VOCAL AND GESTURE BASED COMMUNICATION FOR/BY DISABLED PERSONS.

PROJECT REPORT

Submitted by

SRINIVASAN.M [REGISTER NO: 211417104269]

MILIND KUMAR.M [REGISTER NO: 211417104326]

RIYAZ UDEEN.S [REGISTER NO: 211417104501]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2021

BONAFIDE CERTIFICATE

Certified that this mini project report **"VOCAL AND GESTURE BASED COMMUNICATION FOR/BY DISABLED PERSONS"** is the bonafide work of "**SRINIVASAN.M(211417104269)**, **MILIND KUMAR.M(211417104326) and RIYAZ UDEEN.S(211417104501)"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.S.MURUGAVALLI,M.E.,Ph.D.,	**Mr.A.KARTHEKEYAN**
HEAD OF THE DEPARTMENT	**Professor**
DEPARTMENT OF CSE,	DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,	PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,	NAZARATHPETTAI,
POONAMALLEE,	POONAMALLEE,
CHENNAI-600 123.	CHENNAI-600 123.

Certified that the above candidate(s) were examined in the Anna University Project Viva-Voce Examination held on...........................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Thiru.C.SAKTHIKUMAR,M.E.,** and **Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my Project Guide , **Mr.A. KARTHEKEYAN M.E.,** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

We would also like to thank our parents and our friends who helped us in the progress of the project.

NAME OF THE STUDENTS

SRINIVASAN.M

MILIND KUMAR.M

RIYAZ UDEEN.S

ABSTRACT

In our country 6% of people are visually impaired person and 2.78% of people are not able to speak. The Vision and voice are the major defects for these two persons. Sign language is the language of communication for deaf and dumb people. Most of these physically impaired communities are dependent on sign language translators to express their thoughts to rest of the world. This causes isolation of these people in society. Hence, Sign language recognition is one of the most growing fields of research today. A sign language is composed of various gestures formed by physical movement of body parts i.e. hand, arms or facial expressions. In this project, a method is proposed that makes the use of hand gestures for recognition of Indian sign language. Hand gesture recognition system provides us an innovative, natural, user friendly way of interaction with the computer which is more familiar to the human beings. The communication between the dumb and visually impaired person are made only by their hand gestures. This project presents various methods of hand gesture and sign language recognition for blind and dumb person.

KEYWORDS:

Deep Learning, Cloud computing, Convolutional Neural Network, Image processing, Natural Language Processing, Recurrent Neural Network, Long-short term memory.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

CHAPTER-1
INTRODUCTION

1.1 OVERVIEW

Humans are social beings that rely on one another in their surroundings and are interrelated with other individuals. Communication, both verbally and nonverbally, is the only tool that can connect with other people in their environment. Through communication, we speak to ourselves, know and assess ourselves. Through communication, we get to understand and communicate with others and communicate our emotions towards others and through interaction, we solve all types of issues, create fresh thoughts, and share experiences and expertise with others.

Body gestures are often used to clarify the intent of a conversation. However, for systems without the ability to communicate verbally, body gestures can also stand alone to express certain intentions. The most used parts of the body to communicate are the hands and palms. The combination of hand and palm movements can be used to provide information or order a purpose, for example, the gesture of waving goodbye, ordering to stop and call.

There is a huge communication gap between a normal and a differently-abled person, our project mainly aims to bridge the communication gap between them.

It is really difficult for us to understand what they are trying to say exactly and vice versa in case of deaf and dumb.

To avoid this complexity in understanding each other we develop a Vocal and Gesture based Companion for disabled persons. This product is used for two-way communication.

For Differently abled to a Normal person, We use Deep Learning to analyse sign language shown by the disabled person and the analyzed result is converted into text format and produced as vocal output.

In case of a Normal person to a Differently abled, The person speaks the content like (ONE, TWO, ALL THE BEST, etc..) and this will get converted into sign language using NLP and is fed as a sign language on the display screen. It is decrypted by users who have the appropriate qualities to determine the access structure.

1.2 PROBLEM DEFINITION

In our project, we use iot device to capture the image from differently-abled people. We then use deep learning to match the image with the stored images in the database to understand the sign language, the image gesture is then conveyed into the text and it is conveyed to the normal person as a vocal output. Similarly, when a normal person has to convey to the differently-abled person, he uses one words such as (one, two, hello) and this is converted into sign language using natural language processing and it is displayed in the screen for the differently-abled person to understand.

We use a single portable iot device, to capture the images using raspberry pie camera and with the use of mike and speakers for vocal input and output.

This two-way communication is done by a single iot device rather than the other conventional methods such as multiple devices for input and output.

LITERATURE SURVEY

CHAPTER-2

LITERATURE SURVEY

Paper 1:" HOG-based Hand Gesture Recognition Using Kinect"

In this paper, we build interactions between humans and computers using hand gestures. The hand gesture is recognized by the palm of the hand which is obtained from the results of human skeleton segmentation through camera Kinect. Recognition of palm gestures is performed on a series of RGB Kinect output frames. Dynamic Time Warping (DTW) is used as a classifier that will compare the description of the input gesture with the template gesture description. Based on the results of the experiment, the performance of the hand gesture recognition system reached 76.7%.

YEAR: 2019

JOURNAL: IEEE

METHODOLOGY: KINECT

AUTHOR: Kevin Nathanael Kris Andria, Bima Sena Bayu Dewantara, Dadet Pramadihato.

Paper 2: “IoT Device for Disabled People”

The proposed design makes use of hand gloves mounted with flex sensors which recognize the characters and commands. The gestures recognized will be displayed as audio and visual output through LCD and Bluetooth speaker. The Optical Character Recognition is used for the blind people in order to recognize the text-based images for audio and LCD display. This system consists of live tracking as one of the modules for tracking the physically challenged people. The purpose is to enhance and improve the system for detecting sign language. The device not only converts sign languages to speech but also have incorporated modules like Optical Character Recognition (OCR) and live tracking.

YEAR: 2019

JOURNAL: ICRTAC

METHODOLOGY: IOT

AUTHOR: Shankar, Mallika Chowdhary, Priyadarshini.

Paper 3: “Smart hand gloves for disable people for communication”

Smart Hand Gloves help disable people to live with normal people. As dumb person cannot speak then this smart glove helps him to convert his hand gesture into text and pre-recorded voice. This also help normal person to understand what he is trying to say and reply accordingly. This Smart Gloves has facility of Home Appliance control from which a physically impaired person become independent to live. The main objective of the implemented project is to develop a reliable, easy to use, light weight smart hand gloves system which can minimizes the obstacles for disable people where they can stand with the race.

YEAR: 2018

JOURNAL: IRJET

METHODOLOGY: IOT

AUTHOR: Dhaval L. Patel, Hershel S. Tapase, Praful A. Landge, Parmeshwar P. Prof. A. P. Bagade

Paper 4: “Virtual talk for deaf, mute, blind and normal humans,”

This paper describes a new method of developing wearable sensor gloves for detecting hand gestures which uses British and Indian sign language system. The outputs are produced in the text format using LCD and audio format using APR9600 module. The hand gesture or the hand signs are converted to electrical signals using flex sensor. These electrical signals are processed to produce appropriate audio and text output. Previously designed devices were not accurate in tracing the hand gestures. The paper employs method of tuning in order to improve the accuracy of detecting hand gesture.

YEAR: 2013

JOURNAL: IEEE

METHODOLOGY: Hand gesture recognition, British and Indian sign language, Flex sensor, APR9600 module.

AUTHOR: Vikram Sharma M, Vinay Kumar N, Shruti C Masaguppi, Suma MN, D R Ambika

SYSTEM ANALYSIS

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

There are very few Existing systems/devices for disabled persons (deaf and dumb) and one among them is the **Smart glove to Capture Gesture.**

Even Though it’s existing it’s not affordable for real time utilization because of the following drawbacks like both the users should have a device and its very Bulky to wear and difficult to handle and Consists of dedicated components which are quite expensive.

EXISTING SYSTEM DISADVANTAGES

Smart glove to Capture Gesture system consist of two gloves(devices) for both the users to wear which makes it difficult to be used at every circumstances.

The size of the device is huge and difficult to handle due to which it can only be used with proper handling and care and not eco-friendly to use.

It cannot be affordable for real-time utilization due to dedicated and sensitive components.

3.2 PROPOSED SYSTEM:

When compared to the typical system of smart gloves for gesture, we build a single wearable device for disabled people with an embedded camera, mike and speaker, but the other user does not need a device.A normal person just needs to speak, so the vocal input is captured and converted to a sign language which is visible for a disabled person to understand.

In case of a differently abled people, with the help of an Embedded camera and speaker, the gestural reply is captured, analyzed and converted to vocal format which makes it understandable for the normal user.

Advantage:

One device is sufficient**.**

Lightweight and compact using 3D Modeling

Easy to handle

3.3 REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATIONS:

The system Requirement specification(SRS) document describes all data, functional and behavioral requirements of the software under production or development. System Requirements mentioned below are not most Necessary, the system is designed in a way of functioning with all devices that are comparable in using web browsers. It is produced at the culmination of the analysis task. The software and hardware requirements of this project are listed below,

3.4 HARDWARE REQUIREMENTS:

Raspberry pi 4
RPI Camera
Audio devices (Speaker, Mike)
Display screen
Supporting IoT Components

3.5 SOFTWARE REQUIREMENTS:

Mongo Database

Python, Embedded C++

VNC viewer

Azure IOT Hub

SYSTEM DESIGN

CHAPTER 4
SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM:

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap. The below
Architecture diagram depicts the overall system implemented in the project.

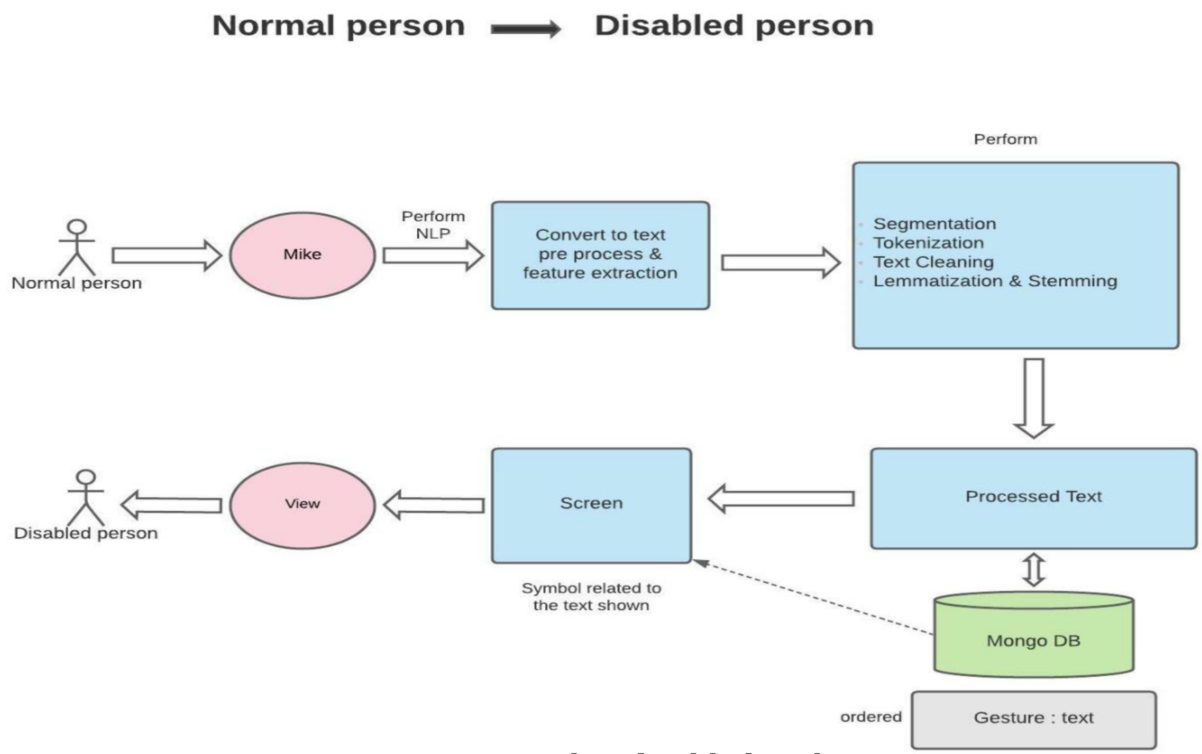


Fig 4.1.1 Normal to disabled architecture.

The normal person speaks through the mike to communicate. Then the vocal input is recorded which is analyzed and transformed into narrative using NLP (Natural Language Processing), then this raw data goes under various process to turn it into a processed data this process undergoes segmentation, tokenization, text cleaning, lemmatization and stemming. Now the unprocessed input is converted into a processed text, this processed text is compared with the text contained in the mongo database. The corresponding gesture images of the matched text are retrieved. The retrieved text is displayed through a screen to make it understandable for the disabled person. Thus, the communication between the normal and the disabled person is achieved.

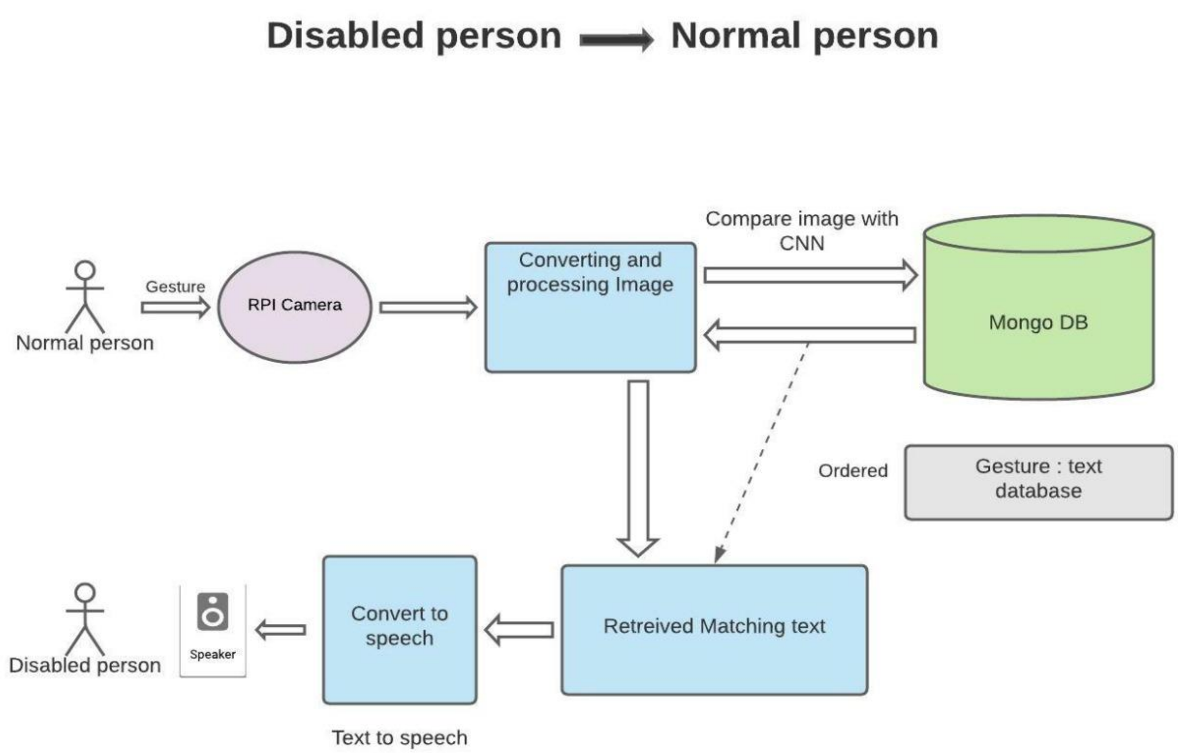


Fig 4.1.2 Disabled to Normal architecture

A disabled person performs gesture through sign language. This sign language is captured by the RPI camera, the image is converted from raw content into a processed image. This image is sent into the Mongo database for further development where it is Compared using CNN (Convolutional Neural Network) with the images in the database, the matched images are retrieved from the database as a text. Then this text is converted into speech using NLP (Natural language processing), this text is then fed into the speaker. Thus, the communication between the disabled and normal person is achieved.

4.2 FLOW CHART:

A flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams. Flowcharts, sometimes spelled as flow charts, use rectangles, ovals, diamonds and potentially numerous other shapes to define the type of step, along with connecting arrows to define flow and sequence. They can range from simple, hand-drawn charts to comprehensive computer-drawn diagrams depicting multiple steps and routes.

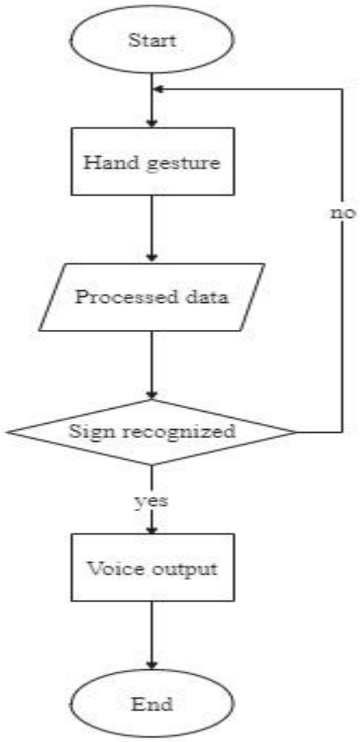


Fig 4.2.1 Flowchart I

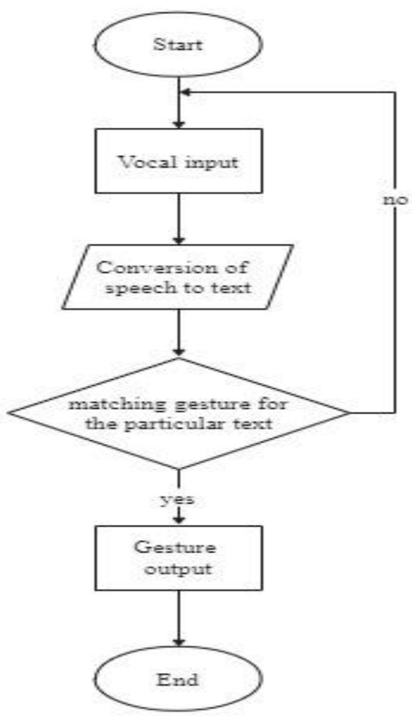


Fig 4.2.2 Flowchart II

4.3 DATA FLOW DIAGRAM:

A data flow diagram is a graphical tool used to describe and analyze movement of data in which the project progressed. In this, the workflow of the project is explained which includes the following steps:

The frames are extracted from the video conference screen using edge detection with the help of matplot library.The output from the conference screen is used as the system's input. To extract facial data sets and count heads, the input screen is converted to grayscale and fed to a model trained with haarCascade. The extracted facial dataset is fed into a Convolutional Neural Network (CNN) model with a long short-term memory (LSTM).

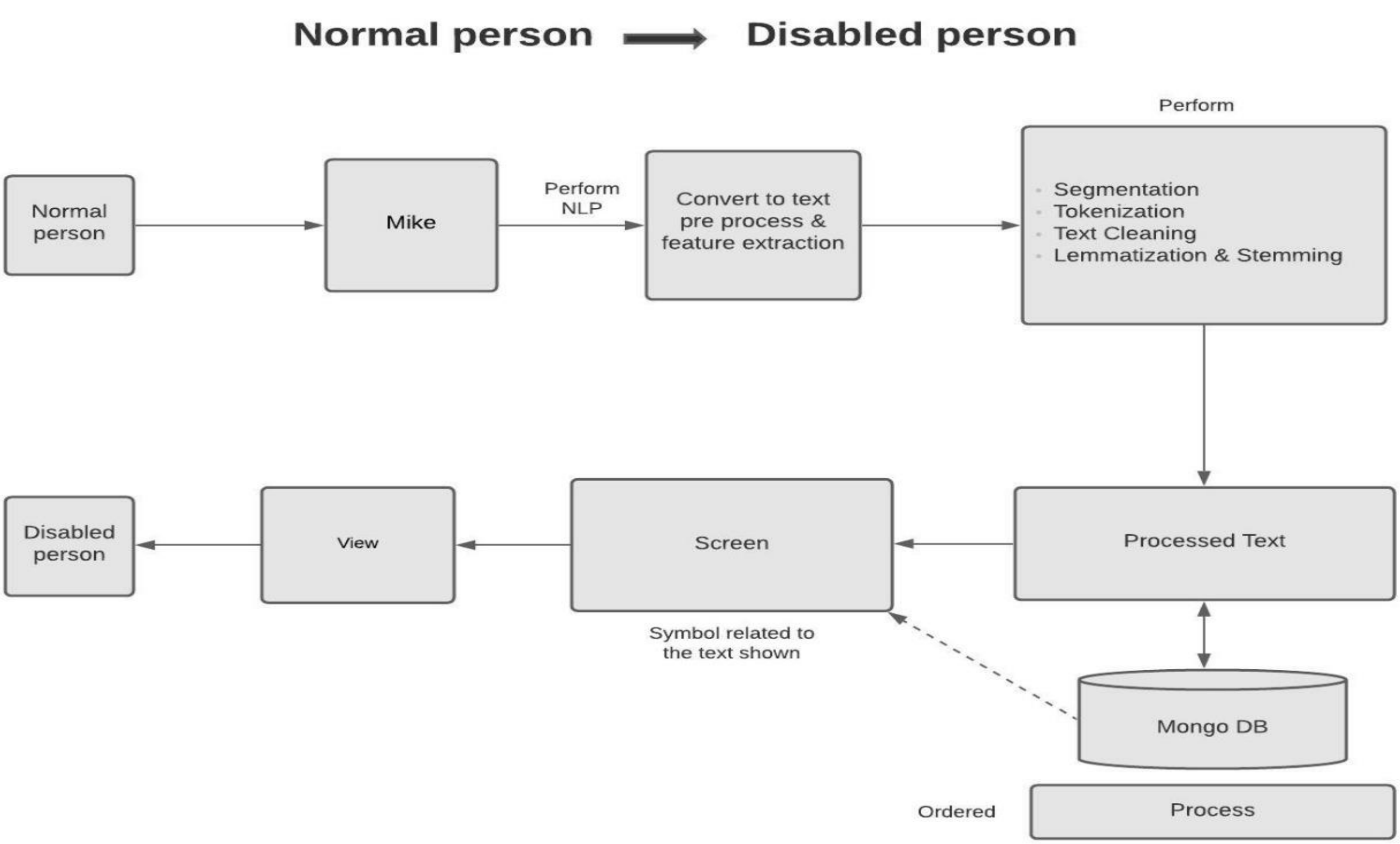


Fig 4.3.1Data Flow Diagram I

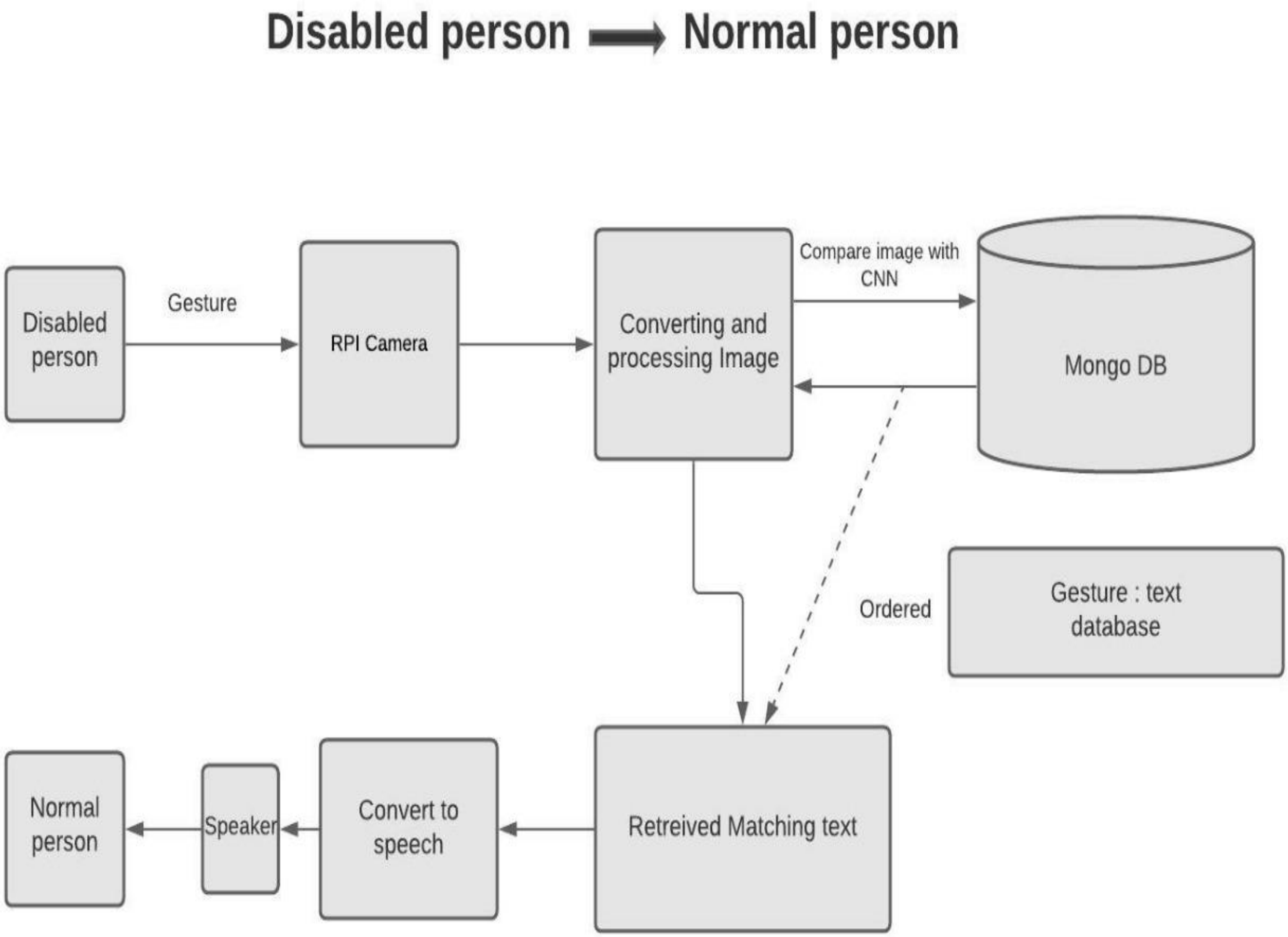


Fig 4.3.2 Data Flow Diagram II

4.4 USE CASE DIAGRAM:

The purpose of the use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and Statechart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified. When the initial task is complete, use case diagrams are modelled to present the outside view.

In brief, the purposes of use case diagrams can be said to be as follows −

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify the external and internal factors influencing the system.

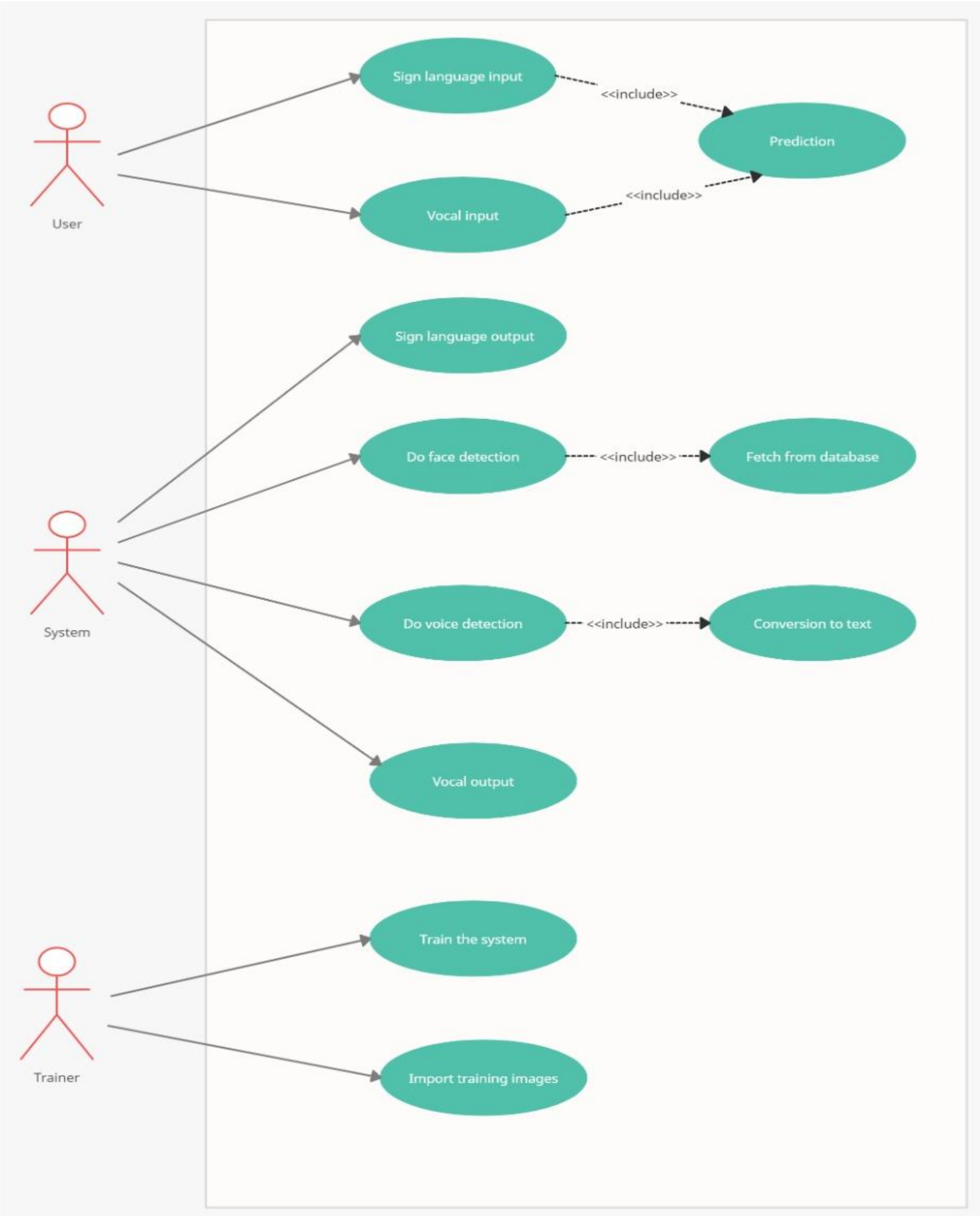


Fig 4.4.1 Use Case Diagram

The project workflow can be explained in following steps,

- ➢ Gathering data
- ➢ Data cleaning/Data preprocessing
- ➢ Researching the model that will be best for the type of data
- ➢ Training the model
- ➢ Evaluation.

Gathering Data:

The process of gathering data depends on the type of project we desire to make. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data.We can also use some free data sets which are present on the internet. Kaggle and UCI machine learning repository are the repositories that are used the most for making Machine learning models. Kaggle is one of the most visited websites that is used for practicing machine learning algorithms, they also host competitions in which people can participate and get to test their knowledge of machine learning. In this project we have used self-generated Data.

Data preprocessing/Data cleaning:

Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data And then replacing, modifying, or deleting the dirty or coarse data. Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing. In this project, initially, we have 10 image data of our Team Members. In order to avoid data that are repeated fields,

We use data cleanup processes using machine learning models. Data cleaning tasks are done using pandas, NumPy and seaborn libraries.

Evaluation

Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. To improve the model we might tune the hyper-parameters of the model and try to improve the accuracy and also looking at the confusion matrix to try to increase the number of true positives and true negatives.

ALGORITHM AND MODULE DESCRIPTION

CHAPTER-5

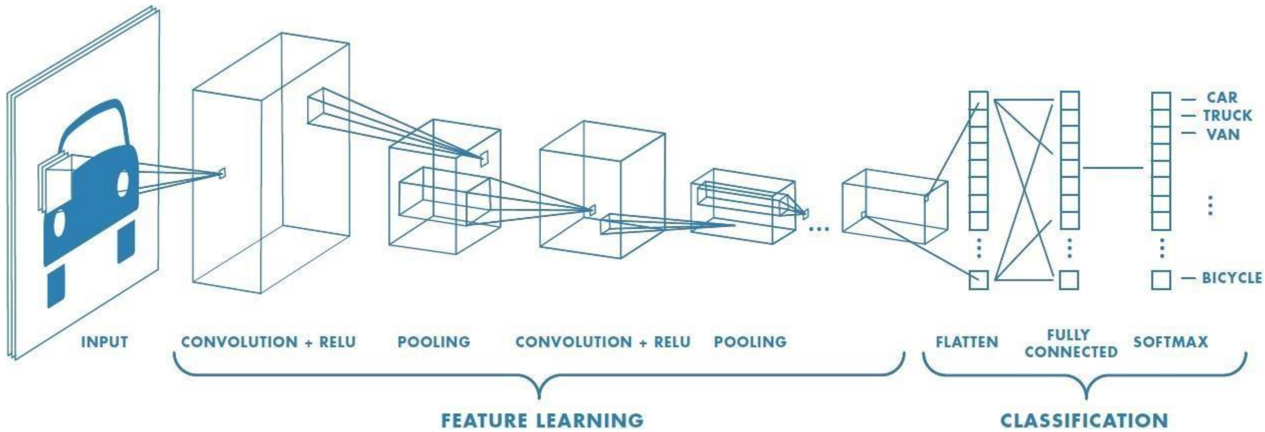
ALGORITHM AND MODEL DESCRIPTION

5.1 CNN based Training

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

In this project we use CNN to train the model based on Facial datasets. The model is later exported so that Further training can me done over cloud along with LSTM.



A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully
connected layer.

Convolutional layer:

The convolution layer is the core building block of the CNN. It carries the main portion of the network's computational load. This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field.

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

Fully connected layer:

Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect. The FC layer helps to map the representation between the input and the output.

5.2 RNN:

Recurrent Neural Network(RNN) are a type of Neural Network where the **output from previous step are fed as input to the current step**. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is **Hidden state**, which remembers some information about a sequencer have a **"memory"** which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

5.3 NLP

Natural Language Processing or **NLP** is a subfield of Artificial Intelligence research that is focused on developing models and points of interaction between humans and computers based on natural language. This includes text, but also speech-based systems. In the NLP context, a basic problem would be that for a given paragraph, the computer understands exactly the meaning of it and then possibly it acts accordingly. For this to work, we need to go through a few steps.

Tokenization Is the process of segmenting running text into sentences and words. In essence, it's the task of cutting a text into pieces called *tokens*, and at the same time throwing away certain characters, such as punctuation. Following our example, the result of tokenization would be:

Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. **Stemming** is important in natural language understanding (NLU) and **natural language processing** (**NLP**). ... **Stemming** is also a part of queries and Internet search engines.

Keyword Identification.

Keyword extraction (also known as *keyword detection* or *keyword analysis*) is a text analysis technique that automatically extracts the most used and most important words and expressions from a text. It helps summarize the content of texts and recognize the main topics discussed. Keyword extraction uses machine learning artificial intelligence (AI) with natural language processing (NLP) to break down human language so that it can be understood and analysed by machines. It's used to find keywords from all manner of text: regular documents and business reports, social media comments, online forums and reviews, news reports, and more.

5.4 LSTM

Long Short-Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN. A recurrent neural network is also known as RNN is used for persistent memory.

Let's say while watching a video you remember the previous scene or while reading a book you know what happened in the earlier chapter. Similarly, RNNs work, they remember the previous information and use it for processing the current input. The shortcoming of RNN is, they cannot remember Long term dependencies due to vanishing gradient. LSTMs are explicitly designed to avoid long-term dependency problems.

At a high-level LSTM works very much like an RNN cell. Here is the internal functioning of the LSTM network. The LSTM consists of three parts, as shown in the image below and each part performs an individual function.

The first part chooses whether the information coming from the previous timestamp is to be remembered or is irrelevant and can be forgotten. In the second part, the cell tries to learn new information from the input to this cell. At last, in the third part, the cell passes the updated information from the current timestamp to the next timestamp.

These three parts of an LSTM cell are known as gates. The first part is called **Forget gate, the** second part is known as **the Input gate** and the last one is **the Output gate**.

5.5 Module Description:

As the communication in the device is happened, the modules developed for the system includes

1) Training of Sign Language
2) Implementation of Sign Language using IOT
3) Training and Implementation using NLP

5.5.1 TRAINING OF SIGN LANGUAGE:

In this module, the image captured during the sign language communication is trained using CNN for image dataset, then using convolution neural network and artificial neural network the image is partitioned into pixels which is stored in layers. Then this image is recorded so that It can be matched with texts in the database to achieve accuracy. The voice captured during communication is trained for vocal dataset using NLP and RNN. Natural language processing is used to convert the text into a vocal format by performing segmentation, tokenization, and text cleaning. Now the processed vocal output is fed into the speaker. We also use ISTM for immediate recovery of data and to prevent false prediction. Recurrent neural network has a particular dataset for matching the text.

5.5.2. IMPLEMENTATION OF SIGN LANGUAGE IN IOT:

In this module, the RPI camera is used for capturing image with Azure IOT Hub with Azure ML Platform. IOT Hub is a managed service which is used for communication between attached devices and applications part of the IOT. It is a central message hub. Using this hub, we store the images of sign language gestures which are captured and trained to produce the vocal output. We match the gesture captured by our camera with the pre-defined images in the cloud for results. The hand gestures of sign language is captured and sent to the cloud for processing. Results after processing are fetched and data is stored in LSTM. For every wrong prediction it one step back and reiterates the prediction. For the right prediction the sample data is stored in LSTM. Long Short-Term Memory (LSTM) is an algorithm which is capable of learning by itself based on the problems and respective predictions of a sequence. They are a type of recurrent neural network. Using this, we predict the accurate gestures and its vocal output

5.5.3. TRAINING AND IMPLEMENTATION USING NLP:

In this module, Natural Language Processing is done over cloud in Azure. Natural Language Processing is used for the automatic manipulation of natural language, like speech and text, by software. Using NLP, we convert text into vocal output or vocal input to text. Auto correction is done on the vocal input and converted to text using RNN. RNN has a particular dataset for the matching text.

String stripping, stemming and keyword identification is done in processed text to detect accurate sign language. Stemming is the process of producing morphological type of a word. This is done by reduction of words from their word stem or root form. The original word may not be identical to the stem. Keyword extraction is a methodology to automatically detect important words for representation of a particular text. This is a very efficient way to get insights from a huge amount of unstructured text data.

SYSTEM IMPLEMENTATION

CHAPTER 6
SYSTEM IMPLEMENTATION

6.1 GESTURE.PY

```
import cv2
import numpy as np
import math
cap = cv2.VideoCapture(0)

while(1):

    try:  #an error comes if it does not find anything in window as it cannot find
contour of max area
          #therefore this try error statement

        ret, frame = cap.read()
        frame=cv2.flip(frame,1)
        kernel = np.ones((3,3),np.uint8)

        #define region of interest
        roi=frame[100:300, 100:300]
```

```
        cv2.rectangle(frame,(100,100),(300,300),(0,255,0),0)
        hsv = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)



    # define range of skin color in HSV
        lower_skin = np.array([0,20,70], dtype=np.uint8)
        upper_skin = np.array([20,255,255], dtype=np.uint8)

     #extract skin colur imagw
        mask = cv2.inRange(hsv, lower_skin, upper_skin)



    #extrapolate the hand to fill dark spots within
        mask = cv2.dilate(mask,kernel,iterations = 4)

    #blur the image
        mask = cv2.GaussianBlur(mask,(5,5),100)



    #find contours
        _,contours,hierarchy=
cv2.findContours(mask,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

   #find contour of max area(hand)
```

```
    cnt = max(contours, key = lambda x: cv2.contourArea(x))

#approx the contour a little
    epsilon = 0.0005*cv2.arcLength(cnt,True)
    approx= cv2.approxPolyDP(cnt,epsilon,True)



#make convex hull around hand
    hull = cv2.convexHull(cnt)

 #define area of hull and area of hand
    areahull = cv2.contourArea(hull)
    areacnt = cv2.contourArea(cnt)

#find the percentage of area not covered by hand in convex hull
    arearatio=((areahull-areacnt)/areacnt)*100

 #find the defects in convex hull with respect to hand
    hull = cv2.convexHull(approx, returnPoints=False)
    defects = cv2.convexityDefects(approx, hull)

# l = no. of defects
    l=0

#code for finding no. of defects due to fingers
    for i in range(defects.shape[0]):
        s,e,f,d = defects[i,0]
```

```
start = tuple(approx[s][0])
end = tuple(approx[e][0])
far = tuple(approx[f][0])
pt= (100,180)


# find length of all sides of triangle
a = math.sqrt((end[0] - start[0])**2 + (end[1] - start[1])**2)
b = math.sqrt((far[0] - start[0])**2 + (far[1] - start[1])**2)
c = math.sqrt((end[0] - far[0])**2 + (end[1] - far[1])**2)
s = (a+b+c)/2
ar = math.sqrt(s*(s-a)*(s-b)*(s-c))

#distance between point and convex hull
d=(2*ar)/a

# apply cosine rule here
angle = math.acos((b**2 + c**2 - a**2)/(2*b*c)) * 57


# ignore angles > 90 and ignore points very close to convex hull(they
generally come due to noise)
if angle <= 90 and d>30:
    l += 1
    cv2.circle(roi, far, 3, [255,0,0], -1)

#draw lines around hand
```

```
    cv2.line(roi,start, end, [0,255,0], 2)



l+=1

#print corresponding gestures which are in their ranges
font = cv2.FONT_HERSHEY_SIMPLEX
if l==1:
    if areacnt<2000:
        cv2.putText(frame,'Put hand in the box',(0,50), font, 2, (0,0,255), 3,
cv2.LINE_AA)
    else:
        if arearatio<12:
            cv2.putText(frame,'0',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
        elif arearatio<17.5:
            cv2.putText(frame,'Best of luck',(0,50), font, 2, (0,0,255), 3,
cv2.LINE_AA)

        else:
            cv2.putText(frame,'1',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

elif l==2:
    cv2.putText(frame,'2',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

elif l==3:

     if arearatio<27:
```

```
            cv2.putText(frame,'3',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
        else:
            cv2.putText(frame,'ok',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

    elif l==4:
        cv2.putText(frame,'4',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

    elif l==5:
        cv2.putText(frame,'5',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

    elif l==6:
        cv2.putText(frame,'reposition',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

    else :
        cv2.putText(frame,'reposition',(10,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

    #show the windows
    cv2.imshow('mask',mask)
    cv2.imshow('frame',frame)
except:
    pass



k = cv2.waitKey(5) & 0xFF
if k == 27:
    break
```

```
cv2.destroyAllWindows()
cap.release()
```

6.2 **Voice**

```
import speech_recognition as sr
from subprocess import call
import time
import cv2
import os

if _name=="main_":

    '''img=cv2.imread("/home/pi/Desktop/gesture/call_me.jpg")
    cv2.imshow("frame",img)
    cv2.waitKey(0)'''

    text=""
    r = sr.Recognizer()
    print("say {English or Indian } followed by the guesture Name")
    #call(["espeak","-ven-us+f4","ay {English or Indian } followed by the guesture
Name"])
    i=0
    while True:
        try:
            with sr.Microphone() as source:
                print("Hi if you have something to Convey or test you can proceed :")
                audio = r.listen(source)
```

```
            try:
                text = r.recognize_google(audio)
                print(text)
                break

            except:
                print("Sorry could not recognize what you said")
                i+=1
                if i>=5:
                    break
                #call(["espeak","-ven-us+f4","Did you say something ! ,sorry i could
not recognize"])
        except KeyboardInterrupt:
            print("Bye iam going out. ")
            break

    # When everything done, release the capture
    #cap.release()
#---------------------------------------------------------------------------------------------
--------------

    sl=text.split(' ')
    print(sl)
    for gesture in sl:
        if gesture.lower() == "call":
            print('in call')
            img=cv2.imread("/home/pi/Desktop/gesture/call_me.jpg")
```

```
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "luck":
            img=cv2.imread("/home/pi/Desktop/gesture/finger_cross.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "super" or gesture.lower() == "ok" :
            img=cv2.imread("/home/pi/Desktop/gesture/okay.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "paper" or gesture.lower() == "5" or gesture.lower()
== "five":
            img=cv2.imread("/home/pi/Desktop/gesture/paper.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "peace":
            img=cv2.imread("/home/pi/Desktop/gesture/peace.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
```

```
            #time.sleep(2)

        elif gesture.lower() == "zero" or gesture.lower() == "0" or gesture.lower() ==
"rock" or gesture.lower() == "first":
            img=cv2.imread("/home/pi/Desktop/gesture/rock.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "swag":
            img=cv2.imread("/home/pi/Desktop/gesture/rock_on.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "scissor" :
            img=cv2.imread("/home/pi/Desktop/gesture/scissors.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)

        elif gesture.lower() == "best" or gesture.lower() == "mountain" or
gesture.lower() == "Mountain":
            img=cv2.imread("/home/pi/Desktop/gesture/thumbs.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)
            #time.sleep(2)
```

```
        elif gesture.lower() == "loser" or gesture.lower() == "up":
            img=cv2.imread("/home/pi/Desktop/gesture/up.jpg")
            cv2.imshow("frame",img)
            cv2.waitKey(0)

        #cv2.waitKey(0)
        cv2.destroyAllWindows()

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    #time.sleep(2)
#--------------------------------------------------------------------------------------------
----------------------


    cv2.destroyAllWindows()
```

6.3 Test Script:

```
subprocess import call
import numpy as np
import math
import cv2

cap = cv2.VideoCapture(0)

gesture_text=""
gesture_text_count=0

while(True):
    gesture_text1=""

    try:  #an error comes if it does not find anything in window as it cannot find
contour of max area
          #therefore this try error statement

        # Capture frame-by-frame
        ret, frame = cap.read()

        kernel = np.ones((3,3),np.uint8)

        #region of interest
        roi=frame[100:350, 100:350]
        #cv2.imshow("roi",roi)
```

```
#put rectangle in the roi of frame
cv2.rectangle(frame,(100,100),(350,350),(0,255,0),0)

# Our operations on the frame come here
#gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
#convert roi to hsv
hsv = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)

# define range of skin color in HSV
lower_skin = np.array([0,20,70], dtype=np.uint8)
upper_skin = np.array([20,255,255], dtype=np.uint8)

 #extract skin colur imagw
mask = cv2.inRange(hsv, lower_skin, upper_skin)

#extrapolate the hand to fill dark spots within
mask = cv2.dilate(mask,kernel,iterations = 4)

#blur the image
mask = cv2.GaussianBlur(mask,(5,5),100)

#find contours
contours,hierarchy=
cv2.findContours(mask,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

#find contour of max area(hand)
```

```
cnt = max(contours, key = lambda x: cv2.contourArea(x))
#print(contours)
#print("------------------")
#print(hierarchy)
#print(cnt)
#break

#approx the contour a little
epsilon = 0.0005*cv2.arcLength(cnt,True)
approx= cv2.approxPolyDP(cnt,epsilon,True)



#make convex hull around hand
hull = cv2.convexHull(cnt)

 #define area of hull and area of hand
areahull = cv2.contourArea(hull)
areacnt = cv2.contourArea(cnt)

#find the percentage of area not covered by hand in convex hull
arearatio=((areahull-areacnt)/areacnt)*100

#find the defects in convex hull with respect to hand
hull = cv2.convexHull(approx, returnPoints=False)
defects = cv2.convexityDefects(approx, hull)

#print(len(defects),defects,defects.shape[0],defects.shape,sum(defects[0][0]))
```

```
#break
# l = no. of defects
l=0

#code for finding no. of defects due to fingers
for i in range(len(defects)):#.shape[0]):
    s,e,f,d = defects[i,0]
    start = tuple(approx[s][0])
    end = tuple(approx[e][0])
    far = tuple(approx[f][0])
    pt= (100,180)



    # find length of all sides of triangle
    a = math.sqrt((end[0] - start[0])*2 + (end[1] - start[1])*2)
    b = math.sqrt((far[0] - start[0])*2 + (far[1] - start[1])*2)
    c = math.sqrt((end[0] - far[0])*2 + (end[1] - far[1])*2)
    s = (a+b+c)/2
    ar = math.sqrt(s*(s-a)(s-b)(s-c))

    #distance between point and convex hull
    d=(2*ar)/a

    # apply cosine rule here
    angle = math.acos((b*2 + c2 - a*2)/(2*b*c)) * 57
```

```
                    # ignore angles > 90 and ignore points very close to convex hull(they
generally come due to noise)
                    if angle <= 90 and d>30:
                        l += 1
                        cv2.circle(roi, far, 3, [255,0,0], -1)

                    #draw lines around hand
                    cv2.line(roi,start, end, [0,255,0], 2)



                l+=1

                #print corresponding gestures which are in their ranges
                #print(l)
                font = cv2.FONT_HERSHEY_SIMPLEX
                if l==1:
                    if areacnt<2000:
                        cv2.putText(frame,'Put hand in the box',(0,50), font, 2, (0,0,255), 3,
cv2.LINE_AA)
                        gesture_text1="put hand in the box"
                    else:
                        if arearatio<12:
                            cv2.putText(frame,'0',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                            gesture_text1="zero"

                        elif arearatio<17.5:
                            cv2.putText(frame,'Best of luck',(0,50), font, 2, (0,0,255), 3,
```

```
cv2.LINE_AA)
                    gesture_text1="best of luck"

                  else:
                    cv2.putText(frame,'1',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                    gesture_text1="one"

            elif l==2:
                cv2.putText(frame,'2',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                gesture_text1="two"

            elif l==3:

                 if arearatio<27:
                    cv2.putText(frame,'3',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                    gesture_text1="three"
                 else:
                    cv2.putText(frame,'ok',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                    gesture_text1="ok"

            elif l==4:
                cv2.putText(frame,'4',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                gesture_text1="four"

            elif l==5:
                cv2.putText(frame,'5',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                gesture_text1="five"
```

```
                elif l==6:
                    cv2.putText(frame,'reposition',(0,50), font, 2, (0,0,255), 3,
cv2.LINE_AA)

                else :
                    cv2.putText(frame,'reposition',(10,50), font, 2, (0,0,255), 3,
cv2.LINE_AA)



            #-------------------------------------------------------------------------------



            #----------------------------------------------------------------------------------

                # Display the resulting frame
                cv2.imshow('frame',mask)
                cv2.imshow('frame1',frame)

            except:
                pass

            # speak gesture and increase count to identif stability:
            if gesture_text1==gesture_text:
                count+=1
                if count%10==0:
```

```
            print(gesture_text)
            call(["espeak","-ven-us+f4",gesture_text])
    elif gesture_text1 != gesture_text:
        count=0
        gesture_text=gesture_text1

    #print("you said Best of Luck")
    #call(["espeak","-ven-us+f4","You said Best of Luck"])

    # Display the resulting frame
    #cv2.imshow('frame',mask)
    #cv2.imshow('frame1',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

SYSTEM TESTING

CHAPTER 7
SYSTEM TESTING

7.1.1 GENERAL

Software testing is a crucial element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive. A strategy for software testing integrates software test case design methods as a well-planned series of steps that result in the successful construction of the software. Testing is the set of activities that can be planned in advance and structured systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively and strategically apply to both large and small-scale systems.

7.1.2 STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behaviour, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines to decrease the level of abstraction on each turn. A strategy for software testing may also be viewed in the context of the spiral.

Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in the source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Taking another turn outward on the spiral we encounter validation testing where requirements established as part of software requirements 44analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

7.2 TYPES OF TESTING

7.2.1. UNIT TESTING

Unit testing focuses verification effort on the smallest unit of the software design, the module. The unit testing we have is white box oriented and some modules are conducted in parallel.

7.2.2 WHITE BOX TESTING

This testing ensures that

- All independent paths have been exercised at least once.
- All logical decisions have been exercised on their true and false sides.
- All loops are executed at their boundaries and within their operational bounds.
- All internal data structures have been exercised to assure their validity. To follow the concept of white box testing we have tested each form. We have created independently to verify that Data flow is correct, all conditions are exercised to check their validity, all loops are executed on their boundaries.

7.2.3 BASIC PATH TESTING

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions.

The main steps for deriving the test cases were:

- Using the design on the code draw the corresponding flow graph.
- Determine the Cyclomatic complexity of the resultant flow graph, using formula:

$V(G) = E – N + 2$ or $V (G) = P + 1$ or 45

$V (G)$ = Number of Regions Where $V (G)$ is Cyclomatic complexity,
E is the number of edges, N is the number of flow graph nodes,
P is the number of predicate nodes.

From this we determine the basis of a set of linearly independent paths.

7.2.4 CONDITIONAL TESTING

In this part of testing each of the conditions were tested to both true and false aspects and all the resulting paths were tested .So that each path that may be generated on particular condition is traced to uncover any possible errors.

7.2.5.DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This type of testing is used only when some local variables are declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.

7.2.6 LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all the loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loops.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

7.3.TEST CASES AND REPORTS

The following are some of the important test cases that are to be run to ensure smooth functioning of the system.

.Facial Detection:

HOST ID	**MODULE NAME**	**INPUT**	**EXPECTED OUTPUT**	**ACTUAL OUTPUT**	**REMARKS**
TC01	Sign Language gesture Module	Detection of the sign languages	Corresponding meaning to those sign languages in vocal format	Pass	Module works as expected.
TC02	Vocal input Module	The message to be sent is spoken	The words spoken is converted into a sign language	Pass	Module works as expected.

CONCLUSION

CHAPTER 8

8.1 CONCLUSION:

This device solves one of the major problems of communication between the differently-abled and a normal person by making the process easier and portable. However, our project also overcomes the obstacles faced by the existing system in which the device has to be carried by both parties and is expensive and cannot be mass produced. This process is made hassle-free as a normal person doesn't have to know sign language to communicate. This device can be mass produced as it is cost efficient and can be used by many people to overcome everyday problems of communication. This device uses NLP and CNN to convert the speech-to-sign and sign-to-speech and displays it through a screen and outputs the speech through a speaker efficiently. Thus the main problem of bridging the ineffective communication between the normal and differently-abled is solved.

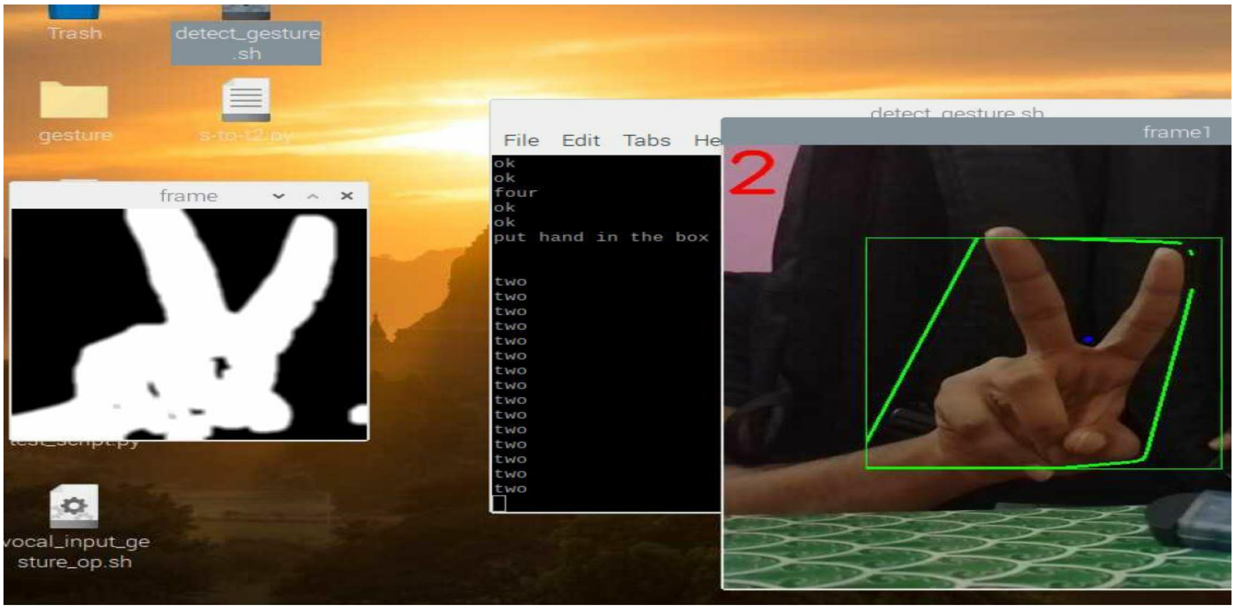
8.2 FUTURE SCOPE:

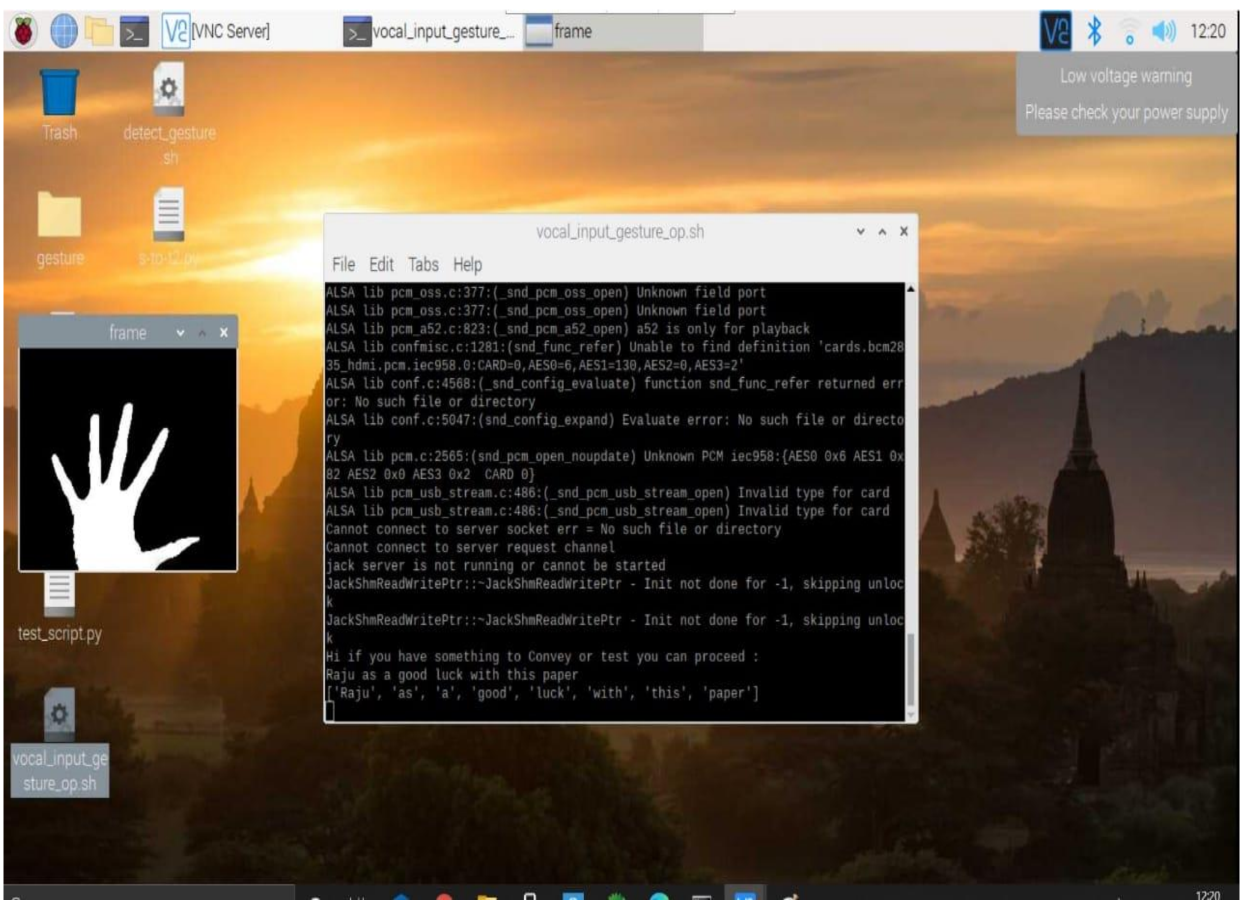
There are few who can neither see nor hear. For them, in the future we decide to enable touch sensitivity in the device which could help them communicate with others, so they won't have the communication gap. Thus, some of our future goals are to implement touch sensors in the device so it can be used by anyone with any disability.

APPENDICES

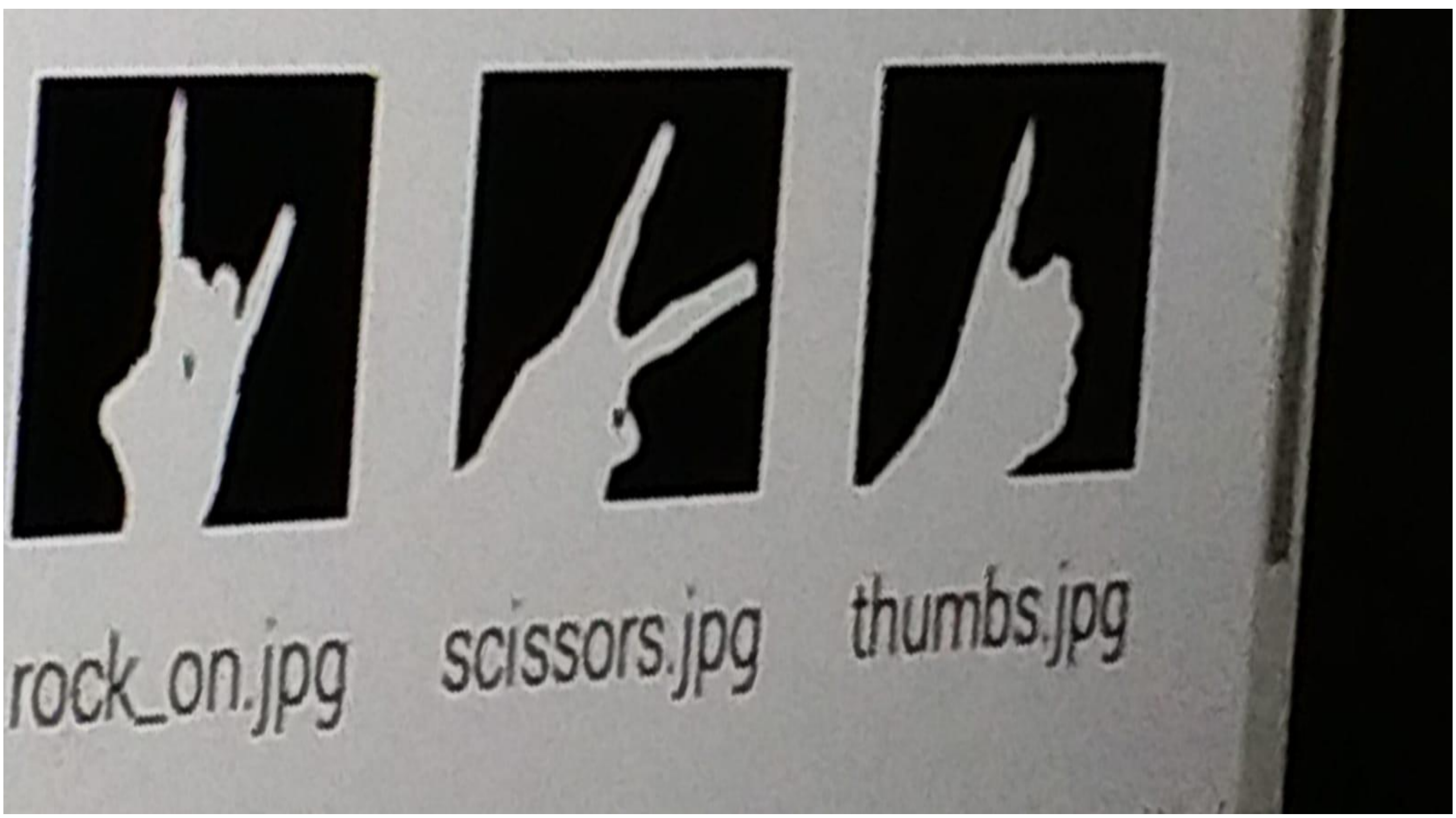
APPENDICES

A.1 SAMPLE SCREENSHOTS:






Raspberry Pi Camera
Rev 1.3


rock_on.jpg
scissors.jpg
thumbs.jpg

REFERENCES:

[1] Kevin Nathanael Kris Andria, Bima Sena Bayu Dewantara, Dadet Pramadihato. "HOG-based Hand Gesture Recognition Using Kinect". Faculty of Science, Mohammed V University, Vol. 4, Issue 4, April 2019``

[2] Shankar, Mallika Chowdhary,Priyadarshini. "IoT Device for Disabled People". VIT University, Chennai, (ICRTAC),Vol. 3 Issue 11, November- 2019.

[3] Dhaval L. Patel, Hershel S. Tapase, Praful A. Landge, Parmeshwar P. Prof. A. P. Bagade. "Smart hand gloves for disable people for communication". U.G. Scholar, Department of Electronics & Telecommunication Engineering. (IRJET), -ISSN: 2395-0056, Volume: 05 Issue: 04, Apr-2018

[4] Vikram Sharma M, Vinay Kumar N, Shruti C Masaguppi, Suma MN, D R Ambika, "Hand Gesture Recognition: A Literature Review", Panimalar Institute of Technology (IJCRT), Volume 8, Issue 3 March 2020

[5] Sanish Manandhar, Sushana Bajracharya, Sanjeev Karki, Ashish Kumar Jha. "Hand Gesture Vocalizer for Dumb and Deaf People.", Department of Computer Science, Nepal Engineering College (SCITECH), Volume 14 ,April 2019

[6] Ms. Pallavi Verma, Mrs. Shimi S. L., D r. S. Chatterji. "Design of Smart Gloves". International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181.Vol. 3 Issue 11, November-

2014.

[7] Rafiqul Zaman Khan and Noor Adnan Ibraheem, “Hand Gesture Recognition: A Literature Review”, International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.4, July 2012.

[8] Priya Matnani, “Glove Based And Accelerometer Based Gesture Control: A Literature Review”, International Journal of Technical Research and Applications e-ISSN: 2320-8163, www.ijtra.com Volume 3, Issue 6 (November-December, 2015), PP. 216-221.

[9] H.V Anupreethi and Vijaykumar, ―MSP430 based sign language recognizer for dumb patients‖, Elsevier International conference on modeling optimization and computing, Vol- 38, pp 1374-138, 2012.

[10] Xu Zhang, Xiang Chen, ―A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors‖, IEEE transactions on system, man, and cybernetics- part A: Systems and Humans, Vol-41, No- 6, pp- 1064-1078, Nov 2011.

[11] RuizeXu, Shengli Zhou, and Wen J. Li, ― MEMS Accelerometer Based Non-specific User Hand Gesture Recognition‖, IEEE Sensor Journal, Vol. 12, No. 5, pp: 1166-1173, May 2012.

[12] Anuja V. Nair et al, ―A Review on Indian Sign Language Recognition‖ International Journal of Computer Applications, Vol 73– No.22 pp 1-5, July 2013.

[13] G. Fang, W. Gao, and D. Zhao, ―Large vocabulary sign language recognition based on fuzzy decision trees, ‖ IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 34, no. 3, pp. 305–314, 2004.

[14] Vikram Sharma M, Vinay Kumar N, Shruti C. Masaguppi, Suma MN, D R. Ambika, ―Virtual Talk for Deaf, Mute, Blind and Normal Humans‖, Proceeding TIIEC '13 Proceedings of Texas Instruments India Educators' Conference, pp 316-320, 2003.

[15] Gupta, Dhiraj, ―Design and development of a low-cost Electronic Hand Glove for deaf and blind‖, 2nd International Conference on Computing for Sustainable Global Development (INDIA Com), pp 501-505, 11-13 March 2015.

[16]https://www.analyticsvidhya.com /blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/