4.3 Hierarchical Chamfer Matching

Similar to the pyramidal Lucas-Kanade optical flow method, Hierarchical Chamfer Matching (HCM) makes use of a pyramid of images at different resolutions to speed up Chamfer Distance Matching. A pyramid of images, which are copies of the original image, but at different resolutions is produced. This is called the resolution pyramid. Chamfer Distance Matching is carried out on the lowest resolution image to obtain a coarse measure of the location of the target object. This is used to narrow the search in the next level down in the pyramid. The procedure is repeated until the search takes place on the original image to locate the target object. By using this hierarchical structure of an image pyramid, the computational cost of matching is reduced significantly. An overview of this algorithm can be found in Figure 3.8.

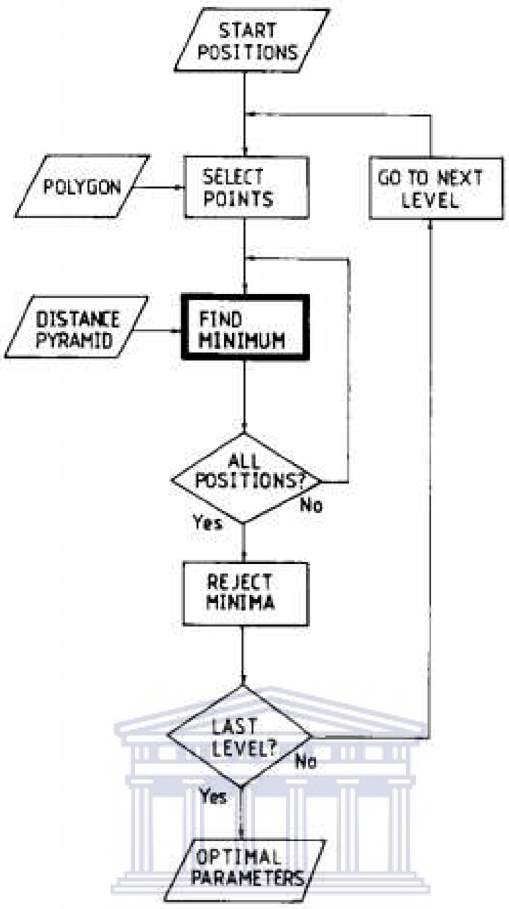


FIGURE 3.8: The Hierarchical Chamfer Matching algorithm [7].

3.1.5 Morphological Operations

Morphological operations are powerful non-linear image analysis tools based on predefined spatial structures known as structuring elements. Morphological image processing describes a range of image processing techniques that deal with shape of features in images. These operations are typically applied to remove imperfections introduced during segmentation, and so typically operate on binary images.

The following subsections describe key morphological operations employed in the proposed system. Subsection 3.1.5.1 discusses the dilation operator which causes objects to appear to expand. Subsection 3.1.5.2 discusses the erosion operator which shrinks objects by eroding away their boundaries. Subsection 3.1.5.3 discusses the opening operator which consists of the erosion operator followed by the dilation operator. Subsection 3.1.5.4 discusses the closing operator which consists of the dilation operator followed by the erosion operator.

3.1.5.1 Dilation

The dilation operator is performed by laying a structuring element B on an image A and sliding it across the image. The algorithm is described in the following sequence of steps:

1. If the origin of the structuring element coincides with a white pixel in the image then there is no change. The algorithm moves to the next pixel.
2. If the origin of the structuring element coincides with a black pixel in the image, the pixel is covered by the structuring element.

See Figure 3.9 for an illustration of the effect of this operator.

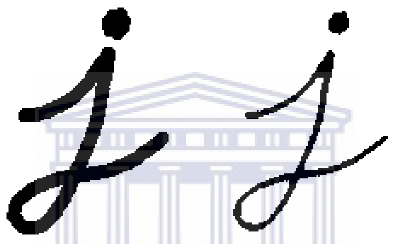


FIGURE 3.9: An illustration of the dilation operator. Dilated image (right) resulting from the original image (left).

Formally, the operator $\oplus$ takes place between the previously defined A and B as follows:

$$A \oplus B = z | B_z \cap A \neq \emptyset \tag{3.24}$$

3.1.5.2 Erosion

The erosion operator is similar to the dilation operator, with the exception that the covered pixel is a white pixel and not a black pixel. The structuring element slides across the image, and the following sequence of steps is applied:

1. If the origin of the structuring element coincides with a white pixel, then there is no change and the algorithm shifts to the next pixel.
2. If the origin of the structuring element coincides with a black pixel, and at least one of the black pixels in the structuring element falls over a white pixel in the image, then the black pixel is changed to a white pixel.

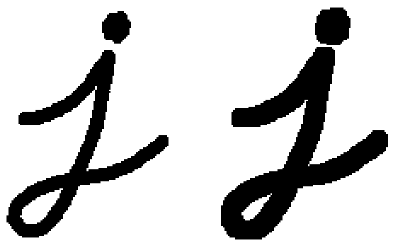


FIGURE 3.10: An illustration of the erosion operator. Eroded image (right) resulting from the original image (left).

See Figure 3.10 for an illustration of the effects of this operator.

Formally, the operator $\ominus$ operates on A and B as follows:

$$A \ominus B = z|(B)_z \subseteq A \tag{3.25}$$

3.1.5.3 Opening

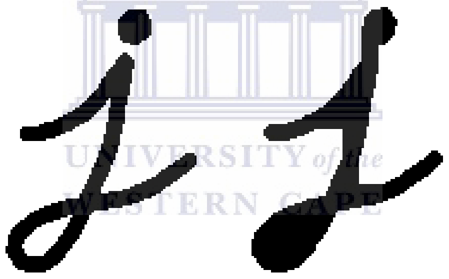


FIGURE 3.11: An illustration of the opening operator. Opened image (right) resulting from the original image (left).

Opening is the application of erosion, followed by dilation on an image A by a structuring element B. Opening smooths the contours of an object, breaks narrow isthmuses, and eliminates thin protrusions. The intensity of all bright features is decreased, depending on the size of the features compared to the structuring element. Opening has a negligible effect on dark features, and the effect on the background is negligible. See Figure 3.11 for an illustration of the effects of this operation. The operator $\circ$ is given by:

$$A \circ B = (A \ominus B) \oplus B \tag{3.26}$$

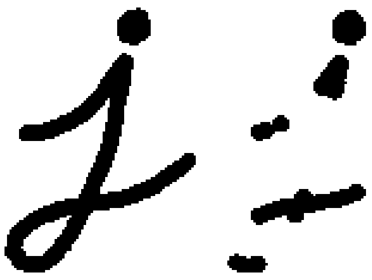


FIGURE 3.12: An illustration of the closing operator. Closed image (right) resulting from the original image (left).

3.1.5.4 Closing

Closing is the application of dilation, followed by erosion on an image A by a structuring element B. Closing smooths sections of the contours, fuses narrow breaks and long thin gulfs and eliminates small holes. See Figure 3.12 for an illustration of the effects of this operation.

Formally, the operator $\bullet$ is given by:

$$A \bullet B = (A \oplus B) \ominus B \tag{3.27}$$

3.1.6 Connected Component Analysis

Connected Component Analysis (CCA), alternatively called connected-components labeling, is an algorithmic application of graph theory to detect connected regions in an image. The algorithm scans over an image, pixel-by-pixel, from top to bottom and from left to right, in an attempt to identify connected pixel regions. It identifies regions of adjacent pixels which share the same set of intensity values. The algorithm operates on binary or gray-level images and different measures of connectivity is possible.

The labeling of connected components is one of the fundamental operations in many intelligent vision systems [35]. This operation operates on binary images by allocating an identifying label to pixels that belong to the same connected area. Two widespread types of labeling are the 4-connected and 8-connected neighbors labeling methods. See Figure 3.13 for an illustration of two different connected neighbors.

There are typically four stages in a Connected Component Analysis algorithm. The first stage is the pre-processing of the image which is normally done by filtering and thresholding to segment objects from the background. The next stage makes use of connected components labeling to assign each region a unique label which enables the

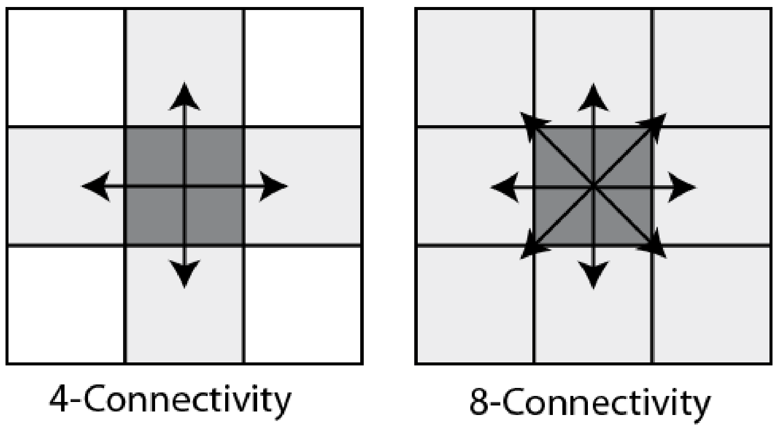


FIGURE 3.13: Different connected neighbors [44].

identification of distinguished objects. The third stage processes each region, based on its label, to extract features of the object that are represented by a particular region. The final stage analyses these features to classify each region into a particular class.

The classic connected components algorithm requires two raster-scan passes through the image. Assuming an 8-connectivity labeling approach, in the first pass, when an object pixel is encountered, the 4 neighbors that have already been processed are examined. If one of those regions are already labeled, then the label is copied to the current pixel. Otherwise, if none are labeled, then a new label is assigned to the current pixel. The second pass in the algorithm replaces each temporary label by the smallest label of its equivalent class.

Choi *et al.* [26] used a single pass streamed algorithm to extract the features of interest for each component or region while performing CCA. This eliminates the need to produce a labeled image and the need to perform the second re-labeling pass. It is also necessary to perform merging and re-labeling on-the-fly to ensure that consistent results are obtained. The new single pass algorithm combines the connected components labeling with the subsequent analysis step.

Once region boundaries have been detected, regions which are not separated by background can be extracted. Any set of pixels which is not separated by a boundary is determined to be connected. Each maximal region of connected pixels is then called a connected component. The set of connected components partition an image into segments. Image segmentation is a useful operation in many image processing applications.

3.1.7 Minimum Bounding Box

The application of the minimum bounding box algorithm contributes towards the normalization process. This algorithm is often used to segment blobs in an image, which are characterized by contiguity and uniformity. A blob has the properties of being either a single object or a significant part of an object. A blob is, thus, also a bounded region that can have a well-defined bounding rectangle. A minimum bounding rectangle is defined as the smallest rectangle that contains every point in a region. The boundaries of the rectangle are aligned with the major and minor axes of the bounded region. The axes of the bounding box are aligned with the image axes and not necessarily with the principal axes of the blob.

Freeman and Shapira [46] introduced a novel method of computing a minimum bounding rectangle. The approach uses four steps to arrive at a minimum bounding box around a blob, given an object A with n boundary points given by $\{(x_i, y_i)|i = 1, \ldots, n\}$.

The first step is to compute the centroid of the object of interest which is also the center of the minimum bounding rectangle. The centroid is given by:

$$\begin{aligned} x_c &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ y_c &= \frac{1}{n}\sum_{i=1}^{n} y_i \end{aligned} \tag{3.28}$$

The next step is to find the angle θ that the major axis of the minimum bounding box makes with the horizontal axis. Determining this angle also provides the angle that the minor axis makes with the horizontal. This can be computed by attempting to determine θ for which the sum of the squares of the perpendicular distances to the major axis at this angle is minimized. The sum of the squares of the perpendicular distances to the major axis P is formulated as:

$$P = \sum_{i=1}^{n}[(x_i - x_c)\sin\theta - (y_i - y_c)\cos\theta]^2 \tag{3.29}$$

Minimizing P with respect to θ requires that $\partial P/\partial\theta = 0$ which leads to:

$$\tan 2\theta = \frac{2\sum_{i=1}^{n}(x_i - x_c)(y_i - y_c)}{\sum_{i=1}^{n}[(x_i - x_c)^2 - (y_i - y_c)^2]} \tag{3.30}$$

The third step in the procedure is to find the extent of the minor and major axes i.e. the furthest points belonging to the object A in the direction of the minor and major axes. The line representing the major axis is given by:

$$(y - y_c) - \tan\theta(x - x_c) = 0 \tag{3.31}$$

To determine the extent of the minor axis, each point (x_i, y_i) is substituted into the left-hand side of Equation 3.31 as:

$$V = (y_i - y_c) - \tan\theta(x_i - x_c) \tag{3.32}$$

The value of the function V can be one of three types: equal to 0, meaning that the point is on the major axis; greater than zero meaning that the point is some distance away from the major axis, and a viable candidate as the extent of the minor axis in this direction; or smaller than zero, meaning that the point is some distance away from the major axis in the opposite direction, and a viable candidate as the extent of the minor axis in this direction. The function V is evaluated on each of the points. The points which have have the maximum and minimum value are taken to be the extents of the minor axis of the minimum bounding box.

The same operation is carried out to determine the extent of the major axis by substituting each point (x_i, y_i) into the left-hand side of the expression of the line for the minor axis, given by:

$$(y - y_c) + \cot\theta(x - x_c) = 0 \tag{3.33}$$

This yields four points, (x_1, y_1), (x_2, y_2), which are the points of the extents of the minor axis on either side, and (x_3, y_3) and (x_4, y_4) which are the points of the extents of the major axis on either side. The final step is to determine the four vertices on the corners of the minimum bounding box A, B, C and D. Assuming These points are given by:

$$A_x = \frac{x_1 \tan\theta + x_3 \cot\theta + y_3 - y_1}{\tan\theta + \cot\theta} \tag{3.34a}$$

$$A_y = \frac{y_1 \cot\theta + y_3 \tan\theta + x_3 - x_1}{\tan\theta + \cot\theta} \tag{3.34b}$$

$$B_x = \frac{x_1 \tan\theta + x_4 \cot\theta + y_4 - y_1}{\tan\theta + \cot\theta} \tag{3.34c}$$

$$B_y = \frac{y_1 \cot\theta + y_4 \tan\theta + x_4 - x_1}{\tan\theta + \cot\theta} \tag{3.34d}$$

$$C_x = \frac{x_2 \tan\theta + x_3 \cot\theta + y_3 - y_2}{\tan\theta + \cot\theta} \tag{3.34e}$$

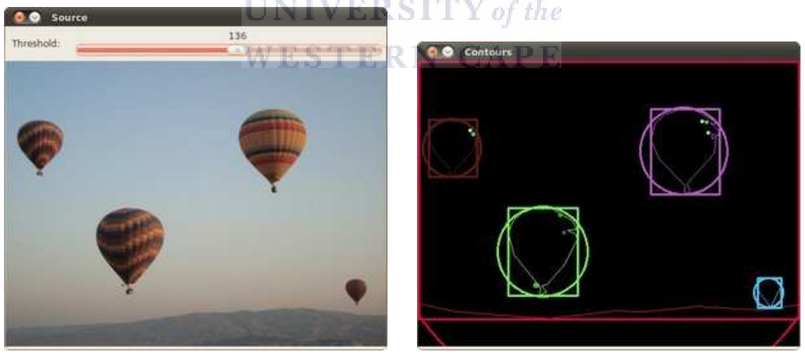
$$C_y = \frac{y_2 \cot\theta + y_3 \tan\theta + x_3 - x_2}{\tan\theta + \cot\theta} \tag{3.34f}$$

$$D_x = \frac{x_2 \tan\theta + x_4 \cot\theta + y_4 - y_2}{\tan\theta + \cot\theta} \tag{3.34g}$$

$$D_y = \frac{y_2 \cot\theta + y_4 \tan\theta + x_4 - x_2}{\tan\theta + \cot\theta} \tag{3.34h}$$

$$\tag{3.34i}$$

where P_x and P_y denote the x and y coordinates of a point P. Figure 3.14 illustrates this algorithm in action.



(a) (b)

FIGURE 3.14: Minimum bounding box computed on a sample image.

3.2 Support Vector Machines in Image Processing

The Support Vector Machine (SVM) is a state-of-the-art classification method introduced in 1992 by Vapnik *et al.* [28]. SVMs are kernel-based methods that have their roots in statistical learning theory [28]. They have been used in various pattern recognition fields such as character, handwriting digit and text recognition [6, 28, 59] and in satellite image classification [62, 87], among other applications.

SVMs offer several advantages over other classifiers. Advantages of the use of SVMs are:

- They are very effective in high dimensional spaces.
- They are still effective in cases where the number of dimensions is greater than the number of samples.
- They use a subset of training points in the decision function, also called support vectors, so it is also memory efficient.
- They are very versatile; different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels for custom cases.

There are many possible choices for classifiers such as decision trees and neural networks. The Support Vector Machine (SVM) is considered a good candidate for its high generalization performance without the need to add *a priori* knowledge, even when the dimension of the input space is high. Intuitively, given a set of points which belong to either one of two classes, a linear SVM finds the hyperplane that leaves the largest possible fraction of points of the same class on the same side, while maximizing the distance of either class from the hyperplane. Hyperplanes minimize the risk of misclassifying examples of a test set.

3.2.1 Optimal Separating Hyperplane

A hyperplane is a linear decision surface that splits the space into two parts. The hyperplane is thus also known as a binary classifier. Let (x_i, y_i) where $(1 \leq i \leq N)$ be a set of training samples with each sample $x_i \in R^d$, d being a dimension of input space, belongs to a class labeled by $y_i \in \{-1, 1\}$. The aim is to find an optimal hyperplane which divides a set of examples such that all the points with the same label are on the same side of the hyperplane. This amounts to finding w and b such that:

$$y_i(w \cdot x_i + b) > 0, i = \{1, \ldots, N\} \tag{3.35}$$

If there exists a hyperplane that satisfies 3.35 then the set is said to be linearly separable [22]. In this case it is possible to rescale w and b such that:

$$[\min_{1 \leq N}] y_i(w \cdot x_i + b) \geq 1, i = \{1, \ldots, N\} \tag{3.36}$$

That is, such that the distance between the closest point to the hyperplane is $\frac{1}{\|w\|}$. Then 3.35 becomes:

$$y_i(w \cdot x_i + b) \geq 1 \tag{3.37}$$

Among the separating hyperplanes that can be drawn, the one in which the distance to the closest points to the hyperplane is maximal is called the optimal separating hyperplane (OSH). Since the distance to the closest point is $\frac{1}{\|w\|}$, finding the OSH amounts to minimizing $\|w\|^2$ under constraints given in equation 3.36.

The quantity $\frac{2}{\|w\|}$ is called the margin, and thus the OSH is the separating hyperplane which maximizes the margin. The margin can be seen as a measure of the generalization ability: the larger the margin, the better the generalization is expected to be [37, 54].

Since $\|w\|^2$ is convex, minimizing it under the linear constraints in Equation 3.36 can be achieved with Lagrange multipliers. If we denote by $\alpha = (\alpha_1, \ldots, \alpha_N)$ the N non-negative Lagrange multipliers associated with the constraints in Equation 3.36, our opimization problem amounts to maximizing:

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j \tag{3.38}$$

with $\alpha_i \geq 0$ and under the constraint $\sum_{i=1}^{N} y_i \alpha_i = 0$. This can be achieved using standard quadratic programming methods [4, 118]. Once the vector $\alpha^0 = (\alpha_1^0, \ldots, \alpha_N^0)$ solution to the maximization problem in Equation 3.37 has been found, the OSH (w_0, b_0) has the following expansion:

$$w_0 = \sum_{i=1}^{N} \alpha_i^0 y_i x_i \tag{3.39}$$

The support vectors are the points for which α_i^0 satisfy Equation 3.36 with equality. Considering the expansion 3.38 of w_0, the hyperplane decision function can thus be written as:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i^0 y_i x_i \cdot x + b_0) \tag{3.40}$$

3.2.2 Linearly Non-Separable Case

When data is not linearly separable we introduce slack variables $(\xi_1, \ldots, \xi_N)$ with $\xi \geq 0$ [15] such that:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, i = 1, \ldots, N \tag{3.41}$$

The purpose of the variables ξ_i is to allow misclassified points, which have their corresponding $\xi_i > 1$. Therefore $\sum \xi_i$ is an upper bound on the number of training errors. The generalized OSH is then regarded as the solution to the following problem:

$$minimize \frac{1}{2} w \cdot w + C \sum_{i=1}^{N} \xi_i \tag{3.42}$$

subject to the constraints in Equation 3.41 and $\xi_i \geq 0$. The first term is minimized to control the learning capacity as in the separable case; the purpose of the second term is to control the number of misclassified points. The parameter C is chosen by the user, a larger C corresponding to assigning a higher penalty of errors.

SVM training requires to fix C in Equation 3.42, the penalty term for mis-classifications. When dealing with images, most of the time, the dimension of the input space is large (≥ 1000) compared to the size of the training set, so that the training data is generally linearly separable. Consequently, the value of C has, in this case, little impact on the performance.

3.2.3 Non-Linear Support Vector Machine

The input data is mapped onto a higher-dimensional feature space through some non-linear mapping chosen priori. In this feature space, the OSH is constructed. If x is replaced with its mapping in the feature space $\Phi(x)$, Equation 3.37 becomes:

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \Phi(x_i) \cdot \Phi(x_j) \tag{3.43}$$

If we have $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, then only K is needed in the training algorithm and the mapping Φ is never explicitly used. Conversely, given a symmetric positive kernel $K(x, y)$, Mercer's theorem [75] indicates that there exists a mapping function Φ such that $K(x, y) = \Phi(x) \cdot \Phi(y)$.

Once a kernel K satisfying Mercer's condition has been chosen, the training algorithm consists of the following minimization problem:

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{3.44}$$

and the decision function becomes:

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i y_i K(x_i, x) + b) \tag{3.45}$$

3.2.4 Multi-class Learning

SVMs are specifically designed for binary classification. An appropriate multi-class classification technique is required when dealing with multi-class problems, as is often the case in object recognition and classification. Different possibilities exist, including the following:

- The design of the SVM can be modified in order to incorporate the multi-class learning directly in the quadratic solving problem [29].
- Several binary classifiers can be combined with a decision strategy to form a single multi-class classifier. Various decision strategies exist including the "One against One", "One against Others" and Directed-Acyclic Graph Rejection tree techniques.

The "one-against-others" strategy was introduced by Vapnik [28] in 1995. In the "one-against-others" strategy, a set of binary classifiers is trained to be able to separate each class from all the others. Each data object is classified to the class for which the largest decision value was determined[54]. This strategy trains N SVMs, where N is the number of classes, and there are N decision functions. Although it is a fast method, it suffers from errors caused by marginally imbalanced training sets.[117]

The "one-against one" strategy was introduced by Vapnik [22] in 1998 as an improvement to the "one-against-others" strategy. In this strategy a series of classifiers is applied to each pair of classes, with the most commonly computed class kept for each object. A max-win operator is used to determine to which class the object will finally be assigned. The application of this method requires $\frac{N(N-1)}{2}$ machines to be applied. Even this strategy is more computationally demanding than the "one against all" strategy, it has been shown that it can be more suitable for multi-class classification problems [54].

The "Directed-Acylic Graph Rejection" strategy was proposed by Baker and Nayar [12]. Its training phase is the same as the "one-against-one" method by solving $\frac{N(N-1)}{2}$ binary SVMs. However, in the testing phase, it uses a rooted binary acyclic graph which has $\frac{N(N-1)}{2}$ internal nodes and N leaves. Each node is a binary SVM of i'th and j'th classes. An advantage of this method is that some analysis of generalization can be established [54].

3.3 Summary

This chapter discussed, in detail, the components of the gesture recognition system proposed in this research. The discussion first discussed the various image processing techniques that are used to carry out hand tracking and segmentation, and hand shape recognition. Details were provided about the Viola-Jones face detection algorithm which was used as the basis for skin detection. The dynamic skin detection method used was discussed. To this end, various prominent colour spaces were also discussed and it was concluded that the HSV colour space was suitable for skin detection. Various background subtraction techniques were detailed culminating in the selection of Gaussian Mixture Models for the proposed system. The use of Hierarchical Chamfer Matching as a robust object detection technique was discussed. Various image enhancement operations – morphological operations – were discussed and their mode of operation detailed. Lastly, the use of Connected Component Analysis to locate large connected regions in an image, followed by the use of a minimum bounding box to isolate these regions were detailed.

Thereafter, a detailed discussion of Support Vector Machines and their mode of data classification was carried out. It was shown that the technique is effective and can classify data in higher-dimensional spaces using kernels. The methods developed to adapt SVMs to multi-class classification problems were also detailed.

The next chapter details the proposed system which makes use of each of these components to achieve SASL gesture recognition.

Chapter 4

Design and Implementation of the Integrated Sign Language Recognition System

4.1 System Design Overview

This chapter discusses the implementation of the proposed sign language gesture recognition system. The shape and the location of the hand in an isolated gesture are the decisive features in this system. The pre-processing procedure strives to ensure that such features are accurately extracted from a video feed with few constraints. A motion thresholding model is used to segment the video feed into isolated gestures. Figure 4.1 provides an overview of the proposed system.

The system consists of three stages. The first stage is the hand tracking stage in which the location of the hand is initially located and then tracked throughout the frame sequence with the CamShift tracking algorithm. Section 4.2 provides further details on the hand tracking strategy employed.

Section 4.3 details the use of a motion thresholding model to segment the image sequence into single frames each containing an isolated gesture to carry out gesture recognition on. Upon successful tracking and segmentation of the hand, the location of the hand in signer space is computed. This process is explained in Section 4.4.

The location and shape of the hand are then used to construct a feature vector used as input to a Support Vector Machine (SVM). The feature extraction and representation process of the recognition of the hand shape is explained in Section 4.5. The process

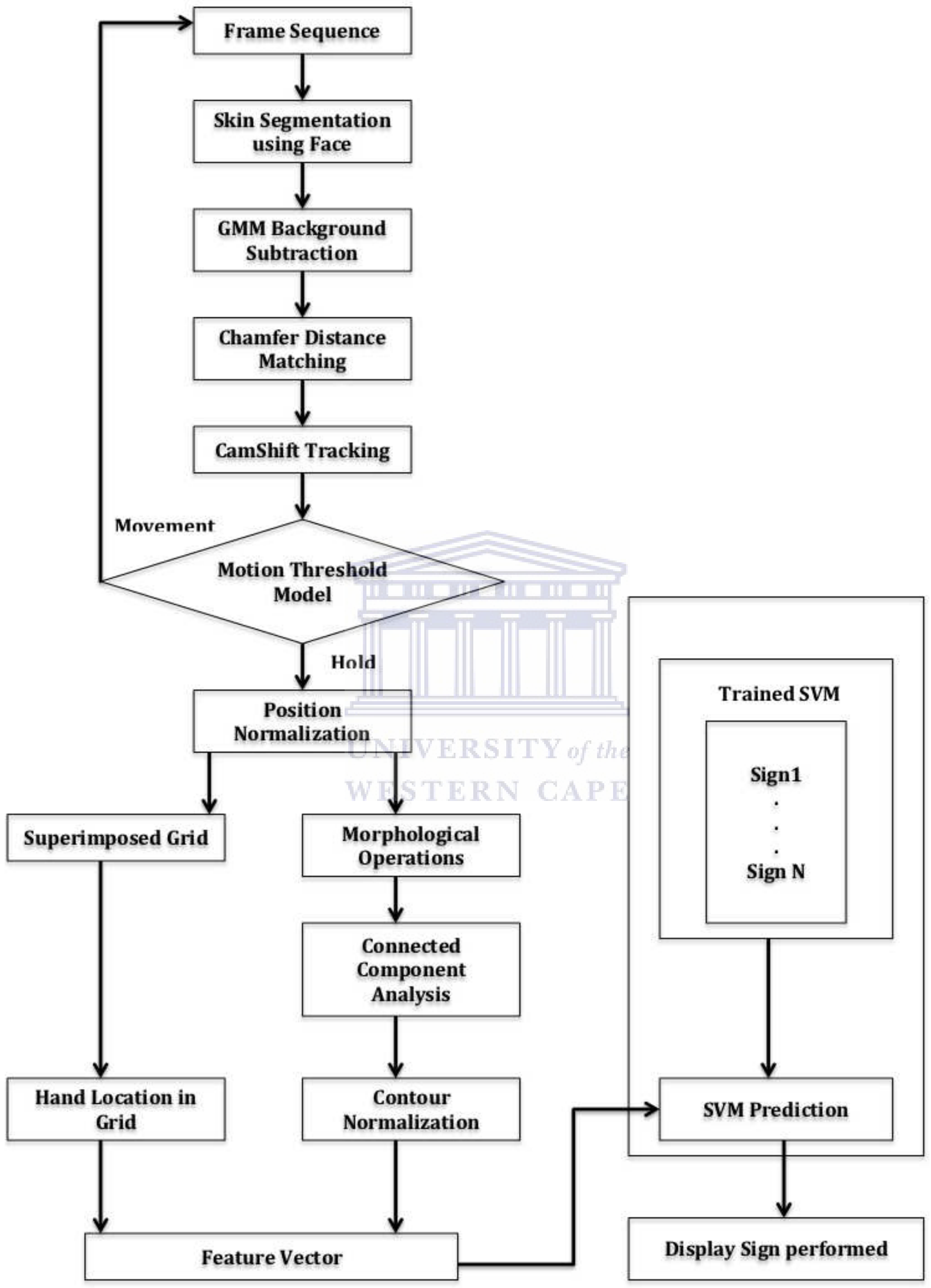


FIGURE 4.1: An overview of the system.

of the classification using this feature vector is explained in Section 4.6. The chapter is concluded in Section 4.7

4.2 Hand Tracking

The most important aspect of the recognition system is a robust hand tracking strategy, used to segment gesture sequences into isolated gestures and segment the hand in each isolated gesture frame.

In this system, the CamShift tracking algorithm is used. CamShift requires an initial tracking window to be specified. This can be done manually but the proposed system requires an automatic procedure of locating the hand to initialize the CamShift tracking window. Once it has been initialized, the algorithm continuously tracks the hand in the image sequence automatically.

The focus of the following subsections is the initial hand segmentation procedure that leads up to the initialization of the CamShift tracking algorithm. Section 4.2.1 describes the dynamic skin detection technique used to identify and segment the regions that are of skin colour. Gaussian Mixture Models are then used to eliminate pixels that resemble skin but which have not moved, and should be considered as part of the background. This is explained in Section 4.2.2. In Section 4.2.3, Hierarchical Chamfer distance matching is applied to accurately locate the hand in the frame. The detected hand region is used to initialize the CamShift tracking algorithm, explained in Section 4.2.4.

4.2.1 Dynamic Skin Segmentation Using Face Detection

Face detection is the initial step of the system. The face detection algorithm is adopted from the Viola-Jones method. It is further used in the system to determine the presence of a signer. The algorithm searches for any number of possible faces in the image. The initial face detection search size is set to the size of the frame and continuously reduced until a face is located. The search is also set to the find the largest possible face in the frame. This allows for the signer to stand at a variable distance from the camera. Once the face is located, the face region is extracted and the system advances to generating a skin model.

Thereafter, the skin detection method proposed by Achmed [2], which was explained in detail in Section 3.1.2.7, is applied to highlight the skin pixels in the frame. An illustration of the results of this application can be seen in Figure 4.2. The image resulting from this procedure is henceforth referred to as the “skin image”.

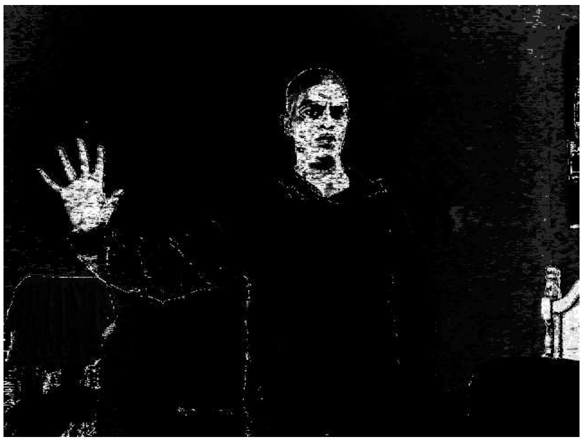


FIGURE 4.2: Skin tone regions highlighted.

4.2.2 Background Subtraction to Obtain a Moving Skin Image

This system makes use of the Gaussian Mixture Models (GMMs) background subtraction method. It is used for its robustness in effectively separating the foreground from the background. The background model is computed and updated by the difference between the current frame and the current background model. The foreground pixels for moving objects can thus be separated automatically from the image. An illustration of applying GMMs to a sample image can be seen in Figure 4.3.

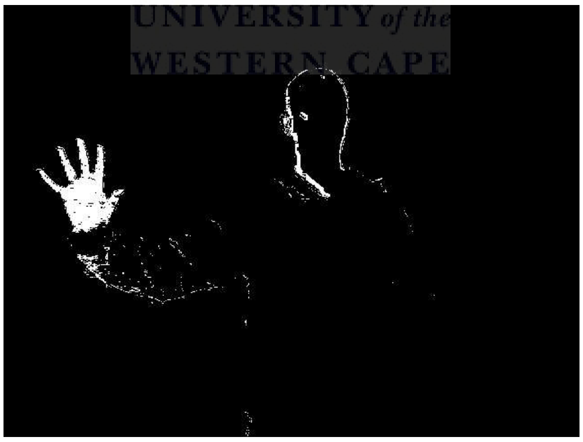


FIGURE 4.3: Gaussian Mixture Model.

The image obtained by applying GMMs to the original image is combined with the skin image using the logical AND operation, resulting in an image in which only those pixels that are skin coloured and have moved are highlighted. This image is referred to as the "moving skin image". The combination of the skin and motion cues is effective in reducing noise and providing robustness to complex backgrounds.

4.2.3 Hand Detection Using Hierarchical Chamfer Distance Matching

The Hierarchical Chamfer Distance Matching (HCDM) algorithm is used to locate the hand of the signer only once at the beginning of the image sequence. To this end, the signer is required to hold up his hand momentarily until the CamShift tracking is visibly initialized. The location of the hand is used to initialize the CamShift tracking window.

Figure 4.4 depicts the image of the signing hand that consists only of the features which in essence is an edge-based hand-template. Figure 4.5 depicts the image consisting of

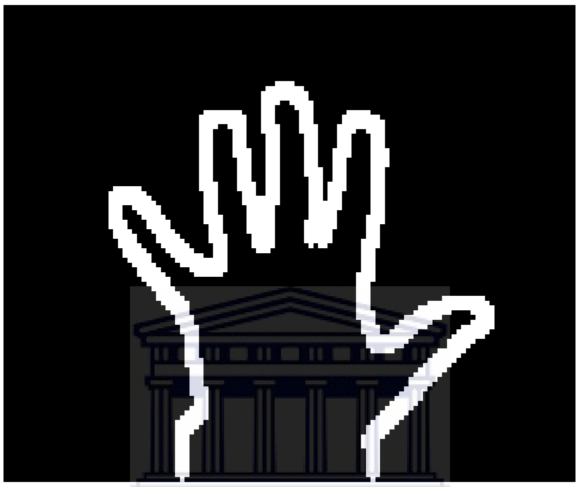


FIGURE 4.4: Edge-based hand template.

non-features and possible feature points in the foreground image that is obtained from the previous section. The two binary images are converted into a distance transform image where each pixel denotes the distance to the nearest feature pixel.

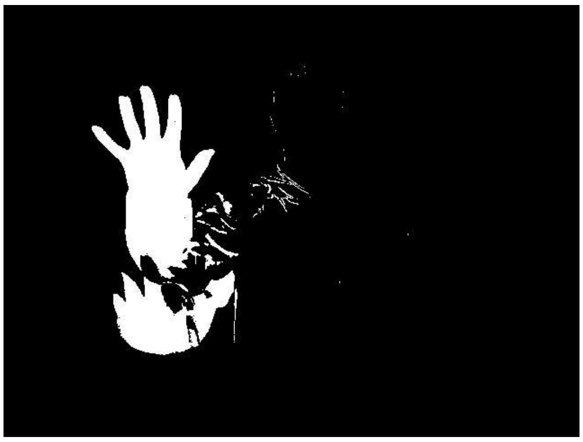


FIGURE 4.5: Foreground image consisting of possible feature points.

4.2.4 Hand Tracking Using the CamShift Tracking Algorithm

Once initialized, the CamShift algorithm is applied to the moving skin image and continuously tracks the signer's hand across the image sequence. In each frame, the algorithm finds the center of the hand region and then adjusts the position and size of the window and determines its optimal rotation. The algorithm returns the rotated rectangle structure that includes the position, size, and orientation of the object. The search window is then shifted to the new location, with the new rotation and size.

See Figure 4.6 for an illustration of the hand of the signer being tracked by CamShift.

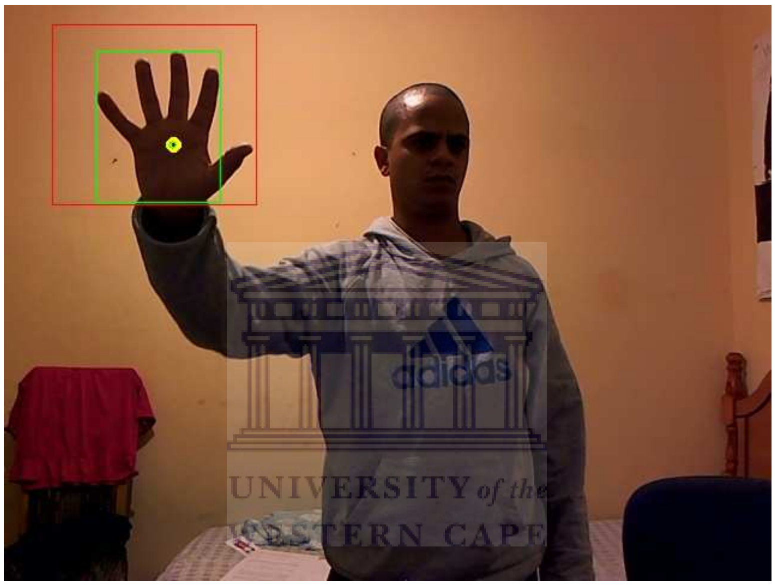


FIGURE 4.6: Hand tracking using CamShift.

4.3 Isolated Gesture Segmentation Using a Motion Thresholding Model

Once hand tracking has been initialized, a motion thresholding model similar to Liang and Ouhyoung [69] is used. In our version of this model, two motion thresholds are used to segment an image sequence, which is assumed to be a continuous sequence of isolated gestures, into movements and holds. It is initially assumed that the signer is in a hold state. However, since this is the initial state of the system, the frames captured are not captured as a "hold" until movement takes place. A threshold is applied to determine the transition of the hold into a movement. This threshold, denoted T_M is chosen to be robust to small natural movements of the hand in a hold state. Once the hand has begun to move, a separate threshold is used to determine the transition of the movement into a hold. This threshold, denoted T_H, is chosen to detect the instance in which the hand is in a stationary or quasi-stationary state.

Once a hold is registered, one frame is extracted and used to determine the isolated gesture in that frame using the feature extraction and gesture classification process explained in later sections. See Figure 4.7 for an illustration of this isolated gesture segmentation model. The experimentation carried out in order to obtain the optimal thresholds T_M and T_H is described in the next chapter.

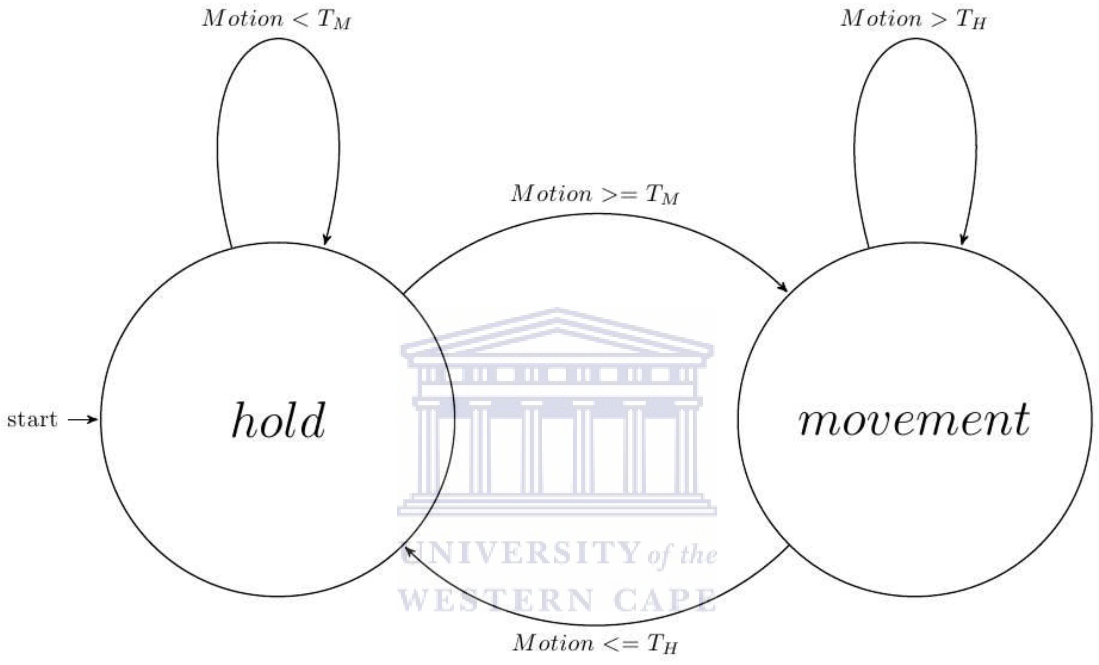


FIGURE 4.7: The isolated gesture segmentation model.

4.4 Hand Location Feature Representation and Extraction

Once an isolated gesture frame is obtained, the first parameter that is extracted from the frame is the hand location. In order to represent the hand location, an appropriate normalized coordinate system is required. The technique used is derived from the location representation technique used by Achmed [2] and Brown [18]. Both researchers found that the technique was appropriate in representing the location parameter towards SASL gesture recognition.

The approach first normalizes the position of the signer in the frame. Section 4.4.1 discusses this in further detail. A dynamical grid is superimposed on the signer, which serves as a representation of the signer space for the signer's hands to work within. The

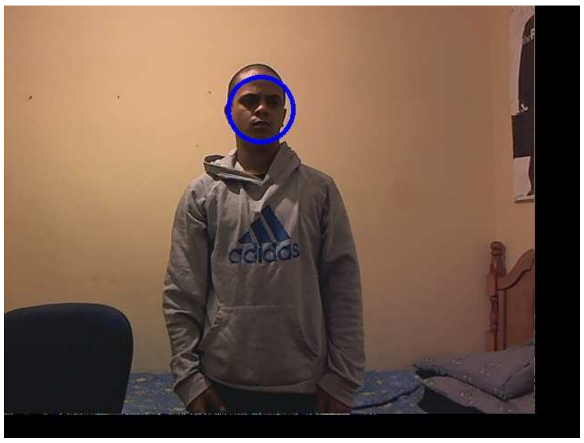


FIGURE 4.8: Signer normalized in the frame.

location of the hand within the grid provides an appropriate representation of the hand location. The following subsections describe the position normalization strategy and grid system used.

4.4.1 Position Normalization

The position of the face is used to normalize the position of the signer in each frame in the frame sequence. In each frame, the pixels in the frame are shifted such that the face of the signer is positioned in the center of the frame. This is sufficient to achieve a frame-centered coordinate system. Figure 4.8 illustrates a signer that is shifted more to one side of the frame that is then normalized using this strategy.

4.4.2 Superimposed Grid

The location of the hand relative to the gesture performed is obtained with the use of a dynamic superimposed grid. The grid serves as a representation of the signing space as well as the gesture performed. The facial frame obtained by means of face detection is used to determine the size of each block in the grid. Similar to Achmed [2] and Brown [18], the grid consists of 14×12 equally sized blocks, with each block the size of a quarter of the face. This yields a total of 168 blocks. See Figure 4.9 for an illustration of the grid.

Referring to the figure, the top-left block is indexed as block 1, and the index increases towards the right and downwards, with the bottom-right block assigned the index 168. The grid is positioned such that the middle of the nose is positioned at the intersection of the 3^{rd} row and the 7^{th} column.

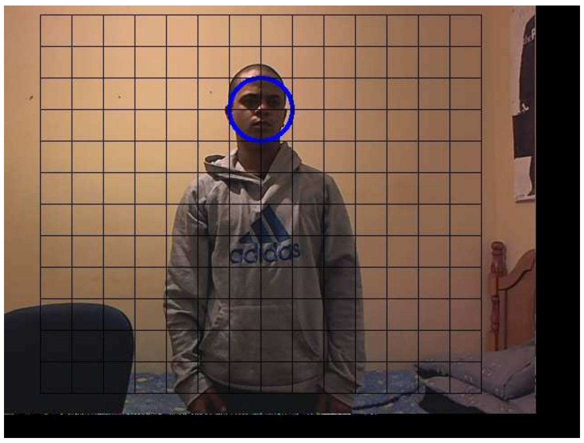


Figure 4.9: Superimposed grid.

The block label in which the (x, y) co-ordinates of the tracked hand in the original image is located is then computed. The label of the block in which the middle point of the hand is in is used as the location feature in the feature vector.

4.5 Hand Shape Feature Representation and Extraction

Using the region of the image in the isolated gesture frame provided by the CamShift tracking window, features are extracted from the hand region-of-interest (ROI) pertaining to the hand shape.

To achieve this, the hand contour has to be obtained. However, the resulting hand image might still contain noise in the form of small isolated islands of moving skin detecting in and around the actual hand region. Noise in the image has to be reduced. Noise in this binary image is reduced with the use of morphological operations explained in Subsection 4.5.1. In Subsection 4.5.2, Connected Component Analysis is used to segment the binary image into blobs. This is used to identify the hand contour. Finally in Subsection 4.5.3 a minimum bounding rectangle is applied to segment and normalize the hand contour from the region of interest.

4.5.1 Morphological Operations

The moving skin image is robust to noise in the image. However, small isolated pockets of noise may still exist in the resulting image. In order to reduce these sources of noise, morphological operations are applied. The morphological operations Opening and Closing, which are both combinations of the more fundamental dilation and erosion morphological operations, are applied.

The Opening operator is applied first to separate and preserve large foreground areas while removing small isolated clusters of noise pixels. The result of the Opening operator can be seen in Figure 4.10.

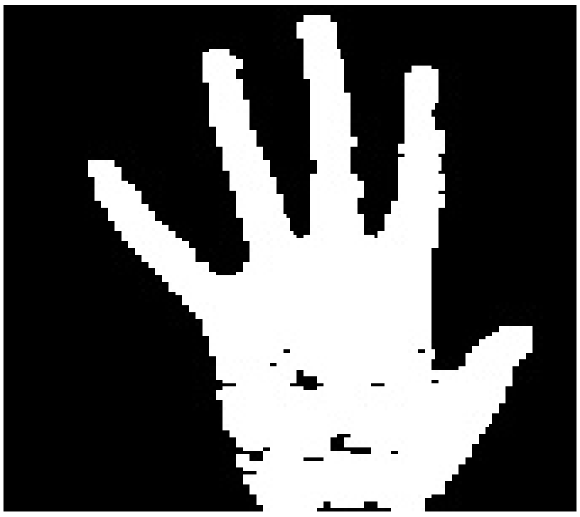


FIGURE 4.10: Opening operator.

The Closing operator is then applied to the result to shrink any holes in the foreground object while preserving the original shape of the object. The result of applying the Closing operation can be viewed in Figure 4.11.

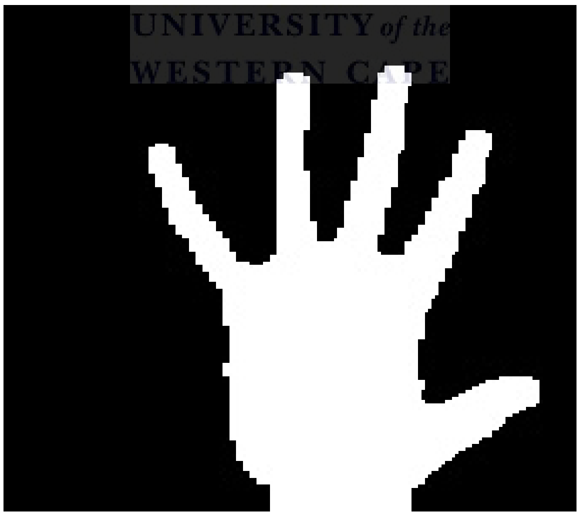


FIGURE 4.11: Closing operator.

4.5.2 Connected Component Analysis

At this stage, the resulting image is a smoothed binary silhouette image of the hand. Connected Component Analysis (CCA) is used to extract and represent all contours in this image. The biggest contour is considered to be the target object which is the hand contour. All other contours not connected to this contour are eliminated. See Figure 4.12 for an illustration of the result of this operation.

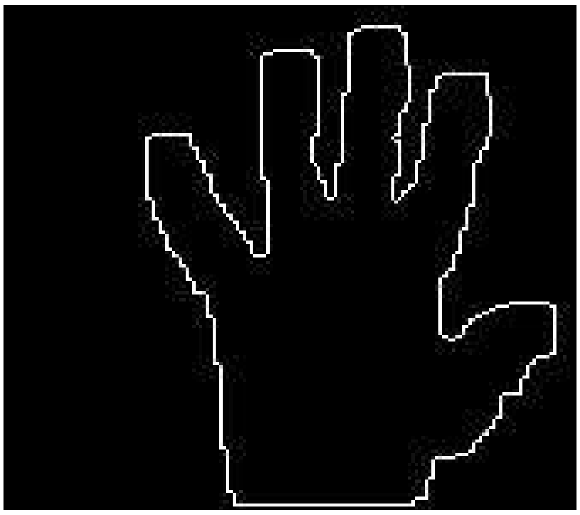


FIGURE 4.12: Connected Component Analysis.

4.5.3 Feature Normalization by Applying a Minimum Bounding Rectangle

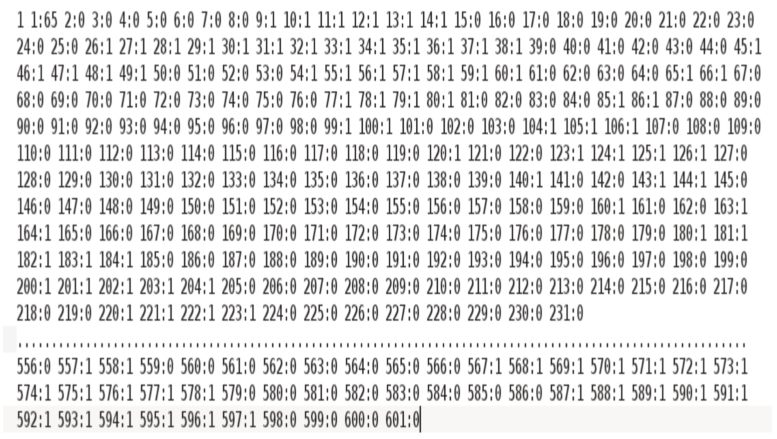
The final stage of the hand shape feature extraction procedure is to apply a minimum bounding rectangle to the hand contour image obtained in the previous section. Normalization is carried out by rotating the hand contour image such that the principal axis of the region is aligned with the vertical image axis. Finally, the resulting image is scaled down to a size of 20×30 pixels. The 20×30 image is used as part of the feature vector explained in Subsection 4.6.1.

4.6 Classification using a Support Vector Machine

This section discusses the procedure used to implement the gesture recognition subsystem of this research.

4.6.1 Feature Vector

Each line in the SVM feature vector file consists of a SASL gesture label, followed by the feature vector of the hand contour, which consists of the location and shape of the hand. Each SASL gesture to be recognized is assigned a label. The format of the labelled feature vector is illustrated in Figure 4.13.



```
1 1:65 2:0 3:0 4:0 5:0 6:0 7:0 8:0 9:1 10:1 11:1 12:1 13:1 14:1 15:0 16:0 17:0 18:0 19:0 20:0 21:0 22:0 23:0
24:0 25:0 26:1 27:1 28:1 29:1 30:1 31:1 32:1 33:1 34:1 35:1 36:1 37:1 38:1 39:0 40:0 41:0 42:0 43:0 44:0 45:1
46:1 47:1 48:1 49:1 50:0 51:0 52:0 53:0 54:1 55:1 56:1 57:1 58:1 59:1 60:1 61:0 62:0 63:0 64:0 65:1 66:1 67:0
68:0 69:0 70:0 71:0 72:0 73:0 74:0 75:0 76:0 77:1 78:1 79:1 80:1 81:0 82:0 83:0 84:0 85:1 86:1 87:0 88:0 89:0
90:0 91:0 92:0 93:0 94:0 95:0 96:0 97:0 98:0 99:1 100:1 101:0 102:0 103:0 104:1 105:1 106:1 107:0 108:0 109:0
110:0 111:0 112:0 113:0 114:0 115:0 116:0 117:0 118:0 119:0 120:1 121:0 122:0 123:1 124:1 125:1 126:1 127:0
128:0 129:0 130:0 131:0 132:0 133:0 134:0 135:0 136:0 137:0 138:0 139:0 140:1 141:0 142:0 143:1 144:1 145:0
146:0 147:0 148:0 149:0 150:0 151:0 152:0 153:0 154:0 155:0 156:0 157:0 158:0 159:0 160:1 161:0 162:0 163:1
164:1 165:0 166:0 167:0 168:0 169:0 170:0 171:0 172:0 173:0 174:0 175:0 176:0 177:0 178:0 179:0 180:1 181:1
182:1 183:1 184:1 185:0 186:0 187:0 188:0 189:0 190:0 191:0 192:0 193:0 194:0 195:0 196:0 197:0 198:0 199:0
200:1 201:1 202:1 203:1 204:1 205:0 206:0 207:0 208:0 209:0 210:0 211:0 212:0 213:0 214:0 215:0 216:0 217:0
218:0 219:0 220:1 221:1 222:1 223:1 224:0 225:0 226:0 227:0 228:0 229:0 230:0 231:0
..........................................................................................
556:0 557:1 558:1 559:0 560:0 561:0 562:0 563:0 564:0 565:0 566:0 567:1 568:1 569:1 570:1 571:1 572:1 573:1
574:1 575:1 576:1 577:1 578:1 579:0 580:0 581:0 582:0 583:0 584:0 585:0 586:0 587:1 588:1 589:1 590:1 591:1
592:1 593:1 594:1 595:1 596:1 597:1 598:0 599:0 600:0 601:0
```

FIGURE 4.13: Format of feature vector.

The label of the feature vector corresponds to the sign performed. For example, a label "1" may be assigned to the SASL gesture "Sorry". Each of the entries in the feature vector that follow are labeled with an index. The first index in the feature vector describes the location value of the sign performed. This numeric location was obtained in Section 4.4.2. The rest of the feature vector entries describe the 20×30 pixel contour that resulted from the aforementioned feature extraction process. The pixels of the 20×30 contour image are represented as a linear vector of size 600×1. Thus, feature indices from $2 - 601$ are the contour features. This yields a feature vector of total size 601. This is illustrated in in Figure 4.13.

Referring to Figure 4.13, the first feature has an index of 1 and a value of 65, which is the numerical value of the grid block in which the hand is located. The second feature has an index of 2 and a value of 0 and the last feature has an index of 601 and a value of 0. The features with indices 2 to 601 are the aforementioned hand contour pixels.

4.6.2 Training Phase

The data set used in the training phase as well as the method of collecting data is described in Subsections 4.6.2.1 and 4.6.2.2. The procedure used for the training of the SVM is illustrated in Figure 4.14.

4.6.2.1 Data Set

To date, there is no known SASL gesture database. In order to train and test the proposed gesture recognition strategy, a SASL data set was manually collected. 50 signs were selected from the Fulton School for the Deaf SASL Dictionary [53]. The properties

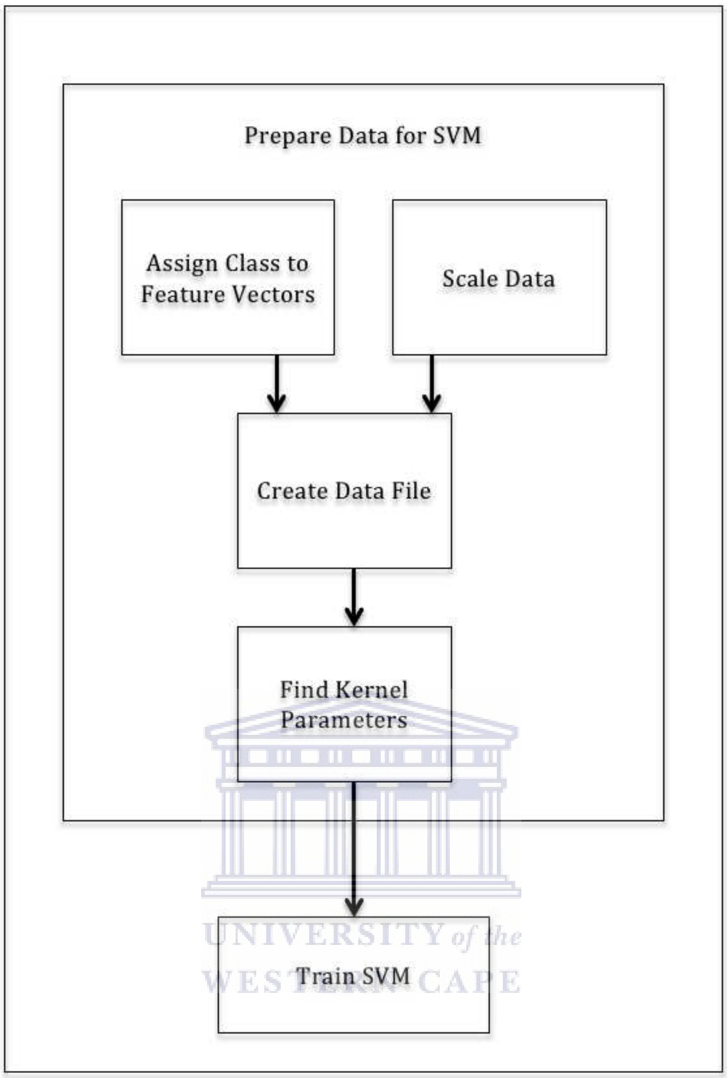


FIGURE 4.14: Procedure used for SVM training.

of all signs were gathered from this book. Signs were chosen such that a wide variety of hand shapes (24 hand shapes) and locations (6 locations) were represented. Table 4.1 below lists the 50 signs and their labels. Table A.1 in Appendix A details each of the 50 signs in terms of the hand shape and location parameters of each sign, sorted according to the six locations. The labels assigned to each sign are also provided in the table. An illustration of each sign performed is given in Figures A.1 – A.3 in Appendix A.

Ten subjects were used for the training of the system, each performing each of the 50 signs once. There was a diversity amongst training subjects' skin colour, gender, and body measurements. This was done in order to represent a wide variety of subjects. Table 4.2 contains details on each of the 10 subjects used in training. In Table 4.2 the attributes of each signer consists of skin colour, age, gender, and body measurements in terms of height. It is important to consider the height of the training signer to prove

Label	**Sign**
1	Yes
2	Animal
3	Bye-Bye
4	Fish
5	Come
6	Goose
7	Snake
8	Cut
9	Sorry
10	Shower
11	Dolphin
12	Go
13	Button
14	Hearing
15	Worm
16	Handbag
17	What
18	Son
19	Which
20	Money
21	Bell
22	Aeroplane
23	Fly
24	Girl
25	Flour
26	Salt
27	Juice
28	Daughter
29	Plum
30	Orange
31	Microwave
32	Naartjie
33	Sponge
34	Coke
35	Yellow
36	Monday
37	Hotel
38	Tuesday
39	Switch On
40	Me
41	Wednesday
42	Down
43	Switch Off
44	Mine
45	Thursday
46	Waiter
47	Kettle
48	Tall
49	Tap
50	Zoo

Table 4.1: The 50 SASL signs selected and corresponding labels.

that superimposed grid has the ability to adapt to signers of different size. The table lists body measurements as either short, everage, or tall. Short signers ranged from 1.4m to 1.5m in length, average signers was at approximately 1.6m, and tall signers were between 1.7m and 1.8m.

Signer	**Skin colour**	**Age**	**Gender**	**Body Measurements**
Signer 1	Light Brown	21	Male	Tall
Signer 2	Light Brown	20	Male	Short
Signer 3	Light Brown	28	Female	Average
Signer 4	White	18	Male	Short
Signer 5	Brown	23	Male	Tall
Signer 6	Dark Brown	25	Female	Average
Signer 7	Brown	20	Female	Average
Signer 8	Dark Brown	25	Male	Average
Signer 9	White	19	Male	Average
Signer 10	Dark Brown	25	Female	Average

Table 4.2: Details of the subjects used in training.

4.6.2.2 Collection of Videos

Each subject was required to stand in front of the camera such that his/her upper body and hands were visible in the frame. A pre-recorded video was played for the subject which demonstrated each sign prior to performance. Thereafter, the subject was instructed to perform each sign. Each subject had two practice rounds and one final round to perform the gesture. This was repeated until the subject had performed all 50 signs. This resulted in a total of 500 videos.

Each of the 500 videos starts with the subject with both hands to the sides, after which the subject proceeds to perform one hold/sign. The segmented hold was pre-processed using the aforementioned procedure, resulting in a feature vector inserted into the file for the training of the system.

4.6.2.3 Training and Optimization Procedure

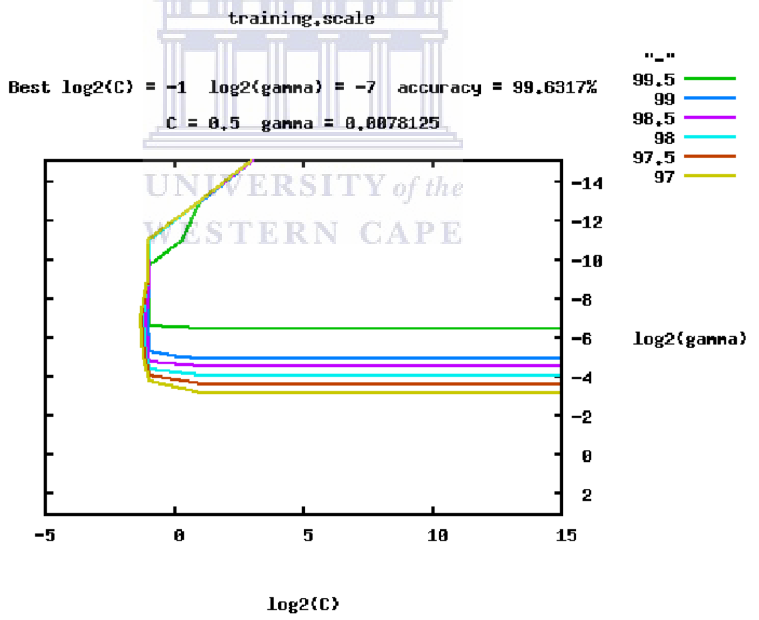


FIGURE 4.15: Grid search optimization function results.

Research shown RBF kernel is the most suitable in many applications [2, 18, 68]. Thus this kernel is chosen for this application. The kernel has two parameters C and γ which can be optimized. The optimization requires an exhaustive search through C and γ value combinations to determine the pair that achieves the highest accuracy separation between the classes in the data set. The LibSVM [21] grid-search function provides this functionality. Given a training set, the function iterates through combinations of C and

γ values and determines the pair that achieves the highest cross-validation accuracy. This was used in this research.

The results of running the LibSVM grid-search function on the training data is depicted in Figure 4.15. The figure depicts the accuracy of varied combinations of C and γ values. The accuracy rate of the kernel was optimized from 88% to 99.6317%. The difference between the two accuracies is an indication that a high accuracy can be achieved with the optimization of the RBF kernel. The optimum parameters were found to be as follows: C was 0.5 and γ was 0.0078125.

The SVM was trained on the training data using the mentioned C and γ parameter values.

4.6.3 Testing Phase

The testing phase involves carrying out a prediction on an unseen input sequence. This involves pre-processing the sequence using the aforementioned segmental model to extract isolated gesture frames and the feature extraction and normalization process on the resulting gesture frame. This results in a feature vector. The format of this feature vector is the same as in Subsection 4.6.1. The procedure used in the testing phase is illustrated in Figure 4.16. The feature vector is used as input to the trained SVM which predicts a label corresponding to one of the 50 signs. N-way classification (where N=50) was performed to classify each sign as one of the possible 50 signs.

4.7 Conclusion

In this chapter, the implementation of the feature extraction process and the feature classification process of the proposed gesture recognition system was described. The features extracted were the location and the shape of the hand. The proposed continuous isolated gesture segmentation approach was also described. It is possible to state at this stage that objectives 1 and 2 described in Chapter 1 have been successfully achieved. In the next chapter, the accuracy of the system is evaluated in order to achieve the two remaining objectives and answer the research question posed in the same chapter.

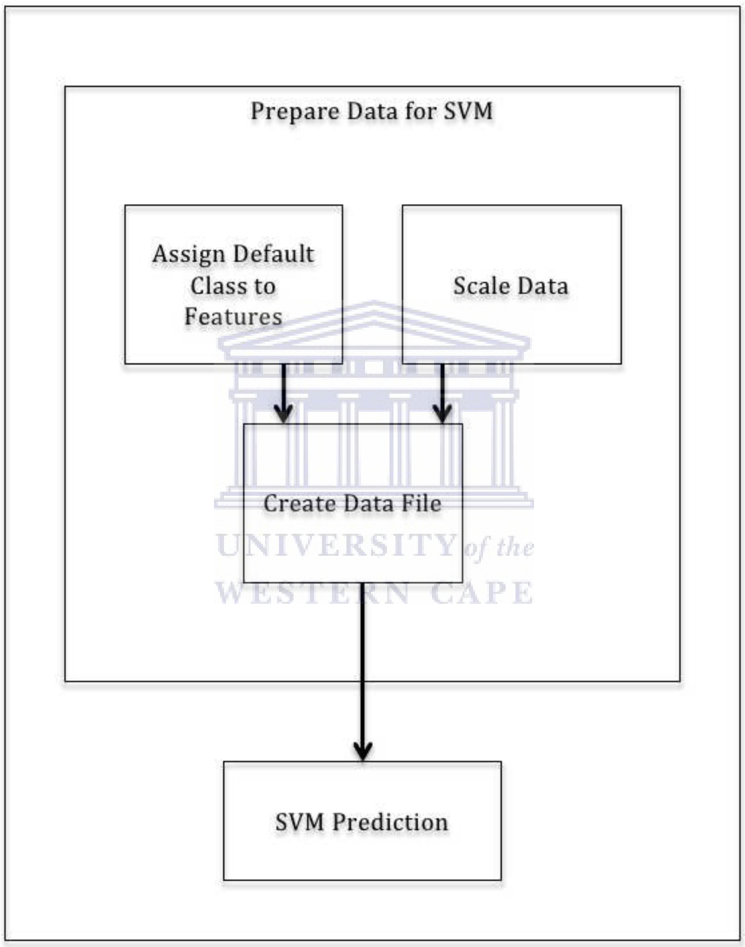


FIGURE 4.16: Procedure used in the testing phase.

Chapter 5

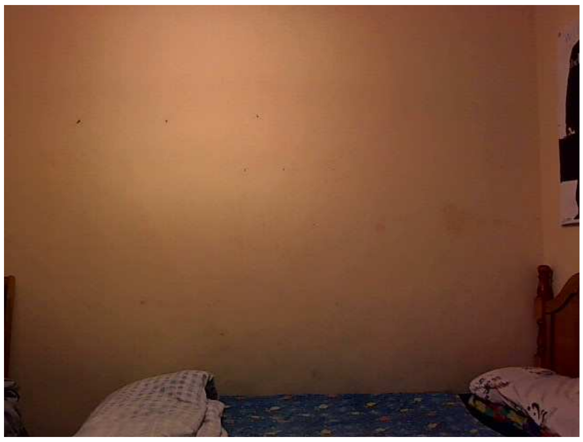
Experimental Setup and Analysis of Results

This chapter evaluates the proposed integrated sign language recognition system in order to answer the research question posed in Chapter 1. The chapter also aims to complete the objectives that were introduced in Chapter 1.

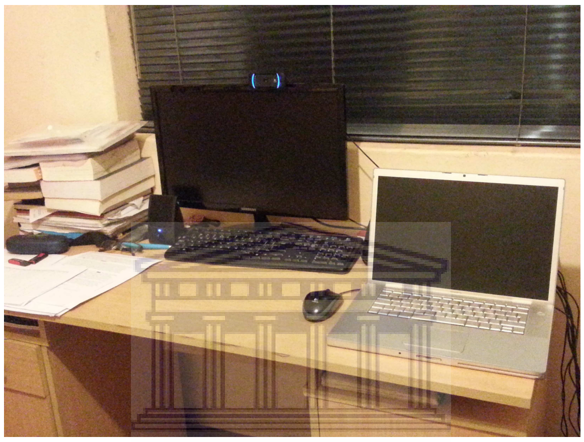
Section 5.1 explains the experimental setup that was used in all of the experiments carried out. Section 5.1.1 describes the data set used in experimentation. The experimentation carried out to assess the tracking accuracy of the system is described in Section 5.2. The experimentation carried out to optimize the two threshold parameters of the proposed isolated gesture segmentation strategy is described in Section 5.3. Finally, Section 5.4 describes the experimentation carried out to assess the gesture recognition accuracy of the proposed system. The chapter then concludes in Section 5.5 with a response to the research question posed in Chapter 1.

5.1 Experimental Setup

The experiments were conducted on a PC containing an Intel i7 3550k 3.4 GHz quad core CPU, an NVIDIA 580GTX GPU and 16 GB RAM and running the Kubuntu 12.10 x64 operating system. A Logitech C910 web camera was used at a resolution of 640×480 pixels and a frame rate of 15 FPS. An illustration of the experimental setup can be found in Figure 5.1.



(a) Background of Experimental Setup



(b) Experimental Setup

Figure 5.1: The 50 SASL signs used in experimentation.

5.1.1 Data Set

Videos of 25 subjects were used to optimize the thresholds of the gesture segmentation model and test the system. These subjects were different to the subjects used in training the system, described in the previous chapter. Testing was thus conducted on unseen data. Each subject was asked to each perform 10 sentences consisting of 5 of the 50 signs mentioned in the previous chapter. The sentences were each a set of five words with no grammar or semantics relating to it. Once again, there was a diversity amongst test subjects' age, skin colour, gender, and body measurements. For reference, Figure A.4 – A.5 in Appendix A provides a picture of each of the 25 subjects. The subject pool consisted of signers that were thought to sign each sign according to the SASL dictionary prior to testing. Using the 50 SASL signs selected in the previous chapter, 10 sentences were composed, each consisting of five signs chosen out of the 50 signs, with each sign only appearing in one sentence. As such, each sentence was distinct. Table

5.1 details each of the 10 sentences in terms of the signs that were used to construct the sentence.

Sentence	1st Sign	2nd Sign	3rd Sign	4th Sign	5th Sign
1	Yes	Animal	Bye-bye	Fish	Come
2	Goose	Snake	Cut	Sorry	Shower
3	Dolphin	Go	Button	Hearing	Worm
4	Handbag	What	Son	Which	Money
5	Bell	Aeroplane	Fly	Girl	Flour
6	Salt	Juice	Daughter	Plum	Orange
7	Microwave	Naartjie	Sponge	Coke	Yellow
8	Monday	Hotel	Tuesday	Switch On	Me
9	Wednesday	Down	Switch Off	Mine	Thursday
10	Waiter	Kettle	Tall	Tap	Zoo

Table 5.1: The 10 sentences of 5 signs each used in experimentation.

5.1.2 Collection of Videos

The 25 subjects were asked to perform the 10 sentences. In each gesture the subject was required to hold their right hand up to initialize the tracker of the system and each of the five signs were then performed in succession. For each sentence, the subject was given 2 practice rounds and 1 final round to perform the sentence. With the use of 25 subjects, a total of 250 videos were recorded. This data set is henceforth referred to as the "sentence data set".

5.2 Tracking Accuracy

This section describes the experimentation carried out to assess the accuracy of the proposed tracking strategy. This assessment was done within the context of the continuous SASL gesture recognition system.

5.2.1 Experimental Procedure

One video was selected from the first 10 subjects to be used in this experiment. The videos were chosen such that all of the signs were represented. Table 5.2 summarizes the videos used in this experiment, as well as the total number of frames in each video. Each video was used as input to the tracking component of the proposed system. For each frame in the video, the resulting CamShift tracking window was determined by the system. The criterion for a successfully tracked frame was defined as a frame that contains a tracking window of size 120 × 100 that completely encloses the hand. Similar to Kolsh and Turk [65], the comparison was carried out by the researcher.

The total number of frames that were correctly tracked and incorrectly tracked were recorded in all cases.

Subject	Sequence	Signs	Total frames
1	1	1–5	646
2	2	6–10	592
3	3	11–15	690
4	4	16–20	683
5	5	21–25	679
6	6	26–30	703
7	7	31–35	692
8	8	36–40	627
9	9	41–45	609
10	10	46–50	599

Table 5.2: Data used in tracking accuracy testing.

5.2.2 Results and Analysis

Subject	Accuracy (%)
1	100.0
2	100.0
3	100.0
4	100.0
5	100.0
6	100.0
7	100.0
8	100.0
9	100.0
10	100.0

Table 5.3: Results of the tracking accuracy testing.

The results, summarized in Table 5.3 clearly demonstrate a consistently perfect tracking accuracy of 100.0% across all test subjects, sentences and signs. A large group of subjects was used to show robustness to variations in skin colour, body dimensions, gender etc. Furthermore, a relatively complex background was used. However, hand tracking was not the primary focus of this research and further testing in this regard may be carried out.

This result can be attributed to the fact that the signs in these sentences were not performed with abnormally rapid changes in speed. Extremely rapid changes in speed may cause a blurring effect in the image sequence which could affect the tracking accuracy. Visually, the signing in the sentences appears to be similar in speed to typical sign language communication. A further investigation may be carried out to confirm this. For the purpose of this research, it is important that the hand tracking facilitates the gesture recognition without having a negative impact on the gesture recognition aspect of the system. This has been shown to be the case. Therefore, it can be concluded that, for the purposes of this research, the selected tracking strategy is ideal.

5.3 Optimization of the Isolated Gesture Segmentation Model Thresholds

5.3.1 Experimental Procedure

An experiment was carried out to determine the optimal thresholds to be used in the isolated gesture segmentation model. One video each of 5 subjects from the sentence data set were used in this experiment. Table 5.4 summarizes the videos used in this experiment.

Subject	Sequence	Signs	Total frames
1	1	1–5	646
2	2	6–10	592
3	3	11–15	690
4	4	16–20	683
5	5	21–25	679

TABLE 5.4: Data used in tracking accuracy testing.

An experiment was carried out to optimize T_H and T_M jointly. In order to optimize T_H – the hold threshold which determines the point of transition from a movement into a hold – the values of this parameter were varied by decreasing it from 20 to 0 in increments of 5. For each T_H parameter value, the values of T_M – the movement threshold which determines the point of transition from a hold to a movement – were varied by increasing it from 0 to 30 in increments of 5.

It was, however, observed that T_M must always be at least T_H for the system to function appropriately. In this regard it should be noted that if $T_M < T_H$, when the system transitions from the initial hold state to a movement state, it will instantly transition back into the hold state since the hold threshold is less than the movement threshold. Thus, the model cannot function if $T_M < T_H$. Therefore, for each T_H value investigated, the T_M values investigated were in the range $T_H \leq T_M \leq 30$.

For each parameter pair, all 5 video sequences were used as input to the gesture segmentation component of the system. Two outcomes were recorded: the total True Positive (TP) count and the total False Positive (FP) count which will be explained.

Each video sequence was known to contain 5 signs, with each sign representing a hold. The first hold and movement of the video in which the signer holds his/her hand up to initialize the tracker, and subsequently moves to the first sign, are ignored. It was reasoned that what was relevant and desired was for the system to be able to detect and segment all the correct holds exactly once, with fully formed hand shapes in the correct locations, with no other additional (erroneous) detections.

Based on this, for each hold, the resulting segmented frame produced by the segmentation procedure was visually inspected. If it contained the correct hand shape and was at the correct location, it was deemed a TP. If it did not contain the correct hand shape, including cases in which the hand shape has not fully transitioned into the correct hand shape or was at the wrong location, or was a repetition of a hold for a gesture in the same location and hand shape, it was deemed a FP. Therefore, any segmented hold frames with any of the following qualities were considered FPs:

1. Hand was slightly out of location.
2. Hand shape was not fully formed.
3. Hand was in correct location and with fully formed shape, but this hold was detected in a previous frame i.e. it is a repeat detection.

The total number of TPs and FPs for each T_H and T_M parameter pair selection were recorded. The results were used to select the parameter pair that yielded the highest number of TPs out of a total of 25 true holds, and lowest number of FPs.

5.3.2 Results and Analysis

The results of the experiment are summarized in Table 5.5. As seen in the table, it is clear that at all parameter values, all 25 holds are correctly detected and segmented. However, the number of FPs varies. Therefore, this is used to make a selection of the optimal parameter value pair that yields the lowest number of FPs.

It is clear from the results that increasing the value of T_H generally results in a great increase in FPs. This is along with expectation since a higher T_H causes the segmentation model to detect a hold while the hand is still in motion and the gesture is not yet fully formed. The number of FPs is as high as 50 in some cases. A FP count as high as this

T_H	T_M	**True Positives (25)**	**False Positives**
20	20	25	50
	25	25	50
	30	25	50
15	15	25	45
	20	25	45
	25	25	45
	30	25	45
10	10	25	45
	15	25	45
	20	25	45
	25	25	40
	30	25	40
5	5	25	30
	10	25	30
	15	25	25
	20	25	25
	25	25	25
	30	25	25
0	0	25	50
	5	25	20
	10	25	10
	15	25	10
	20	25	0
	25	25	0
	30	25	0

TABLE 5.5: Optimization results for the hold threshold T_H.

can render the segmentation component unusable. The value of T_H which yields the lowest number of FPs is $T_H = 0$. At this value of T_H, it is apparent that, in contrast to the case of T_H, lower values of T_M yield a higher FP count. This is also along with expectation since a low threshold causes the system to be very sensitive to slight motions of the hand. This is as high as 50 FPs at a value of $T_M = 0$.

Increasing the value of T_M results in a decrease in the number of FPs and at $T_M \geq 20$, the ideal case of zero FPs detected is achieved. Therefore, it is observed that there is an option of three parameter pairs $\{T_H, T_M\}$ that yield an ideal TP count of 25 and FP count of 0: $\{0, 20\}$, $\{0, 25\}$ and $\{0, 30\}$. In order to select one parameter pair, it should be considered that selecting the lowest value of T_M for which the number of FPs is zero is of greatest advantage. This will ensure the highest sensitivity of the system to movements without registering any FPs.

Therefore, the optimal parameters selected were $T_H = 0$ and $T_M = 20$.

5.4 Accuracy Testing

In this section the gesture recognition accuracy of the proposed system is assessed. The experiment aims to convey the ability of the system to distinguish between a large vocabulary of signs using the two sign language gesture parameters.

5.4.1 Experimental Procedure

All the videos of all 25 subjects from the sentence data set were used in this experiment. Each video was used as input to the proposed gesture recognition system. For each video, the video was automatically segmented into isolated signs by the system. Each segmented isolated sign was processed to extract features from it which were used as input to the classification component, culminating in a prediction of the sign as being one of the 50 SASL signs in the vocabulary. For each sign, the result of prediction was recorded as a dichotomous outcome: correctly recognized or not correctly recognized. This procedure was chosen due to the large vocabulary size used.

These results were used to determine an overall per-sign accuracy, per-subject accuracy, as well as a per-sentence accuracy. The per-sentence accuracy was defined as the number of signs that were correctly recognized out of the total of 5 signs per sentence.

Before analyzing the results, it is important to consider the success rate of random guessing in place of the classifier under the conditions of this experiment as a base comparison. For each image of each sign, a classification into one of 50 classes is carried out. The success rate of a random guess for each image is then $\frac{1}{50}$ which is 2%.

Therefore, for all 1250 images across all signs and subjects, the success rate for each image for random guessing is 2%. Computing the average success rate across all the images results in an average success rate of 2% when guessing randomly in place of the classifier. Achieving an accuracy that is higher than this success rate is to be better than random guessing.

5.4.2 Results and Analysis

Table A.2 in Appendix A provides the comprehensive set of results obtained for this experiment. These results are summarized per sign in Table 5.6. In the table, the accuracy of each sign as a percentage of the total of 25 samples of each sign is presented. Figure 5.2 depicts these results graphically, sorted according to the accuracy obtained in descending order. The average sign recognition accuracy of the system is 74.1% across all signs and subjects. This is a very encouraging result considering the large vocabulary size and the variations in test subjects. Comparing this to the random success rate base case, it indicates a very high success rate by the classifier.

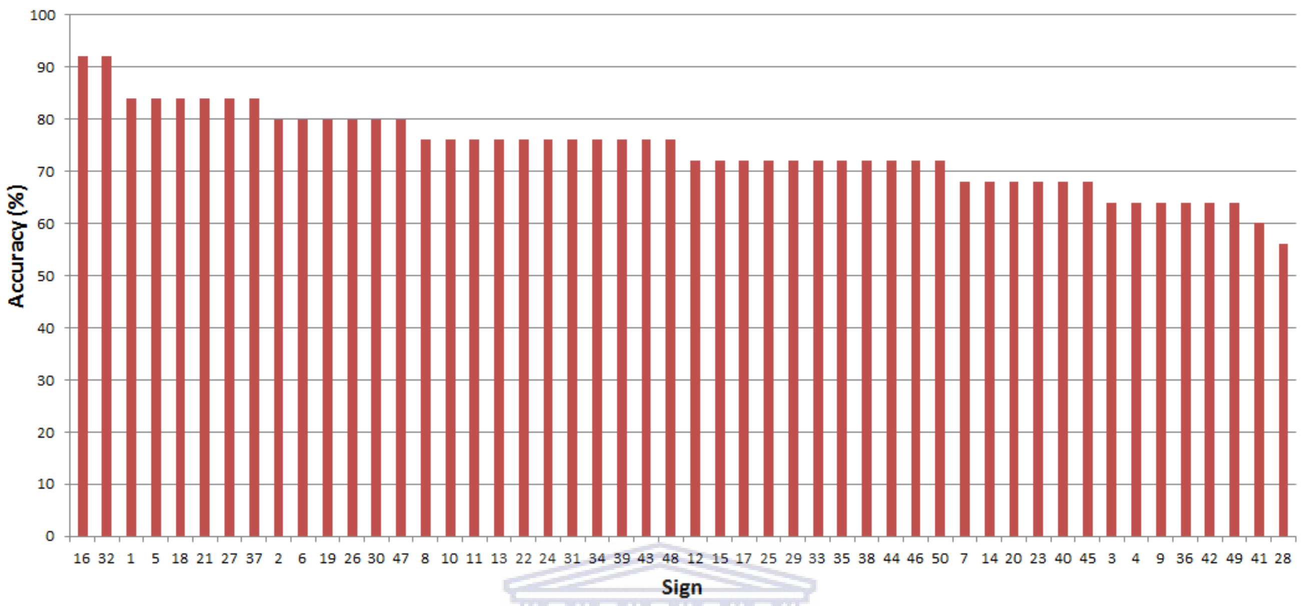


FIGURE 5.2: Accuracy for each of the 50 SASL signs, sorted in descending order according to accuracy.

As mentioned in Section 1.2 in Chapter 1, while a direct comparison of the proposed system with those of Naidoo and Rajah is not possible, a comparison in terms of accuracy and vocabulary size is carried out. The vocabulary size recognised by the proposed system is 50 signs. This is a vocabulary size that is 2.5 times larger than Naidoo's set, and 2.1 times larger than Rajah's set. It is at least double the size of each of these sets.

In spite of the vast increase in vocabulary size, the system recognizes signs at a higher accuracy than both of these systems. Rajah's system recognized signs at an average accuracy of 71% and Naidoo's average system accuracy was 69%. In comparison, the proposed system recognizes signs at an accuracy of 74.1%. As such, it can be firmly stated that research objective 3 set out in Chapter 1 has been achieved. A very significant advantage of this system over existing systems is the ability to identify and recognize signs within a sentence without context, recognizing the signs with a comparable accuracy to existing systems. The existing systems expect a video containing a single sign.

The following subsections provide a further analysis of the results.

Sign	**Accuracy (%)**	**Sign**	**Accuracy (%)**
1	84	26	80
2	80	27	84
3	64	28	56
4	64	29	72
5	84	30	80
6	80	31	76
7	68	32	92
8	76	33	72
9	64	34	76
10	76	35	72
11	76	36	64
12	72	37	84
13	76	38	72
14	68	39	76
15	72	40	68
16	92	41	60
17	72	42	64
18	84	43	76
19	80	44	72
20	68	45	68
21	84	46	72
22	76	47	80
23	68	48	76
24	76	49	64
25	72	50	72
(cont. right)		**Average**	**74.1**

Table 5.6: Accuracy per sign.

5.4.2.1 Per-Sign Accuracy

Observing Figure 5.2, it can be noted that the results ranged between 92% and 56%. It is extremely encouraging to note that no sign achieved an accuracy below 56%. A total of 49 of the 50 signs – 98% of signs – achieved an accuracy of 60% and above. Furthermore, a total of 14 of the 50 signs – 28% of the signs – achieved a very high accuracy of 80% and above and 36 of the 50 signs – 72% of the signs – achieved a high accuracy of 70% and above. Only one sign achieved an accuracy below 60%. This was an accuracy of 56%. Considering that this accuracy is 28 times higher than the random guessing success rate accuracy base case, it is clear that this result can by no means be considered low but rather a very commendable result. This indicates that the proposed system is highly successful and accurate in spite of the large vocabulary used.

It is also noted that the standard deviation amongst the signs is only 7.68%. This is a very small standard deviation and expresses the consistency and robustness of the

Sentence	Signs	Correct (125)	Accuracy (%)
1	1–5	94	75.2
2	6–10	91	72.8
3	11–15	91	72.8
4	16–20	99	79.2
5	21–25	94	75.2
6	26–30	93	74.4
7	31–35	97	77.6
8	36–40	91	72.8
9	41–45	85	68.0
10	46–50	91	72.8
Average		**92.6**	**74.1**

Table 5.7: Accuracy per sentence.

proposed gesture recognition strategy across variations in a large vocabulary of signs.

5.4.2.2 Per-Sentence Accuracy

The accuracy per sentence was computed by determining the number of signs that were correctly recognized in all videos of each sentence used as input to the system. Each sentence was performed once by each of the 25 subjects. Each sentence contained 5 signs. Therefore, the total accuracy per sentence is determined as the number of correctly recognized signs out of a total of 125 signs per sentence, across all videos. Table 5.7 summarizes the accuracy of each sentence. The table includes the total number of signs correctly identified out of a total of 125 samples, as well as the corresponding percentage accuracy.

The per-sentence accuracy ranged from 68.0% and 79.2% with a standard deviation of 3.04%. With the exception of Sentence 9, all of the sentences achieve accuracies above 72%. Sentence 9 also achieves an accuracy of 68% that is very close to 70%. Therefore, the accuracies of all the sentences are very close to the average of 74.1% and no outliers are observed.

Table 5.8 depicts the number of videos of each sentence for each level of sign recognition. The level of sign recognition pertains to the number of signs (out of 5) of each video that were correctly recognized. It should be noted, once again, that each of the 10 sentences were performed once by each of the 25 subjects, hence, a total of 25 videos per sentence. For example, the second column of the table demonstrates that, out of the 25 videos of sentence 1: 3 of the videos had all 5 signs in the sentence correctly recognized; 13 of the videos had 4 out of 5 signs in the sentence correctly recognized; and 9 of the videos had

3 out of the 5 signs correctly recognized. None of the videos of sentence 1 had any less than 3 of the 5 signs in the sentence correctly recognized.

It is observed from the table that a total of 63% of the videos of the sentences achieve at least 4 out of 5 correct sign recognition in a sentence. A total of 95% of the videos of the sentences achieve at least 3 out of 5 correct sign recognition in a sentence. Only a small percentage of sentences – 5% – achieve less than 3 out of 5 correct sign recognition in a sentence. It is important to note that there were no sentences in which all of the signs were unrecognized, hence the final row in the table is consistently 0. This result directly reflects the per-sign accuracy of 74.1% which dictates that, on average, every sentence has at least 1 sign and at most 2 signs incorrectly recognized, but the majority of sentences should have only 1 sign incorrectly recognized. This is the trend observed in the table.

These results collectively demonstrate clearly that the segmentation model of the system works very well. The result also shows that the system is highly robust to variations in sentences and that the segmentation model selected is very appropriate. This is a very significant milestone for the SASL group towards a system that can translate entire SASL sentences into English.

5.4.2.3 Per-Subject Accuracy

The accuracy per subject was computed similarly to the accuracy per sentence. Each subject performed each of the 50 signs once. Therefore, the accuracy per subject was computed as the number of correctly recognized signs out of the total of 50 signs performed by each subject. Table 5.9 summarizes the accuracy achieved by each subject.

The accuracies ranged between 68% and 84% with a standard deviation of 6.2%. It is very encouraging to note that the per-subject accuracies are all 68% and above. The small standard deviation evidences the fact that variations in test subjects are handled very well by the system. The system is highly robust to such variations.

Signs	**Sentence**										**Total**	**Total**
Correct (5)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**(250)**	**(%)**
5	3	3	2	5	2	3	5	5	1	5	34	14
4	13	11	13	14	15	16	14	8	10	9	123	49
3	9	10	9	6	8	4	5	10	12	8	81	32
2	0	1	1	0	0	0	0	2	2	3	9	4
1	0	0	0	0	0	2	1	0	0	0	3	1
0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE 5.8: Number of videos of each sentence per level of correct sign recognition.

Signer	**Correct (50)**	**Accuracy (%)**
1	39	78
2	36	72
3	37	74
4	41	82
5	37	74
6	34	68
7	42	84
8	32	64
9	37	74
10	35	70
11	40	80
12	34	68
13	39	78
14	38	76
15	41	82
16	31	62
17	36	72
18	35	70
19	39	78
20	40	80
21	35	70
22	32	64
23	37	74
24	37	74
25	42	84
Average	37.05	74.1

TABLE 5.9: Accuracy per subject.

Two points should be noted to fully appreciate this result. The first point to note is the very large number of test subjects used and the variations, in terms of skin colour, body dimensions, gender etc. amongst the test subjects in the data set. The second point to note is the random guessing accuracy base comparison result. Taking these two factors into account, the ability of this system to produce highly robust, consistent and accurate results is evidence of a robust gesture recognition system as a whole.

It can therefore be stated that the final research objective set out in Chapter 1 has been achieved successfully.

5.5 Conclusion

In this chapter, the experiments carried out to assess the accuracy of the proposed gesture recognition system and its components were described. Ultimately, the aim was to achieve the two remaining objectives set out in Chapter 1, with the first two objectives achieved in the previous chapter, and finally answer the research question.

The tracking accuracy of the proposed system was tested and it was found that the system achieves a perfect accuracy of 100.0% under the conditions in which the proposed system is used. It was concluded that an appropriate tracking strategy has been selected.

The two motion thresholds T_M and T_H of the proposed isolated gesture segmentation model were optimized with respect to the objective of correctly segmenting continuous sign sequences without any erroneous detections. It was found that the selection $T_H = 0$ and $T_M = 20$ were ideal.

Finally, an experiment was carried out to assess the gesture recognition accuracy of the proposed system. It was shown that the system is highly accurate, achieving an average recognition accuracy of 74.1% in spite of the large vocabulary. It was demonstrated that, in spite of the fact that the vocabulary size is at least twice the size of those of previous systems, the proposed system achieves a higher accuracy than previous systems. It was also shown that the system is highly robust to variations in signs, sentences and test subjects, with very small standard deviations between signs, sentences and test subjects and without any outliers observed in any of these comparisons.

Therefore, in response to the research question posed in Chapter 1, it can be firmly stated that static SASL gestures can be continuously recognized at a high accuracy for a person of any skin tone from a monocular video feed using a combination of two sign language parameters – hand shape and hand location.

Chapter 6

Conclusion

This research has made several important contributions to the field of sign language gesture recognition, specifically South African Sign Language gesture recognition.

The most significant contribution of this system was the integration of two sign language gesture parameters to recognize a large vocabulary of sign language gestures. This is a very significant milestone towards the provision of a usable phrase book to the Deaf. It is also a significant milestone towards a recognition system that includes all five sign language parameters to accurately recognize the entire set of sign language gestures. This work provides a crucial framework in which to integrate additional parameter recognition components towards this goal.

In addition to this contribution, a second crucial contribution was the use of a segmentation model to enable the system to continuously recognize isolated gestures. To-date, all of the SASL gesture recognition systems have aimed towards recognizing single signs at a time. Such systems can, at best, make for semi-synchronous communication. This research has pioneered the framework which can be used to provide a real-time synchronous translation service between SASL and English.

Variations in test subjects was previously shown to be a significant challenge in the field of gesture recognition in the literature. The proposed continuous gesture recognition strategy was shown to be robust to variations in test subjects. This was, therefore, another crucial contribution to the field of gesture recognition. This is especially commendable given the extreme variations in the body dimensions, skin tone, gender, age etc. of test subjects used.

It is hoped that this research can benefit other researchers in the field of gesture and sign language recognition.

6.1 Directions for Future Work

This section is divided into three directions for future work.

6.1.1 Two-Handed Gesture Recognition

Although the proposed system was shown to be highly robust and useful for one-handed signs, it is required that support for tracking both hands simultaneously is included to recognize two-handed gestures. Tracking both hands is a complex field and requires for detection and resolution of occlusions of the hands. This should be investigated in future.

6.1.2 The Inclusion of More Parameters

The proposed system showed that the integration of two sign language gesture parameters made it possible to achieve high accuracy recognition on a significantly larger sign language gesture dataset than previously possible. It is desired that the remaining parameters – hand orientation, hand motion and facial expressions – also be included into the proposed framework. With the inclusion of all five of these parameters, it is expected that the vocabulary size that is recognizable at a high accuracy can significantly increase to include all SASL signs.

6.1.3 Mobile Implementation

It is desired that this system be ported to and implemented on a mobile platform. It is desired that the feature extraction process takes place on the mobile platform. This has the potential of turning the mobile phone into a mobile isolated sign language gesture translator. This can be used similar to a phrase book in day-to-day situations and can prove very beneficial to the Deaf as a stand-alone application.

6.2 Concluding Remarks

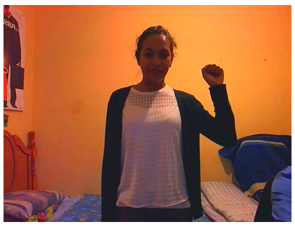
This researcher has gained extensive experience throughout the period of research. With the implementation of this research, a great appreciation of the possibilities and capabilities of computer vision has developed in the researcher. It is desired that this research will serve as an aid to further advancements and developments in the field of sign language gesture recognition within the SASL project and in the field of gesture recognition as a whole.

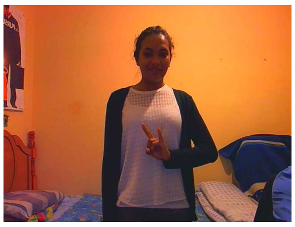
Appendix A

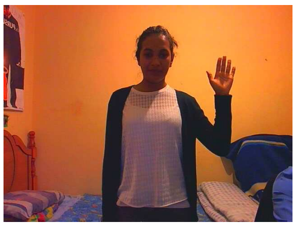
Additional Test Results

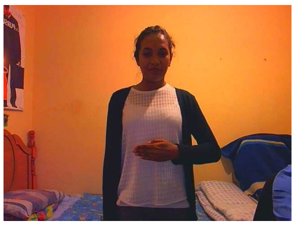
Shape	Right Shoulder	Mid Chest	Right Side of Chest	Right Side of Face	Left Side of Chest	Above Head
A- Shaped	(16) Handbag	(44) Mine	(33) Sponge	(9) Sorry	(18) Son	
Flat Hand	(3) Bye-Bye	(4) Fish	(12) Go	(11) Dolphin		
Mid+Ring on thumb		(2) Animal	(38) Tuesday			
Flat O	(21) Bell	(20) Money		(6) Goose		
1-Shaped Slightly Bent	(15) Worm	(40) Me				
Y-shaped	(35) Yellow	(22) Aeroplane	(19) Which			
Number 1	(31) Microwave	(42) Down	(17) What			
F-shaped		(13) Button	(28) Daughter			
Bent-5 up	(30) Orange	(32) Naartjie				
O-shaped	(39) Switch On	(29) Plum				
Thumbs-Up	(26) Salt	(27) Juice				
Palm-up flat hand		(25) Flour				
Bent-5 Down	(24) Girl	(49) Tap				(10) Shower
Large C		(34) Coke				
Bent V		(7) Snake				
L-shaped				(14) Hearing		
Bent Closed Flat hand	(5) Come			(48) Tall		
Flat F	(37) Hotel		(36) Monday	(23) Fly		
V-shaped			(8) Cut			
Tilted Y-shaped			(47) Kettle			
S-shaped	(1) Yes					
r-shaped	(50) Zoo					
Mid+thumb	(41) Wednesday					
Pinky+thumb	(45) Thursday					
Fist	(43) Switch Off					
Flat Palm	(46) Waiter					

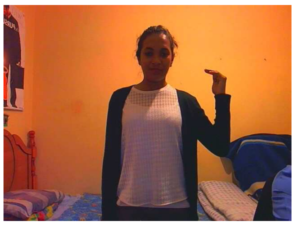
Table A.1: The 50 SASL signs used in experimentation.

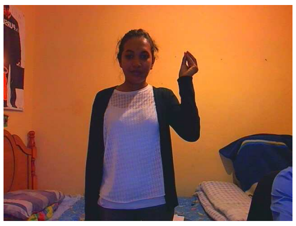

(a) Sign 1

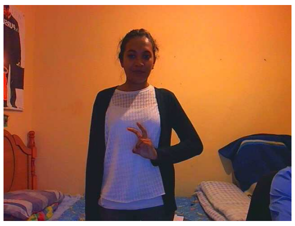

(b) Sign 2

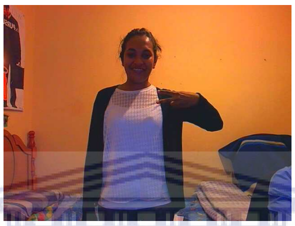

(c) Sign 3

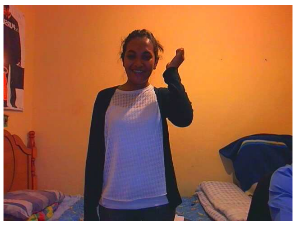

(d) Sign 4

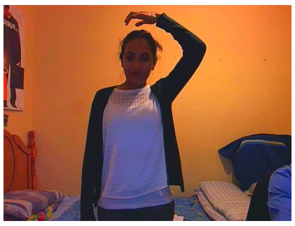

(e) Sign 5

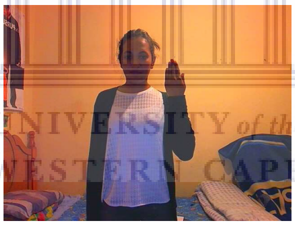

(f) Sign 6

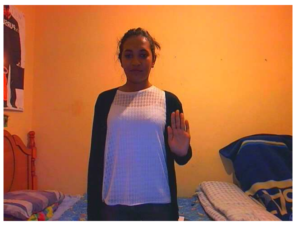

(g) Sign 7

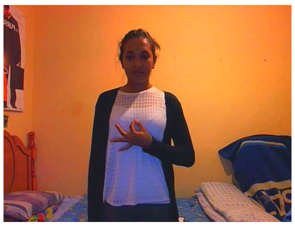

(h) Sign 8

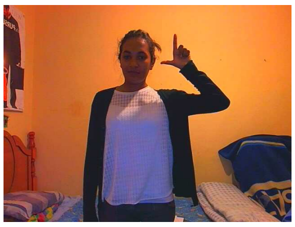

(i) Sign 9

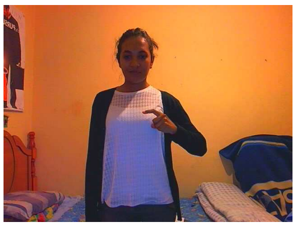

(j) Sign 10

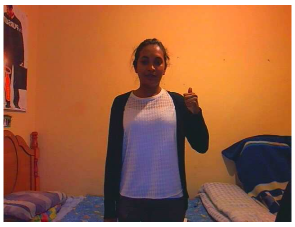

(k) Sign 11

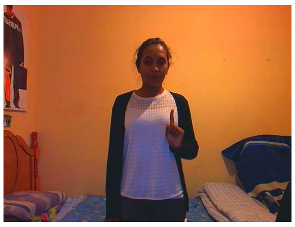

(l) Sign 12

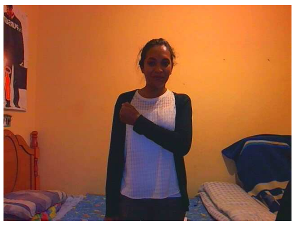

(m) Sign 13

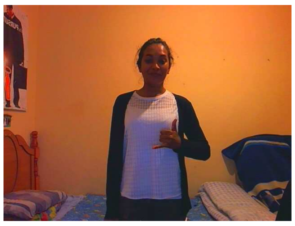

(n) Sign 14


(o) Sign 15


(p) Sign 16


(q) Sign 17


(r) Sign 18


(s) Sign 19

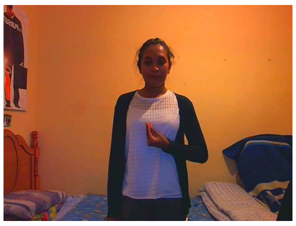
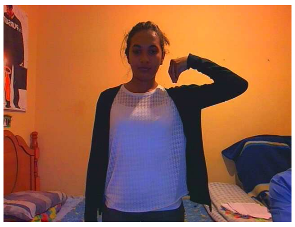
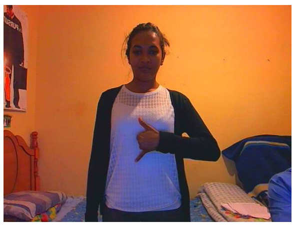
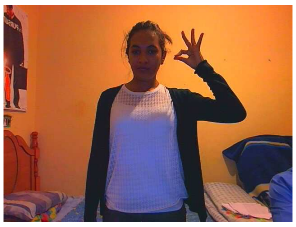
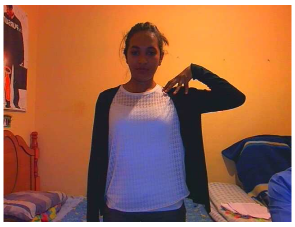

(t) Sign 20

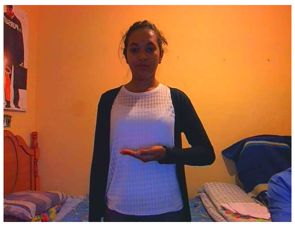
FIGURE A.1: Illustration of SASL signs 1 – 20 used in experimentation.

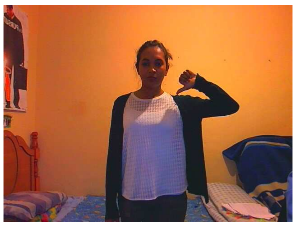

(a) Sign 21

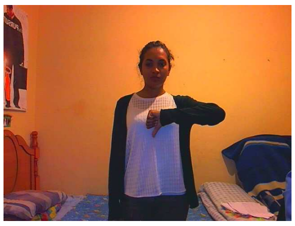

(b) Sign 22

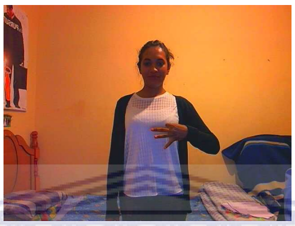

(c) Sign 23

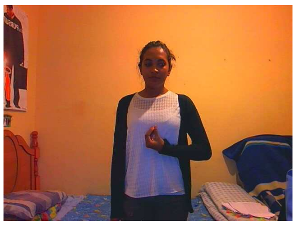

(d) Sign 24

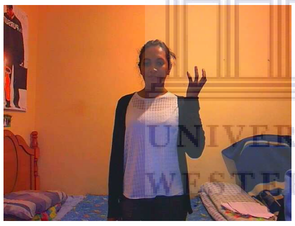

(e) Sign 25

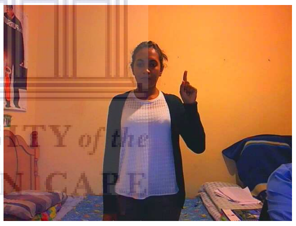

(f) Sign 26

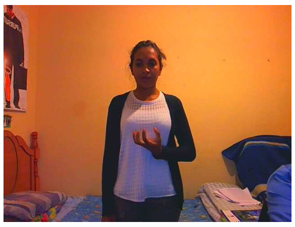

(g) Sign 27

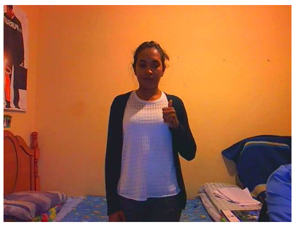

(h) Sign 28

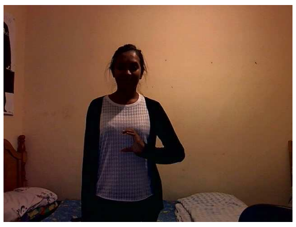

(i) Sign 29

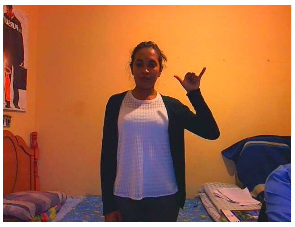

(j) Sign 30

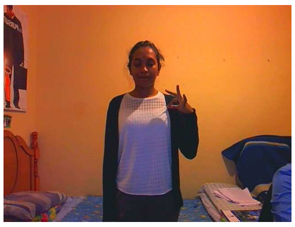

(k) Sign 31

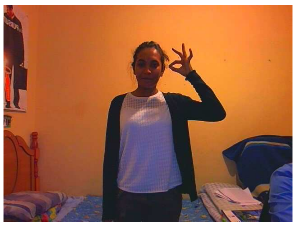

(l) Sign 32

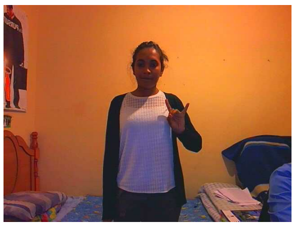

(m) Sign 33

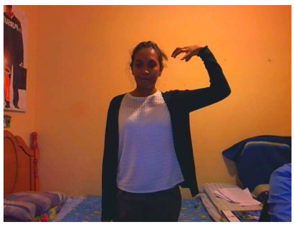

(n) Sign 34


(o) Sign 35


(p) Sign 36


(q) Sign 37


(r) Sign 38


(s) Sign 39

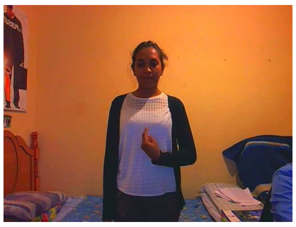
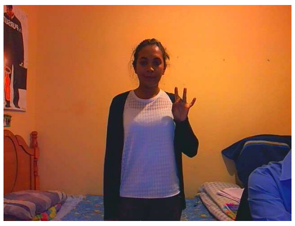
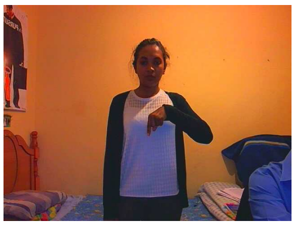
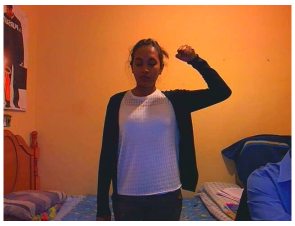
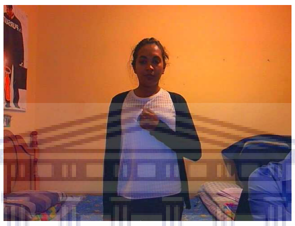

(t) Sign 40

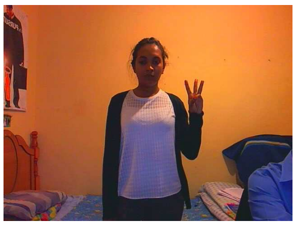
FIGURE A.2: Illustration of SASL signs 21 – 40 used in experimentation.

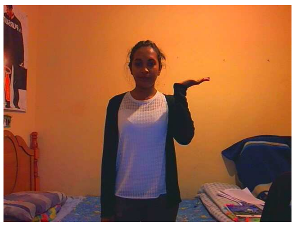

(a) Sign 41

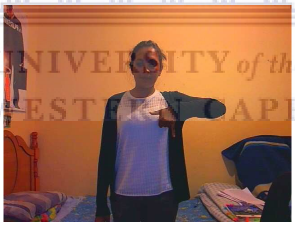

(b) Sign 42

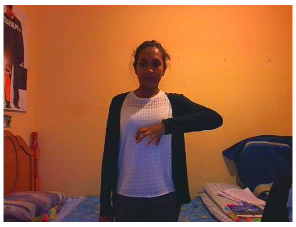

(c) Sign 43

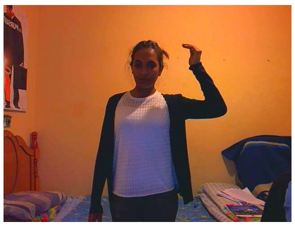

(d) Sign 44


(e) Sign 45


(f) Sign 46


(g) Sign 47


(h) Sign 48


(i) Sign 49

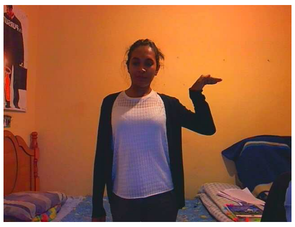
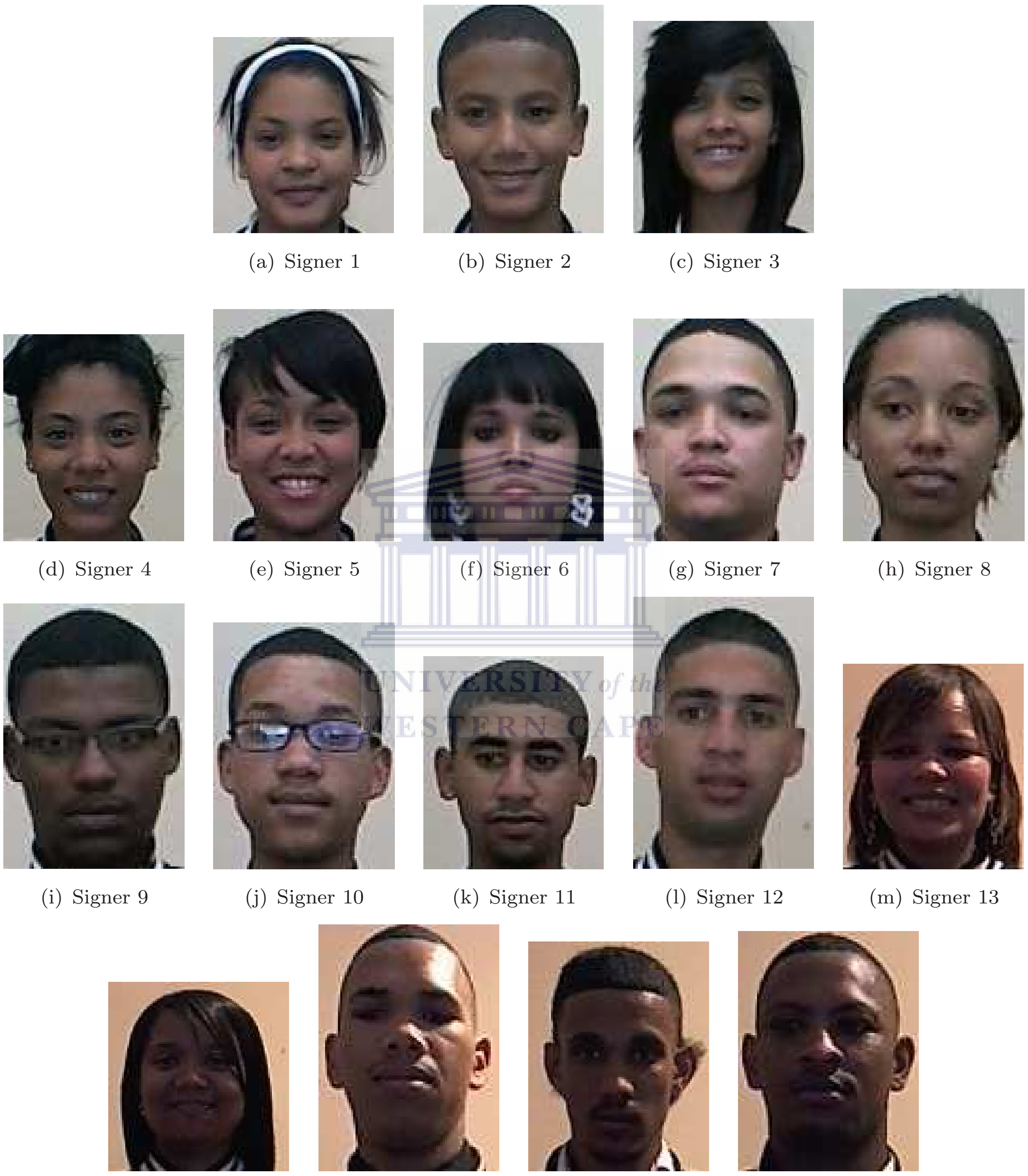

(j) Sign 50

Figure A.3: Illustration of SASL signs 41 – 50 used in experimentation.



(a) Signer 1 (b) Signer 2 (c) Signer 3

(d) Signer 4 (e) Signer 5 (f) Signer 6 (g) Signer 7 (h) Signer 8

(i) Signer 9 (j) Signer 10 (k) Signer 11 (l) Signer 12 (m) Signer 13

(n) Signer 14 (o) Signer 15 (p) Signer 16 (q) Signer 17

FIGURE A.4: Test subjects 1 – 17 used in experimentation.

(a) Signer 18 (b) Signer 19 (c) Signer 20 (d) Signer 21 (e) Signer 22

(f) Signer 23 (g) Signer 24 (h) Signer 25

FIGURE A.5: Test subjects 18 – 25 used in experimentation.

	Sign																																																	
Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
1	1	0	1	1	1	1	1	1	0	1	1	1	1	0	0	1	0	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	1
2	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	0	1	0	1	1	0	1	0	1	1	0	1	1	0	1	0	1	0	1	1	0	1	1	1	0	1	1
3	1	1	1	0	0	1	1	0	0	1	1	0	1	0	1	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1
4	1	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1
5	1	0	0	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	0	1	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	0	1	1	1	0	1	1	0	1
6	1	1	0	1	1	1	0	1	1	0	1	1	1	0	1	0	1	1	1	1	1	0	1	1	1	1	0	1	1	0	1	0	0	0	0	1	1	0	1	1	1	1	1	0	1	1	0	1	0	1
7	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	0
8	1	1	0	0	1	0	1	0	1	1	1	0	1	1	0	1	0	1	0	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	0	1	1	0	1	1	0	0
9	0	1	1	0	1	0	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	0	1	0	1	1	1	0	1	0	1	1	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1
10	1	1	0	1	1	1	1	1	0	1	0	1	0	1	1	1	0	1	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	1	0	0	1	1	0	0
11	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	1	0	1	0	1	1	1	1	1
12	1	0	1	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	0	1
13	1	1	1	0	1	0	1	1	0	1	1	1	0	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0	1	0
14	0	1	1	0	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	1	0	0	1	0	1	1
15	1	1	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	0	1	1	1	0	1	1	1	1	0	1	1	1	1	1
16	1	0	0	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	0	1	0	0	0	1	1	0	1	1	1	1	0	1	1	1	0	1	0	1	0	1	1	0	0
17	1	1	1	0	1	1	0	0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	1	1	1	0	1	0	1	1	1	0	0	1	1	1	0	0	1	1	1	1	1	1	0	1	0
18	1	1	0	1	1	0	1	1	0	0	0	0	1	1	0	1	1	0	1	1	1	1	1	0	1	1	1	0	1	1	1	1	0	1	1	1	1	0	1	1	1	1	0	0	1	1	0	1	1	1
19	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	1	1	1	1	1	0	1	0	1	1	1	0	0	1	1	0	1	1	1	1	0	1
20	1	1	1	0	0	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	1	1	1
21	0	1	1	0	1	0	1	1	0	1	1	0	1	0	1	1	1	1	1	0	0	1	1	1	0	1	1	0	1	1	1	1	0	1	1	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1
22	1	1	1	1	0	1	0	1	1	0	1	1	1	0	0	1	1	0	1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0	0	1	1	1	0	1	1	1	0
23	0	1	0	1	1	1	1	1	1	0	1	1	0	1	1	1	0	1	1	0	1	1	0	1	0	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	1	0	1	1
24	1	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	1	1	1	0	1	1	0	1	1	1	0	1	1	1	1	1	0	1	0	0	1	1	0	1	0	1	1	0	1	1	1	0	1
25	1	1	1	1	0	1	0	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	0	1

Table A.2: Results of the gesture recognition accuracy experiment. Correct recognition is represented by '1' and incorrect recognition is represented by '0'.

Bibliography

[1] R. Abdel-Kader, R. Ramadan, and R. Y. Rizk, "Rotation invariant face recognition based on hybrid lpt/dct features," *International Journal of Electrical and Computer Engineering*, vol. 3, no. 7, pp. 488–493, 2008.

[2] I. Achmed, "*Upper body pose recognition and estimation towards the translation of South African Sign Language*," Master's thesis, University of the Western Cape, Computer Science, 2010.

[3] J. Ahn and J. Kim, "A stable hand tracking method by skin colour blob matching," *International Journal of Computer Science and Engineering*, vol. 02, no. 3, pp. 560–565, 2010.

[4] M. S. Anderson and L. Vandenberghe, "Support vector machine training using matrix completion techniques," 2010, [Online] Available at http://www.seas.ucla.edu/~vandenbe/publications/svmcmpl.pdf.

[5] S. Anila and N. Devarajan, "Simple and fast face detection system based on edges," *International Journal of Universal Computer Sciences*, vol. 1, no. 2, pp. 54–58, 2010.

[6] G. Anthony, H. Gregg, and M. Tshilidzi, "An svm multiclassifier approach to land cover mapping," [Online] Available at http://arxiv.org/pdf/1007.1766.

[7] R. Araki, S. Gohshi, and T. Ikenaga, "Real-time both hands tracking using camshift with motion mask and probability reduction by motion prediction," in *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, 2012, pp. 1–4.

[8] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002.

[9] S. Askar, Y. Kondratyuk, K. Elazouzi, P. Kauff, and O. Schreer, "Vision-based skin-colour segmentation of moving hands for real-time applications," in *Visual Media Production, 2004. (CVMP). 1st European Conference on*, 2004, pp. 79–85.

[10] M. Assan and K. Grobel, "Video-based sign language recognition using hidden markov models," in *Gesture and Sign Language in Human-Computer Interaction*, ser. Lecture Notes in Computer Science, I. Wachsmuth and M. Frhlich, Eds. Springer Berlin Heidelberg, 1998, vol. 1371, pp. 97–109. [Online]. Available: http://dx.doi.org/10.1007/BFb0052992

[11] S. S. B. Klare, "Background subtraction in varrying illuminations using an ensemble based on an enlarged feature set," 1994, [Online] Available at http://www.cse.msu.edu/~klarebre/docs/OTCBVS09.pdf.

[12] S. Baker and S. Nayar, "Pattern rejection," in *Proceedings of Computer Vision and Pattern Recognition*, 1996, pp. 544–549.

[13] N. Bauer, P. Pathirana, and P. Hodgson, "Robust optical flow with combined lucas-kanade/horn-schunck and automatic neighborhood selection," in *Information and Automation, 2006. ICIA 2006. International Conference on*, 2006, pp. 378–383.

[14] S. Belgacem, C. Chatelain, A. Ben-Hamadou, and T. Paquet, "Hand tracking using optical flow embedded particle filter in sign language scene," 2012, [Online] Available at http://link.springer.com/chapter/10.1007%2F978-3-642-33564-8_35#page-1.

[15] A. Ben-hur and J. Weston, "A user's guide to support vector machines," 2012, [Online] Available at http://pyml.sourceforge.net/doc/howto.pdf.

[16] Y. Benezeth, P. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Comparative study of background subtraction techniques," *Journal of Electronic Imaging*, vol. 19, pp. 1342–1352, 2010.

[17] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, 2008.

[18] D. Brown, "*Faster Upper Body Pose Estimation and Recognition using CUDA*," Master's thesis, University of the Western Cape, Computer Science, 2012.

[19] Y. Cai, N. de Freitas, and J. J. Little, "Robust visual tracking for multiple targets," in *Proceedings of the 9th European Conference on Computer Vision - Volume Part IV*, ser. ECCV'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 107–118. [Online]. Available: http://dx.doi.org/10.1007/11744085_9

[20] G. Caridakis, O. Diamanti, K. Karpouzis, and P. Maragos, "Automatic sign language recognition: Vision based feature extraction and probabilistic recognition scheme from multiple cues," in *Proceedings of the 1st International*

Conference on PErvasive Technologies Related to Assistive Environments, ser. PETRA '08. New York, NY, USA: ACM, 2008, pp. 89:1–89:8. [Online]. Available: http://doi.acm.org/10.1145/1389586.1389687

[21] C. Chang and C. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[22] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, pp. 131–159, 2002.

[23] B. Cheng and D. Titterington, "Neural networks: A review from a statistical perspective," 1995, [Online] Available at http://www.msandri.it/reti_neurali/RNA2.pdf.

[24] Y. Cheung, "k-means: A new generalized k-means clustering algorithm," in *Pattern Recognition Letters*, 2003, pp. 2883–2893.

[25] A. Ching and C. Kamath, "Robust techniques for background subtraction," in *Proceedings of the SPIE*, 2004, p. 881.

[26] J. Choi, T. Whangbo, Y. Yang, M. Viswanathan, and N. Kim, "Pose-expression normalization for face recognition using connected components analysis," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 20, no. 06, pp. 869–881, 2006. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0218001406005010

[27] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, 2000, pp. 142–149 vol.2.

[28] C. Cortes and V. Vapnik, "Support vector networks," 1995, [Online] Available at http://www.rpi.edu/~bennek/class/mmld/papers/svn.pdf.

[29] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 2, 2001.

[30] F. Dadgostar and A. Sarrafzadeh, "An adaptive real-time skin detector based on hue thresholding," in *Proc. VIIth Digital Image Computing: Techniques and Applications*, 2003, pp. 419–427.

[31] F. Dadgostar and A. Sarrafzadeh, “An adaptive real-time skin detector based on hue thresholding: A comparison on two motion tracking methods,” *Pattern Recognition Letters*, vol. 27, no. 12, 2006.

[32] N. Degtyarev and O. Seredin, “Comparative testing of face detection algorithms,” [Online] Available at http://www.ican-sa.co.za/.

[33] K. Deguchi, O. Kawanaka, and T. Okatani, “Object tracking by the mean-shift of regional color distribution combined with the particle-filter algorithms,” in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3, 2004, pp. 506–509 Vol.3.

[34] J. R. del Solar, R. Verschae, and M. Correa, “Recognition of faces in unconstrained environments: A comparative study,” *EURASIP Journal on Advances in Signal Processing*, vol. 2009, 2009.

[35] L. Di Stefano and A. Bulgarelli, “A simple and efficient connected components labeling algorithm,” in *Image Analysis and Processing, 1999. Proceedings. International Conference on*, 1999, pp. 322–327.

[36] H. Ekenel and R. Stiefelhagen, “Analysis of local appearance-based face recognition: Effects of feature selection and feature normalization,” in *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW ’06. Conference on*, 2006, pp. 34–34.

[37] I. ElNaqa, Y. Yang, W. M.N., G. N.P., and R. Nishikawa, “A support vector machine approach for detection of microcalcifications,” in *Image Processing. 2002 International Conference*, 2002, pp. 953–956.

[38] A. Elgammal, C. Muang, and D. Hu, “Skin detection – a short tutorial,” IRP Report, 2007.

[39] P. Elinas, “Matching maximally stable extremal regions using edge information and the chamfer distance function,” in *Computer and Robot Vision (CRV), 2010 Canadian Conference on*, 2010, pp. 17–24.

[40] M. Elzemain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, “A hidden markov model-based isolated and meaningful hand gesture recognition,” *International Journal of Electrical and Electronic Engineering*, vol. 3, pp. 156–163, 2009.

[41] A. Faizi, “*Robust Face Detection using Template Matching Algorithm*,” Master’s thesis, University of Toronto, 2008.

[42] G. Fang, W. Gao, and D. Zhao, "Large-vocabulary continuous sign language recognition based on transition-movement models," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 37, no. 1, pp. 1–9, 2007.

[43] T. Finley and T. Joachims, "Supervised k-means clustering," 2005, [Online] Available at http://www.cs.cornell.edu/~tomf/publications/supervised_kmeans-08.pdf.

[44] C. C. for Welding and Joining. (1999) Image Analysis: Morphological Operations. [Online]. Available: http://www.ualberta.ca/~ccwj/teaching/image/morph/

[45] J. Forster, C. Oberdorfer, O. Koller, and H.Ney, "Modality combination techniques for continuous sign language recognition," in *IbPRIA*, 2013, pp. 89–99.

[46] H. Freeman and R. Shapira, "Determining the minimum-area encasing rectangle for an arbitrary closed curve," *Communication of the ACM*, vol. 18, no. 7, pp. 409–413, 1975.

[47] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *Information Theory, IEEE Transactions on*, vol. 21, no. 1, pp. 32–40, 1975.

[48] W. Gao, G. Fang, D. Zhao, and Y. Chen, "Transition movement models for large vocabulary continuous sign language recognition," in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, 2004, pp. 553–558.

[49] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, pp. 107–113, 1993.

[50] T. Heseltine, N. Pears, J. Austin, and Z. Chen, "Face recognition: A comparison of appearance-based approaches," in *7th Digitial Image Computing: Techniques and Applications*, 2003, pp. 59–68.

[51] J. Holt, *American Annals of the Deaf*, 8th ed. Stanford Achievement Test, 1993.

[52] P. Hong, M. Turk, and T. S. Huang, "Gesture modeling and recognition using finite state machines," in *In Proceedings of the Fourth IEEE International Conference and Gesture Recognition*, 2000, pp. 410–415.

[53] S. Howard, *Finger Talk: South African Sign Language dictionary.* South Africa: Mondi, 2008.

[54] C. Hsu, C. Chang, and C. Lin, "A practical guide to support vector classification," 2012, [Online] Available at http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

[55] M. S. Iraji, "Skin color segmentation in ycbcr color space with adaptive fuzzy neural network (anfis)," [Online] Available at http://www.mecs-press.org/ijigsp/ijigsp-v4-n4/IJIGSP-V4-N4-5.pdf.

[56] M. Isard and A. Blake, "Condensation–conditional density propogation for visual tracking," *International Journal for Computer Vision*, vol. 1, pp. 5–28, 1998.

[57] A. H. J. Owens and E. Fletcher, "A fast model-free morphology-based object tracking algorithm," 2002, [Online] Available at http://www.bmva.org/bmvc/2002/papers/99/full_99.pdf.

[58] R. Jafri and H. R. Arabnia, "A survey of face recognition techniques," *Journal of Information Processing Systems*, vol. 5, no. 2, pp. 41–67, 2009.

[59] T. Joachims, "Transductive inference for text classification using support vector machines," www.cs.cornell.edu/people/tj/publications/joachims_99c.pdf, 1998.

[60] E. J. R. Justino, F. Bortolozzi, and R. Sabourin, "A comparison of svm and hmm classifiers in the off-line signature verification," *Pattern Recogn. Lett.*, vol. 26, no. 9, pp. 1377–1385, Jul. 2005. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2004.11.015

[61] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A survey of skin-color modeling and detection methods," *Pattern Recognition*, vol. 40, pp. 1106–1122, 2007.

[62] A. Karlsson, "Classification of high resolution satelite images," Master's thesis, Chalmers University of Technology, 2003.

[63] Khulisani, "Disibility newslatter," Newsletter, April 2013, [Online] Available at http://www.ican-sa.co.za/.

[64] K. Kollreider, H. Fronthaler, and J. Bigun, "Real-time face detection using illumination invariant features," in *Proceedings of the 15th Scandinavian Conference on Image Analysis*, ser. SCIA'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 41–50. [Online]. Available: http://dl.acm.org/citation.cfm?id=1768615.1768622

[65] M. Kolsch and M. Turk, "Fast 2D hand tracking with flocks of features and multi-cue integration," in *In IEEE Workshop on Real-Time Vision for Human-Computer Interaction at CVPR*, 2004, p. 158.

[66] V. S. Kulkarni and D. S. Lokhande, "Appearance based recognition of american sign language using gesture segmentation," *International Journal of Computer Science and Engineering*, vol. 02, no. 3, pp. 560–565, 2010.

[67] V. Kyrki, "Integration of model-based and model-free cues for visual object tracking," in *3d, Proc. of the IEEE Int. Conf on Robotics and Automation (ICRA05*, 2005, pp. 1554–1560.

[68] P. Li, "*Hand shape estimation for South African Sign Language*," Master's thesis, University of the Western Cape, Computer Science, 2010.

[69] R.-H. Liang and M. Ouhyoung, "A real-time continuous gesture recognition system for sign language," in *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, 1998, pp. 558–567.

[70] N. Liu and B. C. Lovell, "Mmx-accelerated real-time hand tracking system," in *IVCNZ*, 2001, pp. 381–385.

[71] E. Maggio and A. Cavallaro, "Hybrid particle filter and mean shift tracker with adaptive transition model," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 2, 2005, pp. 221–224.

[72] R. Mapari and D. G. Kharat, "Hand gesture recognition using neural network," [Online] Available at http:/citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.300.7098.

[73] A. Maruch, "Talking with the hearing-impaired," January 2010, [Online] Available at http://www.deafsa.co.za/index-2.html.

[74] N. A. Ming-Hsuan Yang, David J. Kriegman, "Detecting faces in images: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[75] H. Q. Minh, P. Niyogi, and Y. Yao, "Mercer's theorem, feature maps, and smoothing," [Online] Available at http://math.stanford.edu/~yuany/publications/MinNiyYao06.pdf.

[76] D. Mohr and G. Zachmann, "Silhouette area based similarity measure for template matching in constant time," [Online] Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.175.2307.

[77] D. Mushfieldt, M. Ghaziasgar, and J. Connan, "Robust facial expression recognition in the presence of rotation and partial occlusion," in *SAICSIT Conf.*, 2013, pp. 186–193.

[78] J. J. Nadir Nourain Dawoud, Brahim Belhaouari Samir, "Fast template matching method based on optimized metrics for face localization," in *Proceedings of the International MultiConference of Engineers and Computer Scientist*, 2012.

[79] N. Naidoo, "*South African Sign Language recognition using feature vectors and Hidden Markov Models*," Master's thesis, University of the Western Cape, Computer Science, 2009.

[80] K. Nallaperumal, S. Ravi, C. Babu, R. Selvakumar, A. Fred, C. Seldev, and S. S. Vinsley, "Skin detection using color pixel classification with application to face detection: A comparative study," in *Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on*, vol. 3, 2007, pp. 436–441.

[81] K. Pan, S. Liao, Z. Zhang, S. Li, and P. Zhang, "Part-based face recognition using near infrared images," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1–6.

[82] J. Phu and Y. Tay, "Computer vision based hand gesture recognition using artificial neural network," [Online] Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.68.9148.

[83] M. Piccardi, "Background subtraction techniques: a review," in *IEEE International Conference on Systems, Man and Cybernetics*, 2004, pp. 3099–3104.

[84] N. Prabhakar, V. Vaithiyanathan, A. P. Sharma, A. Singh, and P. Singhal, "Object tracking using frame differencing and template matching," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 24, pp. 5491–5501, 2012.

[85] M. Pupilli and A. Calway, "Real-time camera tracking using a particle filter," in *In Proc. British Machine Vision Conference*, 2005, pp. 519–528.

[86] F. Quek, "Unencumbered gestural interaction," *MultiMedia, IEEE*, vol. 3, no. 4, pp. 36–47, 1996.

[87] J. Radoux and P. Defourny, "GIS-driven classification of satelite imagery," in *International Society for Photogrammetry and Remote Sensing*, 2010, pp. 1–6.

[88] C. Rajah, "*Chereme-based recognition of isolated, dynamic gestures from South African Sign Language with Hidden Markov Models*," Master's thesis, University of the Western Cape, Computer Science, 2006.

[89] A. Rakhlin and A. Caponnetto, "Stability of k-means clustering," 2007, [Online] Available at http://cbcl.mit.edu/publications/ps/rakhlin-stability-clustering.pdf.

[90] S. Rao and J. Hong, "Analysis of hidden markov models and support vector machines in financial applications," [Online] Available at http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-63.pdf.

[91] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted guassian mixture models," in *Digital Signal Processing*, 2000, pp. 19–41.

[92] H. Ribeiro and A. Gonzaga, "Hand image segmentation in video sequence by gmm: a comparative analysis," http://sibgrapi.sid.inpe.br/col/sid.inpe.br/sibgrapi@80/2006/06.26.11.48/doc/Ribeiro-HandImageSegmentationInVideoSequenceByGMM.pdf, 2006.

[93] G. Ros, J. del Rincon, and G. Mateos, "Articulated particle filter for hand tracking," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012, pp. 3581–3585.

[94] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002, pp. 23–38.

[95] S. Roy and S. Podder, "Face detection and its applications," *International Journal of Research in Engineering & Advanced Technology*, vol. 1, no. 2, pp. 1–10, 2013.

[96] O. Rumyantsev, M. Merati, and B. Ramachandran, "Hand sign recognition through palm gesture movement," 2012, [Online] Available at http://www.stanford.edu/~mamakar/mentorship/ee368_3.pdf.

[97] Y. Sato, M. Saito, and H. Koike, "Real-time input of 3d pose and gestures of a user's hand and its applications for hci," in *Virtual Reality, 2001. Proceedings. IEEE*, 2001, pp. 79–86.

[98] R. E. Schapire, "The strength of weak learnability," in *Foundations of Computer Science, 1989., 30th Annual Symposium on*, 1989, pp. 28–33.

[99] B. Scherrer, "Guassian mixture model classifiers," 2007, [Online] Available at http://www.music.mcgill.ca/~scherrer/MUMT611/a03/Scherrer07GMM.pdf.

[100] O. Schreer and S. Ngongang, "Real-time gesture recognition in advanced video-communication services," in *Image Analysis and Processing, ICIAP IEEE*, 2007, pp. 253–258.

[101] V. Segers, "The efficacy of the eigenvector approach to south african sign language identification," Master's thesis, 2009.

[102] A. Selinger and D. A. Socolinsky, "Appearance-based facial recognition using visible and thermal imagery: A comparative study," Tech. Rep., 2001.

[103] Z. Shaaban, "Face detection methods," in *Proceedings of the 11th WSEAS International Conference on Applied Computer Science*, ser. ACS'11. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2011, pp. 17–20. [Online]. Available: http://dl.acm.org/citation.cfm?id=2051254.2051259

[104] C. Shan, Y. Wei, T. Tan, and F. Ojardias, "Real time hand tracking by combining particle filtering and mean shift," in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, 2004, pp. 669–674.

[105] M. Sheikh, "*Robust Recognition of Facial Expressions on Noise Degraded Facial Images*," Master's thesis, University of the Western Cape, Computer Science, 2011.

[106] N. Shimada, K. Kimura, and Y. Shirai, "Real-time 3d hand posture estimation based on 2d appearance retrieval using monocular camera," in *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001. Proceedings. IEEE ICCV Workshop on*, 2001, pp. 23–30.

[107] M. Shin, K. Chang, and L. Tsap, "Does colorspace transformation make any difference on skin detection," in *Proceedings of 6th IEEE Workshop on Applications of Computer Vision*, 2002, pp. 275–279.

[108] W. Skarbek and A. Koschan, "Color image segmentation: A survey," University of Berlin, Tech Report, 1994.

[109] V. Spruyt, A. Ledda, and S. Geerts, "Real-time multi-colourspace hand segmentation," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 2010, pp. 3117–3120.

[110] V. V. Starovoitov, D. I. Samal, and D. V. Briliuk, "Three approaches for face recognition," in *6 th International Conference on Pattern Recognition and Image Analysis, Velikiy*, 2002.

[111] W. Stokoe, "Sign language structure: An outline of the visual communication systems of the american deaf," *Journal of Deaf studies and Deaf education*, vol. 10, no. 34, pp. 3–37, 2005.

[112] R. Stolkin, I. Florescu, M. Baron, C. Harrier, and B. Kocherov, "Efficient visual servoing with the abcshift tracking algorithm," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 2008, pp. 3219–3224.

[113] J.-C. Terrillon and S. Akamatsu, "Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images," in *Proc. of the12th Conf. on Vision Interface (VI '99*, 2000, pp. 180–187.

[114] J.-C. Terrillon and S. Akamatsu, "Explicit image detection using ycbcr space color model as skin detection," in *Applications of Mathematics and Computer Engineering*, 2011, pp. 123–128.

[115] F. Tomaz, T. Candeias, and H. Shahbazkia, "Improved automatic skin detection in color images," in *Proc. VIIth Digital Image Computing: Techniques and Applications*, 2003, pp. 419–427.

[116] L. Torresani and A. Hertzman, "Robust model-free tracking of non-rigid shape," 2003, [Online] Available at http://cs.nyu.edu/csweb/Research/TechReports/TR2003-840/TR2003-840.pdf.

[117] A. Tzotsos, "A support vector machine approach for object based image analysis," [Online] Available at http://www.ican-sa.co.za/.

[118] J. Umamabeswari and D. Radhamani, "Quadratic program optimization using support vector machine for ct brain image classification," *IJCSI International Journal of Computer Science Issues*, vol. 9, pp. 305–310, 2012.

[119] M. Vafadar and A. Behrad, "Human hand gesture recognition using motion orientation histogram for interaction of handicapped persons with computer," in *Proceedings of the 3rd International Conference on Image and Signal Processing*, ser. ICISP '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 378–385. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-69905-7_43

[120] V. Vezhnevets, V. Sazonov, and A. Andreeva, "A survey on pixel based skin color detection techniques," in *Proceedings of Graphicon*, 2003, pp. 85–92.

[121] P. K. Vijay, N. N. Suhas, C. S. Chandraskekhar, and D. K. Dhananjay, "Recent developments in sign language recognition: A review," in *Proceedings of the 8th Asian conference on Computer vision*, 2007, pp. 608–617.

[122] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, pp. 137–154, 2004.

[123] C. Vogler and D. Metaxas, "Handshapes and movements: Multiple-channel asl recognition," in *Lecture Notes in Computer Science*. Springer, 2004, pp. 247–258.

[124] C. Wang, X. Chen, and W. Gao, "Expanding training set for chinese sign language recognition," in *Proceedings of the 7th International Conference*

on Automatic Face and Gesture Recognition, ser. FGR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 323–328. [Online]. Available: http://dx.doi.org/10.1109/FGR.2006.39

[125] J. Whitehill, "*Automatic Real-time Facial Expression Recognition for Signed Language Translation*," Master's thesis, University of the Western Cape, Computer Science, 2006.

[126] B. Zarit, B. Super, and F. Quek, "Comparison of five color models in skin pixel classification," in *Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, 1999, pp. 58–66.

[127] C. Zhang, Y. Qiao, E. Fallon, and C. Xu, "An improved camshift algorithm for target tracking in video surveillance," in *9th IT & T Conference*, 2009, pp. 363–366.

[128] J. Zieren and K. friedrich Kraiss, "Robust person-independent visual sign language recognition," in *IbPRIA*, 2005, pp. 520–528.

[129] Z. Zivkovic, "Improved adaptive guassian mixture model for background subtraction," in *Proc. ICPR*, 2004.