ADDIS ABABA INSTITUTE OF TECHNOLOGY

CENTER OF INFORMATION TECHNOLOGY AND SCIENTIFIC COMPUTING

DEPARTMENT OF INFORMATION TECHNOLOGY

Finger Spellings Recognition for Ethiopian Sign Language

Team Members

1. Buruk Sahilu	ATR/7166/07
2. Daniel Moges	ATR/6585/07
3. Natnael Berhanu	ATR/7159/07
4. Yohannes Endale	ATR/0842/07

Advisor: Nebiat Fikru

March 2018

Addis Ababa Institute of Technology

Information Technology and Scientific Computing

Finger Spelling Recognition for Ethiopian Sign Language

This Project documentation submitted in partial fulfillment of the requirements for the Degree of Bachelor of Science in **IT**.

Project Advised by:

Advisor Name: Nebiat Fikru

Name and signature of Members of the examining board:

	Name	Title	Signature	Date
1.	____________________	Advisor	______________	______________
2.	____________________	Chairperson	______________	______________
3.	____________________	Examiner	______________	______________
4.	____________________	Examiner	______________	______________
5.	____________________	Examiner	______________	______________

March 2018

Declaration of Originality

We declare that this project is our original work and has not been presented for a degree in any other university.

Name	Signature	Date
1. Name of Student 1	Buruk Sahilu_______	March 6, 2018_______
2. Name of Student 2	Daniel Moges_ ___	March 6, 2018_______
3. Name of Student 3	Natnael Berhanu _	March 6, 2018_______
4. Name of Student 4	Yohannes Endale _	March 6, 2018_______

This project documentation has been submitted for examination with my approval as university advisor:

Advisor Name: Nebiat Fikru

March 2018

ACKNOWLEDGEMENT

We the development team, would like to express our very great appreciation to the Center of Information Technology and Scientific Computing(ITSC) for enabling us to undertake the first phase of our final project, which is this documentation.

There are a lot of peoples who have helped us to grow this idea into what it is now. So we would like to thank these peoples for their support. And also like to thank Addis Ababa Institute of Technology department of ITSC staff specifically the committee who are going to see this work and give us their view on the project proposal.

ABSTRACT

Humans interact with each other using a natural language channel such as words, writing, or by body language (gesture). As understanding natural language is important, understanding sign language is also very important. The sign language is the basic communication method between people with a hearing disability. People with hearing disability face problems in communicating without a translator. For this reason, the introduction of a system that recognizes sign language would have a significant benefit impact on deaf people's social life. In this paper we have proposed visual Ethiopian sign language recognition system using image processing, computer vision and neural network methodologies, to identify the characteristics of the hand in image taken through web camera. This approach will convert video of daily frequently used full sentence gestures into text.

Table of Contents

List of Figures

List of Tables

ACRONYMS

WHO	World Health Organization
E.C	Ethiopian Calendar
CNN	Convolutional Neural Network
HMM	Hidden Markov Model
SRS	Software Requirement Specification
EthSL	Ethiopian Sign Language
SVM	Supervised Machine Learning
GUI	Graphical User Interface
CNN	Convolutional Neural Network
API	Application Programming Interface
GPL	General Public License
FSRFEthSL	Finger Spelling Recognition for Ethiopian Sign Language

Chapter 1: INTRODUCTION

1.1 Background

One of the many physical disabilities people face is being hearing-impaired. According to WHO [1], over 5% of the world's population (360 million people) has a disabling hearing loss. A person is said to have a disabling hearing loss if s/he has a hearing loss that is greater than 40dB. Hearing loss can result from many causes, some of which are genetic, complications at birth, certain infectious diseases, chronic ear infections, the use of particular drugs, exposure to excessive noise and aging.

In Ethiopia, according to 2007 E.C census, 1.1% of the population (805,492) [2] have disabilities. Among this population, 18.2% (146,913) have hearing disability. From children born with disability, only 2% have access to proper school education [3].

Sign Language

Deaf people use sign language [4] to communicate with others. Sign language is defined as a language which uses manual communication such as hand gestures, facial expressions and movements of arms or body to convey meaning as opposed to acoustically conveyed sound patterns. Wherever deaf people exist, sign languages have developed as a means of communication.

According to Ethnologue [5], there are about 138 sign languages in the world. Each sign language is named after the geographical location it is widely used in. For example, we say American Sign Language to refer to the sign language used in America. A common misconception among people who are new to sign language is to assume that sign languages are named after spoken languages. For instance, there is no Amharic sign language, or Oromiffa Sign language; there is only Ethiopian Sign Language.

Sign language has its own rules, structure and grammar. The following are the basic components of sign language [7].

- Hand shape
- Hand movement
- Orientation
- Location

- Non manual features like facial gestures.

Ethiopian Sign language is originated from American Sign Language with some influence from the Nordic Countries [7, 8]. The language also includes local signs which originated from local deaf schools. The language didn't get the chance to be developed and standardize like spoken languages [9]. 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.

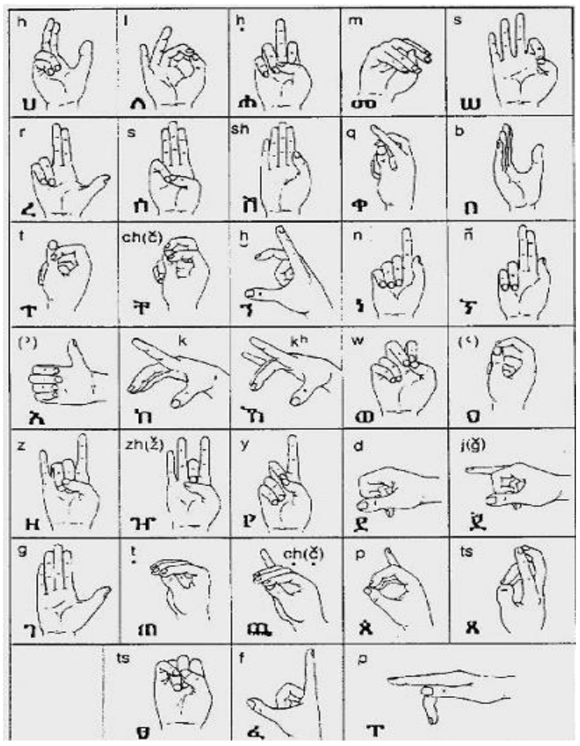


Figure 1 Ethiopian Finger Spelling

Deaf education in Ethiopia began in the 1950s, which was given by foreigners teaching in missionary schools. The Ethiopian sign language has been used at primary school level since 1956 [6] and at higher school levels since 1971. However, its use is limited to small portion of the deaf community.

This is due to the small number of deaf people getting proper school education.

1.2 The Existing System

The work in [11] discusses vision based finger spelling recognition for Ethiopian Sign Language. This work is a continuation of a previous research work [12] which was able to recognize finger spelling for Ethiopian sign language without considering vowels. The work in [11] extends the study for the recognition of Ethiopian finger spelling with their corresponding vowels. The additions of vowels in the finger spelling will add movements to the basic finger shapes of the consonants.

The work used distance based selection technique to select frames from the input videos. After selecting the frames, the next step was preprocessing the selected frames to detect the hand from the other parts of the body. In this process image processing techniques such as converting RGB to Gray scale, adjusting brightness and removing noise with the help of median filter were conducted. Global thresholding was also applied to segment components. Grouping neighborhood was the other technique to select the largest component which was considered as a signed hand. By calculating the distance between the centers of masses of the detected signs in two consecutive frames, it was possible to identify the motion of the hand which was very useful in determining the corresponding vowel of the finger spelling. Finite state automata were the preferred technique for the process of recognition of the finger spelling.

The signers wore white gloves with long sleeved black shirts to segment hands from other parts of the body parts using image processing algorithms. The videos were captured in a constant background. Four signers participated in performing 238 Ethiopian finger spellings. The overall performance of the work was 90.76%.

The system in [14] accepts a video of isolated sign of Ethiopian sign language and generates equivalent text as an output. This system has five main components Skin color detection, Hands and Head segmentation, extracting features from the segmented body parts, Training HMM model using the extracted features and Recognition.

The process of skin detection applied an algorithm from Mahidra [13] and to get more precise segmentation of the hands and the head we used various filtering algorithms and the 8- connected method. Baum Welch algorithm is used for training our HMM model. They trained a separate model for each sign and Viterbi decoding applied in the recognition process. In this research 3 signers participated and each signer performed a sign 20 times. Out of these 20 videos they used 15 for training and the rest 5 video for testing purpose. The system has been tested using the videos which were captured for training purpose. They found overall recognition of 86.9% using the HMMs trained by 8 features whereas we found the overall recognition of 83.5% using the HMMs trained by only 3 features.

1.3. Statement of the Problem

Sign language is a primary tool for the deaf people to communicate with the community around them. But most hearing people do not have the knowledge of sign language. This situation leads to creation of a huge communication gap between the deaf and hearing people. The problems are expressed as follows:

- A communication gap to interact with the hearing people to accomplish their daily tasks. Most of the deaf people have the problem of writing and speaking. This problem is more visible on those deaf people who lost their hearing ability since birth. The inability of writing and speaking widens the communication gap with the hearing people [9].
- Problems that are encountered in the formal education process that includes ○ Lack of sufficient interpreters for each school in which deaf students are enrolled [9, 10].
  ○ The knowledge of the interpreter is limited to translate the subject matter [10].
- The number of schools that provide sign language translation service are very limited [9,10]. 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 deaf patients and the health professional [10].

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

- o There are very limited interpreters [1, 2].
- o There are some issues such as court cases that must be kept secret from the interpreters [2].

Therefore, some technological solutions should be prepared to solve the communication gap between the hearing and deaf people.

1.4 Objective of the Project

1.4.1 General Objective

The general objective of this study is to design a model which can interpret isolated (fingerspelling) signs in Ethiopian Sign Language and give equivalent text to a given sign.

1.4.2 Specific Objective

The specific objectives of the study are

- Study the structure of Ethiopian Sign Language.
- Find suitable existing model to extract a sign from a given video.
- Develop a method to convert the extracted sign to equivalent voice.
- Develop a prototype which demonstrates the translation of the given sign to equivalent text.
- Test the model.

1.5 Proposed System

The proposed System consists of three tasks to be done in real time:

1. Obtaining video of the user signing (input)
2. Classifying each frame in the video to a letter
3. Reconstructing and displaying the most likely word from classification scores (output)

Our system features a pipeline that takes video of a user signing a word as input. We then extract individual frames of the video and generate letter probabilities for each using a CNN with the use of a variety of heuristics, we group the frames based on the character index that each frame is suspected to correspond to. Finally, we use a language model in order to output a likely word to the user.

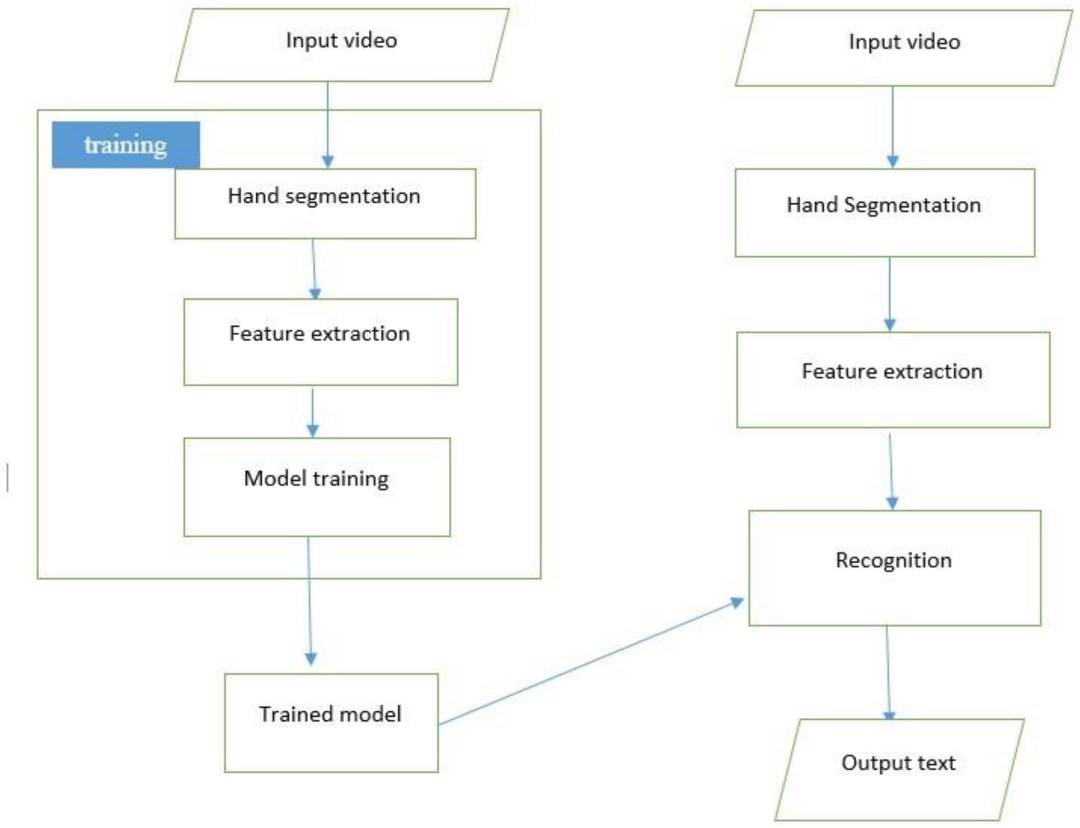


Figure 2 System Architecture

1.6 Feasibility Study

In this section we will try to analyses project's viability and its degree of being easily or conveniently done. There are many challenges on this project and we will try to be prepared for them as much as possible. That way we will not have problem after the project start.

1.6.1 Economic Feasibility

1.6.1.1 **Developmental *cost***

The technical resources used in this process are mostly found freely. For example OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use, TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and also used for machine learning.

1.6.1.2 Operational Cost

There will be additional costs like copying printing and material fulfilment.

1.6.2 Technical Feasibility

We believe that the system will be feasible because of the number of resources available online for this project. There are two difficulties we will be facing on this project. The first one is obtaining datasets, the second one is training the system with the limited resources we have.

1.6.3 Schedule Feasibility

Most of the time spent on this project will be to make the datasets and train the system so in order to do these things efficiently we have planned to start the programming aspects of the project ahead of the given time line.

1.7 Scope

The study of sign language recognition focuses on the recognition of finger spelling or the recognition of signs in different sign languages.

The scope of this system is limited only to translating finger spelt Ethiopian Sign Language to Amharic Text.

The system won't cover this features and functionality:

- This system doesn't handle the task of translating Amharic text to Ethiopian Sign Language.
- The system will not cover the study of continuous Ethiopian sign language.
- The system will not cover the recognition of non-manual feature which is facial expression.

Limitation

From a computer vision perspective, this problem represents a significant challenge due to a number of considerations, including:

- Environmental concerns (e.g. lighting sensitivity, background, and camera position)
- Occlusion (e.g. some or all fingers, or an entire hand can be out of the field of view)
- Sign boundary detection (when a sign ends and the next begins)
- Co-articulation (when a sign is affected by the preceding or succeeding sign

1.8 Methodology

The methodology we use for implementing this system can be categorized into different sections depending on its modules and development process.

1.8.1 Data Collection

All necessary data which is needed to train our neural net will be collected. Various 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 incorporate.

1.8.2 Hand Segmentation

The first stage in our approach is to localize the hand in the image. Much previous work has tackled the task of hand detection and tracking.

1.8.3 Image Description

The second stage is to describe the salient appearance properties of the hand. Such properties are often geometrical information, e.g. positioning or orientation, which is translated in to a normalized format. A descriptor is then generated from normalization. Ideally, the descriptor is a vector fixed length, as such vectors are well comparable and simplify classification.

1.8.4 Classification

The third stage is classification. We will develop CNN that learn features as well as the weights corresponding to each feature. Like other machine learning algorithms, CNNs seek to optimize some objective function, specifically the loss function.

1.8.5 Testing

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

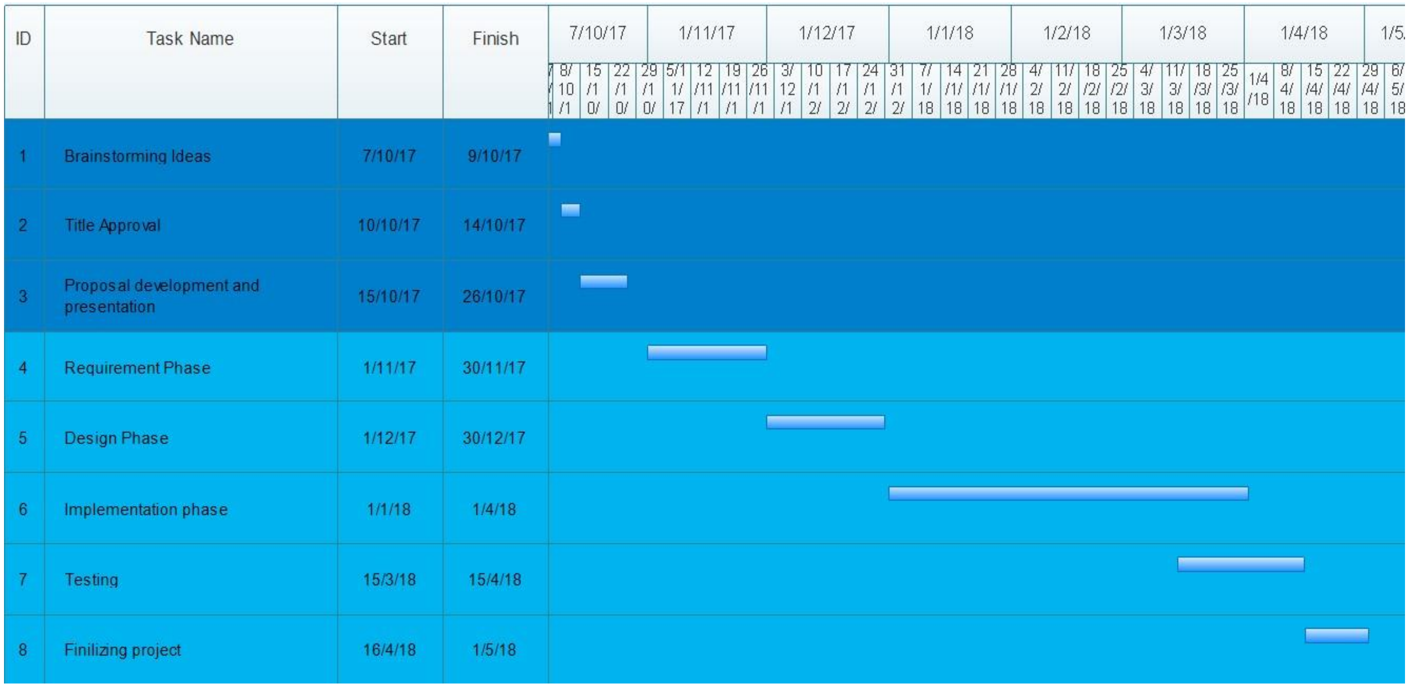
Software Developing Methodology

The Spiral Development Model is a risk-driven process model generator for software projects. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping. [15]

The Lifecycle we chose is spiral Development Model. We chose it because the various modules in the system (Hand Segmentation, Feature Extraction and Model Training) need continuous assessment and improvements. The risk involved in all the module is accuracy. Using this Development Model will help improve the project out come until the last minute.

1.9 Project Management plan

1.9.1. Time Management plan



ID	Task Name	Start	Finish
1	Brainstorming Ideas	7/10/17	9/10/17
2	Title Approval	10/10/17	14/10/17
3	Proposal development and presentation	15/10/17	26/10/17
4	Requirement Phase	1/11/17	30/11/17
5	Design Phase	1/12/17	30/12/17
6	Implementation phase	1/1/18	1/4/18
7	Testing	15/3/18	15/4/18
8	Finilizing project	16/4/18	1/5/18

Figure 3 Gantt chart

Task Name	Duration (days)	Start	Finish
Amharic book reading application using Optical Character Resolution and Natural Language Processing	245	10/7/2017	6/17/2017
Brainstorming Ideas	2	10/7/2017	10/9/2017
Title Approval	4	10/10/2017	10/14/2017
Proposal Writing and Presentation	11	10/15/2017	10/26/2017
Requirement Phase	**30**	**11/1/2017**	**11/30/2017**
Gathering Project Requirement	10	11/1/2017	11/10/2017
Preparing Software Requirement Document(SRS)	15	11/11/2017	11/25/2017
Data preparation	5	11/26/2017	11/30/2017
Design Phase	**30**	**12/1/2017**	**12/30/2017**
Creating system design	15	12/1/2017	12/15/2017
Preparing Software Design Document(SDS)	15	12/16/2017	12/30/2017
Implementation Phase	**90**	**1/1/2018**	**4/1/2018**
Developing Hand Segmentation	20	1/1/2018	1/20/2018
Developing Feature Extraction	20	1/20/2018	2/10/2018
Developing Module Training	30	2/10/2017	3/10/2018
Integrating the modules	20	3/10/2017	4/30/2018
Testing	**30**	**3/15/2018**	**4/15/2018**
Finalizing Project	**15**	**4/16/2018**	**6/20/2018**

Table 1 Time management Plan

1.9 .2 Quality Management Plan

There are two main potential risks to the software quality. The first one is from computer vison perspective like lighting sensitivity, background, and camera position. This could be improved by using better feature extraction algorithms. The second potential risk accuracy of classification. With regards to this, we will try to train our model with as much possible data as we can.

1.9 .3 Communication Management Plan

The purpose of communication plan is to define the communication requirements for the project and to apprehend how information will be distributed among the development team. Communication activities are performed by all team members and everyone will be kept informed of ongoing process of the project.

In order to achieve consistent and effective communication, standardization of documentation is used to simplify the complexities of project management communications. The team will utilize an agreed standard template for meeting agenda and meeting minutes. The meetings will be held every two days and all team members should participate. Concerning the project milestones, the team will hold a meeting with the advisor. These project milestones are expected to be presented to the project advisor so that he gives feedbacks and comments.

Meeting Agenda will be discussed one day in advance of the meeting. The first item in the agenda should be a review of action items from the previous meeting.

Type of Communication	Method / Tool	Frequency /Schedule	Information	Participants / Responsibilities
Internal Communication:				
Project Meetings	Face to face	3 times a week	Project status, problems, risks, changed requirements	Project Team Members
Sharing of project data	Team Group chat	When available	All project documentation and reports	Project Team Members
Sprint meeting	Face to face	Once every 3 week	Plan the next sprint, talk about issues raised in the previous sprint, project status	Project Team Members Advisor
Milestone Meetings	Face to face	Before milestones	Project status	Project Team Members, Advisor
Final Project Meeting	Face to face	Before final project	Wrap-up Experiences	Project Mgr Project Team
External Communication and Reporting:				
Project Report	Word document	As needed	Project status - progress - risks	Advisor, Project Team Members, Project Committee

Table 2 Communication Management Plan

Chapter 2: Requirement Analysis

2.1 Introduction

This document is a software requirement specification (SRS) of Vision Based Finger Spelling Recognition for Ethiopian Sign Language. In this document we outline product perspective, product functions, user characteristics, general constraints and assumption and dependencies of the system to be developed. It also includes external interface requirements, use cases, non-functional requirement, inverse requirement, user interface, design constraints and logical database requirement and other requirements of the system.

2.2 General Description

This section of the document describes product perspective, product functions, user characteristics, general constraints, and assumptions and dependencies.

2.3 Product Perspective

Sign language is a primary tool for the deaf people to communicate with the community around them. But, most of our communities do not have the knowledge of sign language [2]. This situation leads to creation of a huge communication gap between people with hearing disabilities and the community. The problems are expressed as follows:

- **Economic cost**: - Hiring a translator is not always affordable for all people with hearing disabilities due to economic situations.
- **Privacy issue**: - Some secrets and personal information may be acquired by translators which the communicating individuals/parties may not be comfortable with sharing.
- **Lack of enough translators**: - The number of translators is very low compared to the number of people with hearing disabilities.
- Task of translating is exhausting for the translators that work in long meetings and broadcasts.

There is a system developed as part of research project titled Finger Spelling Recognition for Ethiopian Sign Language, however this project used basic Machine Learning algorithm called SVM for classification and the signer should wear a white glove in order to make it easier for hand segmentation. The points that make our system more useable are, we will use artificial neural network for classification which is more suitable for Image related classification and the signer shouldn't have to wear a white glove.

The target goal is to overcome the above limitations by developing an automated Ethiopian sign language recognizer.

2.4 Product Functions

The main function of the system is translation of Ethiopian sign language to Amharic text. This can be summarized by the figure below.

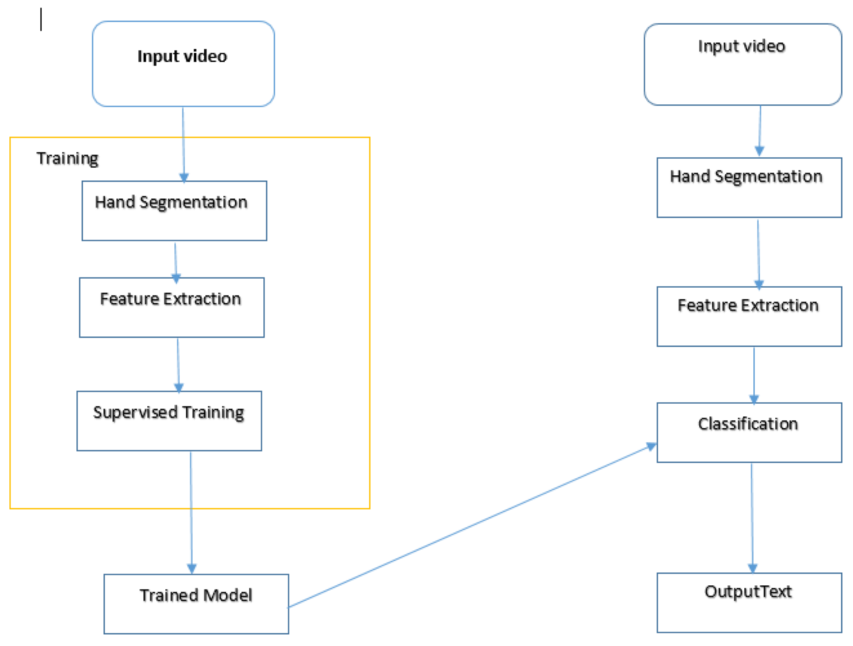


Figure 4 System Architecture

As depicted in figure, the system is comprised of four modules:

1. **Video Acquisition (Input Video)**: - The first task before proceeding to any recognition process is the acquisition of the input video in appropriate manner. We captured videos in a way that will help us to extract all necessary information.

2. **Hand Segmentation**: - The first step in hand sign language recognition is obviously to find the hand region by eliminating all the other unwanted portions in the video sequence. We will use the concept of running averages. Running average is a concept used in background extraction [3], instead of always using the first frame as a clear background, it will update it constantly, by calculating a moving average of it.
   After figuring out the background model using running averages, we will use the current frame which holds the foreground object (hand in our case) in addition to the background. We calculate the absolute difference between the background model (updated over time) and the current frame (which has our hand) to obtain a difference image that holds the newly added foreground object (which is our hand). This is what Background Subtraction is all about. To detect the hand region from this difference image, we need to threshold the difference image, so that only our hand region becomes visible and all the other unwanted regions are painted as black. After thresholding the difference image, we find contours in the resulting image. The contour with the *largest area* is assumed to be our hand. (Counter is boundary of an object located in an image).

3. **Feature Extraction: -** The module helps to describe the salient appearance properties of the hand. Such properties are often geometrical information, e.g. positioning or orientation, which are features of the hand.
   Features are components of an image which have very important information that could represent the image. The feature extraction for sign language focuses on features of the moving hands and head (in our case only hand). Important points that could describe these body parts should be extracted efficiently. In addition to this the recognition algorithms for image classifications require a numerical description of the image to be recognized.

4. **Classification**: - This module performs the task of classifying each features to the corresponding Amharic letter representation. We will develop CNN that learn features as well as the weights corresponding to each feature.
   CNNs are a category of Neural Networks that have proven very effective in areas such as image recognition and classification [4]. The CNN will have a learning data set of previously labeled raw pixel generated images, which are matrixes of pixel values along with the corresponding identifiers of the Amharic letter.

2.5 User Characteristics

We expect our user to have the ability to read Amharic text and he/she is not expected to have the knowledge of Ethiopia sign language.

2.6 General Constraints

The following are the general constraints of the system: -

- The system will run on windows devices.
- The system user interface language is available in English and Amharic.
- Smooth background with a color that doesn't resemble skin color.

2.7 Assumptions and Dependencies

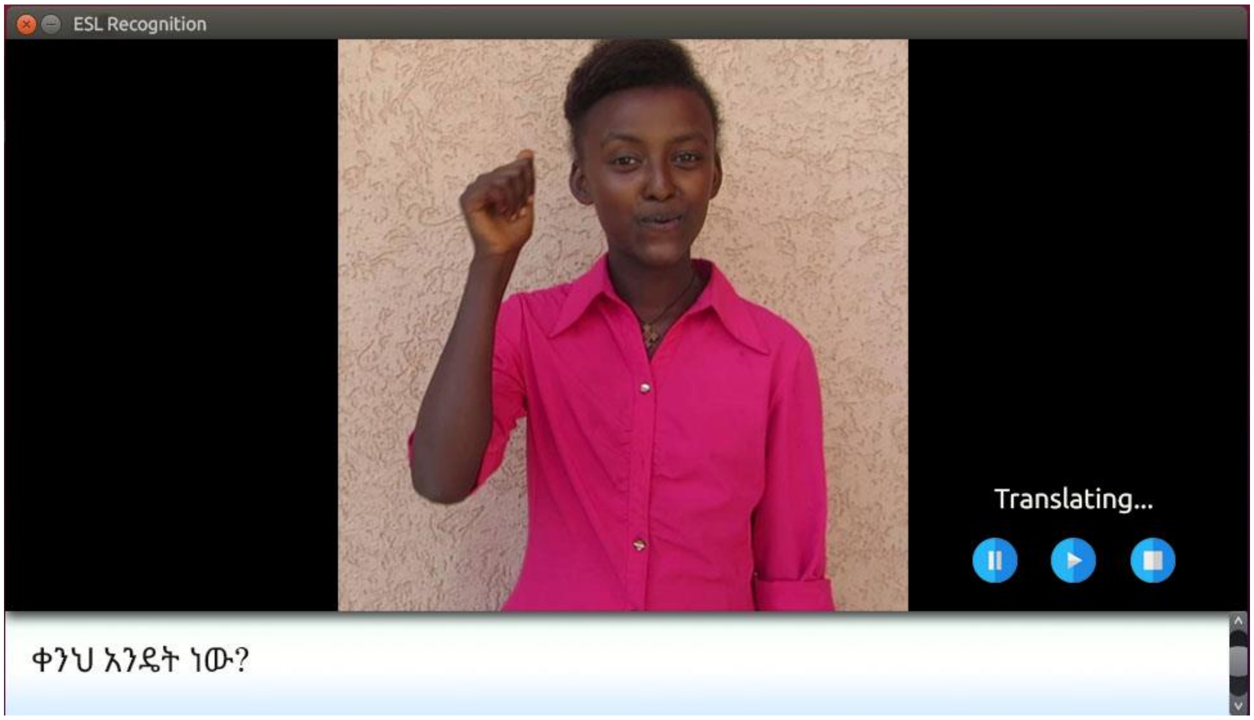
In order for the system to work properly and consistently, the following assumptions and dependencies should be fulfilled.

- The input language is pure finger spelt Ethiopian sign language.
- The input camera can any mobile device camera.
- There should be a smooth background with a color that doesn't resemble skin color.

2.8 External Interface Requirements

2.8.1 User Interfaces

The system is a desktop based Graphical User Interface (GUI) application. The first time the application is launched, the user will be asked to start the translation as shown on Figure 2. The user can also pause and start the application by pressing a "Pause/Start" button during translation.



ቀንህ እንዴት ነው?

Figure 5 User Interface

2.8.2 Hardware Interfaces

The system doesn't require any other hardware interfaces other than Windows machine.

2.8.3 Software Interfaces

The system doesn't require any software interface.

2.8.4 Communications Interfaces

The system doesn't require any communication interface.

2.9 Functional Requirements

Table 3 Functional Requirement 01

ID	**FR:01**
Name	Accepting video as an input.
Summary	The system shall accept video input.
Input	The system shall use the video obtained from the mobile camera.
Processing	There is no processing involved in this requirement.
Output	The system shall output the stream of video data into a buffer.
Error handling	In case the cameras used is not functioning properly, the system shall notify the user about the specific error.

Table 4 Functional Requirement 02

ID	**FR:02**
Name	Segmenting hand.
Summary	The system shall segment the hand from the video obtained from the camera.
Input	The system shall read the video data obtained from a buffer.
Processing	The system shall segment the hand and extract the background.
Output	Only hand region becomes visible and put the video data into a new buffer.
Error handling	If there is no moving image in the video, the system will notify the user to move his hands.

Table 5 Functional Requirement 03

ID	**FR:03**
Name	Extracting features from segmented region.
Summary	The system shall extract specific features from hand sign.
Input	The system shall read video data obtained from the buffer.
Processing	The system shall extract geometrical information of the hand.
Output	A numerical description of the image (the hands).
Error handling	Whenever the system cannot extract hand features, the system will display the specific error message.

Table 6 Functional Requirement 04

ID	**FR:04**
Name	Classification.
Summary	The system shall analyze the numerical description in order to predict its corresponding value.
Input	Numerical values.
Processing	The system shall use its accumulated knowledge during training in order to predict the corresponding value.
Output	Percentage value with numerical representation of Amharic letter IDs.
Error handling	None.

Table 7 Functional Requirement 05

ID	**FR:05**
Name	Translating Ethiopian sign language to Amharic text.
Summary	The system shall search for Amharic letter mapping of the sign in a dictionary file.
Input	Percentage value with numerical representation of Amharic letter IDs.
Processing	The system shall read from a dictionary file that contains a mapping from Sign Language representation to Amharic Letter. Each Amharic letter is given unique ID.
Output	A list of Amharic letter IDs that correspond to the input signs.
Error handling	If the system cannot find a corresponding Amharic letter value for a given Sign in the dictionary file, it will notify the user that it could not translate that sign.

Table 8 Functional Requirement 06

ID	**FR:06**
Name	Displaying Amharic text.
Summary	The system shall display the Amharic letter representation of the corresponding sign.
Input	List of Amharic letter IDs.
Processing	The system shall run scripts that will use Amharic letter ID to display the letter.
Output	Amharic text.
Error handling	None.

2.10 Use Cases

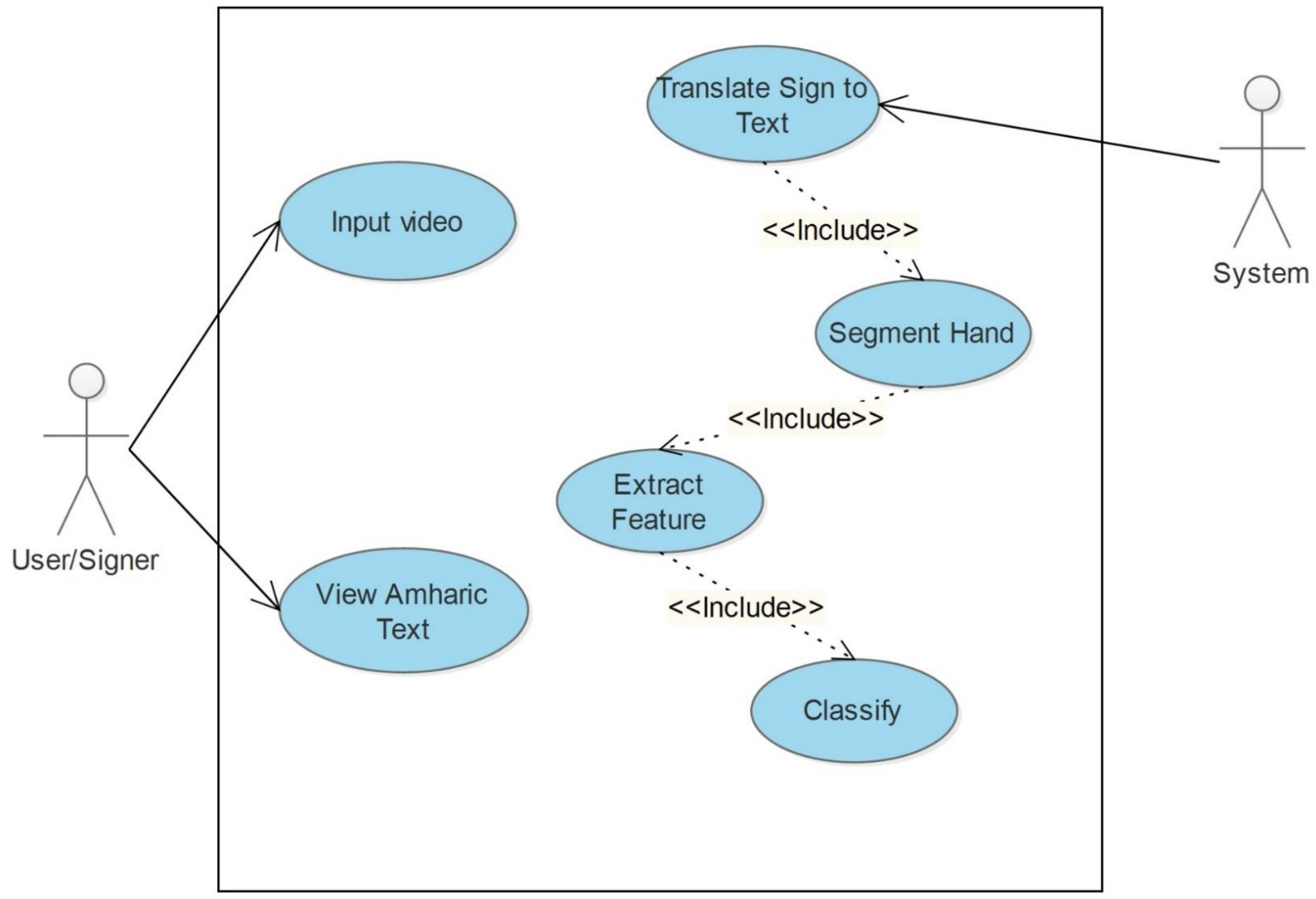


Figure 6 Use Case Diagram

2.10.1 Use Case #1 Input Video

Name: Input Video

Actors: Signer/User

Description: Enables the signer to input video of them signing

Precondition: The user needs camera

Flow of events:

1. The user clicks on Start for the application to start recognizing.
2. The user records through the camera.
3. End of the use case.

Alternate: camera not working properly, the system shall notify the user that the camera is not working properly.

Post Condition: Input video is received

2.10.2 Use Case #2 Segment Hand

Name: Segment Hand

Actors: System

Description: The system accepts the video data from the camera and segment the hand portion only

Precondition: The user succeeds in inputting video through the camera

Flow of events:

1. The system accepts the video data from the camera attached to the machine on which it is running on.
2. It then computes the running average to get the background.
3. It calculates the absolute difference between the background model (updated over time, obtained from step 2) and the current frame (which has our hand).
4. It thresholds the image difference.
5. It then finds counters in the resulting image.
6. End of use case.

Alternate:

1. The system cannot get video data from the camera.
   - The system notifies the user to check if the camera is working and to try again.
2. The system cannot detect the background because there a movement in the background.
   - The system notifies the user about the error and to try again by sitting in front of a static background.

Post Condition: Hand segmented from the video

2.10.3 Use Case #3 Extract Feature

Name: Extract feature

Actors: System

Description: The System accepts the segmented hand video data and extract geometrical information of the hand

Precondition: Hand must be segmented

Flow of events:

1. The System looks at the segmented hand.
2. It generates raw pixel values for the segmented hand.
3. It computes the center of mass of the segmented hand.
4. It analyses the direction and distance of movement of the hand.
5. It assigns the corresponding vowel representation of the movement.
6. It then sends the image description to classifier.
7. End of use case.

Alternate: None

Post Condition: extracted features are sent to Predictor (trained model)

2.10.4 Use Case #4 Classification

Name: Classification

Actors: System

Description: The classifier accepts the extracted features from feature extractor and make a prediction about the sign

Precondition: The feature extractor has successfully extracted features

Flow of events:

1. The System outputs the extracted features to the classifier.
2. The classifier receives the output from features extractor using API calls.
3. The classifier makes its prediction by detecting some kind of pattern from the extracted features.
4. End of use case.

Alternate: An error occurs during transmission of the features to the classifier
The classifier sends an error message notifying the system that an error has occurred.

Post Condition: The Classifier successfully classified the images.

2.10.5 Use Case #5 Translate sign to text

Name: Translate sign to text

Actors: System

Description: The system maps the classified images to the corresponding Amharic letter representation

Precondition: The classifier successfully classified the images

Flow of events:

1. The system receives the representation of the classified image.
2. The system then maps the Amharic letter representation from a dictionary file.
3. The system then displays the Amharic letter representation.
4. End of use case.

Post Condition: Segmented Amharic text displayed on the screen

2.11 Non-Functional Requirements

2.11.1 Performance

There will always be delays during translation process whether it is done by a human or a machine. This system will also have delays in translating sign language to text as it has to accept video input, analyze it and translate it to appropriate text representation. Hence, there is some level of acceptable delay in getting output from this system. However, the time delay of a translating a given word should not exceed 5 seconds for the system to be usable properly.

The translation of the system should be reliable in that it should provide accurate translation of sign language text. We will perform different evaluation method to calculate our systems accuracy in order to know if it is reliable or not.

2.11.2 Maintainability

The system will be developed by taking extensibility in mind. To ensure the code we write to develop the system is reusable, we will follow the SOLID principles. The code will also be available on GitHub, hence, anyone who is interested can extend and contribute further.

2.12 Inverse Requirements

The Sign to Amharic text translator **CANNOT** be used for the following scenarios:

- For translating Amharic Speech to Sign language.
- For translation Amharic text to sign language.
- Recognizing gender of the signer based on his/her sign/face.

2.13 Design Constraints

There are two design constraints that must be taken into account for the development of the system.

The first one is constraints related to the input of the video.

- Environmental concerns (e.g. lighting sensitivity, background, and camera position)
- Occlusion (e.g. some or all fingers, or an entire hand can be out of the field of view)
- Sign boundary detection (when a sign ends and the next begins)
- Co-articulation (when a sign is affected by the preceding or succeeding sign.

The second constraint is the recognition of the signs using the CNN.

- Processing power: - The CNN requires heavy processing power for fast computation.
- Training Data: - The accuracy is limited by the amount of training data.

2.14 Logical Database Requirements

This is system doesn't use a database, hence, it doesn't have any logical database requirements.

2.15 Other Requirements

Packaging Requirements

The system will be distributed in an installer package that will install it on the major platforms. However, as an alternative, we will provide the code along with build files that will be used by advanced users to build the system by themselves.

Legal Requirements

The system will be distributed under GPLv3 license. The reason we chose this license among other open-source licenses is that it imposes a requirement on users who wish to distribute a software licensed under GPL to license their distribution under GPLv3 too. We believe this encourages anyone who wants to build upon or extend our system to make their product open-source, thereby, contributing to the development of work in the area of Sign Language translation.

2.16 Change Management Process

Requirements may change during development course of the project. This changes are applied only after going through the following change management process.

- A team member proposes a requirement change. The proposed change should include scope, reason and impact for the change.
- The pros and cons of the change will be examined by the team.
- The requirement will be submitted to advisor for approval.
- The advisor will decide if the change is relevant.
- The requirement change will be added to this document.

Chapter 3: SYSTEM DESIGN

3.1 General Overview

Currently, sign language interpreters/translators do the (manual) job of translating from a spoken language to sign language and vice versa. The proposed system is to develop a system that translates Ethiopian sign language to Amharic text.

The system is designed for any one who can read Amharic text. Its major goal is to reduce the communication barrier between hearing imparied person and hearing person.

As depicted in the Figure 1, the system consists of four modules.

1. **Video Acquisition (Input Video)**: - The first task before proceeding to any recognition process is the acquisition of the input video in appropriate manner.

2. **Hand Segmentation:** - this module handles the task of segmenting the hand.

3. **Feature Extraction:** - The module helps to describe the salient appearance properties of the hand. Such properties are often geometrical information, e.g. positioning or orientation, which are features of the hand.

4. **Classification: -** This module performs the task of classifying each features to the corresponding Amharic letter representation.

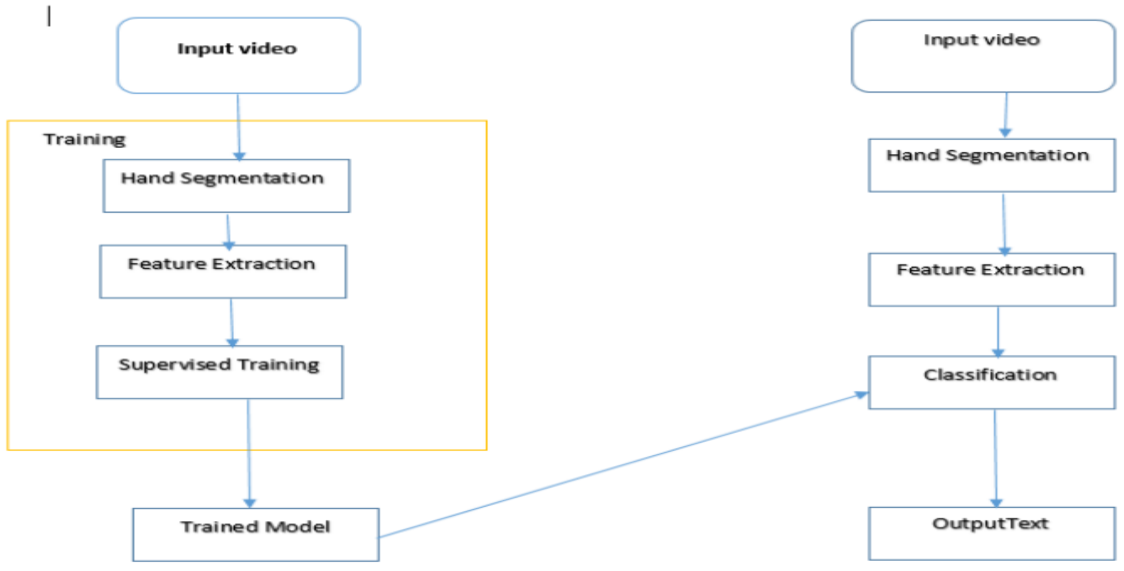


Figure 7 system architecture for FSRFEthSL

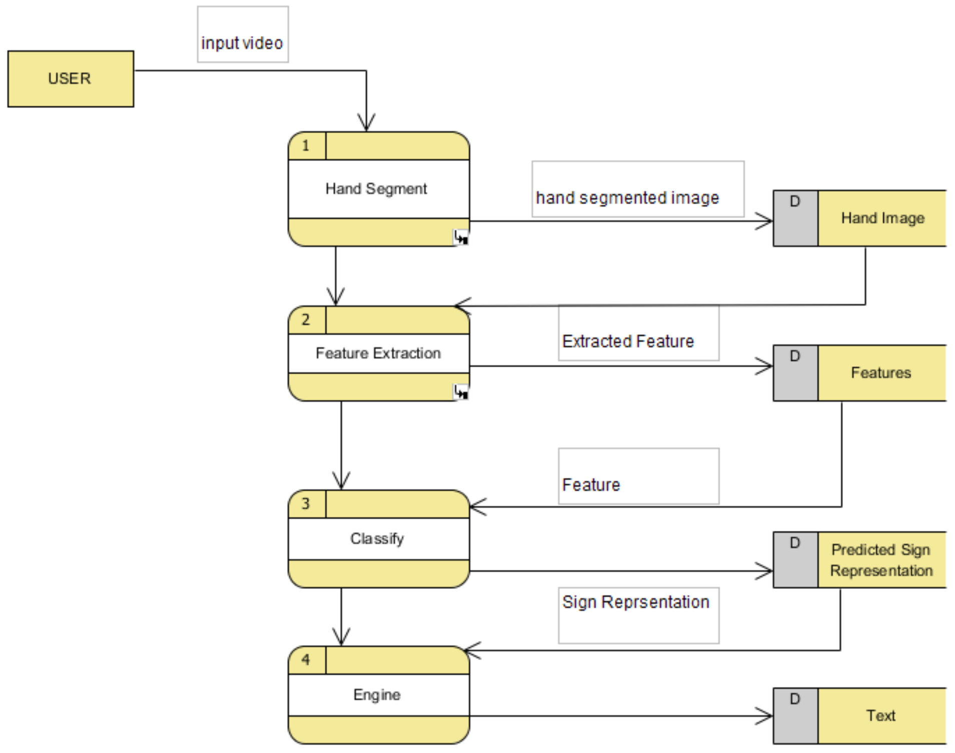


Figure 8 high-level context diagram for FSRFEthSL

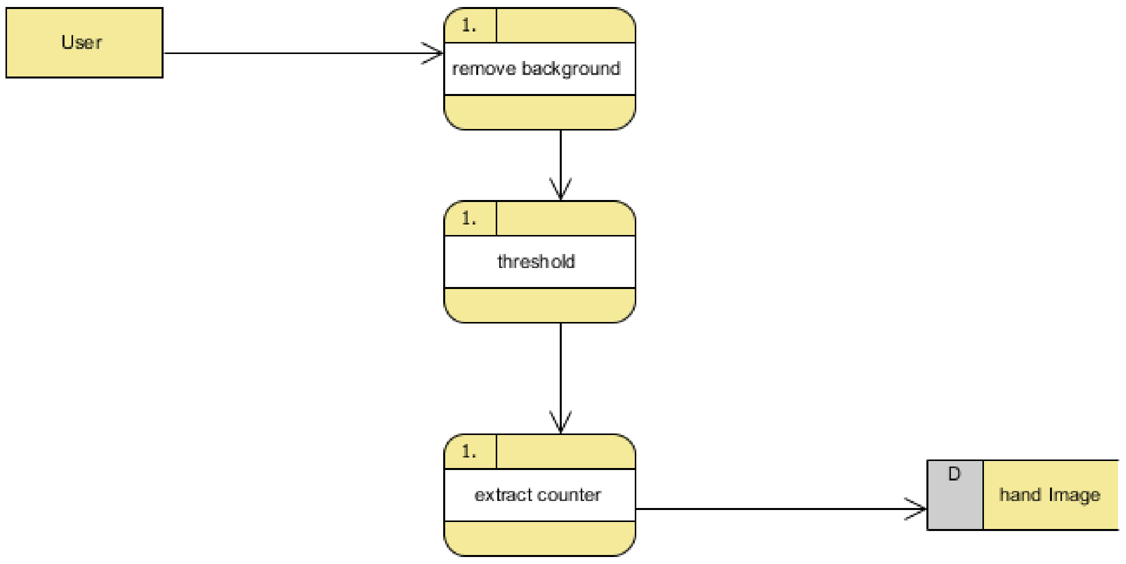


Figure 9 data flow diagram for FSRFEthSL

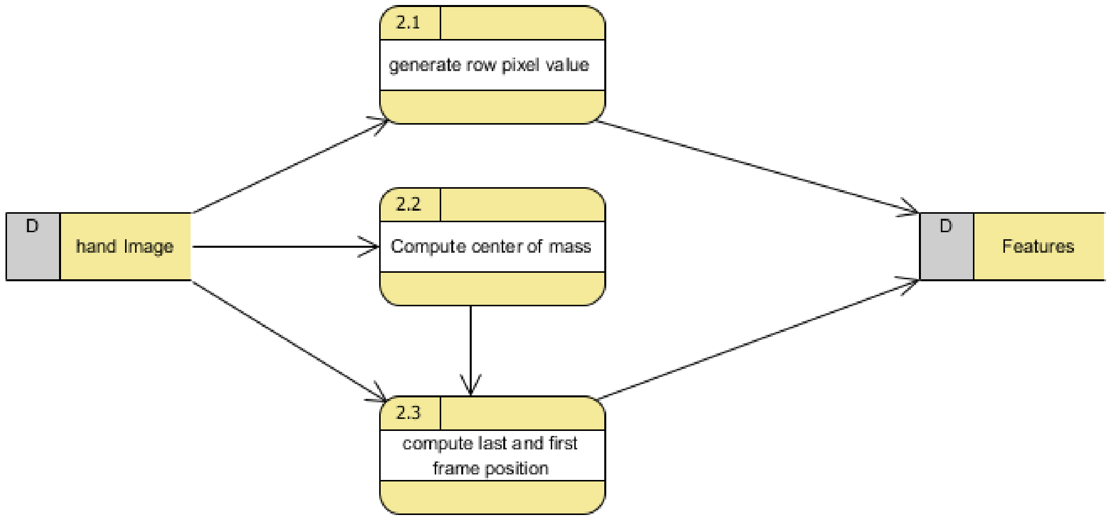


Figure 10 level 2 data flow diagram for FSRFEthSL

3.2 Development Methods & Contingencies

The system is developed on an Object Oriented Approach. We also use the following method and tools described below to develop the system.

- **OpenCV** is one of the most popular libraries used to develop Computer Vision applications. It enables us to run many different Computer Vision algorithms in real time. It has been around for many years, and it has become the standard library in this field. One of the main advantages of OpenCV is that it is highly optimized and available on almost all the platforms.
- **TensorFlow** is an open-source software library for machine learning across a range of tasks. It is a symbolic math library, and also used as a system for building and training neural networks to detect and decipher patterns and correlations, analogous to human learning and reasoning.
- **Python** programming language for development purpose.

3.3 System Architecture

3.3.1 Subsystem decomposition

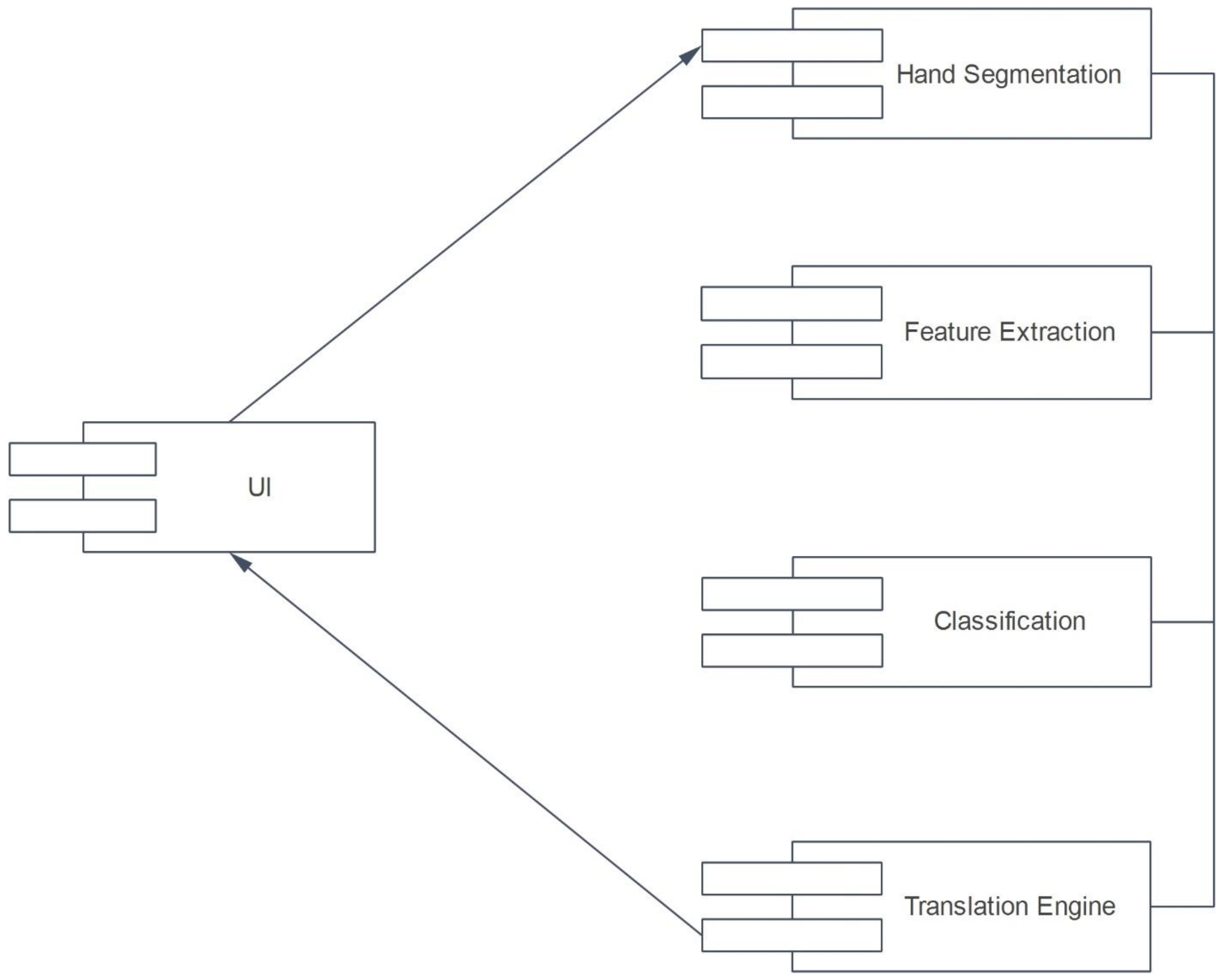


Figure 11 Component diagram for FSRFEthSL

3.3.2 Hardware/software mapping

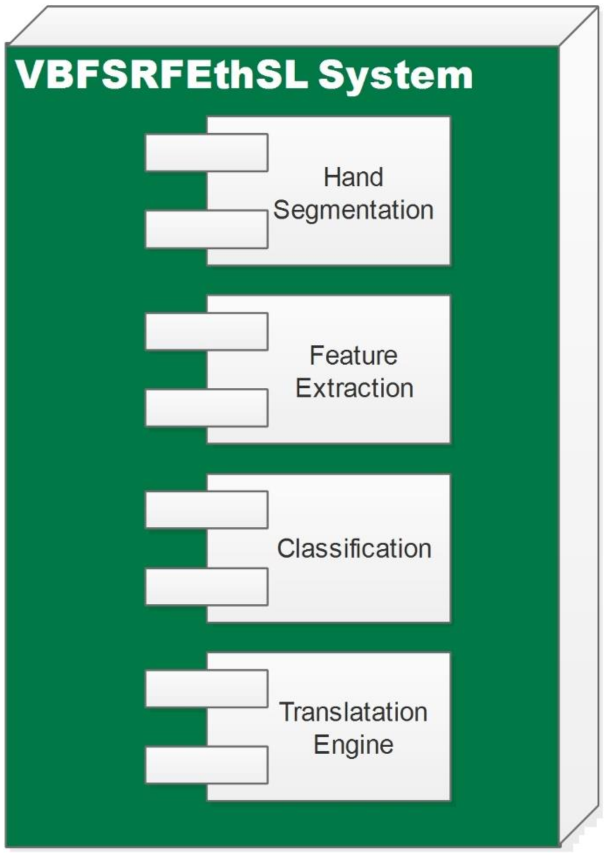


Figure 12 Deployment Diagram for FSRFEthSL

3.4 Object Model

3.4.1 Class Diagram

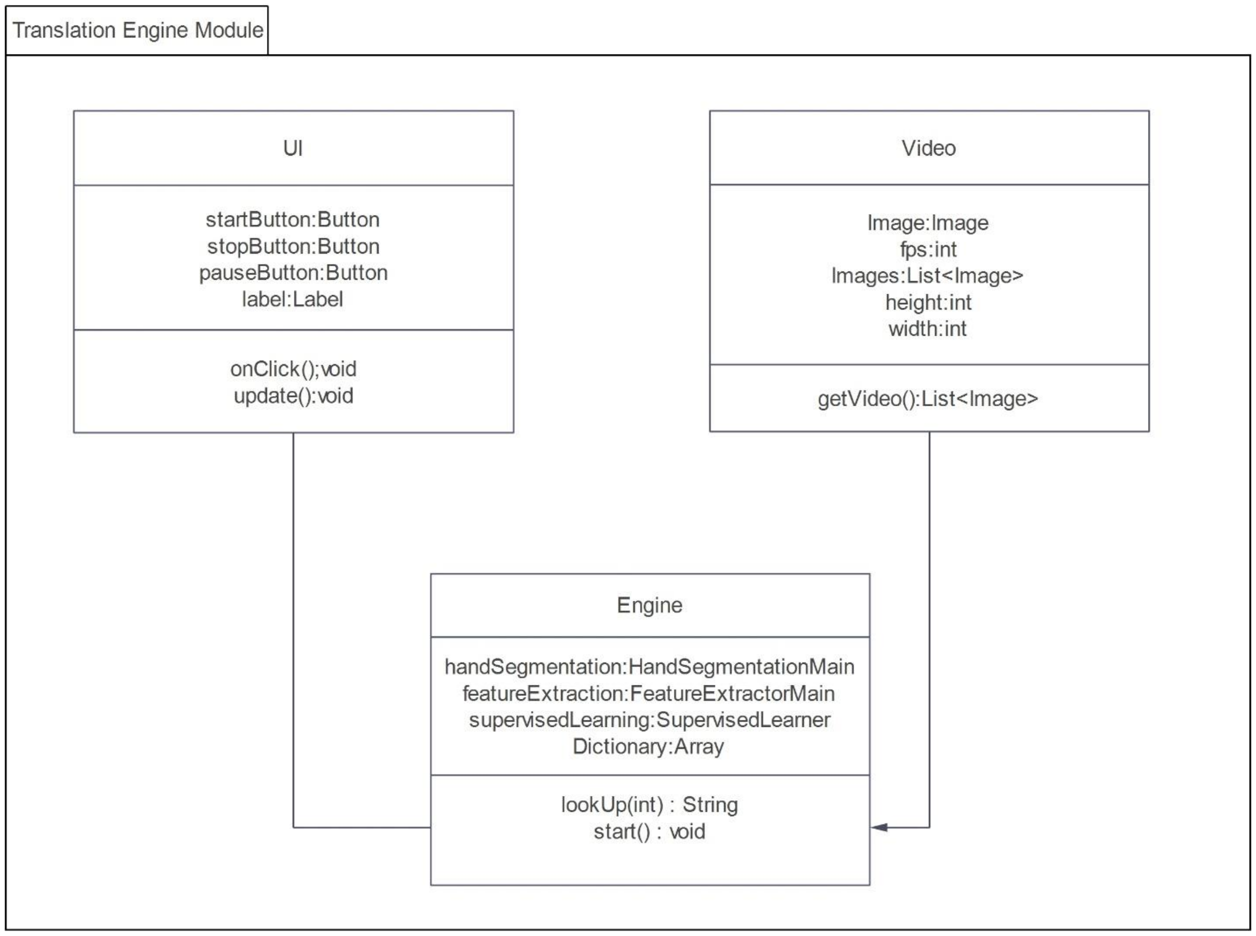


Figure 13 Translation Engine Module class Diagram for FSRFEthSL

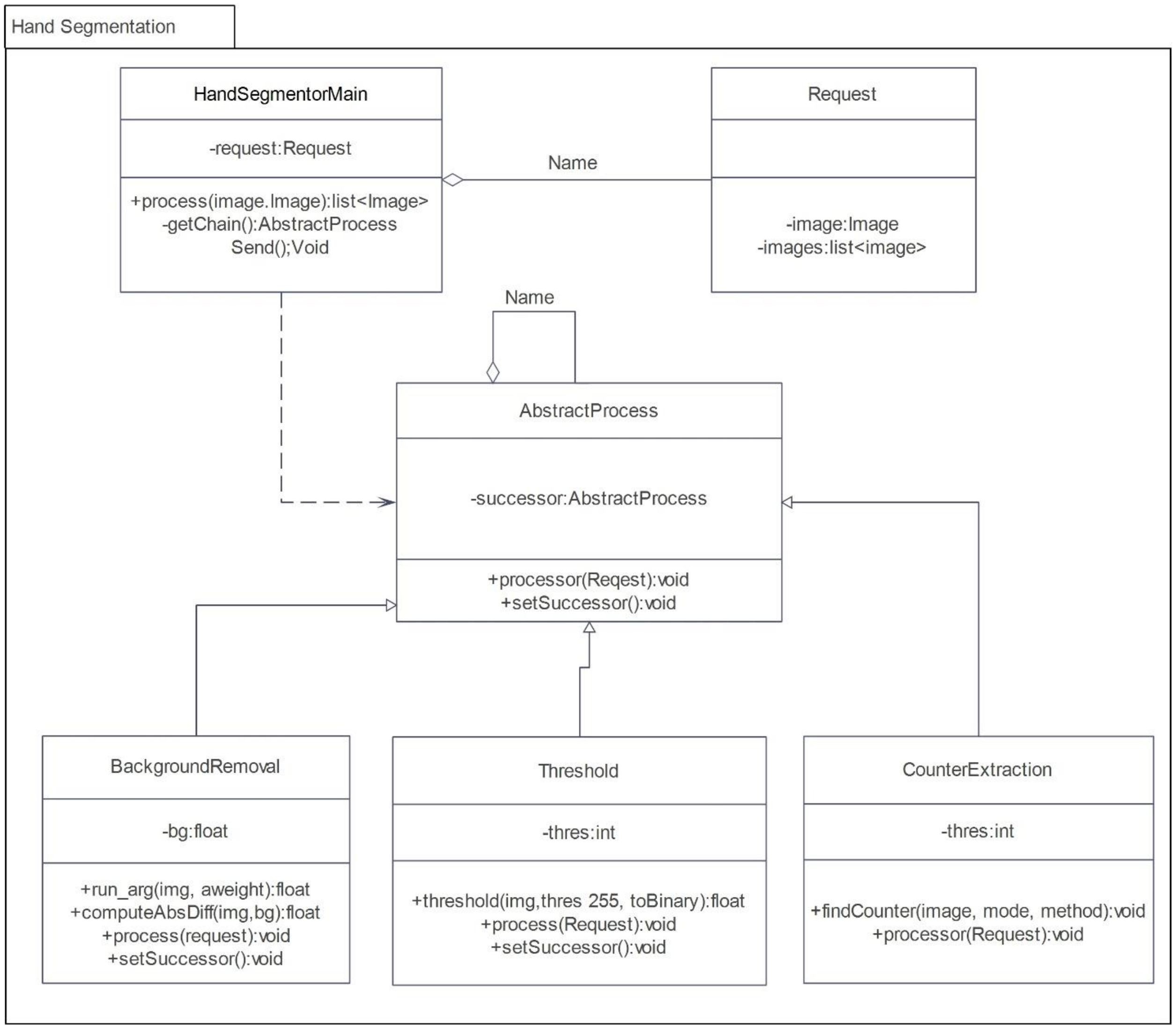


Figure 14 Hand Segmentation class Diagram for FSRFEthSL

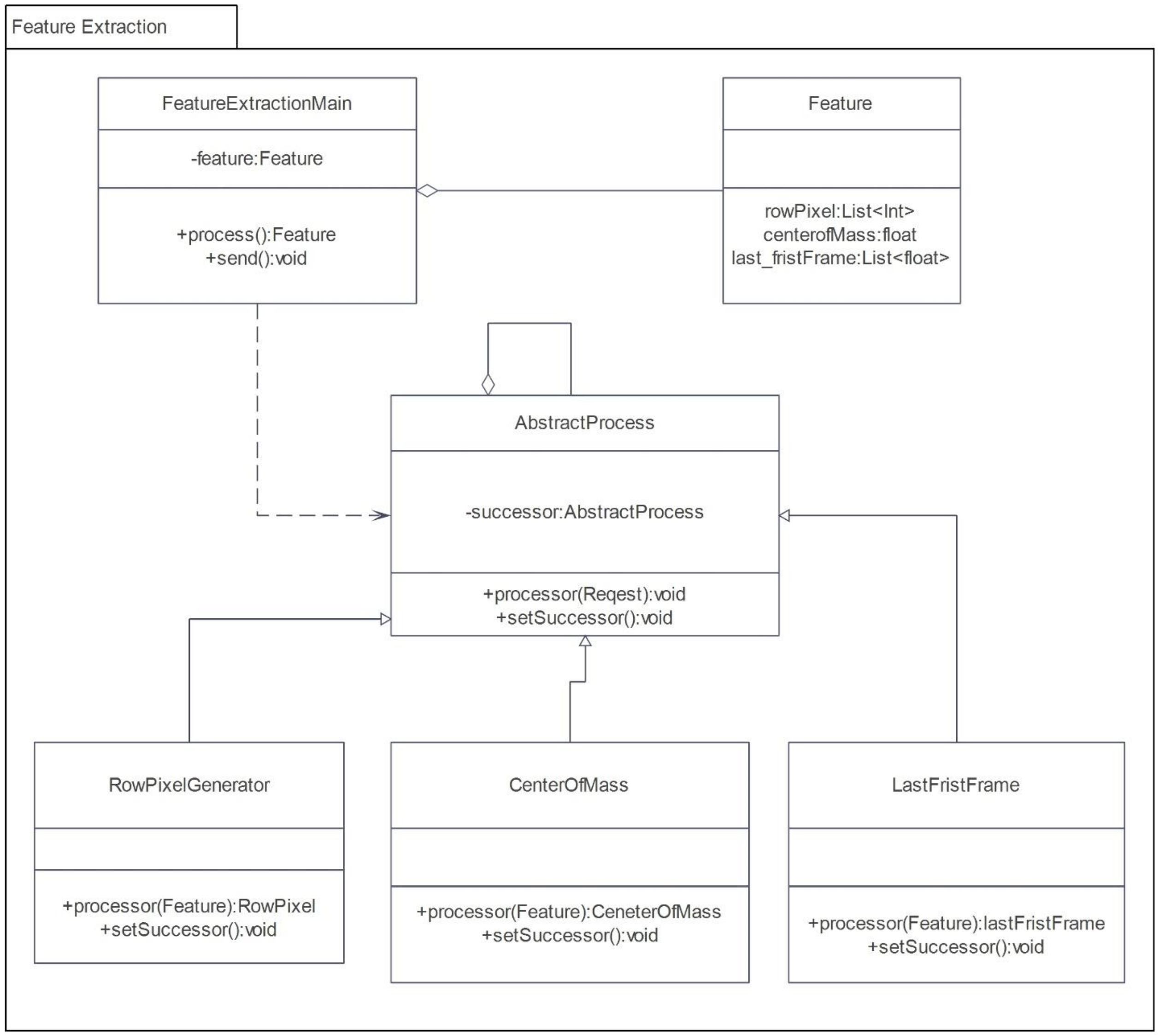


Figure 15 feature extraction class diagram for FSRFEthSL

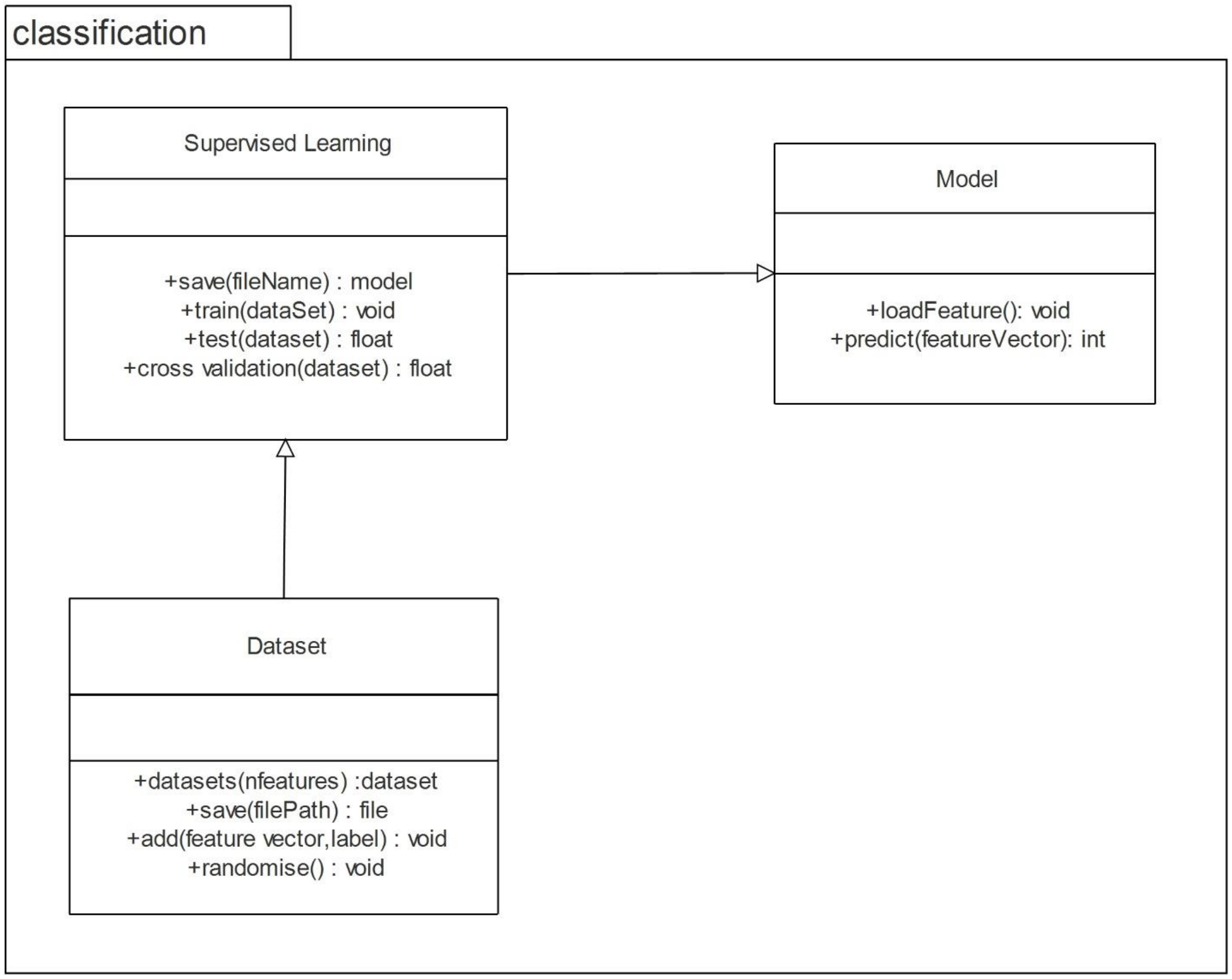


Figure 16 Classification class diagram for FSRFEthSL

3.4.2 Sequence Diagram

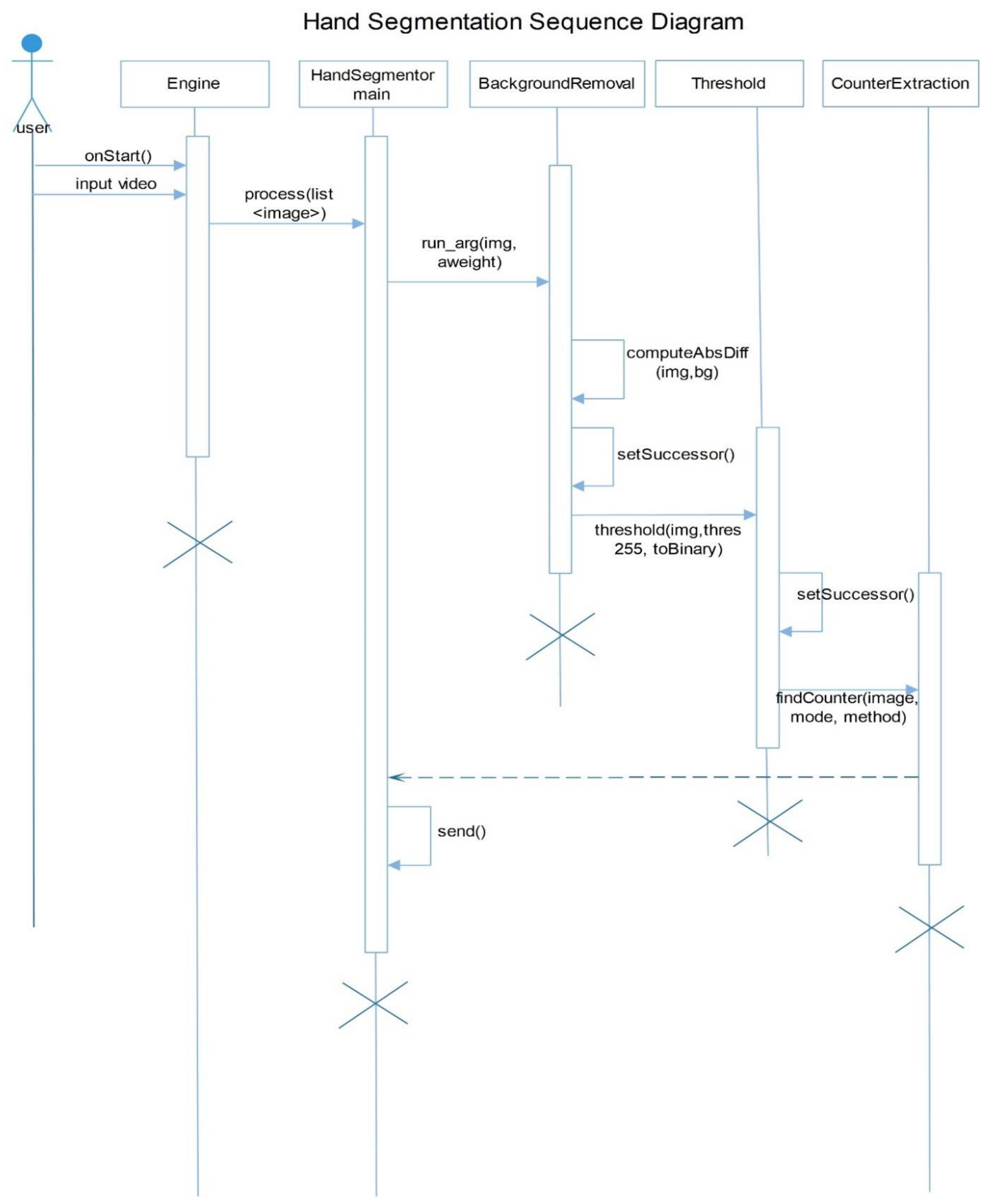


Figure 17 hand segmentation sequence diagram for FSRFEthSL

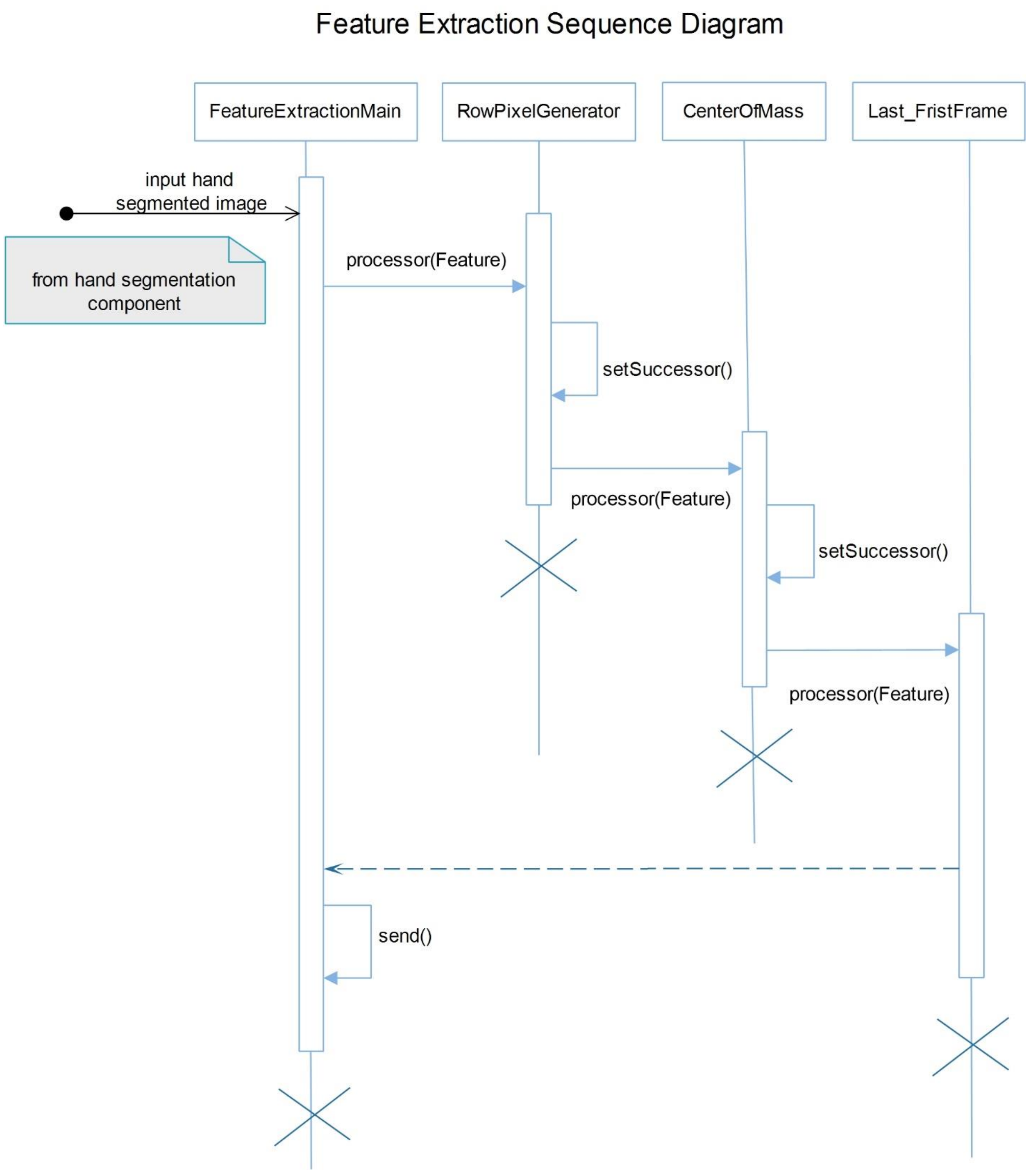


Figure 18 Feature extraction sequence diagram for FSRFEthSL

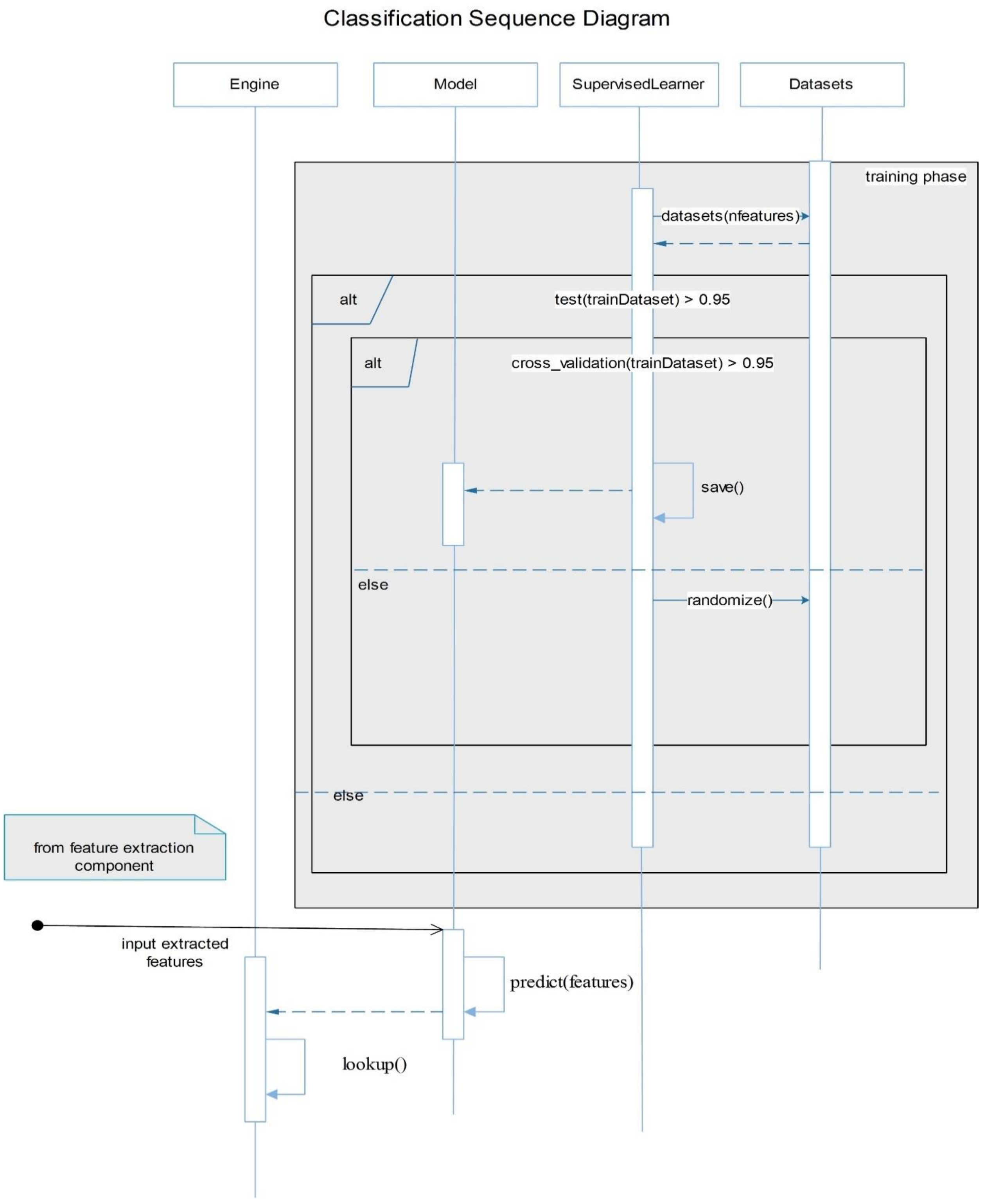


Figure 19 Classification sequence diagram for FSRFEthSL

3.5 Detailed Design

Table 9 UI class

UI
- startButton:Button - stopButton:Button - pauseButton:Button -label:Label
+onClick():void +update():void

Table 10 Attributes description for UI class

Attribute	Type	Visibility	Invariant
startButton	Button	Private	
stopButton	Button	Private	
pauseButton	Button	Private	
label	label	Private	

Table 11 Operation description for UI class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
onClick	Public	void	-	-	Will be applied to the buttons
update	Public	void	-	-	Updates the label whenever a new sign is recognized.

Table 12 Video class

Video
-Image:Image -Images:list<Images> -fbs:int -height:int -width:int
+getVideo(): List<Image>

Table 13 Attributes description for Video class

Attribute	Type	Visibility	Invariant
Image	Image	Private	Image <> NULL and must contain a matrix of vertices to be preprocessed.
Images	List<Image>	Private	Images contains lists of segmented vertices after preprocessing
fbs	Int	Private	Number of images per second
height	Int	Private	
Width	Int	Private	

Table 14 Operation description for Video class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
getVideo	Public	List<Image>	-	-	Initiates webcam loads the images

Table 15 Engine class

Engine
-handSegmentation:HandSegmentation -featureExtraction:FeatureExtractionMain -supervisedLearner:SupervisedLearner -dictionary:Array
+start():void +lookup(int) : char

Table 16 Attributes description for Engine class

Attribute	Type	Visibility	Invariant
handSegmentation	HandSegmentation	Private	Contain the object that perform hand segmentation process
featureExtraction	FeatureExtractionMain	Private	Contain the object that perform feature extraction process
supervisedLearner	SupervisedLearner	Private	Contain the object that perform the classification process
dictionary	Array	Private	Contain array of Amharic letter with their representation

Table 17 Operation description for Engine class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
lookup	Public	Char	int	Classification module should classify the image	Lookup the Amharic letter corresponding to its integer
Start	Public	Void			Launches the application

4.2 Hand Segmentation package

Table 18 Request class

Request
-Image:Image -Images:list<Images>

Table 19 Attributes description for Request class

Attribute	Type	Visibility	Invariant
Image	Image	Private	Image <> NULL and must contain a matrix of vertices to be preprocessed.
Images	List<Image>	Private	Images contains lists of segmented vertices after preprocessing

Table 20 BackgroundRemoval class

BackgroundRemoval
-background : float
+run_arg(img, aweight):float +computeAbsDiff(img,bg):float +process(request):void +setSuccessor():void

Table 21 Attributes description for BackgroundRemoval class

Attribute	Type	Visibility	Invariant
background	Float	Private	background <> NULL contains background value to label vertices as background

Table 22 Operation description for BackgroundRemoval class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
run_arg	Public	float	img, aweight	A Image object must be provided	Calculate the moving average of the background
computeAbsDiff	Public	float	img,bg	The Background moving average must be calculate	Calculates the difference between the background and the new frame to find the new foreground object
process	Public	void	Request	A matrix containing vertices to be segmented must be provided in the request object	Background removed set of vertices will be provided to the next stage.
setSuccessor	Public	void	-	-	The next preprocessor object will be assigned.

Table 23 Threshold class

Threshold
- thres:int
+threshold(img,thres 255, toBinary):float +process(Request):void +setSuccessor():void

Table 24 Attributes description for Threshold class

Attribute	Type	Visibility	Invariant
Thres	Int	Private	TRESHOLD <> NULL contains the threshold value to segment a set of vertices

Table 25 Operation description for Threshold class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
threshold	Public	float	img,thres 255, toBinary	A Image object must be provided	Threshold the images using thres as a maximum value
process	Public	void	Request	A matrix containing vertices to be segmented must be provided in the request object	threshed set of vertices will be provided to the next stage.
setSuccessor	Public	void	-	-	The next preprocessor object will be assigned.

Table 26 CounterExtraction class

CounterExtraction
+findCounter(image, mode, method):void +processor(Request):void

Table 27 Attributes description for CounterExtraction class

Attribute	Type	Visibility	Invariant
Thres	Int	Private	TRESHOLD <> NULL contains the threshold value to segment a set of vertices

Table 28 Operation description for CounterExtraction class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
findCounter	Public	Void	image, mode, method	A Image object must be provided	Calculates largest area counters assumed to be hand
process	Public	void	Request	A matrix containing vertices to be segmented must be provided in the request object	set of vertices containing the segmented hand will be provided to the next stage.

4.3 Feature Extractor package

Table 29 FeatureExtractionMain class

FeatureExtractionMain
+feature: Feature
+process (): Feature +send ():void

Table 30 Attributes description for FeatureExtractionMain class

Attribute	Type	Visibility	Invariant
feature	Feature	Public	feature <> NULL contains a single feature used by classification algorithm.

Table 31 Operation description for Mobile Client class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
process	Public	void	.	The specific feature expected to be extracted is known	This method calls the process methods of each feature extractor object and will provide a feature object containing all the required features
send	Public	void	.	The feature object is created	Send the feature to classification

Table 32 Feature class

Feature
-rowPixel: array -centerOfMass: float -lastFirstFrame: float

Table 33 Attributes description for Feature class

Attribute	Type	Visibility	Invariant
rowPixel	Int	Private	Array of pixel for segmented hand
centerOfMass	Float	private	Center of mass for segmented hand

Table 34 AbstractProcess

AbstractProcess
-successor: AbstractProcess
+processor (Request): void +setSuccessor ():void

Table 35 Attributes description for Mobile Client class

Attribute	Type	Visibility	Invariant
successor	AbstractProcess	Public	- successor <> NULL contains the next operation in the preprocessing chain

Table 36 Operation description for Mobile Client class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
processor	Public	void	.		Provides declaration for process method
setSuccessor	Public	void	.		Provides declaration for setSuccessor method

Table 37 RowPixelGenerator

RowPixelGenerator
+processor (feature): RowPixel +setSuccessor ():void

Table 38 Attributes description for RowPixelGenerator class

Attribute	Type	Visibility	Invariant
			-

Table 39 Operation description for RowPixelGenerator class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
processor	Public	RowPixel	.	Matrices representing the segmented hand	Each values of vertices are extracted

setSuccessor	Public	void	.		The next preprocessor object will be assigned.

Table 40 CenterOfMass

CenterOfMass
+processor (Feature): CenterOfMass +setSuccessor ():void

Table 41 Operation description for CenterOfMass class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
processor	Public	CenterOfMass	.		Compute center of mass
setSuccessor	Public	void	.		The next preprocessor object will be assigned.

Table 42 LastFirstFrame class

LastFirstFrame
+processor (Feature): LastFirstFrame +setSuccessor ():void

Table 43 Operation description for LastFirstFrame class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
processor	Public	void LastFirstFrame	.		Compute the first and last frame position of the segmented hand images
setSuccessor	Public	void	.		The next preprocessor object will be assigned.

4.4 Classification Package

Table 44 SupervisedLearner class

SupervisedLearner
+save(filename):Model +train(dataset):Void +test(dataset):float +cross_validation(dataset):float

Table 45 Operation description for SupervisedLearner class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
save	Public	Model	Filename	The model should have an accuracy value of 0.95	Model has been save successfully and ready to deployed
train	Public	void	Dataset	-	Learn patterns from the training dataset
test	Public	Float	Dataset	The model has been trained on train dataset	Helps to us to evaluate the accuracy on test dataset
cross_validation	Public	Float	Dataset	-	Helps us to evaluate accuracy on the whole dataset

Table 46 Dataset class

Dataset
+datasets(nfeatures):Dataset +save(filePath):File +add(featureVector, Label):void +randomise():void

Table 47 Operation description for Dataset class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
datasets	Public	Dataset	nFeatures	-	Initiates the datasets of object
Save	Public	File	Dataset	-	Saves the datasets to be loaded in the SupervisedLearner object
add	Public	Float	featureVector, Label	Dataset should exist	Adds new feature to datasets
randomise	Public	Void	-	Datasets should exist	Randomize the datasets

Table 48 Model class

Model
+load_feature():void +predict(features):Int

Table 49 Operation description for SupervisedLearner class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
load_dataset	Public	Model	-	-	Loads the features extracted from segmented hand
predict	Public	void	features	Features should be loaded	Classify the features as Amharic letter representation

Chapter 5: Testing

5.1 Introduction

Software testing is an activity to check whether the actual results match the expected results to ensure the software system is defect free. It involves execution of a software component or system component to evaluate one more property of interest

5.2 Features to be tested/not to be tested

5.2.1 Features to be tested

Functional requirements to be tested:

Features	**Level of Risk**
Hand Segmentation	High
Feature Extraction	High
Classification	High

5.2.2 Features not to be tested

Every feature is tested

5.3. Pass/Fail criteria

Every feature will be examined and the success or fail will be measured in the following manner based on the output: -

If the actual output and the expected result are the same the pass/fail criteria is satisfied (success), if the actual output and the expected result are different the pass/fail criteria fails

5.4. Approach/Strategy

Every member of the group will test the whole system and its components separately. Two testing approaches where used, these are unit testing and integration testing.

Unit Test is a test written by the programmer to verify that a relatively small piece of code is doing what is intended to do. Every components of the system have been first been tested before being used in the whole system.

Integration Test is done to demonstrate that different pieces of the system work together. Integration test cover the whole application. Integration test for our system has been done by all members of the group.

No specific test tools are available for this project

Chapter 6: User Manual

EthSL is a desktop application used for translating finger spelling of Ethiopian sign language to text.

6.1 scope

The manual covers the following:

- How to How to start the system
- How to write consonants?
- How to write vowels?
- Space and Delete

6.2 Installation and configuration

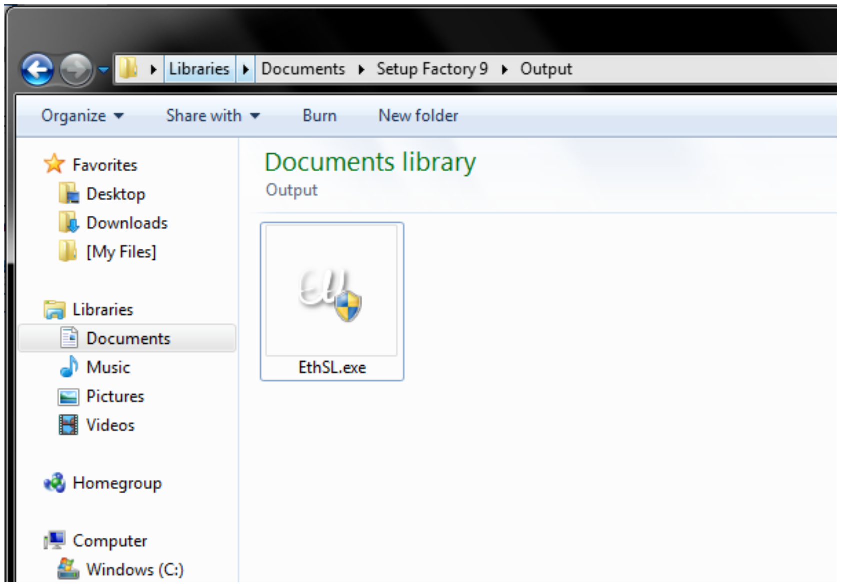
Pre-requirement

The software is compatible with the following Windows Operating Systems.

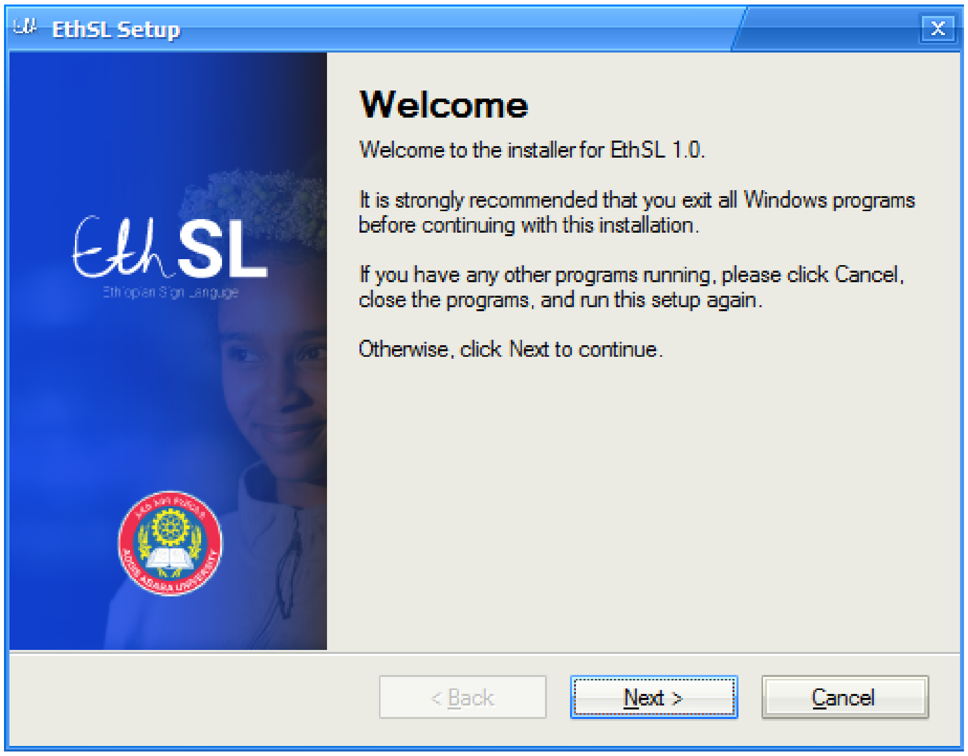
- Windows 7, 8, 10

Installation Procedure

1. Open EthSL setup file



2. Follow Instructions on the setup



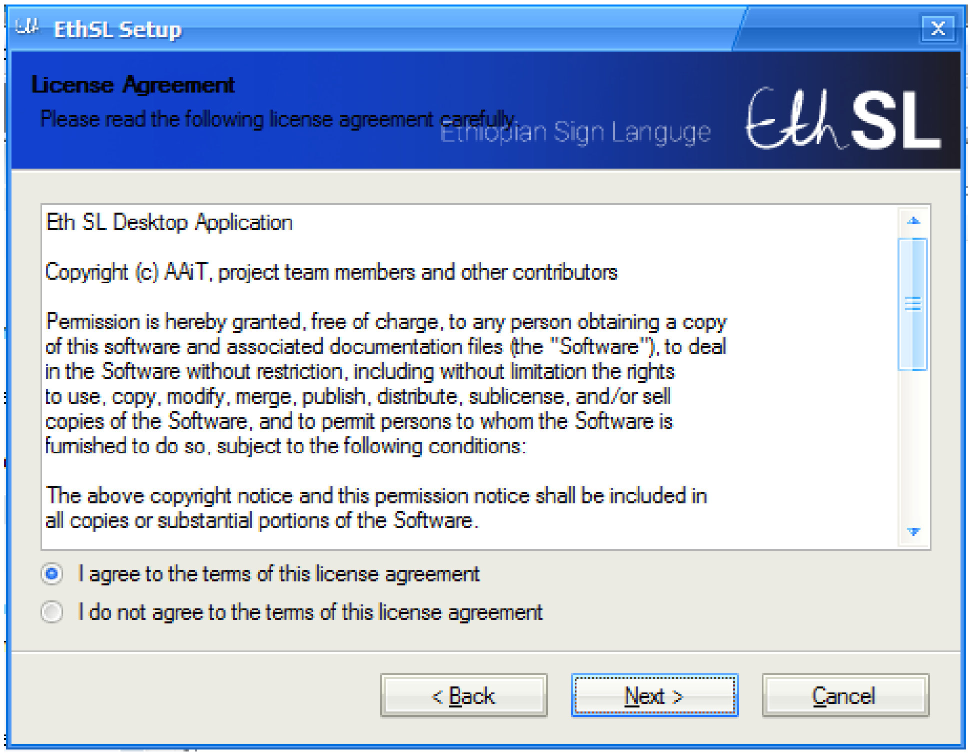
Welcome

Welcome to the installer for EthSL 1.0.

It is strongly recommended that you exit all Windows programs before continuing with this installation.

If you have any other programs running, please click Cancel, close the programs, and run this setup again.

Otherwise, click Next to continue.



License Agreement

Please read the following license agreement carefully.

Eth SL Desktop Application

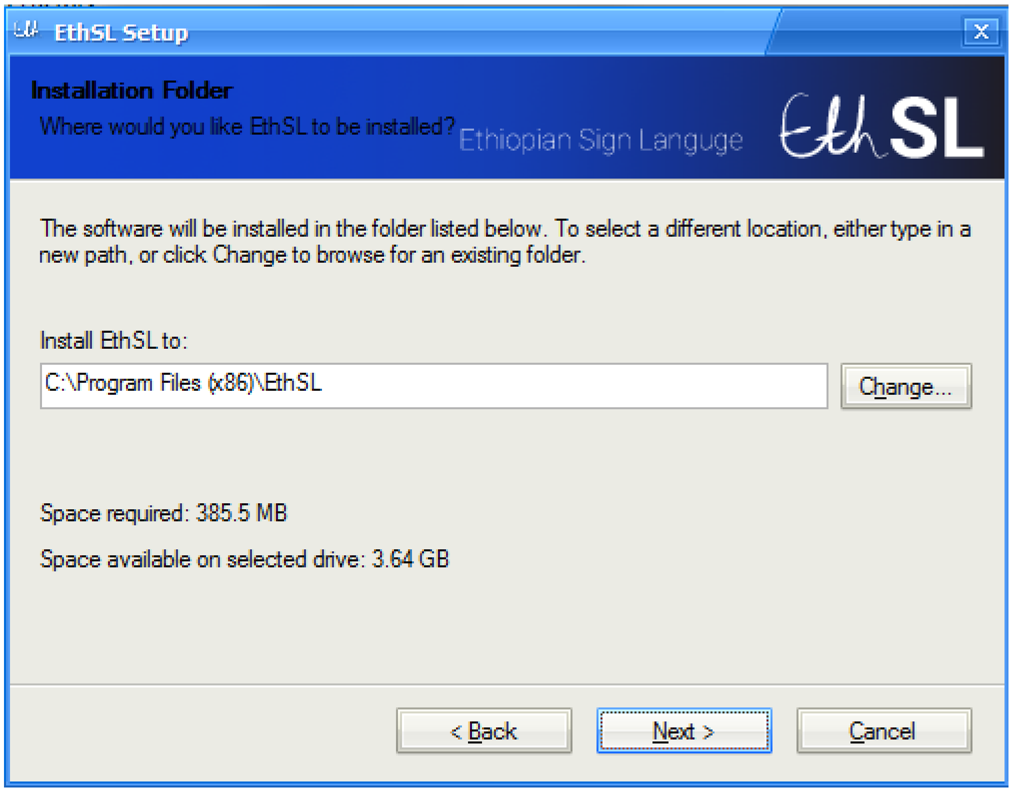
Copyright (c) AAiT, project team members and other contributors

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

[✓ I agree to the terms of this license agreement]

I do not agree to the terms of this license agreement



Installation Folder

Where would you like EthSL to be installed?

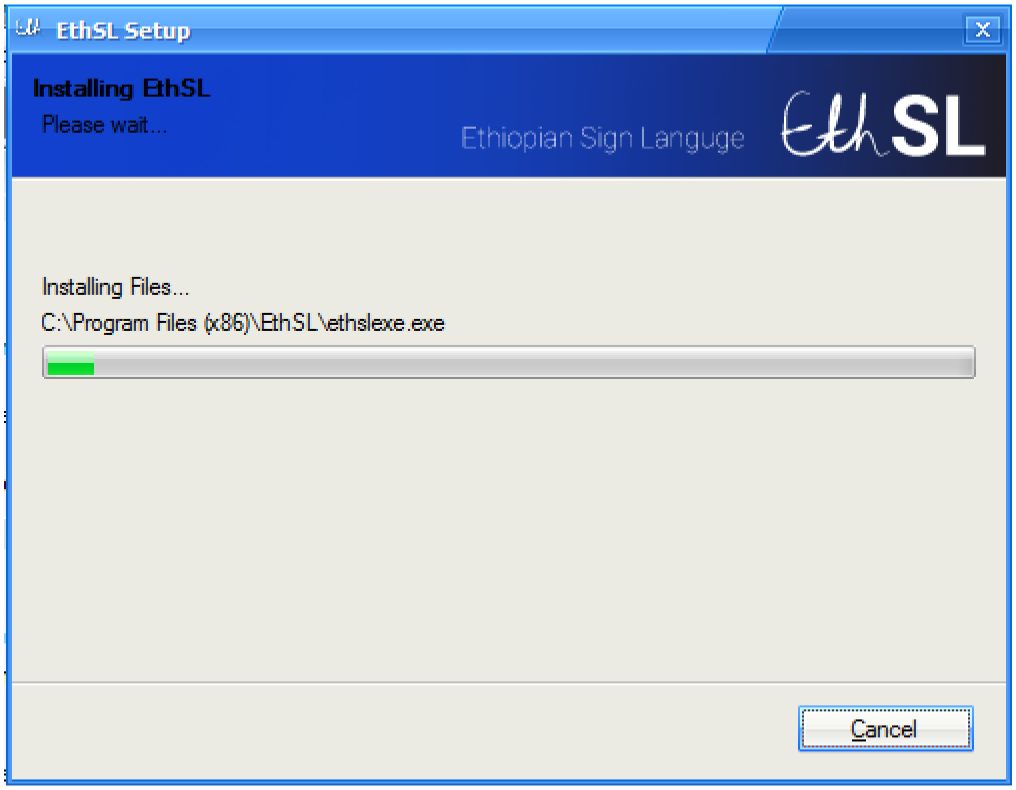
The software will be installed in the folder listed below. To select a different location, either type in a new path, or click Change to browse for an existing folder.

Install EthSL to:

C:\Program Files (x86)\EthSL

Space required: 385.5 MB

Space available on selected drive: 3.64 GB



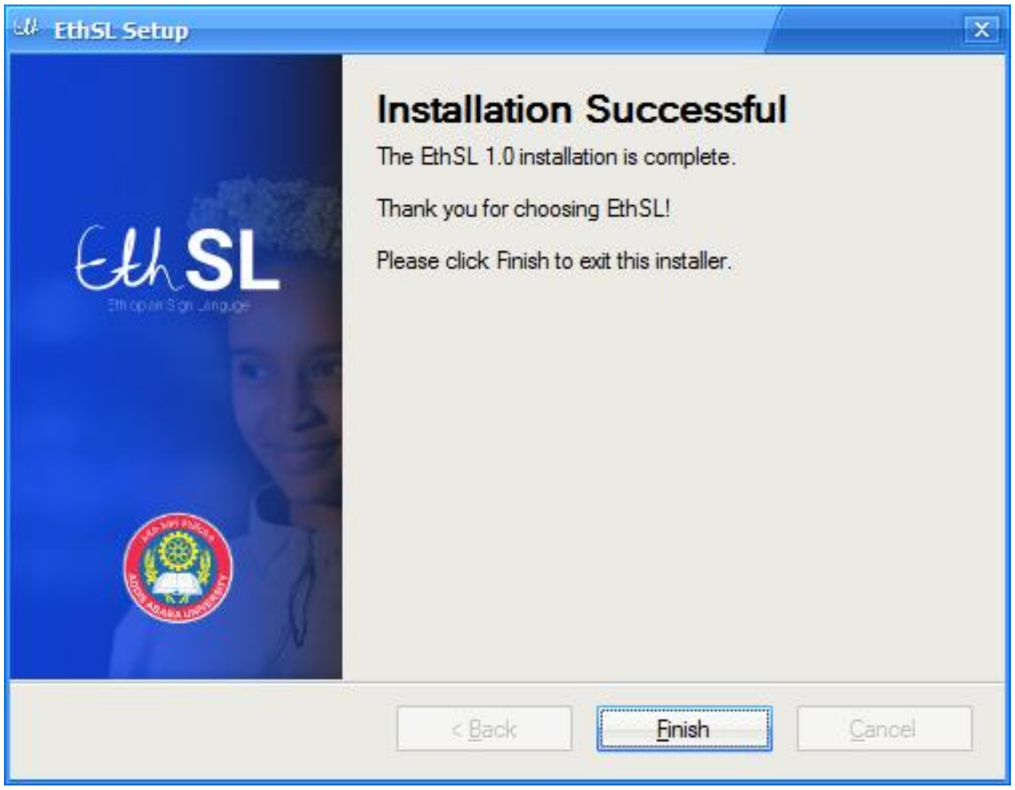
Installing EthSL

Please wait...

Installing Files...

C:\Program Files (x86)\EthSL\ethslexe.exe

3. Finish Installation



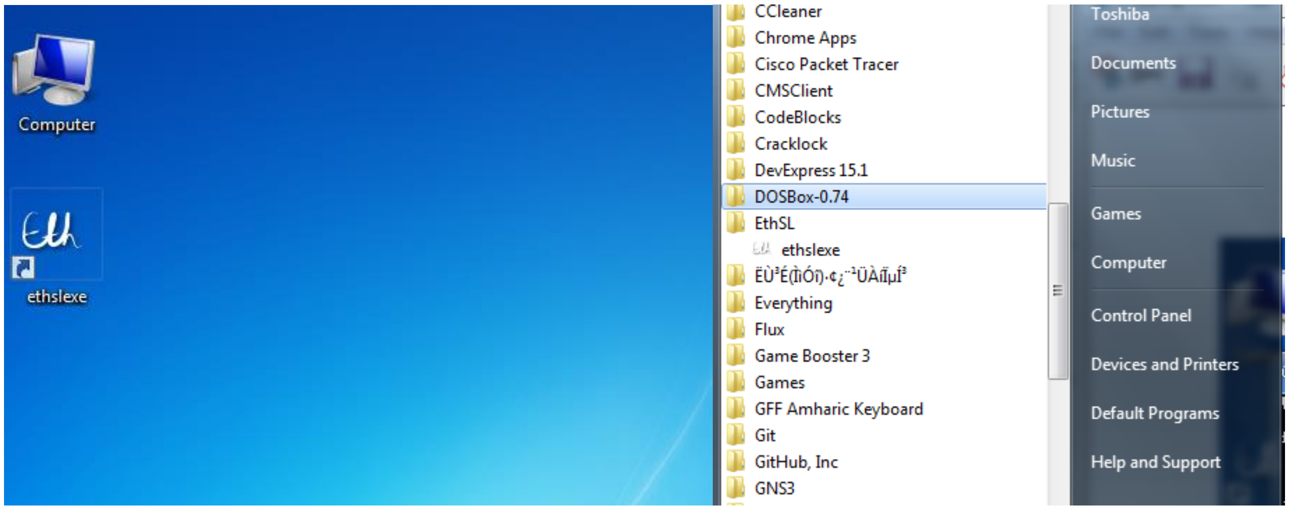
Installation Successful

The EthSL 1.0 installation is complete.

Thank you for choosing EthSL!

Please click Finish to exit this installer.

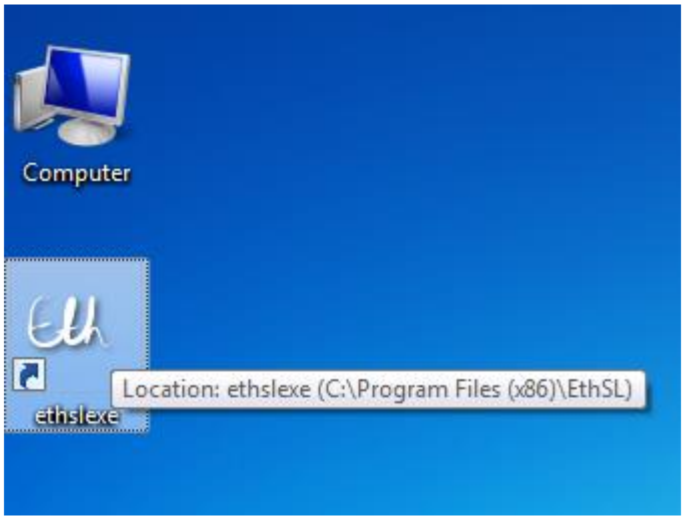
4. Shortcuts

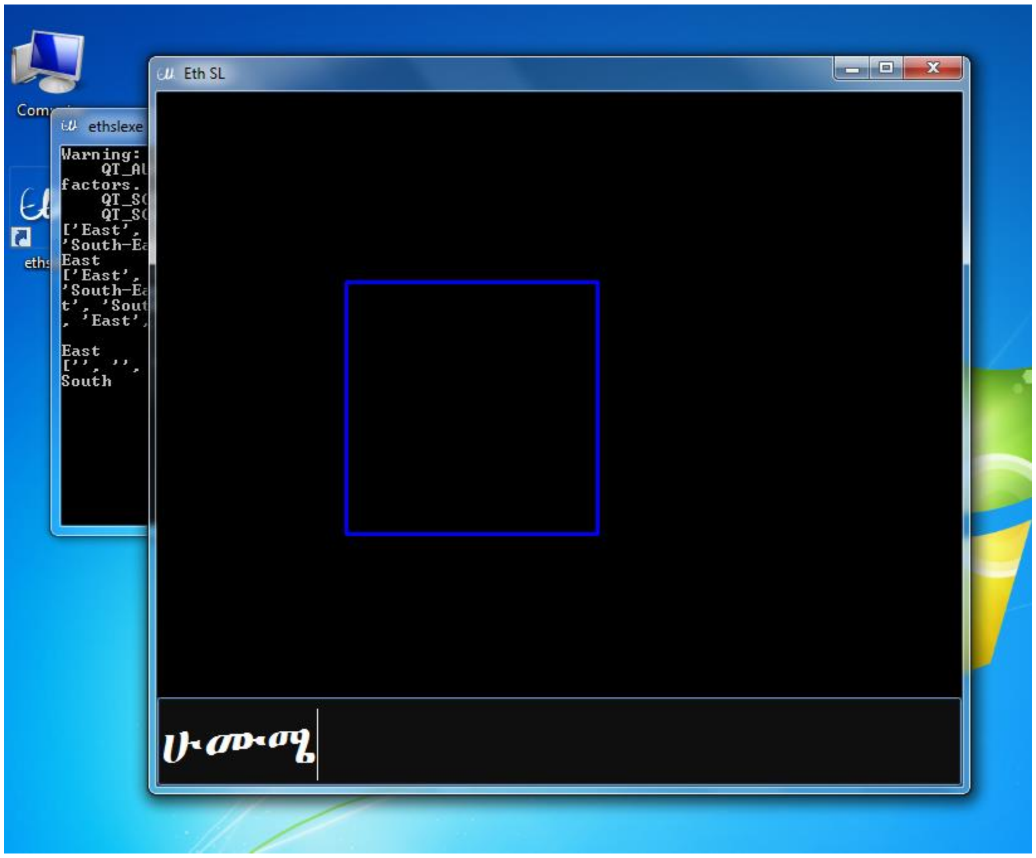


6.3 How to operate the system

A. How to start the system

Double Click on the desktop shortcut



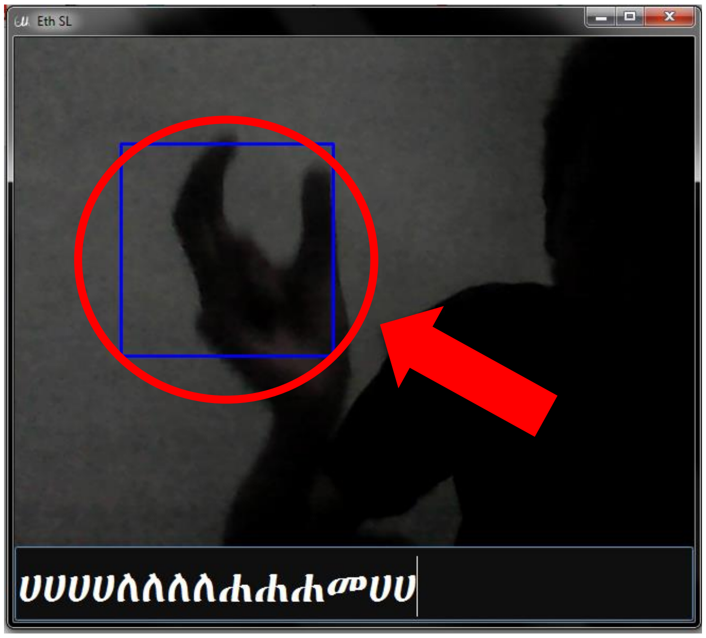


```
Warning:
    QT_AU
factors.
    QT_SC
    QT_SC
['East',
'South-Ea
East
['East',
'South-Ea
t', 'Sout
, 'East',

East
['', '',
South
```

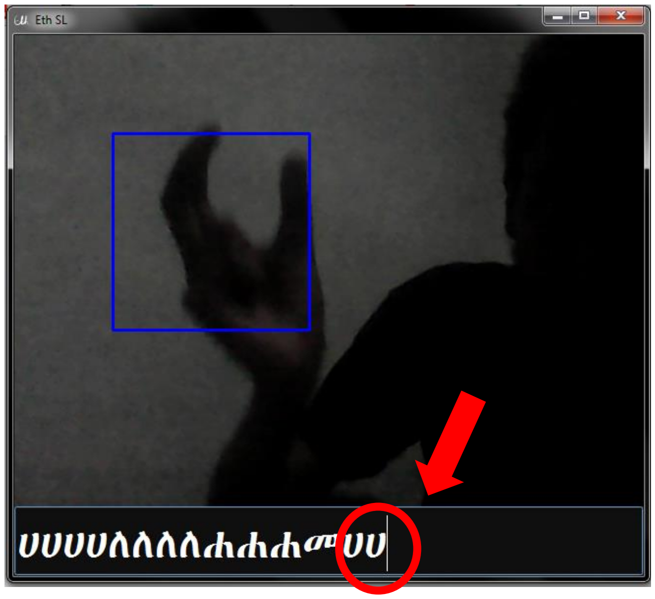
B. How to write consonants

1. Put your hand in the rectangle signing an Eth SL character



ሀሀሀሀለለለለሐሐሐመሀሀ

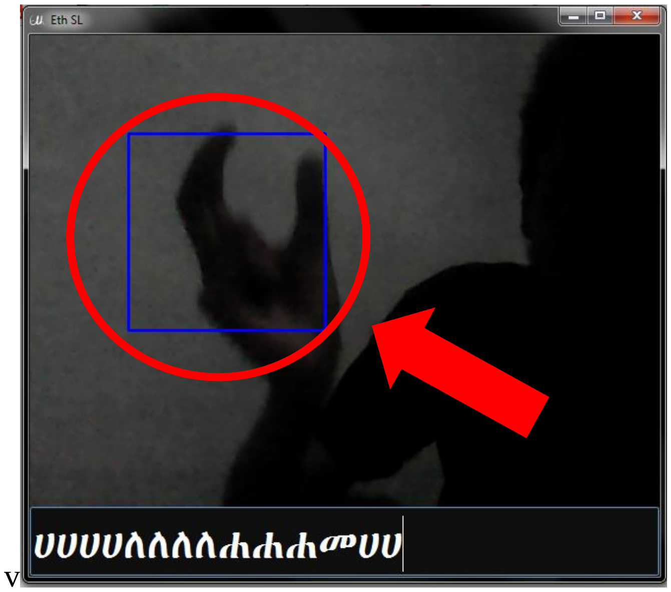
2 Then, keep hands still until letter appears in the text box



ሀሀሀሀለለለለሐሐሐመሀሀ

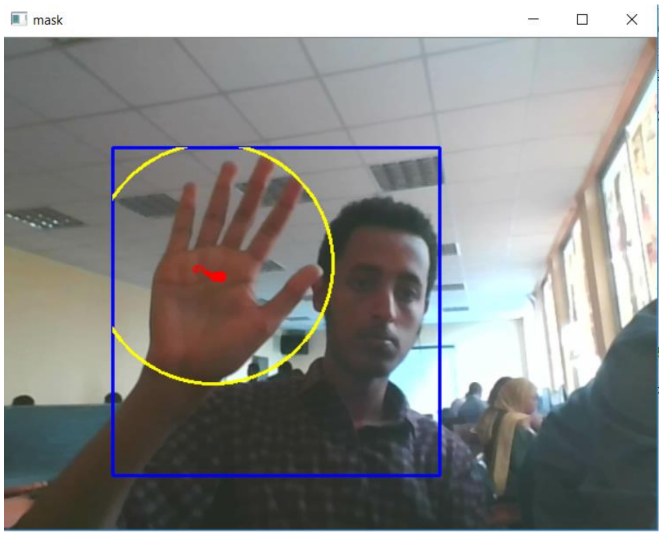
C. How to write vowels

1. Put your hand in the rectangle

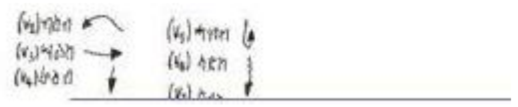
v 

ሀሀሀሀለለለለሐሐሐመሀሀ

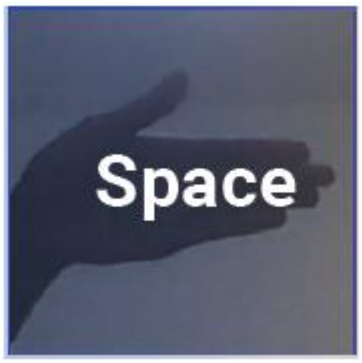
2. Move hands according to the vowel

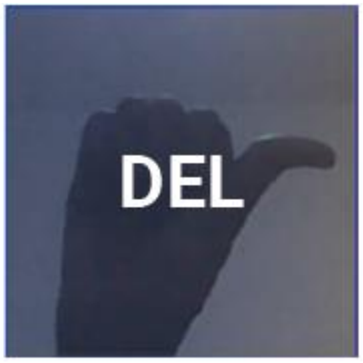


Vowel Movements



Space and Delete





Use these custom signs to put a space in between characters and delete characters

Chapter 7: CONCLUSION AND RECOMMENDATION

7.1 Conclusion

This project provides a way to translate Ethiopian sign language finger spelling to text. In an everyday scenario this would have been done using a translator. This system allows the user to translate sign language from their laptop or desktop (which has a webcam). In the software we have implemented both the recognition of consonants and vowels. Due to the time constraints the amount to data gathered to train our system was limited.

We believe that this project paves a way in removing the barrier that language creates. Especially between the disabled, whose problems are mostly overlooked, and the community.

7.2 Recommendation

To get the optimum performance from the software make sure to use it under good lighting conditions. The quality of the webcam also matters.

The code and dataset of this project will be publically available to the developers' community in hopes of encouraging development on this project. This project can be further pushed to include words and phrases of the Ethiopian sign language.

BIBLIOGRAPHY

[1] "Deafness and Hearing Loss". *World Health Organization*. N.P., 2017. Web. 2 Nov. 2017.

[2] "Census 2007". *Central Statistics Agency of Ethiopia*. N.P., 2017. Web. 2 Nov. 2017.

[3] *Baseline Study On The Status Of Persons With Disabilities And The Influence Of The African Decade Pronouncement In Ethiopia*. 1st ed. FDRE Ministry of Labor and Social Affairs, 2010. Print.

[4] "Sign Language". *En.wikipedia.org*. N.P., 201. Web. 2 Nov. 2017.

[5] "Deaf Sign Language". *Ethnologue*. N.P., 2017. Web. 2 Nov. 2017.

[6] "Ethiopian Sign Languages". *En.wikipedia.org*. N.P., 2017. Web. 2 Nov. 2017.

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

[8] Vista Wide, "World Language and Culture", Available at *http://www.vistawide.com /languages/top_30_languages.htm, Last accessed on 2 Nov. 2017.*

[9] *Dagnachew Feleke, "Machine Translation System for Amharic Text to Ethiopian Sign Language", Unpublished Master's Thesis, Addis Ababa University, 2011.*

[10] *Legesse Zerubabel, "Ethiopian Finger Spelling Classification", Unpublished Master's Thesis, Addis Ababa University, 2008.*

[11] *Eyob Gebretinsae,"Vision Based Finger Selling Recognition for Ethiopian Sign Language", Unpublished Master's Thesis, Addis Ababa University, 2012.*

[12] *Yonas Admasu and Raimond Kumudha, "Ethiopian Sign Language Recognition using Artificial Neural Network", 10th International Conference on Digital Object Identifier, pp 995-1000, 2010.*

[13] Mahendran, "Skin tone detection Mathlab code", Availble at: *http://mahendranmathan.blogspot.com/2013/01/skin-tone-detection-matlab-code.html,Last accessed on January 12,2013.*

[14] *Tefera Gimbi, " Recognition of Isolated Signs in Ethiopian Sign Language", Masters Thesis, Addis Ababa University, 2014.*

[15] *Spiral Development Model Spiral development https://en.wikipedia.org/wiki/Spiral_model*

APPENDIX

A.1 Appendix 1

Artificial neural networks (**ANNs**): are computing systems inspired by the biological neural networks that constitute animal brains. Such systems learn (progressively improve performance) to do tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the analytic results to identify cats in other images. They have found most use in applications difficult to express in a traditional computer algorithm using rule-based programming. [7]

A.2 Appendix 2

SOLID: is an acronym for the first five object-oriented design principles by Robert C. Martin [4]. These principles, when combined together, make it easy for a programmer to develop software that are easy to maintain and extend. The five principles are summarized in the table below [8]:

S – SRP	The Single Responsibility Principle	A class should have one, and only one, reason to change.
O – OCP	The Open Closed Principle	You should be able to extend a classes behavior, without modifying it.
L – LSP	The Liskov Substitution Principle	Derived classes must be substitutable for their base classes
I – ISP	The Interface Segregation Principle	Make fine grained interfaces that are client specific.
D – DIP	The Dependency Inversion Principle	Depend on abstractions, not on concretions.

A.3 Appendix 3

GPLv3: is an open-source license in which has permissions conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights [9].