A Project Report on

Intelligent and Interactive Sign Language Converter for specially abled people

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

Information Technology

by

Rikesh Kamra(15104006)
Ankit Gupta(15104004)
Denis Vaghasia(15104020)
Rakesh Sharma(15104017)

Under the Guidance of

Prof. Kiran Deshpande
Prof. Vishal Badgujar

Department of Information Technology
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W), Mumbai-400615
UNIVERSITY OF MUMBAI

Academic Year 2018-2019

Approval Sheet

This Project Report entitled ***"Intelligent and Interactive Sign Language Converter for specially abled people"*** Submitted by ***"Rikesh Kamra"(15104006), "Ankit Gupta"(15104004), "Denis Vaghasia"(15104020), "Rakesh Sharma"(15104017)*** is approved for the partial fulfillment of the requirement for the award of the degree of ***Bachelor of Engineering*** in ***Information Technology*** from ***University of Mumbai***.

Prof. Vishal Badgujar
Co-Guide

Prof. Kiran Deshpande
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Place:A.P.Shah Institute of Technology, Thane
Date:

CERTIFICATE

This is to certify that the project entitled ***"Intelligent and Interactive Sign Language Converter for specially abled people"*** submitted by ***"Rikesh Kamra" (15104006), "Ankit Gupta" (15104004), "Denis Vaghasia" (15104020), "Rakesh Sharma" (15104017)*** for the partial fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Information Technology***, to the University of Mumbai,is a bonafide work carried out during academic year 2018-2019.

Prof. Vishal Badgujar
Co-Guide

Prof. Kiran Deshpande
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Dr. Uttam D.Kolekar
Principal

External Examiner(s)

1.

2.

Place:A.P.Shah Institute of Technology, Thane
Date:

Declaration

We declare that this written submission represents 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 declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(Rikesh Kamra 15104006)
(Ankit Gupta 15104004)
(Denis Vaghasia 15104020)
(Rakesh Sharma 15104017)

Date:

Acknowledgement

We have great pleasure in presenting the report on **Interactive and Intelligent Sign Language Converterfor specially abled people.** We take this opportunity to express our sincere thanks towards our guide **Prof. Kiran Deshpande** & Co-Guide **Prof. Vishal Badgujar** Department of IT, APSIT thane for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Kiran B. Deshpande** Head of Department,IT, APSIT for his encouragement during progress meeting and providing guidelines to write this report.

We thank **Prof. Vishal S. Badgujar** BE project co-ordinator, Department of IT, APSIT for being encouraging throughout the course and for guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Student Name1:Rikesh Kamra
Student ID1:15104006

Student Name2:Ankit Gupta
Student ID2:15104004

Student Name3:Denis Vaghasia
Student ID3:15104020

Student Name4:Rakesh Sharma
Student ID4:15104017

Abstract

Communication is important as well as a very integral part of any working, active and a prosperous action. Without proper communication, it is difficult to know everyone's liking, hate, comfort, discomfort, etc and hence makes it difficult to work and process accordingly and hence in a way two hampers the development of anyone or anything. Some people have the hearing and speaking disabilities and hence because of this, they are unable to express and communicate in the normal way and express their thoughts, feelings or reviews. These people use Indian Sign language to communicate with everyone but most of the people around are unable to understand the Sign language. With the help of this project, we aim to build an application that will be working as a medium or as a translator and translate all the Sign Language to text/audio and vice versa. This application will be using the camera of the mobile device and with which it will record the gestures performed by a person and with help of Image processing these gestures will be converted accordingly and then with the help of Deep Learning these can be then translated properly and the meaning of those gestures will be available in the form of text. This application will have a Rich User Interface and easy navigation with a plethora of functions. Users will be given full access to learning sign language with the help of tutorials provided as well.

Contents

List of Figures

List of Abbreviations

SL:	Sign Language
CNN:	Convolutional Neural Network
ISL:	Indian Sign Language

Chapter 1

Introduction

Communication is the basic form for exchanging thoughts, views, perception and the ideas we think. The person suffering from hearing and speaking disabilities knows sign language and whenever they want to communicate with us they do it via gestures i.e. sign language and most of the times we are unaware as to what it means. Hence to bridge the gap, we are creating a sign language translator which would not only translate gesture performed by these person and it will convert it into text.

1.1 Problem Definition

Today we all want to live in a world where at least we can try and convey and explain all our requirements to someone else and get understood. A lot of people are unable to keep their points and even express their views on anything because whatever they convey are not understood by many people around. This is because these people communicate with the help of sign language or gestures. Due to this a lot of people feel left out of society and feel dejected. We will be using Deep Learning to understand the signs/gesture performed by these users and convert it into strings or etc for others to understand, thus making communication easy and equal for everyone

1.2 Motivation

- Our main objective is to create an android app which will translate the sign language i.e gesture into the English language
- Usually people with these disabilities could not communicate with other people hence to make them feel comfortable.
- Lack of understanding by people in Emergency situations.

Chapter 2

Literature Review

[1] Rini Akmeliawati, Melanie Po-Leen Ooi, Ye Chow Kuang, Real-Time Malaysian Sign Language Translation using Colour Segmentation and Neural Network, IMTC 2007 - Instrumentation and Measurement Technology Conference Warsaw, Poland, 1-3 May 2007.
Automatic sign-language translator provides a real-time English translation of the Malaysian SL.
Using custom made colored gloves makes it easy to recognize the hand positions and hand gestures and also makes it easy to use color segmentation technique.
Custom made gloves hinders the natural way of signing. Making gloves for everyone is costly and not feasible.

[2] Philippe Dreuw, Daniel Stein, Thomas Deselaers, David Rybach, Morteza Zahedi, Jan Bungeroth and Hermann Ney, Spoken language processing techniques for sign language recognition and translation, Human Language Technology and Pattern Recognition, Computer Science Department 6, RWTH Aachen University, Germany, 2008.
A system that recognizes complete sentences in sign language.
Vision based approach which does not require special data acquisition devices
Developing Sign Recognition methods for mobile applications.

[3] Purva C. Badhe, Vaishali Kulkarni, Indian Sign Language Translator Using Gesture Recognition Algorithm, IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), 2015.
To develop the application which help the deaf and mute people to communicate efficiently with other people .
Database contain more than the 1,30,000 videos most of the sign gets detected easily.
Methods used : VIsion Based Method, YCbCr skin color Approach, Scale Invariant Feature Transform(SIFT).
Since it uses only YCbCr skin color approach it is difficult to understand the sign performed by the people in the low light. Other limitation is that it use SIFT approach to detect and matching of the object which is slower than SURF.

[4] Mahesh M, Arvind Jayaprakash, Geetha M, SIGN LANGUAGE TRANSLATOR FOR MOBILE PLATFORMS, International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017.
Developing Sign Recognition methods for mobile applications.
Add gestures in database Recognize the gesture and display the result.
Uses all the skin color approach i.e. (RGB, Ycbcr ,HSI) and uses ORB technique.
Since the OpenCV version 2.4.1 is used for computer vision and machine language, which is not much comfortable with android.

Paper Title	Aim	Merits	Limitations
[1] Real-Time Malaysian Sign Language Translation using Colour Segmentation and Neural Network	Automatic sign-language translator provides a real-time English translation of the Malaysian SL.	Using custom made colored gloves makes it easy to recognize the hand positions and hand gestures and also makes it easy to use color segmentation technique	Custom made gloves hinders the natural way of signing. Making gloves for everyone is costly and not feasible.
[2] Spoken language processing techniques for sign language recognition and translation	A system that recognizes complete sentences in sign language.	Vision based approach which does not require special data acquisition devices	Developing Sign Recognition methods for mobile applications.
[3] Indian Sign Language Translator Using Gesture Recognition Algorithm .	To develop the application which help the deaf and mute people to communicate efficiently with other people .	Database contain more than the 1,30,000 videos most of the sign gets detected easily **Methods used :** VIsion Based Method, YCbCr skin color Approach, Fourier Descriptor.	Since it uses only YCbCr skin color approach it is difficult to understand the sign performed by the people in the low light. Other limitation is that it use Fourier Descriptor approach to detect and matching of the object which is slower than SURF .
[4] Sign Language Translator for mobile Platform.	Developing Sign Recognition methods for mobile applications.	Add gestures in database Recognize the gesture and display the result. Uses all the skin color approach i.e. (RGB, Ycbcr ,HSI) It uses ORB technique.	Since the OpenCV version 2.4.1 is used for computer vision And machine language, which is not much comfortable with android.

Chapter 3

Existing System Architecture

3.1 Sign Language Translator For Mobile Platforms

This paper describes using a camera for real-time hand gesture recognition. An android application is developed for the same which is able to recognize only the static gesture(i.e the gesture which doesn't involve motion like moving hands, expression, etc)
Different phases used for recognition of gestures are

- Preprocessing
- Skin Detection using RGB
- Skin Detection using YCbCr
- Skin Detection using HSI
- Thresholding
- Resizing
- Recognizing existing gestures

Limitation

- Only the static gesture is recognized.
- Since the OpenCV for Android is used for processing the gesture, the accuracy of the Android app decreases

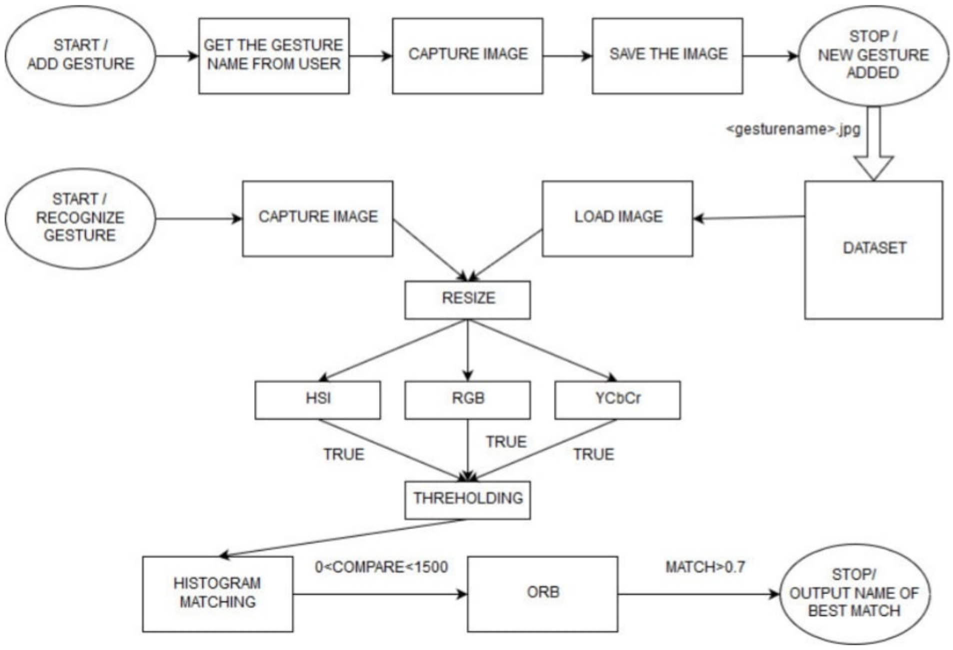


Figure 3.1: Process flow of Current System

- In the first layer gestures are captured and stored by the device camera on local system which is then added to datasets.
- Datasets are then used to train the model which later help in recognition.
- During recognition image is captured again and resized to 3 color models i.e. HSI, RGB and YCbCr.
- After resizing the image thresholding is done to match it with the required Histogram level.
- After histogram matching results is passed to ORB function to get the best match for results.

3.2 Indian Sign Language Translator Using Gesture Recognition Algorithm

This paper also describe using a camera for real time hand gesture recognition . Different phases used for recognition of gestures are

- Data Acquisition
- Pre-Processing and Hand Tracking
- Template Matching
- Gesture Recognition

Limitation

- Only the static gesture is recognized.
- The Desktop application is developed which make it difficult to use, as one cannot always have the laptop or desktop with him or her.

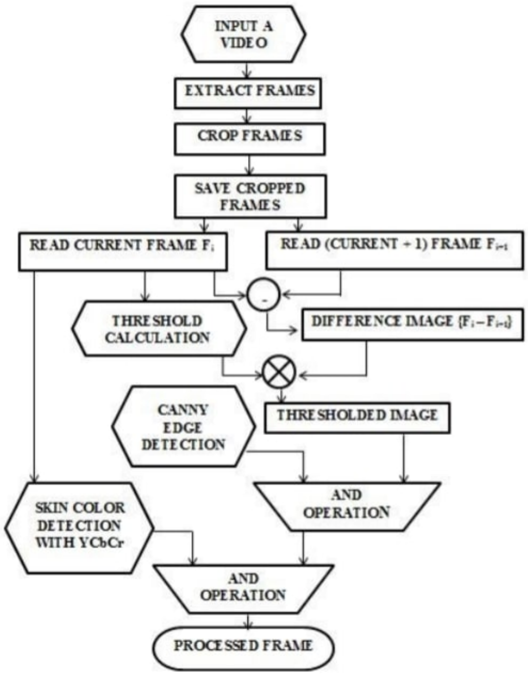


Figure 3.2: Flowchart for obtaining Processed frames

- Input video is taken and then frames are extracted, copied and then saved.
- Two Frames are compared and the difference between them is calculated.
- Thresholding for current frame and the difference between the two frames are calculated and one thresholded image is generated.
- Skin colour detection with YCbCr model is performed on current frame and canny edge detection is performed on thresholded image.
- Then the final processed frame is generated.

Chapter 4

Proposed System

4.1 Architecture

We propose the android application which bridges the communication gap between the deaf and mute community with society. This android app takes the gesture as input from a phone camera, the input gesture is processed and the corresponding meaning of that gesture is shown on the phone.
The flow diagram of the same is shown below

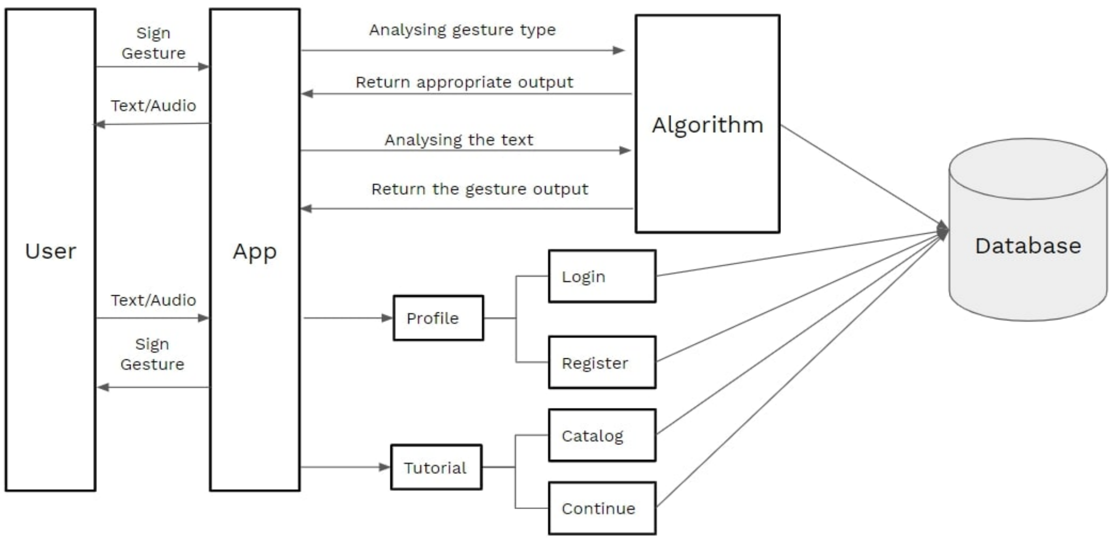


Figure 4.1: Process system Flow Diagram

4.2 Deep Learning

4.2.1 What is Deep Learning

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driver-less cars, enabling them to recognize a stop sign or to distinguish a pedestrian from a lamppost.

It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning technology is grabbing attention lately and because of its human-centered result. Its achieving results that were difficult before due to advancements in training methods. In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. For training, large data sets are required which are properly labelled; training is possible with a set of neural network providing proper layers of neurons and applying appropriate algorithms.

4.2.2 Examples of Deep Learning

- Facial recognition
  Deep learning is being used for facial recognition not only for security purposes but for tagging people on Facebook posts and we might be able to pay for items in a store just by using our faces in the near future. The challenges for deep-learning algorithms for facial recognition is knowing its the same person even when they have changed hairstyles, grown or shaved off a beard or if the image taken is poor due to bad lighting or an obstruction.

- Medicine and pharmaceuticals
  From disease and tumor diagnoses to personalized medicines created specifically for an individuals genome, deep learning in the medical field has the attention of many of the largest pharmaceutical and medical companies.

- Personalized shopping and entertainment
  Ever wonder how Netflix comes up with suggestions for what you should watch next? Or where Amazon comes up with ideas for what you should buy next and those suggestions are exactly what you need but just never knew it before? Yep, its deep-learning algorithms at work. The more experience deep-learning algorithms get, the better they become. It should be an extraordinary few years as technology continues to mature.

4.2.3 What's the Difference Between Machine Learning and Deep Learning?

Deep learning is a specialized form of machine learning. A machine learning workflow starts with relevant features being manually extracted from images. The features are then used to create a model that categorizes the objects in the image. With a deep learning workflow, relevant features are automatically extracted from images. In addition, deep learning performs end-to-end learning where a network is given raw data and a task to perform, such as classification, and it learns how to do this automatically.

Another key difference is deep learning algorithms scale with data, whereas shallow learning converges. Shallow learning refers to machine learning methods that plateau at a certain level of performance when you add more examples and training data to the network.

A key advantage of deep learning networks is that they often continue to improve as the size of your data increases. In machine learning, you manually choose features and a classifier to sort images. With deep learning, feature extraction and modeling steps are automatic.

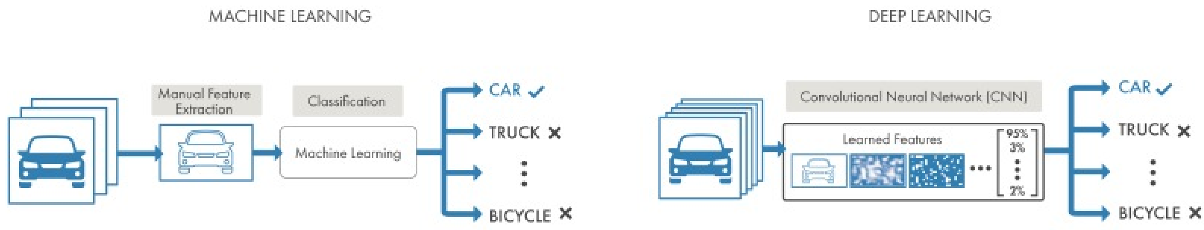


Figure 4.2: Comparsion of ML and DL

4.3 Keras

4.3.1 What is keras?

Keras is a high-level neural networks API, which acts as frontend to many other deep learning libraries. It's written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It gives better and fast results which makes training and prediction of models faster.

4.3.2 Why use Keras?

- Keras is an API designed for developers, not machines. It follows best practices resulting in reduced cognitive load; in common use cases it minimizes the number of user actions required. This makes Keras easy to learn and use. Keras helps to ease the work and make developers more productive.
- Keras models can be easily deployed across a wide range of independent platforms compared to any other deep learning framework.
  - On iOS, via Apples CoreML (Keras support officially provided by Apple) .
  - On Android, via the TensorFlow Lite .
  - In the browser, via GPU-accelerated JavaScript runtimes such as Keras.js and WebDNN .
  - On Google Cloud, via TensorFlow-Serving .

4.4 Open CV

4.4.1 What is OpenCV?

- OpenCV is a free-to-use open source computer vision and machine learning software library. The platform is designed to offer a general infrastructure for computer vision applications and to fast-track the use of machine perception in commercially available creations.

- Additionally, OpenCV is operating under a BSD license, which means there is minimal to no restrictions on the use of the library. This allows commercial entities are able to employ the code and tweak it as they want.

- With OpenCV, users have access to more than 2,500 algorithms that can be utilized to deploy a number of computer vision and machine learning capabilities such as facial recognition object identification, and more.

4.4.2 OpenCV Benefits

- Cross-Platform Library
  OpenCV is a cross-platform library, enabling users on different operating systems to use it without complications and even making it accessible on mobile systems like iOS and Android, which makes it a portable library.

- Vast Algorithms
  OpenCV offers users to access over 2,500 algorithms. This vast library allows programmers to perform a multitude of tasks in their software such as extracting 3D models of objects, removing red eyes, following eye movements, and more.

- Efficient Solution
  OpenCV has been optimized to provide efficiency especially for the processing of real-time programs. On top of that, its design allows it to take advantage of multi-core systems and of hardware acceleration to deploy.

4.5 Convolutional Neural Network

4.5.1 What is Convolutional Neural Network

- A convolutional neural network also known as a CNN or comp net is an artificial neural network that is so far been most popularly used for analyzing and processing images that is specifically designed to process pixel data.

- Although image analysis has been the most widespread use of CNN's they can also be used for other data analysis or classification problems as well.

- CNN has some specialization for being able to pick out or detect patterns and make sense of them this pattern detection is what makes CNN so useful for image analysis.

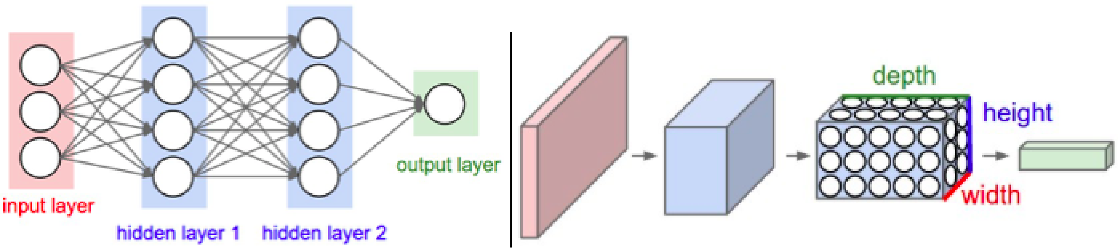
- A CNN is much like a multilayer perceptron (MLP) that has been designed to reduce processing required in image processing.

4.5.2 Architecture

- The layers of a CNN consist of hidden layers, an input layer and an output layer that includes multiple convolutional layers, fully connected layers, pooling layers and normalization layers. Finally, there is a last fully-connected layer the output layer that represent the predictions.

- A CNN layer receives input then transforms the input in required format and then outputs the transformed input to the next layer with a convolutional layer this transformation is a convolution operation.

Figure 4.3: Architecture of CNN



Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

4.5.3 2D CNN

- In 2D CNN the layers have 2 dimensions: width and height. Further, filter is used which is passed in the neurons. The filter is of MxM matrix is used which is passed to the processed image, where M is greater than or equal 3.

- 2D CNN uses this filter which is passed on each matrix of a single processed image and generates the output.

- Pooling layer is used after every convolutional layer to extract the main features and reduce the matrix size.

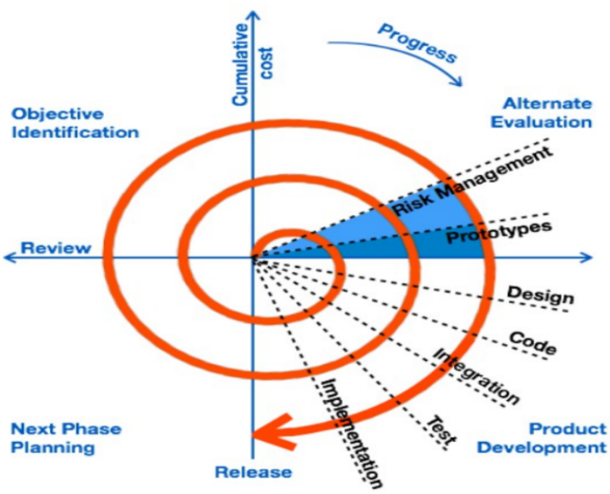
Chapter 5

Results and Implementation

5.1 Software Engineering Model used

5.1.1 Spiral Model

- Spiral model was selected as the software development model for the project since it describes an incremental process wherein the project is built in from the smallest of the features step by step in iterations.
- This model provides the potential for rapid development of increasingly more complete versions of the software.
- During later iterations, increasingly more complete versions of the engineering systems are produced.



Spiral Model

5.2 Code:

Figure 5.1: Code for training the deep learning model

```
import numpy as np
import pickle
import cv2, os
from glob import glob
from keras import optimizers
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.utils import np_utils
from keras.callbacks import ModelCheckpoint
from keras import backend as K
K.set_image_dim_ordering('tf')

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'

def get_image_size():
    img = cv2.imread('gestures/1/100.jpg', 0)
    return img.shape

def get_num_of_classes():
    return len(glob('gestures/*'))

image_x, image_y = get_image_size()
```

```python
def cnn_model():
    num_of_classes = get_num_of_classes()
    model = Sequential()
    model.add(Conv2D(16, (2,2), input_shape=(image_x, image_y, 1), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2), padding='same'))
    model.add(Conv2D(32, (3,3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(3, 3), strides=(3, 3), padding='same'))
    model.add(Conv2D(64, (5,5), activation='relu'))
    model.add(MaxPooling2D(pool_size=(5, 5), strides=(5, 5), padding='same'))
    model.add(Flatten())
    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(num_of_classes, activation='softmax'))
    sgd = optimizers.SGD(lr=1e-2)
    model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
    filepath="cnn_model_keras2.h5"
    checkpoint1 = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode
    callbacks_list = [checkpoint1]
    from keras.utils import plot_model
    plot_model(model, to_file='model.png', show_shapes=True)
    return model, callbacks_list

def train():
    with open("train_images", "rb") as f:
        train_images = np.array(pickle.load(f))
    with open("train_labels", "rb") as f:
        train_labels = np.array(pickle.load(f), dtype=np.int32)

    with open("val_images", "rb") as f:
        val_images = np.array(pickle.load(f))
    with open("val_labels", "rb") as f:
        val_labels = np.array(pickle.load(f), dtype=np.int32)

    train_images = np.reshape(train_images, (train_images.shape[0], image_x, image_y, 1))
    val_images = np.reshape(val_images, (val_images.shape[0], image_x, image_y, 1))
    train_labels = np_utils.to_categorical(train_labels)
    val_labels = np_utils.to_categorical(val_labels)

    print(val_labels.shape)

    model, callbacks_list = cnn_model()
    model.summary()
    model.fit(train_images, train_labels, validation_data=(val_images, val_labels), epochs=20, bat
        500, callbacks=callbacks_list)
    scores = model.evaluate(val_images, val_labels, verbose=0)
    print("CNN Error: %.2f%%" % (100-scores[1]*100))
    #model.save('cnn_model_keras2.h5')

train()
K.clear_session();
```

Figure 5.2: Flask code to run python script on html

```
from importlib import import_module
import os
import pyrebase
from flask import Flask, render_template, Response,redirect, url_for,request

from find import find_now


app = Flask(__name__)

config = {
    "apiKey": "AIzaSyCVb8WGYsjUwYPAS1ZY808s7JXGPETn9gM",
    "authDomain": "webtesting-rkamra.firebaseapp.com",
    "databaseURL": "https://webtesting-rkamra.firebaseio.com",
    "projectId": "webtesting-rkamra",
    "storageBucket": "webtesting-rkamra.appspot.com",
    "messagingSenderId": "751221935891"
}

firebase = pyrebase.initialize_app(config);

auth = firebase.auth()

@app.route('/')
def index():
    return render_template('index.html')
# def login1():
#    return render_template('login.html')

@app.route('/login',methods = ['POST'])
def login():
    if request.method == 'POST':
        return redirect(url_for('login1'))

@app.route('/index1',methods = ['POST'])
def index1():
    if request.method == 'POST':
        fname = request.form['first_name']
        lname = request.form['last_name']
        bday = request.form['birthday']
        email = request.form['email']
        passwd = request.form['pass2']
        phone = request.form['phone']
        gender = request.form['gender']
        authe = auth.create_user_with_email_and_password(email,passwd)
        db =firebase.database()
        db.child("users").push({"First_name":fname,"Last_name":lname,"birthday":bday,"email":email,
            password":passwd,"phone":phone,"gender":gender})
        return redirect(url_for('index'))

@app.route('/translate',methods = ['POST'])
def translate():
    if request.method == 'POST':
        img_res = find_now()
        return render_template('dashboard.html', value_res = img_res)
```

```python
@app.route('/dashboard1',methods = ['GET','POST'])
def dashboard1():
    if request.method == 'POST':
        email = request.form['email']
        passwd = request.form['passwd']
        authe = auth.sign_in_with_email_and_password(email,passwd)
        # print(authe)
        # if auth.get_account_info(authe['idToken']).exists():
            # return "<h3>Incorrect Credentials</h3>"
        # else:
        # if authe == Null:
        return redirect(url_for('dashboard'))
        # else:
            # return "<h3>Incorrect Credentials</h3>"

@app.route('/dashboard')
def dashboard():
    return render_template('dashboard.html')

# @app.route('/translate1')
# def translate1():

@app.route('/signup',methods = ['POST'])
def signup():
    if request.method == 'POST':
        return redirect(url_for('signup1'))

@app.route('/login1')
def login1():
    return render_template('login.html')
```

```python
@app.route('/signup1')
def signup1():
    return render_template('signup.html')

if __name__ == '__main__':
    app.debug = True
    app.run(host='0.0.0.0', threaded=True)
```

Figure 5.3: Python Code to recognize gestures

```python
import cv2
import numpy as np
import tensorflow as tf
from cnn_tf import cnn_model_fn
import os
import sqlite3
from keras.models import load_model
from keras.models import model_from_json
class find_now():

    def __init__(self):
        os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
        tf.logging.set_verbosity(tf.logging.ERROR)
        classifier = tf.estimator.Estimator(model_dir="tmp/cnn_model2", model_fn=cnn_model_fn)
        prediction = None
        model = load_model('cnn_model_keras2.h5')
        image_x, image_y = cv2.imread('gestures/100.jpg', 0).shape
        self.keras_predict(model,image_x, image_y, np.zeros((50, 50), dtype=np.uint8))
        self.recognize(model,image_x, image_y)


    def keras_process_image(self,image_x, image_y,img):
        img = cv2.resize(img, (image_x, image_y))
        img = np.array(img, dtype=np.float32)
        img = np.reshape(img, (1, image_x, image_y, 1))
        return img
```

```python
def keras_predict(self,model,image_x, image_y, image):
    processed = self.keras_process_image(image_x, image_y,image)
    pred_probab = model.predict(processed)[0]
    pred_class = list(pred_probab).index(max(pred_probab))
    return max(pred_probab), pred_class

def get_pred_text_from_db(self,pred_class):
    conn = sqlite3.connect("gesture_db.db")
    cmd = "SELECT g_name FROM gesture WHERE g_id="+str(pred_class)
    cursor = conn.execute(cmd)
    for row in cursor:
        return row[0]

def split_sentence(self,text, num_of_words):
    '''
    Splits a text into group of num_of_words
    '''
    list_words = text.split(" ")
    length = len(list_words)
    splitted_sentence = []
    b_index = 0
    e_index = num_of_words
    while length > 0:
        part = ""
        for word in list_words[b_index:e_index]:
            part = part + " " + word
        splitted_sentence.append(part)
        b_index += num_of_words
        e_index += num_of_words
        length -= num_of_words
    return splitted_sentence
```

```python
def put_splitted_text_in_blackboard(self,blackboard, splitted_text):
    y = 200
    for text in splitted_text:
        cv2.putText(blackboard, text, (4, y), cv2.FONT_HERSHEY_TRIPLEX, 2, (255, 255, 255))
        y += 50


def recognize(self,model,image_x, image_y):
    global prediction
    cam = cv2.VideoCapture(1)
    if cam.read()[0] == False:
        cam = cv2.VideoCapture(0)
    # hist = get_hand_hist()
    x, y, w, h = 300, 100, 300, 300
    while True:
        text = ""
        img = cam.read()[1]
        img = cv2.flip(img, 1)
        img = cv2.resize(img, (640, 480))
        imgCrop = img[y:y+h, x:x+w]
        img_ycrcb = cv2.cvtColor(imgCrop, cv2.COLOR_BGR2YCR_CB)
        blur = cv2.GaussianBlur(img_ycrcb,(11,11),0)

        lower_blue = np.array([20, 133, 77])
        upper_blue = np.array([255, 173, 127])

        mask = cv2.inRange(blur, lower_blue, upper_blue)
        contours = cv2.findContours(mask.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)[0]
        if len(contours) > 0:
            contour = max(contours, key = cv2.contourArea)
            #print(cv2.contourArea(contour))
            if cv2.contourArea(contour) > 10000:
                x1, y1, w1, h1 = cv2.boundingRect(contour)
                save_img = mask[y1:y1+h1, x1:x1+w1]

                if w1 > h1:
                    save_img = cv2.copyMakeBorder(save_img, int((w1-h1)/2) , int((w1-h1)/2) , 0, 0
                        cv2.BORDER_CONSTANT, (0, 0, 0))
                elif h1 > w1:
                    save_img = cv2.copyMakeBorder(save_img, 0, 0, int((h1-w1)/2) , int((h1-w1)/2)
                        cv2.BORDER_CONSTANT, (0, 0, 0))

                pred_probab, pred_class = self.keras_predict(model,image_x, image_y, save_img)

                if pred_probab*100 > 80:
                    text = self.get_pred_text_from_db(pred_class)
                    print(text)
        blackboard = np.zeros((480, 640, 3), dtype=np.uint8)
        splitted_text = self.split_sentence(text, 2)
        self.put_splitted_text_in_blackboard(blackboard, splitted_text)
        #cv2.putText(blackboard, text, (30, 200), cv2.FONT_HERSHEY_TRIPLEX, 1.3, (255, 255, 255))
        cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
        res = np.hstack((img, blackboard))
        cv2.imshow("Recognizing gesture", res)
        cv2.imshow("thresh", mask)
        if cv2.waitKey(1) == ord('q'):
            break
```

5.3 Output:

Figure 5.4: Steps on how to execute the flask code

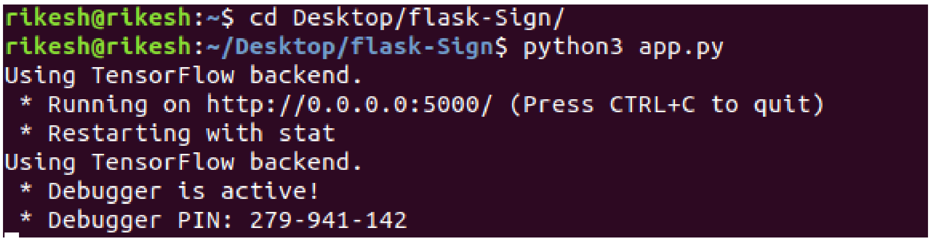
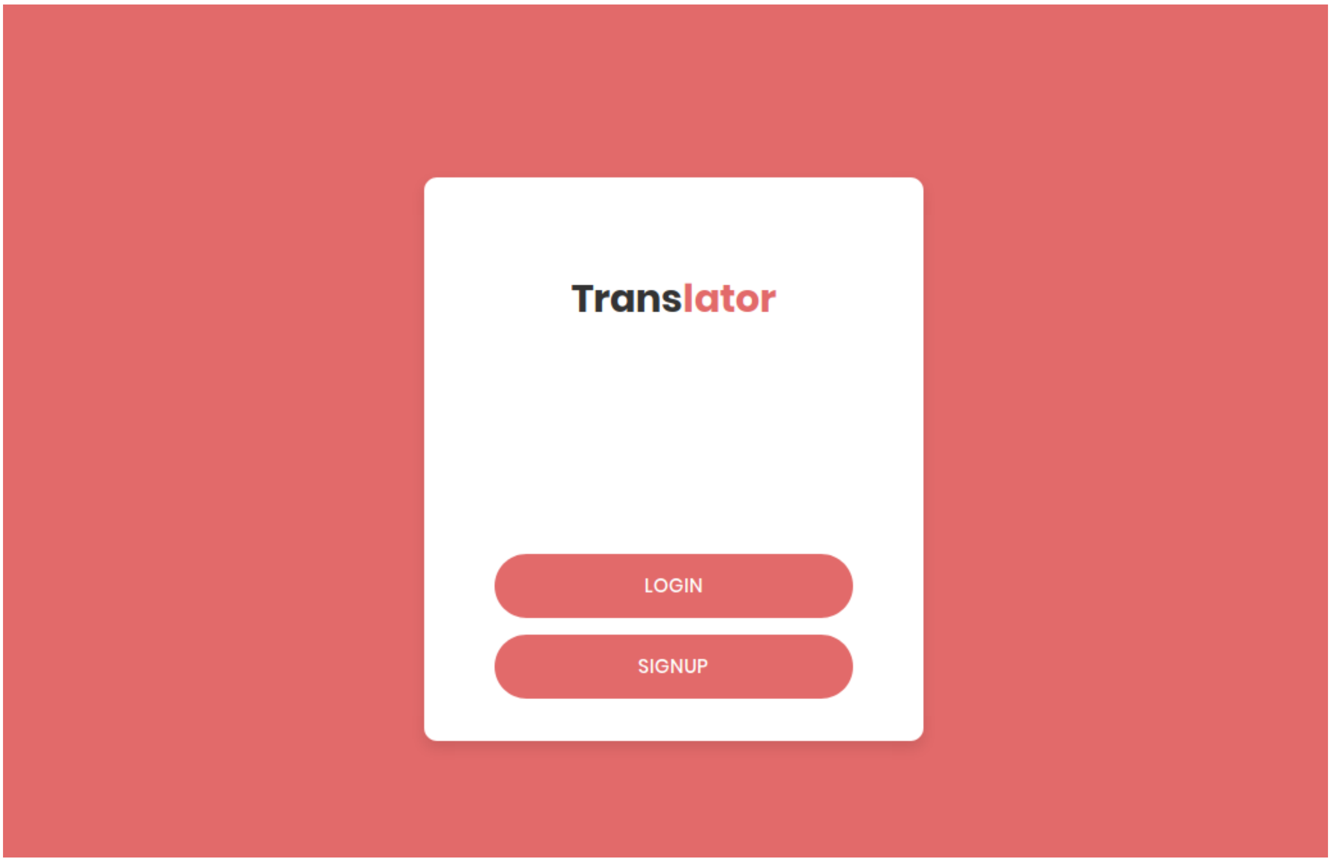


Figure 5.5: Login Screen:



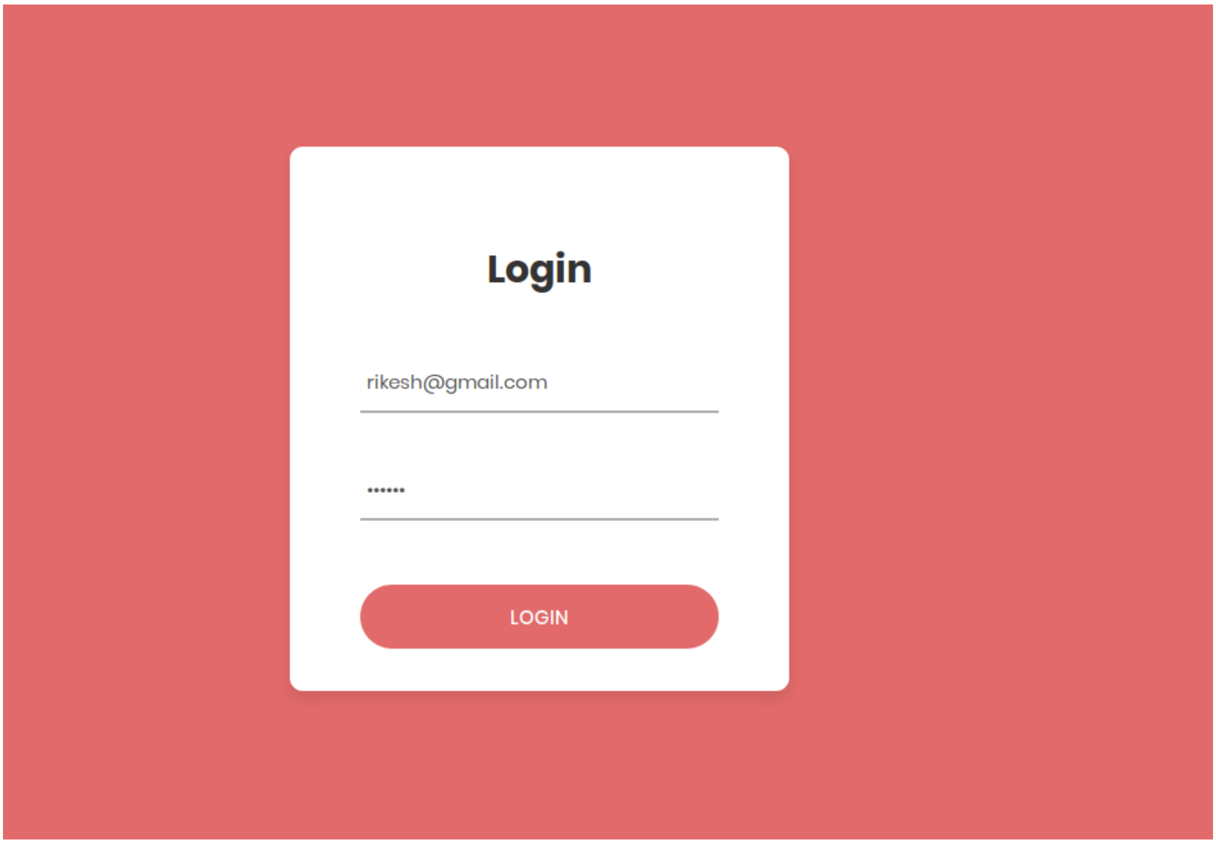
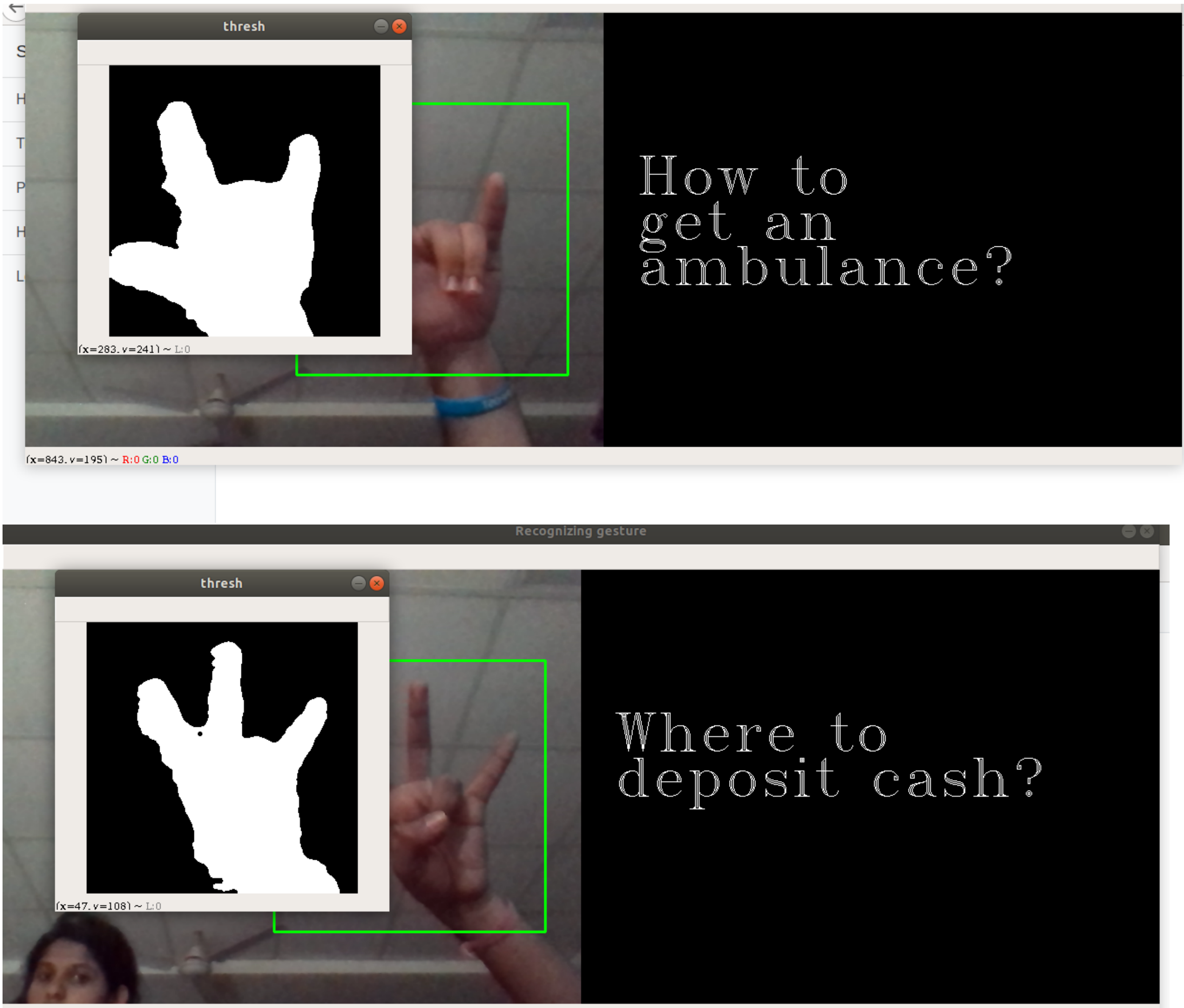
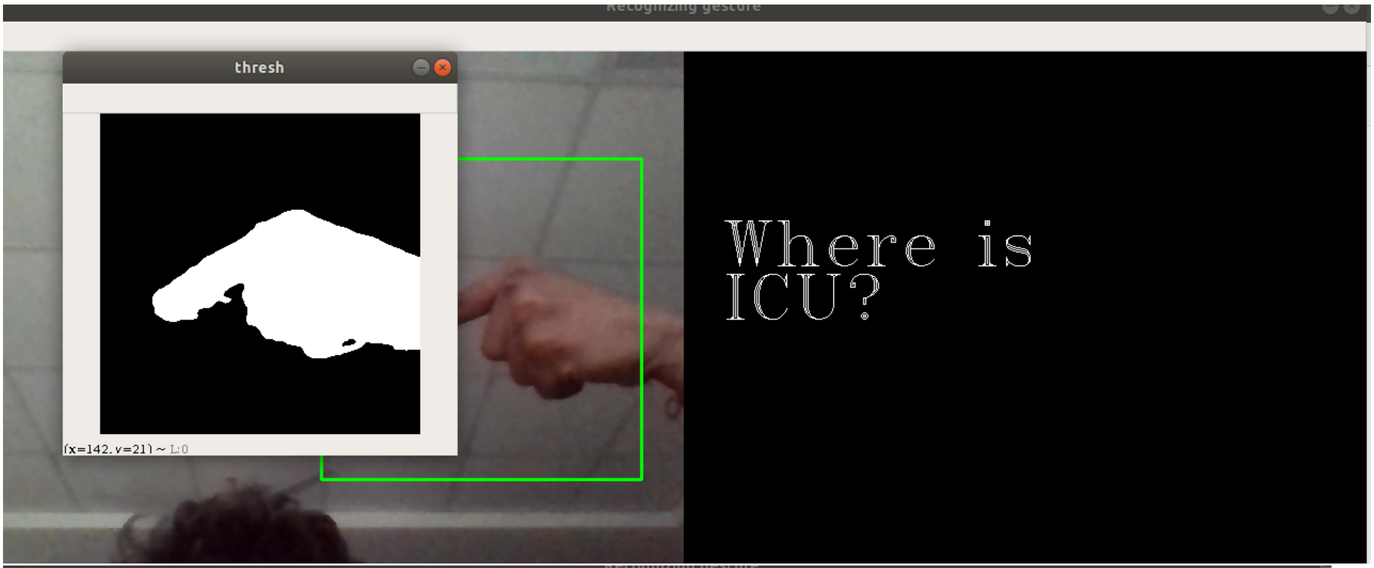
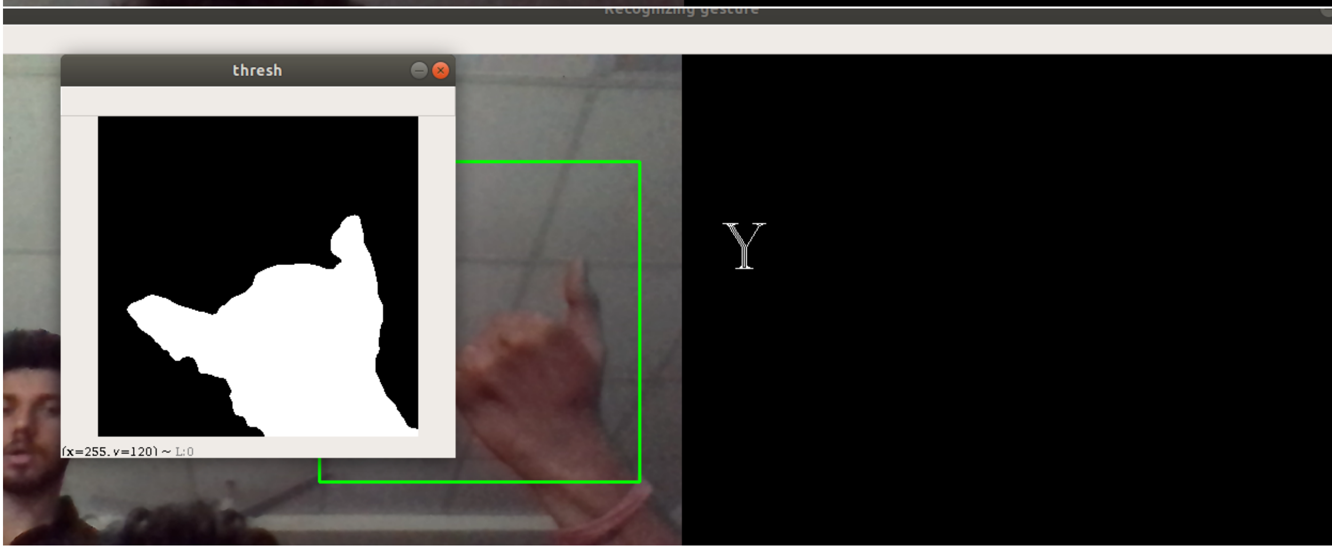


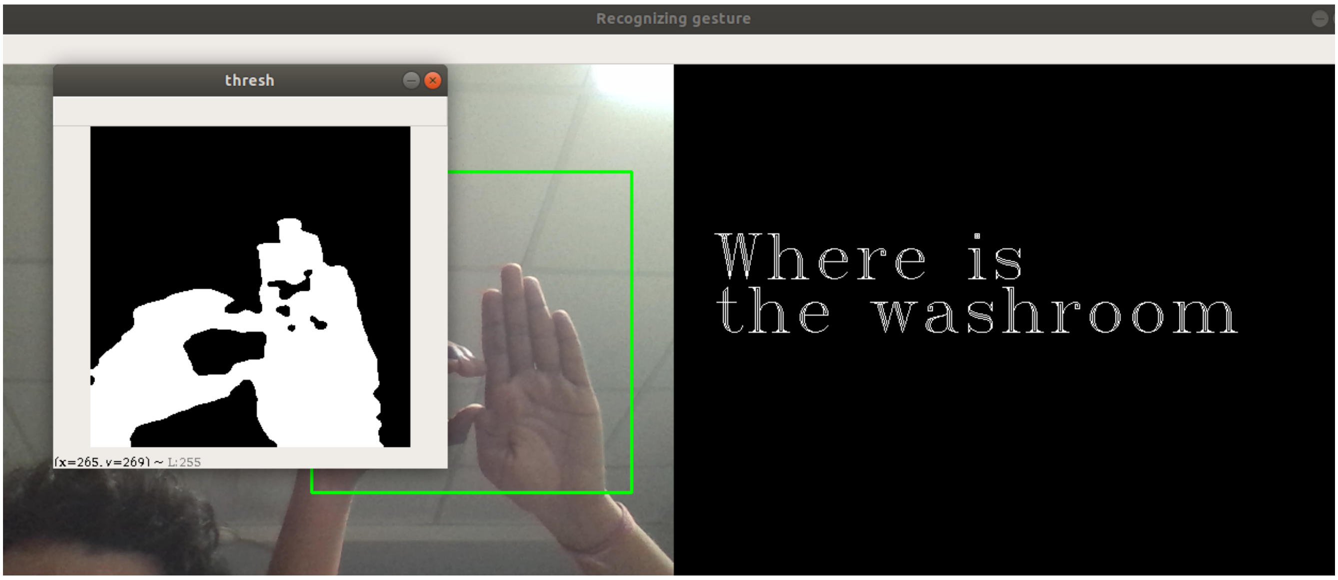
Figure 5.6: Dashboard:

Figure 5.7: Few of sample translations




thresh
(x=142, y=21) ~ L:0
Where is
ICU?


thresh
(x=255, y=120) ~ L:0
Y


Recognizing gesture
thresh
(x=265, y=269) ~ L:255
Where is
the washroom

Chapter 6

Conclusion and Future Scope

6.1 Conclusions

In this Black Book an app has been developed that translates the gestures made in Indian Sign language (ISL) into English. The app aims to help the society in terms of better communication aspects. ISL translator system helps in such a way that, it inputs the ISL sign gesture, decodes it, interprets it and outputs the meaning of the gesture in English.

6.2 Result Findings

With the application we are able to convert static gestures into sentences or letters. We can successfully have basic communication and it is made sure that the difficulties faced by the specially abled people are resolved at least in some of the domains or sectors. Basic communication gestures are successfully translated with 82 Percent accuracy.

6.3 Future Scope

Dynamic gestures that have dynamic movement of the hand requires video processing to determine the gesture. The goal is to develop an android application that recognizes both static and dynamic gestures at a higher accuracy rate. This app will make sure to work on the actual guidelines of ISL and we will have a mobile application which can successful capture video of gesture performed and get the gesture translated with help of 3dCNN.

Bibliography

[1] Rini Akmeliawati, Melanie Po-Leen Ooi, Ye Chow Kuang, Real-Time Malaysian Sign Language Translation using Colour Segmentation and Neural Network, IMTC 2007 - Instrumentation and Measurement Technology Conference Warsaw, Poland, 1-3 May 2007.

[2] Philippe Dreuw, Daniel Stein, Thomas Deselaers, David Rybach, Morteza Zahedi, Jan Bungeroth and Hermann Ney, Spoken language processing techniques for sign language recognition and translation, Human Language Technology and Pattern Recognition, Computer Science Department 6, RWTH Aachen University, Germany, 2008.

[3] Purva C. Badhe, Vaishali Kulkarni, Indian Sign Language Translator Using Gesture Recognition Algorithm, IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), 2015.

[4] Mahesh M, Arvind Jayaprakash, Geetha M, SIGN LANGUAGE TRANSLATOR FOR MOBILE PLATFORMS, International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017.

Appendices

6.4 Appendix-A: Python Download Installation

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

- Download and execute the latest Python installation package from Python.org. For Liquid Web servers, youll most likely want the 64-bit version of the installer, but you may want to discuss software requirements with your developer.
- Choose the recommended installation options for the simplest installation experience (You can also choose Customize Installation if you need to adjust locations or features, but this may require additional configuration. See Python.org for further instructions on custom installation options).
- Check the box for Add Python 3.7 to PATH. This will adjust your System Environment Variables automatically so that Python can be launched from any command prompt.
- Verify a successful installation by opening a command prompt window in your Python installation directory (probably C:yourusername*37 if youve installed the latest available version). You should receive a message similar to what is shown below. If you selected Add Python 3.7 to PATH, you can verify the installation from any command prompt window.

6.5 Appendix-B: Installing PIP

One of the best tools to install and manage Python packages is called Pip. Pip has earned its fame by the number of applications using this tool. Used for its capabilities in handling binary packages over the easy installed packaged manager, Pip enables 3rd party package installations.

- Download get-pip.py to a folder on your computer.
- Open a command prompt and navigate to the folder containing get-pip.py.
- Run the following command:
  python get-pip.py
- Pip is now installed.

- You can verify that Pip was installed correctly by opening a command prompt and entering the following command: pip -V
- You should see output similar to the following: pip 18.0 from c:37-packages (python 3.7)

6.6 Appendix-C: Downloading and Installing Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:
A flexible Gradle-based build system.
A fast and feature-rich emulator.
Instant Run to push changes to your running app without building a new APK.

- Download the Android Studio from the official website of google.
- After downloading double click the .exe file, the installer responded by presenting the Android Studio Setup dialog box .
- Click Next to complete the setup.