A Project Report

On

# Two-Way Sign Language Converter for Speech-Impaired

Submitted in partial fulfilment of the requirement of

**University of Mumbai**

For the Degree of

**Bachelor of Engineering**

*in*

**COMPUTER ENGINEERING**


*Submitted by*

**Ankita Shinde**
**Raveena Dandona**

*Supervised by*

**Mrs. Rakhi Kalantri**




**Department of Computer Engineering**

**Fr. Conceicao Rodrigues Institute of Technology**
**Sector 9A, Vashi, Navi Mumbai - 400703**


**UNIVERSITY OF MUMBAI**

**2019-2020**

# APPROVAL SHEET

This is to certify that the project entitled

**"Two-Way Sign Language Converter for Speech-Impaired"**

**Submitted by**

**Raveena Dandona 101611**
**Ankita Shinde 101650**

**Supervisors :** _____

**Project Coordinator :** _____

**Examiners : 1.** _____

**2.** _____

**Head of Department :** _____

**Date :**

**Place :**

# Declaration

We declare that this written submission for B.E. Declaration entitled "**Two-Way Sign Language Converter for Speech-Impaired**" represent our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declared that we have adhere to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas / data / fact / source in our submission. We understand that any violation of the above will cause for disciplinary action by institute and also evoke penal action from the sources which have thus not been properly cited or from whom paper permission have not been taken when needed.

**Project Group Members:**

1. Raveena Dandona, 101611

_____

2. Ankita Shinde, 101650

_____

# Abstract

Sign language is a visual language that is used by deaf and dumb people as their mother tongue. It is achieved by simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions.

This project is based on converting the audio signals received to text using speech to text-api (python modules or google-api) and then using the semantics of Natural Language Processing to breakdown the text into smaller understandable pieces which requires Machine Learning as a part. Data sets of predefined sign language are used as the input so that the software can use artificial Intelligence to display the converted audio into the sign language.

The next objective is to translate sign language to text/speech. The framework provides a helping-hand for speech-impaired to communicate with the rest of the world using sign language. This leads to the elimination of the middle person who generally acts as a medium of translation. This would contain a user-friendly environment for the user by providing speech/text output for a sign gesture input.

Thus, the project aims at converting sign language to audio/text as well as audio to sign language for speech impaired.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Background

The Sign Language is the primary mode of communication for deaf people. It helps them to communicate by gestures rather than speaking. It is a major linguistic challenge which many researchers have taken up and built systems to facilitate the process of understanding the natural language by means of sign language. If a vocal person is unaware of sign language, communication between hearing impaired people and vocal person becomes difficult and is then, mostly achieved through a sign interpreter. It is always not possible to have an interpreter when a person is unaware of sign language. So, it becomes highly important to develop a system which can generate signs for a particular word.

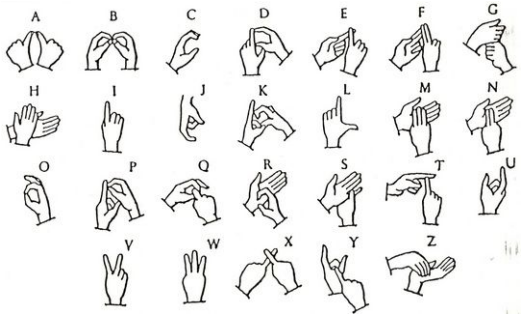


Figure 1.1: Indian Sign Language Gestures - Alphabets

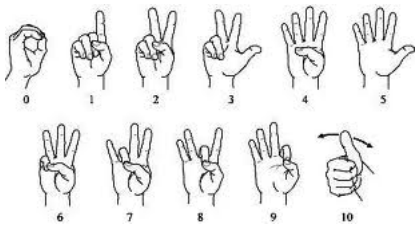


Figure 1.2: Indian Sign Language Gestures - Numbers

## 1.2 Motivation

To put numbers in context, a close look at the census 2011, reveals that of the 13.4 million people with disabilities in India in the employable age group of 15-59 years, 9.9 (73.9 percent) million were non-workers or marginal workers. Which means that only 26.1 percent of the productive age group of the country is employed. People with hearing disabilities face many issues, when it comes to employment.

31 percent of people feel they are treated differently because of their deafness, hearing loss and tinnitus

33 percent of people who are deaf avoid social situations because they find it difficult to communicate

68 percent of people with hearing loss feel isolated at work as a result of not being able to communicate

## 1.3 Aim and Objective

Aim: With advancements in computers and technology it has become possible to develop an automatic interpreter which can be used to convert Speech to Text which can then further be translated to Sign Language Gestures and can be accessed globally. A government website called www.indiansignlanguage.org has been launched for empowering the deaf, which presents a huge database of Indian Sign Language (ISL) signs. Very less work has been done on Indian Sign Language so far. That is why it would be very useful to have an automatic Indian Sign Language Interpretation system.

We aim to complete the Phase-I of the project which deals with converting gestures to text using CNN.

Objective: For this objective NLP can be used to convert audio to text and the text can be used to generate relevant videos of sign language using python libraries like Moviepy and Pygame. The next objective is to develop sign language to audio converter using gesture recognition and classification and converting it to text and eventually output in the form of the relevant audio.

## 1.4 Report Outline

Many research papers on the same project were read to understand more about this project. A comparison was made to understand the advantages and disadvantages of using different deep learning algorithms. We con-

cluded that Convolution Neural Network is the best algorithm as it gives the highest accuracy. Different designs were made to understand the flow of our project. First, a self-created dataset with reference from the ASL database. No detailed data preprocessing is needed as the images were clear. After this, we will train the dataset using CNN and the last step is prediction. Our model was able to predict the 44 characters in the ASL.

# Chapter 2

# Study of the System

## 2.1 Introduction

In this chapter the relevant techniques in literature of sign-language and gesture display though various technologies is detailed. It describes various techniques used in the work. We identified the current literature on related domain problem. The techniques that have been developed and present the various advantages and limitation of these methods used extensively in the table below. Also, the advantages as well as disadvantages have been enumerated in the table.

This review is an objective, critical summary of published research literature relevant to a topic under consideration for research. Its purpose is to create familiarity with current thinking and research on a the sign language converter, and may justify future research into a previously overlooked or understudied area.

## 2.2 Main Body

We propose a two-way sign language interpreter system that converts audio to sign language and vice versa without the use of any specialized hardware.

1. For the first objective we will develop sign language to audio converter. This will be achieved using gesture recognition and classification for identifying the sign.

2. Once identified it will be converted to text and eventually the output in the form of the relevant audio.

3. For the next objective we will develop audio to sign language converter.

4. The audio signals received will be converted to text using speech-to-text api (python modules or google api).

5. Then using the semantics of CNN to breakdown the text into smaller understandable pieces.

## 2.3 About the Technique

We propose a two-way sign language interpreter system that converts audio to sign language and vice versa without the use of any specialized hardware. For the first objective we will develop sign language to audio converter. This will be achieved using gesture recognition and classification for identifying the sign. Once identified it will be converted to text and eventually the output in the form of the relevant audio. For the next objective we will develop audio to sign language converter. The audio signals received will be converted to text using speech-to-text api (python modules or google api). Then using the semantics of CNN to breakdown the text into smaller understandable pieces.

**Convolution Layer:** This layer performs a dot product between two matrices, where one matrix is the set of learn able parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image, but is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels. During the forward pass, the kernel slides across the height and width of the image producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called stride.

**Pooling Layer:** The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

**Fully connected Layer:** Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular Fully Connected Neural Network. This is why it can be computed as usual

by a matrix multiplication followed by a bias effect. The Fully Connected Layer helps map the representation between the input and the output.

## 2.4 Various Available Technique

#### 2.4.0.1 Digital Signal Processing

In DSP[1], we store the number of pixels (frequencies) of same intensity values into a histogram array, which is commonly called "bin". For an 8-bit grayscale image, the size of histogram bin is 256, because the range of the intensity of 8-bit image is from to analyze the brightness and contrast of an image. It shows how the intensity values of an image is distributed and the range of brightness from dark to bright.

#### 2.4.0.2 Natural Language Processing

It is an 8-step process[6][7]. The first step is sentence segmentation. The second step is Word Tokenization and the third step is predicting parts of speech for each token. The next step is text lemmatization. The fifth step is Identifying stop words. The sixth step is dependency parsing. The seventh step is Name Entity Recognition and the final step is Conference resolution.

#### 2.4.0.3 Support Vector Machine

There are many folds advantages of using the supervised learning approach of Support Vector Machine (SVM). They are very effective when we have very high dimensional spaces. Also, when number of dimensions becomes greater than the existing number of samples, in such cases too SVM is found to be very effective. SVM uses a subset of training point also known as support vectors to classify different objects hence it is memory efficient. Support Vector Machines are versatile, for different decision function we can define different kernel as long as they provide correct result. Depending upon our requirement and application we can choose types of kernel which is most productive for our application.

#### 2.4.0.4 Convolutional Neural Network

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of neural networks[4].

that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid-like fashion that contains pixel

values to denote how bright and what color each pixel should be. A CNN typically has three layers: a convolutional layer, pooling layer, and fully connected layer.

## 2.5 Related Works

The comparison table presents inferences drawn from papers researched on similar domains. Comparative study of Sign Language Converter

| Sr.no | Paper | I/p and o/p Algorithm used | Accuracy Issues |
|---|---|---|---|
| 1 | **Corneliu Lungociu,*Real Time Sign Language Recognition Using Artificial Neural Networks*, Studia Univ. Babes_Bolyai, Informatica, Volume LVI, Number 2011.** | A neural network-based approach for the sign language recognition with recognition | The accuracy achieved is 80%. |
| 2 | **Kanchan Dabre, Surekha Dholay. *Machine Learning Model for Sign Language Interpretation using Webcam Images*. 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA).** | -Haar Cascade classifier for classification.<br>-Microsoft .NET framework for speech synthesis. | The speech synthesis phase of sign recognition process sometimes gives delayed response. |
| 3 | **Hasan, M., Sajib, T. H., & Dey, M. (2016). *A machine learning based approach for the detection and recognition of Bangla sign language*. 2016 International Conference on Medical Engineering, Health Informatics and Technology (MediTec).** | -Hand Gesture recognition is performed using HOG (Histogram of Oriented Gradients).<br>-SVM (Support Vector Machine) used as classifier. | 86.53% accuracy for only 16 predefined static gestures. |
| 4 | **Rajaganapathy, S., Aravind, B., Keerthana, B., & Sivagami, M. (2015). *Conversation of Sign Language to Speech with Human Gestures. Procedia Computer Science, 50, 10–15.*** | Microsoft's Kinect sensor with program developed on .NET platform. | -The gesture tracking is limited only to 2 individuals.<br>-The sensor cannot recognize the human objects beyond 40cm to 4m range. |

Table 2.1: Comparative study of Sign Language Converter

# Chapter 3

# Proposed System

## 3.1 Problem-Statement

We aim to develop a project, that achieves two-way sign language interpreter (sign language to audio and vice versa) without using any specialized hardware.

## 3.2 Scope and Proposed System

The scope of the project involves developing a Two-Way Sign Language Converter for Speech-Impaired by the application of Deep Learning algorithm. This project is based on converting the audio signals received to text using speech to text-api (python modules or google-api) and then using the semantics of Natural Language Processing to breakdown the text into smaller understandable pieces which requires Machine Learning as a part. Data sets of predefined sign language are used as the input so that the software can use artificial Intelligence to display the converted audio into the sign language.The next objective is to translate sign language to text/speech. The framework provides a helping-hand for speech-impaired to communicate with the rest of the world using sign language. This leads to the elimination of the middle person who generally acts as a medium of translation. This would contain a user-friendly environment for the user by providing speech/text output for a sign gesture input.All the work is planned to be done in the course of one year.

We propose a two way sign language interpreter system that converts audio to sign language and vice versa without use of any specialized hardware. For the first objective we will develop sign language to audio converter which has been illustrated in Figure .This will be achieved using gesture recognition and classification for identifying the sign. Once identified it will

be converted to text and eventually the output in the form of the relevant audio.

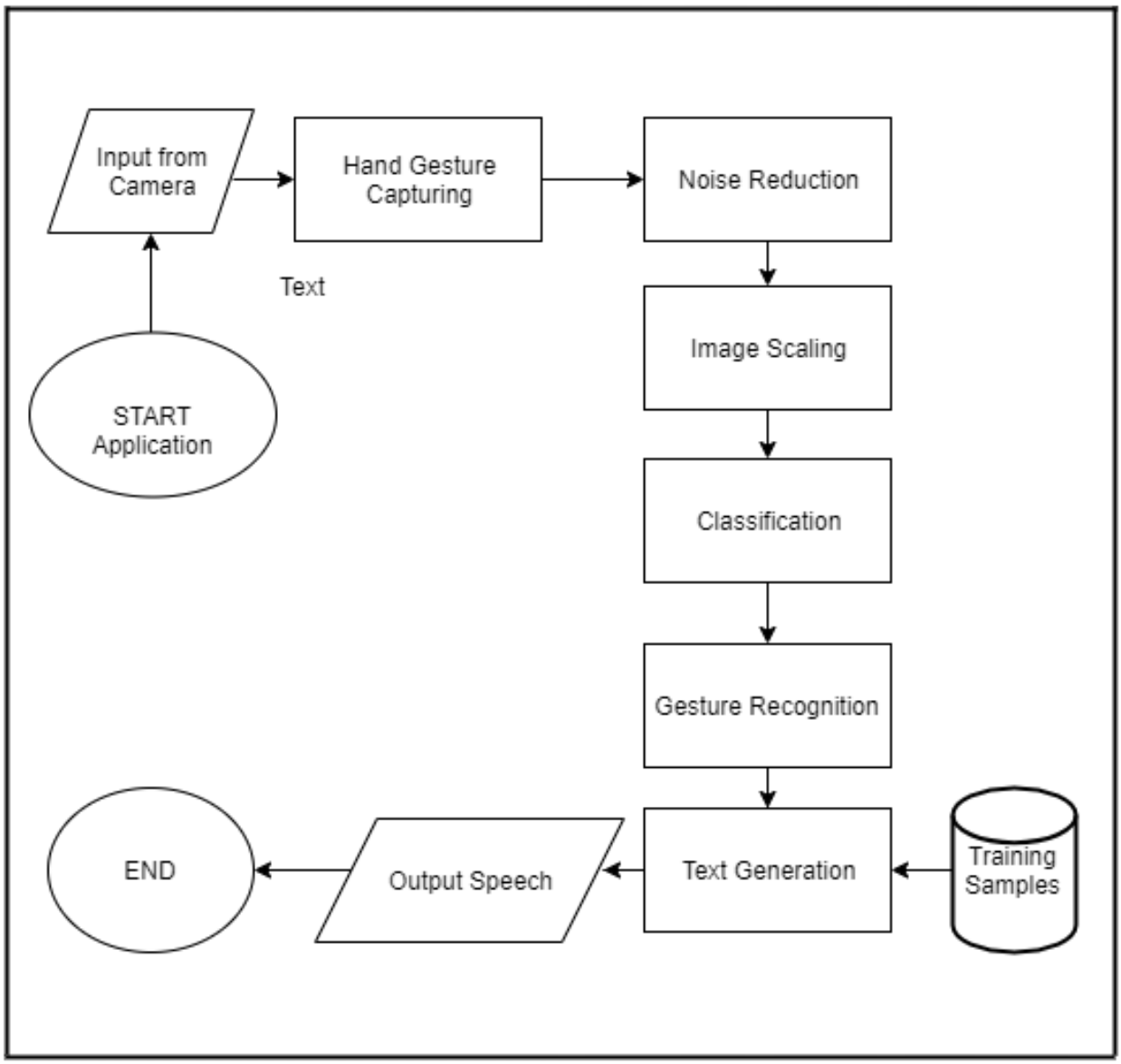


Figure 3.1: Sign to speech converter

# Chapter 4

# Design of the System

## 4.1 Requirement Engineering

### 4.1.1 Requirement Elicitation

The Sign Language is the primary mode of communication for deaf people. It is a major linguistic challenge which many researchers have taken up and built systems to facilitate the process of understanding the natural language by means of sign language. If a vocal person is unaware of sign language, communication between hearing impaired people and vocal person becomes difficult and is then, mostly achieved through a sign interpreter. It is always not possible to have an interpreter when a person is unaware of sign language. So, it becomes highly important to develop a system which can generate signs for a particular word. Figure shows a use case diagram which is a dynamic behavior diagram in UML. The use case diagram models the functionality of our system using actors as the physically impaired and use cases including sign to speech and speech to sign converters as elaborated in the previous chapters. Use cases are a set of actions,services, and functions that the system aims to perform.

### 4.1.2 Software life cycle model

Agile software model is used for developing software applications where project implementation is done iteratively or incrementally. This model helps to make changes or modification as per the user requirement for different traffic scenarios . The cycle stages are executed in parallel. The system explained in the report has been developed based on the agile framework model.

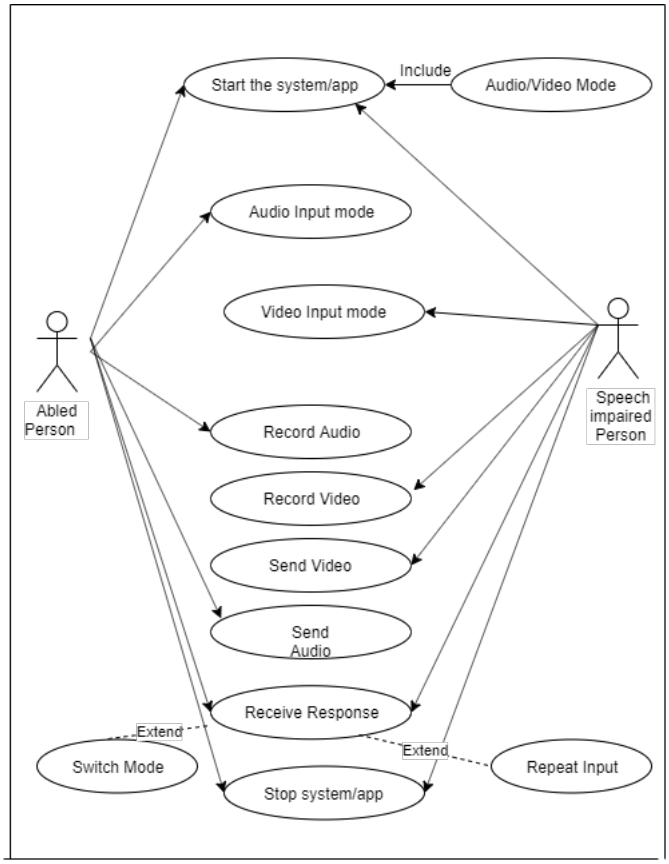


Figure 4.1: Use Case Diagram

### 4.1.3 Requirement Analysis

The system is expected to be used by the hearing and speech impaired people as well as normal people.The people with disability will record their hand gestures which will be converted to text and also speech so that the normal people can easily interpret it. The words or sentences spoken by normal people will be converted to text and then followed by converting this text into corresponding sequence of hand gesture images. These hand gesture images will help the speech impaired person to interpret the message faster. Thus, the system will help to bridge the communication gap.

#### 4.1.3.1 Data Flow Diagram

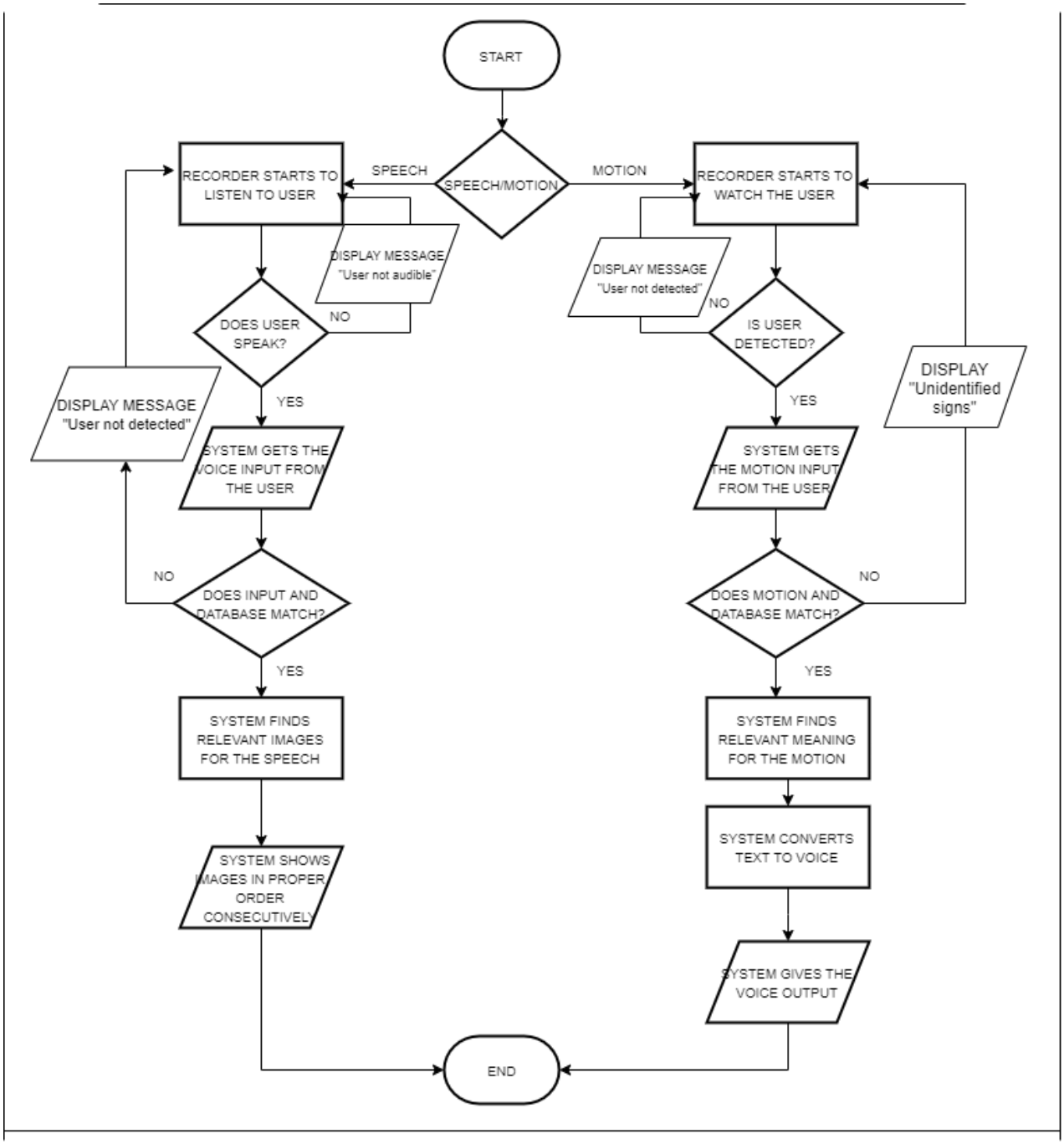


Figure 4.2: Data Flow Diagram

#### 4.1.3.2 Cost Analysis

The current scope of the project deals with only it's implementation locally in one system.The future scope may include it's implementation on user mobile phones which will require servers to host the software as well as a good processor to run the system efficiently. As of now, considering that this project is completely software based and laptop/computer is already available, all the software required to implement the project is currently available free of cost.

#### 4.1.3.3 Hardware and software requirement

Software Requirements

- Python 3.3+(required)
- SpeechRecognizer 3.6.3
- Tensorflow 1.11.0
- OpenCV
- tkinter
- Jupyter notebook
- PyAudio 0.2.9+

Hardware Requirements

- Microphone
- Camera (currently Webcam)
- 32bit(x86) or 64bit(x64) Dual-core 2.66-GHZ or faster processor
- A minimum of 8 GB of RAM
- GTX 1050 Ti GPU (minimum)

## 4.2 System architecture

### 4.2.1 Block Diagrams

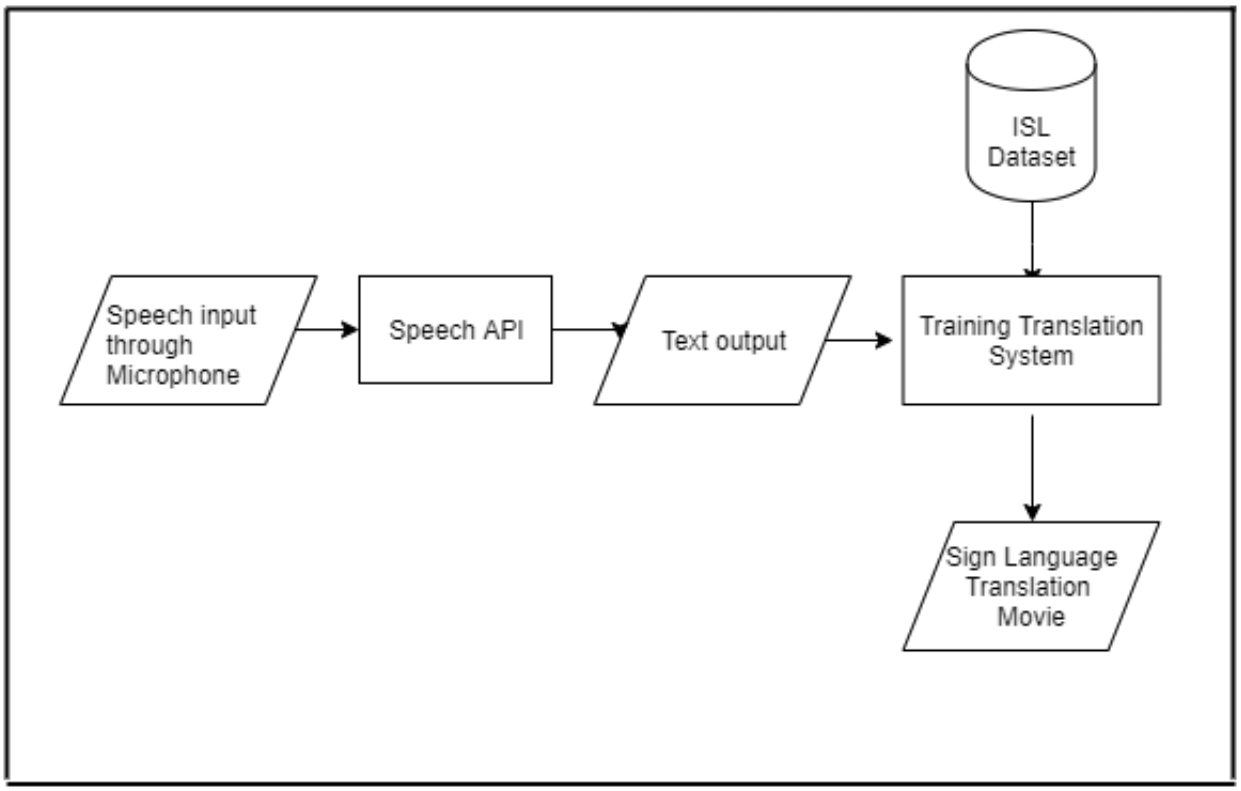


Figure 4.3: Speech to Sign Language Converter

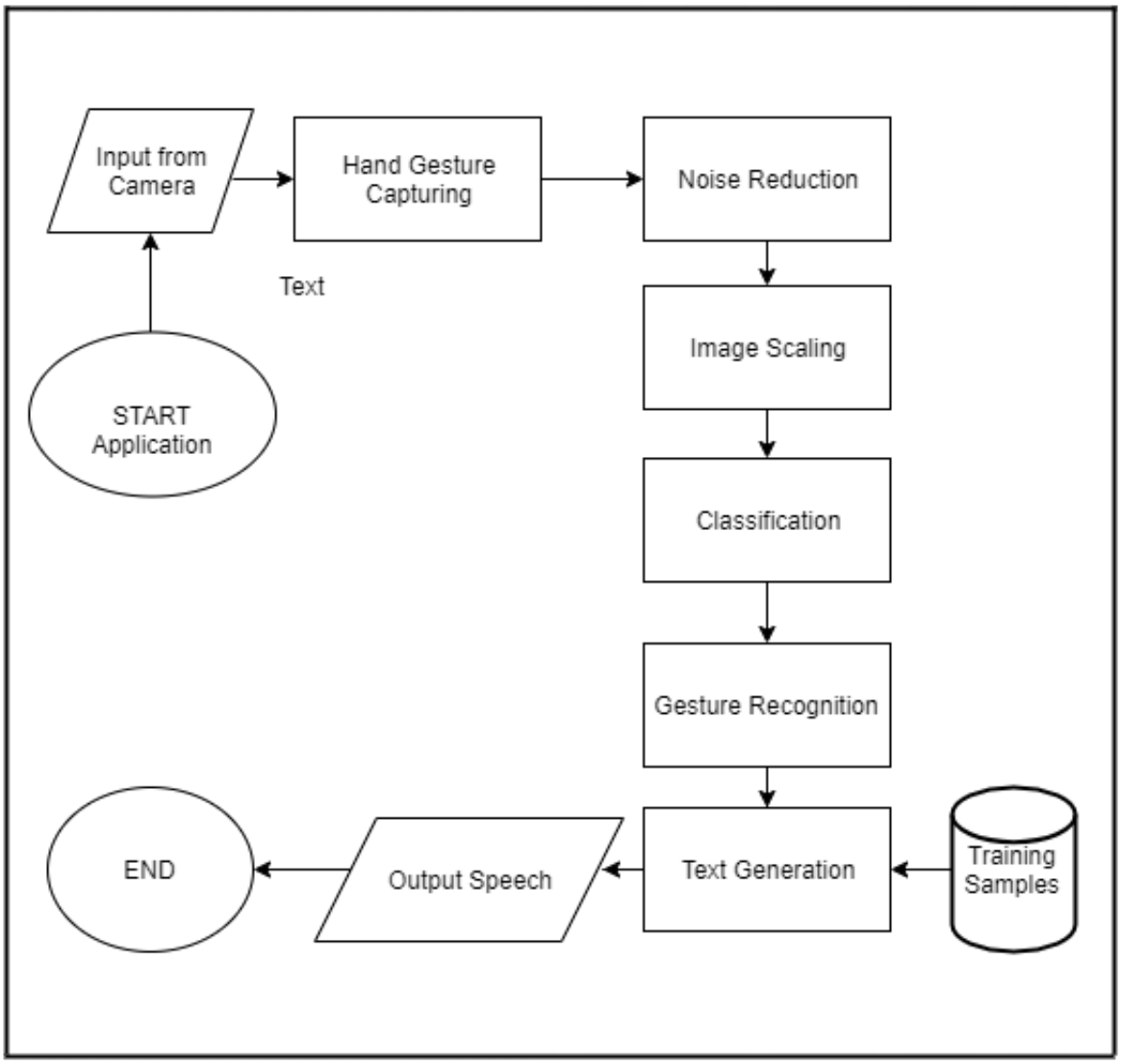


Figure 4.4: Sign Language to Speech Converter

# Chapter 5

# Result and Discussion

## 5.1 Screenshots of the System

Below are the screenshots of the system working in 2 phases. First is the Sign language to Text as well as speech. The next screenshot is of Speech to Sign language conversion in the form of image gestures.

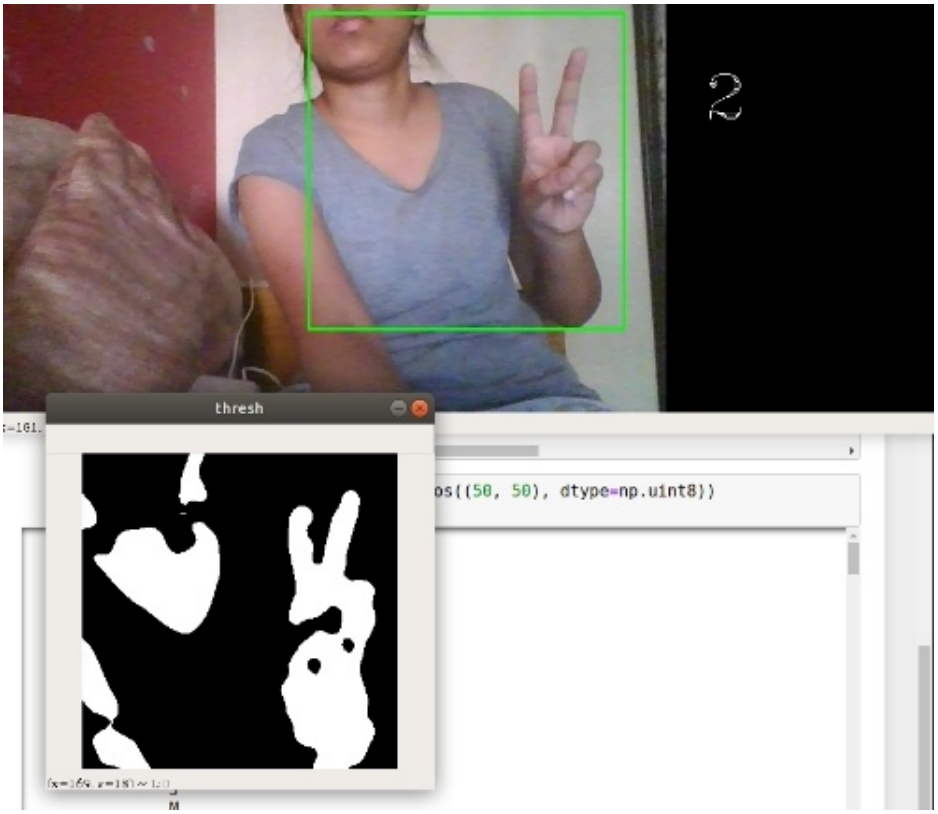


Figure 5.1: Phase 1 Output


Figure 5.2: Phase 2 Output for How are you

## 5.2 Sample Code (of imp part/ main logic)

Below are the screenshots of the code used in implementation of phase 1 and phase 2.

## Phase 1 - dataCollection.py

```
Enter the gesture name: you
```

In [14]:

```
cam=cv2.VideoCapture(0)
# define range of skin color in HSV
lower_skin = np.array([0,30,70], dtype=np.uint8)
upper_skin = np.array([20,255,255], dtype=np.uint8)
kernel = np.ones((5,5),np.uint8)
while i<=1000:
    ret,frame=cam.read()

    #drawing rectange on the frame
    cv2.rectangle(frame,(154,165),(434,445),(0,255,0),2)
    cv2.imshow('Test',frame)

    #getting the dimensions of rectangle for cropping
    x=154
    w=280
    y=165
    h=280

    #cropping only for gesture
    cropped1=frame[y+10:y+h,x+10:x+w]
    #cropped=cv2.resize(cropped,(150,150))

    #preprocessing
    roi2=cv2.cvtColor(cropped1,cv2.COLOR_BGR2HSV)
    #extract skin colur imagw
    roi2 = cv2.inRange(roi2, lower_skin, upper_skin)
    cv2.imshow("ROI", roi2)

    roi =cv2.bitwise_and(cropped1,cropped1,mask=roi2)
    roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
    cv2.imshow("Col", roi)
    cropped=cv2.resize(roi,(150,150))

    if cv2.waitKey(1)& 0xFF ==27: #escape key
        break

    if cv2.waitKey(1)& 0xFF == 13:


        #saving the image to relevent gesture folder
        cv2.imwrite('./dataset/samples/'+gesture+'.'+str(i)+'.jpg',cropped)


        #flipping the image and saving it
        flipped=cv2.flip(cropped,1)
        cv2.imwrite('./dataset/samples/'+gesture+'.'+str(1000+i)+'.jpg',flipped)

        i+=1
```

## Phase 1 - preprocessing.py

***Loading images***

In [1]:
```
from os import listdir
from os.path import isfile, join

mypath="./dataset/samples/"

file_names= [f for f in listdir(mypath) if isfile(join(mypath,f))]

print(str(len(file_names))+' images loaded')
```

```
10000 images loaded
```

In [6]:
```
print(file_names[1].split('.')[0])
```

```
you
```

***Splitting data into test and train datasets***

In [3]:
```
import cv2
import numpy as np
import sys
import os
import shutil

#total 2000 images are present for each class
#1250 for training data and 750 for testing data

good_count=0
peace_count=0
bad_count=0
you_count=0
wait_count=0

training_size=1000
test_size=500

training_images=[]
training_labels=[]
test_images=[]
test_labels=[]

good_dir_train="./dataset/train/good/"
peace_dir_train="./dataset/train/peace/"
bad_dir_train="./dataset/train/bad/"
you_dir_train="./dataset/train/you/"
```

## Phase 1 - training.py

```
Using TensorFlow backend.
```

In [16]:
```
#building model

model=Sequential()
model.add(Conv2D(32,(3,3),padding='same',activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(.25))


# model.add(Conv2D(64,(3,3),padding='same',activation='relu'))
# model.add(Conv2D(64,(3,3),activation='relu'))
# model.add(MaxPooling2D(pool_size=(2,2)))
# model.add(Dropout(.25))

model.add(Flatten())
model.add(Dense(256,activation='relu'))
model.add(Dropout(.5))
model.add(Dense(num_of_classes,activation='softmax'))

#compiling model
model.compile(loss='categorical_crossentropy',
              optimizer=SGD(.01),
              metrics=['accuracy'])
```

***TRAINING MODEL***

In [17]:
```
batch_size = 5
epochs=5
```

In [18]:
```
model.fit(x_train,y_train,
          batch_size=batch_size,
          epochs=epochs,
          validation_data=(x_test,y_test))

score=model.evaluate(x_test,y_test,verbose=1)
print('Test loss: ',score[0])
print('Test accuracy: ',score[1])
```

## Phase 1 - testing.py

```
_
=================================================================
Total params: 44,871,077
Trainable params: 44,871,077
Non-trainable params: 0
_________________________________________________________________
None
```

***SAVING MODEL***

In [20]:
```
model.save('./sign_language_model_5_classes.h5')
print('Model Saved')
```

```
Model Saved
```

***Visualising and Loading Model***

In [21]:
```
#visualising
from keras.utils.vis_utils import plot_model
plot_model(model,to_file='sign_language_model_vis.png',show_shapes=True,
    show_layer_names=True)
```

In [1]:
```
import imutils
#testing model
import cv2
import numpy as np
from keras.models import load_model
classifer=load_model('./sign_language_model_5_classes.h5')

inp_image=cv2.imread('./dataset/test_img4.jpg')
#inp_image = cv2.resize(inp_image,(150,150))
inp_image=np.array(inp_image,dtype='float')/255.0

cv2.imshow('img',inp_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
# inp_image=np.asarray(inp_image)
inp_image=inp_image.reshape(1,150,150,3)
# inp_image=inp_image.astype('float32')
# inp_image/=255

res=str(classifer.predict_classes(inp_image,batch_size=1)[0])
print(res)
```

```
Using TensorFlow backend.
```

## Phase 2 - speechRecognizer.py

In [2]:
```
def speak():
    r=sr.Recognizer()
    with sr.Microphone(device_index=8,chunk_size=3024) as source:
        #saving the voice input in variable audio
        audio=r.listen(source,timeout=5,phrase_time_limit=5)


        try:
            inputDialogue=r.recognize_google(audio)
#             label3=Label(root,text='You Said: '+inputDialogue,fg='blue',font=tk_font,bg='#fff')
#             label3.place(relx=.5,rely=.2,anchor=CENTER)
#             inputDialogue='good morning how are you'
            process_text(inputDialogue)
        except:
            label1=Label(root,text='COULD NOT RECOGNIZE',fg='blue',font=tk_font,bg='#fff')
            label1.place(relx=.5,rely=.2,anchor=CENTER)
            root.update()
            time.sleep(2)
            label1.destroy()
```

In [3]:
```
def process_text(inputDialogue):


    wordsSpoken=inputDialogue.split()
    numberOfWordsSpoken=len(wordsSpoken)
    #to lower case
    for i in range(numberOfWordsSpoken):
        wordsSpoken[i]=wordsSpoken[i].lower()


    for i in range(numberOfWordsSpoken):

        for words in glob.glob('./images/*'):
            actualWord=words.split("/")[2].split(".")[0]
            if actualWord !=wordsSpoken[i]:
                 continue
            else:

                label2=Label(root,text=actualWord,fg='blue',font=tk_font,bg='#fff')
                label2.place(relx=.5,rely=.25,anchor=CENTER)
                load = Image.open(words)
                render = ImageTk.PhotoImage(load)

                img = Label(root, image=render)
                img.image = render

                img.place(relx=.5, rely=.6, anchor=CENTER)
                root.update()
                time.sleep(3)
```

## 5.3 Testing

### 5.3.1 Unit Testing

The goal of unit testing to separate each part of the program and test that the individual parts are working correctly and as intended. Test Objective: Proper working of individual component.

| Test Condition | Input Specification | Output Specification | Success/Fail |
|---|---|---|---|
| Phase-1: Data set generation(including new dataset) | Running capture_gesture.py | Set of various gestures generated | Success |
| Phase-1:Image Preprocessing | Running preprocessing.py | Image dataset conversion by cropping, grayscale,hsv | Success |
| Phase-1: Trained network applied on sample image | Running testing.py | Image correctly classified | Success |
| Phase-2: Input Speech conversion to text | Running speech_to_text.py | Speech correctly recognized | Success |

Table 5.1: Unit Testing

### 5.3.2 Integration Testing

Combine the unit tested module one by one and test the functionality of the combined unit. Test Objective: To obtain proper result of individual phases.

| Test Condition | Input Specification | Output Specification | Success/Fail |
|---|---|---|---|
| Phase-1: Testing on input from webcam | Running testing.py | Conversion of gesture to text | Success |
| Phase-2: Testing Speech conversion to single image | Running recognizer.py | Conversion of speech to image from data-set | Success |

Table 5.2: Integration Testing

### 5.3.3 Black box Testing

In Black Box Testing, we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.
Test Objective: To obtain conversion of gesture to speech in phase 1 and conversion of speech to gesture images in phase 2.

| Test Condition | Input Specification | Output Specification | Success/Fail |
|---|---|---|---|
| Phase-1: Gesture to speech conversion for words | Running phase1.py | Speech output generated for observed gesture | Success |
| Phase-2: Speech to series of corresponding gesture images | Running speechRecognizer.py | Series of images generated for corresponding speech | Success |

Table 5.3: Blackbox Testing

# Chapter 6

# Conclusion & Future Scope

The project followed a different approach to gesture recognition, using a very large,annotated data set of dynamic hand gesture videos and neural networks trained with the existing ISL data. This allowed us to build a gesture recognition system that is robust and works in real time using only an RGB camera. To train our system,we used a large data set of short, densely labeled video clips that was crowd-acted by our community of crowd workers. The project aimed on tackling a linguistic challenge faced by a significant number of population in India with respect to communication. The prototype is specifically designed for the speech-impaired and successfully demonstrates a solution to bridge the communication gap. The prototype can recognize 320+ words and convert them to hand gestures with 90 percent accuracy. It is also capable of breaking up sentences and displaying appropriate hand gestures for a bunch of keywords in the sentence. The system is capable of recording and converting the spoken statements into gestures. The future scope includes developing a mobile application for the same.

# References

1] Ilan Steinberg, Tomer M. London, Dotan Di Castro," Hand Gesture Recognition in Images and Video", Technion-Israel Institute of Technology, 2003.

[2] J. Jones. (1991, May 10). Networks (2nd ed.) [Online]. Available:http://www.atm.com R. Kurdyumov, P. Ho, J. Ng. (2011, December16) Sign Language Classification Using Webcam Images.

[3] Taner Arsan and Oğuz Ülgen ,"Sign Language Converter",International Journal of Computer Science Engineering Survey (IJCSES) Vol.6, No.4, August 2015.

[4] Adithya, V., Vinod, P. R., Gopalakrishnan, U. (2013). "Artificial neural network-based method for Indian sign language recognition". 2013 IEEE CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGIES. doi:10.1109/cict.2013.6558259.

[5] Warrier, K. S., Sahu, J. K., Halder, H., Koradiya, R., Raj, V. K. (2016). Software based sign language converter. 2016 International Conference on Communication and Signal Processing (ICCSP). doi:10.1109/ iccsp .2016.7754472.

[6] Abdulla, D., Abdulla, S., Manaf, R., Jarndal, A. H. (2016). Design and implementation of a sign-to-speech/text system for deaf and dumb people. 2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA).

[7] Hasan, M., Sajib, T. H., Dey, M. (2016). A machine learning based approach for the detection and recognition of Bangla sign language. 2016 International Conference on Medical Engineering, Health Informatics and Technology (MediTec).

[8] Rajaganapathy, S., Aravind, B., Keerthana, B., Sivagami, M. (2015). Conversation of Sign Language to Speech with Human Gestures. Procedia Computer Science, 50, 10–15.

[9] M.Suresh Anand, A.Kumaresan, An Integrated Two Way ISL (Indian Sign Language) Translation System – A New Approach, International Journal of Advanced Research in Computer Science, 4 (2), Jan. –Feb, 2013,7-12.

# Acknowledgement

Success of a project like this involving high technical expertise, patience and massive support of guides, is possible when team members work together. We take this opportunity to express our gratitude to those who have been instrumental in the successful completion of this project. We would like to show our appreciation to **Mrs. Rakhi Kalantri** for their tremendous support and help, without them this project would have reached nowhere. We would also like to thank our project coordinator **Mrs. Rakhi Kalantri** for providing us with regular inputs about documentation and project timeline. A big thanks to our HOD **Dr. Lata Ragha** for all the encouragement given to our team. We would also like to thank our principal, **Dr. S. M. Khot**, and our college, **Fr. C. Rodrigues Institute of Technology, Vashi**, for giving us the opportunity and the environment to learn and grow.

**Project Group Members:**

1. Raveena Dandona, 101611

_____

2. Ankita Shinde, 101650

_____

# Appendix A : Timeline Chart

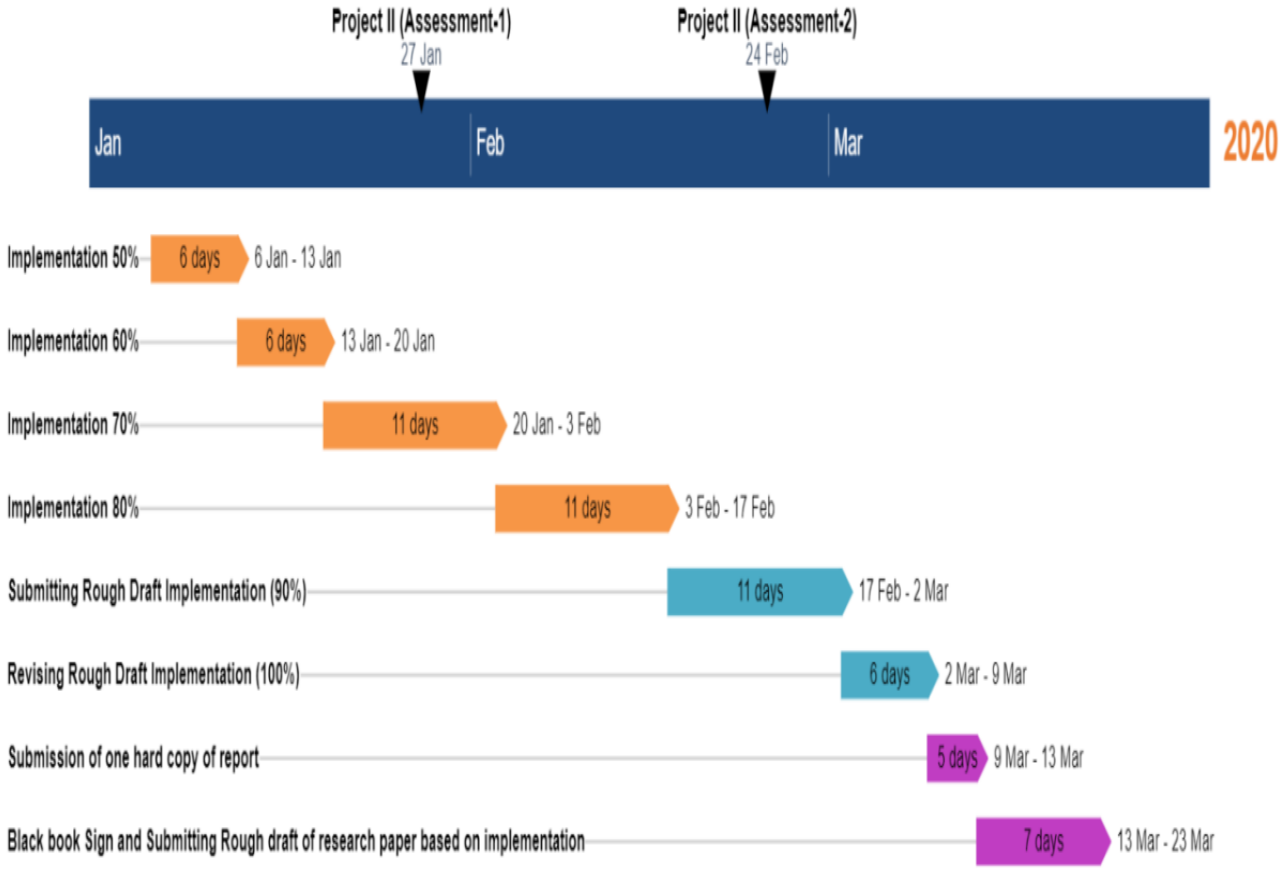


Figure 6.1: Timeline Chart

# Appendix B : Publication Details


- Authors : Ankita Shinde , Raveena Dandona
- Paper ID : IJERTV9IS020311
- Volume  Issue : Volume 09, Issue 02 (February 2020)
- Published (First Online): 04-03-2020
- ISSN (Online) : 2278-0181
- Publisher Name : IJERT
- URL: https://www.ijert.org/two-way-sign-language-converter-for-speech-impaired