VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018, Karnataka, INDIA

PROJECT REPORT
on

"Conversion of Hand Gestures to Text and Speech in Multiple Languages"

Submitted in partial fulfillment of the requirements for the VIII Semester

Bachelor of Engineering
IN
COMPUTER SCIENCE AND ENGINEERING

For the Academic year
2018-2019
BY

SAPTARSHI DUTTA GUPTA	**1PE15CS139**
SOMISETTY VR DINESH	**1PE15CS159**
NATASHA P	**1PE15CS095**
JYOTI M ANGADI	**1PE16CS410**

Under the Guidance of
Prof. Preethi Sangamesh
Assistant Professor, Dept. of CSE
PESIT-BSC, Bengaluru-560100

PES INSTITUTIONS

Department of Computer Science and Engineering
PESIT BANGALORE SOUTH CAMPUS
Hosur Road, Bengaluru -560100

PESIT BANGALORE SOUTH CAMPUS

Hosur Road, Bangalore -560100

Department of Computer Science and Engineering

CERTIFICATE

Certified that the project work entitled ***"Conversion of Hand Gestures to Text and Speech in Multiple Languages"*** *is a bonafide work carried out by* ***Saptarshi Dutta Gupta, Somisetty VR Dinesh, Natasha P*** *and* ***Jyoti M Angadi*** *bearing USN* ***1PE15CS139****,* ***1PE15CS159, 1PE15CS095*** *and* ***1PE16CS410*** *respectively, students of* ***PESIT Bangalore South Campus*** *in partial fulfillment for the award of* ***Bachelor of Engineering*** *in* ***Computer Science and Engineering*** *of the* ***Visvesvaraya Technological University****, Belagavi during the year 2018-2019. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated and the project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.*

Signatures:

Project Guide **Prof. Preethi Sangamesh** Assistant Professor, Dept. of CSE PESIT-BSC, Bengaluru	Head Dept of CSE **Dr. Sandesh B J** Professor, Dept. of CSE, PESIT-BSC, Bengaluru	Director/Principal **Dr. J. Suryaprasad** PESIT-BSC, Bengaluru

External Viva

Name of the Examiners	**Signature with date**
1. _____	_____
2. _____	_____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned my effort with success.

We are indebted to our Guide, **Prof. Preethi Sangamesh**, Assistant Professor, Department of Computer Science and Engineering, PESIT - Bangalore South Campus, who has not only coordinated our work but also given suggestions from time to time.

We are also extremely grateful to our Project Co-ordinators, **Prof. Keerti Torvi,** Assistant Professor**, Prof. Sangeetha R,** Assistant Professor**, Prof. Prethi Sangamesh,** Assistant Professors, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for their constant support and advice throughout the course of preparation of this document.

We are greatly thankful to **Dr. Sandesh B J**, Professor and HOD, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for his able guidance, regular source of encouragement and assistance throughout this project.

We would like to express our immense gratitude to **Dr. J. Suryaprasad,** Director and Principal, PESIT Bangalore South Campus, for providing us with excellent infrastructure to complete our project work.

We gratefully acknowledge the help lent out to us by all faculty members of the Department of Computer Science and Engineering, PESIT Bangalore South Campus, at all difficult times. We would also take this opportunity to thank our college management for the facilities provided during the course of the project. Furthermore, we acknowledge the support and feedback of my parents and friends.

Saptarshi Dutta Gupta
Somisetty VR Dinesh
Natasha P
Jyoti M Angadi

ABSTRACT

For ages, communication has always been the medium for conveying expressions and information among individuals. However, there are people who are unfortunately born without the power of hearing or speaking. Communication for speech impaired people has always been a major challenge.

For comprehending what they want to convey, we need to understand their hand gestures or sign language. Therefore, our proposed model aims at helping the differently abled communicate more effectively with people by converting their hand gestures into speech in multiple languages according to the user choice.

CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Purpose

- The process in which 2 or more people want to convey a meaningful message using a shared system consisting of signs and semantic rules is called communication. One of the important goals of communication is to understand the motive and meaning of the information and to pass the learned information to others.
- Our communication is unique due to the wide range of languages we use, but not everyone is fortunate enough to be born with the ability to speak or hear. Therefore they depend on sign language to convey their messages. But people who are not differently abled don't understand these hand gestures.
- The proposed model aims at helping the differently abled communicate more effectively with people by converting their hand gestures into speech in different languages.

1.2 Scope

- The scope of this project is to provide an efficient, easy and mechanism for the automatic translation of static and dynamic hand gestures to textual and speech version in multiple languages.
- Individual can communicate by using hands gestures rather than by speaking since our model since our model will map a particular hand gesture into various ASL characters and colloquial words.

1.3 Literature Survey

1. **A Hand Gesture Recognition using Feature Extraction:** In this paper, A Pradhana, M.K. Ghosea and M.Pradhana use webcams and mobile integrated cameras to collect the pictures of the hand gestures shown and to later classify them using the predefined hand gestures stored in the system. This system has two main parts which are pre- processing and classification of the hand gesture using template matching. Thresholding is used to convert the hand image to binary form an median filter is used to remove any noise that the image may contain.

2. **Intelligent Sign Language Recognition Using Image Processing:**S pramada used HCI(human computer interface) to determine hand gestures, it consisted of four main parts which were camera interfacing, image processing, pattern matching and text and audio generation. This system basically took the colour image of the hand gesture and collected the RGB(red,blue,green) values of this image and made only the required part of the hand gesture image to white using thresholding and the rest is made black.

3. **New Methodology for Translation of static sign Symbol to words in Kannada language:** This paper written by R. Kagalkar, Kannada Sign Language (KSL) and Hearing- Impaired (HI) were used as the sign language, it consisted of 36 kannada letters. Histogram technique was used for feature extraction, the gesture was extracted from the image using Hough segmentation and the classification is done using neural networks that is trained using the training data.

4. **Hand Gesture Recognition based on Digital Image Processing using MATLAB:** This paper was proposed mainly by Tahir Khan for American Sign Language (ASL), it used digital Image Processing, Colour Segmentation, Skin Detection and Image Segmentation to classify the ASL hand gesture obtained. It used template matching and feature detection to recognise the gesture.

5. **Gesture to Speech Conversion in Hindi language for Hearing and Speech disabled person in INDIA:** This paper is mainly designed for Hindi hand gesture recognition by A.Singh and B.Timande. The Hindi language has 46 letters to be considered. Gesture area segmentation was used to obtain the depth of the image, the image was converted from colour to black and white using fast indexing of the SURF algorithm. Neural Network and feature extraction using Principal Component Analysis were the two main techniques used for the classification of the image. Windows text to speech was the API used for speech recognition.

6. **Hand gesture recognition and voice conversion system for dumb people:** V.Padmanabhan, M.Sornalatha have used a hardware based system for hand gesture recognition. A glove which compromised of flex detector, instrument sensors was used to obtain the hand gesture. The flex detector was used to measure the degree to which the fingers are bent while the accelerometer measured the tiling of the hand

1.4 Existing Systems

There are existing systems which converts Hand Gestures to Text and Speech using Pattern Matching, Neural Networks and Image Processing Algorithms. There are also several hardware based system which uses micro controllers and hand gloves for converting the required gesture to text.

1.5 Proposed System

The proposed model automates the process of hand gesture detection by capturing the gestures from a web camera video feed. Our system will also take into account gestures which includes A-Z, 0-9 and also several colloquial hand gestures which are used in different parts of India. Finally, another addition to our model is the conversion of the gestures into a multitude of languages according to the choice of the user.

1.6 Statement of the problem

Conversion of hand gestures of speech impaired people into a multitude of languages(both text and speech) for reducing the difficulty in communication between the differently abled and normal people with the help of image processing and machine learning algorithms.

1.7 Summary

This section explains to us what exactly the system is meant to do. It gives an insight into how out system is going to function and how we can create a communication between the differently abled and the normal people.

Chapter 2

System Specifications

2.1 Software Requirements Specifications

This document will give a description of the system, it will explain the purpose and features of the software, the interfaces of the software, what the software will do, the constraints under which the system must operate and how it will react to external stimuli. This document is intended for developers and the end users. Requirements specification is the activity of translating the information gathered during the analysis into a requirements document.

2.2 Operating Environment

This section gives a brief about the hardware and software prerequisites for the project.

2.2.1 Hardware Requirements

- **Processor**: 1.6GHz or faster processor
- **RAM**: 4 GB(32 bit) or 4GB(64 bit)
- **Storage**: 100 of available hard disk space
- GPU is preferable for training of the model
- Other general hardware such as a mouse, keyboard and webcam for inputs and a monitor for display.
- Speakers to get the voice output from device.

2.2.2 Software Requirements

- **Operating system**: Ubuntu 14.04 and above
- **Programming languages**: Python3
- **Packages**: OpenCV, Tensorflow, Keras, Django

- **API**: Microsoft Text To Speech, Googletrans
- **Documentation**: Overleaf

2.3 Functional Requirements

Functional requirement describes the functionality of a system software and its various components. A functional requirement defines the various inputs outputs and the behaviour, it can be in the form of data manipulation technical or calculations or it can be a goal that the system should accomplish. The functional requirements describe the core functionality of the system and includes the system and user requirements.

1. The camera should be able to capture and track the hand movements of the user.
2. There must be no object placed between the user and camera.
3. The must be able to detect the position of the hand and the path of gesture drawn.
4. The system must be able to extract the coordinates of each pixel in the gesture.
5. The system should compare the predefined gesture with selected gesture.
6. If there is no match found with gesture then the error message is to be generated.

2.4 Non-Functional Requirements

he requirements that are not directly related to the specific functions which are delivered by the system, but they are actually the constraints that are placed on the functions and services offered by the system. These constraints include constraints on development process, time, standards which can be anything like the data representation used in system interface or the input/output devices. Non-functional requirements apply to the whole system rather than individual features or services.

- Usability: The proposed system is implemented to be user friendly, and is built in such a way that even a person who has no idea how to use it can grasp it without any difficulty. It also provides a well-formed graphical user-interface which is easily understood by the individual handling the system.

- Performance: The main goal of the software is to reduce the number of calculations for processing the hand gestures and detecting it with great accuracy.
- User's Tolerance: The loss caused due to vision based interactions should be reduced, to avoid any type of loss and instead of giving a wrong answer when the input provided by the user is incorrect it should ask the user for another input.
- Extensibility: The model must be built in order to support the future developments, extensions and add-ons to the existing gestures.
- Maintainability: For the maintainability of the software is required to be designed using modular programming approach that is different modules are to be designed performing specific functions and are independent of other modules.

2.5 Applications

Hand Gesture Recognition is applicable to, but not limited to, the following areas:

1. **Sign language Recognition**: Sign language is the mean of communication that is used by the differently abled to communicate with the outside world. These languages in nature are highly structural they can be used as test beds for any visual algorithms. These languages not only help disabled people interact with other people but it can be used by them to interact with computers as well. American sign languages is one of the most popular sign languages out there.
2. **Robotics:** Another interesting application of hand gesture recognition system is its use in robotics. We can have a camera that collects live feed from the camera and recognizes the gesture which in turn is converted to action performed by the robots like, movements of their hands and legs or reaching out for another object or just causing them to move from point A to point B.
3. **Immersive game technology:** There are many popular games out there that recognize hand gesture to cause particular actions in the game. In Freeman the players hand movements are tracked to control the different objects and cars in the game, another popular game is Konrad in which hand gestures were used to move and control the aviators in a virtual environment. The PlayStation is using these technologies to implement their newer games such as eye toy.
4. **Virtual Reality:** One of the greatest and most trending application on gestures is in virtual reality. Here the gestures are recognized and used to implement actions on the virtual objects, it

can also give a realist feel of holding the object. They are also used in 3D and 2D to show how a design may look once implemented in real.

2.6 Advantages of The System

A few advantages of the proposed system are:

- Developing means to communicate for the people impaired of hearing.
- To help the young children interact with the computer.
- Making use of the virtual environment by navigating and manipulating it.
- Means for normal people to communicate with disabled people.
- Provide support for distance learning and tele-learning.
- Medically checking patients in the emergency wards.

2.7 Summary

Hardware and software aspects of the system are discussed here. Also, the requirements which can be non-functional or functional for the same are analyzed. The various applications where the project can be useful and finally the obvious advantages of using the project.

Chapter 3

High Level Design

This chapter covers the design technique of the entire system which involves the implementation of the entire model as a whole.

3.1 Design Considerations

Here are two methodologies for software designing:

- Top-down Design:It takes the entire programming framework as one entity and after that disintegrates it to accomplish in excess of one subsystem or some components based on few attributes.
- Bottom-up Design: The model begins with most particular and essential components. It accedes with making more elevated amount out of subsystems by utilizing essential or lower level components.

3.1.1 Assumptions and Dependencies

Our system makes certain assumptions which includes:

- The hand gestures made by the user are already known to the computer by the training data it was provided.
- We assume that the camera is still or fixed so it can click proper images of the hand.
- It is assumed that hand movements are kept to a minimal while clicking the pictures.
- Finally, the system is designed for American sign language so it is also assumed that the hand gestures given as input to the computer are of the required form.

3.1.2 Goals and Constraints

The main goal of this project is to create a model that enables the differently abled to communicate with ease with the rest of the world just like any other person would. Further it also intends to make this communication process as simple as possible so that anyone can understand and make use of it. Finally our goal is for people from different parts of the world to communicate with the differently

abled, therefore it provides hand gesture to speech conversion in multiple languages.

The constraints of this project is that the hand gestures have to be in American sign language, and the system must be trained with all possible hand gestures as well as different variations of this hand gesture(like the distance and angle between the fingers),for the hand gesture to be recognized.

3.2 System Architecture

The diagram shown below is the architecture of the whole system that enables us the distinguish between the various segments of the system like the primary segments and the interfaces for them. The user inputs a query which can be either a new hand gesture or an already present gesture to the application through the webcam. The applications takes this input from the user and crops out only the hand portion of the image where the gesture is shown. Then the application preprocesses the image to get the image in the desired form. Once the image is processed the next step is feature extraction.

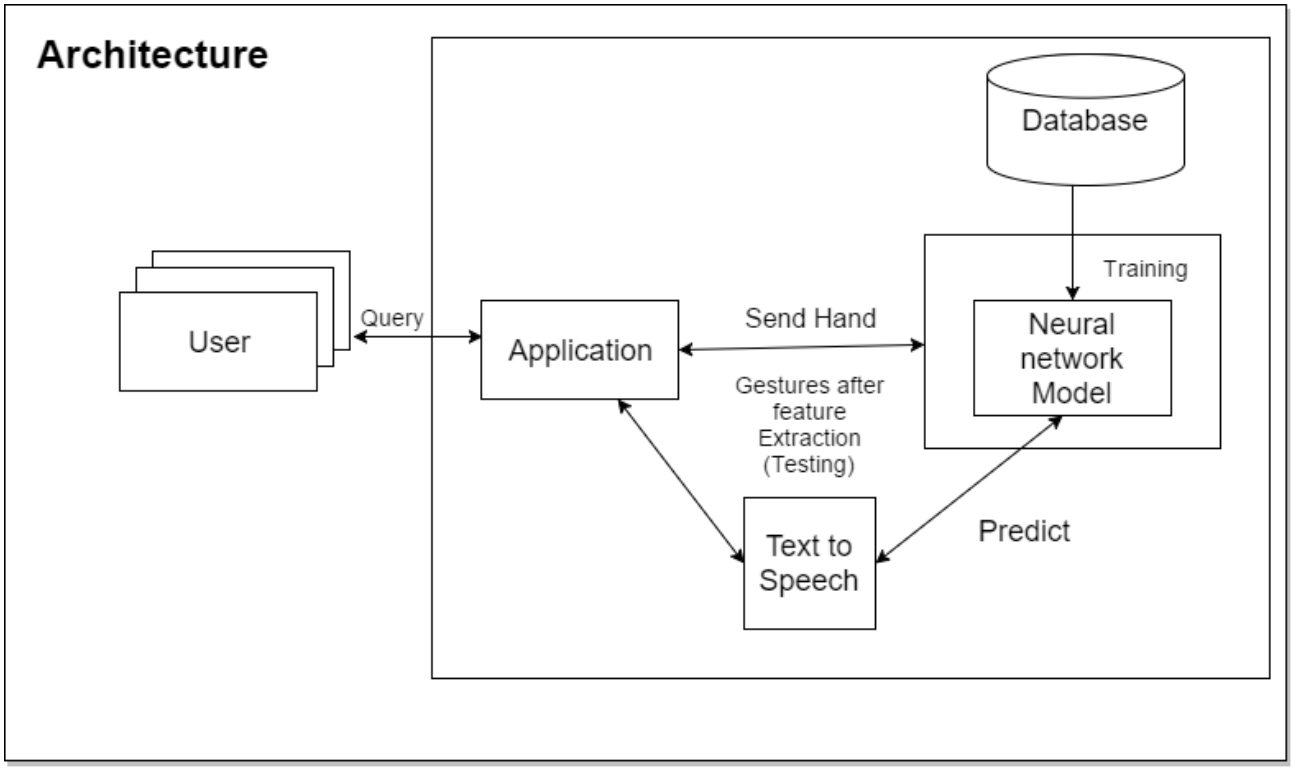

Figure 3.1: System Architecture

This feature extracted image is then passed on to the next module which is the neural network model. It predicts the best gesture by finding the closest match. The corresponding text output for the predicted

gesture is produced. Then this text output is converted to speech in the desired language input by the user and is given as the output.

3.3 Data Flow Diagram

A dataflow diagram also represented as DFD shows the flow of data or information throughout the system. Some of the other popular uses of DFD is in the visualization of the data processing stage. A DFD shows what kind of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. Using a data flow diagram we can see what kind of data comes into the system, what kind of data is produced as output from the system and which data it store in the system, but it does not give any information about the time taken to process the data or in which sequence they are processed.

Level 0: This is the level where the user shows the hand gesture to the webcam. The gesture is then captured by the gesture recognition system. Level 1: This level consists of the gesture recognition system. Here the captured image is processed to get it in the desired form and to finally predict the gesture that it represents and convert it to text output. Level 2: This is the last level that basically outputs the shown gesture in desired speech output.

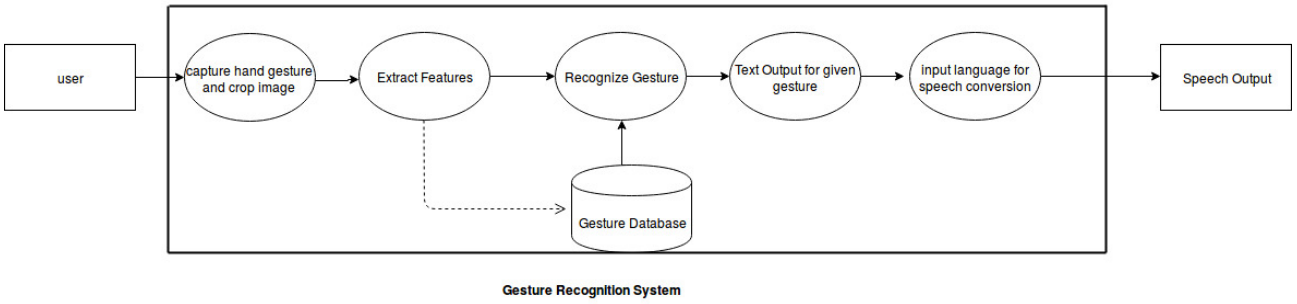

Figure 3.2: Data Flow Diagram

3.4 Sequence Diagram

A graphical representation which demonstrates how objects interact and is arranged in time sequence is known as sequence diagram. This representation shows all the classes and objects that are a part of

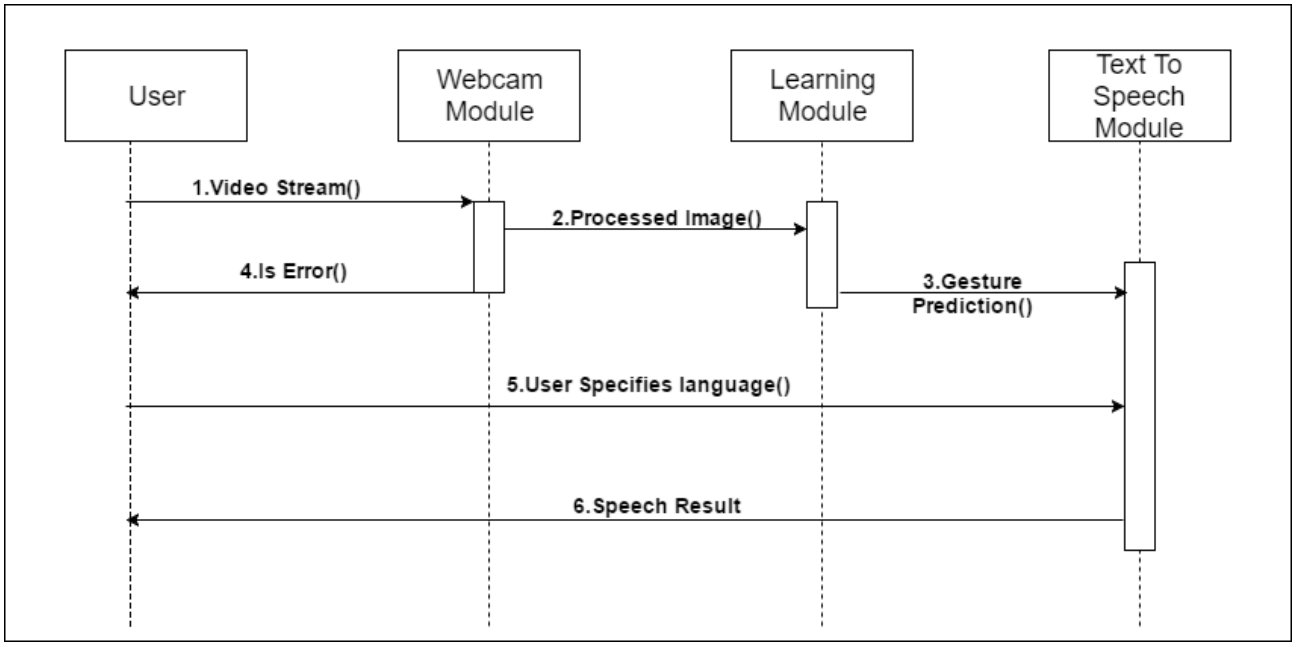

Figure 3.3: Sequence Diagram

the the scenario and also the various message sequence exchanged between the objects which is needed to implement the functionality of the scenario. Parallel vertical lines (lifelines) in a sequence diagram shows different processes or objects that live simultaneously whereas horizontal arrows shows the messages exchanged between them in the specific order in which they occur. Our sequence diagram consists of four objects: the user, the webcam module, learning module and the text to speech module.

In the first time sequence the video stream function is used to capture the image in the webcam module. In the next time sequence the webcam module inputs this image to the learning module that processes this image and predicts the gesture. If no gesture is predicted then an error is sent to the user, otherwise the user inputs the language preference for speech output to the text to speech conversion module and the result is returned back to the user.

3.5 Activity Diagram

The graphical representations of work flows of activities step wise with additional support for iteration,choice and concurrency is known as activity diagram.Both computational and organizational processes are modeled through activity diagram in the Unified Modeling Language. Further, data flows intersecting with the related activities are also modeled. The figure shown below is an activity

diagram for hand gesture to speech conversion system. The system first takes input of the hand gesture through the webcam and it is processed to get the image in binary format. Then we verify whether the hand gesture is captured properly, if it is then it goes on to the next step if not then an error message is generated.

Subsequently, we detect the edges of the hand using Canny Edge Detection Algorithm in order to obtain the feature vector. This feature vector is then given to the neural network that predicts the text output for the given input image. Lastly another input is taken from the user for their preferred language and the speech output is given.

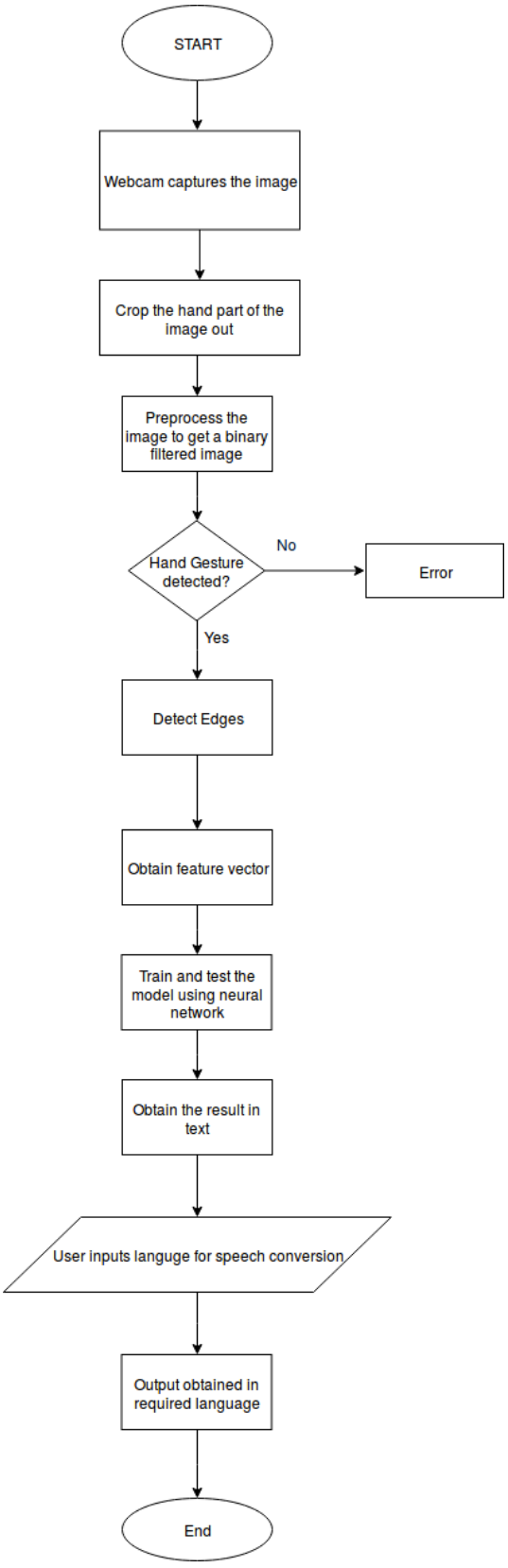

Figure 3.4: Activity Diagram

3.6 Use Case Diagrams

In system analysis, use case is made to recognize, elucidate, and sort system requirements. Unified Modeling Language makes use of use case diagram for modeling real-world objects and systems.

The various components are:- - The system of interest or the boundary. - Actors with their roles - The use cases - How the actors and the use cases interact and the relationship between them

The use-case diagram modeled by us has 2 actors which are the user and the system, which interact with the hand gesture recognition system. Once the user provides the gesture to the system the neural network model processes it and predicts the gesture in text format. Then the user inputs their language preference to the system which converts it to the requested language and outputs it to the user.

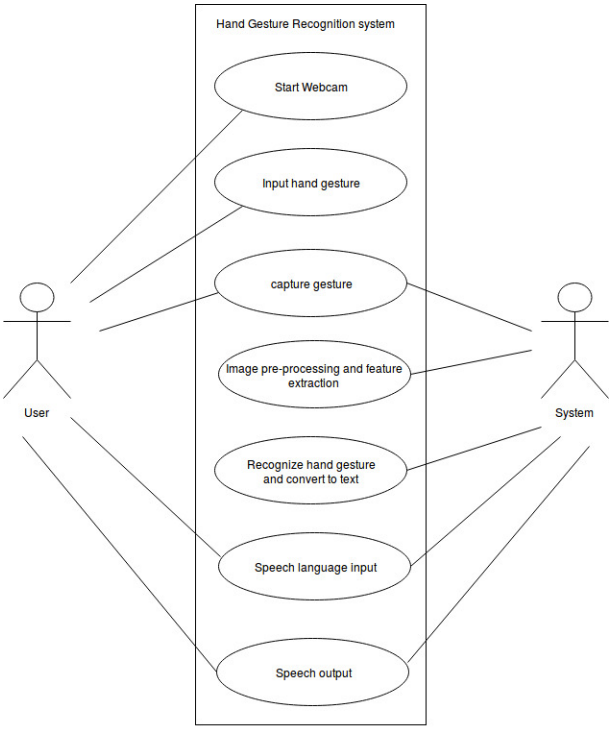

Figure 3.5: Use case Diagram

3.7 Summary

In this chapter, we discussed have the different design patterns which can be used in any product development cycle. We also discussed the flowchart which shows the data flow between various components. This chapter aims to showcase the high level design of a hand gesture to speech conversion system. It explains how hand gestures in American sign language are converted to speech using the various representation including use-case, sequence, data flow and activity diagrams.

Chapter 4

Detailed Design

4.1 Purpose

Detailed design aims at meeting all the requirements identified at the start. In the detailed design we see what is the input data for each model, how the model implementation is carried out and how the output is interpreted.

Among the deaf and dumb, hand gestures is a very convenient form of communication. Therefore, a natural interaction between humans and computing devices can be achieved by using hand gestures for communication between them

4.2 Module 1: Image Acquisition

In this step a GUI is made, which shows the video stream of the scene. From this GUI, when the spacebar is clicked, it takes the image of the scene. The problem is that this scene includes the whole body and other unwanted objects as well. Therefore, we are cropping out only the hand portion which our application is performing.

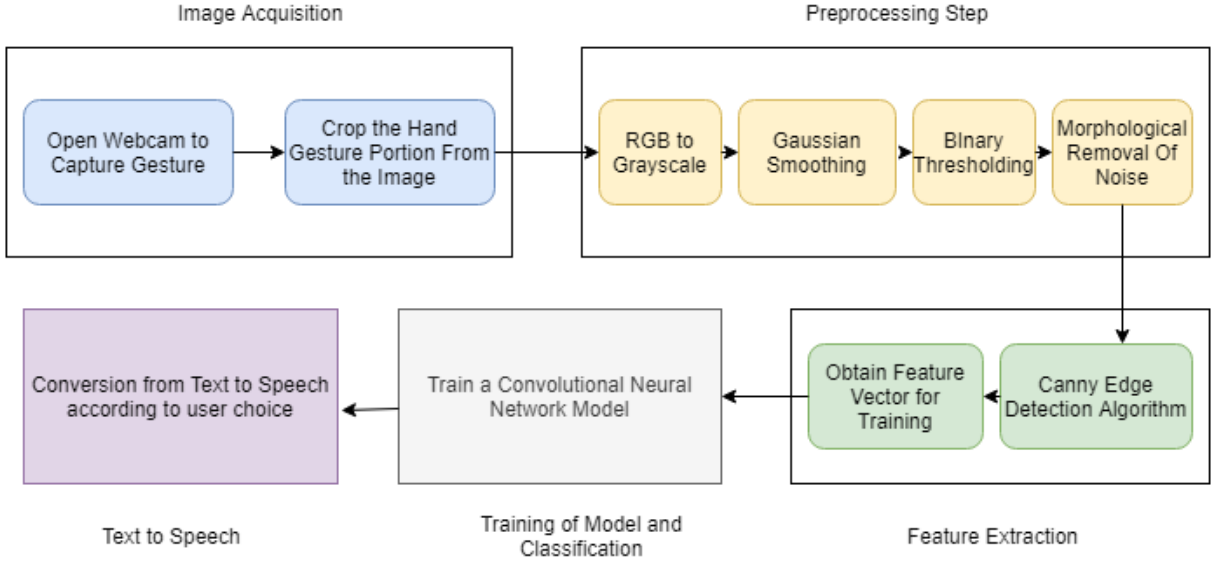

Figure 4.1: Detailed Design

4.3 Module 2:Preprocessing Phase

Preprocessing of the image needs to be performed in order for the queried gesture to be made ready for the learning model.

The following phases are being followed for preprocessing. The preprocessing steps is shown in the figures for the letter 'L'

4.3.1 Converting from RGB to gray scale

A combination of Red, Green and Blue are mixed in order to get the different colours ranging from 0 to 255. Colours have different intensities and shade. Human vision in particular has the capability to distinguish various types of colours. Almost 100 shades of gray can be differentiated by human beings.Therefore,conversion of the cropped from RGB to gray scale is necessary so that maximum information can be extracted

Figure 4.2: RGB to Gray scale

4.3.2 Image smoothing using Gaussian Conversion

A Gaussian function is used to blur an image and also computes what kind of transformation that has to be applied to every pixel of the image. In case of computer vision algorithms Gaussian blurring is used to intensify image details.

4.3.3 Conversion to Binary Image

In order to convert a gray scale image to binary image, the image segmentation method which is used is known as thresholding.Thresholding methods basically does the following :

- If the image intensity I(i,j) is less than some fixed constant T then every pixel in the image is replaced by a black pixel(i.e. I(i,j)¡T)
- Otherwise a white pixel considering the image intensity is greater than the constant.

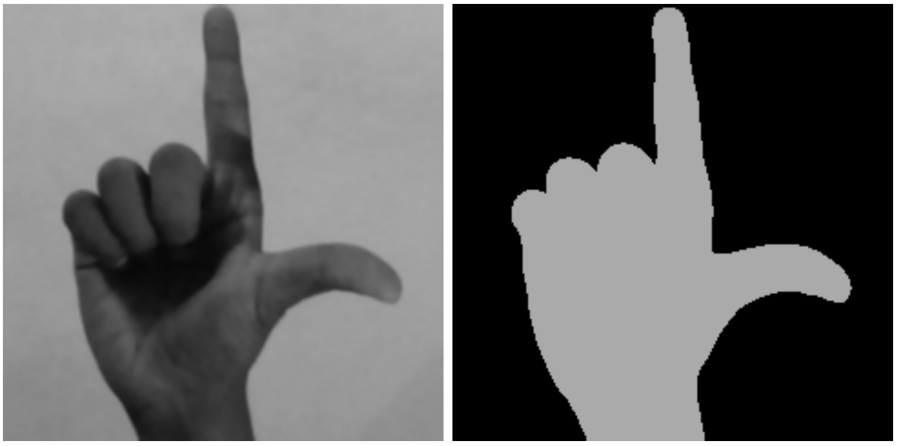

Figure 4.3: Grayscale to Binary

4.3.4 Removing noise using Morphological Technique

Morphological transformations are applied on binary images. The original image and the structuring element i.e. kernel are the two inputs of this operation. Erosion and Dilation which are the two basic morphological technique are discussed below:

- Erosion: The border of foreground object are eroded as the kernel starts skimming through the image . An original image pixel which assumes 0 or 1 will be taken as 1 if the pixels beneath the kernel is 1. In all other cases it is eroded i.e. assumed to be 0.
- Dilation: An original image pixel which assumes 0 or 1 will be taken as 1 if atleast one pixels beneath the kernel is 1. Hence the foreground image size increases.

Erosion is almost always followed by dilation. Although erosion eliminates white noises it also causes the object to shrink. Therefore, dilation is necessary.

4.4 Module 3: Feature Extraction

4.4.1 Detecting Hand Edges By Canny Edge Detection Algorithm

Canny Edge Detection was introduced by John F. Canny in 1986. The various stages in this multi-stage algorithm is as follows:

1. Noise Reduction: A 5x5 Gaussian filter is used to eliminate noise from the image since the process of detecting the edge is vulnerable to noise.
2. Finding Intensity Gradient Of the Image: A Sobel kernel is used to get first derivative in horizontal direction (Gx) and vertical direction (Gy). The edge gradient and direction for each pixel is calculated as shown in (1) and (2) :

$$IntensityGradient(G) = \sqrt{G(x)^2 + G(y)^2} \tag{4.1}$$

$$Angle(\theta) = \text{tan inverse}\,\frac{G(y)}{G(x)} \tag{4.2}$$

3. Non-maximum Suppression: In the third stage, any undesired pixel which may not be the edge, is scanned so that it can be eliminated. To implement the above, the local maximum in its neighborhood in the direction of gradient at every pixel is examined.
4. Hysteresis Thresholding: In the last stage, the final edges are selected. Two threshold values namely MinVal and MaxVal are used in which:
   - If the intensity gradient is greater than MaxVal they are confirmed to be edges
   - If it is below MinVal then it is discarded
   - If the intensity gradient lies in between the thresholds, based on the connectivity they will be considered ot discarded.

   The image below describes the process Since A is above MaxVal it is taken to be a sure edge. C is connected to A in spite of being below MaxVal and therefore it is considered as valid edge. In case of edge B, it is above MinVal but not connected to any sure edge and therefore it is discarded.

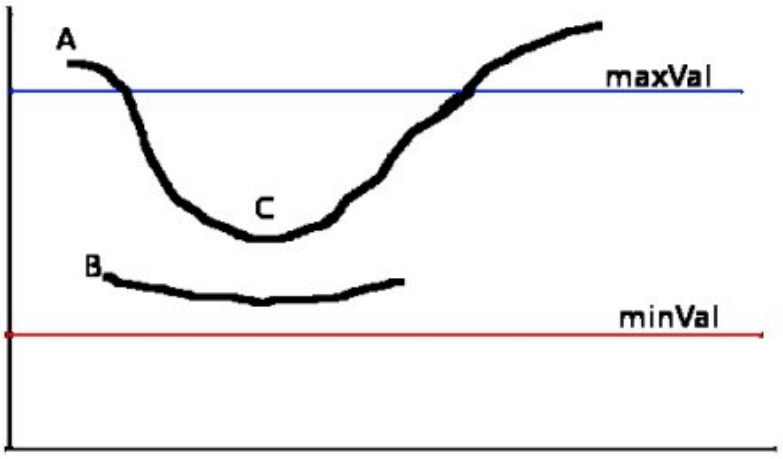

Figure 4.4: Hysteresis Thresholding

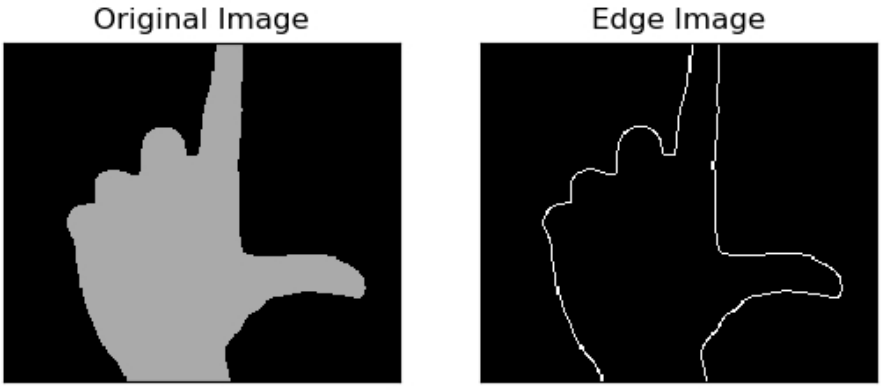

Figure 4.5: Canny Edge Detection

4.5 Module 4: Training and Classification

For the training and classification phase a deep learning model known as Convolutional Neural Network(CNN) is being used. In deep learning, a convolutional neural network (CNN, or ConvNet) is A class of deep neural networks, CNN in most cases is used to analyze images.

Fully connected networks in which each neuron in a particular layer is connected to all the other neurons in the next layer is known as multilayer perceptrons. A systematized version of CNNs are multilayer perceptrons.

Like all other neural networks, CNNs are also inspired by biological processes in context of the connectivity patterns among the neurons. There is a restricted region defined as receptive field in which the individual cortical neurons respond.

The three layers in a convolutional neural network are the input layer, multiple hidden layers and finally an output layer . Most commonly, the hidden layers consists of the convolutional layer, the activation functions, the pooling layers and finally the fully connected layer which in our case is the Softmax function.

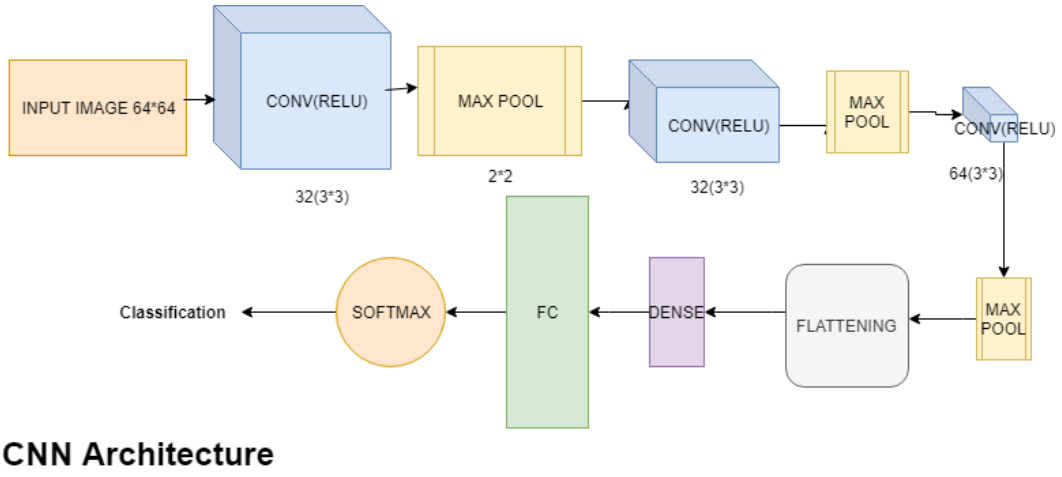

Figure 4.6: Architecture of The Convolutional Neural Network

4.5.1 Convolutional Layer

The Convolution Layer is responsible for extracting high-level features from the input image. Features such as colours, gradient orientation, edges are usually captured by the first convolutional layer. As layers keep getting added, the system architectures gets used to several high level features too therefore helping us to understand the image dataset better

The convolutional layer consists of the following:

1. Filters - These are called as “Neurons” of the network layer. They include input weights and output of value. Each input has a fixed size square called “Receptive Field”.
2. Feature maps – The outcome of filters applied to previous layers. Filter is drawn and each pixel is moved at a single instance of time. This draws to the activation of the neuron which is further stored in our feature map.
3. Zero Padding - The optimization of the output is achieved through 3 parameters and they are:

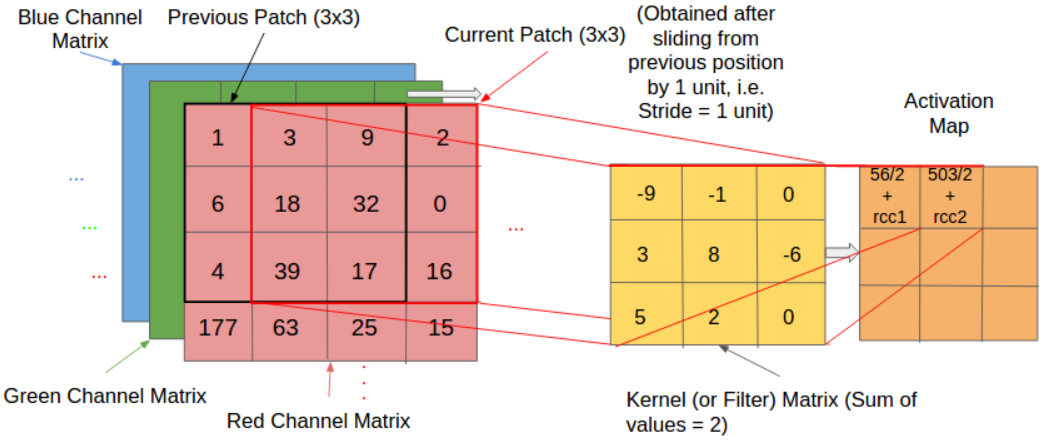

Figure 4.7: Pixel wise computation in convolution layer

- Depth: Depth of the output is permitted to set manually through the neurons to a same region of input. By reducing this parameter, the total number of neurons of the network can be reduced but the capability to recognize the patterns can also be reduced.
- ii. Stride: Stride is defined as the quantity by which the filter is moved and by default it is set to one. By setting to larger value reduces the amount of overlapping and output produced will be in lower spatial dimensions.
- Padding: Padding the input image borders with zero. This method is used to alter spatial dimensionality of convolution layer.

4.5.2 ReLU Activation Function

The activation function, ReLU or Rectified Linear Unit is mathematically defined as y = max(0, x). Graphically it can be represented as shown in Fig 4.8 In order to introduce nonlinear factors into linear convolutional operations ReLU is used. ReLU can also be used for:

1. getting rid of unnecessary data
2. increasing the chance of retention of the features of the image
3. building a sparse matrix

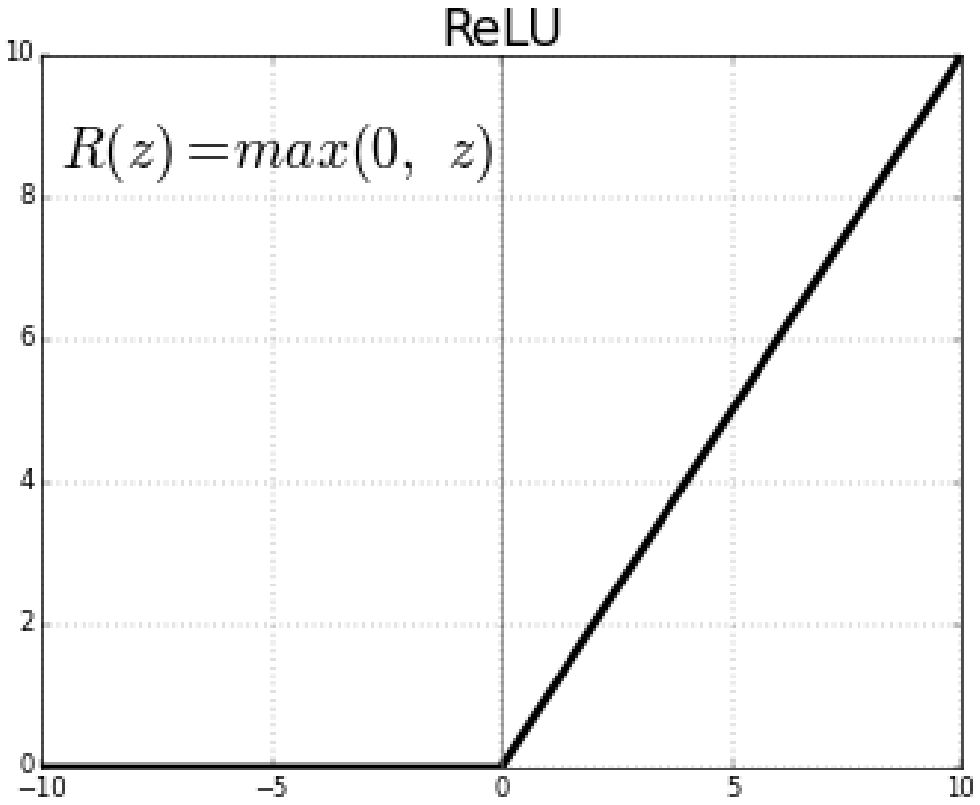

Figure 4.8: ReLU Activation Function

4.5.3 Pooling Layer

As same because the Convolutional Layer, the Pooling layer is to decrease the dimensions of the convolved Feature. this may decrease the machine power needed to method the information by spatiality reduction.

It is also used for extracting important features which are rotational and in different positions, It will lead the process of better training of the model. There are two different types of Pooling: Max Pooling and Average Pooling. The extreme value from the partition of the image by the Kernel is returned by max pooling whereas in Average Pooling the means of every values from the partition of the image covered by the Kernel is returned. Pooling consists of four steps:

1. A window size is selected.
2. Pick a stride
3. The window is moved across the preprocessed image.

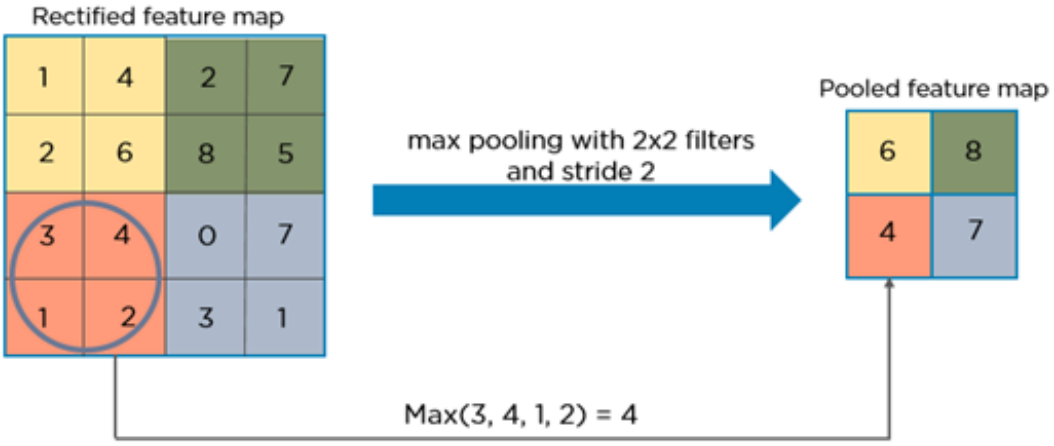

Figure 4.9: Max Pooling Operation

4. The highest value from every window is selected

Max Pooling handles noisy activations. It also performs removal of noise and also dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Therefore, Max Pooling performs better than Average Pooling. Now that we have validated the model to understand the features, next it will go to the flattening layer.

4.5.4 Flattening Layer

Flattening transforms a 2-D matrix of features into a vector of features that can be fed into a neural network or classifier (Fig 4.9)

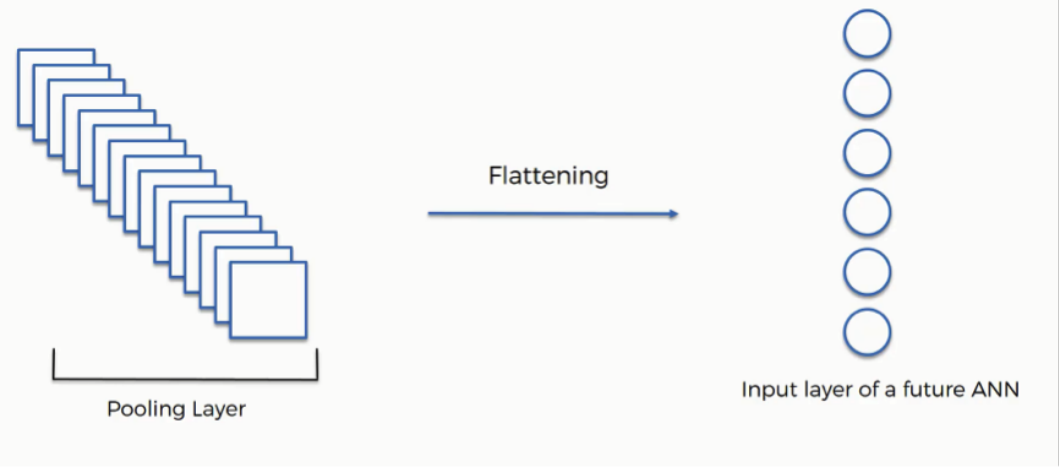

Figure 4.10: Flattening Operation

4.5.5 Fully Connected Layer

Every neuron in a particular layer is connected to all other neurons in the various other layers in a fully connected system. The classification of images is done when the flattened matrix enters the FC layer. This layer is feed forward layer and allows non-linear combination of features in order to predict probabilities of class. These are used after feature extraction is per-formed. Another way of CNN architecture is to mound two convention layers before pooling each layer. This is recommended for more complex features of input vector. The CNN is the most powerful learning algorithm in which the knowledge of specific type of input is exposed.

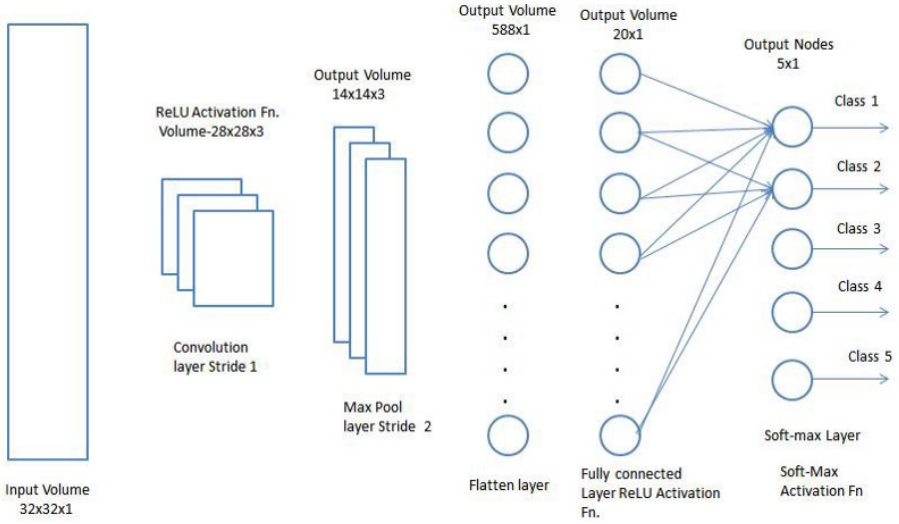

Figure 4.11: The Fully Connected Layer

This full connection process practically works as follows:

- The neuron in the fully-connected layer detects a certain feature; say, a nose.
- It preserves its value.
- It communicates this value to all the classes.
- Both classes check out the feature and decide whether it's relevant to them.

Dropout is also employed in this layer to ignore random neurons and to resist over-fitting.

4.5.6 Softmax Activation Function

Softmax function gives the vector as a output which gives the probability distributions for the list of most valued outcomes. It is one of the most important element used in deep learning classification tasks. If the score (input called logit) is large then the Softmax output is large and vice versa. Being exponential and thus increasing differences, Softmax makes the scores aka logits into probabilities. Cross entropy (cost function) is the most used function for output of softmax and true labels.

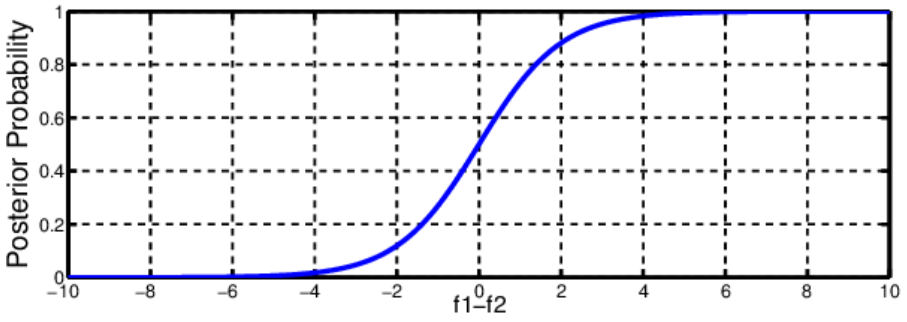

Figure 4.12: Softmax Activation Function

In mathematics, the softmax function, also known as softargmax or normalized exponential function,is a function that takes as input a vector of K real numbers, and normalizes it into a probability distribution consisting of K probabilities. That is, prior to applying softmax, some vector components could be negative, or greater than one; and might not sum to 1; but after applying softmax, each component will be in the interval (0,1) and the components will add up to 1, so that they can be interpreted as probabilities. Furthermore, the larger input components will correspond to larger probabilities. Softmax is often used in neural networks, to map the non-normalized output of a network to a probability distribution over predicted output classes.

4.6 Module 5: Conversion From Text To Speech

The artificial production of human speech is called as Speech synthesis. The speech computer or speech synthesizer used for this purpose, and can be implemented in both on the software or hardware products. The system converts normal language text into speech is called as text-to-speech; other systems pictures the symbolic linguistic representations like phonetic transcriptions into speech.

The concatenation pieces of recorded speech that are stored in a database are which used to create Synthesized speech. According to the size of the stored speech units the systems differ. The system

which stores the phones or diphones provides the larger range of output, it may lack in clarity. For high-quality output, the storage of entire words or sentences for specific usage domains. In another procedure, a synthesizer is modelled and trained for the vocal tract and other human voice characteristics to generate the complete voice output which is ”synthetic” in nature.

The similarity to the human voice and by its ability to be understood clearly are considered to judge the quality of the speech generator or synthesizer . An intelligible text-to-speech software helps the people with visual impairments or reading disabilities to hear to written words from documents which are the home computer. Once the gesture is classified into its correct type in text format, we need to convert it into appropriate speech according to the choice of the user. The default language is English.

The following options are being provided: French-fr, Spanish-es, German-de, Italian-it, Dutch-en and Portuguese-pt.

4.7 Summary

The detailed design of all the five modules of our system are explained and this chapter tells us how the modules are related and the entire system in functioning.

Chapter 5

Implementation

5.1 Programming Language Selection

We are using Python v3.6 which is a general-purpose, high level programming language. Python enables programming that is lucid and easy to understand on both big and small scales.

In the case of Image Processing and Deep Learning applications, Python is the best choice. Numpy, OpenCV, pandas, matplotlib and several other open source packages in python helps in preprocessing and creation of the model with ease.

Features of Python:

1. Python is developer-friendly and high level programming language which is easy to learn and use.
2. Python is an expressive Language which means it is more understandable and readable.
3. Python is a portable and cross-platform language because it can executed on different platforms such as Windows, Linux, Unix and Macintosh etc.
4. Python is Free and Open Source which makes it freely available at official web address. The source code of the python was freely available and its open source make its largely usable by companies and developers.
5. Python has GUI Programming Support Graphical, thus user interfaces can be developed by using it. source-code is also available. Therefore it is open source.
6. Python has a large and broad library and provides rich set of module and functions for rapid application development.
7. Python is a pure object oriented language thus it supports concepts of classes and objects come into existence.

5.2 Platform Selection

Ubuntu is an operating system which constructed from the Debian Linux distribution The operating system is available freely. The Ubuntu project is one of the project which is committed to the principles of free software development, people are encouraged to use free software to improve it.

Some Important features of Ubuntu are:

- Office softwares are pre-installed
- Good User Interface
- Simplicity, Reliability and Stability
- Security and speed
- Available For Almost all type of hardware
- Available in many languages.
- Large Community Support

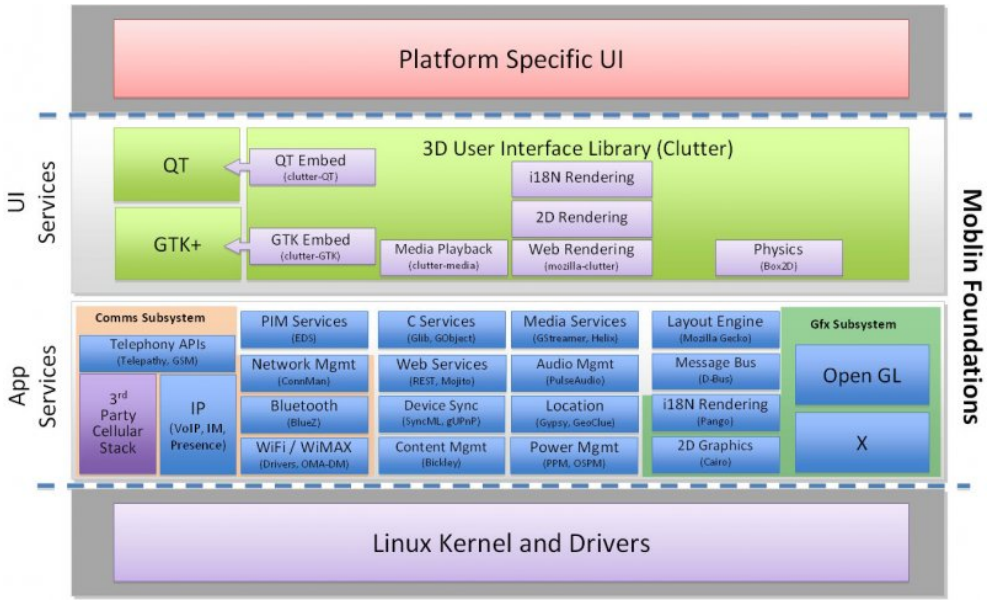

Figure 5.1: Ubuntu Architecture

5.3 Libraries used

5.3.1 Preprocessing Implementation

For all preprocessing tasks, OpenCV Python has been used. OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed

by Intel, it was later supported by Willow Garage. The library is cross-platform and free for use under the open-source BSD license.

For all preprocessing tasks, OpenCV Python has been used. is A library of programming functions, OpenCV (Open source computer vision) was developed to support computer vision problems. OpenCV is free for use by the open-source BSD license and also has cross-platform features. OpenCV has package includes several shared or static libraries which means that it is in modular structure. The following modules are available:

- core - a closed and neatly packed module which contains the basic data structures, including the very dense multi-dimensional array Mat and basic support functions used by all other modules.
- calib3d - this module contains the basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo related algorithms, and basic elements and components of 3D reconstruction.
- gpu - this module contains GPU-accelerated algorithms from different modules.
- imgproc - an image processing module that contains the linear and non-linear image filtering, geometrical image transformations i.e for resize, affine and perspective warping, generic table-based remapping and color space conversion and histograms, and so on such packages.
- highgui - it is an easy-to-use interface for video capturing, image and video codecs, as well as it contains simple UI capabilities.
- video - The module contains motion estimation, background subtraction, and object tracking algorithms is called video-analysis module

Functions used:

- **cv2.VideoCapture()**: This function has been used for opening the webcam to capture the required image.
- **cv2.imread():** Used to read an image. The image is in the current working directory or a full path of image should be given if it not in the same directory.
- **cv2.imshow()**: It is used to render an image in a window. The window adjusts fits to the image measurements. First argument is a name of the window is a string. second argument is image path.
- **cv2.imwrite()**: it is used to save/store an image. First argument is the name of the file, second argument is the image which is to be saved.

- **cv2.waitKey()** : It is a keyboard event listening function. the argument to it is the time in milliseconds. The function waits for the given milliseconds for any keyboard press event. If you press/hit any key in that given time, the program continue to execute.
- **cv2.destroyAllWindows()** : It is Used to destroy all the windows we created when there is no argument passed. If we want to destroy any specific window, the function cv2.destroyWindow() should be given exact window name as argument.
- **cv2.selectROI()**: It is used select a region of interest or region specified, which is used for cropping out the hand portion of the image.
- **cv2.GaussinBlur()**: The Gaussian blur is used for image smoothing. We should give the measurements width and height of kernel should be in positive and odd. We ma also pass the standard deviation in X and Y direction, sigmaX and sigmaY respectively. If only sigmaX is specified, sigmaY will be taken same as sigmaX. If they are given as zeros, will be calculated from kernel size.
- **cv2.threshold()**: For converting the blurred image to binary the cv2.threshold is used. First argument should be the source image path or source image and it should be a grayscale image. Threshold value should be second argument and it is used to classify the pixel values. maxVal should be third argument and it represents the measure of the value to be given if pixel value is more than (sometimes less than) the threshold value. The fourth argument give the different styles of thresholding.
- **cv2.Canny()**: it is the single function which has all the above functions in open cv. Input image is the first argument .minVal and maxVal are the second and third arguments respectively.Forth argument is the size of the aperture. It is used for find image gradients by acting as size soble kernel. By default it is 3.L2gradient is the last argument which gives the equation for finding gradient magnitude.

5.3.2 CNN Model Implementation

For CNN implementation, Keras and Tensorflow has been utilized

Keras is a Python based library which can be used to implement several neural network models.TensorFlow, Theano or Microsoft Cognitive Toolkit can be used as abackend while running Keras. Keras is extremely user-friendly and can be used extensively for deep learning models because of its modular nature.

TensorFlow can be used for differentiable programming for a variety of tasks. It is open-source containing various mathematical libraries along with machine learning and deep learning applications.

Our model consists of three convolutions and max pooling layer with the ReLU Activation function, one flattening and one fully connected layer. For the purpose of classification, the Softmax activation function has been implemented.

The several layers in the CNN model has been implemented as follows:

1. **Convolution Layer:** Code snippet for the convolution layer is as: classifier.add(Convolution2D(32, 3, 3, inputshape = (64, 64, 3), activation = relu)) The first parameter indicates the number of filters used, second and third parameter are the filter sizes, fourth parameter is the size of the input image which is fixed at 64*64 and finally a 'relu' activation function has been used.

2. **Pooling Layer:**Code snippet for the Pooling layer is as: classifier.add(MaxPooling2D(poolsize =(2,2))). The pool size (2,2) indicates the window size.

3. **Flattening Layer:** Code snippet for the Flattening layer is as: classifier.add(Flatten())

4. **Fully Connected Layer:** Code snippet for the FC layer is as: classifier.add(Dropout(0.5))

   classifier.add(Dense(26, activation = 'softmax'))

   A Dropout of 50% has been applied and the Dense indicates the Fully Connected Layer having 26 classes with a sofmax activation function.

For network training **Stochastic Gradient Descent** has been applied. Stochastic gradient descent (often shortened as SGD), is used for optimizing an objective function which is differentiable iteratively. The stochastic nature of SGD arises because of the random selection of the samples. The learning rate has been fixed to 0.01.

While training the model, the other parameters have been kept as follows:

1. **Target Image Size**= 64*64
2. **Batch Size**=32
3. **Number of epochs**=25
4. **Steps per epochs**= 8000

Finally the accuracy achieved is **97.9%.**

The plots for accuracy and loss are shown below:

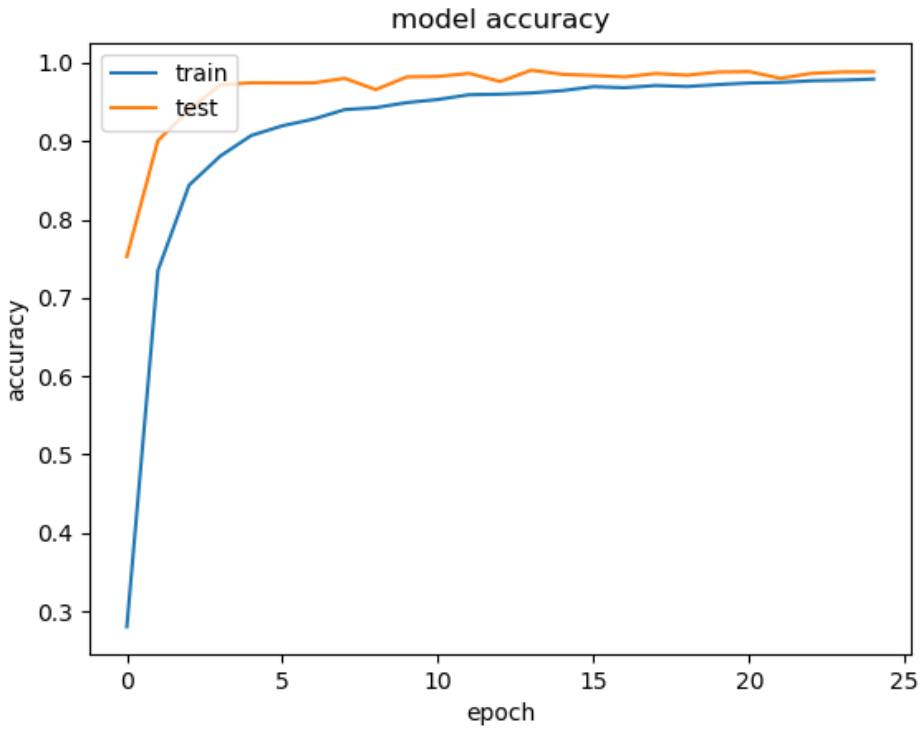

Figure 5.2: Plot for Accuracy

5.3.3 Speech Implementation

Once the gesture has been recognized, it is stored in a file from which the speech module reads and converts into speech. Microsoft Text to speech(TTS) has been used for generating speech, GoogleTrans for translating to other languages and finally PlaySound for playing the audio.

Microsoft TTS Azure Speech Services are the combination of the speech-to-text, text-to-speech, and speech translation into a single Azure cognitive service subscription. It becomes very easy to your applications to have speech services, tools, and devices with the Speech SDK, Speech Devices SDK, or REST APIs.

Text-to-speech from Azure Speech Services is a service that enables your applications, tools, or devices to convert text into natural human-like synthesized speech. It chooses from standard and neural voices,

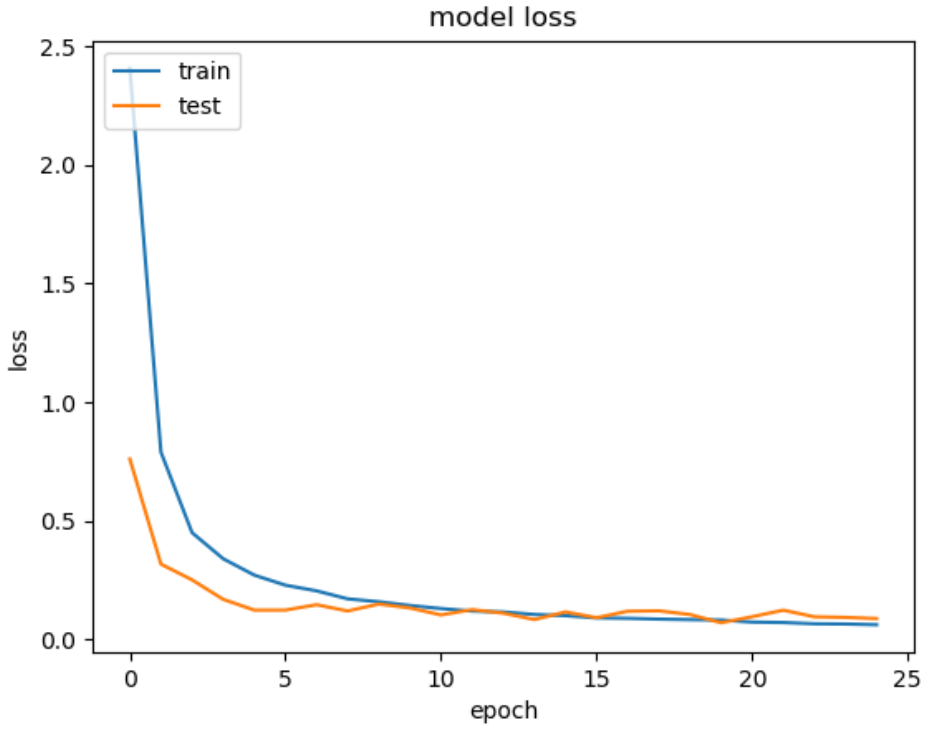

Figure 5.3: Plot for Model Loss

or you can create your own custom voice unique to the product or brand which you are using. It has the more than the 75 voices are available which supports more than 45 languages and locales, and five neural voices are available for 4 languages. For a full list, see supported languages.

Text-to-speech technology has been used to allow content creators to interact with their users in different ways. It provides users with an option to interact with content audibly which improves accessibility. It is used to help the user who a visual impairment, a learning disability, or helps in navigation while they are driving the vehicle, It can improve an existing experience. Text-to-speech feature adds value for voice bots and virtual assistants applications.

Googletrans

Googletrans is a free and unlimited python library that implemented Google Translate API. This uses the Google Translate Ajax API to make calls to such methods as detect and translate.

The features of Googletrans are:

1. Fast and reliable
2. Automatic detection of languages

3. Translation of languages in bulk
4. URL service is customizable
5. HTTP/2 support

Playsound

The playsound module contains only one thing - the function (also named) playsound. It requires one argument - the path to the file with the sound you'd like to play. This may be a local file, or a URL. There's an optional second argument, block, which is set to True by default. Setting it to False makes the function run asynchronously. On Windows, uses windll.winmm. WAVE and MP3 have been tested and are known to work. Other file formats may work as well. On OS X, uses AppKit.NSSound. WAVE and MP3 have been tested and are known to work. In general, anything QuickTime can play, playsound should be able to play, for OS X. On Linux, uses GStreamer. Known to work on Ubuntu 14.04 and ElementaryOS Loki. Support for the block argument is currently not implemented.

5.4 Graphical User Interface Design

For designing the GUI and deploying all the python code on a webpage the frame work used is called the Django. It is the one of the Python-based free, open-source, only web framework and it is the model-view template (MVT) architectural pattern.

The Important and the primary goal of django is create complex, database-driven websites easily. The features of reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself are emphasized in the frame work. The programming language it uses only python for everything such as settings files and data models. It also has an optional administrative interfaces to create, read, update and delete, which is generated dynamically and configured via admin models.

The Model Layer: Django provides an abstraction layer (the models) for structuring and manipulating the data of your Web application. A model is the single, definitive source of information about your data. It contains the essential fields and behaviors of the data you're storing. Generally, each model maps to a single database table. The basics:

- Each model is a Python class that subclasses django.db.models.Model.
- Each attribute of the model represents a database field.

- With all of this, Django gives you an automatically-generated database-access API; see Making queries.

The View Layer: Django has the concept of "views" to encapsulate the logic responsible for processing a user's request and for returning the response.

The Template Layer: Being a web framework, Django needs a convenient way to generate HTML dynamically. The most common approach relies on templates. A template contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted.

A Django project can be configured with one or several template engines (or even zero if you don't use templates). Django ships built-in backends for its own template system, creatively called the Django template language (DTL), and for the popular alternative Jinja2. Backends for other template languages may be available from third-parties.

Django defines a standard API for loading and rendering templates regardless of the backend. Loading consists of finding the template for a given identifier and preprocessing it, usually compiling it to an in-memory representation. Rendering means interpolating the template with context data and returning the resulting string.

The Django template language is Django's own template system. Until Django 1.8 it was the only built-in option available. It's a good template library even though it's fairly opinionated and sports a few idiosyncrasies. If you don't have a pressing reason to choose another backend, you should use the DTL, especially if you're writing a pluggable application and you intend to distribute templates. Django's contrib apps that include templates, like django.contrib.admin, use the DTL.

GUI design: The buttons are created using the <button> tag that defines a clickable button. Inside a <button> element you can put content, like text or images. Our buttons are styled using CSS.

To upload the image we have 2 buttons. The first one is an input button (<input>) with type set as file. Once the file is selected we have another input tag to submit the file to the specified directory.

Once the file is submitted a recognise gesture button appears. It is hidden when there is no file uploaded. On clicking this button the static code to recognise the hand gesture is executed and the output is displayed on another html page. In this html page we also have a convert to speech button to convert the received output to speech.

The last button on the main html page is the dynamic button to run the code that dynamically recognises the hand gestures.

5.5 Summary

The implementation phase of any project development is the most important phase as it yields the final solution, which solves the problem at hand. This chapter deals with the steps involved in the creation of the project. It is defined with the assistance of code explanation for the ease of the reader. Implementation is the phase of the project where the plans created during the design are put into action. The project is brought from concept to reality in this phase.

Chapter 6

Testing

6.1 Software Testing

In order to provide stakeholders with information about the quality of the software product software testing is performed. An objective, self-sufficient view of the software is provided by software testing so that business organizations understand the risk that is involved while implementing a software.

In software testing the following properties measure whether the system under consideration:

- reacts to different types of inputs accurately
- Takes an admissible amount of time to perform all the functions
- is usable under all conditions
- is capable of running in different environments

6.2 Testing Process

The two major methods of testing are White Box Testing and Black Box Testing.

6.2.1 White Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing softwarethat tests internal structures or workings of an application, as opposed to its functionality (ie black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the expected outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process.

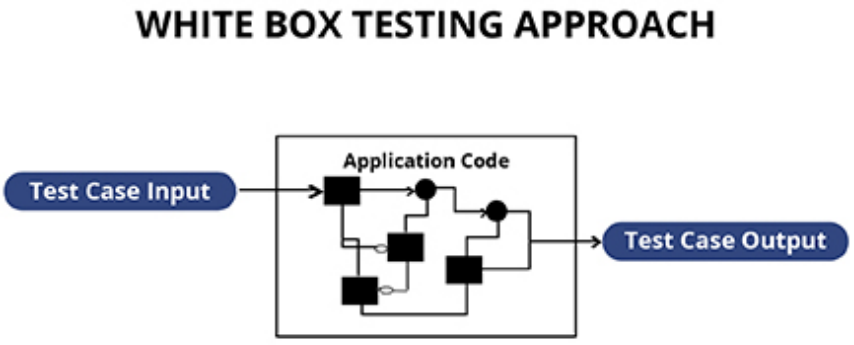

Figure 6.1: White Box Testing

6.2.2 Black Box Testing

Black box testing, also known as Behavioural Testing, is a software testing model in which the internal structure/design/implementation of the item being tested is not known to the tester (Fig 6.1). These tests can be functional or non-functional, though usually functional.

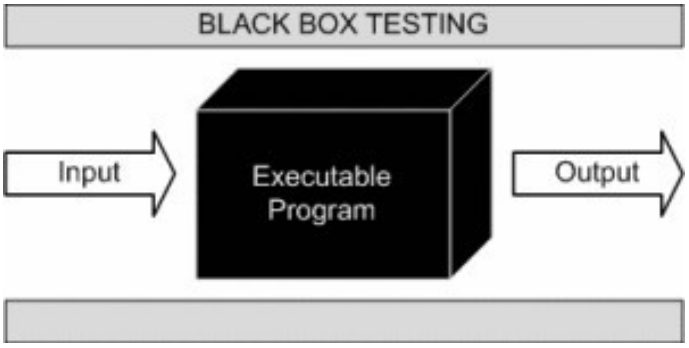

Figure 6.2: Black Box Testing

This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

- Erroneous functions
- Errors in the user interface
- Mistake in access to external database

- Various errors in performance

6.2.3 Acceptance Testing

Acceptance testing is the formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.

6.3 Levels of Testing

The testing process in this project involves the testing of all the modules. It covers the below mentioned levels of testing:

- Unit Testing
- Integration testing
- System Testing

Unit Testing: Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class.

Integration Testing: Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

System Testing: System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable

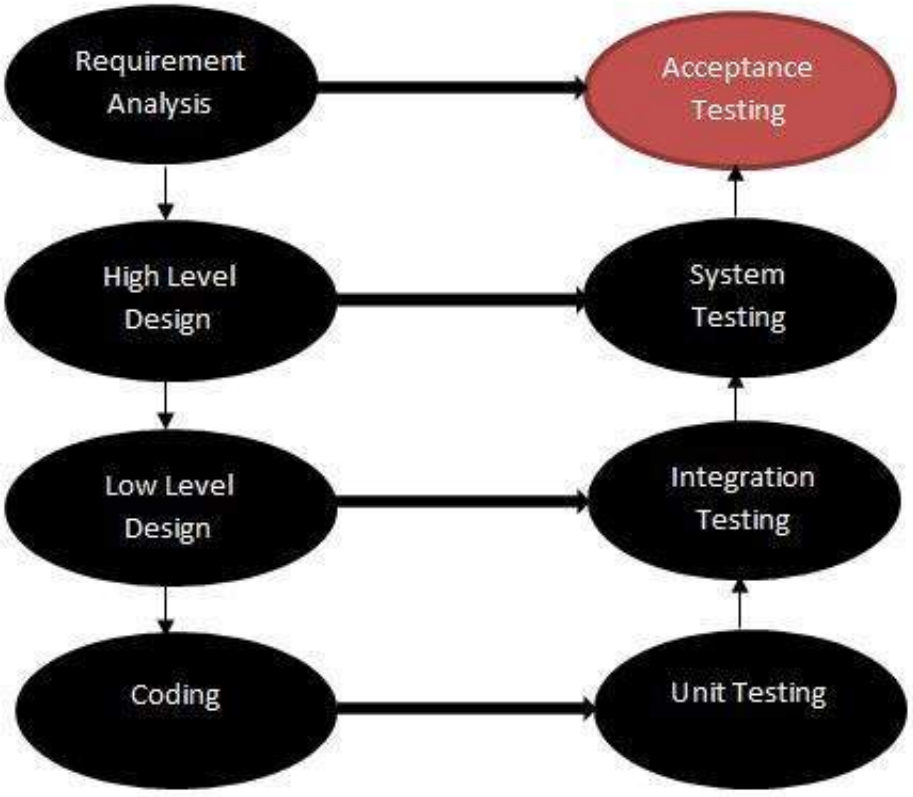

Figure 6.3: Levels of Testing

hardware system(s). System testing is a more limited type of testing; it seeks to detect defects both within the ”inter- assemblages” and also within the system as a whole.

6.4 Integration Testing of Modules

In order to present the failures which is caused by interface defects, integration testing of several software components is performed on one particular platform.

In this project, first the obtained dataset from the camera is integrated with the data preprocessing code. Once the data is preprocessed ,for extracting the features the Canny Edge Detection algorithm is being used. Finally the database is integrated with the learning model to predict the correct gesture. Subsequently, the gestures are stored in a file and goes into the speech synthesis module. All the previous steps are finally deployed on a webpage.

Integration Testing Results: Test cases developed have passed successfully and no errors found.

6.5 Unit Testing of Main Modules

Execution of the recognition system is tested for various conditions and the test cases are tabulated as follows:

Test Case ID	Test Case #1:Image Acquisition
Title	To test whether image is successfully captured by the system
Input	Image captured through webcam
Expected Output	Captured Image
Actual Output	Captured Image
Remarks	The system performed as expected.

Unit Test Case 1

Figure 6.4: Image Acquisition through Webcam

Test Case ID	Test Case #2
Title	Cropping of Image
Input	Selection of the Region of Interest from the Clicked Picture
Expected Output	Cropped Image of the Hand in RGB
Actual Output	Cropped Image of the Hand in RGB
Remarks	The system performed as expected.

Unit Test Case 2

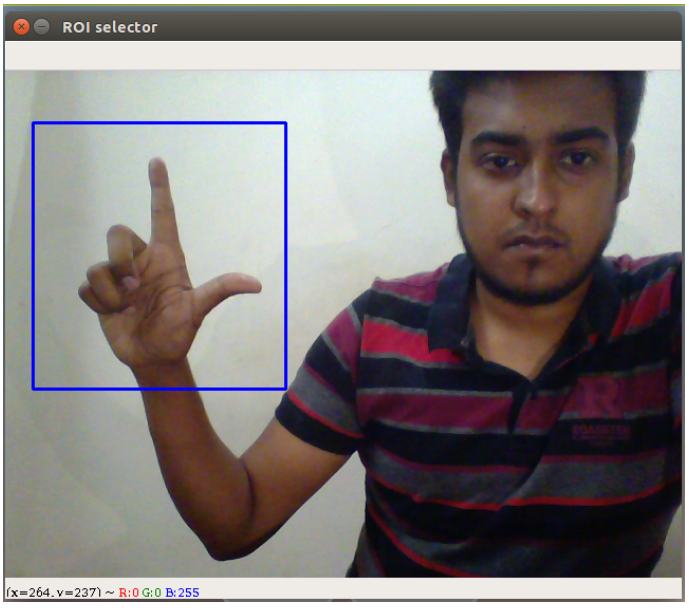

Figure 6.5: Selecting the ROI from the image

Figure 6.6: Cropped Portion of the Image

Test Case ID	Test Case #3
Title	Testing whether the system is correctly preprocessing the image
Input	Cropped Portion of the Image
Expected Output	Binary Image of the Cropped Gesture
Actual Output	Binary Image of the Cropped Gesture
Remarks	The system performed as expected.

Unit Test Case 3

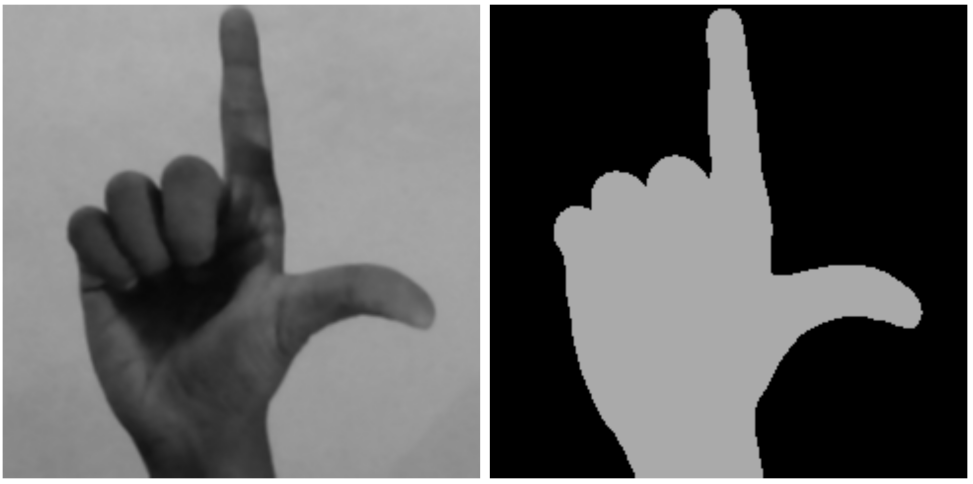

Figure 6.7: Binary Image after Preprocessing

Test Case ID	Test Case #4
Title	To test the system for feature extraction
Input	Binary Image
Expected Output	Image showing the gesture edges
Actual Output	Image showing the gesture edges
Remarks	The system performed as expected.

Unit Test Case 4

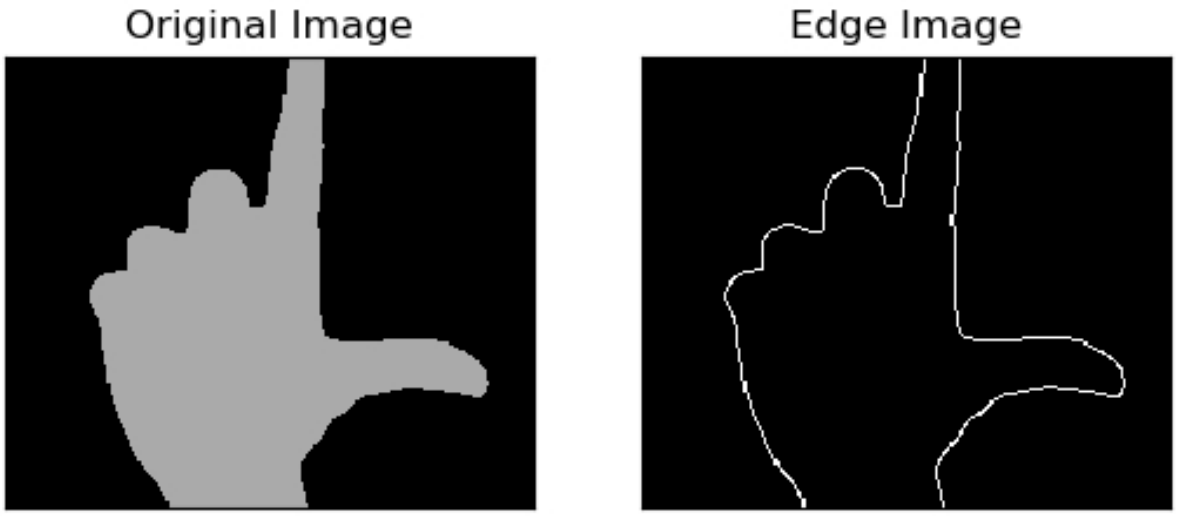

Figure 6.8: Final Image after Feature Extraction

Test Case ID	Test Case #5
Title	To test training of the Convolutional Neural Network
Input	Training and test set of images from A-Z
Expected Output	Trained model with sufficiently high accuracy
Actual Output	Trained model with 97.91% accuracy
Remarks	The system performed as expected.

Unit Test Case 5

```
C:\Users\Saptarshi\Anaconda3\lib\site-packages\h5py\__init__.py:34: FutureWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` is depr
ecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
cnn_model.py:10: UserWarning: Update your `Conv2D` call to the Keras 2 API: `Conv2D(32, (3, 3), input_shape=(64, 64, 3..., activation="relu")`
  classifier.add(Convolution2D(32, 3,  3, input_shape = (64, 64, 3), activation = 'relu'))
cnn_model.py:14: UserWarning: Update your `Conv2D` call to the Keras 2 API: `Conv2D(32, (3, 3), activation="relu")`
  classifier.add(Convolution2D(32, 3,  3, activation = 'relu'))
cnn_model.py:17: UserWarning: Update your `Conv2D` call to the Keras 2 API: `Conv2D(64, (3, 3), activation="relu")`
  classifier.add(Convolution2D(64, 3,  3, activation = 'relu'))
Found 45500 images belonging to 26 classes.
Found 6500 images belonging to 26 classes.
Epoch 1/25
800/800 [==============================] - 1190s 1s/step - loss: 2.4068 - acc: 0.2803 - val_loss: 0.7603 - val_acc: 0.7527
Epoch 2/25
800/800 [==============================] - 953s 1s/step - loss: 0.7893 - acc: 0.7351 - val_loss: 0.3177 - val_acc: 0.9008
Epoch 3/25
800/800 [==============================] - 873s 1s/step - loss: 0.4501 - acc: 0.8438 - val_loss: 0.2512 - val_acc: 0.9397
Epoch 4/25
800/800 [==============================] - 875s 1s/step - loss: 0.3403 - acc: 0.8808 - val_loss: 0.1692 - val_acc: 0.9720
Epoch 5/25
800/800 [==============================] - 1014s 1s/step - loss: 0.2713 - acc: 0.9072 - val_loss: 0.1238 - val_acc: 0.9745
Epoch 6/25
800/800 [==============================] - 1072s 1s/step - loss: 0.2288 - acc: 0.9196 - val_loss: 0.1239 - val_acc: 0.9743
Epoch 7/25
800/800 [==============================] - 1059s 1s/step - loss: 0.2049 - acc: 0.9281 - val_loss: 0.1466 - val_acc: 0.9743
Epoch 8/25
800/800 [==============================] - 2150s 3s/step - loss: 0.1713 - acc: 0.9402 - val_loss: 0.1202 - val_acc: 0.9802
Epoch 9/25
800/800 [==============================] - 879s 1s/step - loss: 0.1589 - acc: 0.9428 - val_loss: 0.1493 - val_acc: 0.9658
Epoch 10/25
800/800 [==============================] - 851s 1s/step - loss: 0.1433 - acc: 0.9492 - val_loss: 0.1329 - val_acc: 0.9820
Epoch 11/25
800/800 [==============================] - 841s 1s/step - loss: 0.1303 - acc: 0.9532 - val_loss: 0.1035 - val_acc: 0.9825
Epoch 12/25
800/800 [==============================] - 839s 1s/step - loss: 0.1198 - acc: 0.9592 - val_loss: 0.1258 - val_acc: 0.9864
Epoch 13/25
800/800 [==============================] - 857s 1s/step - loss: 0.1159 - acc: 0.9599 - val_loss: 0.1109 - val_acc: 0.9761
Epoch 14/25
```

Figure 6.9: CNN Model Training

```
800/800 [==============================] - 1059s 1s/step - loss: 0.2049 - acc: 0.9281 - val_loss: 0.1466 - val_acc: 0.9743
Epoch 8/25
800/800 [==============================] - 2150s 3s/step - loss: 0.1713 - acc: 0.9402 - val_loss: 0.1202 - val_acc: 0.9802
Epoch 9/25
800/800 [==============================] - 879s 1s/step - loss: 0.1589 - acc: 0.9428 - val_loss: 0.1493 - val_acc: 0.9658
Epoch 10/25
800/800 [==============================] - 851s 1s/step - loss: 0.1433 - acc: 0.9492 - val_loss: 0.1329 - val_acc: 0.9820
Epoch 11/25
800/800 [==============================] - 841s 1s/step - loss: 0.1303 - acc: 0.9532 - val_loss: 0.1035 - val_acc: 0.9825
Epoch 12/25
800/800 [==============================] - 839s 1s/step - loss: 0.1198 - acc: 0.9592 - val_loss: 0.1258 - val_acc: 0.9864
Epoch 13/25
800/800 [==============================] - 857s 1s/step - loss: 0.1159 - acc: 0.9599 - val_loss: 0.1109 - val_acc: 0.9761
Epoch 14/25
800/800 [==============================] - 866s 1s/step - loss: 0.1054 - acc: 0.9615 - val_loss: 0.0840 - val_acc: 0.9905
Epoch 15/25
800/800 [==============================] - 870s 1s/step - loss: 0.1004 - acc: 0.9644 - val_loss: 0.1160 - val_acc: 0.9852
Epoch 16/25
800/800 [==============================] - 869s 1s/step - loss: 0.0910 - acc: 0.9696 - val_loss: 0.0918 - val_acc: 0.9838
Epoch 17/25
800/800 [==============================] - 875s 1s/step - loss: 0.0892 - acc: 0.9682 - val_loss: 0.1190 - val_acc: 0.9819
Epoch 18/25
800/800 [==============================] - 894s 1s/step - loss: 0.0858 - acc: 0.9710 - val_loss: 0.1208 - val_acc: 0.9863
Epoch 19/25
800/800 [==============================] - 863s 1s/step - loss: 0.0836 - acc: 0.9697 - val_loss: 0.1052 - val_acc: 0.9841
Epoch 20/25
800/800 [==============================] - 905s 1s/step - loss: 0.0820 - acc: 0.9722 - val_loss: 0.0706 - val_acc: 0.9882
Epoch 21/25
800/800 [==============================] - 950s 1s/step - loss: 0.0733 - acc: 0.9742 - val_loss: 0.0960 - val_acc: 0.9887
Epoch 22/25
800/800 [==============================] - 899s 1s/step - loss: 0.0715 - acc: 0.9750 - val_loss: 0.1236 - val_acc: 0.9802
Epoch 23/25
800/800 [==============================] - 870s 1s/step - loss: 0.0662 - acc: 0.9770 - val_loss: 0.0956 - val_acc: 0.9865
Epoch 24/25
800/800 [==============================] - 859s 1s/step - loss: 0.0652 - acc: 0.9778 - val_loss: 0.0931 - val_acc: 0.9883
Epoch 25/25
800/800 [==============================] - 861s 1s/step - loss: 0.0623 - acc: 0.9791 - val_loss: 0.0882 - val_acc: 0.9885
dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
```

Figure 6.10: CNN Model Training-97.91% accuracy achieved

Test Case ID	Test Case #6
Title	To test Dynamic Hand Gesture Recognition
Input	Make various American Sign Language Gestures through the webcam
Expected Output	Correct Gesture Prediction
Actual Output	Correct Gesture Prediction
Remarks	The system performed as expected.

Unit Test Case 6

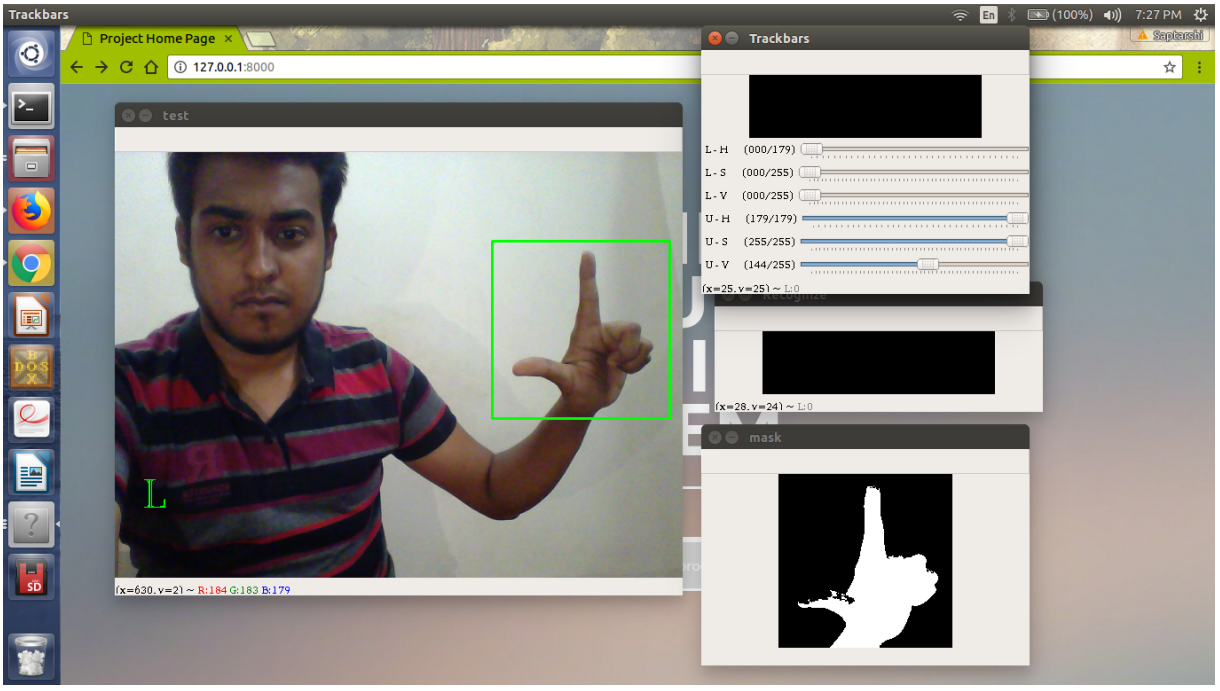

Figure 6.11: Dynamic Gesture Recognition for Letter L

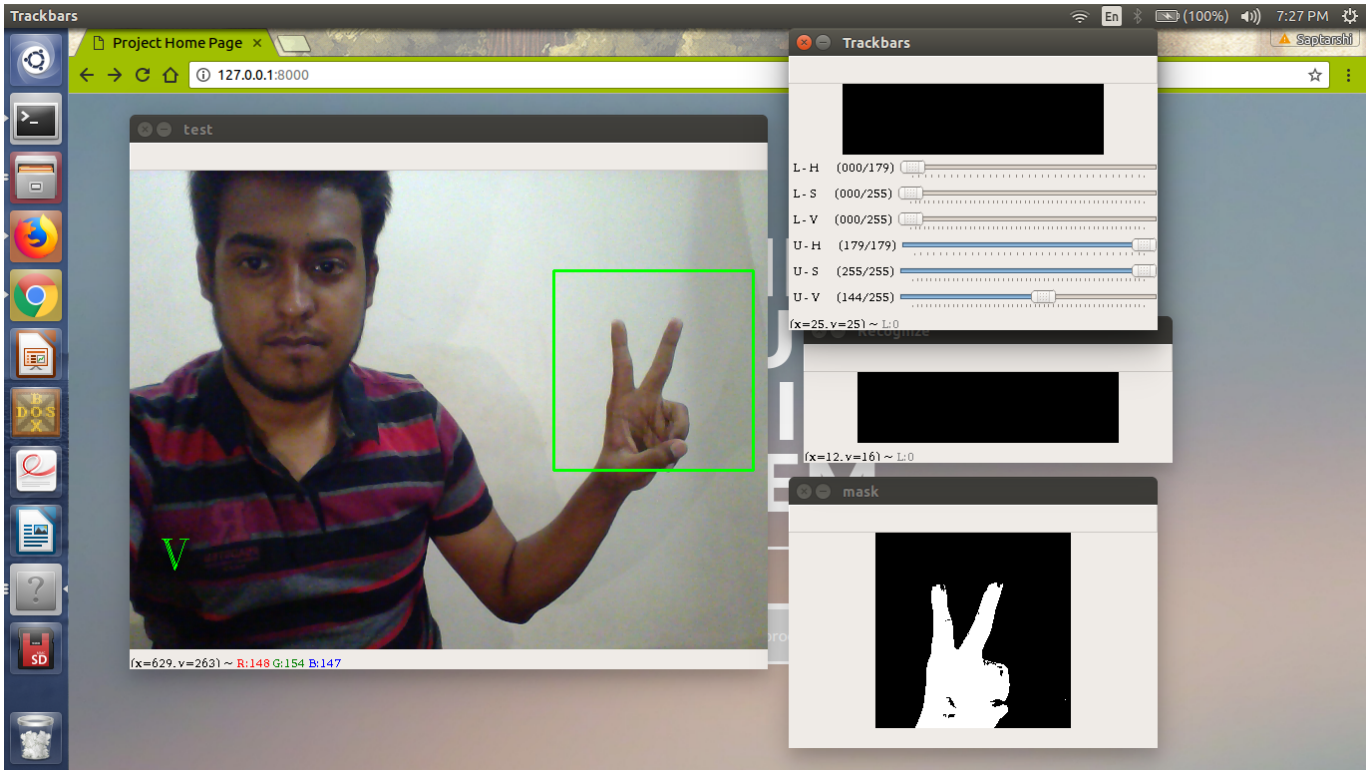

Figure 6.12: Dynamic Gesture Recognition for Letter V

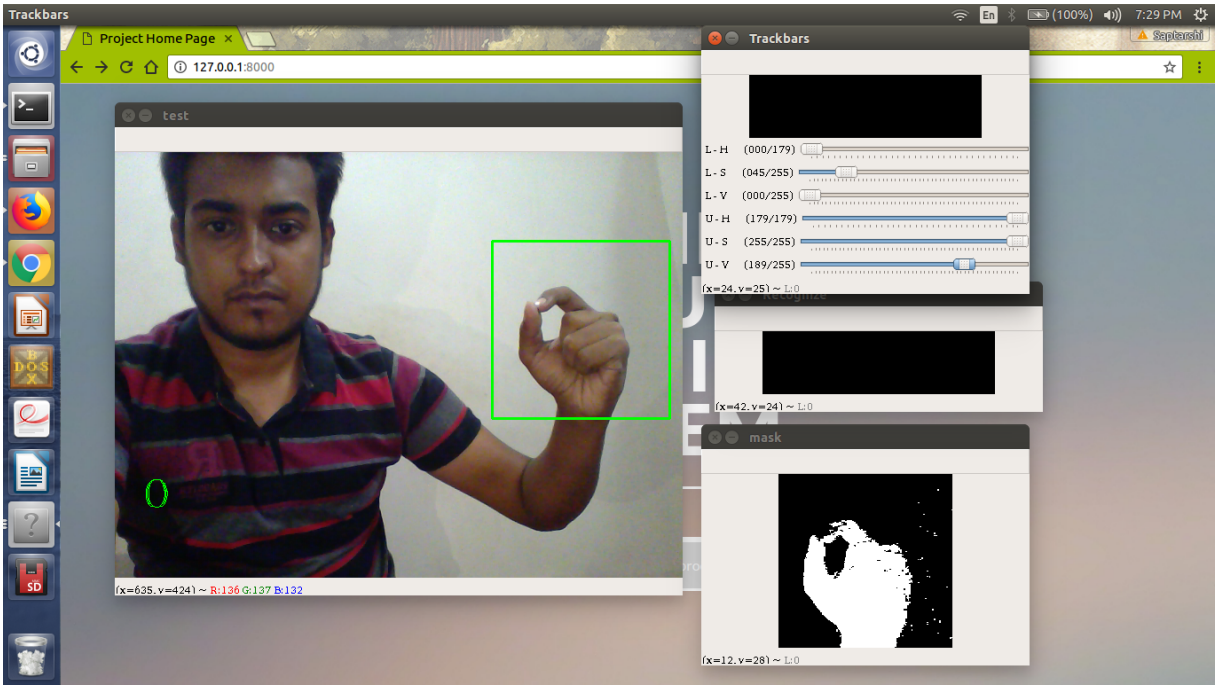

Figure 6.13: Dynamic Gesture Recognition for Letter O

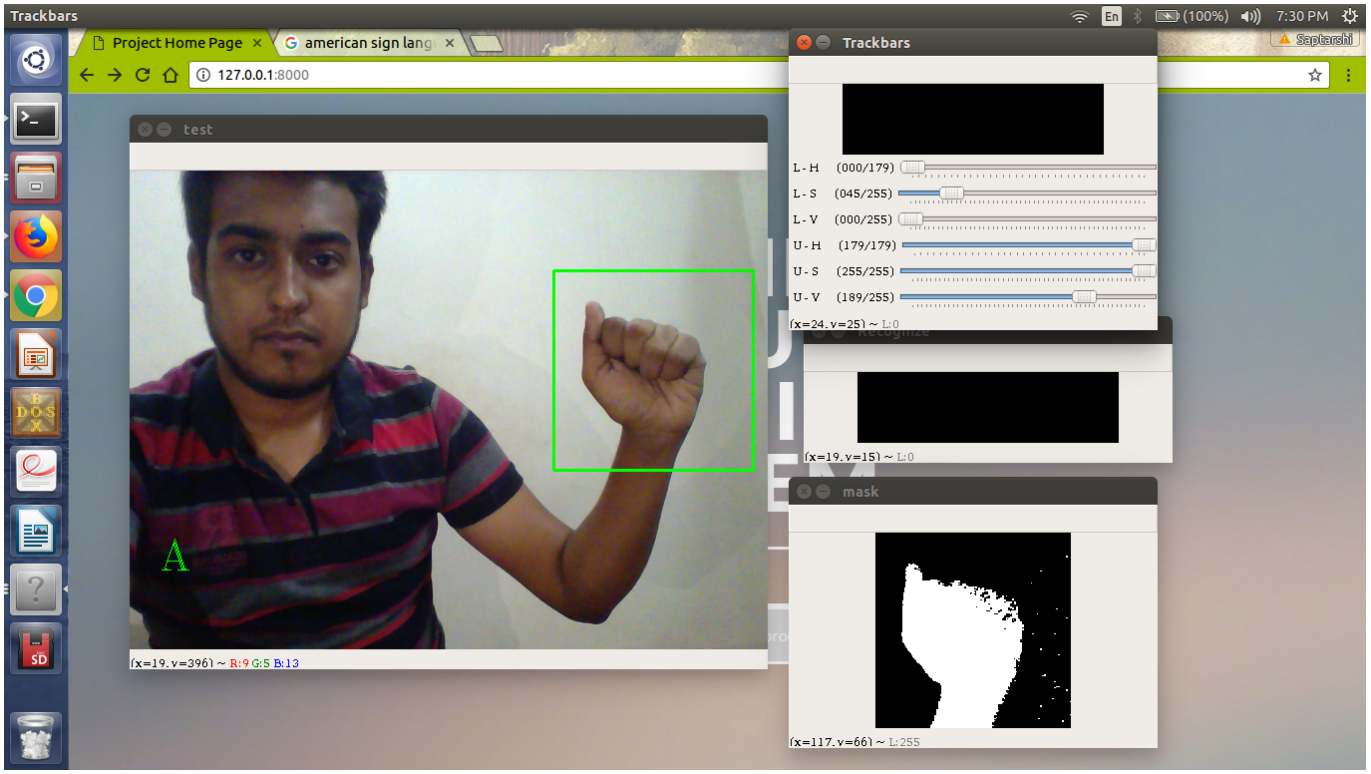

Figure 6.14: Dynamic Gesture Recognition for Letter A

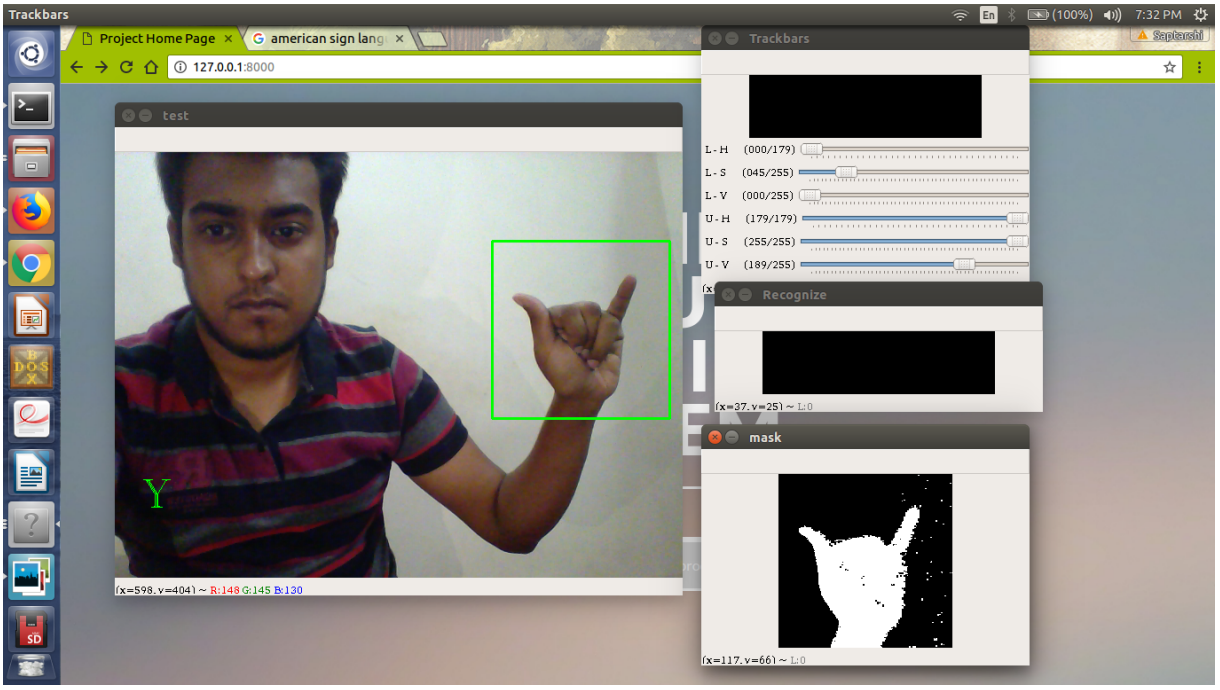

Figure 6.15: Dynamic Gesture Recognition for Letter Y

Test Case ID	Test Case #7
Title	To test Static Hand Gesture Recognition
Input	Upload Static Hand Gesture
Expected Output	Correct Gesture Prediction
Actual Output	Correct Gesture Prediction
Remarks	The system performed as expected.

Unit Test Case 7

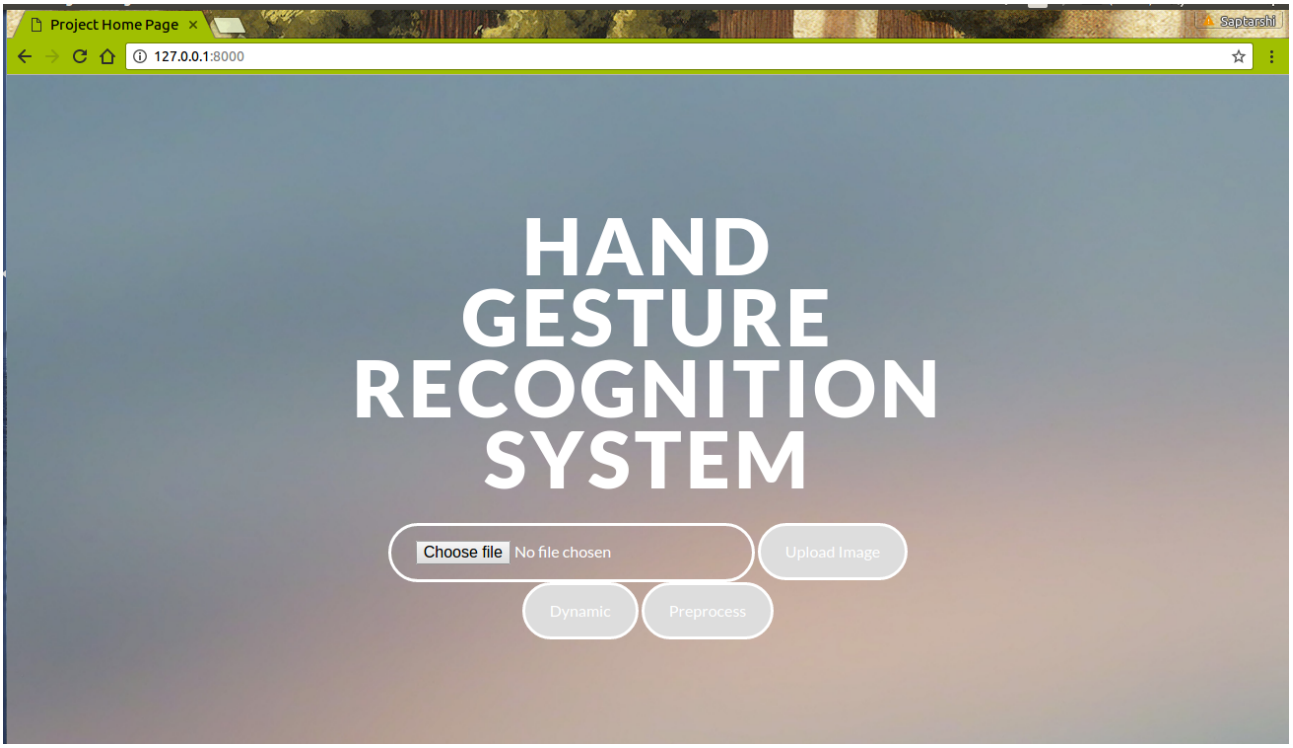

Figure 6.16: UI connecting all the modules

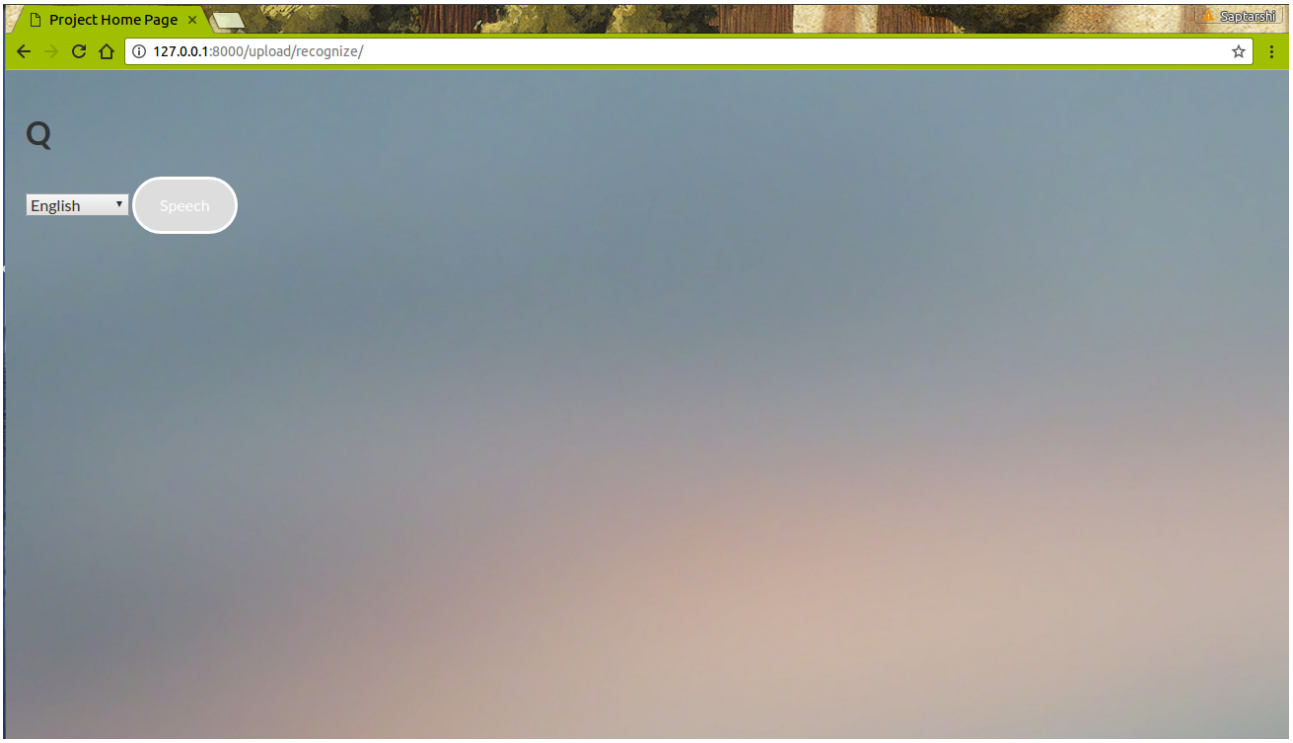

Figure 6.17: Static Hand Gesture Recognition for Letter Q

Test Case ID	Test Case #8
Title	To test conversion of speech in different language
Input	Predicted Gesture
Expected Output	Speech form of the Gesture in the user specified language
Actual Output	Speech form of the Gesture in the user specified language
Remarks	The system performed as expected.

Unit Test Case 8

6.6 Summary

We can easily demonstrate the quality of a particular product using software testing. In order to test one module at a time, unit testing is used .Integration testing aggregates all modules using Integration testing methods.System testing tests the system to check all essentials of the system are fulfilled.

In this section of the document, in order to ensure the validity of the model, software testing has been done .They are unit testing,integration testing and system testing.We then move on to tabulate 3 of the major unit tests carried out.

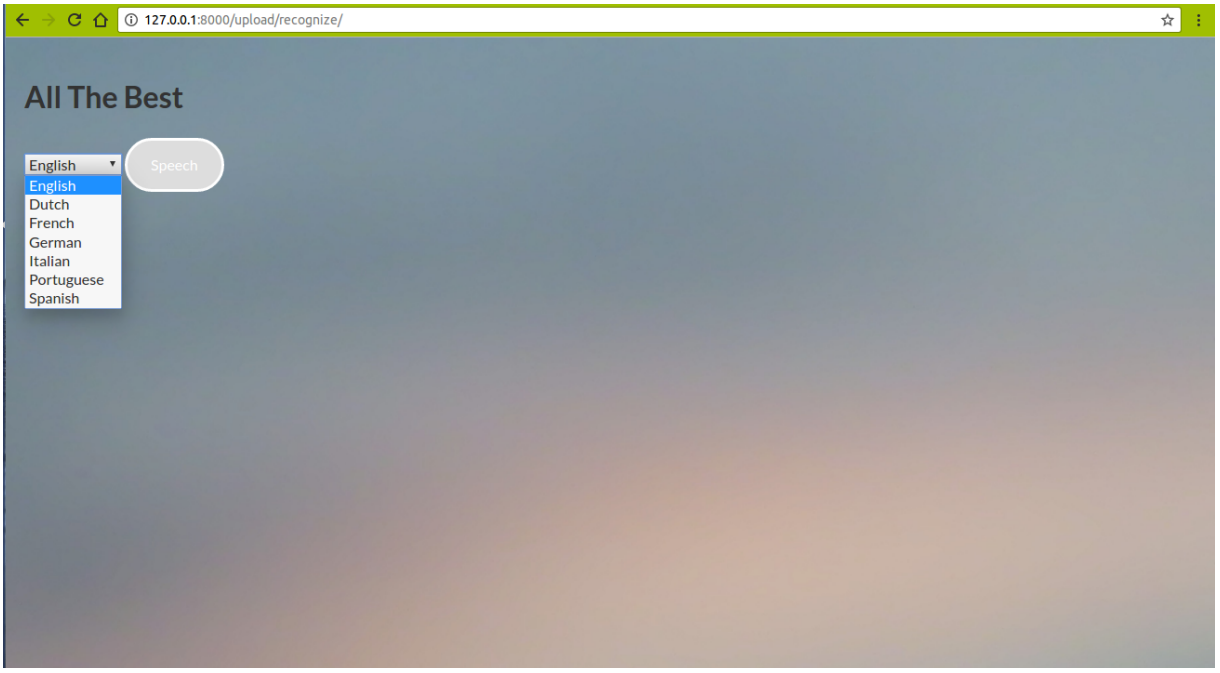

Figure 6.18: Conversion to speech in user specified language

Chapter 7

Conclusion

People who cant speak rely on hand gestures for their day to day communication, but interpreters are costly because of their maintenance and quite difficult to find and schedule the quantified interpreters, so they cannot rely on them. This system will help disabled persons in improving their quality of life significantly. The automatic recognition of sign language is an attractive prospect; the technology exists to make it possible, while the potential applications are exciting and worthwhile. The system is proved robust against changes in gesture.

This project will help in improving the quality of life of the differently abled and it provides an example of how technology can be used for good. It recognises hand gestures even if there is movement so it is robust as well.

By the conversion of gesture into speech it become very much easy for us to explain our words to hearing and speech disabled persons also they can express their words to us in more efficient way by this the way of communication between hearing and speech disabled person and normal person.

7.1 Limitations of The Project

- Presence of too much noise can cause hindrance to the recognition system.
- Text to speech can only be produced and not the other way round.
- Delay in the processing execution can be occurred due to the large amount of high resolution images
- Lighting conditions play a huge role in detecting correct gestures.

7.2 Future Enhancements

This project can be further enhanced by formation of words from the letters and conversion to speech simultaneously .During the next few years, gesture recognition will be used along with other applications so that we can make main stream applications work with these features to make them more effective for all kinds of people. Companies can use an advanced version of this which can be deployed on mobiles and tablets to increase the mobility rather than using it for simple applications menu bars and clicking.

Bibliography

[1] Ramesh M Kagalkar, Nagaraj H.N *,New Methodology for Translation of Static Sign Symbol to Words in Kannada Language* . International Journal of Computer Applications (0975 – 8887) Volume 121 – No.20, July 2015

[2] Sawant Pramada,, Deshpande Saylee , Nale Pranita, Nerkar Samiksha, Archana S. Vaidya *,Intelligent Sign Language Recognition Using Image Processing* . IOSR Journal of Engineering (IOSRJEN) e-ISSN: 2250-3021, p-ISSN: 2278-8719 Vol. 3, Issue 2 (Feb. 2013), ||V2|| PP 45-51

[3] V.Padmanabhan, M.Sornalatha *, Hand gesture recognition and voice conversion system for dumb people,* . International Journal of Scientific Engineering Research, Volume 5, Issue 5, May-2014 427 ISSN 2229-5518

[4] Anjali Singh,Balram Timande *, Gesture to Speech Conversion in "HINDI" language for Hearing Speech disabled persons in INDIA* . International Research Journal of Engineering and Technology (IRJET) Volume: 03 Issue: 06 | June-2016

[5] Tahir Khan *, Hand Gesture Recognition based on Digital Image Processing using MATLAB* . International Journal of Scientific Engineering Research, Volume 6, Issue 9, September 2015 338 ISSN 2229-5518

[6] Ashis Pradhana, M.K. Ghosea, Mohan Pradhana *, A Hand Gesture Recognition using Feature Extraction*. International Journal of Computer Applications (0975 – 8887) Volume 121 – No.20, July 2015

TEAM INFORMATION

NAME: SAPTARSHi DUTTA GUPTA USN: 1PE15CS139 CONTACT NUMBER: 9972846606 EMAIL ID: saptarshiduttagupta@gmail.com ADDRESS: 4, DR T.N. Majumdar Street, Kolkata-700026	NAME: SOMISETTY V R DINESH USN: 1PE15CS159 CONTACT NUMBER: 9108569481 EMAIL ID: somisettydinesh459@gmail.com ADDRESS: Flatno:102 A block,Sapthagiri enclave,pappula veedhi,Nellore,Andhra Pradesh-524002

NAME: NATASHA P. USN: 1PE15CS095 CONTACT NUMBER: 9483467644 EMAIL ID: natasha.pashu@gmail.com ADDRESS: Q001 Purva Fairmont 2nd sector HSR layout Bangalore 560102	NAME: JYOTI M ANGADI USN: 1PE16CS410 CONTACT NUMBER: 8970031164 EMAIL ID: jyotiangadi9@gmail.com ADDRESS: Flat no. C1, 3rd floor, #60, JAPS apartment 1st main, 3rd cross, naganathapura, Electronic city, Bangalore 560100