IQRA UNIVERSITY IU

AWAZ

Communicate With Deaf & Dumb

Will be presented on

07 FEB 2021

Bhawish Raj

+92 3479344691

rajbhawish3@gmail.com

GUIDELINES FOR MAKING FINAL YEAR PROJECT-1 REPORT

The students of FEST are hereby directed to prepare their final year project 2 reports according to the guidelines provided in the following points.

- The color of front cover binding of the project report should be **Black**, for all the students.
- The project report should be hard bound
- The front cover of the report should be in printed form along-with the university logo. The required print spacing on front cover page is as follows:

  Top: 1.5 inches   Left: 1.5 inches

  Right: 1 inch   Bottom: 1 inch

  Note: For the front cover details contact your project coordinator

- The font used should be **Times New Roman, 12 points plain**, with one-and-a-half (1.5) spacing between two consecutive lines
- For sub-headings, increase Two point and for main headings increase Two more point.
- The overall number of pages in the report must be a minimum of **60, Excluding the appendix**
- All the pages should be properly numbered – preferably section-wise
- The students should get their project reports cross-checked by their advisor before submitting it to the FYPC. Marks will be deduced from the reports not complying the prescribed format.

Final Year Project Report

AWAZ

BS (Computer Science)

Submitted by

Bhawish Raj	REG. #44916
Ritu Bai	REG. #38708
Manisha Chandwani	REG. #40564

Project Advisor

Syed Safdar Hussain

Project Coordinator

Dr. Mansoor Ebrahim

February, 2021

Faculty of Engineering, Sciences and Technology

IQRA University, Karachi

ACKNOWLEDGEMENTS

"START WITH NAME OF ALLAH WHO IS MOST BENEFICENT AND MERCIFUL "

We would like to thank our mentor **Dr. Mansoor Ebrahin** (HOD, Department of Software Engineering) for giving us the opportunity to undertake the project. We thank them for their immense guidance, and appreciate their timely engagement.

We wish to express our gratitude to our project supervisor "**Syed Safdar Hussain**", who gave constant supervision despite his busy schedule, for the technical guidance provided to move us in a correct path, truly for the patience he had when working with our team, and without whom it would not be a success.

We might want to communicate because of all instructors, they encouraged us all through our examinations and guide us, and without their direction, this excursion would not have been conceivable.

INTRODUCTION TO GROUP MEMBERS

Bhawish Raj **Reg. # 44916**
(Python Developer)
Immediate Contact: (+92 333 8381990, rajbhawish3@gmail.com)

Ritu Bai **Reg. # 38708**
(UI/UX Designer)
Immediate Contact: (jhambiaritz@gmail.com)

Manisha Chandwani **Reg. # 40564**
(UI Designer)
Immediate Contact: (manishachandwani@gmail.com)

ABSTRACT

Inability to speak is considered true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language.

Pakistan sign language (PSL) is one of the sign languages of the world used by the Pakistani deaf community. The major problem that we observed is the communication barrier between normal people and a deaf community. The normal people in our society are completely unaware from the signs/gestures of PSL. This communication barrier seizes the basic right of communication between a deaf and a normal. Our contribution in this regard is to decrease the communication barrier between the normal community and a deaf community. One of the solution is to design an automated system, which would be helpful in two-way communication, as if a normal person will record signs and system would be able to convert it in text or voice, and a deaf person will record voice of normal person and system would be able to convert it in signs. All above thoughts could be converted into real world application by using some deep learning algorithms and Artificial intelligence to recognize the signs and voice or text.

We are hopeful that this effort will surely help to develop the systems for reducing gap between deaf-to-deaf, dumb-to-dumb, dumb to normal and Pakistan deaf and dumb community to any other deaf and dumb community.

LIST OF ACRONYMS

PSL - Pakistan Sign Language

D&D – deaf and dumb

AI - Artificial Intelligence

ML - Machine Learning

IP - Image Processing

API - Application Programming Interface

FAT – Finite Automata

TF – Tensorflow

CV – Capture Video

CNN – Convolutional Neural Network

HCI – Human Computer Interaction

TABLE OF CONTENTS

"The reference must be in accordance with the given reference format"

"The appendix must include all the circuit diagrams, PCB Layouts, Data sheets, and Source code"

LIST OF FIGURES

CHAPTER # 1 INTRODUCTION

1.1 INTRODUCTION:

A movement of part of the body, especially a hand or the head, to express an idea or meaning is known as gesture. Here for gesture recognition we are using image processing, opencv, numpy and some other liberaries. Gesture acknowledgment empowers computers to comprehend human activities and furthermore goes about as a mediator among computer and human. This could give potential to human to communicate normally with the computers with no actual physical contact of the mechanical gadgets. Gestures are performed by deaf and dumb community to perform sign language. This community used sign language for their communication when broadcasting audio is impossible, or typing and writing is difficult, but there is the vision possibility. At that time sign language is the only way for exchanging information between people.

Normally sign language is used by everyone when they do not want to speak, but this is the only way of communication for deaf and dumb community. Sign language is also serving the same meaning as spoken language does. This is used by deaf and dumb community all over the world but in their regional form like ISL (Isolated Signed Language), CSL (Continuous Isolated Signed Language).

1.2 PROJECT OBJECTIVE

The objective of our project is to introduce you to a new culture and community, by reducing the barrier of communication between normal and deaf or dumb community. This project can change theoretical signed language to digitalized signed language approach. This approach could be beneficial in classroom learning for deaf community.

Sign language is a visual language and consists of 3 major components:

Fingerspelling	Word level sign vocabulary	Non-manual features
Used to spell words letter by letter .	Used for the majority of communication.	Facial expressions and tongue, mouth and body position.

Figure 1 – Project objectives

1.3 RELEVANCE OF DOMAINS

The work done on this project is reliant on an research work which we did from the past education in the Computer Science Field just as gained from the self-improvement, this part clarifies a portion of the fundamental highlights which are intensely used and actualized in our Project. The signs and the database is provided by deaf reach school.

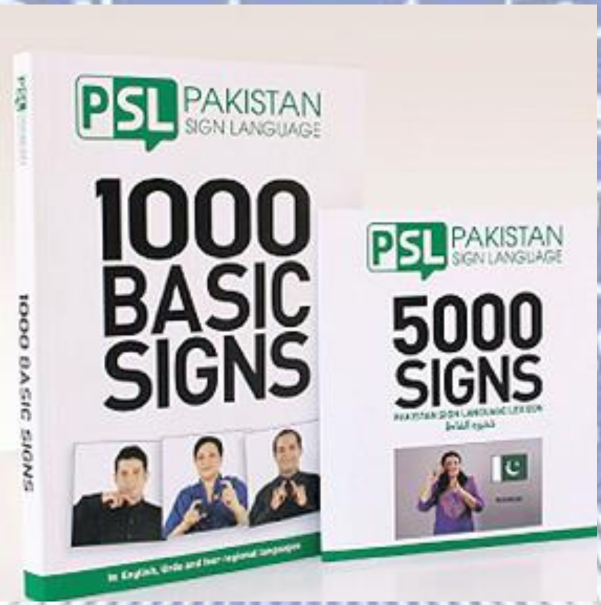


Figure 2 – Relevance of domains

CHAPTER # 2 TECHNOLOGY BACKGROUND :

2.1 Overview of basic technology

In the recent years, there has been tremendous research on the hand sign language gesture recognition. The technology for gesture recognition is given below.

Vision Based

In vision-based methods, computer camera is the input device for observing the information of hands or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vision by describing artificial vision systems that are implemented in software and/or hardware. This poses a challenging problem, as these systems need to be background invariant, lighting insensitive, person and camera independent to achieve real time performance. Moreover, such systems must be optimized to meet the requirements, including accuracy and robustness.

The vision based hand gesture recognition system is shown in figure.

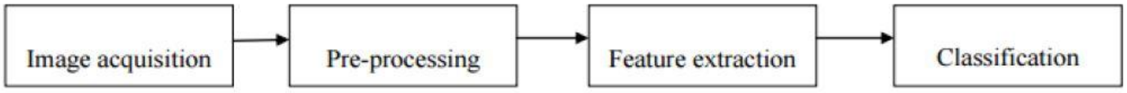


Figure 3 – Project flow

Opencv

OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations.

Figure 4 – OpenCV logo

NumPy

NumPy (pronounced (NUM-py) or sometimes (NUM-pee)) is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

Figure 5 – NumPy Logo

Math (python library)

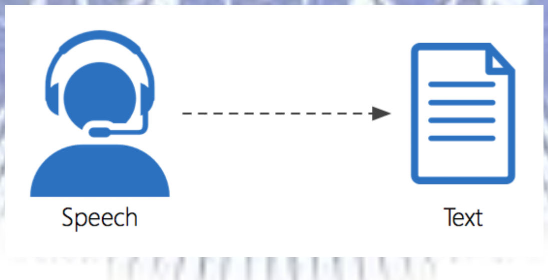


Figure 6 – math python library

Python 2.1 and earlier used C's division behavior. The / operator is integer division if both operands are integers, and floating-point division otherwise. Integer division rounds towards 0, e.g. 7/3 == 2 and -7/3 == -2.

Python 2.2 changed integer division to round towards negative infinity, e.g. 7/3 == 2 and -7/3 == -3. The floor division // operator was introduced. So 7//3 == 2, -7//3 == -3, 7.5//3 == 2.0 and -7.5//3 == -3.0. Adding from __future__ import division causes a module to use Python 3.0 rules for division (see next).

Python 3.0 changed / to always be floating-point division, e.g. 5/2 == 2.5.

In Python terms, / is true division (or simply division), and // is floor division. / before version 3.0 is classic division.

Tkinter

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The name Tkinter comes from Tk interface. Tkinter was written by Fredrik Lundh. Tkinter is free software released under a Python license.

2.2 Background Research

Boltay Haath was one of the pioneers Pakistan Sign Language Recognition System. Boltay Haath used a computer based gesture recognition system and a data glove for converting signs produced by a deaf into audible speech. Their technique is based upon statistical template matching which receives incoming sign as input and compares it with the stored one in dataset of signs/gestures. They have taken hundred instances against each letter of English and Urdu alphabet. They have found an accuracy level ranging from 70 – 80%.

(However, there are several signs in PSL which requires both hands).

Another approach using data glove but this time the colored one is proposed by **Sumaira et al.** They have used a fuzzy classifier to recognize the signs/gestures performed by the deaf. Their algorithm uses the angle between finger tip and joint for classification of gesture. Their dataset was based on Urdu alphabets of Pakistan sign language. They have achieved an overall accuracy of 95% as 35 out of 37 alphabets were recognized correctly. Again, the cost of color data glove, recognition of single hand gestures and static gestures are the limitations of their system.

An inverse system is deployed by **Ahmed et al** which serves the deaf community in different way. Their system “PSLIM‟ reduces the communication gap between the hearing and a hearing impaired person. They listen to the audio and convert it into sign/gesture of PSL. They have used the vocabulary from developed by Zahoor. They have achieved an overall accuracy of about 78%.

Another work for reducing communication gap between the deaf and normal is done by **Asif Ali**. They have proposed a system which takes input in both forms text and image of sign and convert it into other form. They have performed this for Urdu alphabets of PSL using Haar classifier. They have used a simple RGB camera for this purpose. However, they have not specifically mentioned the size and nature of dataset used for experimentation. The accuracy rate of developed system is also missing. An image processing based approach for the recognition of Urdu signs/gestures of PSL has been proposed by Khan et al. Deaf as novice can also learn computer programming and software engineering etc. Farooq et al propose a comprehensive framework and pedagogically effective subset of C++. They have captured 500 images of 37 letters of Urdu alphabets by using an RGB camera. They have used image processing techniques for skin region filtering. After that they extracted the features of interest by applying discrete wavelet transformation (DWT) and trained a neural network for recognition/classification purpose. To the best of our knowledge, this is the first approach without using any kind of hardware like data glove etc. They have presented their output in textual form. Out of 500 images they have selected 426 for training and 74 for testing and achieved an accuracy of about 86%. The salient feature of their work is that it handles the pose variation of hand as well. However, their work is restricted only for static and single hand gestures.

Another recent work related to recognition of Urdu alphabets of Pakistan sign language is done by **Sami et al**. Their work is another addition regarding recognition of Pakistan sign language using computer vision techniques. They have used cross correlation to find closed match between input image and the image dataset. They have performed their experimentation for the 37 alphabets of Urdu sign language and achieved an accuracy of about 75%. Again the limitations of their work include the recognition of static and single hand gestures.

CHAPTER #3 REQUIREMENT AND METHODOLOGY

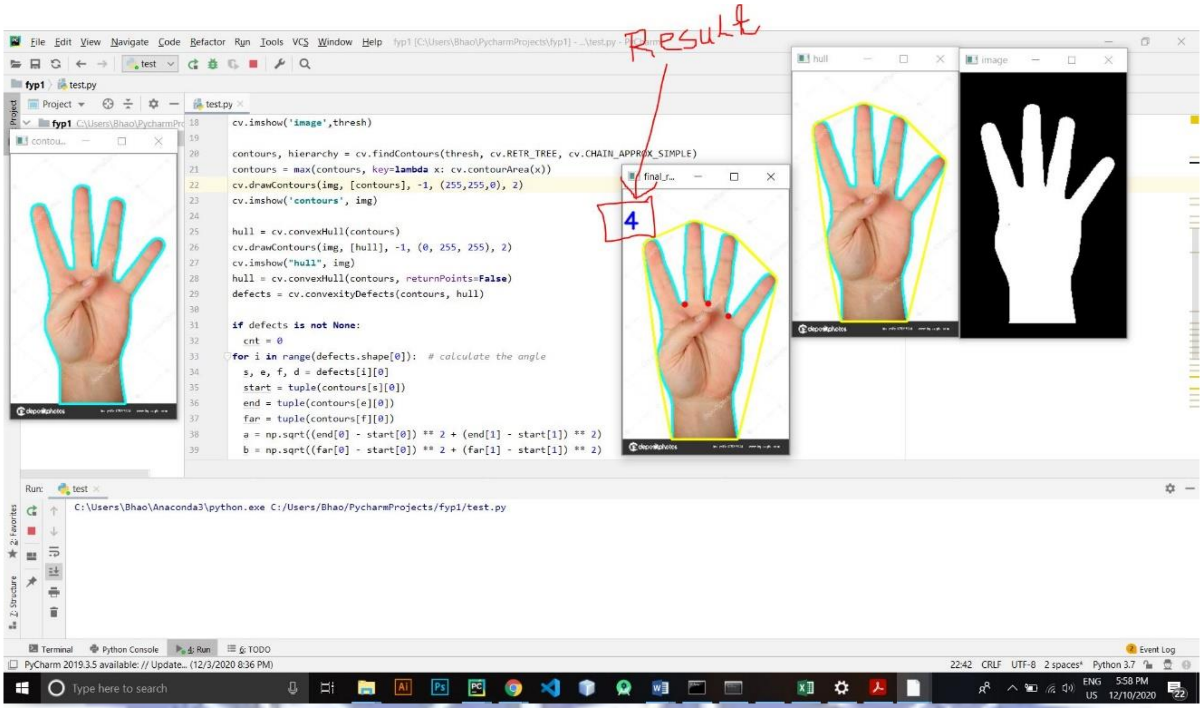
In past those who has worked for supporting PSL (Pakistan Sign Language) are heartily appreciated but those projects contains many flows. Therefore, PSL needs much more improvements and technical development. In addition, after this project we can lighten up the name of our institute and our beloved and respected teachers and coordinators. This application would be more helpful for the deaf and dumb community because of its features like the conversion of audio into text and on other side, to record the live signs from camera and then convert it to text or audio form.

3.1 Overview of Basic Topics

Approach One

- Convert static images to signs
- Create dataset for static images
- Create contours of images

- Background remove
- Detect hands in images and convert to text


Result
4
File Edit View Navigate Code Refactor Run Tools VCS Window Help
fyp1 [C:\Users\Bhao\PycharmProjects\fyp1] - ...\test.py - PyCharm
test
fyp1 > test.py
Project
fyp1 C:\Users\Bhao\PycharmPro
contou...
final_r...
hull
image
test.py
Run: test
Terminal
Python Console
4: Run
6: TODO
Event Log
PyCharm 2019.3.5 available: // Update... (12/3/2020 8:36 PM)
22:42 CRLF UTF-8 2 spaces* Python 3.7
Type here to search
ENG US 5:58 PM 12/10/2020

```
cv.imshow('image',thresh)

contours, hierarchy = cv.findContours(thresh, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
contours = max(contours, key=lambda x: cv.contourArea(x))
cv.drawContours(img, [contours], -1, (255,255,0), 2)
cv.imshow('contours', img)

hull = cv.convexHull(contours)
cv.drawContours(img, [hull], -1, (0, 255, 255), 2)
cv.imshow("hull", img)
hull = cv.convexHull(contours, returnPoints=False)
defects = cv.convexityDefects(contours, hull)

if defects is not None:
  cnt = 0
for i in range(defects.shape[0]):  # calculate the angle
  s, e, f, d = defects[i][0]
  start = tuple(contours[s][0])
  end = tuple(contours[e][0])
  far = tuple(contours[f][0])
  a = np.sqrt((end[0] - start[0]) ** 2 + (end[1] - start[1]) ** 2)
  b = np.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
```

```
C:\Users\Bhao\Anaconda3\python.exe C:/Users/Bhao/PycharmProjects/fyp1/test.py
```

Figure 7 – Approach One

3.1.1 Approach Two

As we probably are aware, the vision-based innovation of hand gesture recognition is an essential piece of human-PC communication (HCI). In the most recent decades, console and mouse assume a noteworthy job in human-PC communication. Nonetheless, attributable to the fast improvement of equipment and programming, new sorts of HCI strategies have been required. Specifically, advances, for example, discourse recognition and gesture recognition get extraordinary consideration in the field of HCI.

- Write Algorithm to detect live images using cosine rule
- Capturing live video
- Frame Extraction and background removal
- Using Gaussian blur technique
- Change color space

- Create binary image
- Find maximum contours
- Use cosine rule to find angle of the far point
- Find convex points

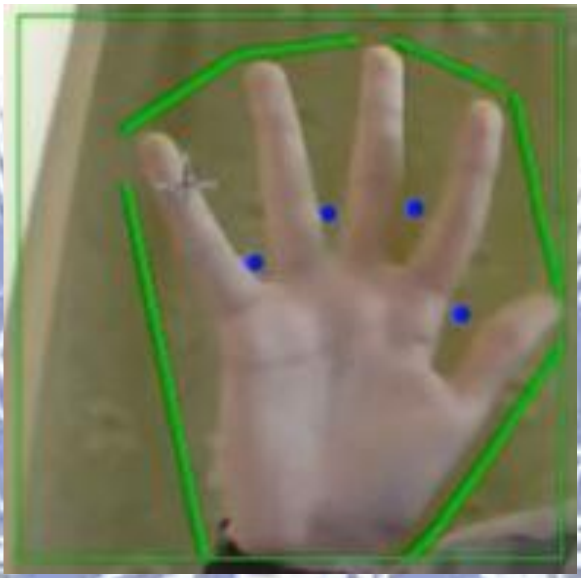


Figure 8 – Approach Two

3.2 Background Research

Boltay Haath was one of the pioneers Pakistan Sign Language Recognition System. Boltay Haath used a computer based gesture recognition system and a data glove for converting signs produced by a deaf into audible speech. Their technique is based upon statistical template matching which receives incoming sign as input and compares it with the stored one in dataset of signs/gestures. They have taken hundred instances against each letter of English and Urdu alphabet. They have found an accuracy level ranging from 70 – 80%.

(However, there are several signs in PSL which requires both hands).

Another approach using data glove but this time the colored one is proposed by **Sumaira et al.** They have used a fuzzy classifier to recognize the signs/gestures performed by the deaf. Their algorithm uses the angle between finger tip and joint for classification of gesture. Their dataset was based on Urdu alphabets of Pakistan sign language. They have achieved an overall accuracy of 95% as 35 out of 37 alphabets were recognized correctly. Again, the cost of color data glove, recognition of single hand gestures and static gestures are the limitations of their system.

An inverse system is deployed by **Ahmed et al** which serves the deaf community in different way. Their system "PSLIM‟ reduces the communication gap between the hearing and a hearing impaired person. They listen to the audio and convert it into sign/gesture of PSL. They have used the vocabulary from developed by Zahoor. They have achieved an overall accuracy of about 78%.

CHAPTER #4 SYSTEM MODEL

4.1 Block Diagram of Overall Circuit (Explained)

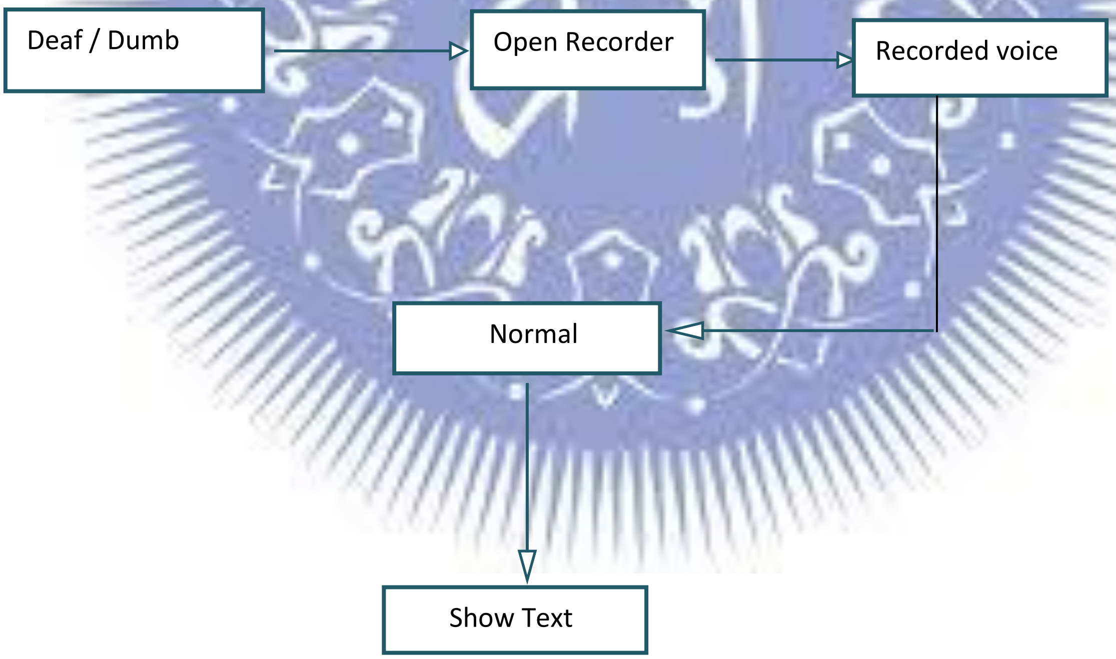


Figure 9 – Block Diagram 1

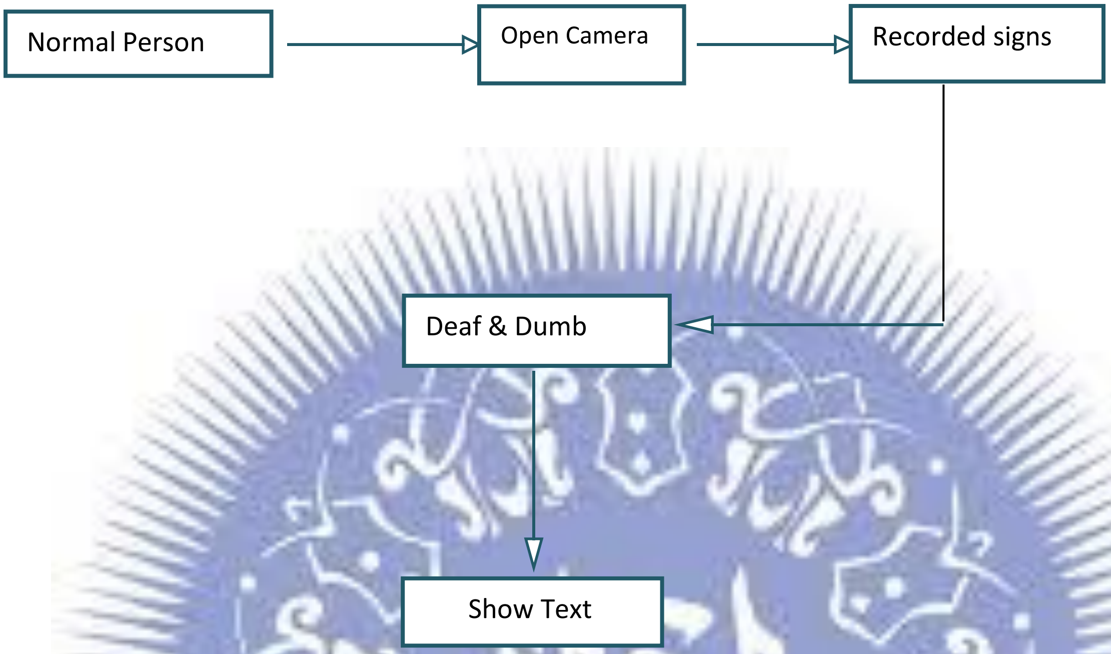


Figure 10 – Block Diagram 2

4.3 Source Code (Only Algorithm) Explained

1. Start
2. Open the camera by using function = capture = cv2.VideoCapture(0) #Open video file or image file sequence or a capturing device or a IP video stream for video capturing.
3. Use while loop for capturing until the camera is switched onn.
4. Create the sub window by using function, img = cv.rectangle(img, rec, color[, thickness[, lineType[, shift]]]

   Of size cv2.rectangle(frame, (100, 100), (300, 300), (0, 255, 0), 0)
5. Now apply the Gaussian blur on selected frame b using function

   blur = cv2.GaussianBlur(crop_image, (3, 3), 0)

   The Gaussian filter is a low-pass filter that removes the high-frequency components are reduced.
6. Now select the Gaussian blur image and change the color space from BGR to HSV hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)
7. # Create a binary image with where white will be skin colors and rest is black by using function `mask2 = cv2.inRange(hsv, np.array([2, 0, 0]), np.array([20, 255, 255]))`

8. The numpy.ones() function returns a new array of given shape and type, with ones. Kernel= np.ones((5, 5)) now kernel is array of 5x5.
9. Now filter out the background noise by using dilation technique.
   The bright area of the letter dilates around the black regions of the background.
   By using functions cv2.dilate() and cv2.erode().
10. After reducing the background noise apply Gaussian blur and threshold and show the threshold by using functions cv2.GaussianBlur(), cv2.threshold().
11. cv2.imshow("Thresholded", thresh)
12. Find the outline of your hand by using contour function. By using function cv2.contours().
13. Find the maximum contours.
14. Create the bounding rectangle around the contour. Cv2.boundingRect().
15. Find convex hull for the contour.
16. Draw contours for available contour and hull.
17. Find convexity (finger tips) defects.
18. By using the Cosine rule we found the angle of the far point from the start and end point.

    <u>cosine rule</u>

    $c=a2+b2+2abcos\theta$

    The cosine rule, also known as the law of cosines, relates all 3 sides of a triangle with an angle of a triangle.

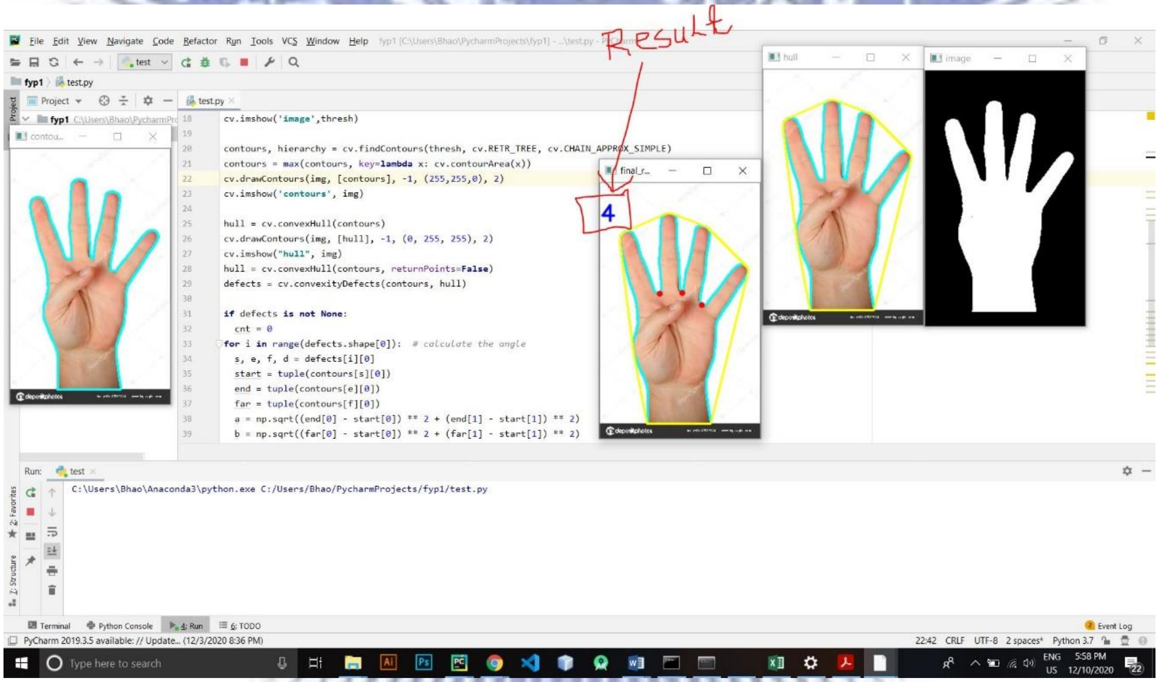
    With the help of cosine rule we found the angles. If the angle is less than 90, count it and make a blue circle over there. Or if the angle is greater than 90 than extract it.
19. Use loop to find convex points(finger tips) of all defects.
20. If angle is greater than 90 than draw a circle at the far point.
21. Print the number of fingers.
22. Show all the screens
23. Release and destroy all screens
24. end

CHAPTER #5 RESULT AND ANALYSIS

5.1Results (Testing & Outcomes of the project)

Result of approach one

- Convert static images to signs
- Create dataset for static images
- Create contours of images
- Background remove
- Detect hands in images and convert to text

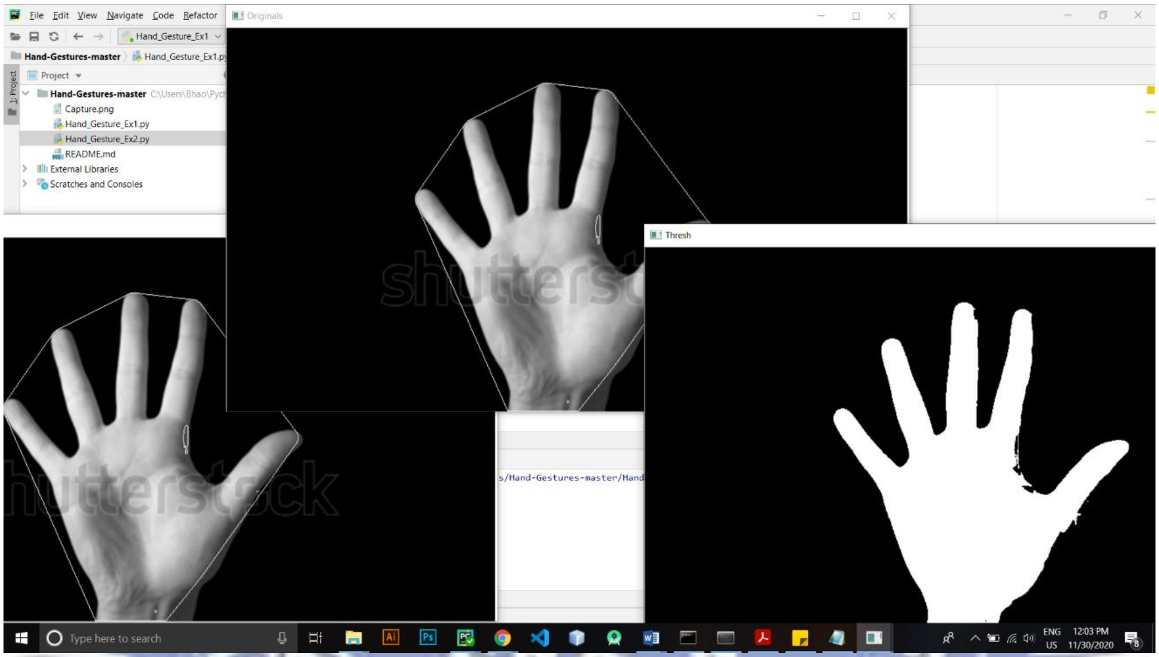

Result
4
File Edit View Navigate Code Refactor Run Tools VCS Window Help
fyp1 [C:\Users\Bhao\PycharmProjects\fyp1] - ...\test.py - PyCharm
fyp1 > test.py
Project
test.py
contou...
final_r...
hull
image
Run: test
Terminal
Python Console
4: Run
6: TODO
Event Log
PyCharm 2019.3.5 available: // Update... (12/3/2020 8:36 PM)
22:42 CRLF UTF-8 2 spaces* Python 3.7
Type here to search
ENG US 5:58 PM 12/10/2020

```
cv.imshow('image',thresh)

contours, hierarchy = cv.findContours(thresh, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
contours = max(contours, key=lambda x: cv.contourArea(x))
cv.drawContours(img, [contours], -1, (255,255,0), 2)
cv.imshow('contours', img)

hull = cv.convexHull(contours)
cv.drawContours(img, [hull], -1, (0, 255, 255), 2)
cv.imshow("hull", img)
hull = cv.convexHull(contours, returnPoints=False)
defects = cv.convexityDefects(contours, hull)

if defects is not None:
  cnt = 0
for i in range(defects.shape[0]):  # calculate the angle
  s, e, f, d = defects[i][0]
  start = tuple(contours[s][0])
  end = tuple(contours[e][0])
  far = tuple(contours[f][0])
  a = np.sqrt((end[0] - start[0]) ** 2 + (end[1] - start[1]) ** 2)
  b = np.sqrt((far[0] - start[0]) ** 2 + (far[1] - start[1]) ** 2)
```

```
C:\Users\Bhao\Anaconda3\python.exe C:/Users/Bhao/PycharmProjects/fyp1/test.py
```

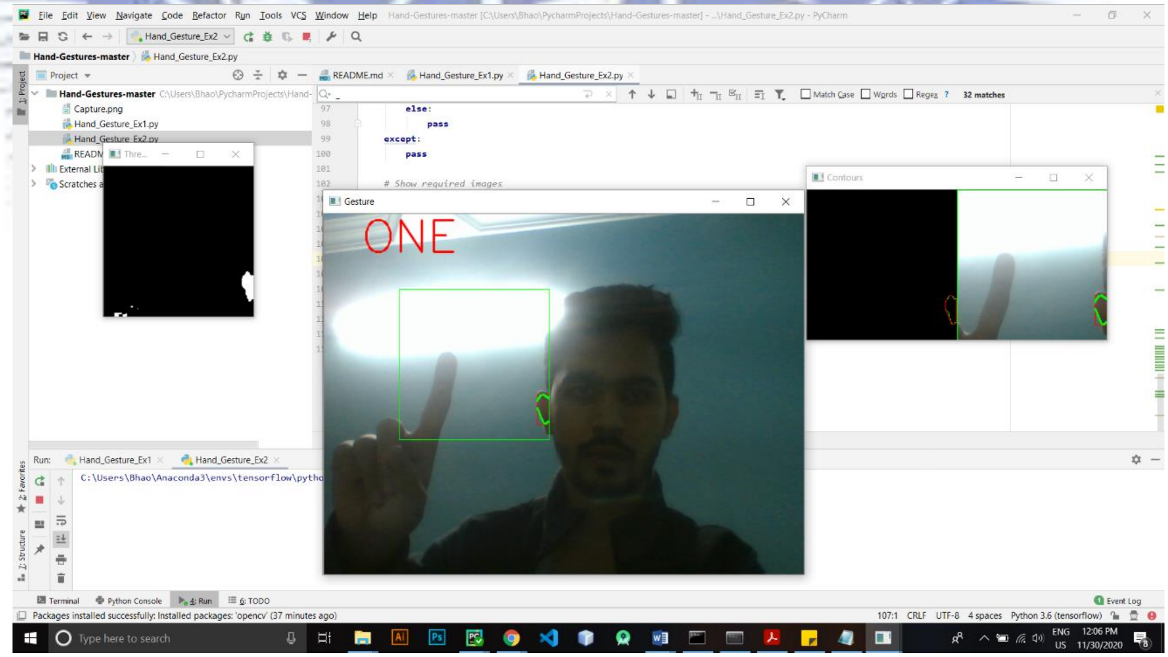
Figure 11 – Result Approach one

Result of approach two



```
s/Hand-Gestures-master/Hand
```

Figure 12 – Result Approach two



```
            else:
                pass
        except:
            pass

        # Show required images
```

```
C:\Users\Bhao\Anaconda3\envs\tensorflow\pytho
```

Packages installed successfully: Installed packages: 'opencv' (37 minutes ago)

Figure 13 – Result Approach Two

5.2Tribulations (Problems faced during Project Making)

- To reduce the communication barrier between the deaf and dumb community and a normal community we need an app to convert the signs into text/audio and text/audio into signs so we need all the signs in JPEG format and also its audio and text version. So for this large valuable dataset we approached deaf reach school.
- Image recognition.
- Tensor flow.
- Searching algorithms.
- Detecting fingers.
- Detecting angles.
- Background removal.

5.3Elucidation (Possible solution of the problems)

- Deaf reach school provided us dataset.
- We detected fingers using cosine rule technique
- By using convex hull technique, Gaussian blur technique we found angles between fingers to detect signs.

CHAPTER #6 CONCLUSION

6.1 Conclusion

Hand gestures are a powerful way for human communication, with lots of potential applications in the area of human computer interaction. Vision based hand gesture recognition techniques have many proven advantages compared with traditional devices. However, hand gesture recognition is a difficult problem and the current work is only a small contribution towards achieving the results needed in the field of sign language gesture recognition. This report presented a vision based system able to interpret isolated hand gestures from the Pakistan Sign Language (PSL).

Videos are difficult to classify because they contain both the temporal as well as the spatial features. We have used two different models to classify on the Spatial and temporal features.

We have used two approaches to solve our problem and both of the approaches only differ by the inputs given to the program as explained in the methodologies above.

6.2 Future Work

We wish to extend our work further in recognizing continuous sign language gestures with better accuracy. This method for individual gestures can also be extended for sentence level sign language. In addition, the current process uses cosine rule for angles detection to find number of fingers. For future work one can focus on detecting live signs with better accuracy.

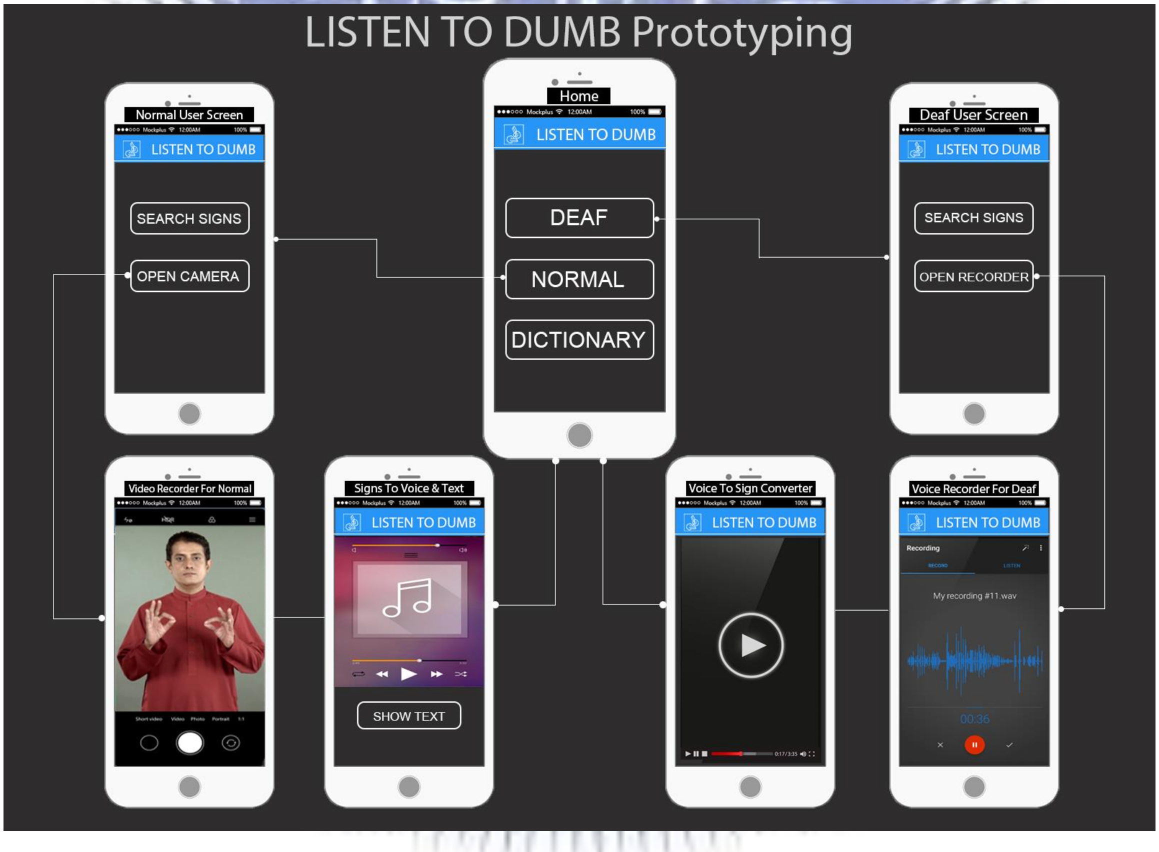


Figure 14 – Future Work