ADDIS ABABA UNIVERSITY

College of Natural Sciences

Translation of Continuous Signs in Ethiopian Sign Language to Amharic Text

Abdulhafiz Jeylan Haji

A Thesis Submitted to the Department of Computer Science in Partial Fulfilment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

November 2017

Addis Ababa University

College of Natural Sciences

Translation of Continuous Signs in Ethiopian Sign Language to Amharic Text

Abdulhafiz Jeylan Haji

Advisor: Yaregal Assabie (PhD)

This is to certify that the thesis prepared by Abdulhafiz Jeylan, titled: *Translation of Continuous Signs in Ethiopian Sign Language to Amharic Text* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

	Name	Signature	Date
Advisor:	_____	_____	_____
Examiner:	_____	_____	_____
Examiner:	_____	_____	_____

Abstract

A Sign Language is a visual language that uses a system of manual, facial and body movements as the means of communication. Sign language is not an universal language, and different sign languages are used in different countries, like the many spoken languages all over the world. Sign languages that exist around the world are usually identified by the country where they are used such as Ethiopian Sign Language (EthSL). Sign language is the basic alternative communication method between hearing impaired people and several dictionaries of words have been defined to make this communication possible. Even if it is widely used in the hearing impaired community, they struggle to communicate with hearing people due to the language barrier. Due to this communication gap hearing impaired people encounter so many problems in their daily life since they are living with the people who communicate with spoken languages. Unfortunately, few people have knowledge of sign language in our daily life. In general, interpreters can help us to communicate with these challengers, but they only can be found in Government Agencies. Moreover, it is expensive to employ interpreter on personal behalf and inconvenient when privacy is required. Consequently, it is very important to develop a system which fills the communication gap between the hearing impaired and hearing people. Many researches are conducted on the recognition of EthSL but mostly they are limited to recognition of isolated words and highly affected by lighting and complex background. The goal of this study is to recognize Continuous Signs in EthSL.

In this research, we use Three Dimensional (3D) depth information from hand motions and body joints generated from Microsoft Kinect. After extracting manual and non-manual sign language features, the system stores them in a gesture dictionary. A Random Forest algorithm is used to match gestures with those stored in the dictionary and later converted to Amharic text. we proposed language model providing simple solution with inverted indexing concept to reorder topic-comment pattern and the subject topic pattern of EthSL sentence to subject-object-verb sentence pattern of spoken Amahric language. We also address challenges such as Movement Epenthesis (ME) and sign segmentation which appears in continuous sign language recognition by analyzing hand movement pauses. The performance of the system was measured in two categories at: word, and sentence level and we got system accuracy of 84.4% at word level and 60% at sentence level.

Keywords: Sign Language, Random Forest, Recognition, Continuous Translation

Dedicated

to

My Sisters

Acknowledgments

First and foremost, I would like to thank the almighty Allah for making me strong and for being with me in all the way. Next, I would like to express my deepest appreciation and thanks to Dr. Yaregal Asabie, for his guidance and help throughout my research. Dr. Yaregal has always been patient and encouraging with great advice. Moreover, this work would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to W/t Meti Lemma and Ato Endalk Abera of Ethiopian National Association of the Deaf. Finally, I would like to express my special gratitude and thanks to my whole family and friends for their moral support and encouragement.

Table of Contents

List of Figures

List of Tables

Acronyms and Abbreviations

AAU	:	Addis Ababa University
ANN	:	Artificial Neural Network
ASL	:	American Sign Language
CA	:	Contour Analysis
DTW	:	Dynamic Time Warping
ENAD	:	Ethiopian National Association for the Deaf
EthSL	:	Ethiopian Sign Language
EV	:	Elementary Vector
HMM	:	Hidden Markov Models
ICF	:	Interco Relation Function
ME	:	Movement Epenthesis
SDK	:	Software Development Kit
OOB	:	Rate Out of Bag
RGB-D	:	Red Green Blue Depth
TVP	:	Time Varying Parameter
VC	:	Vector Contour
3D	:	Three Dimensional

CHAPTER 1: INTRODUCTION

1.1 Background

We, humans have been gifted by nature with the voice capability that allows them to interact and communicate with each other. For this reason, the spoken language becomes one of the main attributes of humanity. Unfortunately, not everybody possesses this capability due to the lack of one sense, i.e., hearing. Sign language is the basic alternative communication method between hearing impaired people and several dictionaries of words or single letters have been defined to make this communication possible. According to the Ethiopian National Association for the Deaf (ENAD) and a report from Central Statistics Authority in 2007, it is estimated that more than 1.5 million hearing impaired people live in Ethiopia [1]. The users of this language characteristically use some shapes and movements of the hands, arms, body, and facial expression to talk. Unfortunately, this kind of communication is often difficult to be understood by hearing people. The different ways on its methods made sign language unfamiliar for most people.

Many people think that sign language is a signed version of spoken language, but this is completely a wrong assumption because sign language by itself is a language that has its own grammar and structure [2].

The following are the basic components of sign language:

- Hand shape
- Hand movement
- Orientation
- Location
- Non manual features like facial gestures.

Sign language is not universal by its nature. The language can be different in countries that speak the same language. For example, the U.S and the U.K, despite having the same written language, have different sign languages, American Sign Language (ASL) and British Sign Language (BSL). Usually, sign languages are identified by the name of the country where

they are used, e.g., ASL for U.S.A and Ethiopian Sign Language (EthSL) for Ethiopia [3]. EthSL originated from ASL with some influence from the Nordic Countries [2, 4]. The language also includes local signs which originated from local hearing impaired schools. The language didn't get the chance to be developed and standardized like spoken languages. Ethiopian finger spelling is developed by ENAD in 1971. ENAD also developed Ethiopian Sign Language dictionary in 2008. In Ethiopian sign language there are distinct finger spelling signs for the 33 'Geez Fidel' and their corresponding letters with vowels [1]. Signing process is conducted in two ways [3]. The first one is defining specific signs for different objects. There will be a predefined sign to mean father, mother, etc. The other one is using finger spelling. Finger spelling is the representation of written alphabets to signs. Finger spelling is another component of sign language for spelling proper nouns, technical terms, acronyms and words from foreign sign language [2, 5]. The use of finger spelling is limited to hearing people or hearing impaired people who have partial hearing ability. Finger spelling is not preferable by those people who are hearing impaired since their date of birth. Most of the time, signers do not use the finger spelling. They use it when there is no clear sign for the specific situation. Therefore, researches focusing on recognition of signs are more advantageous than those researches that are focusing on finger spelling recognition.

The role of sign language recognition systems in the society is to ensure that hearing impaired people have equality of opportunity and full participation in society. A person who can talk and hear properly cannot communicate with hearing impaired person unless he/she is familiar with sign language. However, most hearing people do not have the knowledge of sign language. This situation leads to creation of a huge communication gap between the hearing impaired and hearing people. The problems are expressed as follows:

- The hearing-impaired people often find themselves having a difficulty in asking customer service counters such as in bus stations, shops, airlines, and other public areas. There is communication gap to interact with the hearing people to accomplish their daily tasks. Most of the hearing impaired people have the problem of writing and speaking. This problem is more visible on those hearing impaired people who lost their hearing ability since birth. The inability of writing and speaking widens the communication gap with the hearing people [2, 6].

- Problems that are encountered in the formal education process that includes:
  - Lack of sufficient interpreters for each school in which hearing impaired students are enrolled [2].
  - The knowledge of the interpreter is limited to translate the subject matter [6].
- The number of schools that provide sign language translation services are very limited.
- Most education programs in the electronic media are targeted for hearing people. This is a problem to get necessary and up-to-date information [6].
- The above problems hinder the hearing impaired people from getting equal opportunity in education as well as in participation in community services [2, 6].
- The problems are also visible in health institutions. Since health professionals need accurate input from the patient to conduct examination, there should be some way to mediate the hearing impaired patients and the health professional [6].

These communication gaps can be eased by the help of interpreters. But this solution is not applicable in all situations as:

- There are very limited interpreters [2, 6].
- There are some issues such as court cases that must be kept secret from the interpreters.

1.2 Statement of the Problem

To overcome communication gap between the hearing and hearing impaired people, it is necessary to build a system that is able to translate sign language into written or spoken language and vice versa. Accordingly, many studies are conducted to recognize and translate non-Ethiopian Sign Language [7]. However, these researches are highly language dependant and their effectiveness is not yet tested and proved to recognize EthSL. There are very few researches which are conducted on EthSL. But most of them are conducted on conversion of an Amharic text to equivalent sign in EthSL. For instance, a system that translates a simple Amharic sentence, producing equivalent continuous 3D (Three- Dimensional) animated signs was developed by Daniel Zegeye [8]. But such development should be extended to make a two way communications by developing a system that recognizes and translates EthSL to

written or spoken language. Recently, a research on recognition of isolated signs of EthSL was conducted by Tefera Gimbi [9]. But the study is limited to recognizing the signs in isolated way failing to meet the continuous and real time nature of human communication. In addition, a major limitation of isolated sign recognition is that it requires each gesture to be preceded and followed by non-gesturing intervals, a requirement not satisfied in continuous gesturing as sign language typically appear within a continuous stream of motion. The main challenge in continuous sign language recognition lies in detecting and modeling the extra movement resulting from the transition between the end of a certain sign and the start of the next one. Furthermore, automatically detecting where a gesture starts and ends is a challenging problem. To address such issues, it is important to develop a system that is capable of translating continuous signs in EthSL to Amharic text in real time manner.

1.3 Objectives

1.3.1 General Objective

The general objective of the thesis is to develop a system that translates continuous signs in EthSL to Amharic text.

1.3.2 Specific Objectives

The specific objectives of this research work are to:

- Study the structure of Ethiopian Sign Language.
- Build a database of continuous signs in EthSL.
- Study about gesture recognition and how to define a model of gesture using depth video data.
- Extract hand shapes, orientation and movement of hands, arms or body, and facial expressions from collected data.
- Develop a method to convert the extracted signs to equivalent Amharic text.
- Develop a prototype which demonstrates the translation of the given sign to equivalent Amharic text.
- Test the effectiveness and appropriateness of the system.

1.4 Methods

In order to attain the objectives of the study, the following methods will be used.

1.4.1 Literature Review

In order to get better knowledge and understanding on the area of sign language, different documents such as books, previous research works, journal articles, research reports, manuals, training manuals, and other published and unpublished theses will be reviewed. Researches which are conducted for the recognition of other sign languages such as American Sign Language, Indian Sign Language, Persian Sign Language and Arabic Sign Language will be reviewed in this research work.

1.4.2 Data Collection and Preparation

Every required data which is essential to conduct the study will be collected. Several data collection strategies will be followed to acquire the required data. Experts and instructors which are involved in Ethiopian Sign Language study and development will be advised about the data collection. In addition to experts and instructors, native users of sign language who use it in their daily life will be included.

1.4.3 Design and Developmental Tools

Sign language is important for facilitating communication between hearing impaired and the rest of society. Two approaches have traditionally been used: image-based and sensor-based systems. Sensor-based systems require the user to wear electronic gloves while performing the signs. The glove includes a number of sensors detecting different hand and finger articulations. Image-based systems use camera(s) to acquire a sequence of images of the hand. Each of the two approaches has its own disadvantages. The sensor-based method is not natural as the user must wear a cumbersome instrument while the image based system requires specific background and environmental conditions to achieve high accuracy. In this study, we will use Kinect which is a webcam-style add-on peripheral input line used for full body motion sensing.

1.4.4 Experimentation and Testing

The developed system will be tested as a whole and per each signer to test its effectiveness. This will help us to assess the strengths and weakness of the system.

1.5 Scope and Limitations

The study of sign language recognition focuses on the recognition of finger spelling or the recognition of signs in different sign languages. This research work focuses on the recognition of continuous signs which results in simple Amharic sentence from Ethiopian Sign Language. However, the study will not include the recognition of Amharic Finger Spelling since it was covered by previous studies. In addition, this study will not cover the recognition of non-manual feature which is facial expression but tries to estimate facial expression based on domain knowledge in combination with gestures extracted from hand, finger, head, and body movement.

1.6 Application of Results

This study will empower the effort exerted in solving the communication problem of the hearing impaired people with the hearing people. The research will have significant contribution in recognition of Ethiopian Sign Language. The benefits of this research are the following.

- It helps hearing people to understand EthSL.
- It can be used as learning material for EthSL.
- It will assist the formal education process.
- It will be very handy tool for development of translating EthSL to other Sign Language.

1.7 Organization of the Rest of Thesis

The rest of the thesis is organized as follows. Chapter 2 describes the overall structure of Ethiopian Sign language and the sign language recognition. Chapter 3 discusses and assesses related researches conducted on Ethiopian Sign language and other sign languages. In Chapter 4 the detailed description of the proposed model is presented. Chapter 5 explains the experiment of the proposed model and discusses the evaluation of the proposed system. Chapter 6 is the last chapter which gives conclusions and future works.

CHAPTER 2: LITERATURE REVIEW

This Chapter introduces sign language recognition and highlights its importance. It also presents pertinent background information and a literature review on the recent advances in sign language recognition. Phonology and characteristics of EthSL is discussed briefly. Toward the end of this Chapter, the main methods and models used for sign language recognition are briefly described.

2.1 Hearing Impaired Community

A person who is not able to hear as well as someone with hearing thresholds of 25 dB or better in both ears – is said to have hearing loss [2]. Hearing loss may be mild, moderate, severe or profound. It can affect one ear or both ears, and leads to difficulty in hearing conversational speech or loud sounds. 'Hard of hearing' refers to people with hearing loss ranging from mild to severe. They usually communicate through spoken language and can benefit from hearing aids, cochlear implants and other assistive devices as well as captioning.

'Hearing Impaired' people mostly have profound hearing loss, which implies very little or no hearing. They often use sign language for communication. Deafness is the inability to hear. The problem of deafness can be inherited or caused by various reasons such as complications during birth, some infectious diseases such as meningitis, use of toxic drugs and exposure to excessive noise. Members of hearing impaired cultures communicate via sign languages. Sign language is a naturally occurred language which is used for the communication of hearing impaired people. It is developed by communities of hearing impaired people who are leaving in different parts of the world.

The hearing impaired of Ethiopia live similarly to any other person within their given cultures, but are cut off from meaningful interaction with others. The vast majority of hearing impaired Ethiopians, who live in rural areas, spend their lives in extreme isolation. They are looked down upon as mentally deficient and evil because of their lack of spoken communication. In many places they are misunderstood as being a result of sinful behavior, or some form of supernatural curse. They are not seen as suitable marriage partners and may

even result in the entire family's loss of status [1]. For this reason, they are frequently sheltered even further from the outside world and communicate only with their families or those close to them through small amounts of writing or signing, if they are able. In towns more awareness has been generated regarding the deaf. Many parents are eager to send their children to schools, although the resources available are not sufficient for the number of potential students. Missionaries, and more so lately, the government, have established several schools for the deaf. In more recent years' clubs for the hearing impaired have been established in some towns, helping the hearing impaired to be less isolated and allowing sign language to be brought into use [1].

Late-Deaf and Born-Deaf

People who are born hearing and become deaf late in life, are "physically deaf", but "culturally hearing". They grew up speaking a spoken language, using different communication tools. They think, speak, read, write and base their opinions on the world they knew before they became deaf [6]. People who are born into the hearing impaired community, and whose first native language is a signed language, not a spoken one, are "culturally deaf". Some of them are born-deaf or became deaf at a very young age. Some of them are hearing people born into all-hearing impaired families, and even though they can hear, even though they speak a spoken language, their first language was a signed language, not a spoken language. They base their view of the world from the hearing impaired perspective. They are "physically hearing" but "culturally deaf" [6].

Usually hearing people are afraid of becoming blind. But to them, deafness doesn't seem nearly as bad. They forget that hearing is connected with the development of speech.

Hearing aids do not work for all hearing impaired people and even when hearing aids do work, they work with minimal success for the profoundly deaf, sounds are muffled. It is very hard to distinguish between voices and other sounds. Oftentimes a hearing impaired person can only hear bells or telephones with a hearing aid but cannot hear voices distinctly.

Deafness in hearing impaired families is often genetically based. There are several genes that produce deafness. In hearing impaired families, most often, everyone uses a signed language. They are "native signers", since it is their first language. Later they learn spoken language as their second language.

When a hearing impaired person are born into a hearing impaired family, they communicate in the same language of their parents. Their language development, using sign language from the moment of birth, is as normal as any hearing child's language development in spoken language. Everyone in the family uses the same language and the child starts signing at the expected age. They began absorbing language as babies.

Hearing impaired children born to hearing parents are not always so lucky. Often the hearing parents do not realize the child is hearing impaired until age three when they realize the child does not speak. The first three years are the crucial years for language development, so under those circumstances the child is deprived of "normal language development". Every member of the family is frustrated and communication is often poor at best.

2.2 Sign Language

Sign is the movement of one or both hands followed by facial expression and body movements to construct gestures which have meaning. A sign constructed from the movement of one hand is known as one handed sign and a sign which is constructed from the movement of the two hands is called two handed sign. In signing facial expressions add important information in the emotional aspect of the sign. The term sign in signing represents a word in spoken language. Most of the time sign represents common words. If the word is uncommon or a word which is out of the collection of words which are in use in the daily communication, the signer will be forced to spell it by the use of finger spelling. A sign language is a language which, instead of acoustically conveyed sound patterns, uses manual communication and body language to convey meaning. This can involve simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions to fluidly express a speaker's thoughts. Since it is a visual language, the abilities of speaking and hearing are not mandatory to use the language. Sign languages are indigenous. Many countries have their own sign languages. There is a difference between sign languages which are used within the same country just like the difference of dialect in verbal languages [2]. There are over 200 distinct sign languages in the world [14]. Like in spoken languages they have vocabularies, grammar and spelling (finger spelling) and have comparable language structure with spoken language. With

sign language it is possible to manifest hearing impaired people's culture, sing a song, present poetry and anything that a spoken language can do [2].

Characteristics of Sign Language

There are four major characteristics of sign language [14]:

- **Simultaneity**: refers to the transfer of large idea at a particular time that is by using different signs idea or information, which is considered being larger, can be transmitted.
- **Localization**: is explaining of ideas that can be compared and contrasted using sign language.
- **Movement**: is also one character, which is the nature or behavior of the hand movement.
- **Iconic:** Icons are symbols which share a physical resemblance to objects they represent. Because of the similarities between object and icons, the symbols usually are interpreted with little difficulty. Sign language is iconic or picture-like. Because icons are in most instances visible symbols, it is understandable that observers may relate shape and symbol when they study sign language.

2.3 Ethiopian Sign Language

Sign language was first taught in Ethiopia by American missionaries and is based on American sign language and signed English [15, 16]. It has been modified to suit Ethiopian culture but may still be intelligible with ASL.

In Ethiopia, sign language was first used formally after 1960's in connection with the appearance of American and Nordic missionaries who opened schools for the hearing impaired [17]. The missionaries brought the sign language used in their own countries. For more than 50 years, the foreign sign languages were assimilated with Ethiopian hearing impaired culture and sign language.

The hearing impaired community along with their association (The Ethiopian National Association of the Deaf) and along with other concerned bodies have been working for the institutionalization of the Ethiopian sign language. As a result of their concerned efforts, they have now a well-developed sign language which is used as a medium of instruction in schools and as means of communication by the media.

The hearing impaired in Ethiopia constitute a linguistic minority whose human and constitutional rights should be addressed along with other linguistic entities. To address the needs of this less privileged linguistic group by developing and promoting the national sign language that is being used now is developing and promoting equality and helping them to contribute in the development of the country.

Parameters of EthSL

It is difficult to see in detail when viewing something for the first time, especially something as complicated as sign. There are fingers, hands, arms, cheeks, lips, and eyebrows moving every way continuously and simultaneously. EthSL constitutes five parameters, each having a limited range of possibilities. These parameters will give us something to focus on:

- **Hand Shape:** The proper hand shape is the first parameter needed. How the hands are shaped when making signs can change the meaning of the word or expression we are trying to communicate. The basic hand shapes used for finger spelling are also used to make words when combined with other movements or signs. There are 41 hand shapes in the EthSL which are indicated by distinctive symbols [1]. For example, Figure 2.1 [1] shows hand shape for number two (ሁለት) and three (ሦስት), respectively

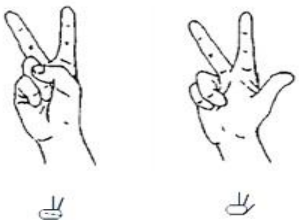


Figure 2.1: *Sign Representation of Number 2 And 3*

- **Hand Orientation:** Orientation refers to which way the hands are facing. Does the palm face toward the signer or away? This could change the meaning of the sign. Changing the orientation of the hands can reverse the meaning of the sign. For example, changing the orientation of phrase እኔ እጠይቅሃለሁ (I will ask you) will be changed to the meaning አንተ ጠይቀኝ (You ask me)" or "እኔ እሰጥሃለሁ (I will give you)" to" አንተ ትሰጠኛለህ (You will give me). Figure 2.2 illustrate the meaning difference that comes by changing the hand orientation from እኔ (I) to አንተ (you) respectively.

Figure 2.2*: Sign Representation of "I"and "You"*

- **Location:** The location or sign area relates to where the hands are held during signing. They can be against the head or other parts of the body, depending on what we are saying. The signing space is an imaginary rectangle, shoulder width, from head to just below waist. Some signs may go outside of this area, depending on what is being communicated, but the majority will be within this location. There are 12 possible locations for signing using EthSL [1]. Depending on what we need to sign the location will be changed. For example, whenever we sign intellectual thing signing will be done around head. For example, አሰበ (think) can be expressed as shown in Figure 2.3 [1].

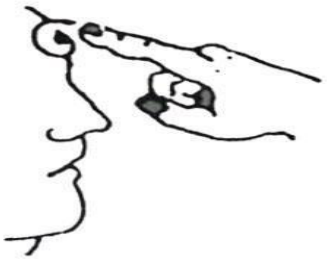


Figure 2.3: *Sign Representation of "Think"*

- **Movement:** Some signs also include a movement. Knowing the proper movement for what we are trying to sign is important in getting our thoughts across. Adding the wrong movement to a sign can change the meaning of the word or phrase we are using. The movement can be used to show the plural form (ዛፎች/Trees፣ ዓመታት/Years), vigor (በጣም ጥሩ/very good፣ በጣም መጥፎ/very bad), and extents (ትልቅ/big), in addition to indicating the type and direction of signing.
- **Expression (Non-Manual features):** Facial expression such as scowl, eye roll or happy expressions will let the person we are signing with know how we feel about what we are saying.

2.4 Signed Amharic

Signed Amharic is a sign language dialect which matches each spoken Amharic word. All aspects of Amharic are signed by following Amharic grammar. People generally speak at the same time when using Signed Amharic to give hearing impaired children the benefit of both signed and spoken Amharic. Example, Amharic phrase "አበበ ወደ ቤት ሄደ (Abebe went home)" can be signed as direct word by word "አበበ + ወደ +ቤት + ሄደ (Abebe + went + home)".

2.5 Signing Hand

Sign language is a body language which uses body parts to convey meanings. It combines hand shapes, orientation, movement of the hand (optional), and facial expression (needs to express feeling). The main component in signing is shape which is constructed by changing the shape of our hands either using both or one hand [1].

The two hands can be classified as dominant and non-dominant hand. If we are right-handed, our right hand is our dominant hand. If we are left-handed, our left hand is our dominant hand. If we are ambidextrous, choose one hand to use as dominant hand, and stick with it.

There are three types of signs when it comes to what hand we will use:

- One-handed signs: Uses only our dominant hand and signing process will be done using a single hand. For example, አንገት (neck)፣ ውሸት(lie)፣ መሳቅ(Laugh) are a one-handed sign, Figure 2.4 [1] shows their sign representation respectively.

Figure 2.4: *Sign Representation of "Neck, Lie, and Smile"*

- Two-handed symmetrical signs: Uses both our dominant and non-dominant hands where they both move the same way. For example, አስተማሪ (teacher)፣ ኮት (coat)፣

ብዙ (many) are two- handed symmetrical signs and are illustrated in Figure.

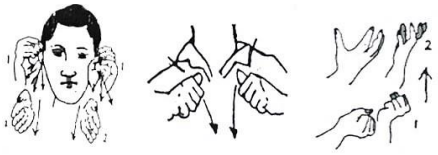


Figure 2.5: *Sign Representation of "Teacher, coat, and many"*

- Two-handed non-symmetrical signs: Uses both your dominant and non-dominant hand where the dominant hand moves while the non-dominant hand remains stationary. For example, ሩቅ (far)፣ ባዶ (empty) are two-handed non-symmetrical signs and are signed as shown in Figure 2.6 [1].

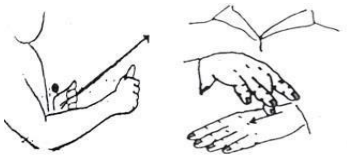


Figure 2.6: *Sign Representation of "Far and Empty"*

2.6 EthSL Grammar

a. Nouns and Plural forms

Nouns are common concepts to all languages. Like American Sign Language, EthSL does not alter the form of nouns to express plurality (for example, a 'noun' denotes a single thing and 'nouns', denotes more than one thing). In EthSL, the plurality can be expressed by repeating the noun or using additional sign ብዙ (many).

Table 2.0: *Example for Noun and Plural Forms*

Noun	***Singular form (Amharic)***	***Plural form (EthSL)***
ውሻ (dog)	ውሾች (dogs)	ውሻ + ብዙ (dog + many)
ሰው (person)	ሰዎች (persons)	ሰው + ብዙ (person + many)
መፅሃፍ (book)	መፅሃፍት (books)	መፅሀፍ + ብዙ (book + many)
ዛፍ (tree)	ዛፎች (trees)	ዛፍ ዛፍ ዛፍ(መደጋገም) (tree tree tree …)

When the noun is unknown noun or does not have any sign that illustrates it, signing process will be done using finger spelling.

b. Pronouns and Indexing

Indexing is when we set up a point to refer to a person or object that is or is not present in the signing area. Table 2.2.and 2.3 [1] shows how to sign pronouns in EthSL.

Table 2.2: *Pronouns and Indexing*

Pronouns	**EthSL**
እኔ (I)	Pointing to myself 
አንተ (You)	Pointing to you 
እሱ (He)	Pointing to the person 
እሷ (She)	Pointing to the person 
እነሱ (They)	Sign half circle by pointing to them 
እኛ (Our)	Sign arc by pointing my chest starting from the right to the left or vice versa 

Suppose we want to talk to someone about a person who is not physically nearby, we should use contrastive structure. The rules of contrastive structure are easy. First, identify the person by finger spelling his or her name; describing a few key features such as hair color or height also help. Then, we can index the person or object to a point in space. Once we have set up this referent, we can refer back to that same point every time we want to talk about that person or object.

c. Possessive Noun

Signing possessive nouns (የእኔ/ mine፣ የአንተ/ yours፣ ያንቺ/ yours፣ የኛ/ ours፣ የነሱ/ them) is similar to signing pronouns except it uses our palm instead of using index finger.

Table 2.3: *Possessive Noun*

Possessive noun	**EthSL**
የኔ (mine)	
የአንተ/ያንቺ (Yours)	
የኛ (Ours)	
የነሱ (them)	

d. Verbs

Verbs are the other common concepts in all languages. They have traditionally been defined as words that show action or state of being. In fact, without verbs, language would cease to exist. Verbs in EthSL come in two types: plain, and inflecting.

- **Plain Verbs**

A plain verb is a normal verb in EthSL. When using plain verbs, the signer must designate the subject and the object. Each plain verb must be signed using a single signing. Examples, መሮጥ (run)፣ መዝፈን (sing)፣ መጫወት (play) are plain verbs which do not have any indication of object and/or subject in a verb and they only show the action.

- **Inflecting/Indicating Verbs**

Inflecting/Indicating verbs incorporate the subject and object into the verb. When the signer sign inflecting verb, he/she signs the verb with the subject and/or object. Examples of inflecting verb in EthSL are shown in Table 2.4 [1].

Table 2.4: *Sample Inflecting/Indicating Verbs*

Amharic	**EthSL**
ሰጠ(he gave)	እሱ + መስጠት (He + gave)
ላከ(sent)	እሱ + መላክ (He + sent)
ከፈለች(she paid)	እሷ + መከፈል(she + paid)

e. Adjectives and Adverbs

Descriptive words are adjectives and adverbs. They modify nouns and verbs in detail. They also add imagery to our writing, speech, and signing.

Typically, EthSL puts an adjective after the noun it modifies, but one may place the adjective before the noun for stylistic purposes. Example, an Amharic phrase ቀይ ውሻ (red dog) is signed as ውሻ + ቀይ (dog red).

In Amharic the adverb is placed before the verb, whereas in EthSL it is placed after the verb. Most of the time, the placement of adverbs are similar with the placement of adjective in EthSL. Table 2.5 [1] shows the placement of adverbs in EthSL.

Table 2.5*: Sample Adjectives and Adverbs*

Amharic	**EthSL**
በፍጥነት ነዳው (he drive speedily)	(እሱ) + መንዳት + ፍጥነት / he + drive + speedily
ቶሎ ሄድኩ (I went quickly)	(እኔ) + መሄድ + ቶሎ/ I went quickly

f. Conjunction

The combining of two sentences in EthSL is different based on the conjunction needed. For example, the concept of the word "and" does not exist. Simply, sign a sentence, take a short pause and then sign the next sentence [1]. Example: An Amharic sentence እኔ እና አንተ አብረን ምሳ በላን (you and me ate lunch together) signed as እኔ+አንተ+አብሮ+መበላት+ምሳ (I + you + together + ate + lunch)

g. Interjections

Interjections are words used to express strong feelings or sudden emotion. They are included in a sentence usually at the start to express a sentiment such as surprise, disgust, joy, excitement or enthusiasm. Most of the time EthSL uses facial expression to express interjections but sometimes a few interjections can be expressed using signs [1]. For example, "በየሱስ ስም (be 'iyesus sem) (*Oh Jesus*) ፣ በስመአብ (be seme'ab) (*in God name*) ይገርማል (yigermal) (*amazing*)" have their sign equivalents.

h. Prepositions

Almost all prepositions are not used in EthSL, because it is reserved more for signing exact Amharic [1]. It is a good idea to avoid prepositions when signing in EthSL, because they are shown in context. For example: ከቤት መጣ (ke bEt meTa) (*coming from home*) is signed as ቤት መምጣት (bEt memeTat) (*home coming*), and the preposition ከ (ke) (*from*) is shown in the context, not signed as a word, but some prepositions are expected to be signed, for example, in the phrase ወደ ቤት ሄደ (wed bEt hEd) (*He went home*), the preposition wed indicating the direction of movement is expected to be signed as ወደ ቤት መሄድ (wed bEt mehEd) (*going to home*).

i. Negation

The role of negation in EthSL is a fairly easy concept to grasp. There are two ways to sign negate in a sentence. The non-manual marker for a negated sentence is simply a shake of the head and it is possible to sign “not” using hand. When signing the word not; one must remember that in EthSL syntax negation words always come at the end. Table 3.6 [1] illustrates the negation of the sentences:

Table 2.6: *Negation*

Amharic	**EthSL**
አልመታሁትም (I didn’t hit him)	እኔ + እሱ + መምታት + አይደለም (‘me + he + hiting + no’)
አልበላሁም (I didn’t eat)	እኔ + መብላት + አይደለም (‘me + eating + no)
አልተኛሁም (‘I didn’t sleep)	እኔ + መተኛት + አይደለም (‘me + sleeping + ‘no)

j. Placement of Time Words

Time words are the only things that come before the sentence in EthSL, but sometimes they come in the middle when the subject is unknown. Example, ጠዋት 2 ሠዓት ላይ ላግኝህ (Tewat 2 se’at lay lageNh) (*shall we meet tomorrow at 8 a.m.*) can be translated as ጠዋት+ሠዓት+2 እኔ+አንተ+መገናኘት+መፈለግ (Tewat + s’at + 2 + ‘nE + ‘ante + megenaNet + mefeleg) (*morning 8 a.m. me you meeting shall*).

k. Word Order of Questions

In other natural languages such as Amharic and English, statements are given in a particular word order. EthSL does not invert its word order nor does it add in any helping words. It uses non-manual signals to display a question asked. These non-manuals can consist of body movements, facial expressions, or eyebrow movements. Let’s examine a simple YES/NO question in EthSL and Amharic.

Amharic: ቁርስህን በላህ? (*have you ate your breakfast*?)

EthSL: አንተ መብላት ቁርስ? (*you eating breakfast?)*

In a YES/NO question the eyebrows are raised and the body is leaned forward slightly. These non- manuals show the receptive signer that the statement is actually a question. Another type of question is a WH-question. These types of question require more of a response than yes or no. They always include signs like WHAT, WHERE, WHEN, HOW, or WHY? These WH- words always come at the end of the question, unlike in Amharic where it is the first or middle word in the question.

Amharic: መቼ ትመጣለህ? (*when will you come?)*

EthSL: መምጣት መቼ? *(coming when ?)*

Similar to a YES/NO question, WH-word questions also have non-manual markers, however this time instead of raising eyebrows, we must lower our eyebrows. In addition to lowering eyebrows the signer must lean the body in slightly and extend the last sign for a couple of seconds. This allows the receptive signer to understand they are being asked a question that requires more of a response.

The final type of question is called an RH-question. The use of an RH-question is like an Amharic speaker using the word because (ምክንያቱም). There is no sign for the word because in EthSL, therefore they sign a question and answer it themselves. The non-manual markers for an RH-question are the same as a YES/NO question.

Amharic: ቁርሴን አልበላሁም ምክንያቱም ስላ ልራበኝ

ASL: እኔ ቁርስ መብላት አይደለም ለምን? መራብ አይደለም

l. General Syntax

With background on how parts of speech are used in EthSL, we can now evaluate the syntax, or word order, of EthSL. The article, the word order of EthSL is different from that of Amharic. Amharic follows a SOV, subject-object-verb sentence pattern, whereas EthSL uses a topic-comment pattern and the subject is topic of the sentence.

Amharic: አበበ በሶ በላ (Abebe ate “Besso”)

EthSL: አበበ መብላት በሶ (Abebe eating “Besso”)

2.7 Manual Alphabet and Numbers

Finger spelling is used for proper nouns. They may include, but are not limited to movie titles, books, names, and street names. The EthSL has 33 letters and each has 7 sub-types which are identified using movement.

Signing numbers is different from the normal signing that most Amharic speaker's use. All numbers below one thousand are signed using a single hand, and the second hand is only used to designate that a number is in the thousands or millions.

2.8 Sign Language Recognition

Sign language recognition systems are used to convert sign language into text or speech to enable communication with people who do not know these gestures. Usually, the focus of these systems is to recognize hand configurations including position, orientation, and movements. Accordingly, these configurations are captured to determine their corresponding meanings, using two approaches: sensor-based and vision-based. While the former requires wearable devices to capture gestures, it is usually simpler and more accurate. On the other hand, vision-based approach utilizes cameras to capture the sequence of images. Although, the latter is a more natural approach, it is usually more complex and less sensitive.

Generally, there are three levels of sign language recognition: finger spelling (alphabets), isolated gestures (single gesture or word), and continuous gesturing (sentences) [13]. Under Alphabet recognition, the signer performs each letter separately. Mostly, letters are represented by a static posture, and the vocabulary size is limited. Unlike alphabet sign recognition; word sign recognition techniques analyze a sequence of images representing the entire sign. Recently, recognizing continuously signed sentences has become the major focus. There are two major issues to be addressed in continuous signing. These are segmentation of continuously signed sentences, and dealing with movement epenthesis.

a. Segmentation in Continuous Signing

Unlike isolated signs, the start and end points of a sign are not well-defined in continuous signing. There are two ways to approach this problem, viz. explicit segmentation, where segmentation is performed prior to the classification stage and implicit segmentation, where segmentation is done along with classification. In explicit segmentation, the main concern is

to choose the correct cues that will allow inferring the physical transition points. Harling and Edwards [18] used hand tension as a cue to perform segmentation on two British sign language sentences. This was based on the idea that intentional gestures are made from one position to another with a tense hand. They also pointed out that higher level inputs such as grammar of the gestural interaction is crucial for segmentation tasks. Minimum velocity of hand movement was used to indicate hand transition boundaries in [19, 20]. Sagawa and Takeuchi [21] proposed that velocity alone was inadequate to segment sign language sentences in general, and used a parameter defined as "hand velocity" which included changes in handshape, direction and position. Minimal "hand velocity" was used as a candidate for a border. In addition, a transition boundary was indicated when a change in the hand movement direction was above a threshold. Recognition was carried out according to the method presented in [22]. Wang *et al.* [23] also used a similar method for trajectory segmentation. In Liang and Ouhyoung [24] work, transition boundaries were identified with time-varying parameter (TVP) detections. They assumed a gesture stream was always a sequence of transitions and posture holdings. When the parameter TVP fell below a threshold, indicating a quasi-stationary segment, it was taken to be a sign segment. 250 signs in Taiwanese sign language were recognized with 80.4% accuracy by HMMs trained with 51 postures, 6 orientations and 8 motions.

b. Movement Epenthesis

In continuous sign language recognition, an important issue is to efficiently tackle movement epenthesis, the extra connecting hand movement between two successive signs. The transition from the ending of one sign to the beginning of the next one is called movement epenthesis (ME). It connects adjacent gestures but does not hold significant information. In the literature, MEs were handled in three different approaches. First, while considering meaningful gestures, some researchers ignored MEs completely. Therefore, sentences were handled as concatenated words. However, this segmentation violates the purpose of natural continuous gesturing. On the other hand, in the second approach, some scholars labeled MEs explicitly as in [25]. The third approach considered them as part of their adjacent gestures. Figure 2.7 [26] depicts the different approaches to handle MEs in sentences.

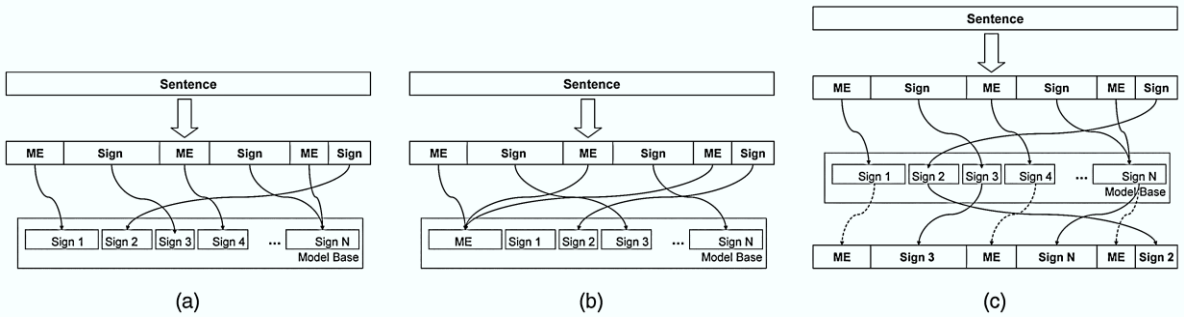


Figure 2.7: *Handling ME*

In (a) If the effect of ME is ignored while modeling, this will result in some ME frames falsely classified as signs. In (b) If ME is explicitly modeled, building such models will be difficult when the vocabulary grows large. In (c) The adopted approach in this paper does not explicitly model MEs; instead, we allow for the possibility for ME to exist when no good matching can be found [26].

2.8.1 Sign Capturing Methods

a. Sensor-based

sensor based approach entails wearable devices to capture gestures, and it is usually simpler and more accurate [12]. One of the traditional sign capturing method is data glove approach in Figure 2.8 [12]. These methods employ mechanical or optical sensors attached to a glove that transforms finger flexions into electrical signals to determine the hand posture. However, the disadvantage of this method is it requires the glove to be worn, which is a wearisome device with many cables connected into the computer that will hamper the naturalness of the user computer interaction.

Figure 2.8: *Example of the Data Glove*

b. Vision-based Sign Extraction

Another common sign capturing method is vision based sign extraction. This method is usually done by capturing an input image using a camera. In order to create a database for a gesture system, the gestures should be selected with their relevant meaning in which each gesture may contain multi samples to increase the accuracy of the system.

Vision-based method is widely deployed for sign language recognition. Sign gestures are captured by a fixed camera in front of signers. The extracted images convey posture, location and motion features of the fingers, palms and face. Next, an image-processing step is required in which each video frame is processed in order to isolate the signer's hands from other objects in the background.

The problem is that errors relate to a dynamic environment. Furthermore, the vast computation needed is another issue with a real time vision system. For example, *Elena Sanchez*-Nielsen *et al*. [27] suggested a real time vision system that uses a fast segmentation method, by using minimum features to identify hand posture in order to speed up the recognition process. Brashear *et al*. [28] used a camera mounted above the signer, so the images captured by this camera clearly solve the overlapping between the signer's hands and head. However, unfortunately, face and body gestures are lost this way.

c. Recent Development in Sign Language Recognition Methods

New trends in sign capturing devices like Microsoft Kinect provides an inexpensive and easy way for real-time user interaction. The software driver released by Microsoft called Kinect Software Development Kit (SDK) with Application Programming Interfaces (API) gives access to raw sensor data streams as well as skeletal tracking Zhang *et al*. [29]. Although there is no hand specific data available for gesture recognition, it does include information of the joints between hands and arms. Little work has been done for Kinect to detect the details of the level of individual fingers. Figure 2.9 [29] is the example of Kinect device.

The Kinect and depth cameras, in general, are well suited for sign language recognition. They offer 3D data from the environment without a complicated camera setup and efficiently extract the users' body parts, allowing for recognition of not just hands and head, but also other parts such as elbows that can be of further help in distinguishing between similar signs. Another advantage is the independency of lighting conditions, as the camera

uses infrared light. All body parts are detected equally well in a dark environment and there is no need from the user to wear special colored gloves or wired gloves.

The recently developed methods of gesture recognition include the use of active depth cameras or sensors, which provide substantial data from the observed environment, such as three-dimensional information. Kinect sensor [30] is an example of such device. It provides color image, depth map, and 3D skeletal data indicating the most important 25 body joints. Among works with Kinect, Lai *et al*. [31] recognized eight static hand gestures with accuracy of 99%, and *Ren et al.* [32] recognized 14 static hand shapes which were controlling an application performing arithmetic operations, and also three shapes for *Rock-paper-scissors* game

(a) Kinect for windows (b) Kinect for Windows v2

Figure 2.9: *Kinect for Windows Hardware*

2.8.2 Sign Language Recognition Methods

In Section 2.8.1, an overview of SLR system was given based on employed data acquisition and feature extraction methods. It is obvious that extracting necessary data is a crucial part in SLR. However, the success of recognition process depends on not only the performance of the feature extraction method but also the performance of the classification method. In this section, how SLR systems classify signs after features are extracted is reviewed. In other words, an overview of SLR systems is given based on the classification methods used. Several methods have been used for sign language recognition, some of which are discussed in the following subsections.

a. Hidden Markov Models (HMM)

HMMs are a type of statistical models that can model spatiotemporal information in a natural way. HMMs have efficient algorithms for learning and recognition, such as the Baum-Welch

algorithm and Viterbi search algorithm [37]. A HMM is a collection of states connected by transitions. Each transition (or time step) has a pair of probabilities: a transition probability (the probability of taking a particular transition to a particular state) and an output probability (the probability of emitting a particular output symbol from a given state).

A gesture sequence is represented as a set of observations. An observation ft, is defined as an observation vector made at time t, where ft= {O1,O2,...,OM} and M is the dimension of the observation vector. A particular gesture sequence is then defined as $G=\{F_1,F_2,...,F_T\}$.

HMM is characterized by the following [37]:

1. N, the number of states in the model. We denote the individual states as $S=\{s_1,s_2,....,s_N\}$, and the state at time t as qt.
2. M, the dimension of the observation vector.
3. $A = \{a_{ij}\}$, the state transition probability distribution. Where A is an N ×N matrix and a_{ij} is the probability of making a transition from state s_i to s_j.
4. B = {bj(f)}, the observation symbol probability distribution. Where bj is the probability distribution in state j and $1 \leq j \leq N$.
5. $\pi = \{\pi i\}$, the initial state distribution.

The compact notation $\lambda= \{A,B,\pi\}$ is used to indicate the complete parameter set of the model where A is a matrix storing transition probabilities aij between states si and sj, B is a matrix storing output probabilities for each state and π is a vector storing initial state probabilities.

They have been utilized for the task of gesture recognition in a large number of works in the literature. HMMs were first used for the task of gesture recognition by Yamato et al. [38] and for the task of sign language recognition by *Starner et al*. [39]. In these seminal works, the authors state that the key characteristics of HMMs, which make them suitable for gesture recognition, is their learning ability and time-scale invariability.

b. Dynamic Time Warping (DTW)

DTW is a well-known technique used to optimally align two-time series data which may vary in time or speed. It is generally used in speech recognition applications to cope with inter-speaker differences in speaking speed. Think about two samples of time-dependent sequences which have local changes independent from each other in time and speed as shown on Figure 2.9 [40]. DTW finds an optimal path which align these sequences without requirement that

they have the same length. Whether or not two sequences differ non-linearly in time, DTW measures similarity or distance between them independent of these variations by warping them non-linearly in time.

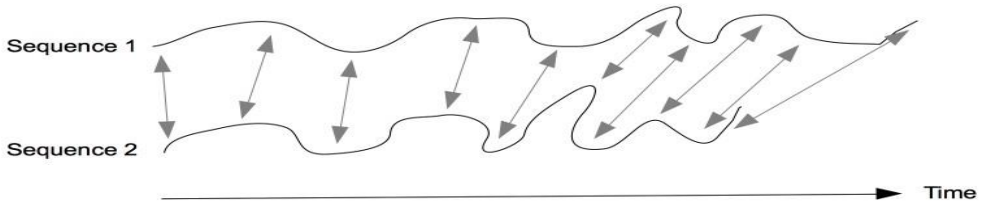


Figure 2.9: *Alignment of two Time-dependent Sequences.*

DTW compares a sequence X = (x1 , x2, ..., xN) of length N with another sequence Y = (y1, y2, ..., yM) of length M , aligns their members and returns an alignment path in addition to a similarity/distance measure. Members of these sequences are, in general, sets of features. Műller [40] names these sets as feature space and denotes it by F . Then, local cost measure, c, is the function designed for comparing two feature spaces x, y ∈ F .

$$c : F \times F \rightarrow R \geq 0. \tag{2.1}$$

Measuring local cost of each pair of sequences X and Y, local cost matrix is obtained. It is defined by C ∈ RN ×M and C (n, m) = c(xn , ym). An optimum alignment path of sequences X and Y goes along the way where the values of local cost are low on the local cost matrix.

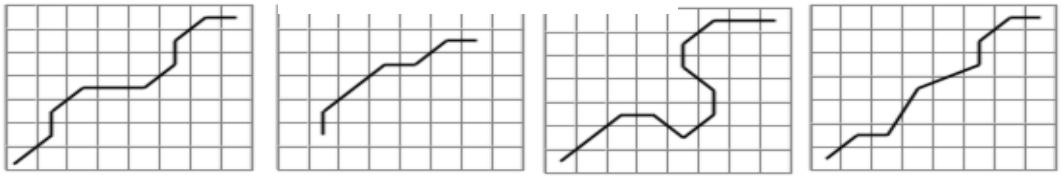
Műller [46] gives the definition of the warping path obtained as a result of DTW technique as :

Warping path of two sequences of length N and M is a sequence p = (p1, p2 , ..., pL) where pl = (nl , ml) ∈ [1 : N] × [1 : M] for l ∈ [1 : L]. The warping path satisfies the following conditions:

1. $p_1 = (1, 1)$ and $pL = (N, M)$.
2. $n1 \leq n2 \leq ... \leq nL$ and $m1 \leq m2 \leq ... \leq mL$.
3. $p_{l+1} - pl \in (0, 1), (1, 0), (1, 1)$ for $l \in [1 : L - 1]$.

An alignment of two sequences X = (x1, ..., xN) and Y = (y1 , ..., yM) is defined by a warping path by matching the element xnl of X with the element yml of Y . A warping path which accomplishes the optimal matching should satisfy the three conditions defined above. Figure 2.10 [40] shows some examples of warping paths. The cost of a warping path p of two sequences X and Y can be calculated as:

$$C_p(X,Y)^{\square} = \sum_{l=1}^{L} c(x_{n1}, y_{n1}) \qquad (2.2)$$



(a) (b) (c) (d)

Figure 2.10*: Illustration of warping path indices of two sequences of length 8 and 7.*

(a) Warping path satisfying all three conditions. (b) Warping path violating the first condition (c) Warping path violating the second condition. (d) Warping path violating the third condition.

DTW guarantees to find an optimal warping path which has the lowest total cost compared to all possible warping paths even if there exists more than one optimal path. The optimal path is denoted by p∗. Then, the DTW distance of two sequences X and Y can be defined as Rashmi D. [10] states :

DT W (X, Y) = cp∗ (X, Y) = min {cp (X, Y) | p is a warping path of X and Y }. (2.3)

In order to get rid of computational complexity of calculating all possible warping paths to find the optimal one, an algorithm based on dynamic programming having O(N M) computational complexity, where length of sequences are N and M , can be used.. Let X and Y be two sequences of length N and M, respectively, and X (1 : n) = (x1, ..., xn) for n ∈ [1 : N] and Y (1 : m) = (y1, ..., ym) for m ∈ [1 : M] be sequence prefixes. Then,

$$D(n, m) = DT W (X (1 : n), Y (1 : m)), \qquad (2.4)$$

where D is a N × M matrix and referred to as accumulated cost matrix. It can be easily obtained from Equation 2.4 that D(N, M) = DT W (X, Y).

$$D(n,m) = \begin{cases} c(x_1, y_1) & if\ n = 1, m = 1 \\ \sum_{k=1}^{n} c(x_k, y_1) & if\ 1 < n \leq N, y = 1 \\ \sum_{k=1}^{m} c(x_1, y_k) & if\ x = 1, 1 < m \leq M \\ min\{D(n-1, m-1), D(n-1, m), & \\ \quad D(n, m-1)\} + c(x_n, y_m) & if\ 1 < n \leq N, 1 < m \leq M. \end{cases} \qquad (2.5)$$

Computation of DT W (X, Y) = D(N, M) using the Equation 2.5 [40] has computational complexity of O(N M).

The cost of warping two sequences in time, i.e., their DTW distance, is given by accumulated cost matrix which is also used to obtain optimal warping path $p*$. Algorithm 2.1 extracts optimal warping path. Note that the algorithm uses only accumulated cost matrix as input. The optimum alignment path goes along the way on the accumulated cost matrix where values of the elements are low, as expected.

Several researchers used DTW in gesture and sign language recognition system. Li and Greenspan in [41], used Compound Gesture Models, in which the temporal endpoints of a gesture were estimated by DTW and a bounded search was performed to recognize the gesture. The proposed method is both computationally efficient and robust. In experiments containing nine different gestures and five subjects, the resulting average recognition rates were 93.3% for single scale and 88.1% for multiple scale continuous gestures. The Microsoft Kinect XBOX 360TM is proposed to solve the problem of sign language translation by D. Capilla [42]. By using the tracking capability of this RGB-D camera, a meaningful 8-dimensional descriptor for every frame is introduced here. In addition, an efficient Nearest Neighbor DTW and Nearest Group DTW are developed for fast comparison between sign languages. For a dictionary of 14 homemade signs, the introduced system achieves an accuracy of 95.24%. Halim and Abbas in [43] presented DTW-based approach for Pakistani Sign Language recognition with the accuracy of 91%.

c. Random Forest (RF) Classification Techniques

The random forest is an ensemble approach that can also be thought of as a form of nearest neighbor predictor. Ensembles are a divide-and-conquer approach used to improve performance [10]. The main principle behind ensemble methods is that a group of "weak learners" can come together to form a "strong learner". The random forest starts with a standard machine learning technique called a "decision tree" which, in ensemble terms, corresponds to our weak learner. In a decision tree, an input is entered at the top and as it traverses down the tree, the data gets bucketed into smaller and smaller sets. In other words, random forests are an ensemble learning method for classification and regression that operate by constructing a lot of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. In decision tree structure, leaves present class labels and branches represents conjunctions of features that lead to those class labels. In this technique, data comes in records of the form:

$$(x,Y)=(x_1, x_2, x_{3,\ldots\ldots,} x_k, Y) \qquad (2.6)\ [10]$$

The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of input variables, $x_1, x_2, x_{3,\ldots\ldots,} x_k$ that are used for classification purpose.

Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set. When a new input is entered into the system, it is run down all of the trees. The result may either be an average or weighted average of all of the terminal nodes that are reached, or, in the case of categorical variables, a voting majority.

How random forest works

Each tree is grown as follows:

1. Random Record Selection: Each tree is trained on the total training data. Cases are drawn at random with replacement from the original data. This sample will be the training set for growing the tree.

2. Random Variable Selection: Some predictor variables (x) are selected at random out of all the predictor variables and the best split on these x is used to split the node. The value of m is held constant during the forest growing. In a standard tree, each split is created after

examining every variable and picking the best split from all the variables.

3. For each tree, calculate the misclassification rate -out of bag (OOB) error rate. Aggregate error from all trees to determine overall OOB error rate for the classification.

4. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes over all the trees in the forest. For a binary dependent variable, the vote will be YES or NO, count up the YES votes. This is the RF score and the percent YES votes received is the predicted probability. In regression case, it is average of dependent variable.

2.8.3 Review of Shape Representation Techniques

There have been many shape representation and description techniques proposed for many different visual feature recognition applications [44]. Various shape techniques have been proposed, including shape signature, signature histogram, shape invariants, moments, curvature, shape context, shape matrix, spectral features etc. In general, shape representation techniques can be classified into two class of methods: contour-based and region based methods. Under each class, the different methods are further divided into structural approaches and global approaches. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections.

a. Global Contour Descriptors

Some common simple global contour descriptors include: area, circularity, eccentricity, major axis orientation and bending energy. These simple global descriptors usually can only discriminate shapes with large differences and therefore are not suitable as stand-alone shape descriptors. Another global contour descriptor is a technique known as correspondence based shape matching which measures similarity between shapes using point to point matching. Hausdorff distance is type of correspondence-based shape matching and has been used to locate objects in an image and measure similarity between shapes [45]. Shape matching using Hausdorff distance is sensitive to noise and slight variations in shape.

Shape signatures represent another class of global contour descriptors. Shape signatures are calculated from a one dimensional function derived from the shape boundary points. Many shape signatures exist, they include centroidal profile, complex coordinates, centroid distance, tangent angle, cumulative angle, curvature, area and chord-length [46]. Contour based features

have also been shown to perform well in hand posture recognition Al-Jarrah et al. [47] extracted features by computing vectors between the contour's center of mass and localized contour sequences. Recognition of 30 gestures was reported with an accuracy of 92.55%. Handouyahia *et al.* [48] presented a sign language alphabet recognition system using a variation of Size Functions [49] called moment based size functions, which recognized 25 different signs with 90% accuracy.

b. Global Region-based Descriptors

In region based techniques, all the pixels within a shape region are taken into account to obtain the shape representation, rather than only use boundary information as in contour base methods. Geometric moments have been widely used for a number of different shape analysis applications [50]. Using nonlinear combinations of lower order moments, a set of moment invariants which have desirable properties of being invariant under translation, scaling and rotation, are derived. The main problem with geometric moments is that the few invariants derived from lower order moments are not sufficient to accurately describe shape on their own. Orthogonal moments, an alternative to geometric moments, was proposed by Teague [51]. Teague extended the idea of algebraic moments to a more general form and introduced Legendre moments and Zernike moments. Orthogonal moments allow for accurate reconstruction of the described shape, and makes optimal utilization of shape information. Although Zernike moment descriptor has a robust performance, it has several shortcomings. First, the kernel of Zernike moments is complex to compute, and the shape has to be normalized into a unit disk before deriving the moment features. Second, the radial features and circular features captured by Zernike moments are not consistent, one is in spatial domain and the other is in spectral domain. It does not allow multi-resolution analysis of a shape in radial direction. Third, the circular spectral features are not captured evenly at each order, this can result in loss of significant features which are useful for shape description.

In this work we use a combination of a global contour based descriptors by matching with template in database. In system design chapter we discussed how these techniques are applied to hand posture recognition.

2.8.4 Models for Division of Signs into Subunits

In speech recognition, there are components called **phoneme** which are defined to be the smallest contrastive unit in a language; that is, a unit that distinguishes one word from another. Similarly, there is an attempt to classify signs based on subunits. Subunits are the equivalent term in sign language recognition for phonemes in speech. In EthSL, an example of such a phoneme would be the downward movement in the sign for "good." Instead of modeling the entire signs it is better to model them using subunits.

Division of a sign into subunits is the major task in modeling the recognition system. The following models are widely known for subdivision of signs into subunits.

a. Stokoe Model

W. Stokoe [52] realized that signs can indeed be broken down into smaller parts . He used this observation for devising a transcription system. This transcription system assumes that signs can be broken down into three parameters (phonemes), which consist of the location of the sign (tabula or tab), the handshape (designator or dez), and the movement (signation or sig). A fundamental assumption of this system is that the tab, dez, and sig contrast only simultaneously. That is, variations in the sequence of these parameters within a sign are considered not to be significant.

Stokoe developed a lexicon for American Sign Language by means of the above mentioned types of cheremes. Cheremes are also the equivalent of phonemes in speech recognition. The lexicon consists of nearly 2500 entries, where signs are coded in altogether 55 different cheremes (12 'tab', 19 'dez' and 24 different 'sig') [53].

The employed cheremes seem to qualify as subunits for a recognition system. However, their practical employment in a recognition system turns out to be difficult. Even though Stokoe's lexicon is still in use today and consists of many entries, not all signs are included in this lexicon. Also most of Stokoe's cheremes are performed in parallel, whereas a recognition system expects subunits in subsequent order. Many transcription systems are based on the Stokoe system, such as [54].

b. Movement-Hold Model

S. Liddell and R. Johnson [55] argued convincingly against Stokoe's assumption that there was no sequential contrast in ASL. They went even further and made sequential contrast the basis of ASL phonology; that is, instead of emphasizing the simultaneous occurrence of phonemes in ASL, they emphasized sequences of phonemes. Such models are called **segmental models.** S. Liddell and R. Johnson describe two major classes of segments in their Movement-Hold model in [55], which they call **movements** and **holds**. Movements are defined as those segments during which some aspect of the signer's configuration changes, such as a change in handshape, a hand movement, or a change in hand orientation. Holds are defined as those segments during which all aspects of the signer's configuration remain stationary; that is, the hands remain stationary for a brief period of time.

Signs are made up of sequences of movements and holds. Some common sequences are *HMH* (a hold followed by a movement followed by another hold, such as "good"), *MH* (a movement followed by a hold, such as "sit"), and *MMMH* (three movements followed by a hold, such as "chair"). Attached to each segment is a bundle of articulatory features that describe the hand configuration, orientation, and location. In addition, movement segments have features that describe the type of movement (straight, round, sharply angled), as well as the plane and intensity of movement.

Although the Movement-Hold model has some shortcomings, such as the absence of non-manual features and the presence of redundancy, its basic sequential structure has been accepted [56]. In addition, a sequential phonological model is ideally suited for a Hidden Markov Model recognition framework. In this work we, follow the ideas of the Movement-Hold model, but focus only on the movement types and the locational features.

CHAPTER 3: RELATED WORK

In this Chapter, a review of different local and international researches on sign language recognition are presented. Many previous researchers have tried developing sign languages recognition and translation systems in general and Ethiopian sign language in particular. There are similar researches that are conducted to recognize non-Ethiopian Sign Languages. But these researches are highly language reliant and none of them are investigated and verified to recognize Ethiopian Sign Language.

Among the international research reviewed Starner and Pentland [57] have done a research work titled visual recognition of ASL using HMM. The proposed system describes sentence level ASL recognizer using HMM with the idea of it being popular in speech recognition as well as in handwriting recognition. In this recognition system, sentences of the form "personal pronoun, verb, noun, adjective," are recognized. Six personal pronouns, nine verbs, twenty nouns, and five adjectives are included making the total lexicon number forty words. As described in this paper HMM have intrinsic properties which make them very attractive for SLR. For the data collection, they used a signer that wears distinctly colored gloves on each hand (a yellow glove for the right hand and an orange glove for the left) and sits in a chair before the camera. For the feature extraction they have used the position of the hands, some concepts of the shape of the hand and the angle of the hand relative to horizontal. Thus, an eight element feature vector consisting of each hand's x and y position, angle of axis of least inertia, and eccentricity of bounding ellipse was chosen. The system attains a word accuracy of 99.2% without explicitly modeling the fingers.

Starner *et al*. [58] continued their research entitled a wearable computer based ASL recognizer which is an extension of their previous work mentioned above. The paper describes a recognizer which uses one color camera pointed down from the brim of a baseball cap to track the wearer's hands in real time and interpret ASL using HMM's. As noted in the paper the hand tracking stage of the system does not attempt a fine description of hand shape rather the tracking process produces only a coarse description of hand shape, orientation, and trajectory. In this paper, the authors have used two methods of hand tracking: one, using solidly-colored cloth gloves (a pink glove for the right hand and a blue glove for the left), and two, tracking the hands directly without aid of gloves or markings

in which the hand is tracked based on skin tone. They have come up with a cap camera mounted vision-based system that is capable of recognizing ASL through the use of HMM with low error rate on both training set and an independent test set. However, the cap camera mount is probably inappropriate for natural sign because signing involves facial gestures and head motion, which would have confounding effect on the hand tracking. So, the authors recommend a necklace that may provide a better mount for determining motion relative to the body, and another possibility is to place reference points on the body in view of the cap camera. The overall system performance in terms of accuracy is reported to exceed 97% per word on a 40-word lexicon.

Numerous studies have attempted to use the Microsoft Kinect to identify hand gestures. Zafrulla *et al*. [59] investigated the potential of the Kinect depth-mapping camera for sign language recognition. They collected a total of 1000 ASL phrases and used a HMM to recognize the signed phrases resulting 85.5% mean accuracy.

Ren *et al*. [60]. researched a robust hand gesture recognition system using a Kinect. They proposed a modified Finger-Earth Mover's Distance metric in order to distinguish noisy hand shapes obtained from the Kinect sensor. They achieved a 93.2% mean accuracy on a 10-gesture dataset.

Chai *et al.* [61] proposed a sign language recognition and translation system based on 3D trajectory matching algorithms in order to connect the hearing impaired community with non-hearing impaired people. They extracted 3D trajectories of hand motions using the Kinect, and collected a total of 239 Chinese sign language words to validate the performance of the proposed system. They achieved rank-1 and rank-5 recognition rates of 83.51% and 96.32%, respectively.

Moreira Almeida *et al*. [62] also proposed a sign language recognition system using a RGB-D sensor. They extracted seven vision-based features from RGB-D data, and achieved an average recognition rate of 80%.

There are very few studies which are conducted on Ethiopian Sign Language. But most of them are carried out on the conversion of an Amharic text to equivalent sign in Ethiopian sign language. Traditionally, there have been three level of image based sign language recognition systems: alphabet, isolated word, and continuous recognition [12].

Alphabet Recognition: Under this situation, the local research, which is entitled Ethiopian Sign Language using artificial neural network, was done by Yoseph Admasu and Raimond [14]. The designed system focused on hand gesture detection and recognition technique for EthSL, for the recognition artificial neural network (ANN) has been employed to recognize the EthSL and translate to Amharic voice. For the data collection, one (right) hand finger spelling was used to capture 34 letters of Ethiopian Manual Alphabet from ten volunteers to have a total of 340 hand gesture images. To reduce difficulty of segmentation caused by high variation in skin color, the signers were instructed to wear white glove in their right hand. Yoseph Admasu and Raimond [14] applied two approaches for feature extraction. The first approach uses PCA and the second approach uses Gabor Filter (GF) together with PCA. The recognition process achieved a result of 95.58% for the first and 98.52% for the second approach.

Isolated Word Recognition: Unlike alphabet sign recognition; word sign recognition techniques analyze a sequence of images representing the entire sign. Under this category, Recognition of Isolated Signs in Ethiopian Sign Language was developed by Tefera Gimbi [9]. The study succeeded to achieve result on the overall system recognition 86.9%. Twenty-five signs have been chosen to be part of the research. Out of these 25 signs, 20 of them are one handed signs while the rest 5 are signs which are conducted using two hands. Each signer is expected to perform a sign 20 times. They used 15 of them for training purpose and the rest five for testing purpose. Therefore, for a specific sign we will have 45 training videos and 15 testing videos. They used a total of 1500 videos for all 25 signs. The videos are captured in MP4 format. The signers perform the sign in a constant speed. Ids are given for each sign and signers. In addition to this, they gave sequence numbers for videos of a sign. Therefore, they organized the captured videos with a signer id, sign id and a sequence. Furthermore, the preparation of the input to train HMM also conducted in this component. Cluster matrices were formed for each sign using the k-means algorithm.

Continuous Sign Language Recognition: While attractive in practice, continuous sign language recognition is more challenging than alphabet and isolated sign recognition. The main challenge lies in detecting and modeling the extra movement resulting from the transition between the end of a certain sign and the start of the next one. Furthermore, automatically detecting where a gesture starts and ends is a challenging problem. So far, we didn't find any related study conducted on Continuous EthSL Recognition.

The proposed methodology addresses issues faced in continuous sign language recognition by handling ME and sign segmentation by analyzing movement pauses. Furthermore, we proposed language model providing simple solution with inverted indexing concept to reorder topic-comment pattern and the subject topic pattern of EthSL sentence to subject-object-verb sentence pattern of spoken Amahric language.

CHAPTER 4: SYSTEM DESIGN

The Chapter describes detailed techniques and ways employed to design and model the recognition of continuous signs in Ethiopian sign language. The system architecture describes the overall design of the system and all the components of the system architecture will be discussed in detail.

The system accepts a video of continuous sign of Ethiopian sign language from Kinect device and generates equivalent text as an output.

4.1 System Architecture

The system has five main components:

- Data Acquisition
- Feature Extraction
- Feature Combination
- Training random forest model using the extracted features and
- Recognition

The system architecture in Figure 4.1 shows how these components interact to accomplish the recognition process. The system architecture is mainly divided into two parts. The first part is the training section in which the system will be trained. The second part is the recognition process which will follow similar step with the first one till the feature extraction step and the recognition process continues. In addition, movement epenthesis handling and gesture segmentation component is included in the recognition part. The system uses different data which was not used during the training process.

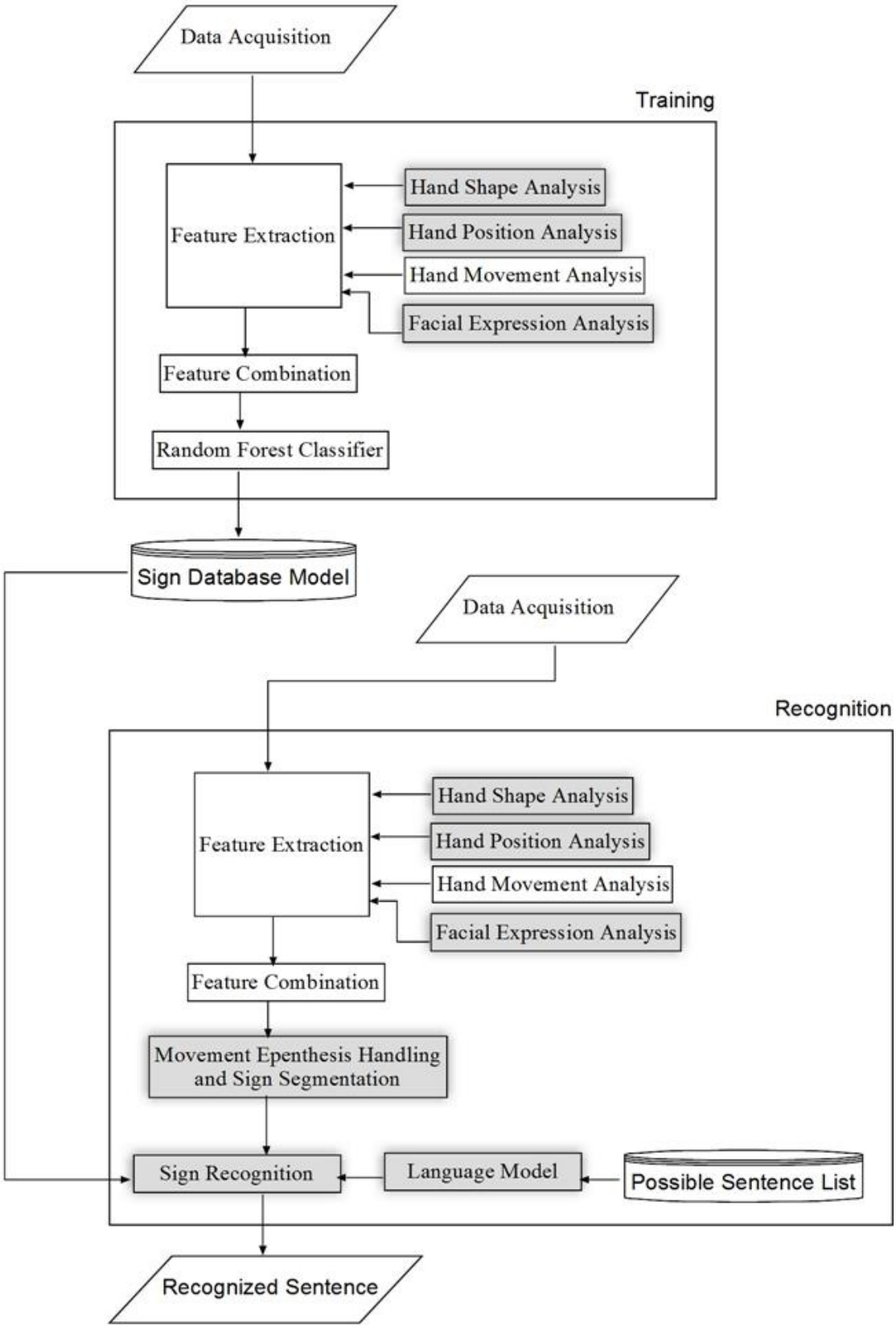


Figure 4.1: *System Architecture*

4.2 Data Acquisition:

During the data acquisition phase, information about RGB images, skeletal joint position as well as 3d depth information of the user are collected using Kinect device. By default, kinect streams 25 joint positions within a frame rate of 30 frames per second, but we are interested on 7 joints namely Hand Right, Hand Left, Shoulder Right, Shoulder Left, Elbow Right, Elbow Left and Spine. Each joint is identified with its name and X, Y, Z position coordinates are given in millimeters from the device.

4.3 Feature Extraction:

As we explained earlier sign language is defined by manual and non-features. Manual feature consists of hand shape, hand position, hand movement and hand orientation while non manual features include facial expression.

a. Handshape recognition

As defined by Stokoe's model [52], a hand posture is made up of the shape and orientation of the hand. EthSL contains many handshapes, but we classify and recognize most frequently used and important hand shapes as shown on Figure 4.2 [52] swhich are closed hand, open hand, thumbs up, little finger and thumb up, spread hand, victory, pointing.

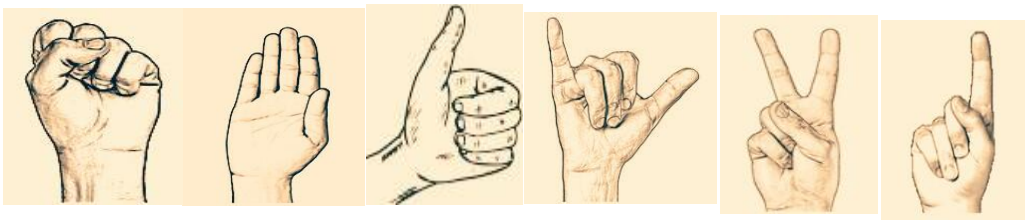


Figure 4.2: *Selected Hand Postures*

Hand shapes are grouped using a shape context which is created by constructing compact shape descriptors from samples extracted from the contour of a hand shape. We used contour analysis to compare and recognize handshapes with template image stored in database.

Once we get depth video provided by Kinect, we start by detecting the Hand joints of a given Body. This way, the algorithm knows whether a joint is properly tracked and whether it makes sense to search for fingers. Since we have detected the Hand joints, we can now limit the search area as depicted on Figure 4.3 [64]. The method only searches within a reasonable distance from the hand. We empirically decided a reasonable distance to be an area that is limited between the Hand and the Tip joints (10-15 cm, approximately). Since we have strictly defined the search area in the 3D space, we can now exclude any depth values that do not fall between the desired range! As a result, every depth value that does not belong to a hand will be rejected. We have an, almost perfect, shape of a hand. The outline of this shape is the contour of the hand

Figure 4.3: *Search Area Approximate Distance and Extracted Hand Shape Contours*

once we get the contour of the hand, the general sequence of an operation at recognition looks as follows [64]:

1. Preliminary handling of the image - smoothing, a filtration of noise, a contrast raise
2. Binarization of the image and selection of contours of objects
3. Initial filtration of contours on perimeter, squares, to a crest factor, fractality and so on
4. Coercion of contours to uniform length, smoothing
5. Search of all discovered contours, searching of the template maximum similar to the given contour. We will not consider points 1 and 3, they are specific to application area, and have the small relation to a Contour Analysis (CA).

Further, we consider an algorithm body - searching and comparing of contours with templates. Then we stop on binarization, coercion to uniform length and smoothing of contours a little.

For fast searching of templates, it is necessary to introduce the certain descriptor characterizing the shape of a contour. Thus, close among themselves contours should have the close descriptors. It would save us the procedure of an evaluation an Intercorrelation Function (ICF) of a contour with each template.

Equalization of Contours

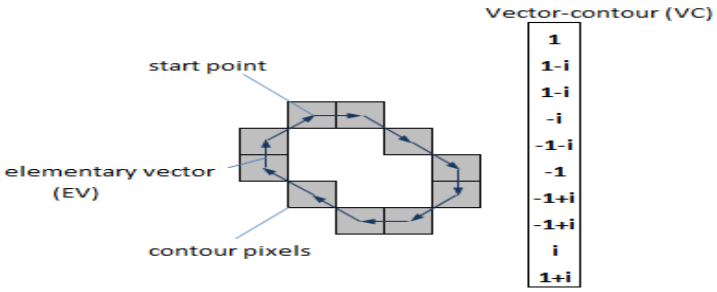


Figure 4.4: *Contour Encoded by The Sequence Consisting of Complex Numbers.*

On a contour, the point which is called as starting point is fixed. Then, the contour is scanned (is admissible - clockwise) as shown on Figure 4.4 [64], and each vector of offset is noted by a complex number a+ib. Where a - point offset on x axis, and b - offset on y axis. Offset is noted concerning the previous point.

Contour analysis methods assume identical length of contours. In the real image contours have arbitrary length. Therefore, for searching and comparing of contours, all of them should be led to uniform length. This process is called *equalization*.

As depicted on Figure 4.5 [64], first, we fix length of a Vector Contour (VC) which we will use in our system of a recognition. We designate it k.

Then, for each initial contour A we create vector-contour N in length k. Further probably two variants - or the initial contour has greater number of an Elementary Vector (EV) than k, or smaller number than k. If an initial contour is more necessary it is sorted out all by its EV, and we consider elements N as the total of all EVs, as follows (C#):

```
Complex[] newPoint = new Complex[newCount];

for (int i = 0; i < Count; i++)

    newPoint[i * newCount / Count] += this[i];
```

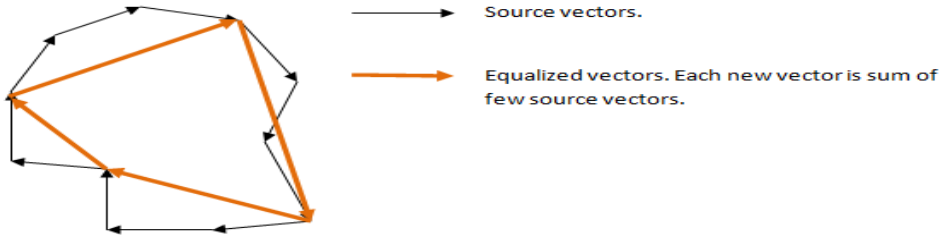


Figure 4.5: *Illustration of equalization.*

This algorithm is rough enough, especially for lengths the little big k, however it quite puts into practice. If the initial contour is less k we produce interpolation, and is considered approximately so:

```
Double[] newPoint = new Double[newCount];

for (int i = 0; i < newCount; i++)

{

    double index = 1d * i * Count / newCount;

    int j = (int)index;

    double k = index - j;

    newPoint[i] = this[j] * (1 - k) + this[j + 1] * k;

}
```

When selecting the value of k, it has an effect on recognition process, the big length k means the big expenditures on evaluations. On the other hand, small values k carries less information, and accuracy of a recognition decreases, and noise recognition increases.

At great values of length, smoothing becomes more and more small-scale, and the contour becomes too detailed, and also becomes more different from template contours.

Correlation Function of Contours

Let's introduce the concept of *intercorrelation function* (ICF) of two contours:

$$\tau(m) = \left(\Gamma, N^{(m)}\right), \qquad m = 0, \dots, k-1 \qquad (4.1)\ [64]$$

Where $N^{(m)}$ - a contour received from N by cycle shift by its EV on m of elements.

For example, if N = (n1, n2, n3, n4), $N^{(1)}$ = (n2, n3, n4, n1), $N^{(2)}$ = (n3, n4, n1, n2) and so on.

ICF shows contours Γ and N are how much similar if to shift starting point N on m positions.

ICF is defined on all set of integral numbers but as cycle shift on k leads us to an initial contour, the ICF is periodic with phase k.

Let's discover the magnitude having the maximum norm among values of ICF:

$$\tau_{max} = max\left(\frac{\tau(m)}{|\Gamma||N|}\right), \qquad m = 0, \dots, k-1 \qquad (4.2)\ [64]$$

It is clear that τ_{max} is a measure of similarity of two contours, invariant to transposition, scaling, rotation and starting point shift. Figure 4.6 shows rotation angles applied to test ICF is invariant to transposition, scaling, rotation and starting point shift.

a) θ =0 b) θ =π c) θ =π/2 d) θ =3/2π

Figure 4.6: *Rotation Applied to Hand Contours*

b. Hand Position in Relation to Each Other and to Other Body Parts

Hand shape identification, is not enough for recognition of larger sets of signs. Another important aspect is hand position relative to other body parts. This involves checking the position of hands relative to each other (are they near or are they touching and in which direction), to the head or to the chest and many more. Hand Locations can be classified in one of three ways: neutral space, primary locations or secondary locations [65]. A hand is classified to be in neutral space when it is not located on or near the body. Most forms of signs with the hand in neutral space take place in the center. However, there are situations in which the hand may be positioned relatively higher or lower, as well as to the left or right of the person's body. Primary locations are distinguished by a hand being on or near the body. Secondary locations are defined by the hands begin near to each other or touching. To ease our recognition process, we use eight signing region. Signing regions are open according to hand position to its placement relative to the body. In addition, we classify sign performed based on the distance from camera and body as sign performed close to body, away mid from body and away from body. Figure 4.7 illustrates most important signing area of EthSL. Based on skeletal joint provided by Kinect we state useful hand position relative with respect to body joints which are described in Table 4.2. Table 4.1 describes the terminologies used for hand position description.

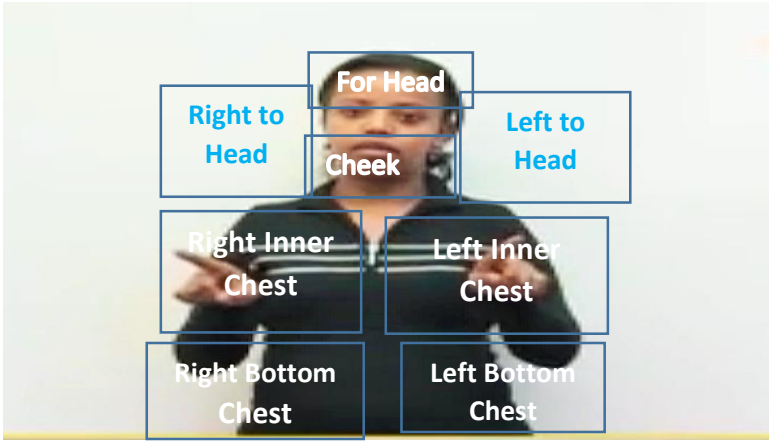


Figure 4.7: *Selected Signing Regions*

Table 4.1: *Terminology Used for Hand position Description*

Left Hand	Right Hand	X	Y	Z	Hand Joint	Shoulder Left Joint	Shoulder Right Joint	Elbow Joint	Spine Joint	Head Joint
L	R	x	y	z	h	sl	sr	e	sp	hd

Table 4.2: *Hand Position Relative with Respect to Selected Skeleton Joints*

Hand, hand joint, axis	**Description**
(Rhx, Rhy, Rhz)	Right hand with hand skeleton joint with reference to X, Y and Z axis
(Rslx, Rsly, Rslz)	Right hand with Shoulder Left skeleton joint with reference to X, Y and Z axis
(Rsrx, Rsry, Rsrz)	Right hand with Shoulder Right skeleton joint with reference to X, Y and Z axis
(Rex, Rey, Rez)	Right hand with elbow skeleton joint with reference to X, Y and Z axis
(Rspx, Rspy, Rspz)	Right hand with Spine skeleton joint with reference to X, Y and Z axis
(Rhdx, Rhdy, Rhdz)	Right hand with Head skeleton joint with reference to X, Y and Z axis
(Lhx, Lhy, Lhz)	Left hand with hand skeleton joint with reference to X, Y and Z axis
(Lslx, Lsly, Lslz)	Left hand with wrist Shoulder Left joint with reference to X, Y and Z axis
(Lsrx, Lsry, Lsrz)	Left hand with wrist Shoulder Right joint with reference to X, Y and Z axis
(Lex, Ley, Lez)	Left hand with elbow skeleton joint with reference to X, Y and Z axis
(Lspx, Lspy, Lspz)	Left hand with Spine skeleton joint with reference to X, Y and Z axis
(Lhdx, Lhdy, Lhdz)	Left hand with Head skeleton joint with reference to X, Y and Z axis

c. Hand Movement (Trajectory) analysis

Movement also conveys valid meaning in sign language. Some of these movements are arcs, straight lines and wavy patterns [2]. Among these movements, hand movements take the major role of sign language. Eye movements and torso movement (movement around the chest) may also be used along with hand movement [6]. The movements may be single, double or repetitive. We should notice these movements during signing because missing them may lead to misunderstanding of signs. Frequency of movements shows frequency of location. For example, a sign for "see" is frequently performed, it means that someone is staring at the

place indicated by the sign. Frequency also shows whether the noun is singular or plural or the distinction between noun and verb [13].

EthSL manual sign consists of sequence of postures connected with motion over a time period. In any sign language recognition, major challenge is the speed of signer which, may vary in time domain for same sign from person to person or for the same person. To overcome this challenge, DTW distance measure was used to find similarity between two motion sequences. It calculates an optimal match between two given sequences of feature vectors which allows for stretched and compressed section of the sequence. It is a well-known method which has been used in various applications such as, signature analysis and speech recognition. So, motion of the hand was tracked and motion descriptor was created with the help of skeleton viewer of Kinect. Initial position of the signer hand was first set for translation invariance at the time of recognition. Motion descriptor in the form of (x, y, z (depth)) values for required skeleton joint information, was taken for each hand. Accordingly, pattern was created in the form of temporal sequence (x1, y1, d1)...(xn , yn , dn) for n number of video frames. Frame rate was considered as 30f/s. Here, each (xi , yi , di) represents the pixel coordinate and depth information of joint in each frame. Single sample training was required for each sign and was stored in database. The algorithm 4.1 and 4.2 are presented for EthSL hand movement recognition using DTW. Algorithm 4.1 is used to store single pattern per sign into database mentioned as Plist. Algorithm 4.2 is used to test unknown sign using DTW distance measure with stored pattern.

Algorithm 4.1: Hand Motion Recognition Training algorithm using DTW

```
// Training is for 1 to N sign gestures and stored into PatternList Plist

PatternListPlist=∑_{i=0}^{N} Plentry i; N number of pattern stored into list
  procedure TrainGesture
     PatternList Plist;          //Plist is a list of trained patterns.
     for  i=1  to N do
           j=GetSkeletonJoints();
     ∀frames                     //For all video frames process following steps
     for  k=1  to j do
          Feature Vectork = Extract Feature(k).
           end for
    AddInPatternlist(Plist,FeatureVector,label)
     end for
         return Plist;
  end procedure
```

Algorithm 4.2: Hand Motion Recognition using DTW

. θ1 is the Maximum distance between the last observations of each train sequence with current test sequence

. θ2 is the Maximum DTW distance between a test example and a train sequence being classified

```
 procedure RecognizeGesture(PatternList Plist,
InputTestSequence)
        minDist  = 9999;
        classification  = ”UNKNOWN”;
        for  i=1  to N do          // Number of trained gesture
pattern in Plist
      for  j=1  to M do     // Number of features of each pattern
         if
          (CalculatedTestPositionDistance(InputTestSeque
          nce,Plist)< θ1)     then
                 d = DT W (I nputT estSequence, P listi )
                      end if
               end for
// /calculate distance  using DTW
             if d < minDist  then
                   minDist = d;
                   classification = true;
             end if
       end for
       return (minDist <θ2? classification: ”UNKNOWN”)
 end procedure
```

d. Facial Expression Recognition

Facial expression has important role in sign language recognition The Microsoft Face Tracking Software Development Kit for Kinect for Windows (Face Tracking SDK), together with the Kinect for Windows Software Development Kit (Kinect for Windows SDK), enables us to create applications that can track human faces in real time. Face Tracking SDK contains a face tracking engine, which can analyze the input from the Kinect camera, it can detect the head pose and face features depending on the points that can be tracked, and generate an information to the application in real time. As an example, this information can be used in tracking person’s head position. The Face Tracking SDK, tracks the 87 2D points, and 13 additional points that belong to the corners of the mouth, the center of each eye, the center of the nose, and for the bounding box around the head**,** Figure 4.8 [29] shows the tracked points. The 87 points are:

- 16 points for the eyes (0-15,8 for the left eye and 8 for the right eye).
- 20 points for the brows (16-35,10 for the left brow and 10 for the right brow).
- 12 points for nose (36-47).
- 20 points for the lips (48-67,12 for the exterior lips, 8 for the interior lips)
- 19 points for the cheek (68-86).

These points are returned in an array, and are defined in the coordinate space of the RGB image (640 x 480resolution) returned by the Kinect sensor

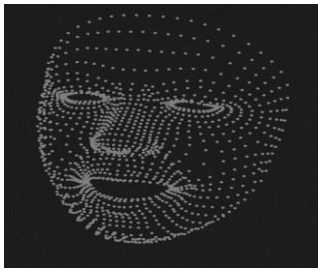


Figure 4.8: *Facial Points Provided by Kinect*

Points location is the method necessary for the feature extraction. Eyebrows, eyes, nose and mouth are the regions which are influenced to the six basic expressions namely “happy”, “sad”, “angry”, “surprised”, “neutral” and “disgusting” of the face. In addition, nose region can be ignored because of its minimal influence on outlet emotions. From this reason, we select fourteen points location including two points on inner eyebrows, two points on middle eyebrows, two points on outer eyebrows, two points on inner eyes, two points on outer eyes and four points on mouth. These points are provided by Kinect as:

LefteyeOutercorner, LefteyeMidbottom, RighteyeOutercorner, RighteyeMidbottom, LefteyebrowOuter, LefteyebrowCenter ,RighteyebrowOuter, RighteyebrowCenter, MouthLeftcorner, MouthRightcorner, MouthUpperlipMidtop, MouthLowerlipMidbottom.

Figure 4.9 depicts the selected fourteen facial points.

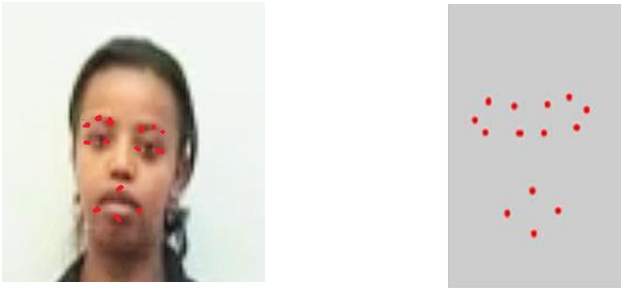


Figure 4.9: *Selected Facial Points*

Once we get necessary point locations we can develop a rule. Table 4.3 shows an illustrative list of rules developed for emotion recognition. Now, Euclidean Distance between the points is to be calculated [59]:

$$D(x,y)= v ((x2-x1)^2+ (y2-y1)^2) \quad (4.3)$$

If we use simple if- then rule, the Euclidean distance would be helpful in that. For example, if R [2] increases and R [3] and R [4] decreases then the expression will be 'happy'. If R[1] ,R [7] and R [8] increases resulting eyebrow to be raised and mouth is open expression will be 'Surprise'. Based on this rule descriptor we can distinguish four users' facial expression namely 'happy', 'sad', 'surprise' and 'neutral', which are most important in sign language recognition. Table 4.3 shows an illustrative list of rules developed for facial expression recognition.

Table 4.3: *Facial Expression Descriptor Rules*

Rule id	**Rule Description**	**points involved**
R[1]	width of the mouth	MouthLeftcorner, MouthRightcorner
[R2]	length of the mouth	MouthUpperlipMidtop, MouthLowerlipMidbottom
[R3]	length of the left eye	LefteyeOutercorners
[R4]	length of the right eye	RighteyeOutercorner
[R5]	distance between both eyebrows	LefteyebrowOuters, RighteyebrowOuters
[R6]	centre of left eyebrow to left side of the left eyebrow	LefteyebrowCenters, LefteyebrowOuters
[R7]	centre of left eyebrow to right side of the left eyebrow	LefteyebrowCenters, RighteyebrowCenters LefteyebrowOuters
[R8]	centre of right eyebrow to left side of the right eyebrow	LefteyebrowCenters, RighteyebrowCenters LefteyebrowOuters
[R9]	centre of right eyebrow to right side of the right eyebrow	LefteyebrowCenters, RighteyebrowCenters LefteyebrowOuters

4.4 Movement Epenthesis Handling and Gesture Segmentation

For continuous sign language recognition, the main issues are how to handle the movement epenthesis (i.e., transition movements between the end of the preceding sign and the start of the following sign) and segmenting each word from sentence by detecting start and end gestures. In speech signal, silence period between two words are detected for a reliable speech parsing. Similarly, for robust continuous sign language recognition majority of direct segmentation approaches exploit inter- sign pauses in stream of hand gestures to demarcate word boundaries. Starting and ending boundary point of gestures is detected whenever the hand pauses during gesturing. Rest and pause positions by identifying points where the velocity dropped below a preset threshold. Velocity of signing hand v_t is estimated as:

$$v_t = p_{t+1} - p_t , \quad \text{where } p_t = (x_t , y_t , z_t) \text{ is the 3-D position at time t.}$$

Based on Movement-Hold model [55] starting and ending boundary point of gestures is detected whenever the hand pauses during gesturing. Movements are defined as those segments during which some aspect of the signer's configuration changes, such as a change in handshape, a hand movement, or a change in hand orientation. Holds are defined as those segments during which all aspects of the signer's configuration remain stationary; that is, the hands remain stationary for a brief period of time. Based on this, we propose to detect movement epenthesis and distinguish it from an actual sign word by observing the motion of the hand between Holds in the input hand motion video. However, sometimes there is no extra movement if a gesture ends in the same position or pose at which the next gesture begins. In this case, we consider only changes on hand shapes to detect holds. A single word gesture can have multiple holds on starting, intermediate and final signing locations. In this case the Hold durations of the intermediate position is fast compared to holds used to mark the end and start position of the sign word. So we empirically defined 0.5 second (if 15 frames remain alike out of 30fps Kinect video stream) to be the minimum Holds duration for start and end of gesture and 0.33 seconds (if 10 frames remain alike out of 30fps of Kinect video stream) for intermediate positions. Generally, once we detect boundary point of gesture based on Holds we can handle movement epenthesis as it was mentioned earlier in Figure 2.7, by allowing for the possibility for ME to exist when no good matching can be found.

4.5 Language Model

Among difficulties for Sign Language interpretation, is to recognize each and every word with grammar and convert it into sentence is really a challenging task. So, simple solution with inverted indexing concept is presented to reorder topic-comment pattern and the subject topic pattern of EthSL sentence to subject-object-verb sentence pattern of spoken Amahric language. The solution is divided into two parts i) creation of index table and ii) searching of possible sentence based on input keyword using index table. Continuous sentence recognition for any Sign Language has major challenges for vision based system which is discussed earlier. The algorithm considers only important recognized keywords of a sentence and interpreted meaningful sentence using inverted indexing concept. The inverted indexing concept is mostly used in the field of information retrieval such as, for search engine indexing algorithm as well as in bioinformatics and in general, document search algorithm. The proposed work identifies this method for use of sentence interpretation in Sign Language. Recognized signs as keywords and possible sentence which are described in Table 4.4. were given as an input to Algorithm 4.3. Each keyword was stored along with sentence number list in index table. For each insertion of keyword, index table was scanned and corresponding entry was updated. The detailed steps are given in algorithm 4.3 which creates index Table 4.5. Once the index table is created, next step is finding the possible sentence based on input keywords.

Table 4.4: *Input Keywords and Possible sentence for EthSL Sentence Algorithm*

Sentence No.	**Keywords (EthSL signs)**	**Possible EthSL (sentence list)**
S1	ትክክል (Correct)	ትክክል ነው ! Correct! or ይህ ትክክል ነው ! This is correct.
S2	(Wrong)	ትክክል አይደለም ! Wrong! or ይህ ትክክል አይደለም:: This is

S3	አንተ/አንች (you), ትክክል (correct)	አንተ ትክክል ነክ:: /አንች ትክክል ነሽ::
S4	እንደምን (how),አንተ/አንች （YOU ）	እንደምን ነክ? እንደምን ነሽ? How are you ?
S5	ስንት (how), ዕድሜ (age), አንተ/አንች (you)	ዕድሜክ ስንት ነው? /ዕድሜሽ ስንት ነው? How old are you ?
S6	ወፍ (Bird), መብረር(FLY)	ወፏ እየበረረች ነው:: The Bird is flying.
S7	አንተ/አንች (YOU), መሄድ(LEAVE/Fly)	መቼ ነው የምትሄደው?/ መቼ ነው የምትሄጅው? when are you

Table 4.5: *Index Table Generated for Table 4.4 Inputs*

Keyword EthSL word	**Sentence No**
correct	S1, S 3
Wrong	S2
You	S3, S4, S5, S7
How	S4, S5
age	S5
Fly	S6, S7

Algorithm 4.3 Index Table Construction for EthSL Sentence Interpretation Algorithm

- Input: Input data set
- $E = \{Sn, Kl, S\}$ //Each entry in dataset D consists of sentence no, keyword list and sentence

  $D=\sum_{i=0}^{N} Ei$; input dataset for N sentences
- Output: Index Table
- IndexTableEntry = K, Sl ; Each index table entry consists of keyword and sentence list

IT=$\sum_{j=0}^{M} ITEi$; M entries in index table. SKS is stored keyword list into index table

```
procedure ConstructIndex(Dataset  D)
SKS=φ;  index,i,j=0;        // Initially stored keyword list is empty
for  i=0  to N do           // N no. of entries in D
for  each K in keyword      // Construct index table for each
if K  SK S then
Index= GetIndex  (K);       // If keyword is already present get index of it.
IT Index : Sl = ∪ Di .Sn; //Append new sentence no. with existing sentence list.
 else
I Tj . K = K; //If new keyword, then insert keyword and sentence no. with
// new entry in index table.
            I Tj .Sl = Di .Sn;
                        SK Sj = K;
                        j++;
                    end if
                end for
        end for
        return IT;
endprocedure
```

Algorithm 4.4 Sentence Interpretation Algorithm using Index Table

- Input: Index Table IT, Input keywords;
- Output: Output set O; set of sentence no. list and final statement no. FSn

```
procedure        Sentence        Interpretation
(IT,InputKewords )
        Index,i=0;
        for   each K in Input Keywords do
              I ndex = GetI ndex(K );
              Oi = I TI ndex .Sl;
              i++;
        end for
    FSn = (O0  ∩ O1 ..∩On );  //By applying intersection operation find correct possible sentence no.
        Display (FSn );
  end procedure
```

To illustrate the proposed algorithm, let us take one example. Consider K is the list of recognized input sign words given below. After finding sentence number. against each input keyword, intersection operation was applied on each sentence number. list and final sentence no. was retrieved and corresponding sentence was displayed. Algorithm 4.4 gives output F Sn as a sentence number.

suppose K= how,age,you -Input recognized signs as a keyword

$IT = \{S4 \cup S5\}, \{S5\}, \{S3 \cup S4 \cup S5 \cup S7\}$. Output of Algorithm 4.3

$F\,S_n = \{S4,S5\} \cap \{S5\} \cap \{S3,S4,S5,S7\}$. Output of Algorithm 4. 4

$F\,S_n = \{S5,$ ዕድሜክ ስንት ነው?/ዕድሜሽ ስንት ነው?How old are you?$\}$.

Possible sentence is displayed

4.6 Database Building from Combined Feature

Based on values we get from handshape analysis, postion modules, facial expression, hand movement (trajectory analysis) we can build a database from features fused from those modules. Table 4.6 summarizes expected values and states from the above modules.

Table 4.6: *Expected output Values from Modules*

Modules	Values/states
Hand shape	Thumbs up/down, hand, hand open/closed, hand pointing, hand victory, UNKNOWN
Hand Movement (trajectory analysis)	Measures the similarity distance of training and test gestures movement based on DTW
Facial Expression	Mouth Open, Mouth Closed, Eye Open, Eye Closed, Happy, Neutral, Eye Brow raised, and surprised
Position (location)	Recognizes the position of hands with respect to the eight signing region

- A Sign can involve only a single dominant hand (mostly Right hand) or can involve both hands.
- If facial expressions are needed to describe a sign their states should be included. However, if their states are not crucial for defining a sign they can be ignored.

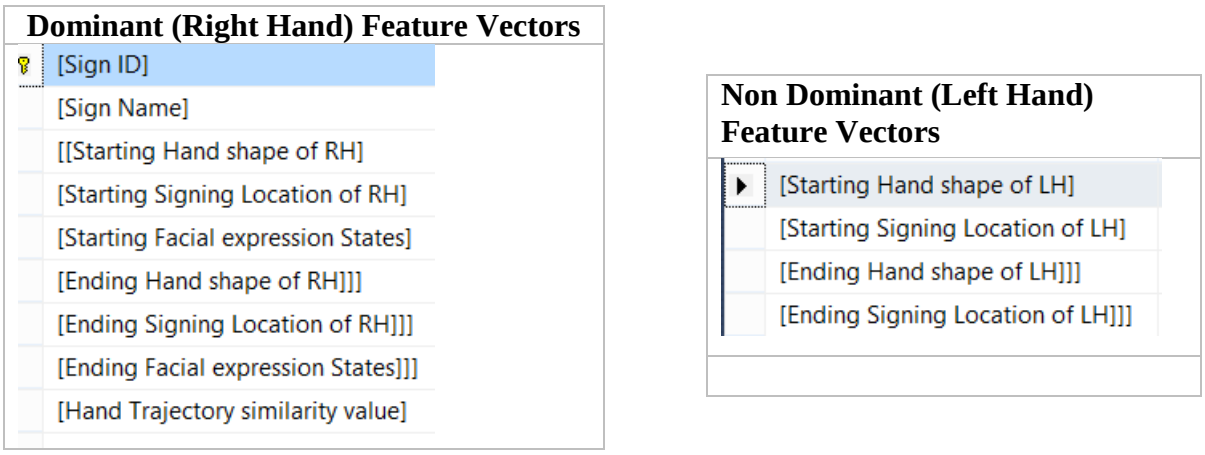


Figure 4.10: *Sign Word Feature Attributes*

Based on the above features vectors a sign can have at least 7 feature descriptor or at most 11 feature descriptor. Figure 4.10 depicts sign word feature attributes.

4.7 Random Forest Training

A random forest classification model consists of several trees. Increasing the number of trees in a random forest helps increase classification accuracy up to a certain number of trees after which the increase in classification accuracy is insignificant [63]. We will select the optimal number of trees through experimentation. We defined the training set as D = {(X1 , Y1), ..., (Xn , Yn)}. Here, (X1, ..., Xn) corresponds to the uniform-length feature vector representing the gesture or non-gesture, and (Y1, ..., Yn) represents their corresponding class labels. Decision trees **t**(x, φk) are constructed until they are fully grown, that is without pruning. Here x is an input vector and φk is a random vector used to generate a bootstrap sample of objects from the training set D. That is, for each decision tree, n samples are chosen randomly with replacement from D.

Let d be the dimensionality of the feature vector of the inputs. At each internal node of the tree, m features are selected randomly from the available d, such that m < d.

We chose: $m=\sqrt{d}$ from the m chosen features. Algorithm 4.5 [63] depicts the steps needed for training random forest.

Algorithm 4.5: Pseudocode for training a Random Forest

Data: N training gesture samples, each represented by a feature vector of length d

Result: Random Forest gesture classification model

```
for each tree in the forest do
   Sample N data points with replacement;

   for each internal node in the tree do
      Randomly sample m attributes (m = √d);

      Select the feature from this randomly selected

      set that provides the most information gain;

      Split the input data points using the selected
      feature, creating left and right
      children nodes;

   end

   return Fully grown decision tree

end
return Random Forest
```

We trained and saved a random forest classification model based on the features that we extracted.

4.8 Sign Recognition

The task during testing is to use our trained random forest model to determine the segmentation of gestures in a continuous stream of input data from Kinect© and accurately classify the segmented gesture. Unlike training videos, test data do not contain information about where gestures start and end. Therefore, we detect the start and end of gesture by analyzing the change on hand shape, hand position and hand trajectory. This is done by comparing the changes and similarities on the previous hand information per 10 frames (333 milliseconds). It is reasonable to use information from previous group of 10 frames, because hand shape cannot change so fast in each frame (1 frame = 33.3 milliseconds). The duration of pauses between hold region pause, word boundaries and sentence boundaries are different. Researchers found that the longest pauses occurred at sentence boundaries and longer pauses are observed in word boundaries than mid hold region pauses [64,65].

Accordingly, if similarity in hand information is observed for 10 frames then Movement Hold pause is detected and the information on this interval is tracked. But, if this similarity persists 20 frames then the information within these frames are tracked and start or end of sign word is marked. Additionally, if hand information is similar for at least 30 frames, end of one sentence is detected. Algorithm 4.6 illustrates the recognition process of a continuous EthSL.

A Sign *Word* W has at least *initial segment* $\mathbf{S_i}$ and *final Segment* $\mathbf{S_f}$ as mentioned earlier in the previous paragraph we also have empirically defined pause duration for *word, and sentence* as $\mathbf{T_w}$, and $\mathbf{T_{sen}}$ respectively. We also defined changes in hand information that includes signing regions as $\mathbf{Ch_{all}}$ and changes in hand information within the same signing regions as $\mathbf{Ch_{wor}}$. All changes are recognized by hand shape, facial expression, hand movement and hand position modules discussed earlier.

Algorithm 4.6 Recognition process of a continuous EthSL

We initially define j=0 , j is used to identify words within a sentence.

While ($\mathbf{T_{sen}}$ is observed) //until sentence boundary is detected

If $\mathbf{Ch_{all}}$ or $\mathbf{Ch_{wor}}$ observed for $\mathbf{T_w}$

$\mathbf{S_{i\,(j)}}$ information within $\mathbf{T_w}$ is tracked and compared with initial word feature information in the database. Then similar sign words that have the same initial word features are filtered.

If Chall or Chwor observed is for Tw

$\mathbf{S_{f\,(j)}}$ information within $\mathbf{T_w}$ is tracked and compared with final word feature information in the database. Then similar sign words that have the same final word features are filtered.

Then random forest classifies the sign word based on information of $\mathbf{S_{i(j)}}$, and $\mathbf{S_{f(j)}}$.

If sign word is identified the system waits to detect $\mathbf{Ch_{all}}$ or Chwor to identify the start of the next sign word.

Then it loops to step 2 by incrementing value of j.

If Chall or Chwor is observed for Tsen,

the system detects sentence boundary and applies language model on recognized sign words to construct Amharic sentence text

CHAPTER 5: EXPERIMENT

In this Chapter, we will discuss the implementation of continuous sign recognition system for Ethiopian Sign Language. The tools used, the database, the training and testing processes and data used to implement the prototype will be discussed.

5.1 Data Collection

The input video was captured with a kinect camera which has frame rate of 30 frames per second. Information about RGB images, skeletal joint position as well as 3D depth information of the user are collected. The position of the camera is stationed in front of the signer and made static. This position of the camera enabled us to get all the body parts which are necessary to have meaningful signs. This position has equal distance of the signers from the camera which enabled us to get equivalent size of the signers in the video. The videos are captured in Xef format in the data collection, three signers were involved. Sixty sign words have been chosen to be part of the research. Out of these sixty signs, ten simple Amharic sentences are constructed. Each signer is expected to perform a sign fifteen times. We used ten of them for training purpose and the rest five for testing purpose. Table 5.1 and Table 5.2 shows the words that we collect and sentences used for testing purpose, respectively. The full list of collected EthSL Words is shown on Appendix C.

Table 5.1 : *Samples of Collected EthSL Words*

ID	Amharic Words	
1	ስራ/ሰራተኛ	worker/work
2	ዶክተር	Doctor
3	ምን	what
4	ስንት	how many/much
5	ያለ	without
6	ማን	who
7	የትኛው	which
8	የእኔ	mine
9	ማልቀስ	cry

Table 5.2: *Simple Amharic Sentences Constructed from EthSL Words*

No:	Amharic Sentences	
1	እኔ በጣም እወዳለሁ።	I love you so much.
2	ወደፊት ዶክተር መሆን እፈልጋለሁ።	I want to be a doctor in the future.
3	ዕድሜህ ስንት ነው?	How old are you?
4	ስጋ መብላት እወ ድሃለው።	I like to eat meat.
5	አዲስ አበባ በጣም ሰፊ ከተማ ናት።	Addis Ababa is a very large city.
6	ዛሬ ወደ ሀዋሳ እሄዳለሁ።	I am going to Hawasa today.
7	የተውለድኩት አሰላ ከተማ ውስጥ ነው።	I was born in Asella city.
8	ዛሬ ምሳ ምን እንበላ?	What shall we have at lunch time?
9	ታቦቱ ዛሬ ይወጣል::	The Arc of Covenant will be displayed today.
10	ወፏ እየበረረች ነው።	The bird is flying.

Database

Back end spread sheet is prepared to store every feature data that is produced in the recognition process. Microsoft SQL Server is chosen to implement the database since it has very important features suitable for the development of various applications in addition to its simplicity and requiring low cost performance.

The database contains two tables to organize the data appropriately. The first table is used to store features that uniquely identify each EthSL words used in this study. Figure 5.1 shows features used to describe each sign words. The detail of the Sign word features such as Starting Hand Shape of RH, Ending Hand Shape of RH, Starting Signing Location of RH, Ending Signing Location of RH, Starting Hand Shape of LH, Ending Hand Shape of LH, Starting Signing Location of LH, Ending Signing Location of LH, Sign Performed, Starting Facial Expression State, Ending Facial Expression State are obtained from feature extraction system component which is discussed earlier in the previous chapter. which signer the sign is conducted (Signer Id), whether the sign is used.

ID	Amharic Words	English	Starting Hand Shape of RH	Ending Hand Shape of RH	Starting Signingn Location Of RH	Ending Signing Location Of RH	Starting Hand Shape of LH	Ending Hand Shape of LH	Starting Signingn Location Of LH	Ending Signingn Location Of LH	Sign Performed	Starting Facial Expression State	Ending Facial Expression State
1	ማልቀስ	cry	Victory	Victory	For Head	Cheek	Not required	Not required	Silent	Silent	Close to body	Eye Closed	Eye Closed
2	በጣም	very	Open	Closed	Right Inner Chest	Left Inner Chest	Open	Closed	Left Inner Chest	Right Inner Chest	Close to body	Mouth Closed	Mouth Closed
3	መሆን	to be	Open	Open	Right Inner Chest	Right Inner Chest	Open	Open	Left Inner Chest	Left Inner Chest	Mid from body	Mouth Closed	Mouth closed
4	መፈለግ	want	Open	Closed	Right Bottom Chest	Right Bottom Chest	Open	Closed	Left Bottom Chest	Left Bottom Chest	Close to body	Mouth Closed	Mouth closed
5	እኔ	I	Pointing/Closed	Pointing/Closed	Right Bottom Chest	Right Inner Chest	Not required	Not required	Silent	Silent	Close to body	Not required	Not require
6	ወደፊት	future	Open	Open	Right Head	Head	Not required	Not required	Silent	Silent	Away from body	Not required	Not require
7	መውደድ/ፍቅር	Love	Thumbs Up/Closed	Thumbs Up/Closed	Right Bottom Chest	Right Inner Chest	Thumbs Up/Closed	Thumbs Up/Closed	Left Bottom Chest	Right Inner Chest	Closed to body	Mouth Open	Mouth Open

Figure 5.1: *Features Describing Sample EthSL Words*

5.2 Prototype Development

- **Tools**

The following tools are used to develop the prototype.

- Microsoft Visual Studio 2013: to develop the front end component of the prototype.
- Microsoft Kinect SDK: we used all necessary Kinect SDK classes which were helpful in implementing the logics, and classification algorithms in Ethiopian Sign Language Recognition.
- Microsoft SQL Server: To implement the back end of the prototype. We used Microsoft SQL server to create the database and to handle data manipulation activities.

- **User Interface**

The prototype of the system has a user interface which show the video of the selected word and get the recognized output text. The user interface is shown below.

Figure 5.2: *User Interface*

5.3 Test Result

In this section, the accuracy of the system is analyzed. We divided the evaluation process into two parts. The first part is evaluation based on signers. Since we found differences on the result of the recognition for each signer, we showed the recognition ratio for signers. The result of the recognition for a signer is calculated with the following equation.

$$\text{Recognition ratio of Signer} = \frac{No\ of\ Recognized\ Signs\ by\ signner}{No\ of\ test\ Signs\ of\ signer} \quad (5.1)\ [9]$$

The other part of the evaluation process is evaluation of the overall system performance. The recognition ratio of the system was calculated by mixing the test results for all signers. The following equation is used to calculate the recognition ratio.

$$\text{Recognition Accuracy} = \frac{No\ of\ Recognized\ Signs}{No\ of\ test\ Signs} \quad (5.2)\ [9]$$

We measure the performance of the system according to the accuracy of the translation on two level; sentence, and word. For testing the performance of the system at sentence level, we took ten simple Amharic sentences. We use sixty words to measure system accuracy at word level. The recognition result is calculated using Equations 5.1 and 5.2 are shown in Table 5.3. Table 5.4 show the results found at sentence level.

Table 5.3: *Recognition Result by Signers at Word Level*

	Total Number of Test Signs	Number of Recognized Signs	Recognition Percentage
Signer 1	60	54	90%
Signer 2	60	48	80%
Signer 3	60	50	83.3%
Total	180	152	84.4%

Table 5.4: *Recognition result by Signers at Sentence Level*

	Total Number of Test sentences	Number of Recognized Sentences	Recognition Percentage
Signer 1	10	7	70%
Signer 2	10	5	50%
Signer 3	10	6	60%
Total	30	18	60%

5.4 Discussion

The test results show as there are wrong translations on sentence level. This error happened due to extra movement created between the end of one word to the start of the next word. The system wrongly considers such transitional gestures as a correct word rather than ignoring such movement. The other issue that we encounter during experimentation is the wrong recognition result for those signs which have similarity. This problem is encountered when different handshapes are used to describe the sign other than the five common handshapes used in our system. In addition, as shown in Table 6.1 and Table 6.3 there is a result variation between signers. This happened because of the way the signers follow to conduct the sign. Some signers conduct the signing process in such a way that someone can see clearly the components of the sign. In this type of signers, we can see the movement direction, the speed variation, the Movements and the Holds of the sign. In addition, these type of signers can show us the shapes of the hand while they are signing. The other type of signers are those who conduct the signing process in a way that someone could not identify the components easily. Usually, these type of signers are those who use the language in their everyday activities. The system recognition performance will decline for these type of signers. To reduce these types of problem we need to use more training signs which are conducted by variety of signers. Since there is no study conducted on continuous EthSl recognition yet, the proposed work is not compared with related EthSL work. Comparing the proposed work with other SL is not feasible and appropriate due to difference in morphology, phonology and grammar of each SL.

CHAPTER 6: CONCLUSION and FUTURE WORK

6.1 Conclusion

This thesis is proposed and designed to develop a method that is capable of recognizing and translating continuous EthSL based on the data acquired from Microsoft® Kinect. The ultimate goal is to facilitate a way of communication between hearing-impaired individuals and other community members. Many researchers have been conducting researches on the translation of Amharic text to EthSL. Similarly, few studies were conducted on isolated recognition of EthSL. However, this is the first step on the development of continuous EthSL recognition.

The system accepts simple stream of video from Kinect and to Amharic text. The performance of the system was measured in two categories at: word, and sentence level

and we got system accuracy 84.4% at word level and 60% at sentence level. Finally, we conclude that in study area of Ethiopian sign language, developing such system that is capable of translating continuous sign language to Amharic text has great impact. In addition, has great contribution in the development of common signing language in a country as well as to filling of communication gap between hearing and hearing impaired people

6.2 Contribution of this Work

The proposed methodology uses the Microsoft Kinect to get depth images of body joints from user. After extracting manual and non-manual sign features, it stores them in a gesture dictionary. A random forest algorithm is used to match gestures with those stored in the dictionary and later converted to Amharic text. We also handled ME and sign segmentation by analyzing movement pauses. In addition, we proposed language model providing simple solution with inverted indexing concept to reorder topic-comment pattern and the subject topic pattern of EthSL sentence to subject-object-verb sentence pattern of spoken Amahric language.

6.2 Future Work

This study has its own contribution on the Ethiopian sign language in filling the communication gap between hearing and hearing impaired people. But to make the system and model complete and plays great roles in the development of sign language communication, we recommend future works need to be conducted by taking the following into consideration:

- The language model should be extended beyond re ordering the grammatical structure of EthSL to spoken Amharic language.
- Currently, the system is a one-way communication, EthSL to Amahric text, and to make the system complete it needs to be a two-way communication
- Additional handshapes should be recognized to make the system more accurate and precise.
- The system can be extended by recognizing gestures from multiple users of EthSL.
- The system could also be extended by translating EthSL to other Sign languages of other country.

References

[1] Ethiopian National Association for Deaf, "BERTAT", Yearly Magazine, Ethiopia, 1997.

[2] Masresha Tadesse, "Automatic translations of Amharic text to Ethiopian Sign Language", Unpublished Master's Thesis, Addis Ababa University (AAU), 2010.

[3] Abadi Tsegay and Kumudha Raimond, "Offline Candidate Hand Gesture Selection and Trajectory Determination for Continuous Ethiopian Sign Language", Journal of Theoretical and Applied Information Technology, Vol. 36, No.1, 2012

[4] Dagnachew Feleke, "Machine Translation System for Amharic Text to Ethiopian Sign Language", Unpublished Master's Thesis, AAU, 2011.

[5] Oya Aran, "Vision Based Sign Language Recognition", PhD Thesis, Institute for Graduate Studies in Science and Engineering, Bogazici University, 2008.

[6] Legesse Zerubabel, "Ethiopian Finger Spelling Classification", Unpublished Master's Thesis, AAU, 2008.

[7] Daniel Martinez Capilla, "Sign Language Translator Using Microsoft Kinect", Master's Thesis, University of Tennessee, 2012.

[8] Daniel Zegeye, "Amharic Sentence to Ethiopian Sign Language Translator", Master's Thesis, AAU, 2014.

[9] Tefera Gimbi, "Recognition of Isolated Signs in Ethiopian Sign Language", Master's Thesis, AAU, 2014.

[10] Rashmi D. Kyatanvar and P.R. Futane "Comparative Study of Sign Language Recognition Systems", International Journal of Scientific and Research Publications, Volume 2, Issue 6, 2012.

[11] Hubbert Wassner, "Development D'une API De Reconnaissance De Gestes Pour Le Capteur Kinect", 2011: Available at "http://www.professeurs.esiea.fr/ wassner/ ?2011/ 05/06/325-kinect-reseau-neurone-reconnaissance-des-gestes", last accessed on December 02, 2014.

[12] Sandjaja, I. and N. Marcos, "Sign language number recognition", in the proceeding of 5th International Joint Conference on INC, IMS and IDC, pp: 1503-1508, 2009.

[13] Liddell, S., and Johnson, R. “American Sign Language: The phonological base”, Gallaudet University Press, Washington. DC, 1989.

[14] Yoseph F. Admasu, and K. Raimond, “Ethiopian Sign Language Recognition Using ArtificiaNeural Network,” 10th International Conference on Intelligent Systems Design and Applications (ISDA), IEEE, 2010.

[15] Ethiopian Sign Language Community, “the deaf people of Ethiopia, people and language detail report”, 2005.

[16] Eyasu Hailu, “Sign language news”, Addis Ababa University, 2009.

[17] Birtat Magazin, monthly newspaper no. 7, Addis Ababa: ENAD, 2003

[18] P. A. Harling and A. D. N. Edwards, “Hand tension as a gesture segmentation cue. In Proceedings of Gesture Workshop”, York, UK, Mar 1996.

[19] J. Kramer and L. Leifer. The “talking glove”: An expressive and receptive “verbal” communication aid for the deaf, deaf-blind, and non-vocal. In Proceedings of Conference on Computer Technology/Special Education/Rehabilitation, pages 335–340, California, Northridge, Oct 1987.

[20] E. Ohira, H. Sagawa, T. Sakiyama, and M. Ohki. “A segmentation method for sign language recognition”. IEICE Transactions on Information and Systems, E78-D(1):49–57, 1995.

[21] H. Sagawa and M. Takeuchi. “A method for recognizing a sequence of sign language words represented in a Japanese sign language sentence”. In Proceedings of International Conference on Automatic Face and Gesture Recognition, pages 434–439, Grenoble, France, 2000.

[22] H. Sagawa, M. Takeuchi, and M. Ohki, “Methods to describe and recognize sign language based on gesture components represented by symbols and numerical values. Knowledge-Based Systems”, 1998.

[23] C. Wang, W. Gao, and Z. Xuan, “A real-time large vocabulary continuous recognition system for Chinese sign language”, China,2001.

[24] R.-H. Liang and M. Ouhyoung, “A real-time continuous gesture recognition system for sign language”, In Proceedings of International Conference on Automatic Face and Gesture Recognition”, 1998.

[25] W. Kong and S. Ranganath, "Towards subject independent continuous sign language recognition: A segment and merge approach,", Pattern Recognition,vol.47, no. 3, 2014.

[26] Ruiduo Yang, Sudeep Sarkar, and Barbara Loeding, "Enhanced Level Building Algorithm for the Movement Epenthesis Problem in Sign Language Recognition", IEEE Computer Society, 2007

[27] ElenaSanchez-Nielsen, L. Anton-Canalís and M. Hernandez-Tejera, "Hand gesture recognition for human-machine interaction", 2003.

[28] Brashear, H., T. Starner, P. Lukowicz and H. Junker, "Using multiple sensors for mobile sign language recognition", Proceedings of the 7th IEEE International Symposium on Wearable Computers, 2003.

[29] Zhang, L., J.C. Hsieh and J. Wang," A kinect-based golf swing classification system using HMM and neuro-fuzzy", Proceedings of the International Conference on Computer Science and Information Processing, 2012.

[30] R. Grzeszcuk, G. Bradski, M. H. Chu, and J.-Y. Bouguet, "Stereo based gesture recognition invariant to 3d pose and lighting", In Proceedings of Conference on Computer Vision and Pattern Recognition., volume 1, pages 826–833. IEEE, 2000.

[31] S. Hadfield and R. Bowden, "Generalized pose estimation using depth", In Trends and Topics in Computer Vision, pages 312–325, 2012.

[32] Y. Hamada, N. Shimada, and Y. Shirai, "Hand shape estimation under complex backgrounds for sign language recognition", In Proceedings of Sixth International Conference on Automatic Face and Gesture Recognition., pages 589–594, 2004.

[33] Kinect for windows sdk beta launch. http://channel9.msdn.com/Events/KinectSDK/BetaLaunch, June 2011. Last checked July 24 2016 12:07.

[34] C. Eisler. Starting february 1, 2012: Use the power of kinect for windows to change the world. http://blogs.msdn.com/b/kinectforwindows/archive/2012/01/09/kinect-for-windows-commercial-program-announced.aspx,. Last checked July 24 2016 12:16.

[35] Microsoft Corporation. Human Interface Guidelines, 1.8 edition, 2013.

[36] Microsoft Corporation. Kinect for Windows SDK Managed Code Reference.

[37] L.R. Rabiner. "A tutorial on hidden markov models and selected applications in speech recognition". Proceedings of the IEEE, 1989.

[38] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on, 1992.

[39] Thad Starner, Alex Pentland, and Joshua Weaver. "Real-time american sign language recognition using desk and wearable computer based video". IEEE PAMI, , 1998

[40] M. Müller. "Dynamic time warping. Information retrieval for music and motion", 2007.

[41] Li, H. and M. Greenspan, "Continuous time-varying gesture segmentation by dynamic time warping of compound gesture models", Queen's University, Kingston, Canada, 2007.

[42] Capilla, D.M., "Sign language translator using microsoft kinect XBOX 360TM". Department of Electrical Engineering and Computer Science, University of Tennessee, 2012

[43] Z. Halim and G. Abbas, "A Kinect-Based Sign Language Hand Gesture Recognition System for Hearing- and Speech-Impaired: A Pilot Study of Pakistani Sign Language," Assistive Technology: The Official Journal of RESNA, 2014,

[44] Dengsheng Zhang and Guojun Lu, "Review of shape representation and description techniques", Pattern Recognition, 2004.

[45] William J. Rucklidge, "Efficiently locating objects using the hausdorff distance. Int. J. Comput. Vision", 1997.

[46] Dengsheng Zhang and Guojun Lu. A comparative study of fourier descriptors for shape representation and retrieval. In Proc. of 5th Asian Conference on Computer Vision, 2002.

[47] Omar Al-Jarrah and Alaa Halawani, "Recognition of gestures in arabic sign language using neuro-fuzzy systems", 2001.

[48] M. Handouyahia, D. Ziou, and S. Wang, "Sign language recognition using moment based size functions", Proc. Int'l Conf. Vision Interface, 1999

[49] C. Uras and A. Verri. Sign language recognition: An application of the theory of size functions. 6th British Machine Vision Conference, 1995.

[50] S.X. Liao and M. Pawlak. On image analysis by moments. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 18(3):254 –266, 1996

[51] M. R. Teague. “Image analysis via the general theory of moments”, Journal of the Optical Society of America (1917-1983), 1980.

[52] William C. Stokoe. “Sign Language Structure: An Outline of the Visual Communication System of the American Deaf. Studies in Linguistics”, Occasional Papers 8. Linstok Press, 1960. Revised 1978.

[53] Ulrich Agris, Jorg Zieren, Ulrich Canzler, Britta Bauer and Karl Kraiss, “ Recent developments in visual sign language recognition ” , Institute of Man–Machine Interaction, RWTH Aachen University, 2007.

[54] Siegmund Prillwitz, Regina Leven, Heiko Zienert, Thomas Hanke, and Jan Henning. HamNoSys. Version 2.0; “Hamburg Notation System for Sign Languages. An introductory guide”, volume 5 of International Studies on Sign Language and Communication of the Deaf, 1989.

[55] Scott K. Liddell and Robert E. Johnson. American Sign Language: The phonological base. Sign Language Studies, 1989.

[56] Geoffrey R. Coulter, “Current Issues in ASL Phonology”, volume 3 of Phonetics and Phonology. Academic Press, Inc., San Diego, CA, 1993.

[57] T. Starner and A. Pentland "Visual Recognition of American Sign Language using Hidden Markov Models", Proc. Intl. Workshop on Automatic Face and Gesture Recognition, 1995.

[58] T. Starner, J. Weaver, and A. Pentland, “A Wearable Computer Based American Sign Language Recognizer” Wearable Computers, 1997. Digest of Papers., First International Symposium, 1997

[59] Zafrulla, Z.; Brashear, H.; Starner, T.; Hamilton, H.; Presti, P. American sign language recognition with the kinect. In Proceedings of the International Conference on Multimodal Interfaces, Alicante, Spain, ,2011.

[60] Ren, Z.; Yuan, J.; Meng, J.; Zhang, Z. “Robust part-based hand gesture recognition using Kinect sensor”. IEEE Trans. Multimed. 2013,

[61] Chai, X.; Li, G.; Lin, Y.; Xu, Z.; Tang, Y.; Chen, X.; Zhou, M. “Sign Language Recognition and Translation with Kinect. In Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition”, Shanghai, China, 2013.

[62] Moreira Almeida, S.; Guimarães, F.; Arturo Ramírez, J. “Extraction in Brazilian Sign Language Recognition based on phonological structure and using RGB-D sensors”. Expert System. 2014

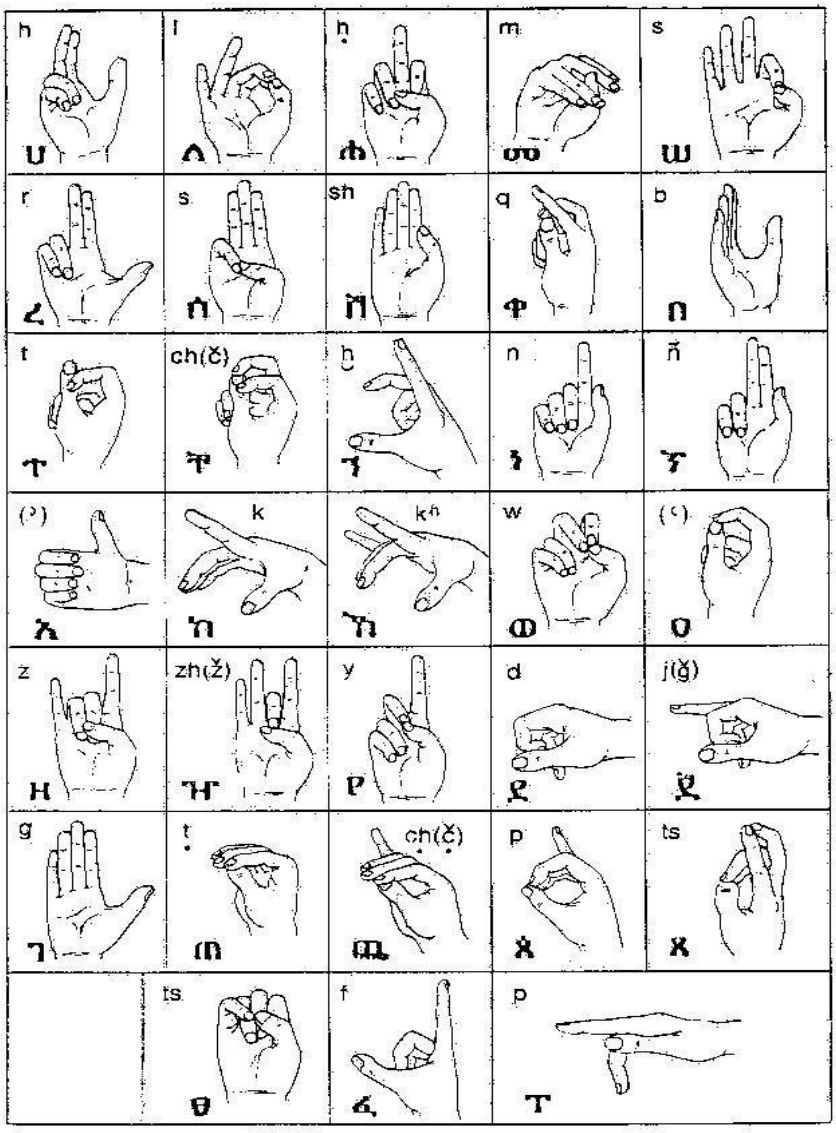
[63] Joshi Ajjen Das, “A Random Forest Approach to Segmenting and Classifying Gestures”, Published Master’s Thesis Boston University, 2014.

[64] . Diane Brentor, Joshua Falk, George Wolford, “The Acquistaion of Prosody in American Sign Language”, Lingussitic Society of America, 2015

[65] Martha E. Tyrone , Hosung Nam , Elliot altzman , Gaurav Mathur , Louis Goldstein,“Prosody and Movement in American Sign Language: A Task-Dynamics Approach”, Department of Linguistics, University of Southern California, Los Angeles, CA, USA, 2014.

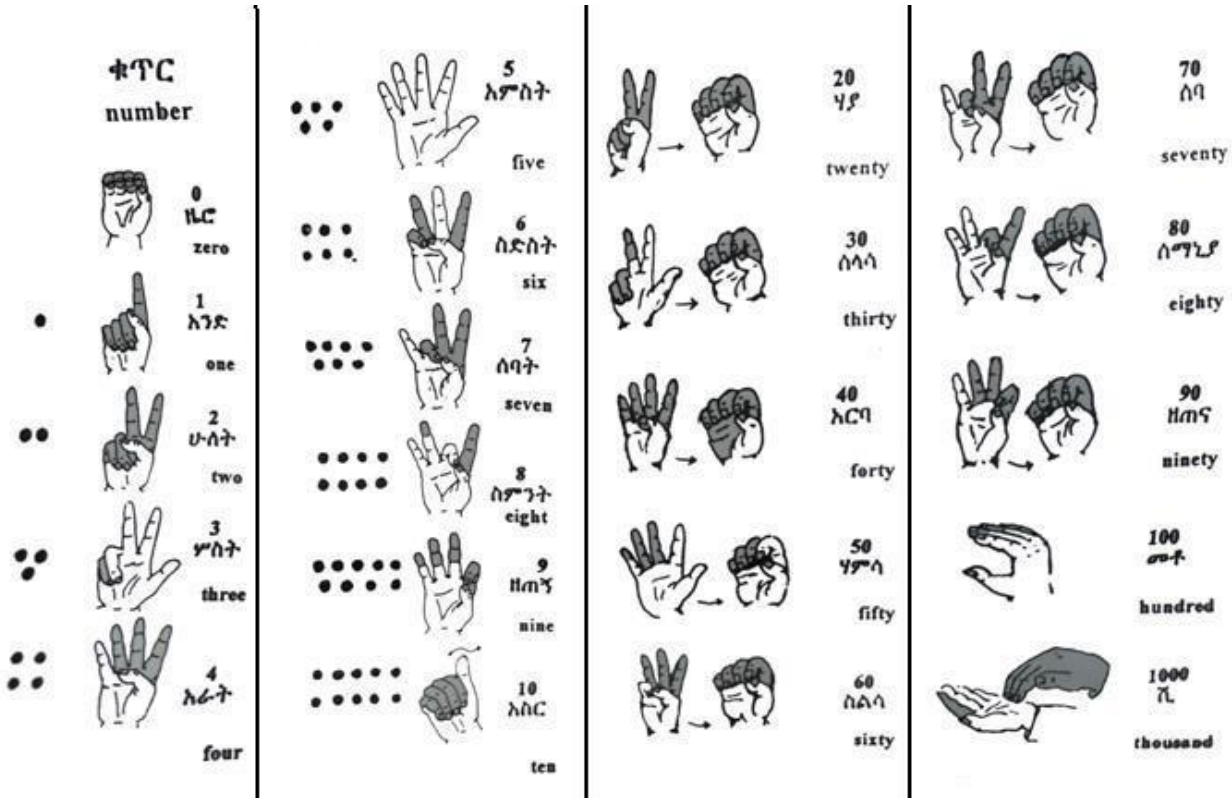
Appendix – A

EthSL Finger Spelling Representation [1]



Appendix – B

Numbers in EthSL [1]



Appendix – C

Sample of Collected EthSL Words

ID	Amharic Words	
1	ሰራ/ሰራተኛ	worker/work
2	ዶክተር	Doctor
3	ምን	what
4	ስንት	how many/much
5	ያለ	without
6	ማን	who
7	የትኛው	which
8	የእኔ	mine
9	ማልቀስ	cry
10	ማሸነፍ	win
11	ረጅም	tall/long
12	ሩቅ	far
13	ርካሽ	cheap
14	ሰፊ	wide
15	ቀላል	light
16	ስም	name
17	ቀጭን	thin
18	በጣም	very
19	አጭር	short
20	ወፍራም	fat
21	ረሃብ	hunger
22	ምሳ	lunch
23	መብላት	eat
24	ስጋ	meat

25	ጩቤ	dagger
26	መሄድ	go
27	መሆን	to be
28	መፈለግ	want
29	እኔ	I
30	አንተ	you
31	አንቺ	you
32	እሱ	he
33	እሷ	she
34	አጎት	uncle
35	ረመዳን	Remedan
36	ታቦት	Arc of covenant
37	እስላም	Islam
38	ማታ	night
39	ቀጠሮ	appointment
40	ተነገ ወዲያ	the day after tomorrow
41	አሁን	now
42	አልፎ አልፎ	sometimes
43	ከሰዓት ቡሃላ	Afternoon
44	ወደፊት	future
45	ዛሬ	today
46	ጧት	morning
47	ሀገር	country
48	አሰላ	Asella
49	ሀዋሳ	Hawasa
50	አዲስ አበባ	Addis Ababa
51	ኢትዮጵያ	Ethiopia
52	ከተማ	city

53	እኛ	we
54	እንጀራ	Enjera
55	መውደድ/ፍቅር	Love
56	ማልቀስ	cry
57	መቼ	when
58	እድሜ/ዓመት	age/Year
59	ትክክል/ልክ	correct/right
60	ማመስገን/ምስጋና	thanks/praise

Appendix – D

Sample Facial Recognition Module Source Code

```
//------------------------------------------------------------------------------

#include "stdafx.h"
#include <strsafe.h>
#include "resource.h"
#include "FaceBasics.h"
#include <algorithm>
#include <vector>

// define the face frame features required to be computed by this application
static const DWORD c_FaceFrameFeatures =
    FaceFrameFeatures::FaceFrameFeatures_BoundingBoxInColorSpace
    | FaceFrameFeatures::FaceFrameFeatures_PointsInColorSpace
    | FaceFrameFeatures::FaceFrameFeatures_RotationOrientation
    | FaceFrameFeatures::FaceFrameFeatures_Happy
    | FaceFrameFeatures::FaceFrameFeatures_RightEyeClosed
    | FaceFrameFeatures::FaceFrameFeatures_LeftEyeClosed
    | FaceFrameFeatures::FaceFrameFeatures_MouthOpen
    | FaceFrameFeatures::FaceFrameFeatures_MouthMoved
    | FaceFrameFeatures::FaceFrameFeatures_LookingAway
    | FaceFrameFeatures::FaceFrameFeatures_FaceEngagement;

/// <summary>
/// Entry point for the application
/// </summary>
/// <param name="hInstance">handle to the application instance</param>
/// <param name="hPrevInstance">always 0</param>
/// <param name="lpCmdLine">command line arguments</param>
/// <param name="nCmdShow">whether to display minimized, maximized, or
normally</param>
/// <returns>status</returns>
int APIENTRY wWinMain(_In_ HINSTANCE hInstance, _In_opt_ HINSTANCE hPrevInstance,
_In_ LPWSTR lpCmdLine, _In_ int nCmdShow)
{
    UNREFERENCED_PARAMETER(hPrevInstance);
    UNREFERENCED_PARAMETER(lpCmdLine);

    CFaceBasics application;
    application.Run(hInstance, nCmdShow);
}

/// <summary>
/// Constructor
/// </summary>
CFaceBasics::CFaceBasics() :
    m_hWnd(NULL),
    m_nStartTime(0),
    m_nLastCounter(0),
    m_nFramesSinceUpdate(0),
    m_fFreq(0),
    m_nNextStatusTime(0),
    m_pKinectSensor(nullptr),
    m_pColorFrameReader(nullptr),
    m_pD2DFactory(nullptr),
```

```
    m_pDrawDataStreams(nullptr),
    m_pColorRGBX(nullptr),
    m_pBodyFrameReader(nullptr)
{
    LARGE_INTEGER qpf = {0};
    if (QueryPerformanceFrequency(&qpf))
    {
        m_fFreq = double(qpf.QuadPart);
    }

    for (int i = 0; i < BODY_COUNT; i++)
    {
        m_pFaceFrameSources[i] = nullptr;
        m_pFaceFrameReaders[i] = nullptr;
              m_pHDFaceFrameSources[i] = nullptr;
              m_pHDFaceFrameReaders[i] = nullptr;
    }

    // create heap storage for color pixel data in RGBX format
    m_pColorRGBX = new RGBQUAD[cColorWidth * cColorHeight];
}


/// <summary>
/// Destructor
/// </summary>
CFaceBasics::~CFaceBasics()
{
    // clean up Direct2D renderer
    if (m_pDrawDataStreams)
    {
        delete m_pDrawDataStreams;
        m_pDrawDataStreams = nullptr;
    }

    if (m_pColorRGBX)
    {
        delete [] m_pColorRGBX;
        m_pColorRGBX = nullptr;
    }

    // clean up Direct2D
    SafeRelease(m_pD2DFactory);

    // done with face sources and readers
    for (int i = 0; i < BODY_COUNT; i++)
    {
        SafeRelease(m_pFaceFrameSources[i]);
        SafeRelease(m_pFaceFrameReaders[i]);
              SafeRelease(m_pHDFaceFrameSources[i]);
              SafeRelease(m_pHDFaceFrameReaders[i]);
    }

    // done with body frame reader
    SafeRelease(m_pBodyFrameReader);

    // done with color frame reader
    SafeRelease(m_pColorFrameReader);
```

```
    // close the Kinect Sensor
    if (m_pKinectSensor)
    {
        m_pKinectSensor->Close();
    }

    SafeRelease(m_pKinectSensor);
}

/// <summary>
/// Creates the main window and begins processing
/// </summary>
/// <param name="hInstance">handle to the application instance</param>
/// <param name="nCmdShow">whether to display minimized, maximized, or
normally</param>
int CFaceBasics::Run(HINSTANCE hInstance, int nCmdShow)
{
    MSG       msg = {0};
    WNDCLASS  wc;

    // Dialog custom window class
    ZeroMemory(&wc, sizeof(wc));
    wc.style         = CS_HREDRAW | CS_VREDRAW;
    wc.cbWndExtra    = DLGWINDOWEXTRA;
    wc.hCursor       = LoadCursorW(NULL, IDC_ARROW);
    wc.hIcon         = LoadIconW(hInstance, MAKEINTRESOURCE(IDI_APP));
    wc.lpfnWndProc   = DefDlgProcW;
    wc.lpszClassName = L"HDFaceBasicsAppDlgWndClass";

    if (!RegisterClassW(&wc))
    {
        return 0;
    }

    // Create main application window
    HWND hWndApp = CreateDialogParamW(
        NULL,
        MAKEINTRESOURCE(IDD_APP),
        NULL,
        (DLGPROC)CFaceBasics::MessageRouter,
        reinterpret_cast<LPARAM>(this));

    // Show window
    ShowWindow(hWndApp, nCmdShow);

    // Main message loop
    while (WM_QUIT != msg.message)
    {
        Update();

        while (PeekMessageW(&msg, NULL, 0, 0, PM_REMOVE))
        {
            // If a dialog message will be taken care of by the dialog proc
            if (hWndApp && IsDialogMessageW(hWndApp, &msg))
            {
                continue;
            }
```

```
            TranslateMessage(&msg);
            DispatchMessageW(&msg);
        }
    }

    return static_cast<int>(msg.wParam);
}

/// <summary>
/// Handles window messages, passes most to the class instance to handle
/// </summary>
/// <param name="hWnd">window message is for</param>
/// <param name="uMsg">message</param>
/// <param name="wParam">message data</param>
/// <param name="lParam">additional message data</param>
/// <returns>result of message processing</returns>
LRESULT CALLBACK CFaceBasics::MessageRouter(HWND hWnd, UINT uMsg, WPARAM wParam,
LPARAM lParam)
{
    CFaceBasics* pThis = nullptr;

    if (WM_INITDIALOG == uMsg)
    {
        pThis = reinterpret_cast<CFaceBasics*>(lParam);
        SetWindowLongPtr(hWnd, GWLP_USERDATA, reinterpret_cast<LONG_PTR>(pThis));
    }
    else
    {
        pThis = reinterpret_cast<CFaceBasics*>(::GetWindowLongPtr(hWnd,
GWLP_USERDATA));
    }

    if (pThis)
    {
        return pThis->DlgProc(hWnd, uMsg, wParam, lParam);
    }

    return 0;
}

/// <summary>
/// Handle windows messages for the class instance
/// </summary>
/// <param name="hWnd">window message is for</param>
/// <param name="uMsg">message</param>
/// <param name="wParam">message data</param>
/// <param name="lParam">additional message data</param>
/// <returns>result of message processing</returns>
LRESULT CALLBACK CFaceBasics::DlgProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM
lParam)
{
    UNREFERENCED_PARAMETER(wParam);
    UNREFERENCED_PARAMETER(lParam);

    switch (message)
    {
    case WM_INITDIALOG:
```

```
        {
            // Bind application window handle
            m_hWnd = hWnd;

            // Init Direct2D
            D2D1CreateFactory(D2D1_FACTORY_TYPE_SINGLE_THREADED, &m_pD2DFactory);

            // Create and initialize a new Direct2D image renderer (take a look at
ImageRenderer.h)
            // We'll use this to draw the data we receive from the Kinect to the
screen
            m_pDrawDataStreams = new ImageRenderer();
            HRESULT hr = m_pDrawDataStreams->Initialize(GetDlgItem(m_hWnd,
IDC_VIDEOVIEW), m_pD2DFactory, cColorWidth, cColorHeight, cColorWidth *
sizeof(RGBQUAD));
            if (FAILED(hr))
            {
                SetStatusMessage(L"Failed to initialize the Direct2D draw device.",
10000, true);
            }

            // Get and initialize the default Kinect sensor
            InitializeDefaultSensor();
        }
        break;

        // If the titlebar X is clicked, destroy app
    case WM_CLOSE:
        DestroyWindow(hWnd);
        break;

    case WM_DESTROY:
        // Quit the main message pump
        PostQuitMessage(0);
        break;
    }

    return FALSE;
}

/// <summary>
/// Initializes the default Kinect sensor
/// </summary>
/// <returns>S_OK on success else the failure code</returns>
HRESULT CFaceBasics::InitializeDefaultSensor()
{
    HRESULT hr;

    hr = GetDefaultKinectSensor(&m_pKinectSensor);
    if (FAILED(hr))
    {
        return hr;
    }

    if (m_pKinectSensor)
    {
        // Initialize Kinect and get color, body and face readers
        IColorFrameSource* pColorFrameSource = nullptr;
```

```
        IBodyFrameSource* pBodyFrameSource = nullptr;

        hr = m_pKinectSensor->Open();

        if (SUCCEEDED(hr))
        {
            hr = m_pKinectSensor->get_ColorFrameSource(&pColorFrameSource);
        }

        if (SUCCEEDED(hr))
        {
            hr = pColorFrameSource->OpenReader(&m_pColorFrameReader);
        }

        if (SUCCEEDED(hr))
        {
            hr = m_pKinectSensor->get_BodyFrameSource(&pBodyFrameSource);
        }

        if (SUCCEEDED(hr))
        {
            hr = pBodyFrameSource->OpenReader(&m_pBodyFrameReader);
        }

        if (SUCCEEDED(hr))
        {
            // create a face frame source + reader to track each body in the fov
            for (int i = 0; i < BODY_COUNT; i++)
            {
                if (SUCCEEDED(hr))
                {
                    // create the face frame source by specifying the required face
frame features
                    //hr = CreateFaceFrameSource(m_pKinectSensor, 0,
c_FaceFrameFeatures, &m_pFaceFrameSources[i]);
                                    hr =
CreateHighDefinitionFaceFrameSource(m_pKinectSensor, &m_pHDFaceFrameSources[i]);

                                }
                if (SUCCEEDED(hr))
                {
                    // open the corresponding reader
       //             hr = m_pFaceFrameSources[i]-
>OpenReader(&m_pFaceFrameReaders[i]);
                                    hr = m_pHDFaceFrameSources[i]-
>OpenReader(&m_pHDFaceFrameReaders[i]);
                }
            }
        }

        SafeRelease(pColorFrameSource);
        SafeRelease(pBodyFrameSource);
    }

    if (!m_pKinectSensor || FAILED(hr))
    {
        SetStatusMessage(L"No ready Kinect found!", 10000, true);
        return E_FAIL;
```

```
    }

    return hr;
}

/// <summary>
/// Main processing function
/// </summary>
void CFaceBasics::Update()
{
    if (!m_pColorFrameReader || !m_pBodyFrameReader)
    {
        return;
    }

    IColorFrame* pColorFrame = nullptr;
    HRESULT hr = m_pColorFrameReader->AcquireLatestFrame(&pColorFrame);

    if (SUCCEEDED(hr))
    {
        INT64 nTime = 0;
        IFrameDescription* pFrameDescription = nullptr;
        int nWidth = 0;
        int nHeight = 0;
        ColorImageFormat imageFormat = ColorImageFormat_None;
        UINT nBufferSize = 0;
        RGBQUAD *pBuffer = nullptr;

        hr = pColorFrame->get_RelativeTime(&nTime);

        if (SUCCEEDED(hr))
        {
            hr = pColorFrame->get_FrameDescription(&pFrameDescription);
        }

        if (SUCCEEDED(hr))
        {
            hr = pFrameDescription->get_Width(&nWidth);
        }

        if (SUCCEEDED(hr))
        {
            hr = pFrameDescription->get_Height(&nHeight);
        }

        if (SUCCEEDED(hr))
        {
            hr = pColorFrame->get_RawColorImageFormat(&imageFormat);
        }

        if (SUCCEEDED(hr))
        {
            if (imageFormat == ColorImageFormat_Bgra)
            {
                hr = pColorFrame->AccessRawUnderlyingBuffer(&nBufferSize,
reinterpret_cast<BYTE**>(&pBuffer));
            }
            else if (m_pColorRGBX)
```

```
            {
                pBuffer = m_pColorRGBX;
                nBufferSize = cColorWidth * cColorHeight * sizeof(RGBQUAD);
                hr = pColorFrame->CopyConvertedFrameDataToArray(nBufferSize,
reinterpret_cast<BYTE*>(pBuffer), ColorImageFormat_Bgra);
            }
            else
            {
                hr = E_FAIL;
            }
        }

        if (SUCCEEDED(hr))
        {
            DrawStreams(nTime, pBuffer, nWidth, nHeight);
        }

        SafeRelease(pFrameDescription);
    }

    SafeRelease(pColorFrame);
}

/// <summary>
/// Renders the color and face streams
/// </summary>
/// <param name="nTime">timestamp of frame</param>
/// <param name="pBuffer">pointer to frame data</param>
/// <param name="nWidth">width (in pixels) of input image data</param>
/// <param name="nHeight">height (in pixels) of input image data</param>
void CFaceBasics::DrawStreams(INT64 nTime, RGBQUAD* pBuffer, int nWidth, int nHeight)
{
    if (m_hWnd)
    {
        HRESULT hr;
        hr = m_pDrawDataStreams->BeginDrawing();

        if (SUCCEEDED(hr))
        {
            // Make sure we've received valid color data
            if (pBuffer && (nWidth == cColorWidth) && (nHeight == cColorHeight))
            {
                // Draw the data with Direct2D
                hr = m_pDrawDataStreams-
>DrawBackground(reinterpret_cast<BYTE*>(pBuffer), cColorWidth * cColorHeight *
sizeof(RGBQUAD));
            }
            else
            {
                // Recieved invalid data, stop drawing
                hr = E_INVALIDARG;
            }

            if (SUCCEEDED(hr))
            {
                // begin processing the face frames
                ProcessFaces();
            }
```

```
            m_pDrawDataStreams->EndDrawing();
        }

        if (!m_nStartTime)
        {
            m_nStartTime = nTime;
        }

        double fps = 0.0;

        LARGE_INTEGER qpcNow = {0};
        if (m_fFreq)
        {
            if (QueryPerformanceCounter(&qpcNow))
            {
                if (m_nLastCounter)
                {
                    m_nFramesSinceUpdate++;
                    fps = m_fFreq * m_nFramesSinceUpdate / double(qpcNow.QuadPart -
m_nLastCounter);
                }
            }
        }

        WCHAR szStatusMessage[64];
        StringCchPrintf(szStatusMessage, _countof(szStatusMessage), L" FPS = %0.2f
Time = %I64d", fps, (nTime - m_nStartTime));

        if (SetStatusMessage(szStatusMessage, 1000, false))
        {
            m_nLastCounter = qpcNow.QuadPart;
            m_nFramesSinceUpdate = 0;
        }
    }
}

/// <summary>
/// Processes new face frames
/// </summary>
void CFaceBasics::ProcessFaces()
{
    HRESULT hr;
    IBody* ppBodies[BODY_COUNT] = {0};
    bool bHaveBodyData = SUCCEEDED( UpdateBodyData(ppBodies) );

    // iterate through each hd face reader
    for (int iFace = 0; iFace < BODY_COUNT; ++iFace)
    {
        // retrieve the latest face frame from this reader
              IHighDefinitionFaceFrame * pHDFaceFrame = nullptr;
              hr = m_pHDFaceFrameReaders[iFace]->AcquireLatestFrame(&pHDFaceFrame);

              TRACE(L"HD Face frame acquired: %d", hr);

        BOOLEAN bFaceTracked = false;
        if (SUCCEEDED(hr) && nullptr != pHDFaceFrame)
        {
```

```
            // check if a valid face is tracked in this face frame
            hr = pHDFaceFrame->get_IsTrackingIdValid(&bFaceTracked);

                    TRACE(L"HD Face tracking valid: %d", bFaceTracked );

        }

            if (bFaceTracked)
            {
                            TRACE(L"HD Face: %d is tracked", iFace);
                            IFaceAlignment* pFaceAlignment = nullptr;
                            HR(CreateFaceAlignment(&pFaceAlignment));
                            HR(pHDFaceFrame-
>GetAndRefreshFaceAlignmentResult(pFaceAlignment));

                            //vector<float> animationUnits;

                            float* pAnimationUnits = new
float[FaceShapeAnimations_Count];

                            HR(pFaceAlignment-
>GetAnimationUnits(FaceShapeAnimations_Count, pAnimationUnits));

                            for (int i = 0; i < FaceShapeAnimations_Count; i++)
                            {
                                    FaceShapeAnimations faceAnim =
(FaceShapeAnimations)i;

                                    if (faceAnim ==
FaceShapeAnimations::FaceShapeAnimations_LefteyebrowLowerer)
                                    {
                                            auto leftEyeBrow =
pAnimationUnits[faceAnim];
                                            TRACE(L"Left Eye Brow: %f", leftEyeBrow);
                                    }
                            }

                            float* pDeformations = new
float[FaceShapeDeformations_Count];
                            IFaceModel * pFaceModel = nullptr;
                            HR(CreateFaceModel(1.0, FaceShapeDeformations_Count,
pDeformations, &pFaceModel));


                            //pFaceModel-
>CalculateVerticesForAlignment(pFaceAlignment, )

                            HR(pFaceModel-
>GetFaceShapeDeformations(FaceShapeDeformations_Count, pDeformations));

                            /*for (int j = 0; j < FaceShapeDeformations_Count; j++)
                            {
                                    TRACE(L"Face Deform: %d = Value %f", j,
pDeformations[j]);
                            }
*/
                RectI faceBox = {0};
```

```
                        //HighDetailFacePoints::
                        PointF hdPoints[36];

                        PointF facePoints[FacePointType::FacePointType_Count];
                Vector4 faceRotation;

                        //auto highDetail =
HighDetailFacePoints_RighteyeOutercorner;


                        HR(    pFaceAlignment->get_FaceBoundingBox(&faceBox));
                        TRACE(L"Face Box Top: %d\tBottom: %d\tLeft: %d\tRight:
%d", faceBox.Top, faceBox.Bottom, faceBox.Left, faceBox.Right);

                        HR(pFaceAlignment->get_FaceOrientation(&faceRotation));
                        TRACE(L"Face Rotation
X:\t%f\t\tY:\t%f\t\tZ:\t%f\t\tW:\t%f", faceRotation.x, faceRotation.y,
faceRotation.z, faceRotation.w);

                        CameraSpacePoint headPivot;
                        HR(pFaceAlignment->get_HeadPivotPoint(&headPivot));
                        TRACE(L"Head Pivot X:\t%f\t\tY:\t%f\t\tZ:\t%f",
headPivot.X, headPivot.Y, headPivot.Z);

                        FaceAlignmentQuality faceQuality;
                        HR(pFaceAlignment->get_Quality(&faceQuality));

                        if (FaceAlignmentQuality::FaceAlignmentQuality_High ==
faceQuality)
                        {
                                TRACE(L"Face Quality: HIGH");
                        }
                        else if (FaceAlignmentQuality::FaceAlignmentQuality_Low
== faceQuality)
                        {
                                TRACE(L"Face Quality: LOW");
                        }
                        else
                        {
                                TRACE(L"Face Quality: %d", faceQuality);
                        }

                        DetectionResult
faceProperties[FaceProperty::FaceProperty_Count];

                 // draw face frame results
                m_pDrawDataStreams->DrawFaceFrameResults(iFace, &faceBox, facePoints,
&faceRotation, faceProperties, &headPivot, pAnimationUnits);

                        delete [] pDeformations;
                        delete [] pAnimationUnits;


            }
            else
            {
                // face tracking is not valid - attempt to fix the issue
                // a valid body is required to perform this step
```

```
                if (bHaveBodyData)
                {
                    // check if the corresponding body is tracked
                    // if this is true then update the face frame source to track
this body
                    IBody* pBody = ppBodies[iFace];
                    if (pBody != nullptr)
                    {
                        BOOLEAN bTracked = false;
                        hr = pBody->get_IsTracked(&bTracked);

                        UINT64 bodyTId;
                        if (SUCCEEDED(hr) && bTracked)
                        {
                            // get the tracking ID of this body
                            hr = pBody->get_TrackingId(&bodyTId);
                            if (SUCCEEDED(hr))
                            {
                                // update the face frame source with the tracking ID
                                m_pHDFaceFrameSources[iFace]-
>put_TrackingId(bodyTId);
                                                        /*if (nullptr ==
m_pFaceFrameSources[iFace])
                                                        {

       HR(CreateFaceFrameSource(m_pKinectSensor, bodyTId, c_FaceFrameFeatures,
&m_pFaceFrameSources[iFace]));

       HR(m_pFaceFrameSources[iFace]->OpenReader(&m_pFaceFrameReaders[iFace]));
                                                        }*/
                            }
                        }
                    }
                }
            }

         SafeRelease(pHDFaceFrame);
    }

    if(bHaveBodyData)
    {
        for (int i = 0; i < _countof(ppBodies); ++i)
        {
            SafeRelease(ppBodies[i]);
        }
    }
}

/// <summary>
/// Updates body data
/// </summary>
/// <param name="ppBodies">pointer to the body data storage</param>
/// <returns>indicates success or failure</returns>
HRESULT CFaceBasics::UpdateBodyData(IBody** ppBodies)
{
    HRESULT hr = E_FAIL;
    if (m_pBodyFrameReader != nullptr)
    {
```

```
            IBodyFrame* pBodyFrame = nullptr;
            hr = m_pBodyFrameReader->AcquireLatestFrame(&pBodyFrame);
            if (SUCCEEDED(hr))
            {
                hr = pBodyFrame->GetAndRefreshBodyData(BODY_COUNT, ppBodies);
            }
            SafeRelease(pBodyFrame);
        }
        return hr;
    }

    /// <summary>
    /// Set the status bar message
    /// </summary>
    /// <param name="szMessage">message to display</param>
    /// <param name="showTimeMsec">time in milliseconds to ignore future status
    messages</param>
    /// <param name="bForce">force status update</param>
    /// <returns>success or failure</returns>
    bool CFaceBasics::SetStatusMessage(_In_z_ WCHAR* szMessage, ULONGLONG nShowTimeMsec,
    bool bForce)
    {
        ULONGLONG now = GetTickCount64();

        if (m_hWnd && (bForce || (m_nNextStatusTime <= now)))
        {
            SetDlgItemText(m_hWnd, IDC_STATUS, szMessage);
            m_nNextStatusTime = now + nShowTimeMsec;

            return true;
        }

        return false;
    }
```

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

Declared by:

Name: _____

Signature: _____

Date: _____

Confirmed by advisor:

Name: _____

Signature: _____

Date: _____