

**A Project Report**  
**on**  
**Indian Sign Language Interpretation using Camera**

Submitted to the  
Savitribai Phule Pune University, Pune  
In partial fulfilment for the award of the Degree of  
Bachelor of Engineering  
in  
Information Technology  
by

**Minal Kapadnis (B120238568)**  
**Kartik Poshattiwar (B120238569)**  
**Samruddhi Naik (B120238592)**

Under the guidance of

**Dr N J Uke**



DEPARTMENT OF INFORMATION TECHNOLOGY ENGINEERING  
SINHGAD COLLEGE OF ENGINEERING S. NO. 44/1, VADGAON (BK.), OFF. SINHGAD  
ROAD, PUNE, MAHARASHTRA, INDIA (2015-2016)



**Sinhgad Institutes**

## **CERTIFICATE**

This is to certify that the project report entitled “**Indian Sign Language Interpretation using Camera**” being submitted by **Minal Kapadnis, Kartik Poshattiwar and Samruddhi Naik** is a record of bonafide work carried out by them under the supervision and guidance of **Dr N J Uke** in partial fulfilment of the requirement for **BE (Information Technology Engineering)** course of Savitribai Phule Pune University, Pune in the academic year 2015-2016.

**Date:**

**Place:**

Dr. N. J. Uke  
Guide

Dr. S. D. Lokhande  
Principal

Dr. N J Uke  
Head of the Department

## ACKNOWLEDGEMENT

It is our privilege to acknowledge with deep sense of gratitude to our project guide and Head of Department (Information Technology), **Dr. N J Uke** for his valuable suggestions and guidance throughout our course of study and timely help given to us for our project on Indian Sign Language Interpretation using Camera.

This acknowledgement would be incomplete without expressing our special thanks to the Head of Department (HOD).

We would also like to thank our parents, group members and friends for providing suggestions, help and moral support.

## **Abstract**

Hand gestures are a strong medium of communication for hearing impaired society. It is helpful for establishing interaction between human and computer. In this project we proposed a continuous Indian Sign Language (ISL) gesture recognition system where single or both the hands are used for performing any gesture. Recognizing a sign language gestures from continuous gestures is a very challenging research issue.

We intend to solve this problem using gradient based key frame extraction method. These key frames are helpful for splitting continuous sign language gestures into sequence of signs as well as for removing uninformative frames. After splitting of gestures each sign has been treated as an isolated gesture. Then features of pre-processed gestures are extracted using Orientation Histogram (OH) with Principal Component Analysis (PCA) which is applied for reducing dimension of features obtained after OH. Analysis of gestures is performed on our own continuous Indian Sign Language (ISL) dataset which is created using a camera by means of Image Processing and various Algorithms like blob detection, template matching etc.

Applications are in industrial, medical, educational and various related fields which have vast scope in the future.

## LIST OF FIGURES

Sr. No.	Description	Page No.
1	Hands and Face localization	1
2	Gesture Recognition System	6
3	ISL video gestures to different light illumination conditions	7
4	Use Case Diagram	8
5	Class Diagram	8
6	Sequence Diagram 1	9
7	Sequence Diagram 2	9
8	Activity Diagram	10
9	State Diagram	10
10	Component Diagram	11
11	Deployment Diagram 1	11
12	Deployment Diagram 2	12
13	System Architecture Diagram	12
14	GUI of the system	16
15	Training the system	17
16	Gesture Recognition	17
17	Gesture 'A' (ideal background)	18
18	Gesture 'A' (complex background)	18
19	Gesture 'C' (ideal background)	18
20	Gesture 'C' (complex background)	18

### III

## LIST OF TABLES

Table No	Description	Page No.
1	Literature Survey	4
2	Test Cases (Testing GUI)	19
3	Test Cases (Creating Dataset)	19-20
4	Test Cases (Recognising Gestures)	21

### IV

## **CONTENTS**

CERTIFICATE	I
ACKNOWLEDGEMENT	II
LIST OF FIGURES	III
LIST OF TABLES	IV

<b>CHAPTER</b>	<b>TITLE</b>	<b>PAGE NO.</b>
<b>1.</b>	<b>INTRODUCTION TO GESTURE RECOGNITION</b>	
1.1	Introduction to Indian Sign Language Using Camera	1
1.2	Problem Statement	2
1.3	Motivation	2
1.4	Objectives of Work	2
<b>2.</b>	<b>LITERATURE SURVEY</b>	<b>3</b>
<b>3.</b>	<b>PROJECT REQUIREMENTS</b>	
3.1	Analysis of a gesture based system	5
3.2	Product Function Modules	5
3.3	Tools for Gesture Recognition	6
<b>4.</b>	<b>DESIGN FOR A GESTURE RECOGNITION SYSTEM</b>	<b>8</b>
<b>5.</b>	<b>IMPLEMENTATION OF GESTURE RECOGNITION SYSTEM</b>	<b>13</b>
<b>6.</b>	<b>RESULT AND EVALUATION</b>	<b>16</b>
<b>7.</b>	<b>CONCLUSION, LIMITATIONS AND FUTURE ENHANCEMENTS</b>	<b>22</b>
<b>8.</b>	<b>REFERENCES</b>	<b>23</b>





## CHAPTER 1

### INTRODUCTION TO GESTURE RECOGNITION

#### 1.1 Introduction to Indian Sign Language Interpretation Using Camera.

Our Project presents the early findings of an exploration in to the suitability of a range sensing camera for recognising Indian Sign Language. A camera is an optical instrument for recording images, which may be stored locally, transmitted to another location, or both. The images may be individual still photographs or sequences of images constituting videos or movies. The functioning of the camera is very similar to the functioning of the human eye.

The camera is a device which is easily available and has an adequate level of accuracy in producing a system which could be used to recognise Indian signs. This functionality could then be incorporated into a system to help young deaf and hard of hearing children to learn Indian signs. The system would be able to demonstrate specific signs using videos and images, and provide feedback to the child on its own about their Indian Sign accuracy through the application developed. This project is aimed specifically for Indian Sign Language and the principles will be relevant to any sign based communication system.

Gesture recognition is concerned with identifying human gestures using technology. This is an established research area with a broad background and many gesture recognition systems have been developed. The Seek and Sign research project is specifically interested in gesture recognition technologies that may be suitable for recognising sign language. Research has been conducted in this area, with the most promising technology to date being glove technology, wrist sensors, 2D and 3D cameras, and the Kinect platform.

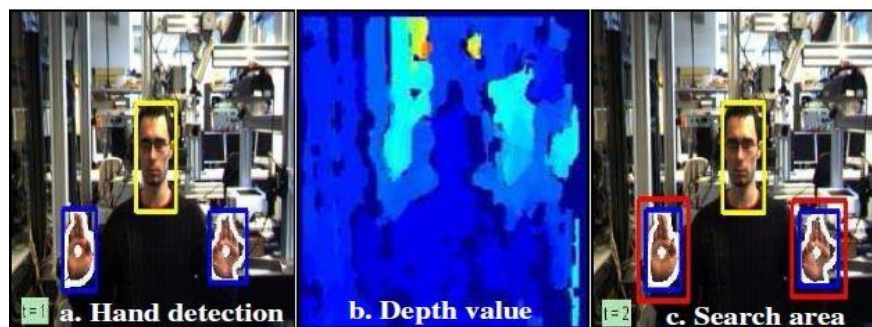


Fig.1 Hands and Face localization

## 1.2 Problem Statement

To develop an Android Application and desktop language for continuous Indian Sign Language (ISL) gesture recognition system where both the hands are used for performing any gesture.

## 1.3 Motivation

Statistics about deafness in India:

1.2 Million People with severe hearing disabilities 0.9

Million people with moderate hearing disability 7.1

Million People with mild hearing disability.

9 out of 10 deaf children born are born to parents who can hear. Hearing impairment in children leads to deficiency in language development and ultimately speech and attention difficulties. Absence of any existing computer technology in recognising Indian sign languages. The Government's negligence towards the educational empowerment of deaf and dumb people. Our Keen interest and curiosity about new and innovative technological platforms. An urge to put our engineering skills to use for the benefit of society

## 1.4 Objectives of Work.

- 1) To suggest and build a new gesture based system by using a camera.
- 2) To develop a Sign language interpretation software using Image Processing and Artificial Intelligence.
- 3) To provide real-time translation of sign language through computer processing.
- 4) To suggest and build a simple application for interpreting input through camera that is more suited to Devnagri Sign language than the pre-existing methods.
- 5) To contemplate the use of an innovative input method for computers that is different from the conventional input devices such as keyboards, mice and touch screens.

## **CHAPTER 2**

### **LITERATURE SURVEY**

Gesture recognition pertains to recognizing meaningful expressions of motion by a human, involving the hands, arms, face, head, and/or body. It is of utmost importance in designing an intelligent and efficient human–computer interface. The applications of gesture recognition are manifold, ranging from sign language through medical rehabilitation to virtual reality. In this paper, we provide a survey on gesture recognition with particular emphasis on hand gestures and facial expressions. Applications involving hidden Markov models, particle filtering and condensation, finite state machines, optical flow, skin colour, and connectionist models are discussed in detail. Existing challenges and future research possibilities are also highlighted.

A gesture recognition system must have dynamic features having orientation from spatiotemporal trajectories which could further be quantized to generate code words. The continuous gestures have a zero code word detection system with static velocity motion. The potential models which could be used to generate similar applications include hidden Markov model, neural network model and fuzzy system.

The major problems usually arise for segmentation which usually creates problems in understanding start and end of gestures. Ergodic models are also created which have a dynamic system with an invariant measure. Gaussian Mixture Model is used for skin colour detection. The orientation between 2 points is considered as the basic feature.

Index Terms—Face recognition, facial expressions, hand gestures, hidden Markov models (HMMs), soft computing, optical flow

<b>Sr. No.</b>	<b>Reference paper</b>	<b>Work done</b>	<b>Problems found</b>	<b>Year of publication</b>
1.	A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation	Real-time system for continuous digit recognition and to enable users to find gestures of interest within a large video sequence.	More improvements possible in accuracy and efficiency through cascaded approach.	September 2009
2.	A Hidden Markov Model-based continuous gesture recognition system for hand motion trajectory	An automatic system that recognizes both isolated and continuous gestures for Arabic numbers using hidden Markov model.	Left Right and Ergodic topologies of hidden Markov models are not preferable.	January 2009
3.	Recognizing & Interpreting Indian Sign Language Gesture for Human Robot Interaction	Approach for classification of ISL gesture for interaction between humanoid robot HOAP-2 and human being.	NA	September 2010

**Table 1 Literature Survey**

## **CHAPTER 3**

### **PROJECT REQUIREMENTS**

#### **3.1 Analysis of a gesture based system**

To develop an Android Application for continuous Indian Sign Language (ISL) we would need a gesture recognition system in which both the hands are used for performing a gesture to create some meaningful sentence.

##### **Sign Language**

There are about 5000 gestures in vocabulary

Each gesture consists of a hand shape, a hand motion and a location in 3D space It must be grammatically correct and syntactically precise.

##### **Software Interface**

Java

Android Studio

Android DSK

##### **Hardware Interface**

Hard Disk: minimum 250 GB

RAM: 4 GB

Android Phone with camera

#### **3.2 Product Function Modules**

**Modules in the system will be as,**

##### **Blob analysis**

In computer vision, **blob detection** methods are aimed at detecting regions in a digital image that differ in properties, such as brightness or color, compared to surrounding regions. Informally, a blob is a region of an image in which some properties are constant or approximately constant; all the points in a blob can be considered in some sense to be similar to each other.

## Template Matching

**Template matching** is a technique in digital image processing for finding small parts of an image which match a template image. It can be used in manufacturing as a part of quality control, a way to navigate a mobile robot, or as a way to detect edges in images.

## 3.3 Tools for Gesture Recognition

1. Static gesture (pose) recognition
  - Template matching
  - Neural networks
  - Pattern recognition techniques
2. Dynamic gesture recognition
  - Time compressing templates
  - Dynamic time warping
  - Hidden Markov Models
  - Conditional random fields
  - Time-delay neural networks
  - Particle filtering and condensation algorithm
  - Finite state machine

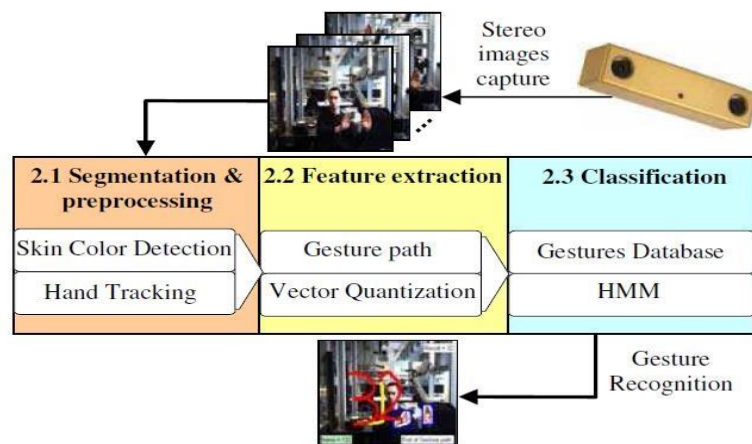


















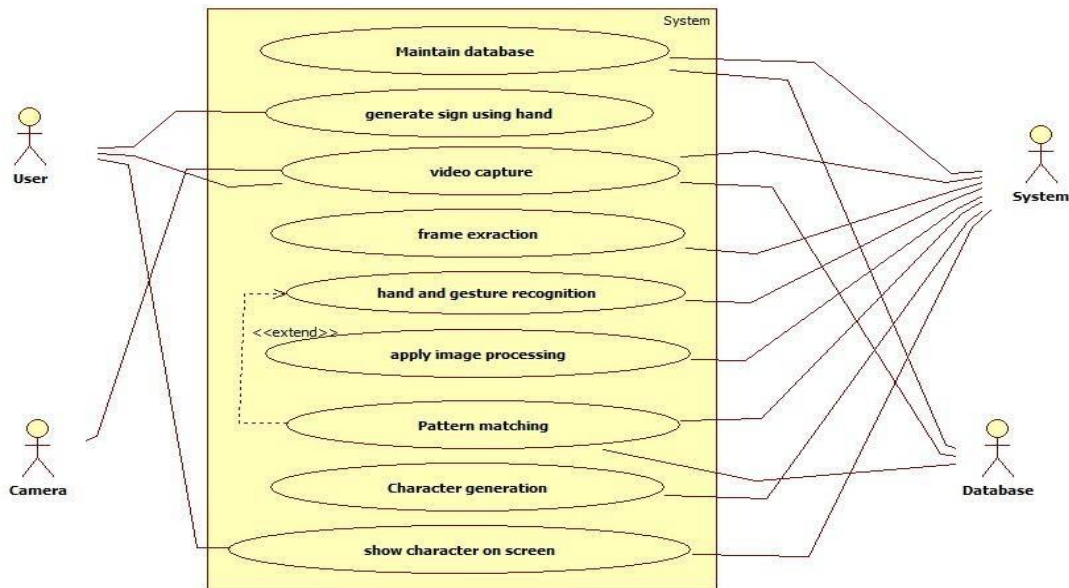
Fig. 2 Gesture Recognition System

ISL Gesture	Initial Position	Final Position
Shoot		
River		
Recover		
Quiet Down		
Miss		
Meet		
Heap		
Go		

**Fig 3 ISL video gestures with different light illumination condition**

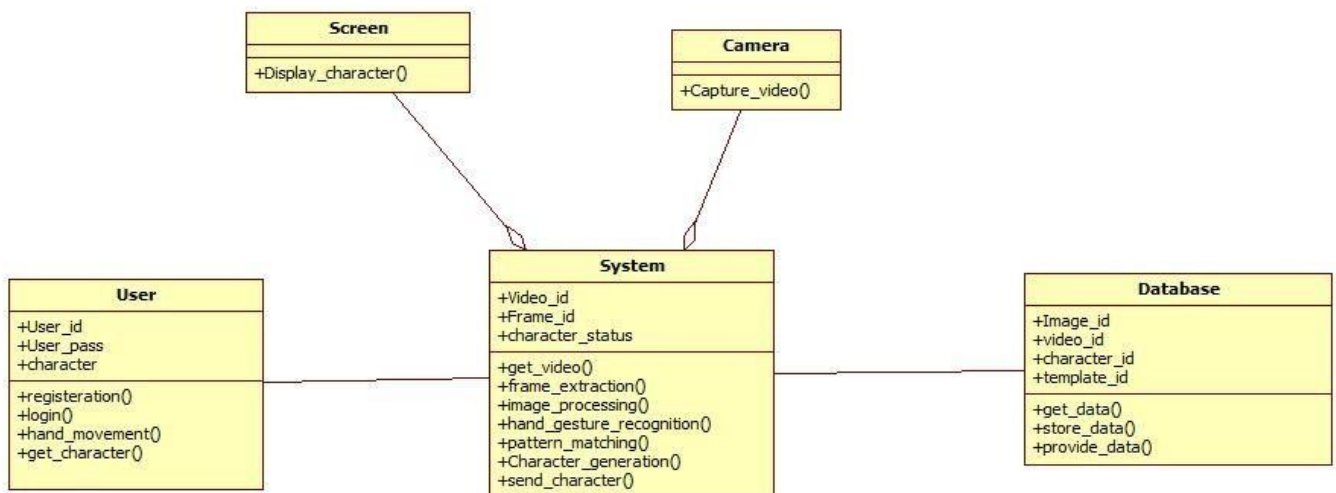
## CHAPTER 4

### DESIGN FOR A GESTURE RECOGNITION SYSTEM



**Fig. 4 Use Case Diagram**

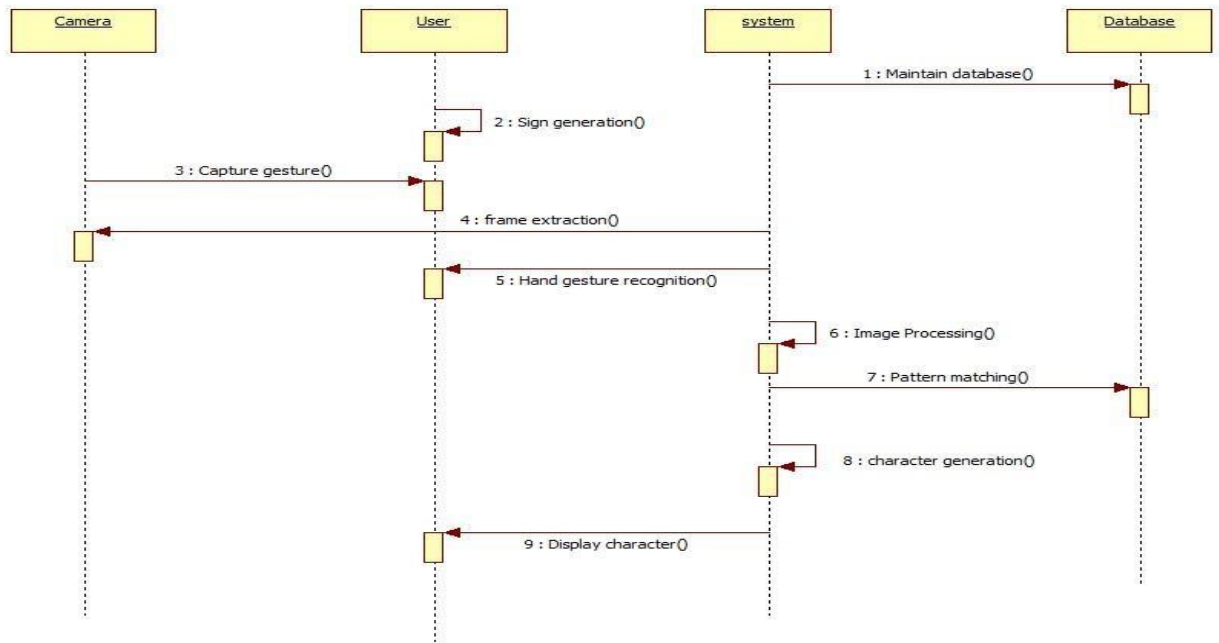
The use case diagram is a graphic depiction of the interactions among the elements of a system. It shows the methodology used in system to display the entire process from generation of hand signals by the user to display of the character on screen. It also shows the interactions of various actors with the system.



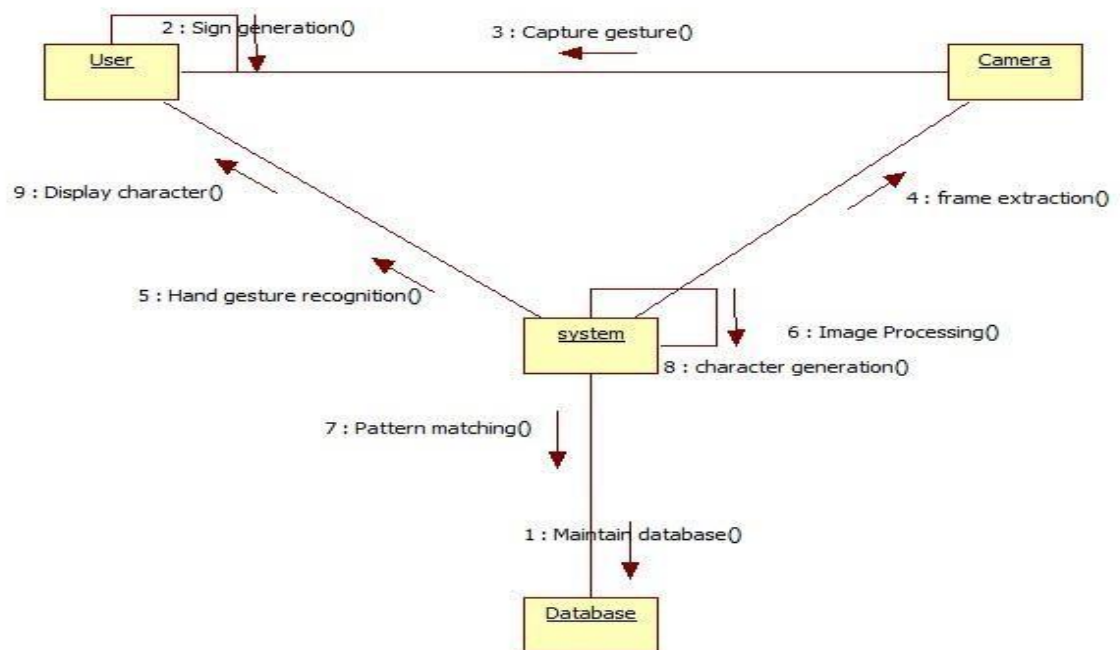
**Fig. 5 Class Diagram**

The class diagram is an illustration of the relationships and source code dependencies among classes involved in the development of the project. In this context, a class defines the methods and variables in an object, which involves proper declaration of variables involved and the functions associated with them.



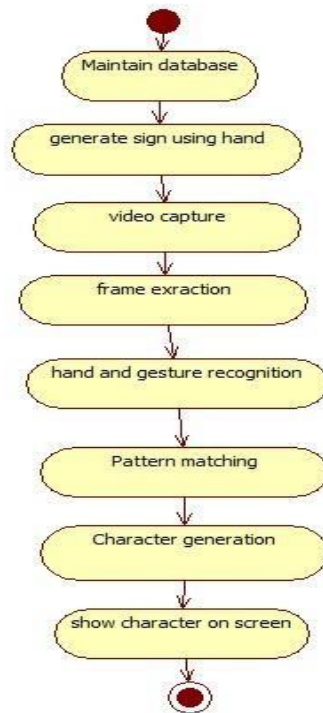


**Fig 6 Sequence Diagram 1**



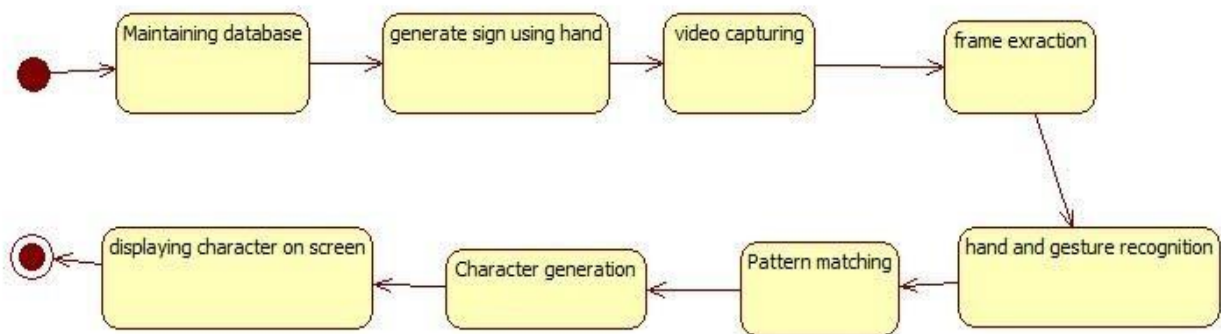
**Fig. 7 Sequence Diagram 2**

The Sequence diagram shows how processes would operate with one another and in what order interactively. It shows the object interactions arranged in a time sequence. It explains how the flow of the project will take place along with its proper time management.



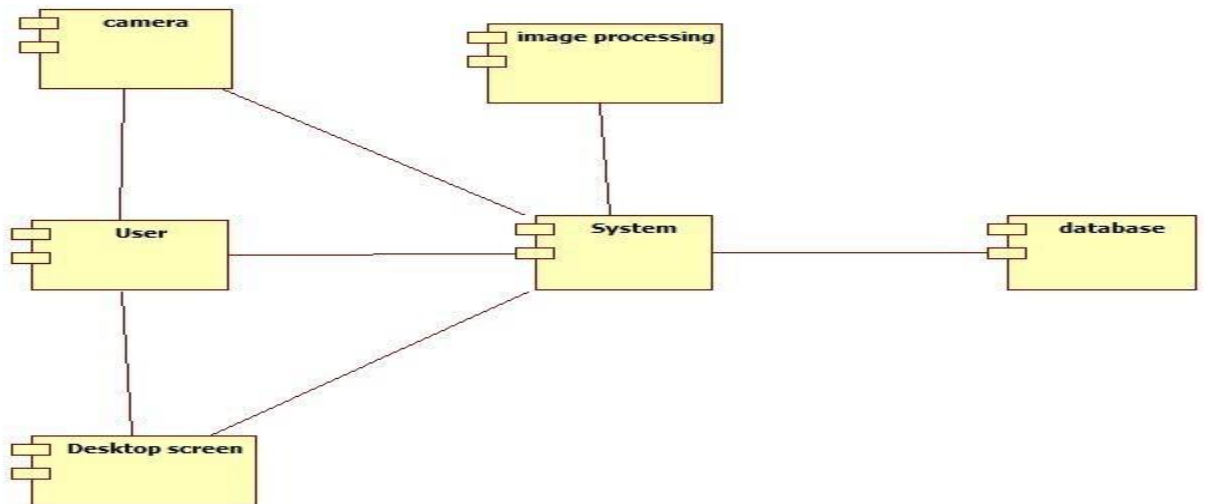
**Fig. 8 Activity Diagram**

Activity diagram is another important diagram used to describe dynamic aspects of this particular system. It is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.



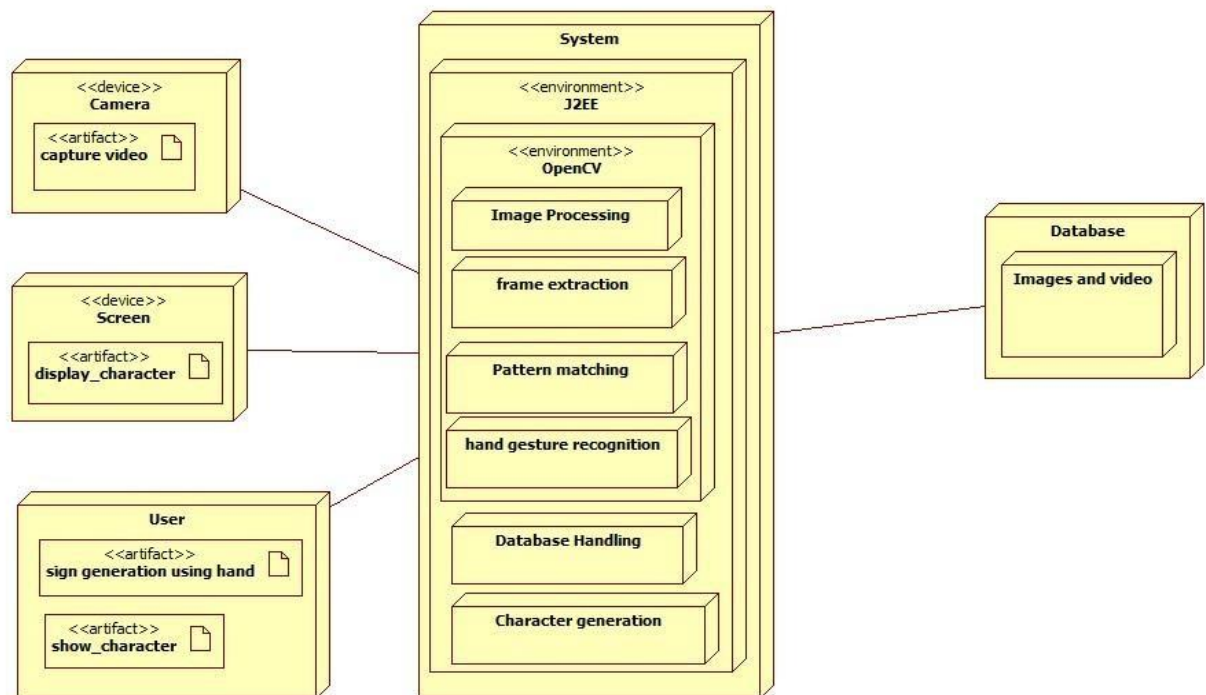
**Fig. 9 State Diagram**

A state diagram is a diagram used to describe the behaviour of the system considering all the possible states of an object when an event occurs. This behaviour is represented and analysed in a series of events that occur in one or more possible states. The possible states are arranged in the order of their execution along with the marking of the initial and final states.



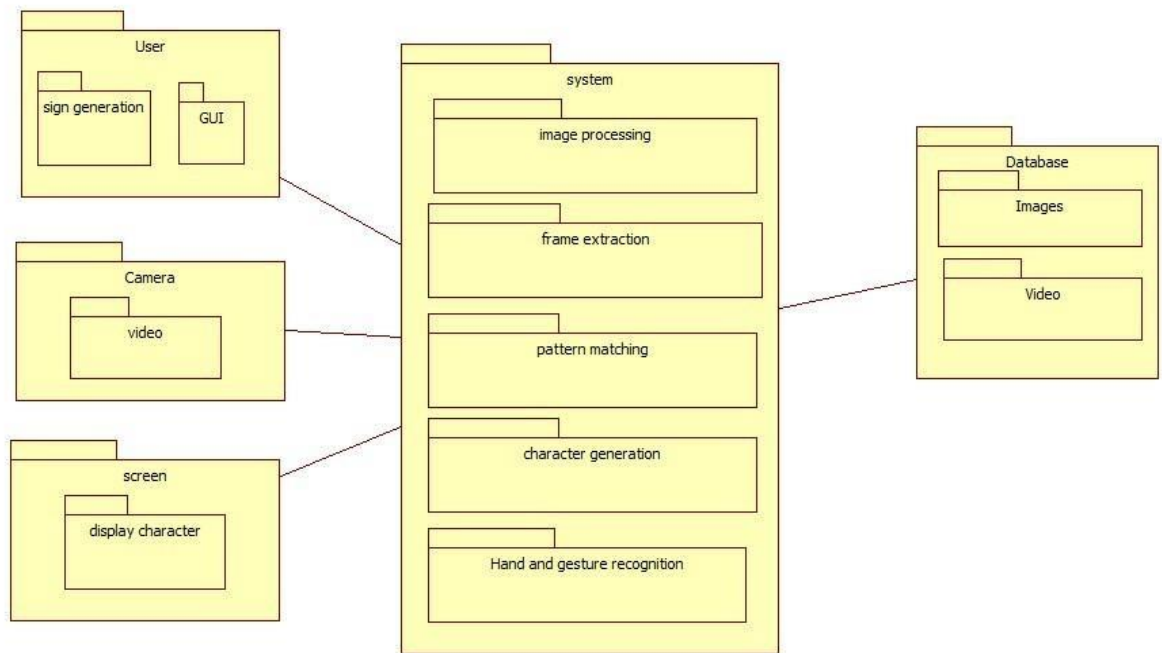
**Fig. 10 Component Diagram**

Component diagrams show the dependencies and interactions between software components. A component is a container of logical elements and represents things that participate in the execution of a system. Components also use the services of other components through one of its interfaces.

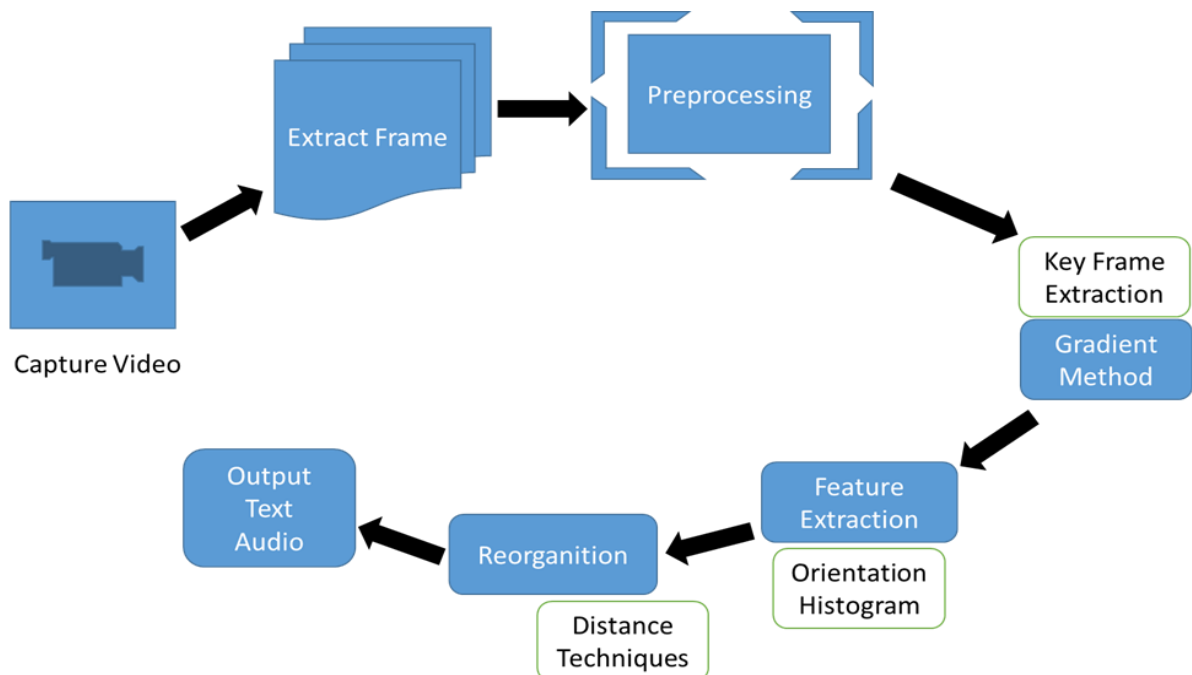


**Fig. 11 Deployment Diagram 1**

Deployment diagram is a structural diagram which shows the entire architecture of the system as deployment (distribution) of software artefacts to deployment targets of the system. Artefacts represent concrete elements in the physical world that are the result of a development process.



**Fig. 12 Deployment Diagram 2**



**Fig. 13 System Architecture Diagram**

The system architecture is the conceptual model that defines the structure, behaviour, and more views of the system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system. It comprises of the system components, the externally visible properties of those components, the relationships (e.g. the behaviour) between them. It also provides a plan from which the products can be procured, and systems developed, which will work together to implement the overall system.

## **CHAPTER 5**

### **IMPLEMENTATION OF GESTURE RECOGNITION SYSTEM**

For the gesture recognition the blob detection algorithm is used with gray scaling. Blob detection refers to mathematical methods that are aimed at detecting regions in a digital image that differ in properties, such as brightness or colour, compared to areas surrounding those regions. Informally, a blob is a region of a digital image in which some properties are constant or vary within a prescribed range of values; all the points in a blob can be considered in some sense to be similar to each other.

For the proposed system, we will be using blob detection methods for detection of areas of interest. It will be the first step in detecting motion. The method used for blob detection works with pixel intensity tracking. The local minima and maxima of the image are calculated and image is converted to a grayscale image.

#### **5.1 Data preparation**

In the proposed system, there is large amount of dataset with various skin tones. A large amount of dataset is to improve the detection accuracy.

#### **5.2 Image Segmentation**

In the image segmentation stage, it is important for reduce the background noise of the original image. Therefore, the pure background colour is required during the image capture progress. Moreover, the lighting condition and background noise also will influence the quality of the training data such as a hand and the background colour is not separated clearly due to the lighting condition. To tackle the problem, threshold value is used to separate the background noise in the training images.

#### **5.3. Thresholding:**

Thresholding in the system is used to create binary images. Binary means images having only black and white colours. On setting a particular threshold, parts of the image can be made entirely white or entirely black depending on the requirement. For the proposed system, thresholding is used to identify gesture and remove unwanted noise from the picture.

#### **5.4 Sample Code**

##### **5.4.1 LIST DIFFERENT CAMERAS**

```
private int[] loadCamera() {  
    ArrayList camDevice = new ArrayList();  
    int count = 0;  
    for (int device = 0; device < 10; device++) {
```

```
VideoCapture cap = new VideoCapture(device);
if (cap.isOpened()) {
    camDevice.add((count++), device);
}
cap.release();
}

int[] intArray = new int[camDevice.size()];
for (int i = 0; i < intArray.length; i++) {
    intArray[i] = (int) camDevice.get(i);
}

return intArray;
}
```

#### 5.4.2 CREATE DATASET

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    if (jTextField1.getText().equals("")) {
        JOptionPane.showMessageDialog(this, "Enter First Sign Name!!");
    } else {
        imgTX = imgX.clone();
        MatToBufImg mtb = new MatToBufImg(imgTX, ".jpg");
        this.img = mtb.getImage();
        Image imageScaled = this.img.getScaledInstance(
            320,
            240, Image.SCALE_SMOOTH);
        ImageIcon iic = new ImageIcon(imageScaled);
        jLabel2.setIcon(iic);
        Highgui.imwrite("" + tempaletePath + "" + jTextField1.getText() + ".jpg", imgTX);
        or = new ObjectRecognizer(new File(tempaletePath));
    }
}
```

#### 5.4.3 TRAIN THE SYSTEM FOR RECOGNITION, READ THE FILE FROM SPECIFIED DIRECTORY CALCULATE THE DESCRIPTOR AND KEYPOINTS USING ORB ALGORITHM

```
public ObjectRecognizer(File trainDir) {

    ArrayList<File> jpgFiles = Utilities.getJPGFiles(trainDir);
```

```
trainImages = Utilities.getImageMats(jpgFiles);
objectNames = Utilities.getFileNames(jpgFiles);

detector = FeatureDetector.create(FeatureDetector.ORB);
descriptor = DescriptorExtractor.create(DescriptorExtractor.ORB);
matcher = DescriptorMatcher.create(DescriptorMatcher.BRUTEFORCE);

trainKeypoints = new ArrayList<MatOfKeyPoint>();
trainDescriptors = new ArrayList<Mat>();

for (int i = 0; i < trainImages.size(); i++) {
    trainKeypoints.add(new MatOfKeyPoint());
    detector.detect(trainImages.get(i), trainKeypoints.get(i));
    trainDescriptors.add(new Mat());
    descriptor.compute(trainImages.get(i), trainKeypoints.get(i),
        trainDescriptors.get(i));
}
matcher.add(trainDescriptors);
matcher.train();
}
```

#### 5.4.4 USE KNN TO FIND THE BEST MATCH

```
private void getMatches_ratioTest(Mat descriptors,
    List<MatOfDMatch> matches, double ratio) {
    LinkedList<MatOfDMatch> knnMatches = new LinkedList<MatOfDMatch>();
    DMatch bestMatch, secondBestMatch;

    matcher.knnMatch(descriptors, knnMatches, 2);
    for (MatOfDMatch matOfDMatch : knnMatches) {
        bestMatch = matOfDMatch.toArray()[0];
        secondBestMatch = matOfDMatch.toArray()[1];
        if (bestMatch.distance / secondBestMatch.distance <= ratio) {
            MatOfDMatch goodMatch = new MatOfDMatch();
            goodMatch.fromArray(new DMatch[]{bestMatch});
            matches.add(goodMatch);
        }
    }
}
```

## CHAPTER 6

### RESULT AND EVALUATION

#### 6.1 Result

Following chapter includes the screenshots of the GUI of the system along with its working. The snapshots of system are given below.

##### 6.1.1 GUI of the system

When the system is started the following GUI appears on the screen. It consists of various buttons clearly specifying their use. A drop down menu has been provided providing the list of cameras. Separate boxes is provided for saving a gesture and recognising a gesture respectively.

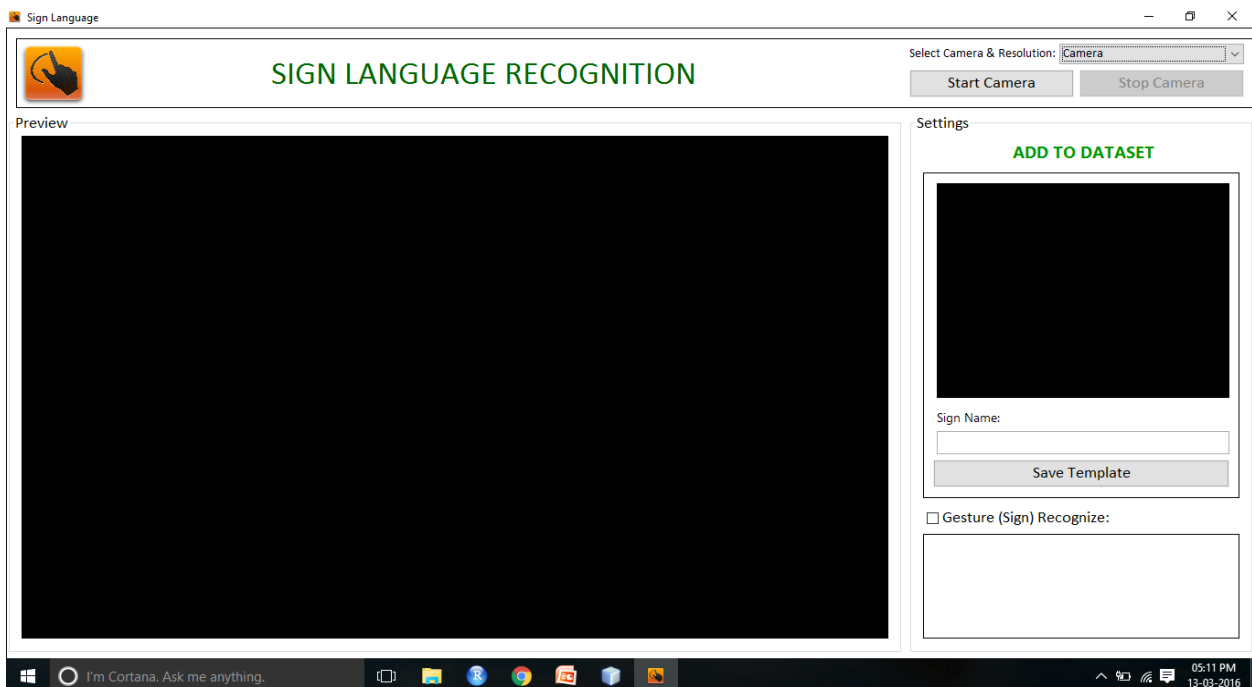


Fig. 14 GUI of the system

##### 6.1.2 Training the system to create gesture dataset

The system training procedure basically involves capturing the gestures via the built-in laptop camera and then saving the images after giving them the respective gesture name. The image is saved by clicking on the ‘Save Template’ button. It uses the ORB (Oriented FAST and Rotated BRIEF) algorithm.



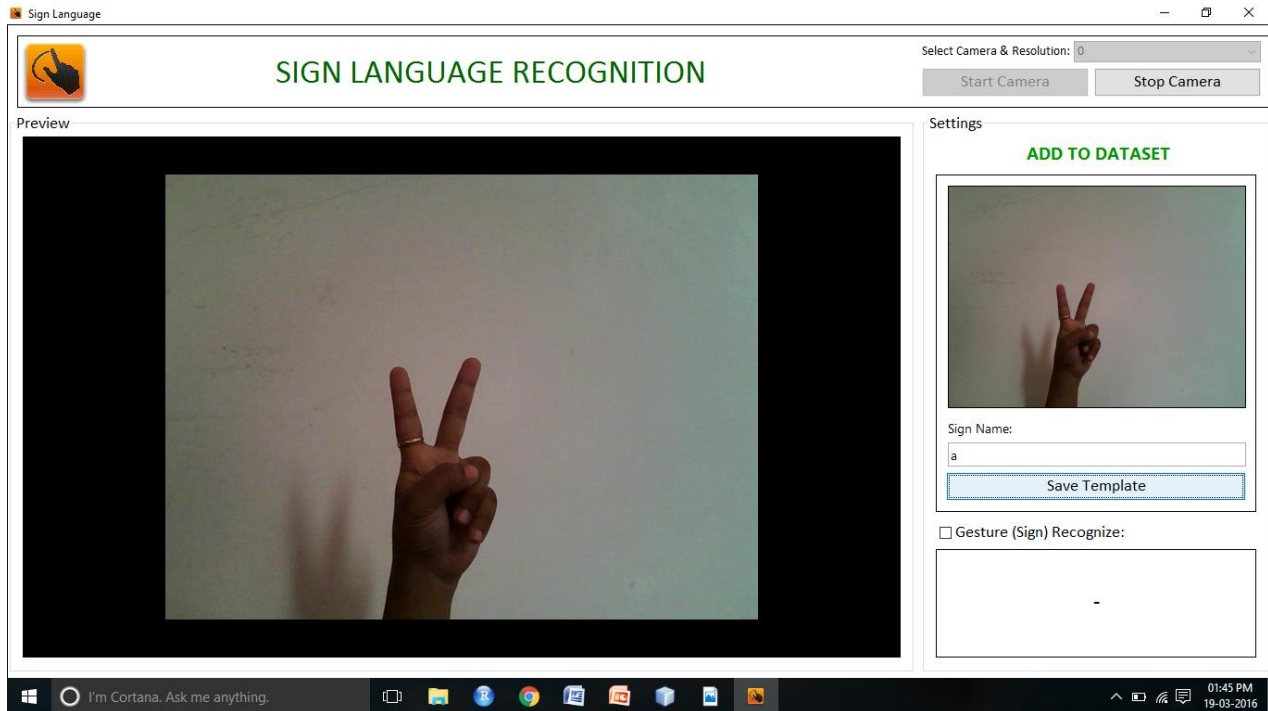


Fig. 15 Training the database

### 6.1.2 Recognition of the gesture

The gesture is recognised by using the blob detection algorithm and grey scaling features together. The feature points are recognised and output is finalised using the best matched output of the KNN.

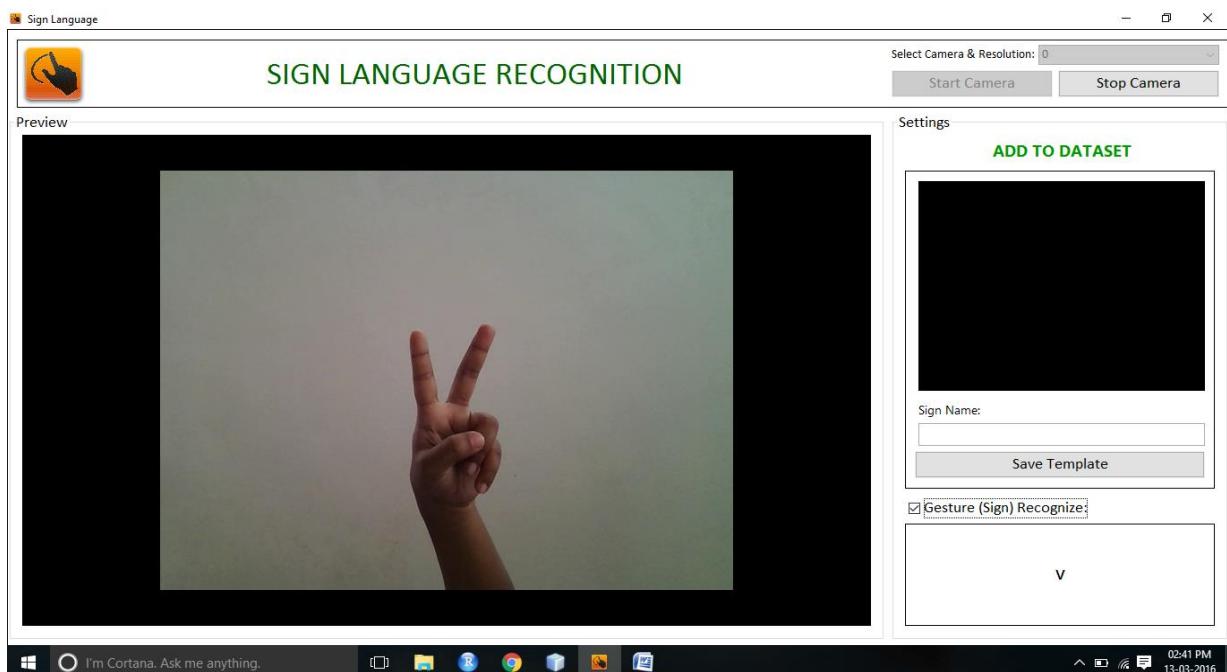


Fig. 16 Gesture Recognition

## 6.2 Result Analysis

The result analysis shows that the system cannot utilise the depth sensing techniques of a camera. As the camera is a VGA camera, it cannot sense depth and so the feature points available from the background are sometimes also included as a part of the gesture. So, a plain background is must. Secondly, as the system is dynamic it sometimes flickers while displaying the output. Also, the system has to be trained with all the skin tones for seamless performance.



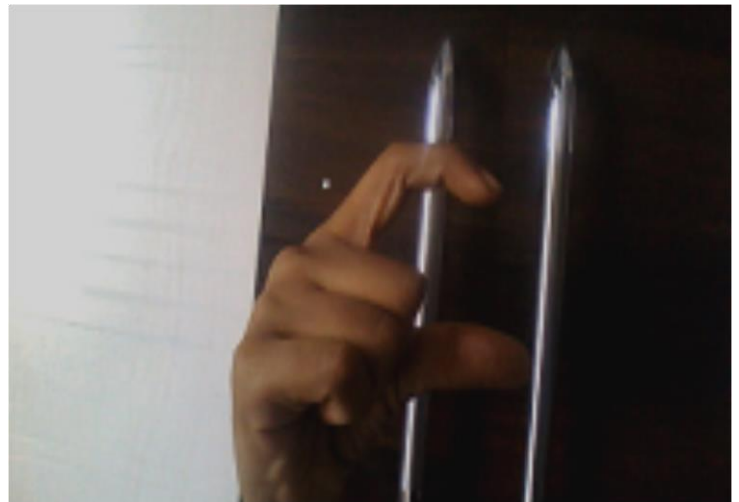
**Fig. 17 Gesture 'A' (ideal background)**



**Fig. 18 Gesture 'A' (complex background)**



**Fig. 19 Gesture 'C' (ideal background)**



**Fig. 20 Gesture 'C' (complex background)**

## 6.2 Test Cases

### 6.2.1 Testing GUI

TC_Name	Objective	Pre requisite	Description	Expected Result	Actual Result	Status
<b>TC01_Open application</b>	Verify open application from device.	Application should get installed on device.	1. Click on application from device.	Application should get opened.	Application is getting opened.	Pass
<b>TC02_Screen</b>	Verify first screen from application .	Application should be open.	1. Open application from.	Sign language window should be opened.	Sign language window is getting opened.	Pass
<b>TC03_Screen_group box</b>	Verify group box from screen.	Application should be open.	1. Open application from.	Application should show 3 group box Image processing, Training database & Result.	Application is showing 2 group on screen.	Pass

Table 2 Test Cases (Testing GUI)

### 6.2.2 Creating Dataset

TC_Name	Objective	Pre requisite	Description	Expected Result	Actual Result	Status
<b>TC04_Image processing</b>	Verify image processing group box from application.	Application should be open.	1. Open application from.	Application should show 4 windows in this group ie. Original screen, Skin detection, large blob, detected hand pos.	Application is showing 4 windows on screen.	Pass
<b>TC05_Add hand Pos name</b>	Verify Add hand pos name button from application.	Application should be open.	1. Keep your hand with particular unique position in front of the camera.	Application should draw red box to the particular position in detected hand pos screen.	Application is showing red box.	Pass

<b>TC06_Original image</b>	<b>Verify original image screen from application.</b>	<b>Application should be open.</b>	<b>1. Keep your hand with particular unique position in front of the camera.</b>	<b>Application should show original image wrt Hand pos.</b>	<b>Application is showing original image window.</b>	<b>Pass</b>
<b>TC07_skin detection</b>	Verify skin detection from application.	Application should be open.	1. Keep your hand with particular unique position in front of the camera.	Application should detect skin image wrt Hand pos.	Application is showing skin detection window.	Pass
<b>TC08_large blob</b>	Verify large blob from application.	Application should be open.	1. Keep your hand with particular unique position in front of the camera.	Application should draw large blob wrt skin large skin detection area.	Application is drawing large blob wrt. Skin detection.	Pass
<b>TC09_Add hand Pos name</b>	Verify Add hand pos name button from application.	Application should be open.	1. Keep your hand with particular unique position in front of the camera. 2. Enter hand position name. 3. Click on Add hand pos name.	Application should capture red box image as a symbolic image & should show in training template.	Application is showing captured image in training template.	Pass

**Table 3 Test Cases (Creating Dataset)**

### 6.2.3 Recognising Gestures

TC_Name	Objective	Pre requisite	Description	Expected Result	Actual Result	Status
<b>TC11_Detect &amp; recognized</b>	Verify detect & recognized from screen.	Application should be open.	1. Start application. 2. Click on detect & recognized button from screen. 3. Keep your saved hand position in front of the camera.	Application should match detect hand position with saved db value & should show value respectively in the result.	Application is showing saved value in the result.	Pass
<b>TC12_clear</b>	Verify clear button from screen.	Application should be open.	1. Do number of hand pos in front of the camera. 2. Click on clear button.	Application should show values on screen, After clicking on clear button detected result should get clear.	Detected result window is getting clear.	Pass
<b>TC13_Close application</b>	Verify close application from device.	Application should be open.	1. Click on close button from screen.	Application should get closed.	Application is closed.	Pass

**Table 4 Test Cases (Recognising Gestures)**

## **CHAPTER 7**

### **CONCLUSION, LIMITATION AND FUTURE ENHANCEMENT**

#### **7.1 Conclusion**

- The Sign Language application is incredibly unique software with a ton of potential for mute people. The brighter side of creating such an application is that this application would promote the use of Devnagri Script and help the deaf and mute kids in learning communication through sign languages.
- It gives an accurate level of detail, as provided by the camera. A modern day camera works as similar as a human eye. With maximum available accuracy, the camera would help in the working of the application.
- It also allows recognisable signs to be programmed or trained into the device for future recognition. This would help in modifying the gestures if needed and help in developing applications further in other Sign Languages.

#### **7.2 Limitations**

- The desktop camera is not as capable as the depth sensing cameras.
- Complex signs are hard to detect as the accuracy is less.
- As real time detection is used the system exhibit dynamic output.
- If the background used while detection is not plain it adds an overhead to the system.

#### **7.3 Future Enhancements**

- Accuracy of the system can be improved by using high quality camera or multiple cameras.
- Each user will be able to modify the dataset to suit his/her needs.
- Security can be increased by making the system password protected.

## **CHAPTER 6**

### **REFERENCES**

- [1] M. Elmezain, A. Al-Hamadi, J. Appenrodt and B. Michaelis, A Hidden Markov Model-based Continuous Gesture Recognition System for Hand Motion Trajectory, 19th International Conference on IEEE, Pattern Recognition, 2008, ICPR 2008, pp. 1–4, (2008).
  
- [2] V. Athitsos, Quan Yuan and S. Sclaroff, A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation, IEEE Pattern Analysis and Machine Intelligence, IEEE Transactions, vol. 31, pp. 1685–1699, September (2009).
  
- [3] Nandy, S. Mondal, J. S. Prasad and P. Chakraborty, Recognizing & Interpreting Indian Sign Language Gesture for Human Robot Interaction, International Conference on Computer and Communication Technology (ICCCT), (2010), pp. 712–717, 17–19 September (2010).