OFFLINE CANDIDATE HAND GESTURE SELECTION AND TRAJECTORY DETERMINATION FOR CONTINUOUS ETHIOPIAN SIGN LANGUAGE

A thesis submitted to the school of Graduate Studies of Addis Ababa University in Partial fulfillment of the requirements for the Degree of Master of Science in Computer Engineering

By

Abadi Tsegay Weldegebriel

Advisor

Dr. Kumudha Raimond

October 2011

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

OFFLINE CANDIDATE HAND GESTURE SELECTION AND TRAJECTORY DETERMINATION FOR CONTINUOUS ETHIOPIAN SIGN LANGUAGE

By
Abadi Tsegay Weldegebriel

ADDIS ABABA INSTITUTE OF TECHNOLOGY
APPROVAL BY BOARD OF EXAMINERS

Chairman, Dept. of Graduate Committee

_____ _____
Signature

Dr. Kumudha Raimond _____
Advisor Signature

_____ _____
Internal Examiner Signature

_____ _____
External Examiner Signature

ACKNOWLEDGEMENTS

My sincere gratitude goes to my advisor Dr. Kumudha Raimond for her timely follow-ups, exhaustive support and very constructive suggestions until the end of this work. This thesis work could have not been completed without the effort of Dr. Raimonds. I would like to pass my appreciation to Yonas Fantahun for his advice and unreserved help. I am very grateful to Ato Eyasu, Fitsum, Kidane, Getahun, Henok and Muluneh for their help during my data collection stage.

I am also indebted to my friends and brothers Assefa Beyene and Alebachew Halefom for their constructive suggestions. My appreciation also goes to Birhanu Agegn, Anteneh Mekbib, Nega Agegn and Moges Birhanu for their continuous appreciation, valuable and unconditional suggestions and comments which were good inputs to the thesis work.

ABSTRACT

There is a clear communication gap between the deaf and hearing community. To bridge this gap, one possible solution is to teach the hearing community to use sign languages. However, a better solution is to develop a translation system that converts a continuous sign language gestures to text or speech. A lot of effort has been invested in developing alphabet recognition and continuous sign language translation systems for many sign languages around the world. In this regard, little attention has been given to Ethiopian sign language (EthSL). However, an Ethiopian Manual Alphabet (EMA) recognition system has been developed recently. For a recognition system that can recognize continuous gestures from video which can be used as a translation, a methodology that extracts candidate gestures from sequence of video frames and determines hand movement trajectories is required. In this thesis, a system that extracts candidate gestures for EMA and determines hand movement trajectories is proposed. The system has two separate parts namely Candidate Gesture Selection (CGS) and Hand Movement Trajectory Determination (HMTD). The CGS combines two metrics namely speed profile of continuous gestures and Modified Hausdorff Distance (MHD) measure and has an accuracy of 80.72%. *The HMTD is done by considering each hand gesture centroid from frame to frame and using angle, x- and y-directions. A qualitative evaluation of the CGS in a correctly divided video clip is found to be 94.81%. The HMTD has an accuracy of 88.31%. The overall system performance is 71.88%*

Keywords*: EMA, candidate gesture selection, CGS, trajectory determination, HMTD, Modified Hausdorff distance, MHD, Speed profile, search window.*

TABLE OF CONTENTS

Acronyms

AKF	Adaptive Kelman Filter
ASL	American Sign Language
ArSL	Arabic Sign Language
Auslan	Australian Sign Language
BD	Block Division
BSL	British Sign Language
BTD	Bayesian Tree of Decision
CSL	Chinese Sign Language
CGS	Candidate Gesture Selection
DD	Dominant Direction
EMA	Ethiopian Manual Alphabet
EthSL	Ethiopian Sign Language
fps	frames per second
GMM	Gaussian Mixture Model
HCI	Human Computer Interaction
HD	Hausdorff Distance
HMTD	Hand Movement Trajectory Determination
MHD	Modified Hausdorff Distance
NZSL	New Zealand Sign Language
PHD	Partial Hausdorff Distance
PME	Perceived Motion Energy
PSL	Persian Sign Language

ROI	Region Of Interest
SGM	Single Gaussian Model
SSL	Spanish Sign Language
TMOF	Temporally Maximum Occurrence Frame
TSL	Taiwan Sign Language

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Introduction

A considerable number of hearing impaired people live around the globe. As individuals, these people communicate with each other using sign languages. The sign languages that exist around the world are usually identified by the country where they are used. For example, American Sign Language (ASL), Australian Sign Language (Auslan), British Sign Language (BSL), New Zealand Sign Language (NZSL), Taiwan Sign Language (TSL), Chinese Sign Language (CSL), Ethiopian Sign Language (EthSL) and so on. Even if there may be some similarities between sign languages, it is usually considered that they are different.

Mostly, the communication among the deaf people involves signs that stand for words by themselves. But, to make the sign language complete as a spoken language, the deaf community around the world use manual alphabets for names, technical terms, and sometimes for emphasis. As there are different alphabets for different spoken languages such as English, Chinese, Greece, Ethiopia and so on, there are different types of manual alphabets or finger spellings used by the deaf people who use different sign languages.

1.2 Background of the problem

Sign language is a means of communication either between deaf people or between hearing people and the deaf community. For most of the communication the deaf community use, there is always a sign which stands for each word form such as noun, verb or adjective. However, whenever names, new terms and emphasis on any word are required, the finger spellings or manual alphabets are used. Therefore, even if the sign languages mostly use signs, the finger spelling is always important to make a sign language complete.

Extensive researches have been done on recognition of signs or finger spellings from static images. For example, Arabic Sign Language (ArSL) alphabet recognition system was built using polynomial classifiers as a classification engine for the recognition [7]. The researchers in [6] have also developed a system that recognizes alphabets in Persian Sign Language (PSL). EMA recognition system has also been developed in [5]. The research of sign language recognition is not limited to recognition from static images; it can also be applied to continuous pictures or videos of sign languages [1, 2, 3, 4].

In general, to bridge the communication gap between the deaf and the hearing people, a lot of effort has been put to produce a translation system that translates sign into spoken language and vice versa. In this regard, little attention was given to EthSL. However, in [5], the researchers have developed a recognition system which converts a given EMA into voice. In comparison to other sign languages such as ASL, CSL, TSL and etc, there is no research done which extracts candidate EMA gestures from continuous pictures or video frames along with the hand movement required to spell EMAs

What if someone wants to interface the recognition system with a system that extracts sequence of candidate EMAs from video clips which can be used as input gestures to the recognition system? What if someone still wants to extend the ability of the recognition system to recognize not only the base EMAs but also the other variations of an EMA? If an EMA recognition system is interfaced with a system just described, it will be possible to translate a given video clip of a word “written” with EMA to voice. So the main purpose of this thesis is to develop a technique which extracts important gestures called candidate gestures from a sequence of video frames and determines hand trajectories for each selected candidate gesture.

In EthSL, there are 34 base alphabets where each base alphabet has 6 other variations. In most sign languages other than EthSL, either there are alphabets that represent vowel sounds such as ***a, e, i, o*** and ***u*** of English language or there are limited back and forth movements to represent certain language accent as in Spanish Sign Language (SSL). However, in EthSL, when the

alphabets are created manually, the 6 variations are created with the same hand posture or form as the base alphabet followed by unique hand movement trajectories for each variation. Figure 1.1 depicts one of the EMAs with the hand movement style and direction for the six variations of the alphabet.

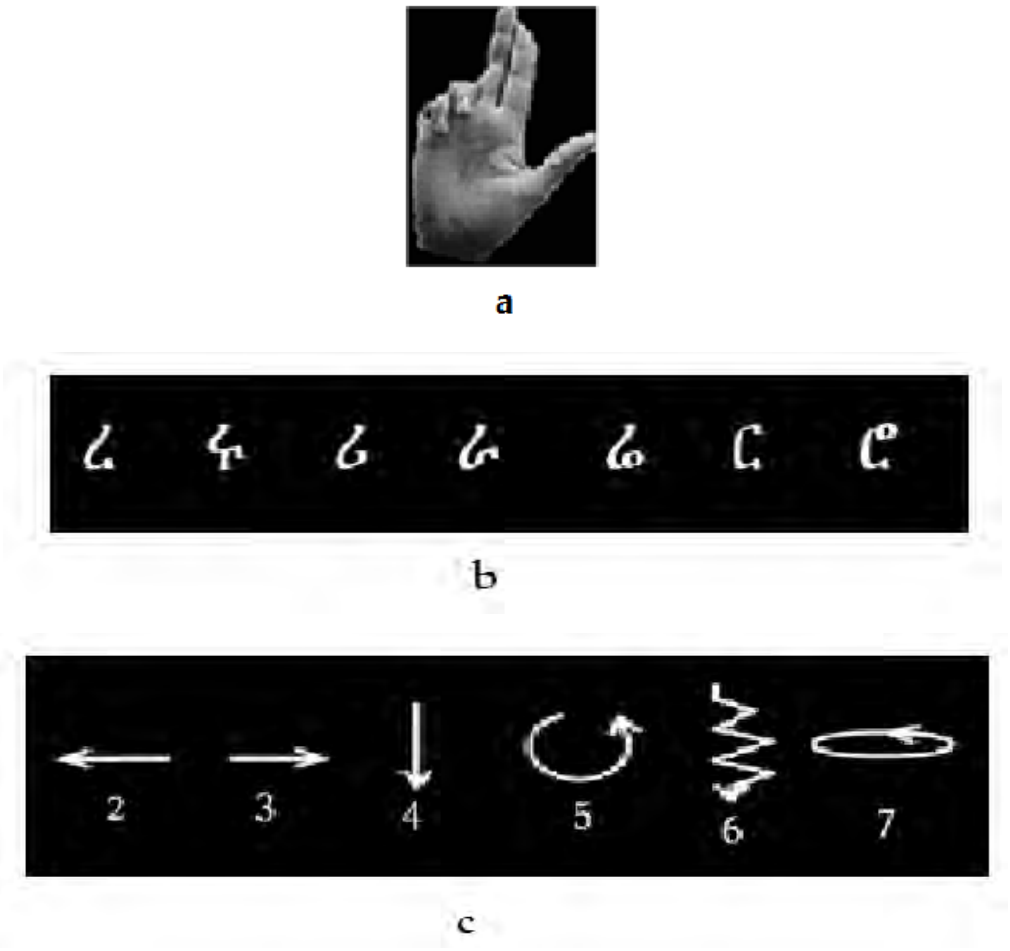


Figure 1.1 An EMA: a) Hand form for base EMA (ረ) b) The 7 forms of (ረ) c) Ideal trajectories for EMA

In Figure 1.1 (a), an EMA called (ረ) is shown while the 7 forms are depicted in (b). An ideal set of hand trajectories that are made when spelling the EMA forms other than the base EMA with a corresponding number is also shown in (c). In this report, the gestures are referred by the numbers shown below them in Figure 1.1 (c). For instance, if one wants to spell the fourth EMA, they show the base EMA and move their hand downwards with the same hand form.

1.3 Objective

The main objective of this thesis work is to design and implement a system that selects important candidate hand gestures for EMAs from a continuous EthSL and determines hand movement trajectories for each selected EMAs.

Specific objectives are:

1. Investigate existing other sign language translation systems for search of best practices implemented and choose an appropriate one for EthSL
2. Select and implement image pre-processing and segmentation techniques
3. Device a technique which successfully isolates the hand from the segmented binary image and selects representative gestures from a given sequence of video frames.
4. Develop a method that determines the hand motion trajectory for each selected EMA.
5. Measure qualitative and quantitative performance of the proposed design

1.4 Methodology

The methodology for the proposed design has the following parts:

- **Literature review: -** Papers, Journals and books related to each stage of the design are reviewed in this stage of the methodology.
- **Image processing: -** This is where sequences of image frames from video are processed to produce a binary image frame with only one hand. For every binary image there is a corresponding grayscale image that is picked as output of the system. Filtering, skin-color segmentation and morphological analysis are applied in this stage. An average filter is used to reduce the noise associated with each video frames.
- **Candidate hand gesture selection**: - When a given video is extracted into image frames, there are several consecutive image frames which are similar to one

another. For a sign language video recorded for one second with a video camera of 30 frames per second (fps), there will be 30 image frames when extracted. For example, if a signer was recorded for three seconds with the above video camera, making a three alphabet word, there will be 90 image frames from which 3 of them are required for recognition. It is therefore not technically acceptable to try to recognize the whole sequence of frames where only 3 of them can represent the given word. In this stage, a combination of hand gesture speed profile and MHD measurement is used to select candidate gestures from a given video clip. The search space for each EMA is localized to a smaller block than to set a threshold over the whole sequence. The output of the candidate hand gesture selection stage is a grayscale hand gesture equivalent to the binary gesture used in the selection process.

- **Hand trajectory determination**: - An EMA has a base alphabet and 6 variations created with hand movement trajectories. So before cropping the hand, centroid of each gesture relative to the image frame is collected for hand tracking. Hand trajectory is constructed from the centroids of the gestures and this trajectory will be examined using angle, x-direction and y-direction as a cue for its shape and orientation.
- **Performance evaluation**: - The qualitative and quantitative system performances are evaluated based on the candidate gesture selection and hand trajectory determination respectively.

1.5 Scope of the thesis

This thesis work is to propose a methodology of handling continuous video frames of EMAs to extract only important EMA gestures and determining hand movement trajectories for EMAs. EMAs except the base EMA and the last or 7^{th} form are considered in this work. The base alphabet is left out because the proposed design works based on speed profile between

successive gestures to divide the sequence of video frames into blocks while the base EMA is created using the gesture without any movement. As shown in Figure 1.1, the 7^{th} EMA is created by starting with the valid EMA and followed by a hand rotation. So it is not convenient to include this EMA because it just rotates around a fixed axis without moving to any of the directions. And this is not suitable to deal with speed profile.

1.6 Thesis contribution

As shown in Section 1.2 Figure 1.1 above, EMAs have unique characteristic when they are created using the hand. In this work candidate gestures are extracted based on a speed profile from a sequence of video frames and using a search window concept and determines the hand movement trajectory for each of the candidate EMAs selected from the sequence. The contributions of the thesis can be summarized as follows:

- This thesis work combines a gesture speed with the MHD measure for candidate hand gesture selection which avoids using thresholds.
- For hand isolation, in [43], the researchers have successfully developed a system that efficiently isolates a hand from a given image frame. However, they stated that their system should be informed whether the signing person is a right or left-handed. As improvement to system designed in [43], the proposed design in this thesis is able to isolate a hand gesture either for a right or a left-handed person without any prior information about the signing person.
- As explained in Section 1.2, there are 6 forms of each EMA other than the base EMA and each of the 6 forms are associated with a hand movement. A rule which discriminates the hand movement trajectories for the 5 forms is proposed.

1.7 Outline of the thesis

The thesis is organized as follows: in Chapter 2, review of research papers, projects and books is presented. Introductory concepts of digital image processing relevant to the methodology are

discussed in Chapter 3. A detail discussion of each stage of the proposed system is presented in Chapter 4. Experimental results and discussion of the results are presented in Chapter 5 where as conclusion and future works are presented in Chapter 6.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

Pattern recognition has long been studied and applied to different fields of study such as medicine, security, geographical information systems and even in business feasibility prediction. Another important application of pattern recognition is in sign language translation systems. Different signs or finger spellings of a sign language are usually done using the hands and hence pattern recognition has been used to recognize each sign or finger spelling using the patterns in the signs or finger spellings.

Traditionally, there have been two main approaches to sign language recognition: ***vision based*** and ***glove based***. In the vision based system, a signer performs the different signs which are then captured with a video camera. From the acquired video stream, the hand regions are segmented followed by a feature extraction stage and finally a classification stage [3]. In the glove based system, the signer wears some instrumented gloves equipped with a number of sensors which generate a set of electrical signals that characterize the intended sign.

The main advantage of glove based systems over vision based systems resides in the fact that there is no need for hand image segmentation, a process that is complicated and computationally expensive [3]. Another advantage of the glove based system is that camera is not required as in the vision-based. But, for practical application of translation systems, it is inconvenient to wear gloves. Researchers in [7] have used colored gloves to avoid the burden of skin segmentation, which in fact has the inconvenience of wearing gloves as in the glove based systems. And hence, the vision based approach is usually considered a good approach in computer vision and sign language translation systems.

Vision based sign language recognition systems usually involve several stages. After image acquisition, image processing is the first stage which includes the skin-color segmentation. CGS and HMTD are the subsequent stages.

2.2 Colors and color spaces

2.2.1 Introduction

Human skin detection systems are usually incorporated in human-computer interaction, surveillance cameras and robotics. In vision-based sign language manual alphabet recognition, hand postures are recognized first by skin color segmentation. Therefore, skin color segmentation is the key step where a robust and accurate skin color detection algorithm is required. This is because the subsequent steps largely depend on the quality of the segmented image. It is important to select the appropriate color space for the application at hand.

2.2.2 Color space

In the literature, color spaces have been studied very well. There are many color spaces where each color space has its own advantages and disadvantages compared to other color spaces. For skin color classification tasks, color space choice is usually considered as the primary step. The RGB color space is the default color space for most available image formats. Any other color space can be obtained from a linear or non-linear transformation from RGB. The color space transformation is assumed to decrease the overlap between skin and non-skin pixels which helps in a robust skin-pixel classification in a varying illumination conditions. It has been observed that skin colors differ more in intensity than in chrominance and this idea is used by [8] to obtain skin color separable and computational advantage by changing the problem from 3D color space to a 2D color space problem.

A wide variety of color spaces with different properties have been studied by researchers in skin modeling. The researchers have tried all these spaces in their study. The most popular among

these spaces are non-uniform color spaces like RGB, Normalized RGB (rgb), YCbCr, HSI, TSL and perceptually uniform color spaces like CIELAB and CIELUV. The choice of appropriate color space is often guided by the skin detection methodology and the application [9]. It is to be noted here that the evaluation of color space goodness for skin modeling cannot be performed because different modeling methods react very differently on the color space change. It is well known that the illumination conditions of the scene clearly affect the color of the objects in the scene. The goal of any color-based system is to minimize this influence to make color-based recognition robust to illumination changes. It seems that chrominance-only color analysis should make the system somewhat independent from the lighting conditions. Hence many researchers [10, 11] have dropped the luminance component in order to take computational advantage.

2.2.2.1 RGB and Normalized RGB

The RGB color space has been widely used for processing and storing digital image data. This model describes each color as a weighted combination of three base components Red, Green and Blue. However, high correlation between components and mixing luminance with chromaticity makes it very sensitive to changes in imaging conditions such as lighting [12]. Normalized RGB tries to reduce the dependence of each component to the brightness of the pixel by normalizing each component using:

$$r = \frac{R}{R + G + B} \tag{2.1}$$

$$g = \frac{G}{R + G + B} \tag{2.2}$$

$$b = \frac{B}{R + G + B} \tag{2.3}$$

In fact, since the sum of the normalized components is equal to 1, (r + g + b = 1), the third component does not hold any significant information. The simplicity of these color spaces has been the main reason for their popularity in skin color detection [12].

2.2.2.2 HSI, HSV, HSL

Hue Saturation Lightness (HSL) model describes color with dominant color (Hue), colorfulness in proportion to the brightness (Saturation), and the amount of luminance (Lightness). The most important characteristic of the model is its explicit discrimination of luminance from chrominance. This makes the model insensitive to brightness at white color and ambient light [12]. The model, however, has the disadvantage of being discontinuous at points where the brightness is very low (very dark points) [12]. H, S, and V components are computed using the following equations:

$$H = arccos \frac{\frac{1}{2}((R-G)+(R-B)}{\sqrt{(R-G)^2+(R-B)(G-B)}} \quad (2.4)$$

$$S = 1 - 3\frac{min\,(R,G,B)}{R+G+B} \quad (2.5)$$

$$V = \tfrac{1}{3}(R+G+B) \quad (2.6)$$

2.2.2.3 YCbCr

YCbCr is an encoded nonlinear RGB signal, commonly used by European television studios and for image compression work. Color is represented by *luma* (which is luminance, computed from nonlinear RGB, constructed as a weighted sum of the RGB values, and two color difference values Cr and Cb that are formed by subtracting *luma* from RGB red and blue components. In YCbCr, Y refers to the luminance component. The Cr and Cb components refer to the red and

blue chrominance, as how much each components deviate from gray [12, 13, 14]. In the YCbCr color space, Y reflects the luminance and is scaled to a range of 16 to 235. The chrominance components, Cb and Cr, are scaled versions of color differences B-Y and R-Y, respectively. Cb and Cr have a range of 16 to 240, inclusive [15].

The three components of YCbCr color space are computed as follows:

$$Y = 0.299R + 0.587G + 0.114B \quad (2.7)$$

$$C_r = R - Y \quad (2.8)$$

$$C_b = B - Y \quad (2.9)$$

The transformation simplicity and explicit separation of luminance and chrominance components makes this color space attractive for skin color modeling [13,16, 17, 18, 19].

It is a common task to select an appropriate color space for applications that involve color segmentation. Color space transformations have been used extensively in computer vision applications usually to gain computational advantage. As an RGB color space is a true color of objects, there is always an information loss when the RGB color space is converted to Normalized RGB, HSV, YCbCr etc. If a skin color is considered, the discrimination between skin and non-skin pixels in a given image decreases when the color space is transformed to other color spaces.

The high correlation between RGB color components and mixing luminance with chromaticity makes RGB very sensitive to changes in imaging conditions such as lighting. The simplicity of the transform which is given in Equation (2.7), (2.8) and (2.9) and the explicit separation of luminance have made the model very attractive for skin color detection [12]. Since the Y component is left out in the skin segmentation, using the YCbCr color space has a computational advantage where the problem of 3-D color space is minimized to a 2-D color

space problem. Therefore, in this thesis, the YCbCr color space is used for skin segmentation as it has a computational advantage.

2.3 Color space modeling for skin segmentation

2.3.1 Introduction

Color segmentation is the most important stage in computer vision and sign language translation systems. The different color spaces have their own advantages and disadvantages when used with various modeling techniques. With all the color spaces, the human skin color usually remains within certain range. However, it is important to know that whenever a skin color is transformed from RGB to other color spaces, the discrimination between skin and non-skin color pixels decreases. The most known color modeling techniques for segmentation are explicit skin-color definition, Bayesian classifiers, Neural Network model and Gaussian classifiers.

2.3.2 Explicit skin-color definition

One method to build a skin classifier is to define explicitly, through a number of rules because the boundaries of skin color usually cluster in some color space. In fact, most of the color spaces have been used with the explicit skin-color segmentation for various purposes. For example, in RGB color space, skin pixel color can be classified as in the following combination of rules [20].

Under daylight uniform illumination, a pixel in (R, G, B) is classified as skin if:

$$R > 95 \text{ AND } G > 40 \text{ AND } B > 20$$

$$\text{AND}$$

$$\text{Max}\{R, G, B\} - \text{Min}\{R, G, B\} > 15 \qquad (2.10)$$

$$\text{AND}$$

$$|R-G| > 15 \text{ AND } R > G \text{ AND } R > B$$

And under flashlight or (light) daylight, a pixel is classified as skin if:

$$R>220 \text{ AND } G>210 \text{ AND } B>170 \text{ AND } |R-G|\leq 15 \text{ AND } R>B \text{ AND } G>B \quad (2.11)$$

In [11], the researchers built a skin color model based on explicitly defined skin regions in normalized RGB color space. Normalized RGB color space was chosen in their work for many reasons; first, it contains no information about luminance which yields a more general skin color model; it has only two components which helps to speed up the calculations; the transformations from RGB color space into normalized RGB color space is done using simple and fast transformations. The main reason they have used explicitly defined skin regions in building the skin detector is its speed in detecting the skin regions. The r and g components of the Normalized RGB color space was used for their skin detection model.

$$\mu_r - \alpha\sigma_r < r < \mu_r + \alpha\sigma_r \quad (2.12)$$

$$\mu_g - \alpha\sigma_g < g < \mu_g + \alpha\sigma_g \quad (2.13)$$

where μ_r, σ_r are the mean and standard deviation of the r components of skin pixels and μ_g, σ_g are the mean and standard deviation of the g components of skin pixels. The value of α determines how accurate the skin detector will be and its value is to be determined experimentally.

In [21], simple and efficient color-based approach to segment human skin pixels from complex background, using a 2-D histogram based approach is used. Several rules were used to discriminate between skin and non-skin pixels.

$$R_1 = \begin{cases} 0.5941 \le \frac{G}{R} < 0.8922 & if\ R > 0.9\ and\ G > B \\ 0.4412 \le \frac{G}{R} < 0.8686 & otherwise \end{cases}$$

$$R_2 = \begin{cases} 0.8255 \le \frac{B}{R} < 1.0262 & if\ B > 0.8500 \\ 0.4059 \le \frac{B}{R} < 0.7902 & otherwise \\ \frac{G}{R} < 0.6667\ and\ \frac{B}{R} < 0.4059 & \end{cases} \quad (2.14)$$

$$R_3 = \begin{cases} 0.5157 \le \frac{B}{G} < 1.0761 & if\ B > 0.3333 \\ 0.5157 \le \frac{B}{G} < 0.8882 & otherwise \\ \frac{G}{R} < 0.6667\ and\ \frac{B}{R} < 0.8882 & \end{cases}$$

An RGB color pixel is classified as a skin candidate if the result of

$$\mathbf{S = R_1 \,\&\, R_2 \,\&\, R_3} \quad (2.15)$$

is true, otherwise it will be classified as a non-skin pixel.

In [52], a simple rule for skin-color segmentation is defined based on a YCbCr color space. First, the RGB image is converted to its YCbCr version. After detecting the skin-pixels in the YCbCr space, the image is converted back to RGB color space. The rule is defined as follows:

Cr > 132 AND Cr < 173 AND Cb > 76 AND Cb < 126 (2.16)

For a given color pixel in the YCbCr color space, if the above rule is computed to be true, the pixel is considered as a skin pixel. As the computational expense is reduced from 3-D to a 2-D problem, the YCbCr color space with the rule from [52] is used in skin segmentation for this work.

2.3.3 Bayesian classifier model

Bayesian theory of decision (BTD) is a fundamental tool of analysis in Machine Learning. Several machine learning algorithms have been derived using BTD. The fundamental idea in BTD is that the decision problem can be solved using probabilistic considerations [22]. Bayesian networks are directed acyclic graphs (DAG), namely, there are no cycles, which allow efficient and effective representation of the joint probability density functions. Each vertex in the graph represents a random variable, and edges represent direct correlations between the variables [23]. In [13], the probability of observing skin given a color vector c is defined as follows:

$$P(skin|c) = \frac{P(c|skin)P(skin)}{P(c|skin)P(skin) + P(c|\neg skin)P(\neg skin)} \qquad (2.17)$$

Where *P(c|skin)* and *P(c|¬skin)* are directly computed from skin and non-skin color histograms. *P(skin)* and *P(¬skin)* are the prior probabilities that can also be estimated from the overall number of skin and non-skin samples in the training set [24]. In this method, there are two assumptions when constructing the probabilities for Bayesian decision making. In the first case, the probability that a pixel is skin is assumed to be the same as the probability that a pixel is non-skin *(P(skin) = P(¬skin)).* In the second case, the values of probabilities *P(skin)* and *P(¬skin)* are estimated from the training data. An inequality $P(skin|c) \geq \Theta$, where Θ is a threshold value, can be used as a skin detection rule. In [4], it has been stated $P(skin|c) \geq \Theta$ is invariant to choice of prior probabilities, due to nature of the Bayes model. This means that *P(skin)* value affects only the choice of the threshold Θ.

2.3.4 Neural Network model

Neural networks have been applied in many pattern recognition problems like optical character recognition and object recognition. This model has been used in works of many researchers particularly for skin color and face detection [9, 11, 25, 19]. A key idea of Neural Network is that after training, it is capable of generalizing from the training patterns and hence predicting the corresponding classes for patterns previously unknown to it. In other words the Neural Network

performs a high order of regression to fit the hidden function that relates its inputs and its outputs. Instead of following a set of predefined human designed rules, the neural network based skin color detection systems learns the underlying rules from given samples [11].

In [9], the researchers have used a neural network model for skin color detection with proper selection of optimum classification threshold that varies from image to image. The classifier gave the detection rate of more than 90% with 7% false positives on an average. They have also conducted a lot of experiment to find the number of neurons in the hidden layer of a 3×n×2 MLP network so as to achieve proper classification of the skin and non-skin samples. And n was found to be 5. The network was trained using Back Propagation Algorithm (BPA). The classifier has the ability to classify the skin pixels belonging to people from different ethnic groups even when they are present simultaneously in an image.

2.3.5 Gaussian classifiers

The most important parameter in developing skin color based segmentation methods lies in choosing the color space and the model used for representing the distribution of the skin color values. A later step involving processing the segmentation results can be affected drastically from the color distribution model parameters and the space used, so an efficient segmentation is the key feature in the successful implementation of detection and tracking systems [12]. Gaussian model, as a parametric color distribution model, tries to describe the chrominance feature space of skin color using a statistical model. Obviously the key problem here is finding the best model and estimating its parameters. The estimation should reasonably well fit the training data where the goodness of fit depends on the shape of the distribution and therefore the color space used [24].

A single bivariate gaussian probability density function (pdf) can be used successfully as a model for the skin color, even when multiple ethnic groups are considered [26]. Under controlled illuminating conditions, skin colors of different individuals cluster in a small region in the color space. Hence, under certain lighting conditions, the skin-color distribution of different

individuals can be modeled by a multivariate normal (Gaussian) distribution in normalized color space. The model can be obtained via the maximum likelihood criterion, which looks for the set of parameters (*mean vector* and *covariance matrix*) that maximizes the likelihood function. The likelihood function for a multivariate Gaussian pdf has a single maximum and then estimates the mean vector ($\boldsymbol{\mu}$) and covariance matrix ($\boldsymbol{\beta}$) for they are obtained analytically. Skin-color distribution is modeled through elliptical Gaussian joint probability distribution function (pdf), defined as:

$$\mathrm{p(c)} = \frac{1}{(2\pi)^{\frac{1}{2}}|\boldsymbol{\beta}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{c}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\beta}^{-1}(\mathbf{c}-\boldsymbol{\mu})\right] \qquad (2.18)$$

Where $\mathbf{c}$ is a color vector, $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$ are the mean vector and the diagonal covariance matrix, respectively and defined as follows:

$$\boldsymbol{\mu} = \frac{1}{\mathrm{n}}\sum_{\mathrm{j}=1}^{\mathrm{n}} \mathbf{c}_{\mathbf{j}} \qquad (2.19)$$

$$\boldsymbol{\beta} = \frac{1}{\mathrm{n}-1}\sum(\mathbf{c}_{\mathrm{j}}-\boldsymbol{\mu})(\mathbf{c}_{\mathrm{j}}-\boldsymbol{\mu})^{\mathrm{T}} \qquad (2.20)$$

The parameters, $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$ are estimated over all the color samples ($\mathbf{c_j}$) from the training data using ML (Maximum Likelihood) estimation approach which has been studied well in the literature. The probability **p(c)** can be used directly as a measure of skin-color likeliness and the classification is normally obtained by comparing it to a certain threshold value estimated empirically from the training data [23, 26]. A quantity $\boldsymbol{\lambda}$ called ***Mahalanobis*** distance from color vector $\mathbf{c}$ to mean vector $\boldsymbol{\mu}$ of the model defined as

$$\lambda = (\mathbf{c}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\beta}^{-1}(\mathbf{c}-\boldsymbol{\mu}) \qquad (2.21)$$

with some threshold, can also be used for skin pixel classification [23, 27, 26].

Though the human skin-color samples for people of different races cluster in a small region in the color space, it has been shown that different modes co-exist within this cluster and hence it cannot be modeled effectively by a single Gaussian distribution. Also, under varying illuminating conditions, the single mode assumption does not hold. Many researchers, therefore, have used

Gaussian mixtures, a more capable model to describe complex shaped distributions [23, 26, 12]. A Gaussian mixture density function is the sum of individual Gaussians, expressed as:

$$p(c) = \sum_{i=1}^{N} w_i \frac{1}{(2\pi)^{\frac{1}{2}} |\boldsymbol{\beta}_i|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{c}-\boldsymbol{\mu}_i)^T \boldsymbol{\beta}_i^{-1}(\mathbf{c}-\boldsymbol{\mu}_i)\right] \qquad (2.22)$$

Where **c** is a color vector and $\boldsymbol{\mu}_i$ and $\boldsymbol{\beta}_i$ are the mean and the diagonal covariance matrix. N is the number of Gaussians and the weight factor, w_i is the contribution of the i^{th} Gaussian. The parameters of a **GMM ($\boldsymbol{\mu}_i$, $\boldsymbol{\beta}_i$, w_i)** are approximated from the training data through the iterative expectation-maximization (EM) technique [12, 23].

As discussed in detail in Section 2.3, skin-color was modeled with various modeling techniques. Explicit skin-color modeling, Bayesian classifiers, neural network model and Gaussian classifiers has been discussed. In this thesis, the YCbCr color space with the Explicit skin-color modeling used in [52] is used.

2.4 Object tracking in video

Tracking is one of the most important tasks in computer vision and robotics. Video-based information collection has become an important research direction, and moving object tracking technique plays a key role nowadays [28]. *Object tracking* is an important task within the field of computer vision. The growth of high-performance computers, the availability of high quality yet inexpensive video cameras, and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms [45]. In robotic applications, some features of a moving object is collected in the subsequent image frames to determine in which direction the object is moving. There are three key steps in video analysis:

- detection of interesting moving objects
- tracking of such objects from frame to frame
- analysis of tracks to recognize their behavior

A number of researches have been done in object tracking in general and gesture tracking in particular [28, 29, 30, 42, 44]. Sign language involves mostly hand movements in different direction and hence translation systems for sign languages should be able to track the hand and determine the behavior of the hand movement.

Gesture tracking techniques have been extensively used in human-computer interaction (HCI) systems. In robot controlling applications with vision-based techniques, hand or gesture tracking is a fundamental stage. Hand Gesture Tracking System Using Adaptive Kalman Filter was developed in [42, 44] where an average accuracy of 97.83% was found in [44]. This system segments the hand region using YCbCr color space and determines the hand position. In [44] a hand tracking system based on Adaptive Kelman Filter (AKF) was proposed. The system consists of two main stages which are initialization and tracking. In initialization, hand region is first detected by combining motion and skin color pixels. A region of interest (ROI) is then created around the detected hand region. In tracking stage, skin and motion pixels are scanned around top, left and right corners of the ROI to detect the moving hand in consecutive video frames. These pixels are used to actually measure the ROI position and fed into measurement update of AKF operation.

2.5 Key-frame selection from video sequence

2.5.1 Video summarization

Video is represented as a sequence of consecutive frames, each of which corresponds to a constant time interval [38] called frame duration which depends on the video camera used. The key frame is a simple yet effective form of summarizing a long video sequence. The number of key frames used to abstract a shot should be compliant to visual content complexity within the shot and the placement of key frames should represent most salient visual content [46].

Key frames can be defined as a subset of a video sequence that can represent the video visual content as close as possible, with a limited number of frame information and are also widely

used in video abstraction [51]. In [51], the researchers developed a novel method to extract key frames to represent video shot based on connectivity clustering. The method dynamically divides the frames into clusters depending on the content of shot, and then the frame closest to the cluster centroid is chosen as the key frame for the video shot.

To extract key frames based on motion patterns, a triangle model of Perceived Motion Energy (PME) has been developed to represent the motion activities in video shots. Compared to the optical flow and the frame difference techniques, PME is a combined metric of motion intensity and motion characteristics with more emphasis on dominant motion [46]. With this triangle model, a video shot is segmented into sub segments of different motion patterns in terms of acceleration and decelerations. Consequently, the accumulated PME along a sub segment reflects its relative salience of visual action and can be used as the criterion for sorting importance of motion patterns. In the model used in [46], the frames at the turning point of the motion acceleration and motion deceleration are selected as key frames.

In [50], the researchers proposed an optimal key frame representation scheme based on global statistics for video shot retrieval. Each pixel in this optimal key frame is constructed by considering the probability of occurrence of those pixels at the corresponding pixel position among the frames in a video shot. Therefore, this constructed key frame is called temporally maximum occurrence frame (TMOF), which is an optimal representation of all the frames in a video shot.

2.5.2 Representative gesture selection

In [2], Fourier Descriptor (FD) was used to compare consecutive frames using frame difference for key-frame (key hand posture) selection. Velocity and signing rate based filter were used to select a valid hand posture in [49]. The posture selection is based on the nature of finger spelling where the signer should hold the hand for some time for people to see it. So, by defining high and low signing rate threshold, they were able to select good postures among the continuous gestures. The researchers in [49] have used 5DT Data Glove 5 which comprises 18

sensors to collect information about posture velocity and signing rate of individuals. The sensors correspond to ten positions on fingers (thumb near, thumb far, index near, index far, middle near, middle far, ring near, ring far, little near, little far), four positions between fingers (thumb/index, index/middle, middle/ring, ring/little), and a position on the back of the hand.

Literatures related to the task at hand are reviewed in this chapter. As vision-based system, the proposed design involves color segmentation which includes color space choice and skin color modeling. Key frame selection is one of the important parts of the proposed design where a consecutive frame difference is used for good key frame (candidate gesture in this case) selection. An HMTD technique is used to determine the form of the trajectory for each selected candidate gestures. For the trajectory determination a simple yet effective technique is proposed.

In this thesis work, an idea of combining metrics is used to extract candidate gestures where speed profile of the whole video frame sequence is used to divide the sequence into several blocks and then a search window is defined within each block. Frame difference within the search window based on the MHD measurement is applied for search of the very similar gestures in order to pick one.

2.5.3 Hausdorff distance

In order to compare different hand geometries the Hausdorff Distance (HD) is a very efficient method. This metric has been used in binary image comparison and computer vision for a long time [53, 54]. The advantage of Hausdorff distance over binary correlation is the fact that this distance measures proximity rather than exact superposition, thus it is more tolerant to perturbations in the locations of points. Unlike most shape comparison methods, the Hausdorff distance between two images can be calculated without the explicit pairing of points of their respective data sets [46]. Given the sets **F** and **G** of the contour pixels of two hands, represented by the sets $F = \{ f_1 , f_2 ,..., f_N\}$ and $G = \{ g_1 , g_2 , ..., g_N\}$, where $\{f_i\}$ and $\{g_j\}$ denote contour pixels for i=1, 2, ..., N_f and j=1, 2, ..., Ng , the Hausdorff distance is defined as follows:

$$H(F,G) = \max\left(h(F,G), h(G,F)\right) \tag{2.23}$$

where $h(F,G) = \max_{f \in F} \min_{g \in G} \|f - g\|$. In this formula $\|f - g\|$ is the norm of the elements over the two sets and obviously the contour pixels (f,g) run over the set of indices $i = 1,2,\ldots,N_g$ and $j = 1,2,\ldots,N_f$. In this case this norm is taken to be the Euclidean distance between the two points. Moreover, the Hausdorff Distance is a match methodology without point-to-point correspondence [55].

Since the original definition of the HD is rather sensitive to noise, several modifications of the HD have been proposed to improve it [55].

The directed distance of the partial HD (PHD) is defined in equation (2.24).

$$h_K(F,G) = K^{th}_{f \in F} d(f,G), \tag{2.24}$$

where $K^{th}_{f \in F}$ denotes the k^{th} ranked value of *d(f,G)*. Thus, the PHD depends on a parameter $p = K/N_F$ standing for the proportion of values taken into account.

Another improvement on the original definition of HD is the Modified HD (MHD) and is defined as follows:

$$h_{MHD}(F,G) = \frac{1}{N_F} \sum_{f \in F} d(f,G). \tag{2.25}$$

where N_f is the number of points in set F.

Unlike the PHD, the MHD measure does not require any parameters [55] to be used as similarity measure and it is used to compare subsequent gestures in this thesis work.

CHAPTER 3

DIGITAL IMAGE PROCESSING

3.1 Introduction

An image is defined as a 2-D function $f(x, y)$ where x and y are spatial coordinates and amplitude of f at any pair of coordinates (x, y) is called the intensity or grey level of image at that point. The image consists of number of elements called pixels and we process these pixels. Digital image processing refers to processing digital images such they are used for human or autonomous machine interpretation.

In computer vision systems, image processing is usually considered as the fundamental stage. Video surveillance, robotics, medical and geographical image processing are some of the areas where digital image processing used extensively. In applications where human-computer interaction is involved, it is common for the system to begin with segmenting parts of the human body and then extracts meanings based on the design. Similarly, researchers have extensively used digital image processing for vision-based sign language translation applications.

3.2 RGB, Grayscale and Binary images

The RGB color space, commonly called true color, is the default color space for most available image formats where any other color space such as HSV, YCbCr and normalized RGB can be obtained from a linear or non-linear transformation from RGB. This image format consists of three color components, red (R), green (G) and blue (B) for individual pixel. The color of each pixel is determined by the combination of the red, green, and blue intensities stored in each color plane at the pixel's location. Graphics file formats store RGB images as 24-bit images, where the red, green, and blue components are 8 bits each. This yields a potential of 16 million colors.

Intensity or grey-scale images are 2-D arrays that assign one numerical value to each pixel which is representative of the intensity at this point [33]. We can convert from an RGB color space to a grey-scale image using a simple transform. Grey-scale conversion is the initial step in many image analysis algorithms, as it essentially simplifies the amount of information in the image. Although a grey-scale image contains less information than a color image, the majority of important, feature related information is maintained, such as edges, regions, blobs, junctions and so on. Feature detection and processing algorithms then typically operate on the converted grayscale version of the image.

The image data in a grayscale image consist of a single channel that represents the intensity, brightness, or density of the image. In most cases, only positive values make sense, as the numbers represent the intensity of light energy and that cannot be negative, so typically whole integers in the range of $[0 \ldots 2^k - 1]$ are used. For example, a typical grayscale image uses k = 8 bits (1 byte) per pixel and intensity values in the range of [0... 255], where the value 0 represents the minimum brightness (black) and 255 the maximum brightness (white) [32].

Binary images are 2-D arrays that assign one numerical value from the set {0, 1} to each pixel in the image. These are sometimes referred to as logical images: black corresponds to zero (an 'off' or 'background' pixel) and white corresponds to one (an 'on' or 'foreground' pixel). As no other values are permissible, these images can be represented as a simple bit-stream, but in practice they are represented as 8-bit integer images in the common image formats. A fax (or facsimile) image is an example of a binary image [33].

3.3 Structuring element

In digital image processing a number of techniques are applied to digital images either to improve the quality of the image or to make the image suitable for subsequent processes. Smoothing and sharpening are some of the techniques that require structuring element.

A structuring element is a rectangular array of pixels containing the values either 1 or 0. A number of example structuring elements are depicted in the following figure. When a

structuring element is positioned on an image to be processed, the value for the pixel of the image directly under the center pixel of the structuring will be a function of the neighboring pixel values under the structuring element. Structuring elements have a designated centre pixel which is located at the true centre pixel when both dimensions are odd (e.g. in 3x3 or 5x5 structuring elements) [33].

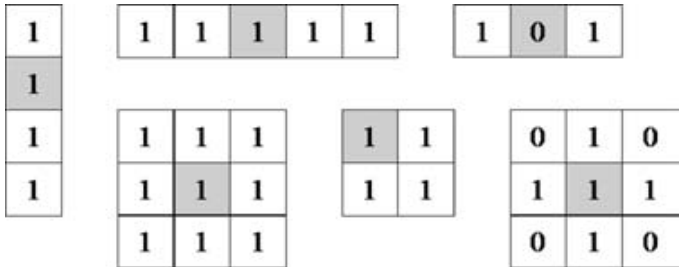


Figure 3.1 Some examples of morphological structuring elements

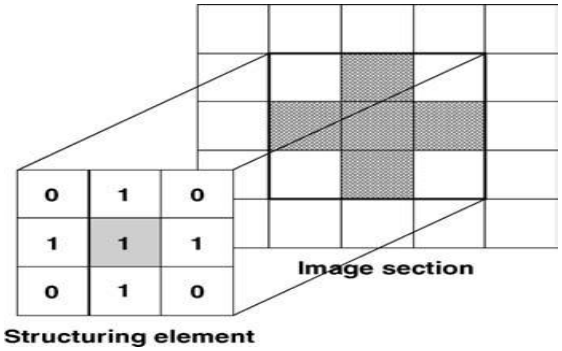
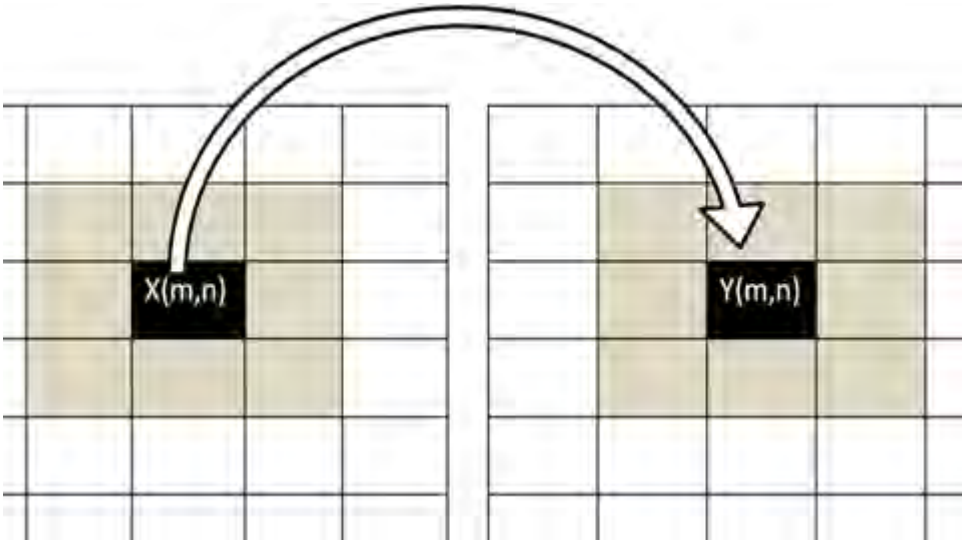


Figure 3.2 The local neighborhood defined by a structuring element

3.4 Spatial image filtering

Spatial filtering is a technique that uses a pixel and its neighbors to select a new value for the pixel. For spatial domain filtering, we are performing filtering operations directly on the pixels of an image. The simplest type of spatial filtering is called linear filtering. It attaches a weight to the pixels in the neighborhood of the pixel of interest, and these weights are used to combine those pixels together to provide a new value for the pixel of interest. Filtering can be linear or non-linear. Linear filtering can be used to smooth, blur, sharpen, or find the edges of an image.

Linear filtering is the cornerstone technique of signal processing. To briefly introduce, a linear filter is an operation where at every pixel $x(m,n)$ of an image, a linear function is evaluated on the pixel and its neighbors to compute a new pixel value $y(m,n)$. A 3x3 neighborhood of pixels is shown in Figure 3.3. The simplest linear filter is usually an averaging filter where the average of the neighborhood defined by the filter size is used as a value for pixel at the center of the considered neighborhood.



a b

Figure 3.3 Image showing pixel in: a) original image b) filtered image

A linear filter in two dimensions has the general form

$$Y_{m.n} = \sum_{j} \sum_{k} H_{j,k} X_{(m-j),(n-k)} \qquad (3.1)$$

where X is the input, Y is the output, and H is the filter or mask. Different choices of H lead to filters that smooth, sharpen, and detect edges, to name a few applications. The right-hand side of the above equation is denoted concisely as $H * X$ and is called the "convolution of H and X." The steps in image filtering can be summarized as follows:

- Place the center of the filter H on each pixel $X(m,n)$ of the image

- Multiply each pixel in the defined neighborhood by the appropriate filter mask coefficient superimposed on it
- Sum all products
- Place the suitably normalized sum into position $X(m,n)$ of the output image

3.4.1 Noise reduction by averaging filter

Images can be noisy due to environmental effects and camera condition. It is usually appropriate to apply filters to noisy images to prepare them for further process. Averaging filters are usually applied to smooth images so that high intensity but infrequent pixels, considered noise, are removed.

3.4.2 Sharpening

When an averaging filter is applied to digital images, the output is usually smooth but blurred. In applications where edges are very important, an image sharpening technique is used to enhance edges from the blurred image. To increase the discrimination ability of the neural network model, the input gestures should somewhat be sharp.

a b

Figure 3.4 Sharpening grayscale image a) blurred image b) sharpened image

3.5 Morphological image processing

3.5.1 Introduction

Morphological image processing is a type of digital image processing in which the spatial form or structures of objects within an image are modified. Dilation and erosion are two fundamental morphological operations. With dilation, an object grows uniformly in spatial extent, whereas with erosion an object shrinks uniformly [34]. For both type of morphological operations, a structuring element is used.

3.5.2 Morphological Dilation and Erosion

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the *structuring element* used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as dilation or erosion.

To perform erosion of a binary image, we successively place the centre pixel of the structuring element on each foreground pixel (value 1). If any of the neighborhood pixels are background pixels (value 0), then the foreground pixel is switched to background. Formally, the erosion of image A by structuring element B is denoted $A \ominus B$. To perform dilation of a binary image, we successively place the centre pixel of the structuring element on each background pixel. If any of the neighborhood pixels are foreground pixels (value 1), then the background pixel is switched to foreground. Formally, the dilation of image A by structuring element B is denoted $A \oplus B$

3.5.3 Morphological Opening and Closing

There are two common morphological transformations which consist of combinations of erosion and dilation. The opening operation involves the application of erosion, followed by dilation. The effect of using a square- or disk-shaped structuring element for opening is to smooth boundaries, to ***break narrow isthmuses***, and to eliminate small noise regions. The erosion operation reduces these features (and associated noise), and the subsequent dilation operation restores regions to their original size, now lacking the above-mentioned features. The companion operation to opening is closing, and this involves the application of dilation, followed by erosion. The effect of using a square- or disk-shaped structuring element for closing is to smooth boundaries, to join narrow breaks, and to fill small holes caused by noise [35]. Figure 3.4 shows a binary image before and after morphological operations are applied to it.

Figure 3.5 Morphological operations: a) Before opening and closing b) After opening and closing

The difference between opening and closing is in the initial iteration, erosion, or dilation. The choice of operation depends on the image and the objective. For example, opening is used when the image has many small noise regions. It is not used for narrow regions where there is the chance that the initial erosion operation might disconnect regions. Closing is used when a region has become disconnected and the desire is to restore connectivity. It is not used when different regions are located closely such that the first iteration of dilation might connect them.

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

4.1 System architecture

The overall flow chart for the proposed system is shown in figure 4.1

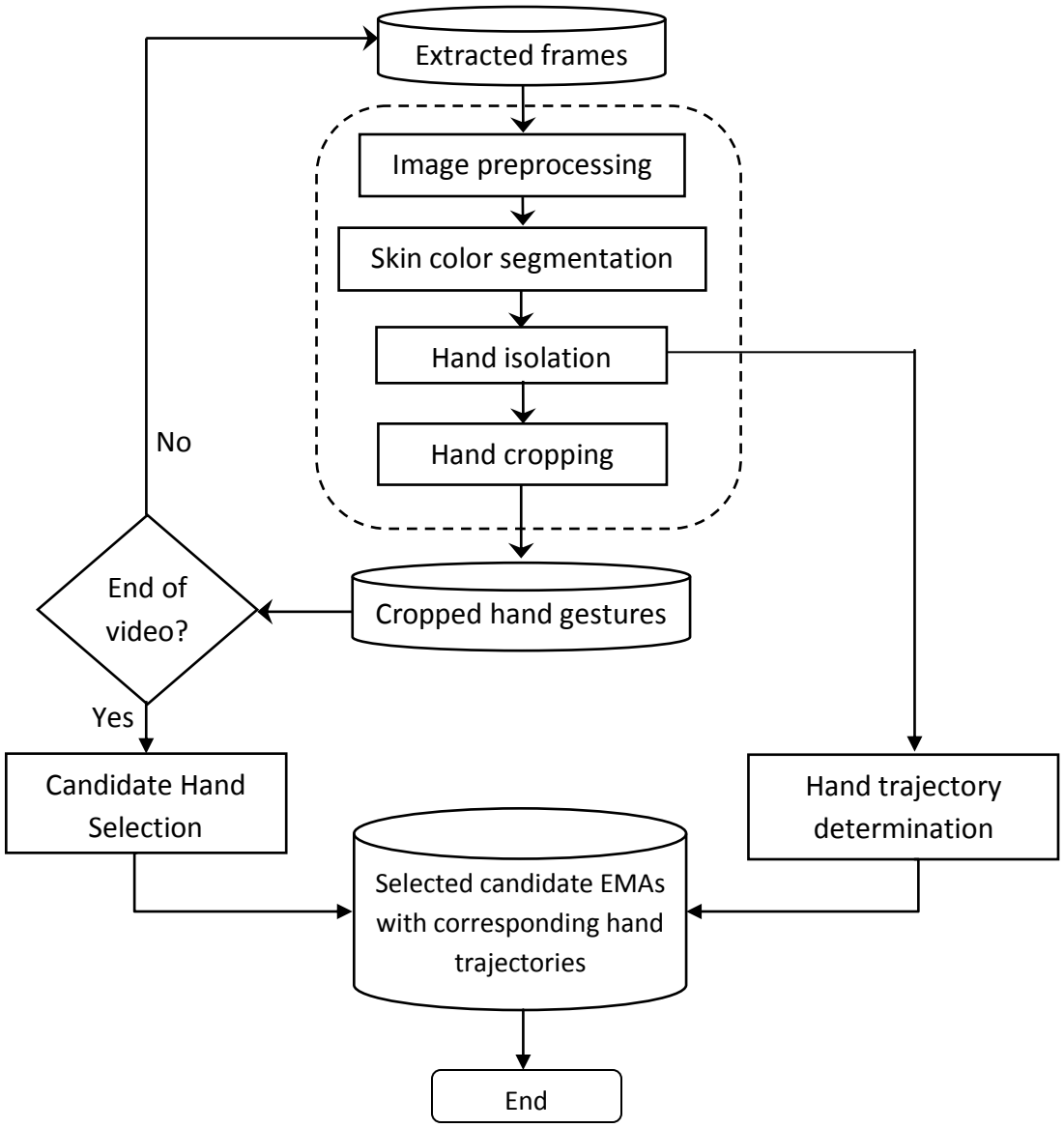


Figure 4.1 The overall proposed design flowchart

4.2 Data collection and image preprocessing

The system is expected to work for various signers that sign with fairly different speed and hand size. Five signers with different hand sizes and color (dark and bright) were considered and names of people and places with 2 and 3 EMAs were recorded in a slightly controlled environment. A 14.1 Mega pixel SONY digital camera were used to record the videos. When recording the videos, orientation was given to the signers to avoid an overlap between the face and the hand.

Table 4. 1 List of words used to design the system

Words used to design the system	
2 EMAs	***3 EMAs***
ሁዳ (HUDA)	ያሲን (YASIN)
ሳኒ(SUNNY)	ራህማ (RAHMA)
ባቲ (BATI)	
ዲላ (DILLA)	
ባሌ (BALLE)	

Table 4. 2 List of words used to test the system

Words used to test the system	
2 EMAs	***3 EMAs***
ሙሉ (MULU)	ሐዋሳ (HAWASSA)
ኪያ (KIA)	ብሩክ (BIRUK)
ባላ (BALLA)	ሃይሉ (HAILU)
ካሴ (KASSIE)	ዳዊት (DAWIT)
ራያ (RAYA)	ኢዛና (EZANA)
ኩኩ (KUKU)	

For the system to be general, different combinations of EMA variations were used. For each word either under the design or the testing column, two samples were recorded from each signer.

- For system designing
    - 7 words x 2 samples x 5 signers = 70 videos
- For system testing
    - 5 words x 2 samples x 5 signers = 50 videos
    - 6 words x 2 samples x 2 signers = 24 videos

Therefore, a total of 144 videos were collected to develop the proposed design where 70 of them were used to design the system and 74 of them were used to test the system. For the testing data, 2 samples for 5 words each were collected from 5 signers. From 2 of the signers additional data: 2 samples for 6 words each were also collected.

144 videos were collected from 5 signers for 18 words. The words were selected to prove that the proposed design works for words with 2 and 3 EMAs. Various combinations of EMA forms were also considered when selecting the words. Each video was extracted into sequences of RGB video frames of size 640x480 pixels.

Free Video to JPG Converter 1.8.6, downloaded from Internet*,* was used to extract the video frames or to convert the video into sequences of image frames. Sample extracted frames from the video for the name HUDA (ሁዳ) is shown in Appendix B. As programming tool, MATLAB 7.7.0 (R2008b) was used for image processing.

4.3 Skin-color segmentation

An RGB image of size 640x480 pixels with any background color other than colors that are close to skin color is used as an input to the skin segmentation module. In fact, any background could be used by avoiding colors that are very close to skin color. Prior to skin-color segmentation, average filter with a structuring element in Appendix A.1 line 3 is used to reduce noise that may

exist in the RGB image. In Figure 4.2, original images extracted from video clips are shown for three signers where (a) and (b) are from right-handed signers and (c) is from left-handed signer.

Figure 4.2 Original RGB images: (a) and (b) from right-handed people, (c) from left-handed person

This module uses an explicit-definition skin color modeling technique discussed in Section 2.3.2 with a YCbCr color space developed in [52] to segment skin areas of the input images shown in Figure 4.2. The function in Appendix A.1 is used for skin segmentation. The function has the form:

```
function [BI RGB]=newSkinSeg(im)
```

The function takes an RGB image as an input parameter and returns segmented binary and RGB images. First, the color space of the video frames is converted from RGB to YCbCr. After detecting the skin pixels in the Cb-Cr plane, the color space of the frames are converted back to RGB. The binary output of RGB input images are shown in Figure 4.3.

Figure 4.3 Binary images after applying skin segmentation to images in Figure 4.2 (a), (b) and (c)

In Figure 4.3, (c) was from a video recorded in a light-controlled room while (a) and (b) were recorded in an uncontrolled environment. After skin color segmentation, there are areas considered as skin while they are not. In fact, the areas detected falsely as skin are very small compared to the skin regions and these are removed by morphological operations discussed in Section 3.5. Opening followed by Closing operations are used with a structuring element of d=5x5 pixels.

Due to light reflections from clothes and background, the skin segmentation module may be affected and may segment some non-skin pixels as skin-pixels where there may exist some connected regions to be larger than the hand and the face. Therefore, to solve this problem and then improve the accuracy of segmentation, morphological operations are applied to images in Figure 4.3 above. Each of the images in Figure 4.4 are the corresponding results of images in Figure 4.3 after applying Opening with a 5x5 structuring element and Closing with the same structuring element.

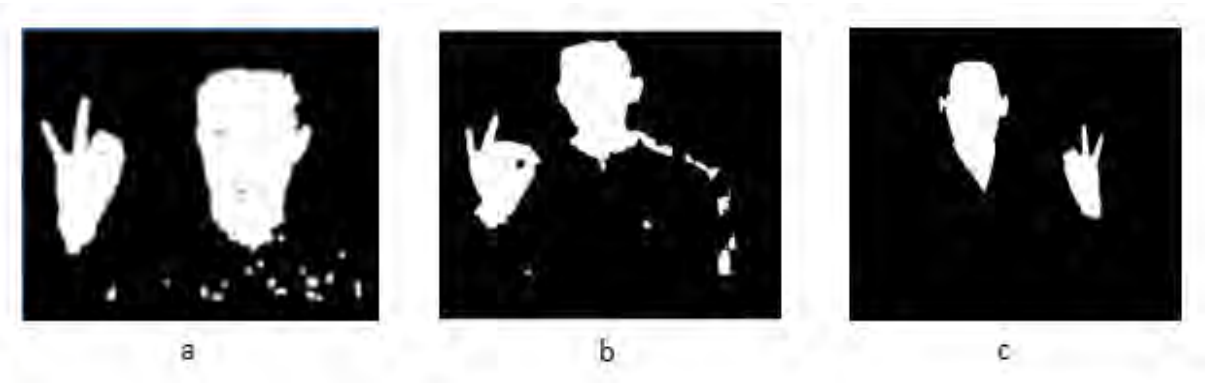


Figure 4.4 Binary images after applying morphological opening and closing to images in Figure 4.3

4.4 Segmented objects analysis for hand isolation

As one of the purposes of this thesis is to select candidate gesture, a hand isolation technique is used to extract the hand from the segmented image. After morphological operations, opening followed by closing, are used for better segmentation of the image in question, each object in the segmented image is analyzed. The main reason that opening followed by closing is used is to

avoid the chance of getting larger non-skin regions. By opening followed by closing, some thin regions (isthmuses) connecting two different regions are removed, as explained in Section 3.5, to still keep the hand as the second largest region in the segmented image. The function in Appendix A.2 is used for the hand isolation purpose and the function has the following form:

```
function [segImage rgbImage]=isolateHand(segImage,rgbImage)
```

The above function takes a binary and an RGB version of the same segmented image and returns a binary and an RGB version of the image with only the hand region.

In [43], it is stated that the system should be informed whether the signing person is a right or a left-handed so that the most right blob in the segmented image is considered as a hand for a right-handed person. But with a reasonable assumption that the face region is usually larger than the hand region, the proposed design is able to isolate the hand either from right or left handed signer without prior knowledge of the signer.

To avoid appearance of some big segmented areas, other than the hand and the face, the video is recorded under a slightly controlled environment and morphological operations are used to minimize this problem.

A hand isolation technique is applied to each segmented binary image in Figure 4.4. The hand isolation module calculates the area of each region in the segmented image and puts them in a descending order. The second largest region is selected as a hand gesture. Areas of regions in Figure 4.4 (a) are displayed in Table 4.3 and by removing areas other than the second largest region, the hand is successfully isolated. The isolated hand from Figure 4.4 (a) is displayed in Figure 4.5. Similarly, from Tables 4.4 and 4.5, Figures 4.6 and 4.7 are obtained respectively.

Table 4.3 Analysis of region areas for hand isolation of (a) in Figure 4.4

Area of each region before sorting	Area of each region after sorting	
AREA =	area =	
1521	2621	← Area of face
8	1521	← Area of hand
2621	89	
14	58	
15	51	
51	28	
58	18	
89	17	
1	16	
28	15	
2	15	
18	14	
17	10	
16	9	
15	9	
10	9	
9	8	
9	7	
9	7	
7	7	
7	2	
7	1	

Figure 4.5 Isolated hand using Table 4.3

Table 4.4 Analysis of region areas for hand isolation of (b) in Figure 4.4

Area of each region before sorting	Area of each region after sorting	
AREA =	area =	
27	4733	← Area of face
94	2052	← Area of hand
80	199	
199	94	
2052	80	
27	27	
7	27	
4733	27	
20	26	
17	21	
7	20	
9	18	
10	17	
16	16	
18	15	
9	10	
7	9	
27	9	
26	9	
15	7	
9	7	
21	7	

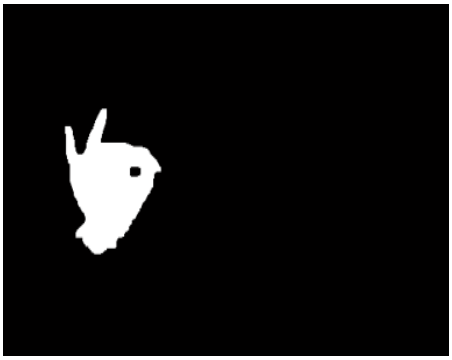


Figure 4.6 Isolated hand using Table 4.4

Table 4.5 Analysis of region areas for hand isolation of (c) in Figure 4.4

Area of each region before sorting	Area of each region after sorting
AREA =	area =
1263	1263 ← Area of face
501	501 ← Area of hand

Figure 4.7 Isolated hand using Table 4.5

4.5 Centroid collection for hand trajectories

After removing the regions other than the hand region, for the purpose of hand trajectory determination, the centroids of each hand gesture in Figure 4.5, 4.6 and 4.7 are collected. This centroid collection is done just before cropping the hand to detect the motion of the hand within the original video frame size. The function **trackHand(I)**, shown below and found in Appendix A.3, is used to find the centroid of a hand gesture at each video frame relative to the 640x480 pixels input image.

```
function [x y]=trackHand(I)
```

The above function takes a binary image containing only the hand region as in Figure 4.5 and returns the x and y components of the centroid 'A' shown in Figure 4.8 of the hand region.

Figure 4.8 An image showing a centroid of an isolated hand

4.6 Hand Cropping and Candidate Hand Gesture Selection

4.6.1 Hand cropping

After collecting centroid of each isolated hand gesture from Figures 4.5, 4.6 and 4.7 for hand tracking, the next task is to crop the hand gesture from the image frame. The proposed design searches for extreme white pixels as shown in Figure 4.9 to determine the top-left corner, height and width of the hand gesture using the function shown below and also listed in Appendix A.4. The function takes a binary image as input parameter and returns a cropped binary image and the top left corner of the hand relative to the whole video frame.

```
function [binaryC x y]=cropHand(binary)
```

The top left corner (x,y) returned from the above function is used as a starting point for cropping the RGB or grayscale images. The top-left corner (x,y) for the gestures shown in Figure 4.9 is (x_1,y_2). The ***height*** and the ***width*** of the cropped gesture are computed as y_4-y_2 and x_3-x_1 respectively.

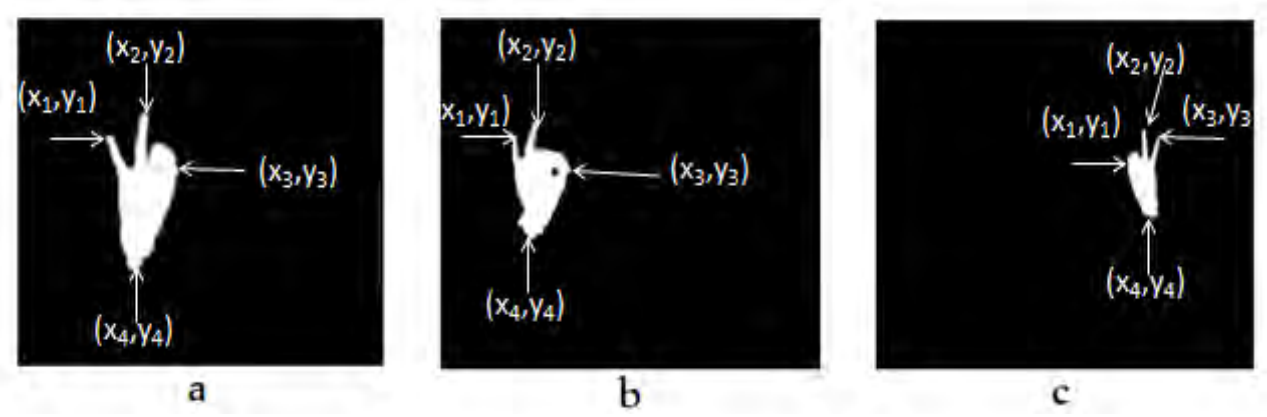


Figure 4.9 Binary images showing points of cropping for Figures 4.5, 4.6 and 4.7

Using the above information, (height and width), the hand region is cropped using ***imcrop*** function from MATLAB which has the following prototype:

```
OUT_PUT= imcrop(BI, [x y width height])
```

where BI is the input image with only the hand region and the output of the hand cropping module is shown in Figure 4.10.

Figure 4.10 Binary images after applying cropping algorithm to images in Figure 4.5, 4.6 and 4.7

For the purpose of gesture size generalization, all cropped hand gestures are normalized to the same size of 96x64 pixels.

4.6.2 Candidate Hand Gesture Selection

For a video clip of a 3 EMA word recorded for 3 seconds with 30 frames per second (fps) video camera, there will be 90 video frames out of which only 3 are required. Because it is not technically acceptable to try to recognize all the 90 frames, a CGS technique is employed in the

design to select the important hand gestures as a candidate for recognition. The proposed design tackles this problem in two steps as discussed in subsequent sections.

a) Dividing the whole gesture sequence into blocks based on a threshold on gesture speed profile using the function found in Appendix A.5. In fact, the researchers in [49] have used hand velocity to divide the video frames where the hand velocity is obtained from a 5DT Data Glove 5 which comprises 18 sensors. In this work, digital image processing techniques are applied to replace the sensor equipped hand glove to avoid the inconvenience of wearing gloves and to make the implementation easier.
b) Select a candidate gesture in each block using MHD measurement using the procedures listed through lines 90 to 152 in Appendix A.7.

a) Dividing the whole gesture sequence into blocks based on speed profile

First, speed profile of the hand gesture throughout the video is collected and the whole sequence of video frames is divided into several blocks of frames based on the speed profile of the gestures. The video camera used in this work records 30 fps where frame duration is 1/30 seconds (33.33 milliseconds).

To calculate the speed between two consecutive gestures and to develop the speed profile for a given video clip, first, the distance between centroids of gestures collected in Section 4.4 of this chapter is computed. However, to detect significant gesture movements and pauses, alternating frames are considered in developing the speed profile for a video clip. In Figure 4.11, the distance between centroids A and B of two alternating frames is computed and the speed of the hand movement is also calculated using equation (4.4)

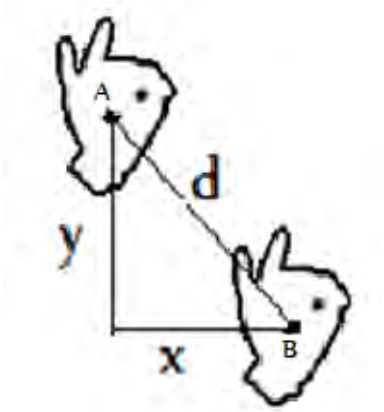


Figure 4.11 Distance between two alternating frames

where x and y are the difference between two coordinates (centroids) $A(x_1, y_1)$ and $B(x_2, y_2)$ from two alternating image frames.

$$x = x_2 - x_1 \quad (4.1)$$

$$y = y_2 - y_1 \quad (4.2)$$

$$d = \sqrt{x^2 + y^2} \quad (4.3)$$

The speed is obtained by dividing the distance computed from Figure 4.11 by the 2 frame durations because alternating frames are considered here

$$v = {}^{d}/_{66.67} \quad (4.4)$$

where v is in pixels per millisecond and 66.67 is the time difference between alternating frames in milliseconds. Figure 4.12 is a plot based on the speed profile from the sequence of images extracted from the input video which shows the nature of the hand movement for the valid gestures and transition or return path gestures. The plot was created using the speed profile for a video clip from signer 1 for the word SUNNY (**ሳኒ**).

In Figure 4.12, the y-axis represents the magnitude of the speed between hand gestures in alternating frames while the numbers in the x-axis stands for the speed of the hand between alternative gestures. For example, 1 in the x-axis represents the speed of the hand between

frames 1 and 3; in the same way, 2 represents the speed of the hand movement between frames 3 and 5 and so on. A speed profile for a 3 EMA video clip for the name RAHMA (ራህማ) is also shown in Figure 4.13 where each of the three EMAs of the word is represented by blocks A, B and C respectively.

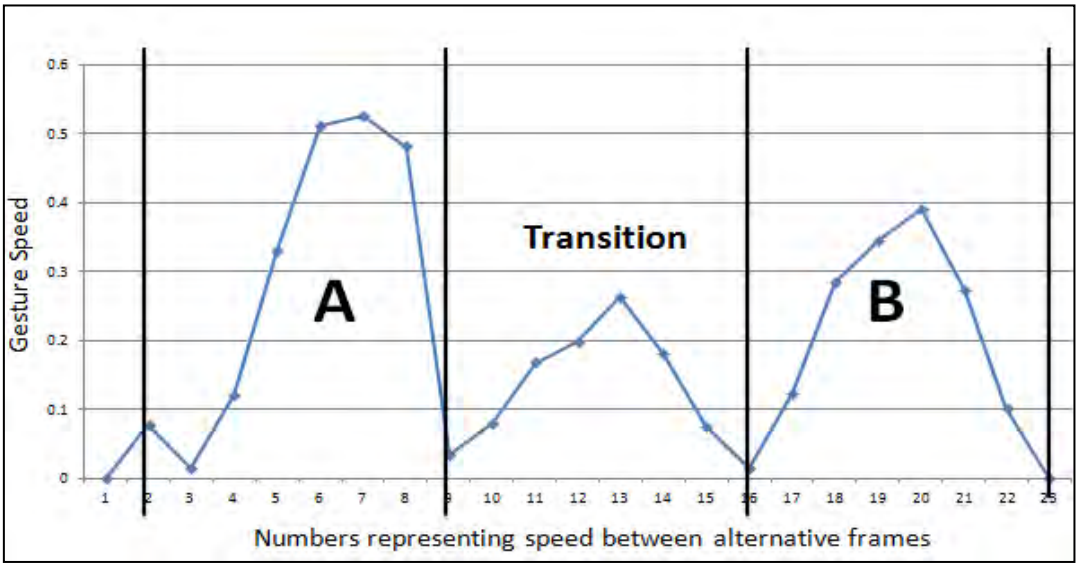


Figure 4.12 speed profile for a 2 EMA video clip for the name SUNNY (ሳኒ)

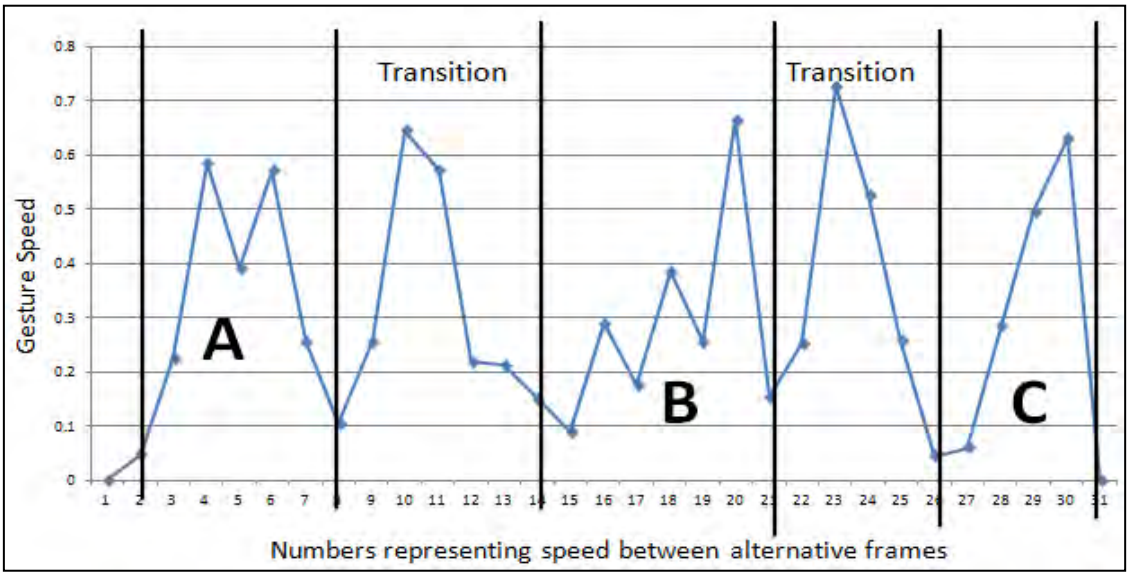


Figure 4.13 speed profile for a 3 EMA video clip for the name RAHMA (ራህማ)

Blocks A and B in Figure 4.12 are called EMA blocks where the center block is called transition block which represents the hand gestures in the return path of signing. And blocks A, B and C in Figure 4.13 are called EMA blocks where the other blocks are called transition blocks which represent gestures in the return paths.

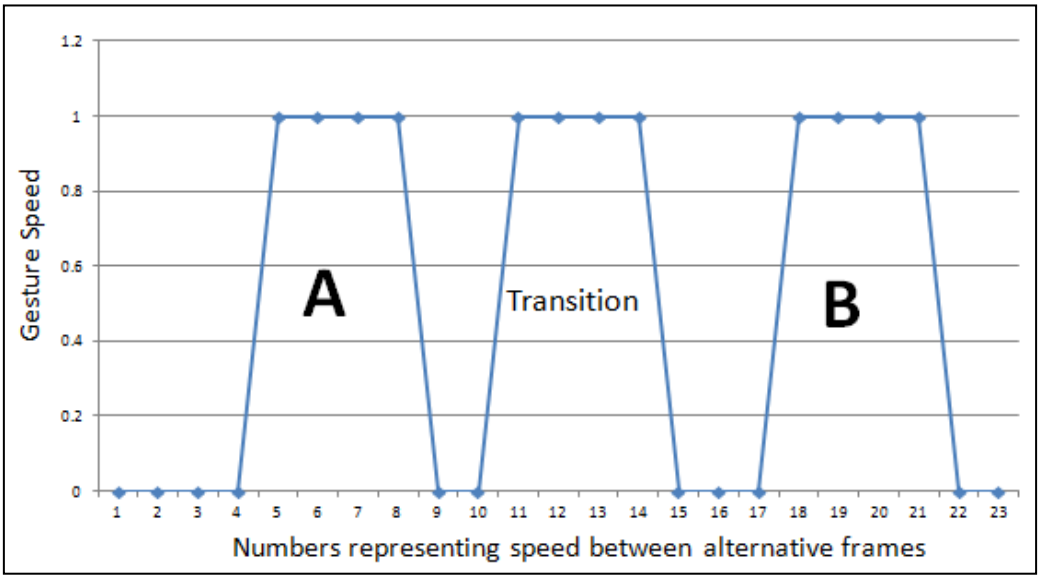


Figure 4.14 The speed profile in Figure 4.12 with a speed threshold of T = 0.155 pixels per millisecond

A total of 70 video clips were processed to select a speed threshold for BD. Gesture speeds from 0.085 to 0.205 pixels per millisecond at interval of 0.01 pixels per millisecond were used to divide the 70 videos into valid blocks that represent either an EMA or a return path. Table 4.6 shows the experimental result for BD.

Table 4.6 Experimental results of the 70 videos for BD

Threshold	Signer 1		Signer 2		Signer 3		Signer 4		Signer 5	
	Correctly divided	*Accuracy*	*Correctly divided*	*Accuracy*	*Correctly divided*	*Accuracy*	*Correctly divided*	*Accuracy*	*Correctly divided*	*Accuracy*
0.085	7	50.00%	5	35.71%	7	50.00%	4	28.57%	3	21.43%
0.095	8	57.14%	5	35.71%	9	64.29%	4	28.57%	4	28.57%
0.105	8	57.14%	7	50.00%	9	64.29%	4	28.57%	4	28.57%
0.115	8	57.14%	9	64.29%	11	78.57%	6	42.86%	9	64.29%
0.125	11	78.57%	9	64.29%	11	78.57%	7	50.00%	9	64.29%
0.135	12	85.71%	8	57.14%	12	85.71%	8	57.14%	10	71.43%
0.145	11	78.57%	8	57.14%	11	78.57%	9	64.29%	10	71.43%
0.155	12	85.71%	8	57.14%	13	92.86%	10	71.43%	11	78.57%
0.165	11	78.57%	7	50.00%	11	78.57%	9	64.29%	11	78.57%
0.175	11	78.57%	7	50.00%	9	64.29%	9	64.29%	11	78.57%
0.185	12	85.71%	6	42.86%	9	64.29%	8	57.14%	11	78.57%
0.195	11	78.57%	7	50.00%	9	64.29%	8	57.14%	11	78.57%
0.205	10	71.43%	8	57.14%	9	64.29%	9	64.29%	11	78.57%

By observing the number of correctly divided videos for each signer, a speed threshold of 0.155 pixels per millisecond is selected as a good threshold for BD.

b) Select a candidate gesture in each block using MHD by defining a search window

To select a representative hand gesture from each block, successive gestures within each block are compared to each other using the MHD. After computing the MHD between successive

gestures, one with the smallest difference is considered as a good and representative hand gesture.

To increase the reliability of the system for selecting good candidate and decrease the effect of transition gestures in the selection, a search window mechanism has been proposed in this work to be applied within a block as shown in Figure 4.15. Two important reasons to create a search window for gesture selection are:

1. EMA signers start with the correct EMA (palm points forward) and rotate their hands which result in invalid gestures from the camera point of view. So, it is reasonable to exclude some gestures to the end of each EMA blocks. In fact, these gestures are important for the purpose of constructing hand movement trajectories.
2. Signers often start with valid EMAs. However, to decrease the effect of gestures in transition blocks to the EMA blocks, some gestures are discarded from the beginning of each EMA block.

In selecting the candidate gesture, no threshold is required because the proposed design searches for a minimum difference between two gestures within the search window and selects the first one.

The alternating frames were used to divide the whole sequence of video frames into blocks. However, to select candidate gesture, the whole sequence of video frames is considered within each block. After dividing the whole sequence of the video frames into blocks using the speed threshold, each EMA blocks need a transform so that the blocks include subsequent gestures instead of alternating ones. For example, in Figure 4.12, block A were defined between 2 and 9 where 2 represents the speed of the hand movement between 3 and 5 and 9 represents the speed of the hand movement between 17 and 19. So for the candidate gesture selection, block A in Figure 4.12 is redefined to be between 3 and 17 by taking the first frames used in speed computation or by taking one less than twice of the extremes in the block because the BD is done using the alternative gestures. The block size is then 15 which comprise frames between 3 and 17 inclusive. As the MHD is computed between successive gestures in the block, there are

14 values in the x-axis, not 15. Computing the MHD between successive frames, the plot in Figure 4.15 is obtained from block A of Figure 4.12.

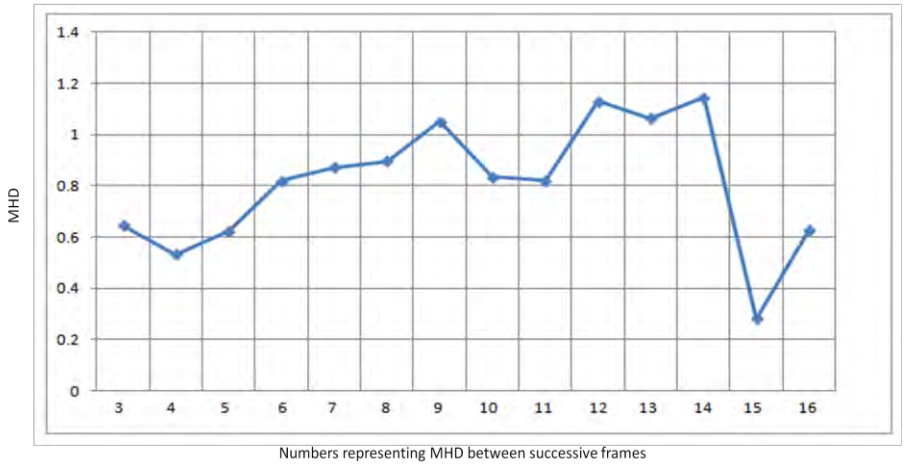


Figure 4.15 MHD with in block A of Figure 4.12

The search window shown in Figure 4.16 starts at point p where $p = \frac{1}{10} * Block\ Size$ and ends at point q where $q = \frac{2}{3} * Block\ Size$. These values are obtained experimentally such that the effect of transition gestures in the selection is minimized. Figure 4.16 is constructed by superimposing the search window over Figure 4.15.

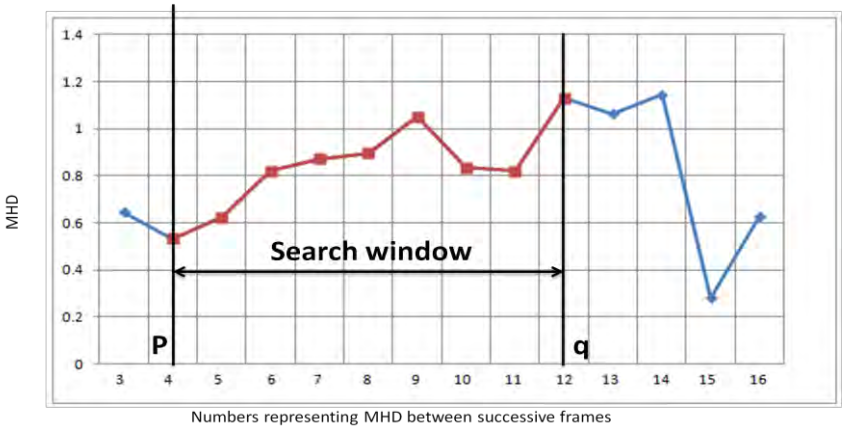


Figure 4.16 Search window definition within a block

In the x-axis of the plot above, 3 represents the MHD between 3rd and 4th gestures, 4 represents the MHD between 4th and 5th gestures and so on. The search window is then defined within the

redefined block (Figure 4.15) and starts at 4 and ends at 12 where the size of the block is reduced from 15 to 10. In Figure 4.15 a search window is defined for block A of Figure 4.12. As discussed above, the search space for CGS is localized in two steps:

- By dividing the whole sequence of image frames into blocks using the speed profile
- By creating a search window

As the search space is localized using the above mentioned steps, the proposed design doesn't use any threshold to select the candidate gesture rather it searches for a minimum difference within the search space and picks the first one of the gesture that resulted in that minimum gesture difference. Figure 4.17 shows the flowchart for selecting a candidate gesture.

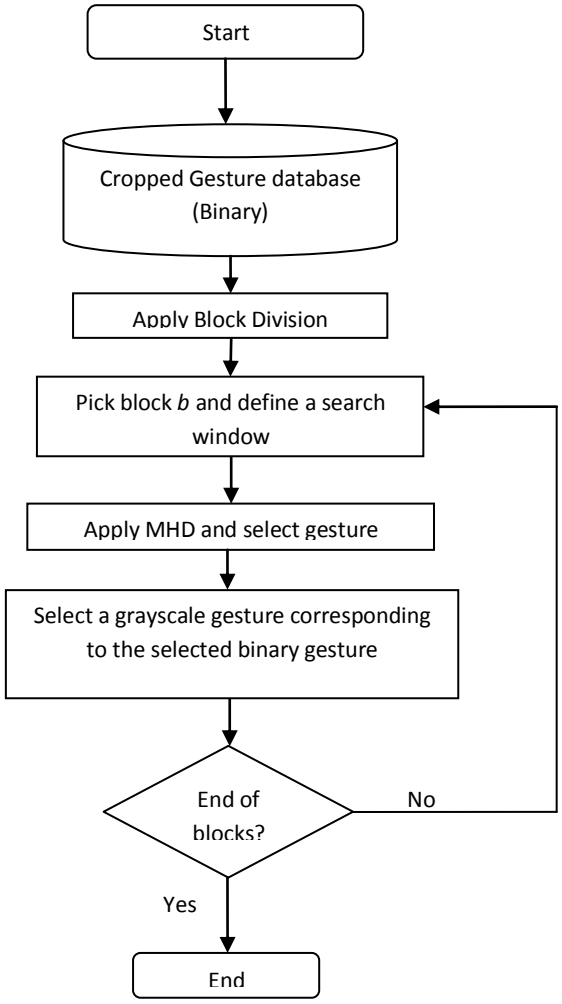


Figure 4.17 Flowchart for CGS

After creating a search window within a block, a MHD discussed in Section 2.5.3 is used to select a candidate gesture within the search window. This is done using the contour of the cropped hand gestures shown in Figure 4.18.

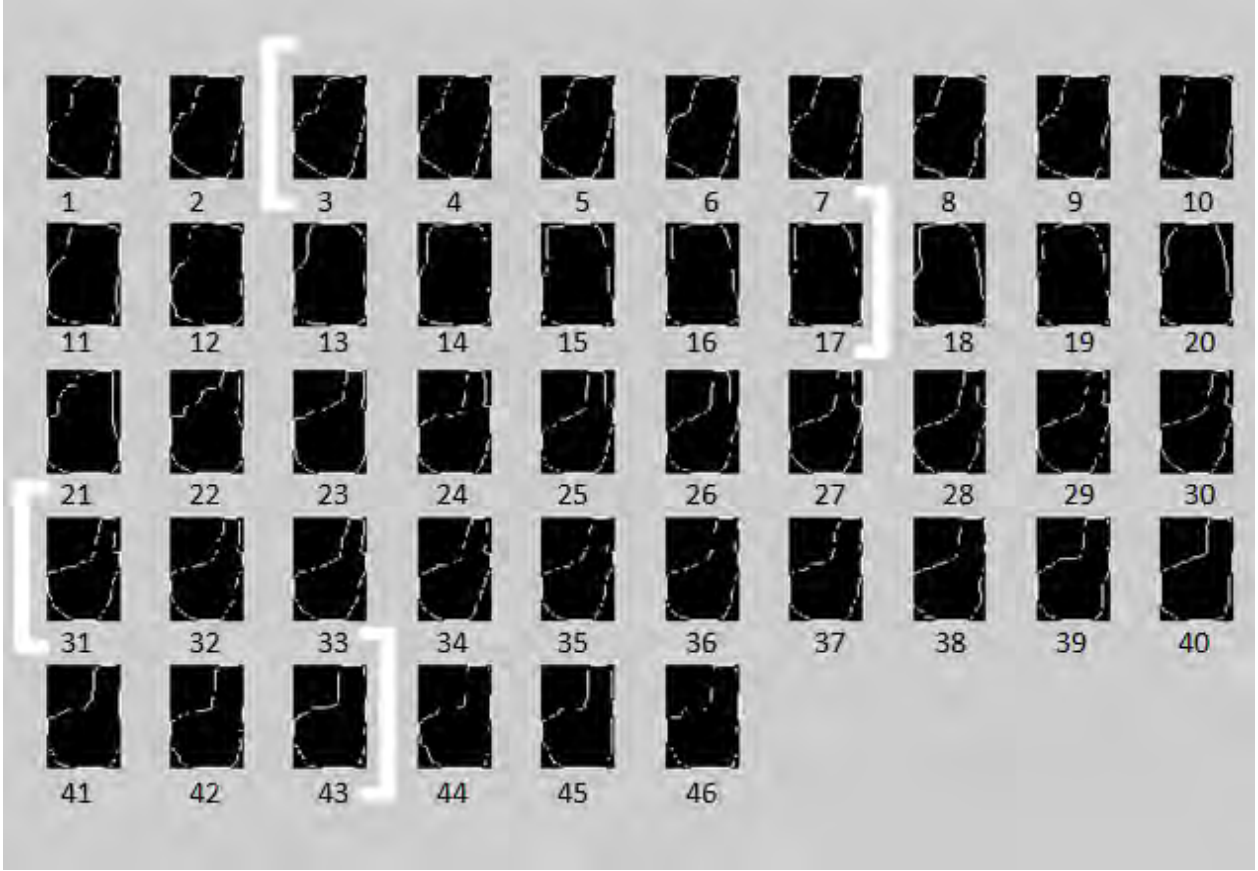


Figure 4.18 Contour of each cropped hand gestures for the name SUNNY (ሳኒ)

The comparison to select a candidate gesture is done using the contours of hand gestures displayed in Figure 4.18 but the output for CGS is picked from a corresponding grayscale shown in Figure 4.19.

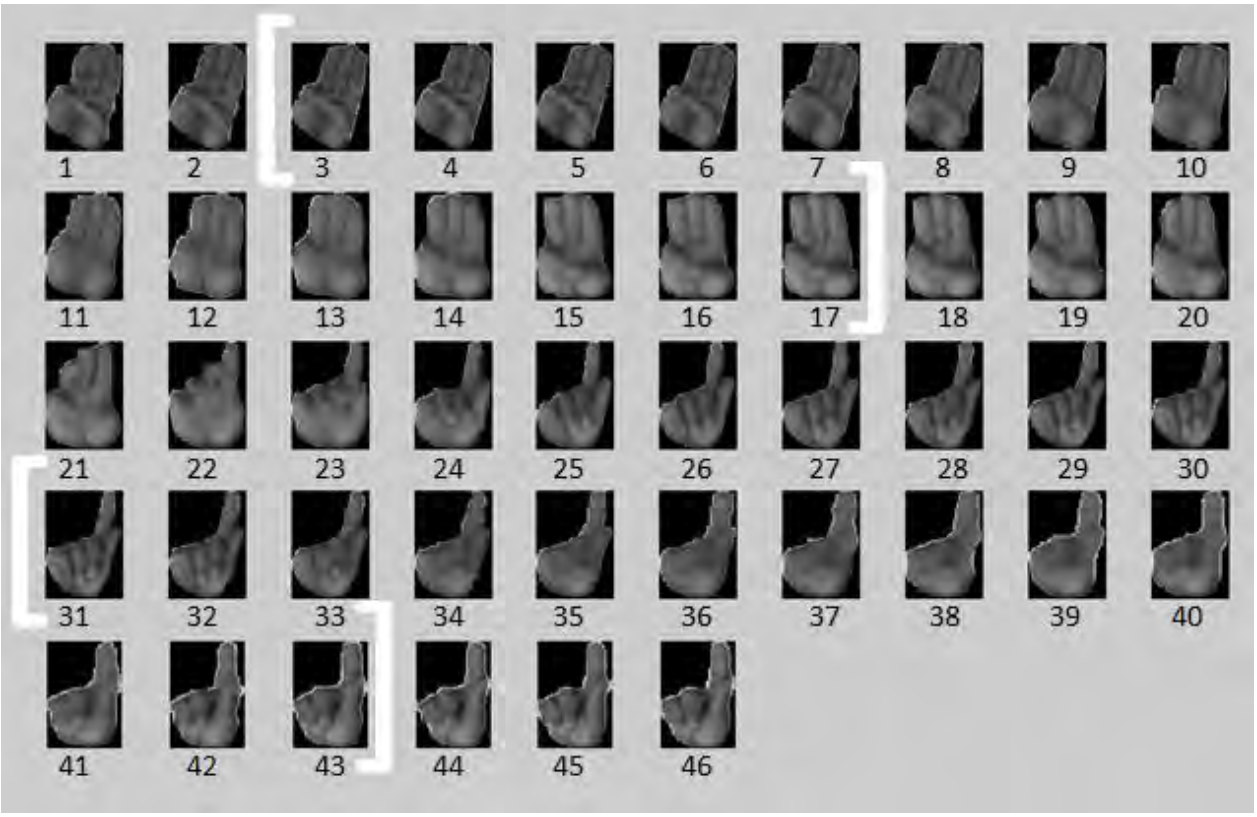


Figure 4.19 The corresponding grayscale image for Figure 4.16

For the search window created in block A of Figure 4.12, the output of the MHD using equation (2.25) discussed in 2.5.3 is shown in Table 4.7 below:

Table 4.7 Sample MHD output

Frames Compared	MHD
4 & 5	0.5324
5 & 6	0.6229
6 & 7	0.8209
7 & 8	0.8728
8 & 9	0.8969
9 & 10	1.0518
10 & 11	0.8341
11 & 12	0.819
12 & 13	1.1293

As the proposed design searches for the minimum value in the MHD output, the MHD between gesture 4 and 5 is the smallest in Table 4.7 and the gesture 4 is selected as a good candidate within the defined search window. No threshold is required to select the candidate after defining the search window. The output for the video of the name SUNNY (ሳኒ) is shown in Figure 4.20 where G-4 stands for gesture number 4 which is the 4th frame in the video frame sequence.

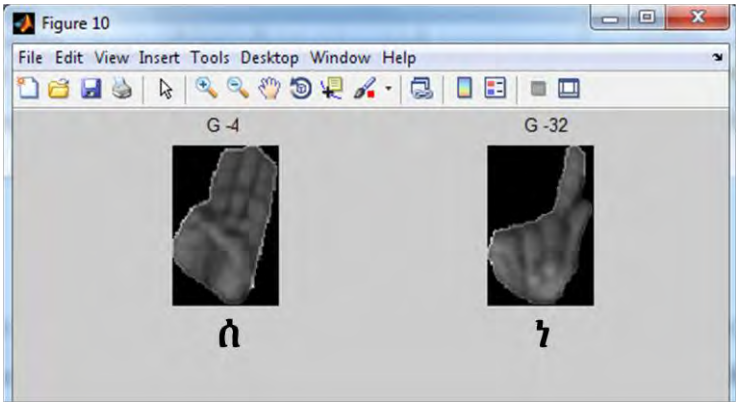


Figure 4.20 Output of the proposed system for candidate selection

The overall procedure for the CGS can be put as follows:

- Store the binary contour and the grayscale versions of the cropped hand gesture separately
- Divide the whole sequence of a video frame into meaningful blocks with the help of the speed profile
- Create a search window to minimize the error due to BD
- Perform the comparison between successive gestures within the search window
- Pick the first gesture that resulted in the minimum distance within the search window; if two or more values are equal to the minimum MHD, then the first occurrence of them is considered.
- Select the corresponding grayscale version of the selected hand gesture

4.7 Hand Movement Trajectory Determination

As explained in Chapter 1 Section 1.2, there are 6 forms of a single EMA other than the base alphabet where each variation is spelled using the base alphabet and a hand movement. So to determine which form of the alphabet is being spelled, it is important to track the hand and determine the nature and direction of the movement.

Before cropping the hand, centroid of each hand gesture extracted from each video frame was collected as explained in Section 4.4 of this chapter for hand tracking. The whole frame sequence was also divided into blocks where alternating block is assumed to represent a single alphabet. For a single video clip, the blocks as in Section 4.5.2 that represent the EMAs are considered for constructing hand trajectory. In Figure 4.12, blocks A and B represent single EMA each. The hand tracking module then considers the centroids corresponding to the image frames in blocks A and B for the EMAs.

There are only 6 types of trajectories used in EMA where the 7^{th} form is not considered in this work. As the types of the trajectories are very small, it is not feasible to use complex ways of hand tracking usually used in computer vision applications. However, by carefully observing the nature of the trajectories used in EMA, in the proposed design, angles and x- and y-directions of a line drawn between successive centroids are used as a cue to decide which trajectory is being formed using the signing hand. In Figure 4.21, sample trajectories are displayed for 2^{nd}, 3^{rd}, 4^{th}, 5^{th} and 6^{th} forms. Actually, the trajectories for the sequence of video frames of video clips are not smooth lines. They are just centroids of subsequent frames drawn in a single frame. However, for the purpose of visualization, the centroids are automatically connected to form continuous lines as displayed in Figure 4.21.

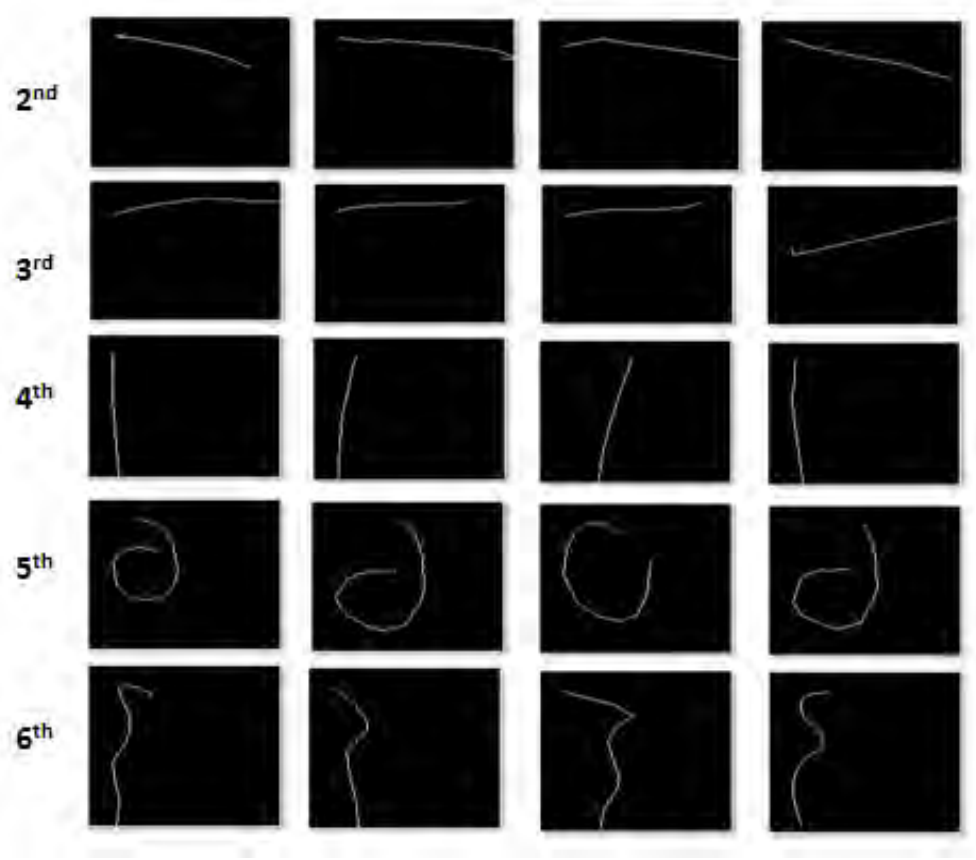


Figure 4.21 Sample trajectories

In general, the forms of the trajectories used in spelling EMAs are divided into four:

- Horizontal (straight) trajectory represents 2nd or 3rd forms
- Vertical (straight) trajectory representing 4th form
- Circular trajectory representing 5th form
- Vertical (zigzag or wavy) representing 6th form

The centroids within a block found in Section 4.5.2 are considered for HMTD. For 2nd, 3rd and 4th trajectories, the angles of lines formed between consecutive centroids are collected for all the centroids in the block considered. These angles show either the trajectory is straight vertically or straight horizontally. As the nature of the trajectories for 2nd and 3rd are horizontal lines, there should be a method to decide whether the trajectory is 2nd or 3rd. The proposed design adds the x-directions and uses the sum to determine the direction of the trajectory. This sum used to determine the direction of the trajectory is termed as Dominant Direction (DD). If DD is

positive, the trajectory is 2nd but if DD is negative, it is 3rd for right-handed person. And for left-handed person, if DD is positive, the trajectory is 3rd otherwise 2nd.

Angles between successive centroids for trajectories of 2nd form are shown in Table 4.8 and angles between successive centroids for trajectories of 3rd form are shown in Table 4.9. The Alphabets A, B, C etc each represent a trajectory.

An observation of Table 4.8 shows that majority of the absolute value of the angles for a trajectory is less than 30 degrees. The same is true for Table 4.9 that it is possible to have one governing rule for 2nd and 3rd form of trajectories.

Table 4.8 Angle history for 2nd trajectories

Angle history for a 2nd gesture							
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
0.00	0.00	0.00	-9.46	18.43	18.43	0.00	0.00
0.00	-11.31	-11.31	-20.56	39.81	18.43	0.00	0.00
-14.04	-4.09	-8.75	-23.96	20.56	14.04	-18.43	-11.31
11.31	0.00	-3.18	-12.99	24.44	6.34	14.04	-4.09
-8.13	0.00	-2.60	-7.13	15.95	10.30	0.00	90.00
90.00	4.40	5.44	-8.97	14.93	4.40	-8.13	0.00
0.00	3.81	-2.12	3.01	16.39	11.31	90.00	0.00
9.46	12.09	0.00	3.01	8.13	-5.19	6.34	7.59
4.40	10.30	0.00	4.76	8.13	-10.30	4.76	12.09
7.59		-45.00	5.71	0.00	0.00	4.40	10.30
12.09		-9.46	-6.34	0.00	90.00	11.31	26.57
23.20			10.01	14.04	0.00	12.09	
45.00			0.00		18.43	23.20	
			0.00			45.00	
			-45.00			-45.00	
			-26.57				
			-45.00				

Figure 4.22 helps much to visualize that the angles in Table 4.8 remain between -30 and 30. However, for the system to have a bit relaxed threshold, a range (-45,45) is used. In fact, there are some angles out of the range and for the system to ignore such unexpected occurrences in the angle history, 75% of the total number of angles are expected to be in the range (-45, 45).

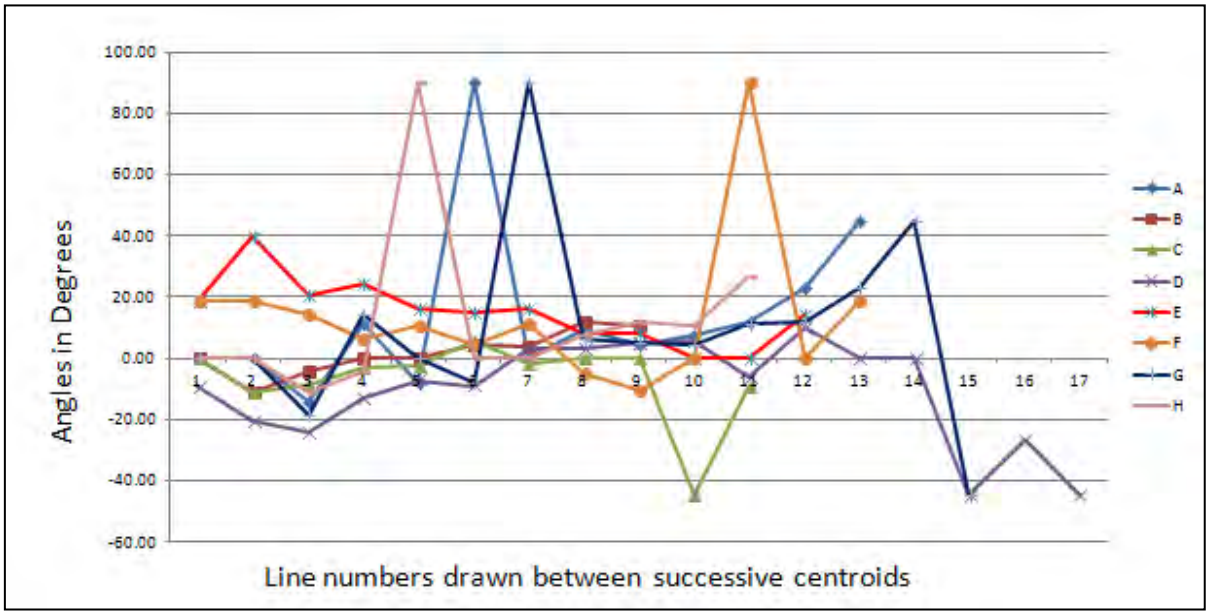


Figure 4.22 Plot for Angle history in Table 4.8

Table 4.9 Angle history for 3rd trajectories

Angle history for a 3rd gesture								
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
0.00	0.00	45.00	45.00	0.00	18.43	18.43	-45.00	18.43
-18.43	-26.57	45.00	5.19	0.00	14.04	21.80	-90.00	18.43
-11.31	0.00	0.00	-5.71	45.00	0.00	16.70	33.69	39.81
-6.34	-9.46	14.04	10.30	18.43	6.34	8.75	90.00	26.57
0.00	0.00	14.04	4.09	21.80	11.31	-90.00	33.69	19.98
0.00	0.00	9.46	13.24	18.43	4.76	6.12	0.00	15.95
-4.76	-5.71	16.70	11.07	20.56	14.04	15.26	18.43	14.93
-4.09	-2.49	9.46	8.75	6.34	11.31		11.31	16.39
0.00	0.00	10.01	15.42	11.31	11.31		18.43	8.13
-11.31	0.00	19.98	13.32	10.30	12.09		14.04	8.13
-7.13	-4.76	4.09	14.04	18.43	6.71		22.25	0.00
-11.31	-6.34	16.19		4.40			8.43	90.00
-63.43	-7.13	10.62		0.00			15.71	14.04
		24.44					14.04	
		7.13					19.98	

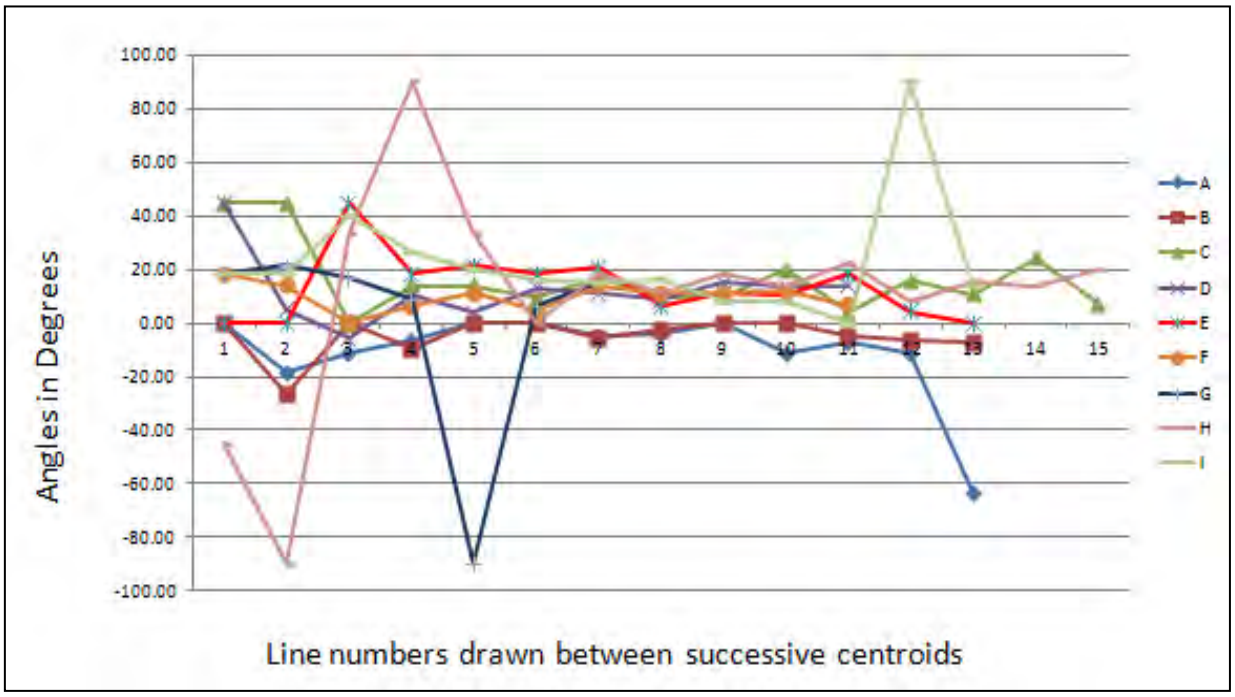


Figure 4.23 Plot for Angle history in Table 4.9

The trajectory for the 4th EMA forms has a vertical nature. Table 4.10 shows the angle history for several 4th trajectories. Each of the columns in the table represents a single trajectory of the 4th type.

Table 4.10 Angle history for 4th trajectories

Angle history for a 4th gesture							
A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
-90.00	-68.20	0.00	-78.69	-63.43	45.00	-45.00	-45.00
-45.00	-85.60	90.00	-82.87	-81.87	56.31	-63.43	90.00
-75.96	-79.99	-80.54	90.00	90.00	63.43	-68.20	85.60
-74.05	-82.57	84.29	-88.32	-85.60	77.47	-75.96	-17.66
-79.70	-85.43	90.00	90.00	90.00	79.70	-78.69	-30.54
-82.41	88.03	90.00	87.14	-85.43	90.00	-79.88	88.32
-79.99	85.43	86.82	90.00	-87.71	87.51	0.00	86.73
-81.87	84.56	86.82	87.27	-85.43	90.00	-83.88	90.00
90.00	78.11	86.63	86.42	-85.03	87.80	-85.60	90.00
90.00		86.19	90.00	90.00	90.00	-84.56	90.00
83.66		86.63	90.00	-87.40	87.95	90.00	-77.91
		90.00			90.00	90.00	-56.31
		-63.43			90.00	-75.96	90.00

A close observation of each column of Table 4.10 reveals that most of the absolute value of the angles remains greater than 80 degrees. However, to include trajectories that may result in angles less than 80 degrees, the proposed system uses 60 degrees as a discriminating angle. The plot in Figure 4.24 helps to visualize the discrimination.

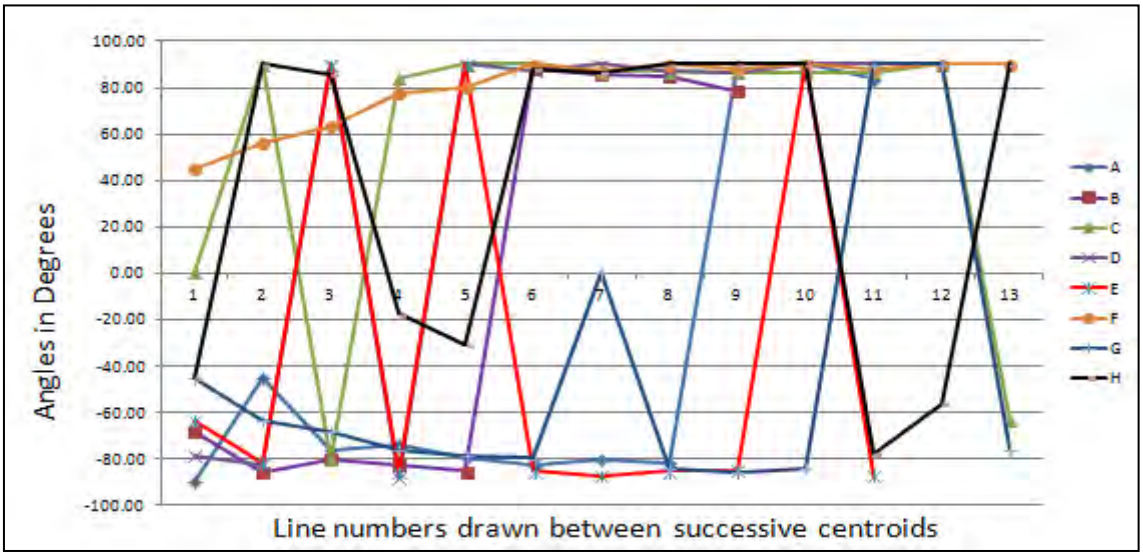


Figure 4.24 Plot for Angle history in Table 4.10

The Situation for the 5th and 6th trajectories is different from that of 2nd, 3rd and 4th. As shown in Table 4.11, each of the trajectories is represented using alphabets A, B, C etc for the 5th EMA form have two directions. Trajectory of form 5 first moves in the positive y-direction which is downward and then moves in a negative y-direction in a circular manner as shown in Figure 4.25 where point Q in the figure shows the starting point.

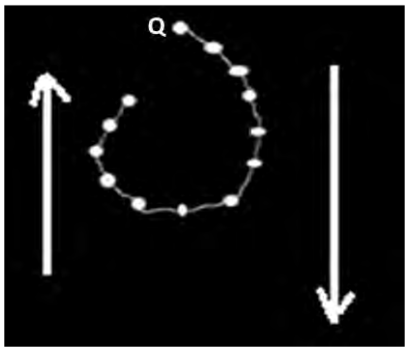


Figure 4.25 Hand trajectory for 5th form of an EMA with the positive and negative movement directions

Table 4.11 shows the y-direction of a line drawn between successive centroids of 5 trajectories and the negative y-directions are shaded.

Table 4.11 Direction history for 5th trajectories

Direction history for 5th trajectory					
	A	**B**	**C**	**D**	**E**
Magnitude and direction of y-axis movement	11	19	10	17	9
	10	16	12	14	20
	15	15	11	26	0
	12	11	31	15	28
	13	10	14	27	37
	17	6	22	6	30
	2	-3	4	-4	22
	-3	-6	1	-10	0
	-4	-10	-4	-6	7
	-9	-9	-10	-6	-4
	-20	-20	-6	-24	-17
	-10	-8	-6		-22
	-5	-5	-9		0
		-3	-15		-18
			-6		-14
			-3		-3
			-1		-1
Percentage of Negative direction	**46.15%**	**53.33%**	**52.94%**	**45.45%**	**47.06%**

The above set of 5th form trajectories were taken from the training videos and the percentage of negative y-directions are computed to be used as a discrimination of these types of trajectories. This percentage of negative y-directions is used to discriminate the 5th form because the other forms do not show this behavior. By observing Table 4.11 and to include trajectories that have even smaller but significant percentage of negative y-directions, 30% is used as a boundary for discrimination.

The nature of the trajectory for the 6th form has a positive y-direction and alternating x-directions because it has a zigzag form with a downward direction as shown in Figure 4.26. The x-direction history for five trajectories is shown in Table 4.12. To successfully discriminate the 6th form from the other forms, the dominant y-direction together with the alternating x-directions is considered.

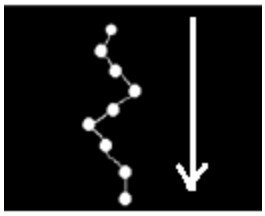


Figure 4.26 Hand trajectory for 6th form of an EMA with one dominant direction

Table 4.12 Direction history for 6th trajectories

Direction history for 6th trajectory					
	A	**B**	**C**	**D**	**E**
Magnitude and direction of x-direction movement	0	4	-2	-2	5
	-6	4	-4	-5	8
	-7	7	-5	-5	9
	-11	6	-2	-5	8
	-3	7	0	-4	5
	-1	4	1	-1	1
	11	-1	1	0	-3
	5	-5	-4	2	-8
	3	-9	-7	5	-11
	-4	-7	-4	6	-11
	-6	-6	0	4	-8
	0	1	-5	0	-1
	0	-1	-2	-2	0
		1	0	-12	4
		0	-1	0	3
		-1		-6	2
		2		-3	1
		4		0	3
		3		2	6
		3		3	6
		1		1	4
Number of sign changes	**4**	**6**	**4**	**5**	**2**

A careful observation of Table 4.12 reveals the alternating direction of x-axis movement for the trajectories. Similar to the trajectories for the 4th form, this form of trajectory has a dominant direction in positive y-direction. By combining the nature of the directions in x-axis and y-axis, the 6th form is successfully discriminated from others. As a threshold for sign changes, s = 2 is

used with 70% of the y-direction movements are considered positive. The overall procedure of HMTD is depicted by a flowchart in Figure 4.27.

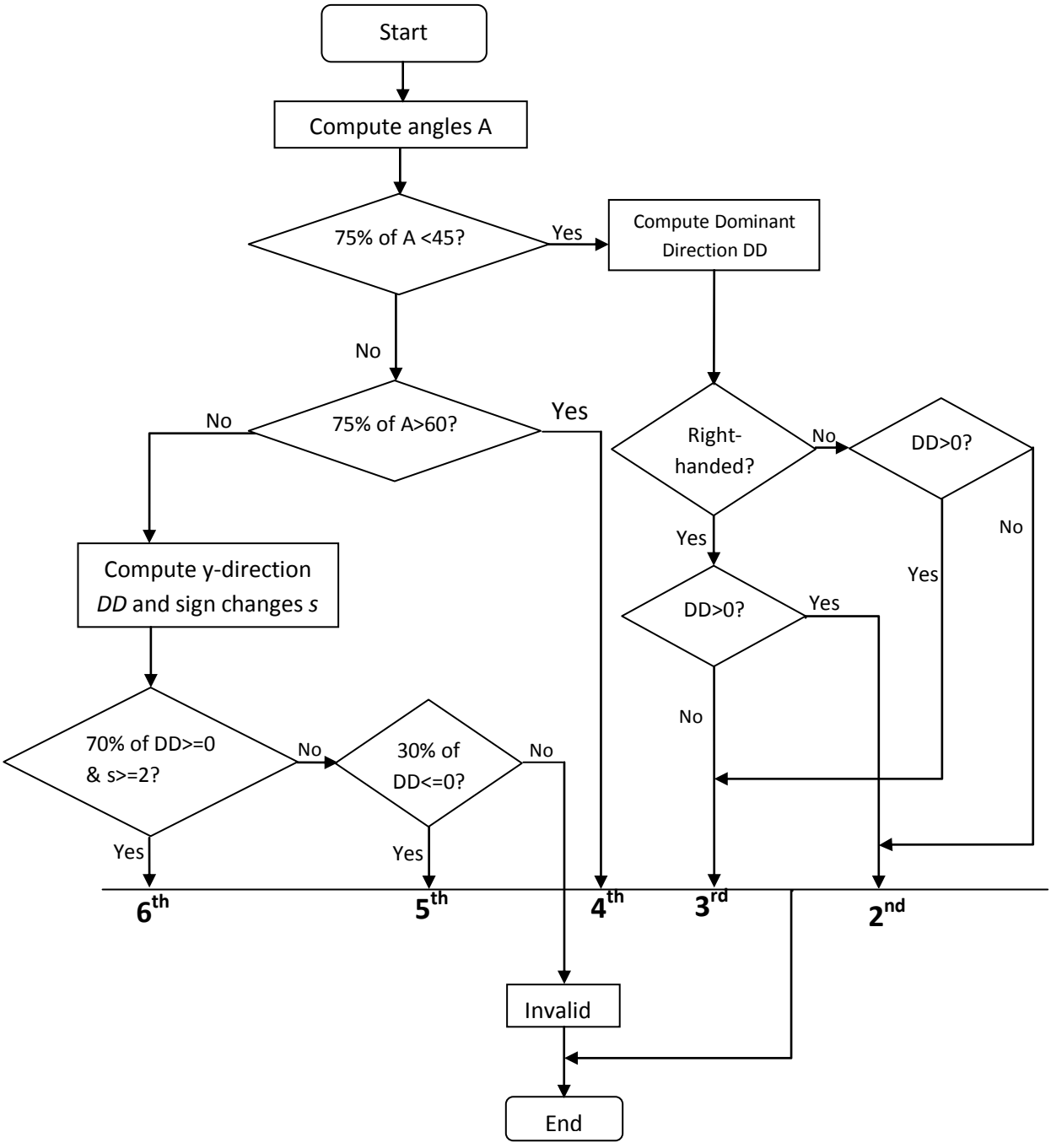


Figure 4.27 Overall flowchart for HMTD

CHAPTER 5

EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Block division and Candidate gesture selection

5.1.1 Block division

To localize the problem of CGS, sequence of image frames of a given video clip is divided into blocks based on speed threshold. In the proposed design, 70 videos of the words in Table 4.1 of Section 4.2 were examined to come up with a speed threshold of **0.155** pixels per millisecond. This speed threshold is used to divide the whole sequence of image frames into meaningful blocks where each block represents either a valid gesture or a return path (transition). The blocks that represent valid EMAs are called EMA blocks while the others are called transition blocks which represent gestures in the return path.

As presented in Section 4.1 of Chapter 4, 5 signers were considered for the system design. Fourteen video clips for each of the five signers were used in the experiment to come up with a speed threshold of 0.155 pixels per millisecond as shown in Table 4.6 by the shaded row.

5.1.2 System test for block division

Using the obtained speed threshold which is **0.155** pixels per millisecond, the proposed design was evaluated with the videos recorded for the words in Table 4.2 of Section 4.2 for BD. 74 videos were used to test the system for BD. Table 5.2 shows the experimental results of the proposed design for BD.

Table 5.1 Results of system test for BD

Signer	Number of videos	Threshold	Correct	Accuracy
signer 1	10	0.155	9	90.00%
Signer 2	10	0.155	9	90.00%
signer 3	10	0.155	8	80.00%
Signer 4	22	0.155	18	81.82%
Signer 5	22	0.155	19	86.36%
Average	**74**	**0.155**	**63**	**85.14%**

Table 5.2 shows the results of system test for BD. As explained in Section 5.1.1, the nature of the data recorded is important for effective BD. For this purpose the signer should be oriented to avoid hand motions that are abrupt and a sudden pause in the middle of signing. With the videos used in the system test, the average performance of the proposed design for BD is found to be 85.14%.

5.1.3 System test for Candidate gesture selection (CGS)

After BD, the next task is to select candidate gestures from the alternating blocks. As explained in Section 4.5.2 of chapter 4, MHD is used to compute gesture similarity for CGS. By dividing the whole video frame sequence into blocks in Section 5.1.2, the gesture similarity comparison is localized to a smaller group than to the whole sequence. Considering the video for a 2 EMA word in Figure 4.12, similarity computation is done in blocks A and B to select one candidate gesture from each block. The performance evaluation of the proposed system for CGS is a qualitative analysis and considers the outputs of the BD stage which are regarded as correct.

For good CGS, the proposed design localizes the search area by creating a search window within a block as in Figure 4.16. Figure 4.16 is constructed by computing the frame difference using MHD between consecutive gestures within a block. After computing the MHD, the minimum difference is considered and the first of the gestures that resulted in the minimum difference is selected as a good candidate. In case more than one equal values are obtained for the minimum variable, the first occurrence of these values is considered. Five people were involved in the qualitative performance analysis of the proposed design. There were 154 EMAs and 770 observations from 5 people out of which 730 observations were correct. The experimental results for the CGS are presented in Table 5.2.

Table 5.2 Results of system test for CGS

Signer	Number of EMAs	Number of people	Total observation	Correct responses	Accuracy
Signer 1	23	5	115	110	***95.65%***
Signer 2	23	5	115	95	***82.61%***
Signer 3	20	5	100	90	***90.00%***
Signer 4	42	5	210	210	***100.00%***
Signer 5	46	5	230	225	***97.83%***
Average	**154**	**5**	**770**	**730**	**94.81%**

As can be seen from Tables 5.2 and 5.3, the CGS module has an accuracy of 85.14% * 94.81% = 80.72%. This multiplication is valid because the video clips that were correctly divided into valid blocks are considered in the qualitative analysis of the CGS.

5.2 Hand Movement Trajectory Determination (HMTD)

Before cropping the hand gesture in Section 4.6, the centroid of each gesture was collected as stated in Section 4.5 for the purpose of determining hand trajectories. For this module, the videos whose sequence of images was divided correctly in Section 5.1.2 were used as input.

Table 5.3 Experimental result of system test for HMTD

Signer	Number of EMA trajectories	Correct trajectories	Accuracy
Signer 1	23	21	***91.30%***
Signer 2	23	17	***73.91%***
Signer 3	20	16	***80.00%***
Signer 4	42	39	***92.86%***
Signer 5	46	43	***93.48%***
Average	**154**	**136**	**88.31%**

As shown in Table 5.3, the accuracy of the system for HMTD is 88.31%. In fact, the proposed system is able to determine trajectories more accurate than shown here for EMAs whose trajectories are carefully made. This is possible if the signer is aware that he is "talking" to a computer so that he/she signs clearly. In the data used in this work, some of the signers do not show clear trajectories for the 6^{th} EMA for the system to determine.

5.3 Overall System performance

The proposed design has two separate outputs where two of them should be correct so that the system output will also be regarded as correct. The first is the output of the CGS and the second

is the output of the HMTD. The accuracy of the CGS is found to be 80.72% and that of the HMTD is 88.31%. However, it would be wrong to multiply the CGS by the HMTD because there may be sometimes at which the CGS is correct but the HMTD is incorrect or vice versa as a result the system output will be incorrect or there may be times at which both the CGS and HMTD are incorrect for a single EMA. To calculate the overall system performance the CGS and the HMTD results of a single gesture should be considered altogether. The incorrect and the correct approaches are shown below:

The incorrect approach

Overall Performance = CGS * HMTD

= 80.72% * 88.31%

= 71.28%

The correct approach

As shown in Table 5.3, out of 154 trajectories, 136 were found to be correct. However, the quality of the gesture whose trajectory is regarded as correct may not be found to be satisfactory. Therefore, multiplying the CGS by the HMTD is a wrong approach. So a separate analysis of the system output both for the qualitative (CGS) and quantitative (HMTD) gives a different result. First, for each correct output of the BD, output of CGS and that of HMTD should be correct altogether. And the percentage of the correct occurrences is multiplied with the BD. Hence, even if there are 146 qualitatively satisfactory gestures and 136 correct trajectories, only 130 of them are correct in both the qualitative result and the HMTD. So the overall system performance is:

Overall Performance = 130/154 * 85.14

= 84.42% * 85.14

= 71.88%

The logic made above for the correct approach is a bit tempting that a decrease in the overall performance is expected. However, it has slightly increased. This is because, in the qualitative and quantitative analysis of the system, EMAs whose CGS and HMTD outputs were found incorrect and this leads to count the error as 2 for a single EMA. Whereas, when the CGS and HMTD outputs are taken together, the error is counted as 1. This implies either the correct or the incorrect approach could be larger depending on the test data.

5.4 System outputs and discussion

The final outputs of the proposed design are group of selected grayscale hand gestures for each EMA in a video clip with corresponding hand trajectory number. The trajectory numbers determine which form of the given EMA is being created in the video clip. Figure 5.1 shows an output of the proposed design for a video clip of the name EZANA (ኢዛና). For example, the second gesture in Figure 5.1 represents the EMA ዘ where when trajectory number 4 is associated with it, the EMA becomes ዛ.

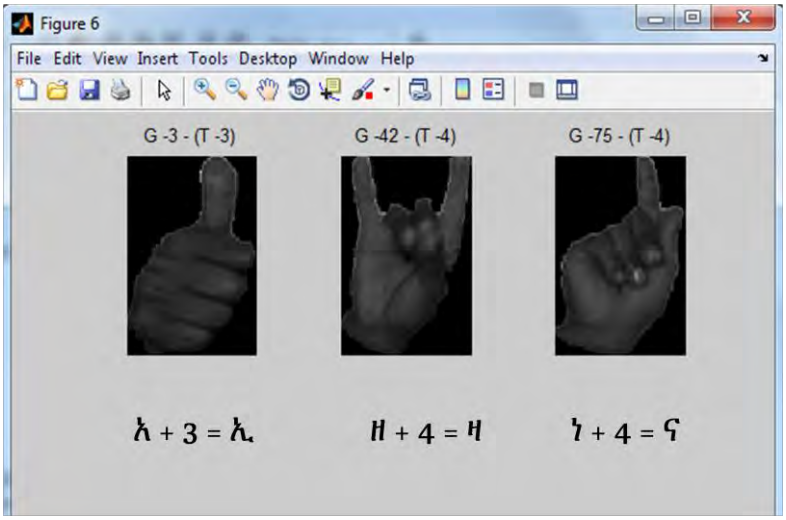


Figure 5.1 Sample output of the proposed design for a video clip of the name EZANA (ኢዛና)

In Figure 5.1 above, the three gestures which represent three EMAs are extracted from 85 video frames. As shown in the figure, G-42 represents the 42nd gesture in the sequence and T-4 stands for the trajectory number 4. The outputs of the proposed design are used as inputs to an EMA

recognition system where the selected base EMAs are recognized and then the system picks the correct form of the EMA from a list such as Table 5.4. This table, in fact, doesn't show an exhaustive list of the EMAs that exist in EthSL.

Table 5.4 List of some EMAs

Base EMAs	Forms of sample EMAs (Gesture Number)					
	2	3	4	5	6	7
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ

Table 5.4 shows four EMAs with their other 6 forms. After a recognition system recognizes the base EMAs, it will pick a correct EMA form based on the trajectory number used as input from this proposed design.

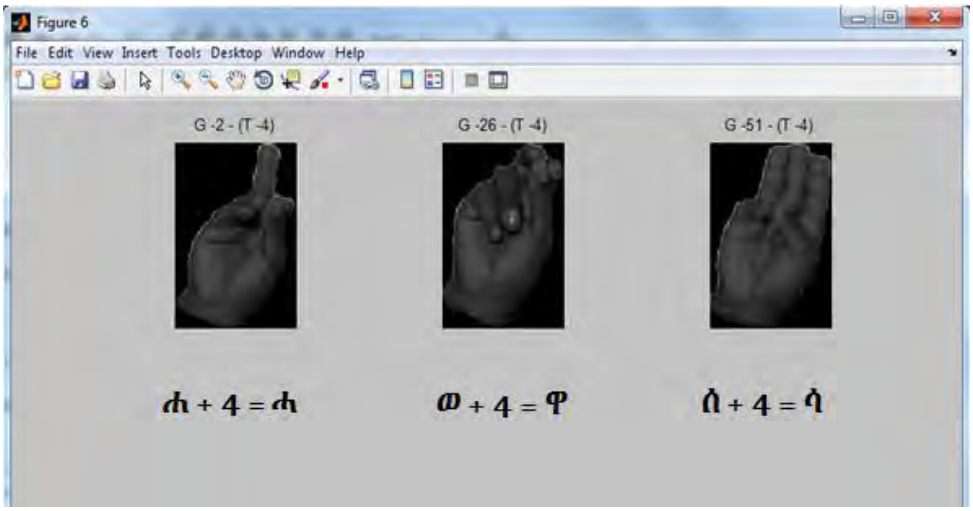


Figure 5.2 Sample output of the proposed design for a video clip of the name HAWASSA (ሓዋሳ)

For example, if a recognition system recognizes the second gesture in Figure 5.2 as **ወ,** the corresponding number which represents a gesture number will be associated with the recognized EMA and a corresponding EMA form will be picked from such as Table 5.4.

Two different outputs are shown in Figure 5.3 for the same word EZANA (ኢዛና) but different videos. It can be seen that the proposed design selects candidate EMA gestures for both videos which have different speed of signing. In the first case, G-3, G-42 and G-75 are selected from the given sequence of video frames. G-2, G-30 and G-53 are selected for the second case.

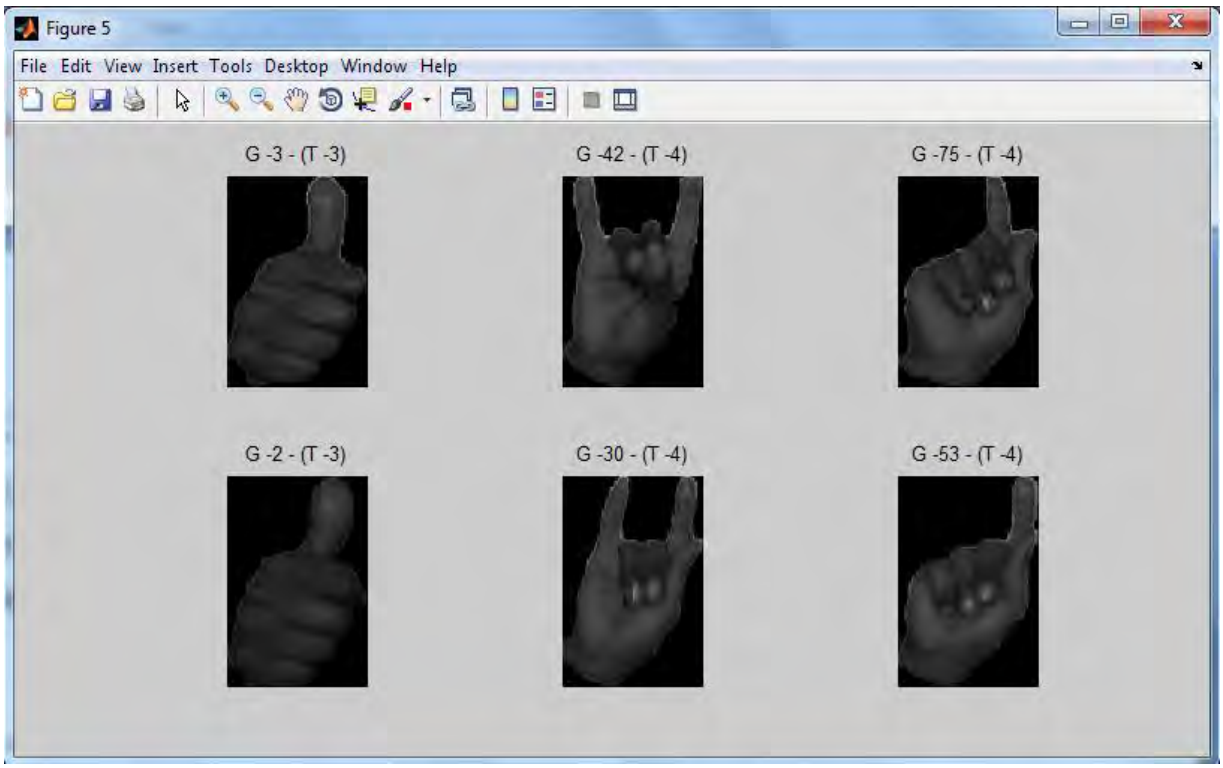


Figure 5.3 Sample output of the proposed design for videos with different speed of signing

In Figure 5.3, it is shown that the proposed design can successfully select good candidate gestures for signers with different speed of signing. In fact, the speed of signing should be the natural speed that signers use during communication. It should not be too fast or too slow.

When a system considers a frame difference over all a video frame sequence, it would be not convenient for words with repeated finger spellings. This is because the system can pick one finger spelling even if they are repeated.

The proposed design makes use of speed profile to divide the given sequence of video frames into number of blocks that represent either an EMA or a return path to the point where signing started. By doing this, the proposed design is able to isolate the repeated EMAs and select one candidate for each one. The output of the system for the name KUKU (ḫ-ḫ-) together with their corresponding trajectories is displayed in Figure 5.4.

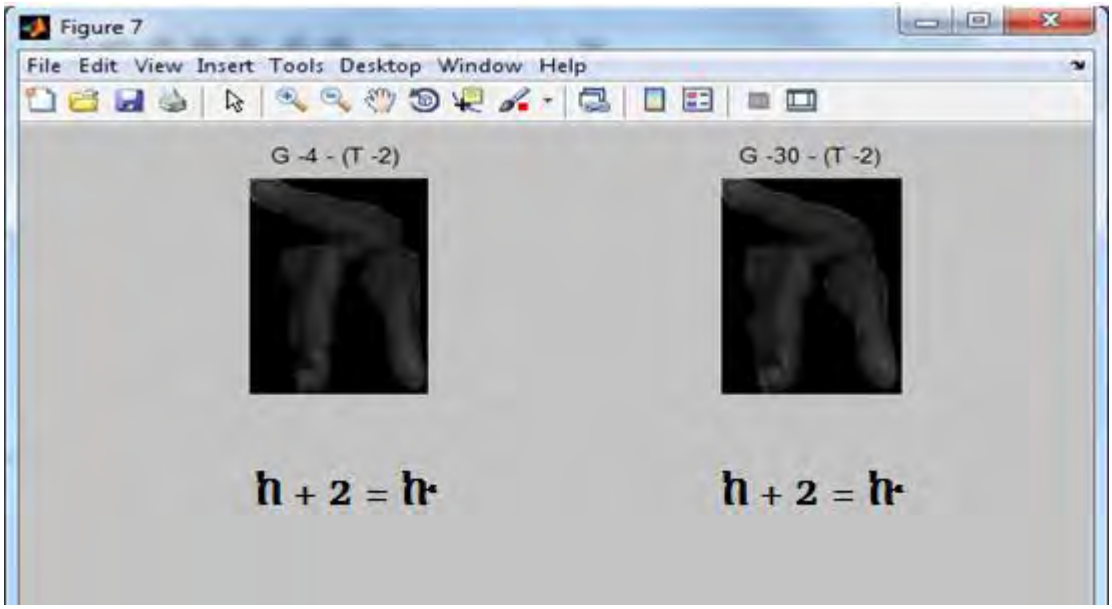


Figure 5.4 Sample output of the proposed design for a word the name KUKU (ḫ-ḫ-) with a repeated EMA

As explained in Section 4.4 of Chapter 4, the proposed design makes a natural assumption that the face region is usually larger than the hand region. Based on this assumption, the system successfully isolates the hand by removing regions other than the second largest in the segmented image. However, in certain cases the hand region is found to be larger than the face region. Some of the situations where the hand region is found to be larger than the face region are:

- If the hand is larger than the face naturally
- If the hand and the face are exposed to different lighting conditions while recording the video which affects the skin segmentation
- If the signer signs with the hand very close to the camera

When the proposed design encounters the third case just mentioned, the output is shown in Figure 5.5.

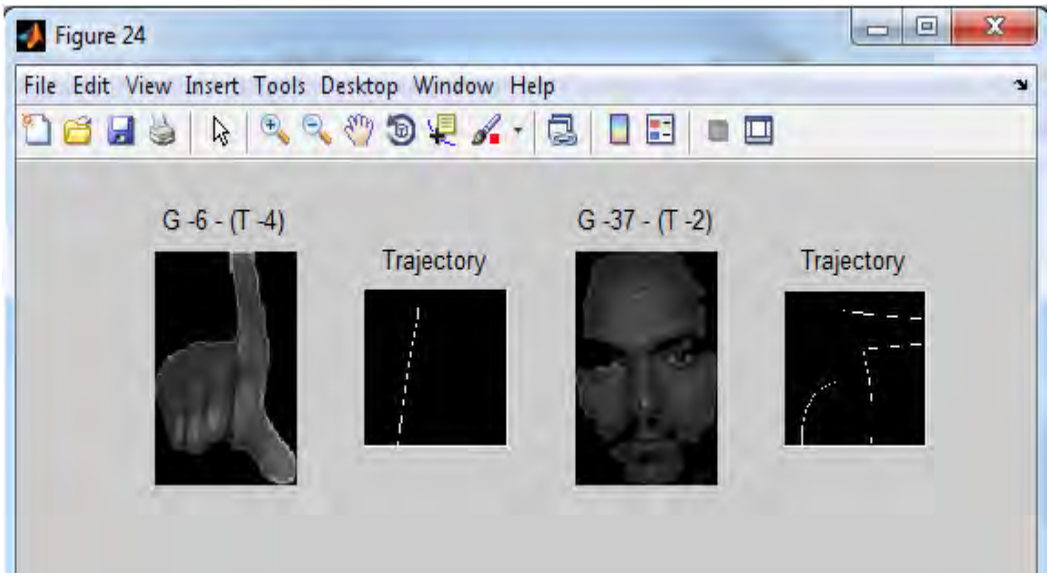


Figure 5.5 Sample output of the proposed design for a situation that invalidates the basic assumption used

The quality of the system output heavily depends on the quality of the video recorded. In fact, the candidate selection and the HMTD depend largely on the signing nature of the signers. As mentioned in Section 4.2, the signers should be given orientation to avoid unexpected movements such as hand flinging and sudden pause in the middle of signing. With this orientation, the proposed design works fine for candidate selection and HMTD even if the quality of the video processed is very poor. The output of the system for videos with poor quality is shown in Figure 5.6 and Figure 5.7.

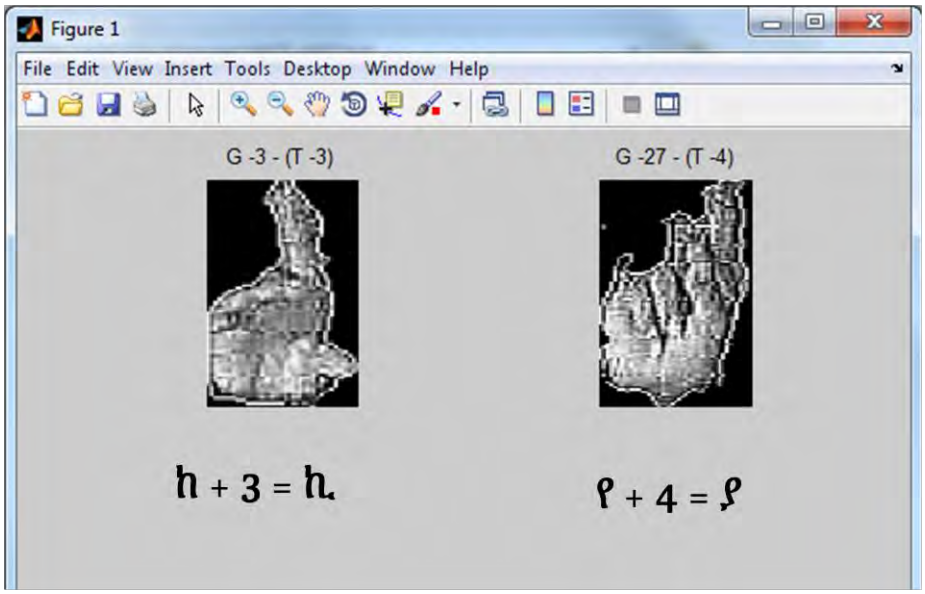


Figure 5.6 Sample output of the proposed design for a video with poor quality

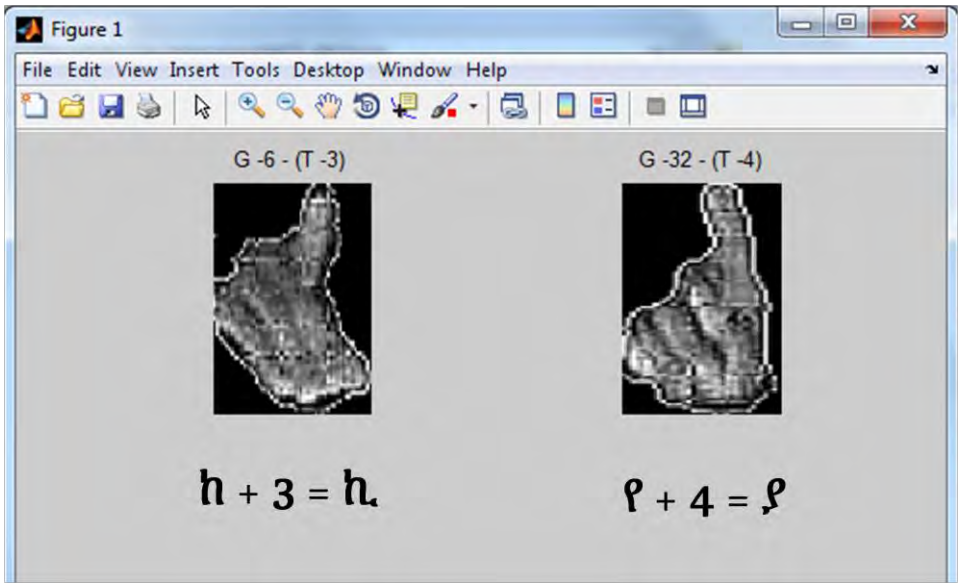


Figure 5.7 Sample output of the proposed design for a video with poor quality

If the RGB versions of the EMAs are required, the proposed design has outputs like shown in Figures 5.8 and 5.9.

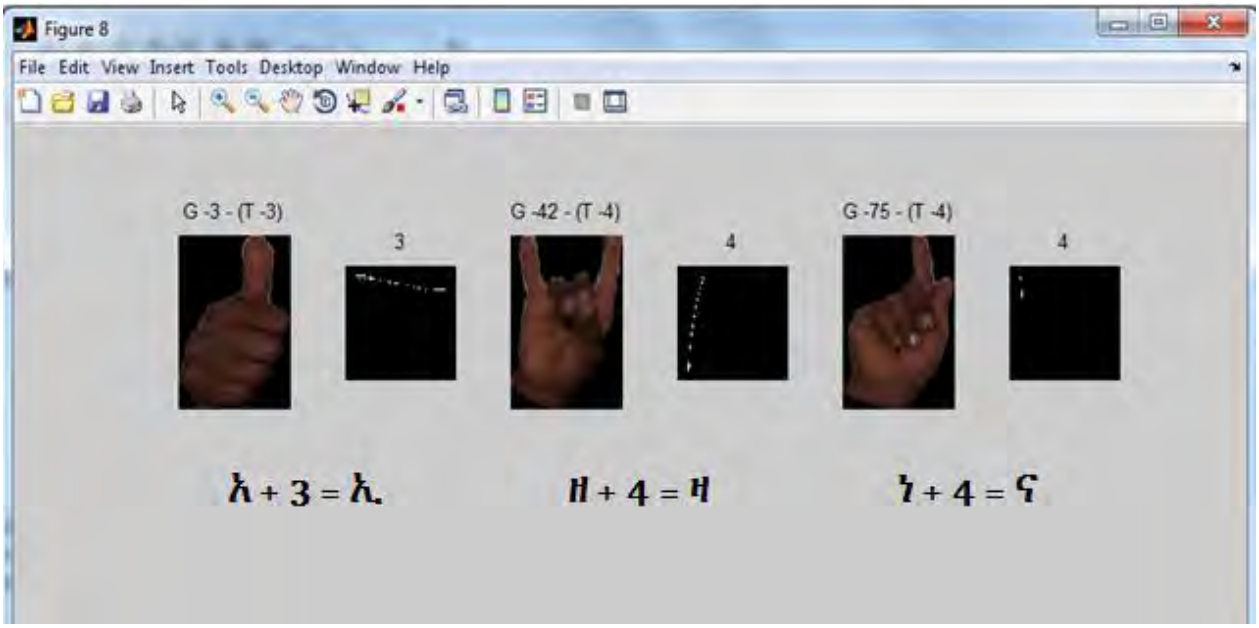


Figure 5.8 An RGB version output for a video clip of the name EZANA (ኢዛና)

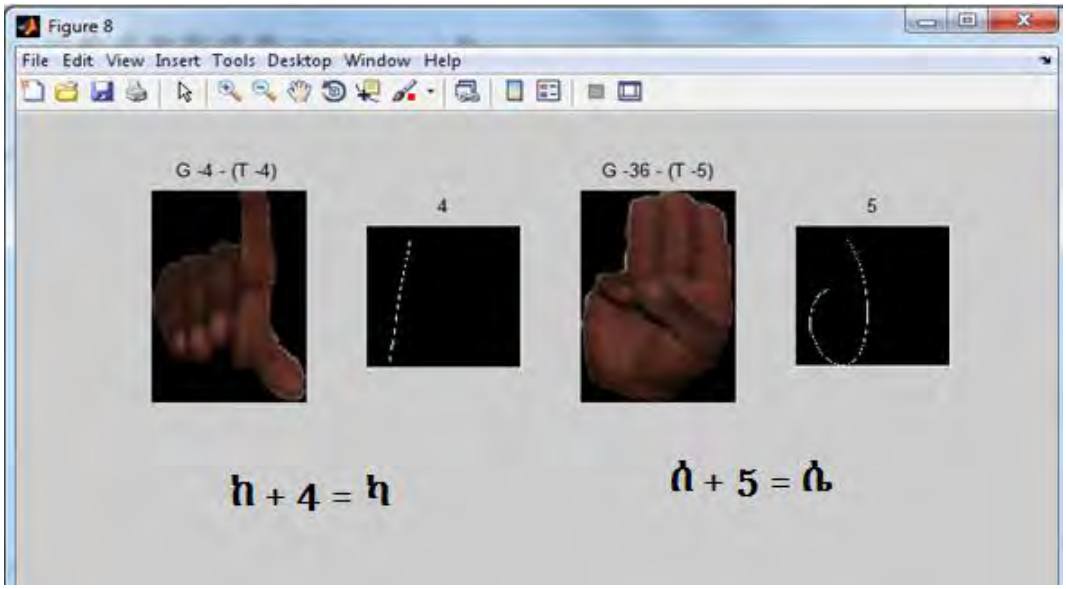


Figure 5.9 An RGB version output for a video clip of the name KASIE (ካሴ)

CHAPTER 6

CONCLUSION, FUTURE WORKS AND CHALLENGES

6.1 Conclusion

In this thesis work, a system that extracts important hand gestures that represent valid EMAs and determines their hand movement trajectories is developed. The system comprises image segmentation, blob analysis for hand isolation, CGS, and hand movement HMTD. YCbCr color space is used for skin-color segmentation because the discrimination between skin and non-skin pixels in the Cb-Cr plane is very good and the computational expense is less. With the color space used, the system is able to segment the image effectively. Following the segmentation is the blob analysis for hand isolation.

The hand isolation module heavily depends on the segmentation stage because it considers blob sizes to isolate the hand. If the segmentation is poor, the hand isolation will also be poor. With the data collected for system design and test, both the skin-color segmentation and the hand isolation are effective.

The CGS uses a speed profile of gestures to divide a sequence of image frames of a video into blocks that represent EMAs and return paths. This approach localizes the candidate search space to a smaller block than to the whole sequence of images. The proposed design has 85.14% of BD accuracy. After dividing sequence of image frames into blocks, a search for EMA is done in alternating blocks. The proposed design introduces a search window for effective candidate search and has an accuracy of 94.81%. The overall accuracy of the CGS module is 80.72%.

The HMTD module uses the angle and direction information from the centroids with a given EMA block. For final decision of trajectory, angle history between successive centroids is computed and the x- and y-directions of the trajectories are considered. This module has an

accuracy of 88.31%. The overall system performance is 71.88%. The reasons for the low overall system performance are:

- Poor video quality
- Some problems with the signers' way of signing such as hand flinging

6.2 Future work

The following things can be considered as future work

- In this thesis, candidate hand gesture selection and trajectory determination for the selected EMA is done. The outputs of the proposed design can be used as an input to a recognition system where a word or sentences level recognition is required.
- The signers have been oriented to avoid overlapping between hand and face. Therefore, to make the signers free from this constraint, the concept of digital image processing called occlusion can be used to separate overlapping objects or colored gloves can be used for a good segmentation even when there is overlapping between hand and face.

6.3 Challenges

To meet the objectives of the thesis and to make the accuracy of the proposed design an excellent one, a lot of effort was required. This effort, in fact, has been exerted exhaustively to make the proposed design real. In course of this work, the main challenge faced was during the data collection. The problem was encountered due to the communication gap between my data sources and me. Honestly, I can use the EMAs to spell what I wanted to say to them but EMAs only are not enough for communicating with the deaf. Due to this communication gap, some of the videos I have recorded were not found suitable to the proposed design.

Another challenge was pertaining to budget insufficiency. As was explained in the proposal of this thesis, there were many times I should meet my data sources and this meeting of people required me its own expenses. This is because, when working together, people should feel very

friendly and will relax when standing in front of a camera. I have found some of my data source as shy and get nervous when I dictate them to spell words.

If the above challenges are overcome, the proposed design will be more accurate.

6.4 Limitations

The approach followed in this work is very good. But still it has its own limitations in BD. As explained in Section 1.2 and depicted in Figure 1.1, the 5th EMA is spelled by associating the base EMA with a circular hand trajectory. For the rest of the trajectories except for the 5th form, the hand motion has a pause at a different point where it has started. However, the 5th form usually ends at a point where it has started. Therefore, when an EMA is spelled after a 5th form, it immediately starts where there will be insignificant transition motion between the 5th form and the next one. The proposed design is therefore not effective in BD for words that have a 5th form preceding any other form such as YARED (ያሬድ).

REFERENCES

[1] Yang quan, "Chinese Sign Language Recognition Based on Video Sequence Appearance Modeling", 5th IEEE Conference on Industrial Electronics and Applications, 2010

[2] Maryam Pahlevanzadeh, Mansour Vafadoost, Majid Shahnazi, "Sign Language Recognition", http://www.osun.org.

[3] Mohamed Mohandes and Mohamed Deriche, "Image based Arabic Sign Language recognition", 6th IEEE Trans. 978-1-4244-5046, 2010.

[4] Justus Piater, Thomas Hoyoux, Wei Du, "Video Analysis for Continuous Sign Language Recognition", 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies, 2010.

[5] Yonas Fantahun Admasu and Kumudha Raimond, "Ethiopian Sign Language Recognition Using Artificial Neural Network", 10th International Conference on Intelligent Systems Design and Applications, 2010.

[6] Azadeh Kiani Sarkaleh, Fereshteh Poorahangaryan, Bahman Zanj and Ali Karami, "A Neural Network Based System for Persian Sign Language Recognition", IEEE International Conference on Signal and Image Processing Applications, 2009.

[7] Khaled Assaleh and M. Al-Rousan, "Recognition of Arabic Sign Language Alphabet Using Polynomial Classifiers", EURASIP Journal on Applied Signal Processing 2005:13, 2136-2145.

[8] Wenmiao Lu and Shaohua Sun, "Face Detection in Color Images", http://www.osun.org.

[9] K.K. Bhoyar and O.G. Kakde, "Skin Color Detection Model Using Neural Networks and its Performance Evaluation", Journal of Computer Science 6 (9): 963-968, 2010.

[10] Wang Zhanjie andTong Li, "A Face Detection System Based Skin Color and Neural Network", International Conference on Computer Science and Software Engineering, 2008.

[11] Lamiaa Mostafa and Sherif Abdelazeem, "Face Detection Based on Skin Color Using Neural Networks", GVIP 05 Conference, 19-21 December 2005, CICC, Cairo, Egypt, 2005

[12] Reza Hassanpour, Asadollah Shahbahrami, and Stephan Wong, "Adaptive Gaussian Mixture Model for Skin Color Segmentation", Proceedings of world academy of science, engineering and technology volume, 31 July 2008.

[13] Vladimir Vezhnevets, Vassili Sazonov and Alla Andreeva, "A Survey on Pixel-Based Skin Color Detection Techniques", http://www.osun.org.

[14] Andreas Carlsson, Andreas Eriksson and Mikael Isik, "image detection using artificial neural networks and statistical methods, IT University of Goteborg, Sweden 2008

[15] Nariman Habili, Cheng Chew Lim and Alireza Moini, "Segmentation of the Face and Hands in Sign Language Video Sequences Using Color and Motion Cues", IEEE Transactions on circuits and systems for video technology, vol. 14, no. 8, August 2004.

[16] Chelsia Amy Doukim, Jamal Ahmad Dargham, Ali Chekima and Sigeru Omatu, "Combining Neural Networks for Skin Detection", Signal & Image Processing: An International Journal (SIPIJ) Vol.1, No.2, December 2010.

[17] Aamer .S.S.Mohamed, Ying Weng, Stan S Ipson and Jianmin Jiang, "Face Detection based on Skin Color in Image by Neural Networks", http://www.osun.org.

[18] Wenjun Tan, Chengdong Wu, Shuying Zhao and Shuo Chen, "Hand Extraction Using Geometric Moments Based on Active Skin Color Model", 978-1-4244-4738, IEEE, 2009.

[19] Jiann-Shu Lee,Yung-Ming Kuo, Pau-Choo Chung and E-Liang Chen, "Naked image detection based on adaptive and extensible skin color model", Pattern Recognition Society. Published by Elsevier Ltd, 2006.

[20] Jure KovaE, Peter Peer, and Franc Solina, "Human Skin Colour Clustering for Face Detection", IEEE, 0-7803-7763, 2003.

[21] Ihab Zaqout, Roziati Zainuddin and Sapian Baba, "Pixel-Based Skin Color Detection Technique", http://www.osun.org.

[22] Professor Lakhmi Jain and Professor Xindong Wu, "Advanced Information and Knowledge Processing Series: Machine Learning for Audio, Image and Video Analysis", Springer-Verlag London Limited, 2008.

[23] P. Kakumanu, S. Makrogiannis and N. Bourbakis, "A survey of skin-color modeling and detection methods", Elsevier Ltd on behalf of Pattern Recognition Society, 0031-3203, 2006.

[24] Benjamin D. Zarit, Boaz J. Super and Francis K. H. Quek, "Comparison of Five Color Models in Skin Pixel Classification", http://www.osun.org.

[25] Ming-Jung Seow, Deepthi Valaparla, and Vijayan K. Asari, "Neural Network Based Skin Color Model for Face Detection", Proceedings of the 32nd Applied Imagery Pattern Recognition Workshop (AIPR'03) IEEE, 0-7695-2029, 2003.

[26] Tiberio S. Caetano, Silvia D. Olabarriaga and Dante A. C. Barone, "Performance Evaluation of Single and Multiple-Gaussian Models for Skin Color Modeling", Proceedings of the XV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'02), IEEE,1530-183, 2002.

[27] Nariman Habili, Cheng Chew Lim and Alireza Moini, "Segmentation of the Face and Hands in Sign Language Video Sequences Using Color and Motion Cues", IEEE

TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 14, NO. 8, AUGUST 2004.

[28] Wei Zeng, Guibin Zhu, and Yao Li, "Point Matching Estimation for Moving Object Tracking Based on Kalman Filter", Eigth IEEE/ACIS International Conference on Computer and Information Science, 2009.

[29] Karthik Hariharakrishnan and Dan Schonfeld, "Fast Object Tracking Using Adaptive Block Matching", IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 7, NO. 5, OCTOBER 2005.

[30] Kap-Ho Seo, Jin-Ho Shin, Won Kim, and Ju-Jang Lee, "Real-Time Object Tracking and Segmentation Using Adaptive Color Snake Model", International Journal of Control, Automation, and Systems, vol. 4, no. 2, pp. 236-246, April 2006.

[31] Divya Mandloi, Mani Kanthi Sarella, and Chance M. Glenn, "Implementation of Image Processing Approach to Translation of ASL Fingerspelling to Digital Text", http://www.osun.org.

[32] Wilhelm Burger and Mark James Burge, "Digital Image Processing", First edition Springer Science+Business Media, LLC, 2008.

[33] Chris Solomon and Toby Breckon, "Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab", First edition by John Wiley & Sons, Ltd, New York, 2011.

[34] Wilhelm K. Pratt, "Digital Image Processing", 3rd ed. John Wiley & Sons, Inc, New York, 2001.

[35] Michael Seul, Lawrence O'Gorman and Michael J. Sammon, "Practical Algorithms for Image Analysis", Cambridge University Press 2000.

[36] Ben Krose Patrick van der Smagt, "An introduction to Neural Networks", Eighth edition, The University of Amsterdam, November 1996

[37] Simona E. Grigorescu, Nicolai Petkov, and Peter Kruizinga, "Comparison of Texture Features Based on Gabor Filters", IEEE transactions on image processing, vol. 11, NO. 10, october 2002.

[38] D. I. Kosmopoulos, A. Doulamis and N. Doulamis, "Gesture-based Video Summarization",

[39] Ryszard S. Chora´s, "Image Feature Extraction Techniques and Their Applications for CBIR and Biometrics Systems", International journal of biology and biomedical engineering, Issue 1, Vol. 1, 2007.

[40] M. Ashraful Amin and Hong Yan, "Sign Language Finger Alphabet Recognition From Gabor-CPA Representations of Hand Gestures", Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 19-22 August 2007.

[41] Chen-Chiung Hsieh, Dung-Hua Liou and David Lee, "A Real Time Hand Gesture Recognition System Using Motion History Image", *2nd International Conference on Signal Processing Systems (ICSPS), 2010.*

[42] Nguyen Dang Binh, Enokida Shuichi and Toshiaki Ejima, "Real-Time Hand Tracking and Gesture Recognition System", GVIP 05 Conference, CICC, Cairo, Egypt, 19-21 December 2005.

[43] Elena Sánchez-Nielsen, Luis Antón-Canalís and Mario Hernández-Tejera, "Hand Gesture Recognition for Human-Machine Interaction", Journal of WSCG, Vol.12, No.1-3, *WSCG'2004, Plzen, Czech Republic, February 2-6, 2003.*

[44] Mohd Shahrimie Mohd Asaari and Shahrel Azmin Suandi, "Hand Gesture Tracking System Using Adaptive Kalman Filter", IEEE 978-1-4244-8136, 2010.

[45] A. Prem Kumar, T. N. Rickesh, R. Venkatesh Babu and R. Hariharan, "Object tracking using Radial basis function networks",

[46] Tianming Liu, Hong-Jiang Zhang, and Feihu Qi, "A Novel Video Key-Frame-Extraction Algorithm Based on Perceived Motion Energy Model", IEEE transactions on circuits and systems for video technology, vol. 13, no. 10, October 2003.

[47] Chia-Shiuan Cheng, Pi-Fuei Hsieh, and Chung-Hsien Wu, "Hand Motion Recognition for the Vision-based Taiwanese Sign Language Interpretation",

[48] Richard O. Duda, Peter E. Hart and David G. Stork, "Pattern Classification",

[49] Nhat Thanh Nguyen and The Duy Bui, "Automated Posture Segmentation in Continuous Finger Spelling Recognition", IEEE, 978-1-4244-7570, 2010.

[50] Kin-Wai Sze, Kin-Man Lam, and Guoping Qiu, "A New Key Frame Representation for Video Segment Retrieval", IEEE transactions on circuits and systems for video technology, vol. 15, No. 9, September, 2005.

[51] Yongliang Xiao and Limin Xia, "Key Frame Extraction Based on Connectivity Clustering", Second International Workshop on Education Technology and Computer Science, 2010.

[52] Simple Skin Segmentation, http://www.mathworks.com/matlabcentral/fileexchange/24934-simple-skin-segmentation/all_files

[53] M. P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching", 12th International Conference on Pattern Recognition, 566-568, Jerusalem, 1994

[54] R. J. LÓPEZ-SASTRE, S. LAFUENTE-ARROYO, P. SIEGMANN, P. GIL-JIMÉNEZ, A. VAZQUEZ-REINA, "Recognition of Mandatory Traffic Signs using the Hausdorff distance", Proceedings of the 5th WSEAS Int. Conf. on Signal Processing, Computational Geometry & Artificial Vision, Malta, September 15-17, 2005

[55] Etienne Baudrier, Frederic Nicolier, Gilles Millon, Su Ruan," Binary-image comparison with local-dissimilarity quantification", Elsevier Ltd, 2007.

APPENDIX A: MATLAB code

A.1. Skin-color segmenting code

The following function takes an RGB image as input and returns binary and grayscale segmented images

```
1 function [BI RGB]=newSkinSeg(im)
2 d=ones(5,5);
3 av=[0.001 0.125 0.001;0.125 0.15 0.125;0.001 0.125 0.001];
4 im=imfilter(im,av);
5 img=rgb2ycbcr(im);
6
7 BI=zeros(size(img,1),size(img,2));
8 for i=1:size(img,1)
9    for j= 1:size(img,2)
10        cb = img(i,j,2);
11        cr = img(i,j,3);
12        if(~(cr > 132 && cr < 173 && cb > 76 && cb < 126))
13             BI(i,j)=0;
14        else
15             BI(i,j)=1;
16        end
17   end
18 end
19 BI=imopen(BI,d);
20 BI=imclose(BI,d);
21 for i=1:size(img,1)
22   for j= 1:size(img,2)
23        if(BI(i,j)==1)
24             im(i,j,1)=im(i,j,1);
25             im(i,j,2)=im(i,j,2);
```

```
26              im(i,j,3)=im(i,j,3);
27          else
28              im(i,j,1)=0;
29              im(i,j,2)=0;
30              im(i,j,3)=0;
31          end
32     end
33 end
35 RGB=im;
```

A.2. Hand isolation code

```
1 % This function is used to isolate the hand both from binary and grayscale
2 % images of the same input image
3 function [segImage grayImage]=isolateHand(segImage,grayImage)
4 L=bwlabel(segImage);
5 [BB NO]=bwlabeln(L);%NO represents number of objects in segmented image
6 prop=regionprops(L,'Area');
7 for ar=1:NO
8     blobArea(ar,1)=prop(ar).Area;
9 end
10 AREA=blobArea;
11 blobArea=sort(blobArea,'descend');
12 if(size(blobArea,1)>=2)
13     for i=1:NO
14         if(AREA(i)==blobArea(2))
15             ind=i;
16             break;
17         end
18     end
```

```
19   for aa=1:size(L,1)
20        for bb=1:size(L,2)
21             if(L(aa,bb)~=ind)
22                  segImage(aa,bb)=0;
23             end
24        end
25   end
26   clear AREA;clear area;
27
28 %Hand isolation from the grayscale image based on the binary image
29   for cc=1:size(segImage,1)
30        for dd=1:size(segImage,2)
31             if(segImage(cc,dd)==0)
32                  grayImage(cc,dd)=0;
33             else
34                  grayImage(cc,dd)=grayImage(cc,dd);
35             end
36        end
37   end
38 end
```

A.3. Centroid finder

```
function [x y]=trackHand(I)
L=bwlabel(I);
props= regionprops(L,'Centroid');
centroid = props(1).Centroid;
x=round(centroid(1));
y=round(centroid(2));
```

A.4. Hand cropping

```
1 function [binaryC x1 y1]=cropHand(binary)
2 x1=size(binary,2);
3 y1=0;
4 y2=0;
5 x2=zeros(50,1);
6 flag1=0;
7 flag2=0;
8 n=0;
9 for u=1:size(binary,1)
10   for v=1:size(binary,2)
11        if(binary(u,v)==1)
12             flag2=1;
13             flag1=flag1+1;
14             if(flag1<=1)
15                  y1=u;
16             else
17                  y2=u;
18             end
19                  if(v<x1)
20                       x1=v;
21                  end
22        else
23             if(flag2==1)
24                  n=n+1;
25                  x2(n,1)= v-1;
26             end
27             flag2=0;
28        end
```

```
29    end
30 end
31 x2=max(x2);
32 h=y2-y1;
33 w=x2-x1;
34 binaryC=imcrop(binary,[x1 y1 w h]);
```

A.5. Block division

In the following function, SPEED is a vector of gesture speed and ***thresh*** is the threshold used to divide the image sequence into blocks.

```
1 function B=block(SPEED,thresh)
2 %this block of code removes single and random speed peaks
3 for b=2:length(SPEED)-1
4    if(SPEED(b)>thresh)
5         if(SPEED(b-1)<thresh && SPEED(b+1)<thresh)
6              SPEED(b)=0;
7         end
8    end
9 end
10 SPEED(b)=0;n=1;
12 for i=1:length(SPEED)
13   if(SPEED(i)<thresh)
14        A(n)=i;
15        n=n+1;
16   end
17 end
18 m=2;c=0;
19 for k=1:length(A)-1
20   if(A(k+1)-A(k)>=2)
```

```
21          B(m)=A(k+1);
22          B(m-1)=A(k)-round(c/2);
23          m=m+1;
24          c=0;
25          end
26     c=c+1;
27 end
```

A.6. HMTD

In the following function, parameter HT is a matrix of centroid.

```
1 function TN=matchTR(HT)
2 cntv=0;
3 cnth=0;
4 psign=0;
5 nsign=0;
6 SC=0;
7 B=calculateDegree(HT);
8 count1=0;
9 for i=1:length(B)
10    count1=count1+1;
11    if(abs(B(i))>=60)
12          cntv=cntv+1;
13    elseif(abs(B(i))<=45)
14          cnth=cnth+1;
15    end
16 end
17 count2=0;
18 for k=2:length(HT)-1
19    count2=count2+1;
20    if((HT(k,2)-HT(k-1,2))>0)
```

```
21        psign=psign+1;
22   elseif((HT(k,2)-HT(k-1,2))<0)
23        nsign=nsign+1;
24 end
25
26   a1=HT(k,1)-HT(k-1,1);
27   a2=HT(k+1,1)-HT(k,1);
28   if(a1*a2<=0)
29        SC=SC+1;
30   end
31 end
33 dv=cntv/count1;
34 dh=cnth/count1;
35 psign=psign/count2;
36 nsign=nsign/count2;
37
38 if(dv>=0.75) %Trajectory has vertical nature
39   TN=4;
40 elseif(dh>=0.75) %Trajectory has horizontal nature
41   dd=0;
42   for h=2:length(HT)
43        dis=HT(h,1)-HT(h-1,1);
44        dd=dd+dis; %summing up for dominant direction
45        if(dd<0)
46             TN=3;
47        else
48             TN=2;
49        end
50   end
51 elseif(nsign>=0.30)
```

```
52    TN=5;
53 elseif(psign>=0.70 && SC>=2)
54    TN=6;
55 else
56    TN=100;    %Invalid
57 end
58 clear HT;
```

This function is used to calculate the angle of a line drawn between two consecutive centroids and collects all such angles within an EMA block.

```
1 function A=calculateDegree(HT)
2 for i=2:length(HT)-1
3     x=HT(i,1)-HT(i-1,1);
4     y=HT(i,2)-HT(i-1,2);
5     theta=atan(y/x);
6     Deg = theta * (180/pi);
7     A(i-1,1)=Deg;
8 end
```

A.7. Main program

```
1 % Addis Ababa Institute of Technology
2 %Department of Electrical and Computer Engineering
3 %Computer Engineering stream
4 %DESIGN AND IMPLEMENTATION OF CANDIDATE HAND GESTURE
5 %SELECTION AND HAND MOVEMENT TRAJECTORY DETERMINATION
6 %FOR CONTINUOUS ETHIOPIAN MANUAL ALPHABETS
7
8
9 function root
10 th=0.155;
11 dn='D:\Ethiopian sign language\MATLAB\Final\videos';
```

```
% subdirectories where extracted images reside
subdf1='\Fitsum\train\';
subdf2='\Fitsum\test\';
subdg1='\Getahun\train\';
subdg2='\Getahun\test\';
subdk1='\Kidane\train\';
subdk2='\Kidane\test\';
subdm1='\Muluneh\train\';
subdm2='\Muluneh\test\';
subdh1='\Henok\train\';
subdh2='\Henok\test\';
H=96;
W=64;
%Putiing the names of the training folders into a single array of (files)
d=strcat(dn,subdh2);
files=dir(d);
files(1:2)=[];
v=0;
for i=17:17
  foldern=files(i).name;
  fd=strcat(d,foldern);
  imgs=dir(fullfile(fd,'*.jpg'));
  for k=1:length(imgs)
       fn=strcat(fd,'\',imgs(k).name);
       im=imread(fn);
       [B RGB]=newSkinSeg(im);
       [binaryImage rgbImage]=isolateHand(B,RGB);
       [x y]=trackHand(binaryImage);
       CS(k,1)=x;
       CS(k,2)=y;
```

```
48          [binaryC x1 y1]=cropHand(binaryImage);
49          bin=imresize(binaryC,[H W]);
50          rgb=imcrop(rgbImage,[x1 y1 size(binaryC,2)
            size(binaryC,1)]);
51          RGB=imresize(rgb,[H W]);
52          bin=bwmorph(bin,'remove');
53          RGB_IMAGES(:,:,:,k)=RGB(:,:,:);
54          BINARY_IMAGES(:,:,k)=bin(:,:);
59    end
61  % Speed calculation
62  for s=3:2:length(imgs)
63    t=((s+1)/2);
64    distance=sqrt((CS(s,1)-CS(s-2,1)).^2+(CS(s,2)-CS(s-
      2,2)).^2);
65    spd(t,1)=distance/66.67;
71  end
73  spd(t,1)=0;
74  spd(1,1)=0;
76  % block division
77  try
78    D=block(spd,th);
79  catch ME
80    fprintf('Error');
81  end
83  for f=1:length(D)
84    E(f)=D(f)*2;
85  end
87  %Copy images in each block
88  ST=0;EN=0;
89  tt=1;
```

```
90 for g=1:length(E)-1
91   if(rem(g,2)~=0)
92   cnt=1;
93   ST=E(g);
94   EN=E(g+1);
95   for j=ST:EN
96        HT(j-ST+1,1)=CS(j,1);
97        HT(j-ST+1,2)=CS(j,2);
98   end
100  mx=min(HT(:,1));
101  my=min(HT(:,2));
103  for mm=1:length(HT)
104       NHT(mm,1)=HT(mm,1)-mx+1;
105       NHT(mm,2)=HT(mm,2)-my+1;
106  end
107 %DEG=calculateDegree(NHT);
108 a=zeros(100,100);
109  for k=1:length(NHT)
110       a(NHT(k,2),NHT(k,1))=1;
111  end
114 TR=matchTR(NHT);
116 clear NHT;
117 clear HT;
119 %Find search window where cnt-1 is the size of the block
120 s=EN-ST;
122 if(s<6)
123       SW=1;
124       EW=s;
125 else
126       SW=round(1/10*(s)+0.5);
```

```
127       EW=round(2/3*(s));
128 end
131 gg=1;
132 for q=ST+SW-1:SW+EW
134 IMG(:,:,cnt)=BINARY_IMAGES(:,:,q);
135 cnt=cnt+1;
138 end
140 %Compare using hausdorff and select one within a search window
141 for nn=1:size(IMG,3)-1
142       DD(nn)=ModHausdorffDist(IMG(:,:,nn),IMG(:,:,nn+1));
143 end
145 ind=find(min(DD(:)));
146 selected=ST+ind+SW;
148 figure( 1 );
149 aa=mat2str(selected);
150 bb=mat2str(TR);
151 cap=strcat('G - ',aa,' - (T - ',bb,')');
152 subplot( 2, 2, v * 2 + tt),imshow
    (rgb2gray(RGB_IMAGES(:,:,:,selected))),title(cap);
153 tt=tt+1;
154 clear DD;
155 clear IMG;
156 end
157 end
158 v=v+1;
159 clear TR;
160 clear selected;
161 clear E;
162 clear CS;
164 clear spd;
```

```
165 clear imgs;
166 end
```

A.8. Modified Hausdorff Distance

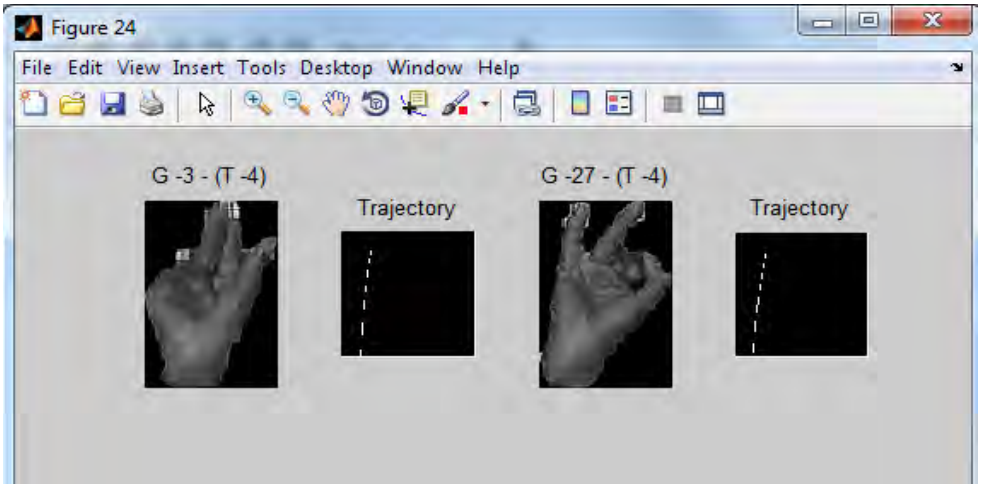
```
function [ mhd ] = ModHausdorffDist( A, B )
% Code Written by B S SasiKanth, Indian Institute of Technology Guwahati.
% Website: www.bsasikanth.com
% E-Mail:  bsasikanth@gmail.com
% Compute the sizes of the input point sets
Asize = size(A);
Bsize = size(B);
% Check if the points have the same dimensions
if Asize(2) ~= Bsize(2)
    error('The dimensions of points in the two sets are not
equal');
end
% Calculating the forward HD
fhd = 0;                    % Initialize forward distance to 0
for a = 1:Asize(1)          % Travel the set A to find avg of d(A,B)
    mindist = Inf;          % Initialize minimum distance to Inf
    for b = 1:Bsize(1)      % Travel set B to find the min(d(a,B))
        tempdist = norm(A(a,:)-B(b,:));
        if tempdist < mindist
            mindist = tempdist;
        end
    end
    fhd = fhd + mindist;    % Sum the forward distances
end
fhd = fhd/Asize(1);         % Divide by the total no to get average
% Calculating the reverse HD
```

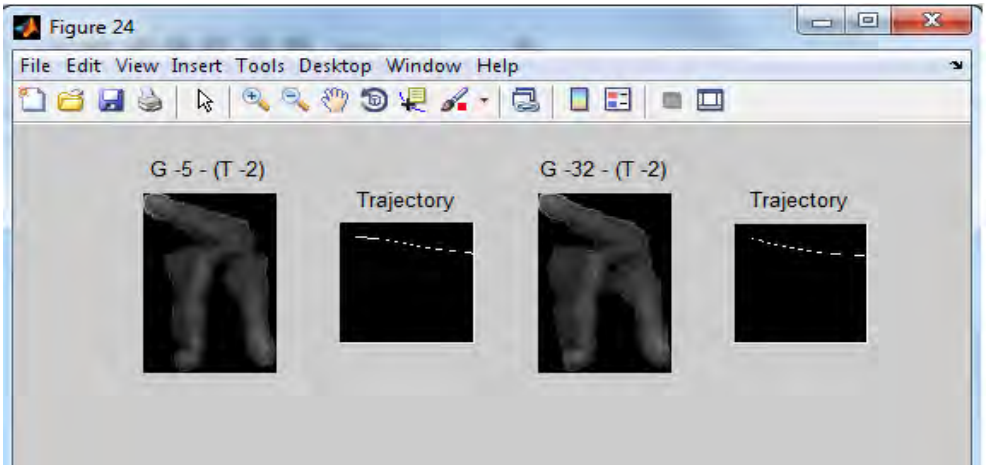
```
rhd = 0;                        % Initialize reverse distance to 0
for b = 1:Bsize(1)              % Travel the set B to find avg of d(B,A)
    mindist = Inf;              % Initialize minimum distance to Inf
    for a = 1:Asize(1)          % Travel set A to find the min(d(b,A))
        tempdist = norm(A(a,:)-B(b,:));
        if tempdist < mindist
            mindist = tempdist;
        end
    end
    rhd = rhd + mindist;        % Sum the reverse distances
end
rhd = rhd/Bsize(1);             % Divide by the total no. to get average
mhd = max(fhd,rhd);             % Find the minimum of fhd/rhd as
                                % the mod hausdorff dist
end
```

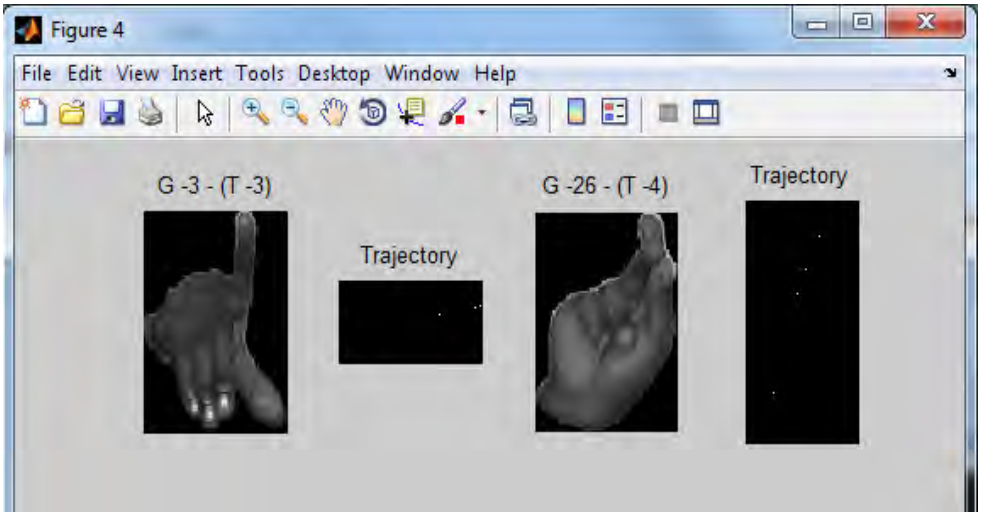
APPENDIX B: SAMPLE DATA USED IN THE SYSTEM DESIGN


0001 huda1 00019
0001 huda1 00020
0001 huda1 00021
0001 huda1 00022
0001 huda1 00023
0001 huda1 00024
0001 huda1 00028
0001 huda1 00029

APPENDIX C: SAMPLE RESULTS OF THE PROPOSED DESIGN







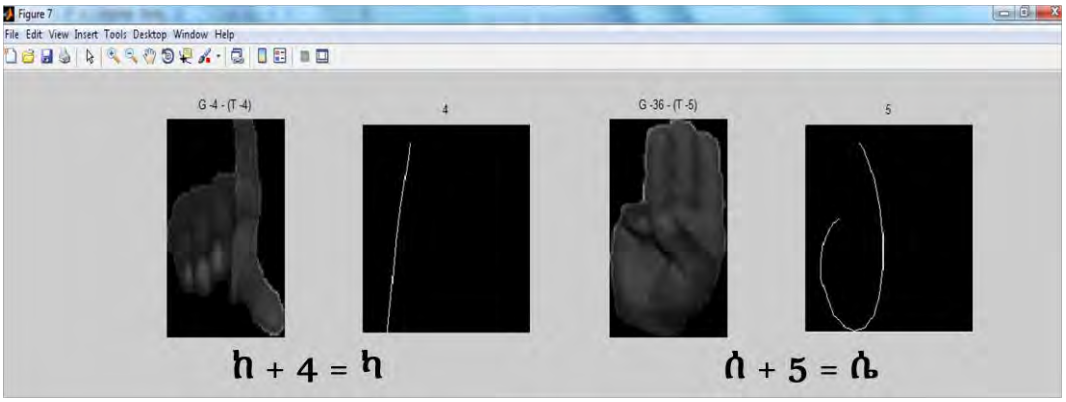


Figure 7
File Edit View Insert Tools Desktop Window Help
G -4 - (T -4)
4
G -36 - (T -5)
5
ከ + 4 = ካ
ሰ + 5 = ሴ

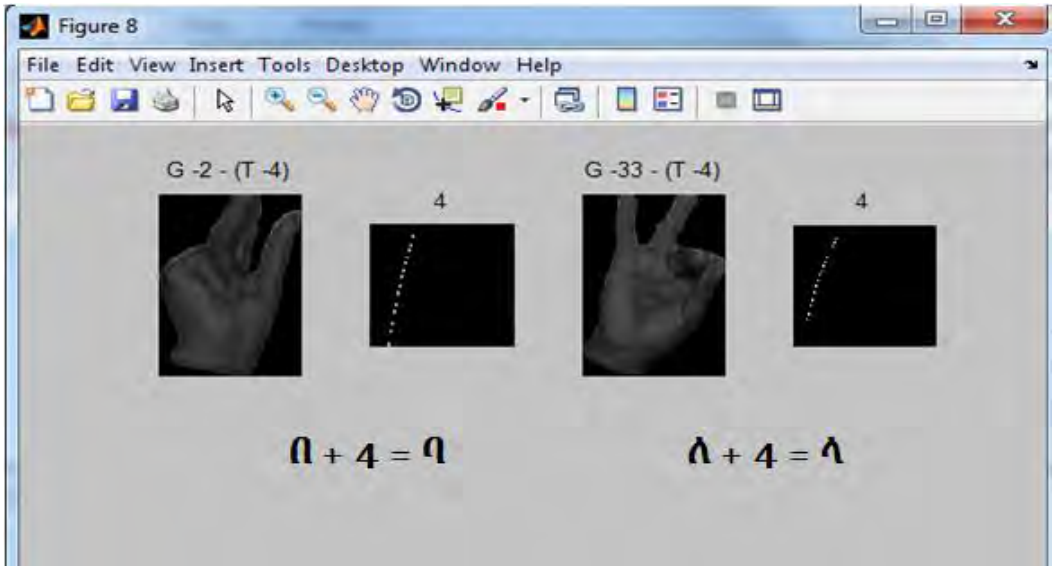


Figure 8
File Edit View Insert Tools Desktop Window Help
G -2 - (T -4)
4
G -33 - (T -4)
4
ቦ + 4 = ባ
ለ + 4 = ላ

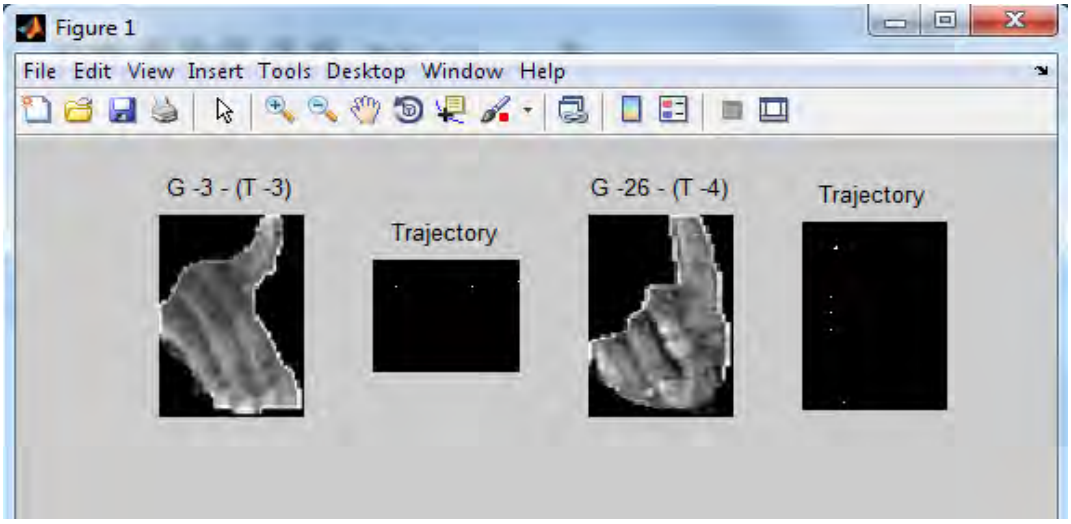


Figure 1
File Edit View Insert Tools Desktop Window Help
G -3 - (T -3)
Trajectory
G -26 - (T -4)
Trajectory

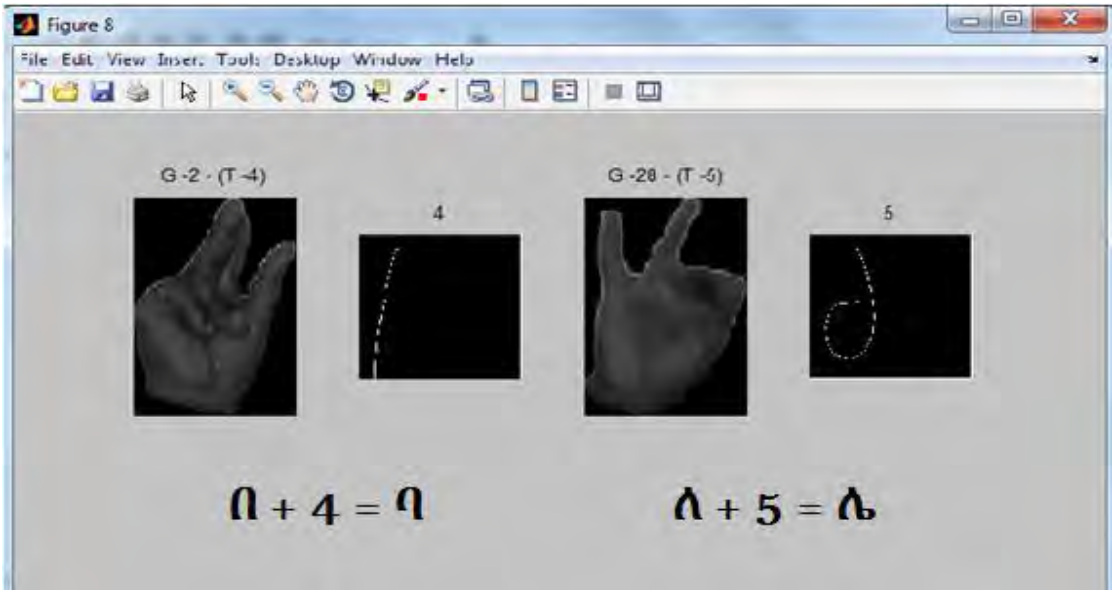


Figure 8
File Edit View Insert Tools Desktop Window Help
G -2 - (T -4)
4
G -28 - (T -5)
5
ሀ + 4 = ሃ
ለ + 5 = ሌ

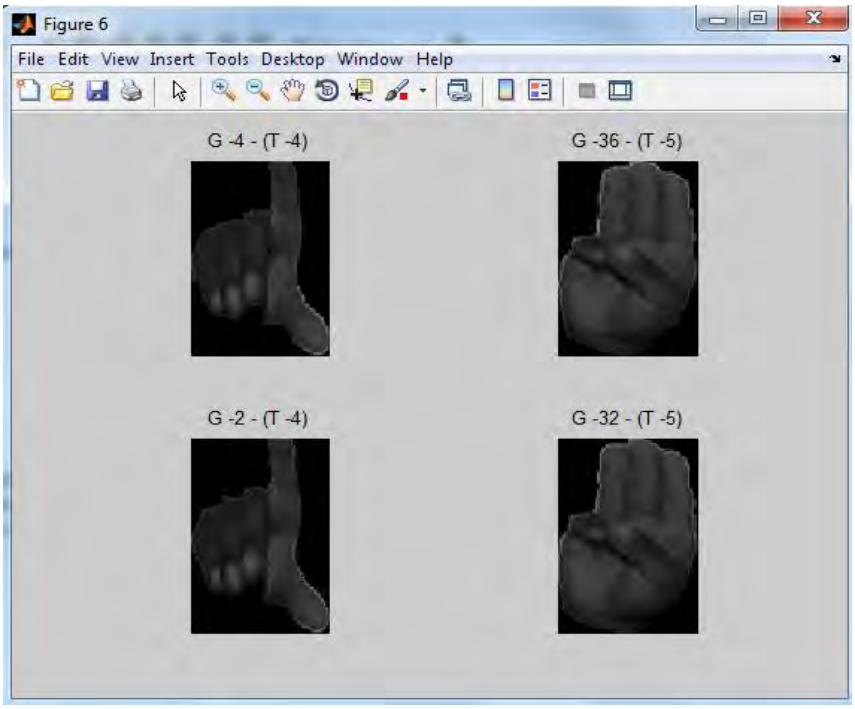


Figure 6
File Edit View Insert Tools Desktop Window Help
G -4 - (T -4)
G -36 - (T -5)
G -2 - (T -4)
G -32 - (T -5)

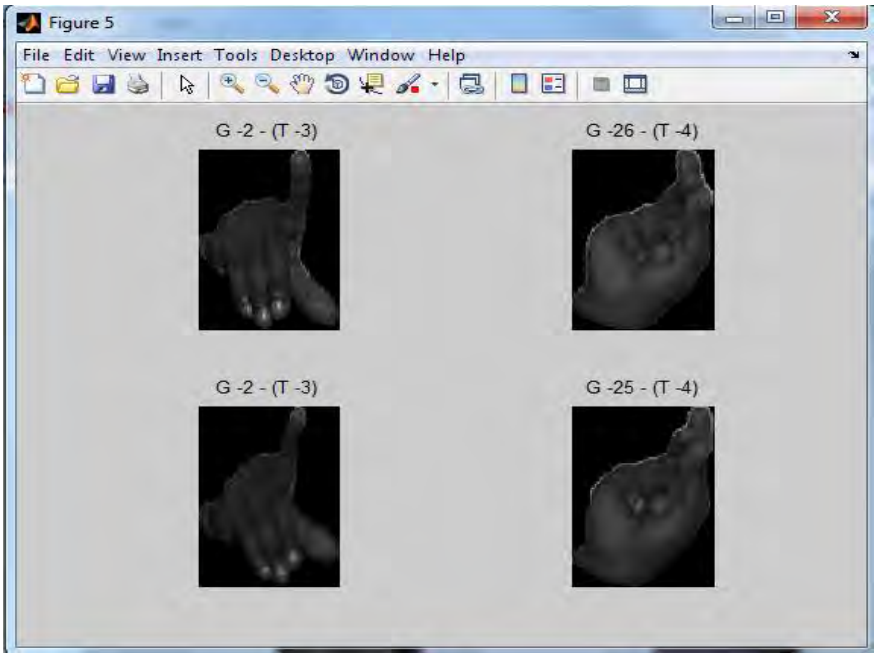


Figure 5
File Edit View Insert Tools Desktop Window Help
G -2 - (T -3)
G -26 - (T -4)
G -2 - (T -3)
G -25 - (T -4)

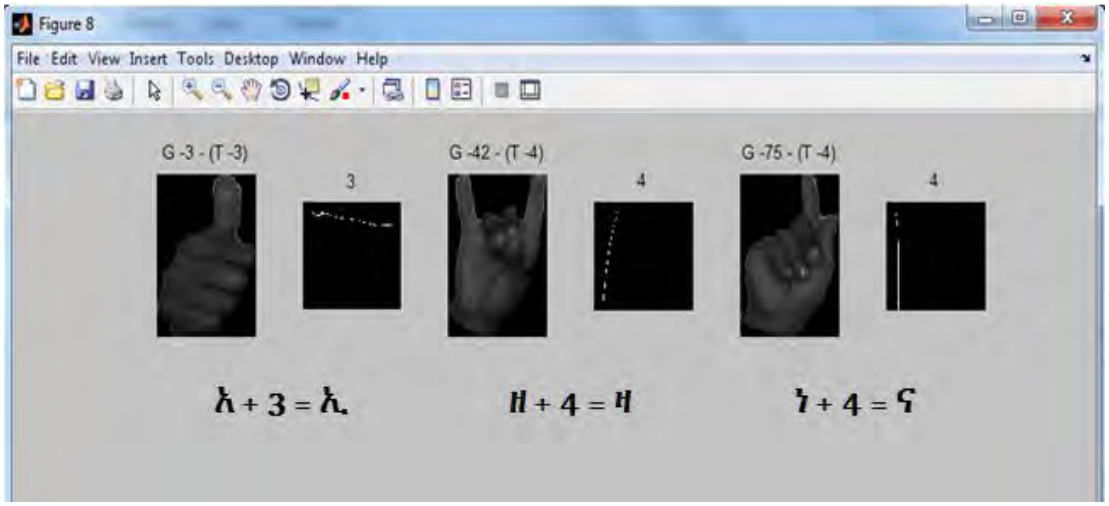


Figure 8
File Edit View Insert Tools Desktop Window Help
G -3 - (T -3)
3
G -42 - (T -4)
4
G -75 - (T -4)
4
አ + 3 = ኢ
ዘ + 4 = ዛ
ነ + 4 = ና

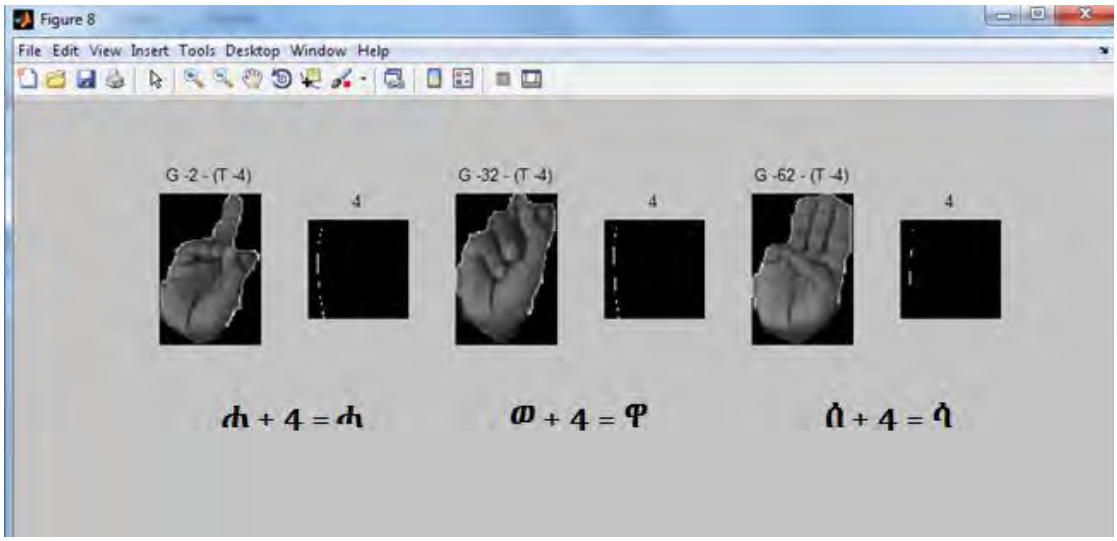


Figure 8
File Edit View Insert Tools Desktop Window Help
G -2 - (T -4)
4
G -32 - (T -4)
4
G -62 - (T -4)
4
ሐ + 4 = ሐ
ወ + 4 = ዎ
ሰ + 4 = ሳ

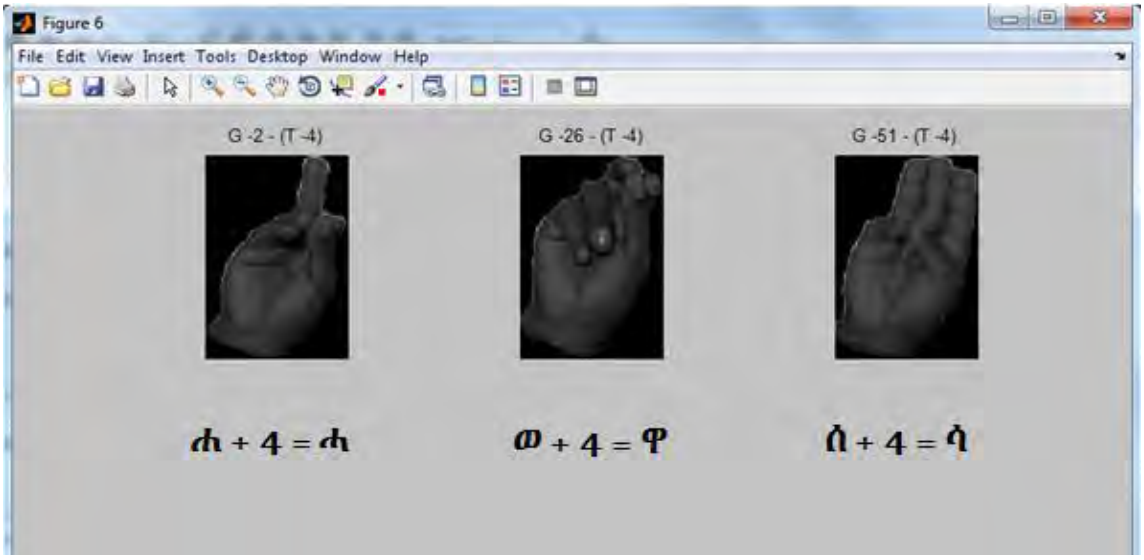


Figure 6
File Edit View Insert Tools Desktop Window Help
G -2 - (T -4)
G -26 - (T -4)
G -51 - (T -4)
ሐ + 4 = ሐ
ወ + 4 = ዎ
ሰ + 4 = ሳ

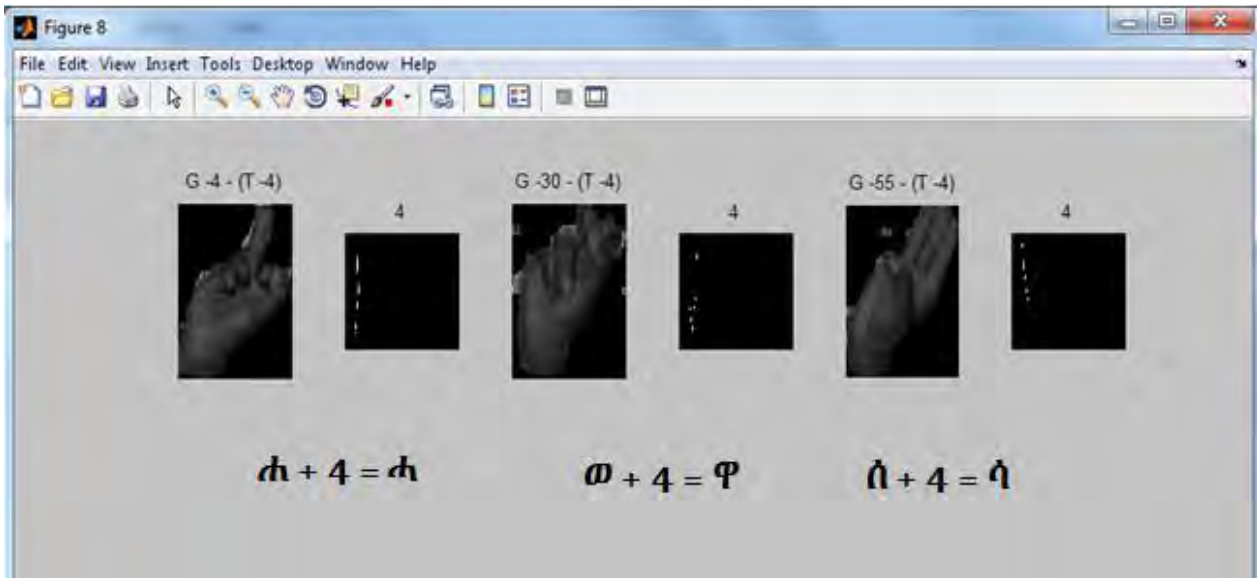


Figure 8
File Edit View Insert Tools Desktop Window Help
G -4 - (T -4)
4
G -30 - (T -4)
4
G -55 - (T -4)
4
ሐ + 4 = ሐ
ወ + 4 = ዋ
ሰ + 4 = ሳ

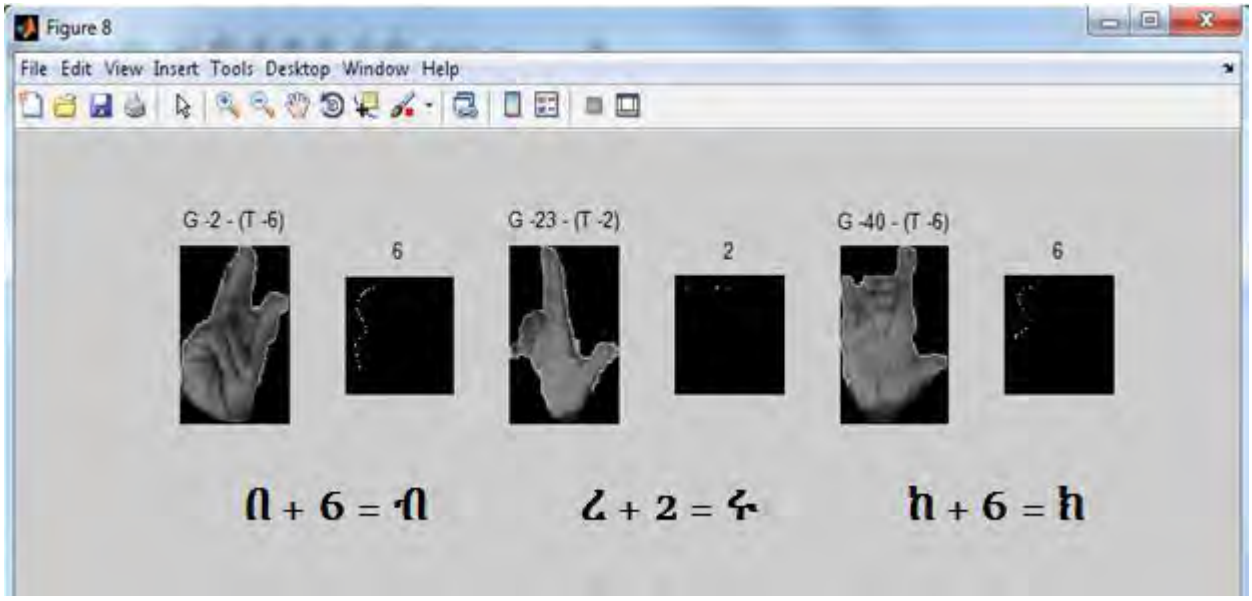


Figure 8
File Edit View Insert Tools Desktop Window Help
G -2 - (T -6)
6
G -23 - (T -2)
2
G -40 - (T -6)
6
በ + 6 = ባ
ረ + 2 = ሩ
ከ + 6 = ኸ

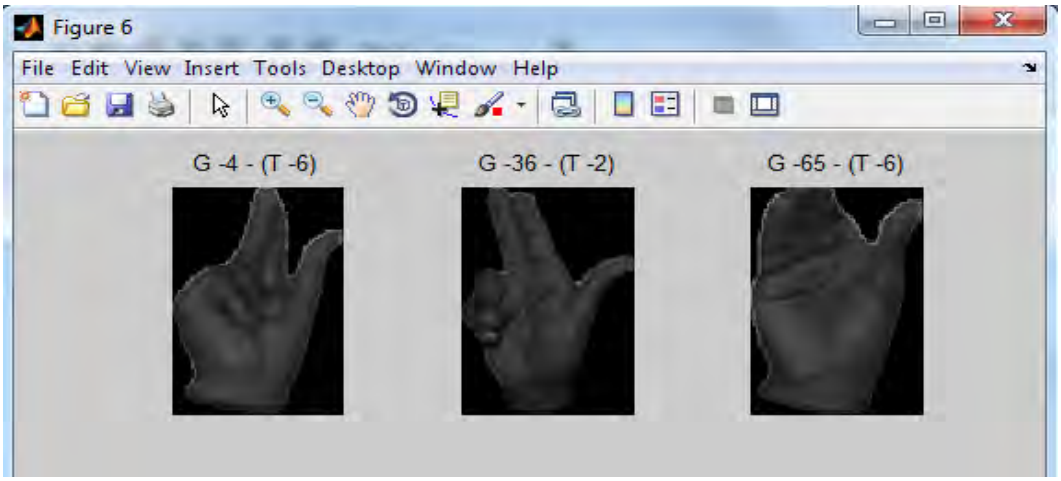
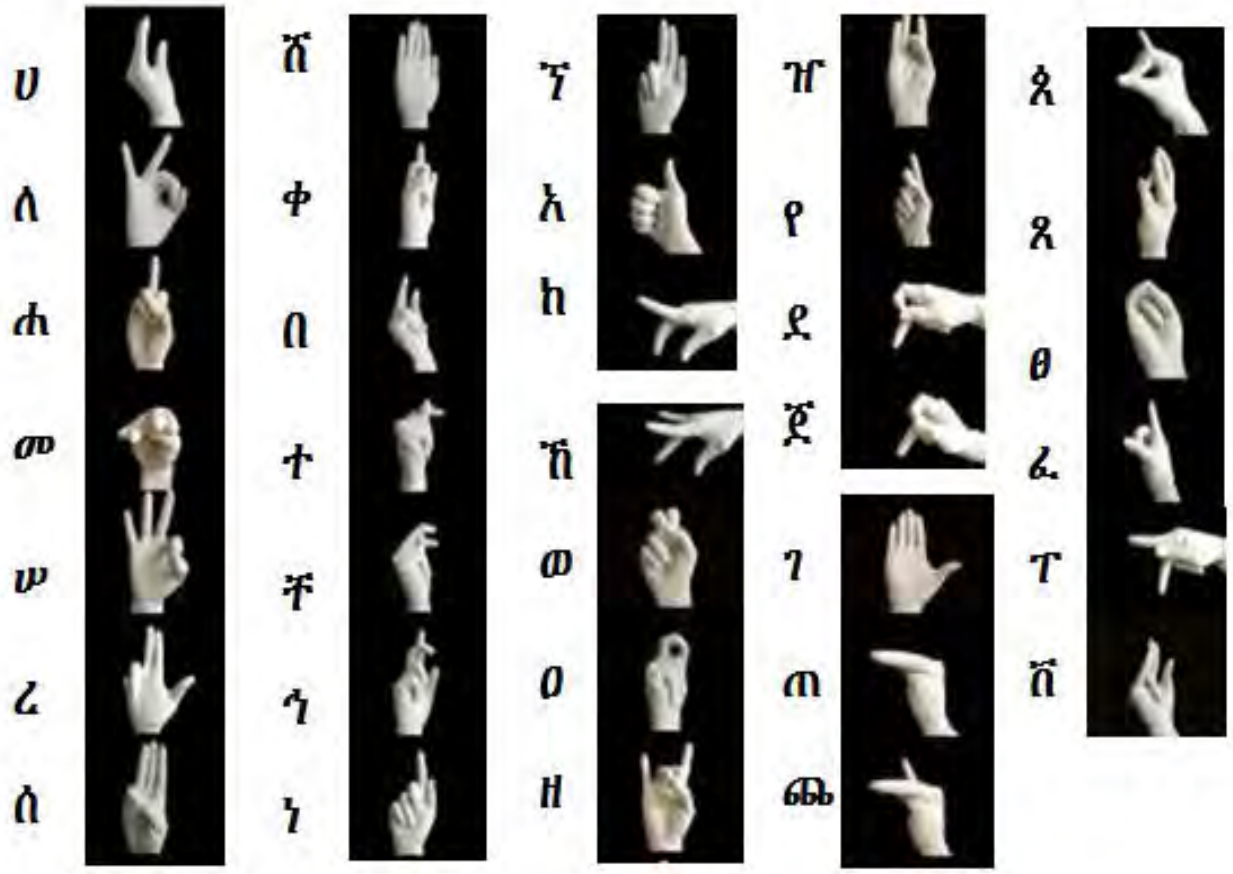


Figure 6
File Edit View Insert Tools Desktop Window Help
G -4 - (T -6)
G -36 - (T -2)
G -65 - (T -6)

APPENDIX D

THE 34 BASE EMAs TAKEN FROM [5]



DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been fully acknowledged.

Name: Abadi Tsegay Weldegebriel Signature: _____

Place: Addis Ababa

Date of submission: October 2011

This thesis has been submitted for examination with my approval as a university advisor.

Dr. Kumudha Raimond Signature: _____

Advisor's Name