Sign Language to Speech Converter

Submitted in partial fulfilment of the requirements of the degree

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

By

Anvisha H. Pathak(5020136)
Niraj Patil(5020138)
Prashant Padhy(5020134)
Adarsh Jadhav(5020118)

Supervisor

Mrs. Smita Rukhande

Department of Information Technology

Fr. C. Rodrigues Institute of Technology

Vashi, Navi Mumbai - 400703

University of Mumbai

(AY 2021-22)

CERTIFICATE

This is to certify that the Mini Project entitled "**Sign Language to Speech Converter**" is a bonafide work of **Anvisha H. Pathak (5020136), Niraj Patil (5020138), Prashant Padhy (5020134) and Adarsh Jadhav (5020118)** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of "**Bachelor of Engineering**" in "**Information Technology**" .

(Mrs. Smita Rukhande)
Supervisor

(Dr. __________________)
Head of Department

(Dr.__________________)
Principal

Mini Project Approval

This Mini Project entitled “Sign Language to Speech Converter” by **Anvisha H. Pathak (5020136), Niraj Patil (5020138), Prashant Padhy (5020134) and Adarsh Jadhav (5020118)** is approved for the degree of **Bachelor of Engineering** in **Information Technology**.

Examiners

1………………………………………
(Internal Examiner Name & Sign)

2………………………………………
(External Examiner name & Sign)

Date:

Place:

ABSTRACT

Sign language is the only tool of communication for the person who is not able to speak and hear anything. Sign language is a boon for the physically challenged people to express their thoughts and emotion. In this work, a novel scheme of sign language recognition has been proposed for identifying the alphabets and gestures in sign language. With the help of help of OpenCV and mediapipe, we capture the gestures and .

Sign language is the way of communication for hearing impaired people. There is a challenge for common people to communicate with deaf people which makes this system helpful in assisting them. This project aims at implementing computer vision which can take the sign from the users and convert them into text in real time. The proposed system contains four modules such as: image capturing, preprocessing classification and prediction. By using image processing the segmentation can be done. Sign gestures are captured and processed using OpenCV python library. The captured gesture is resized, converted to grey scale image and the noise is filtered to achieve prediction with high accuracy. The handtracking and predication are done using mediapipe.

Project is developed using various libraries of Python [1] programming language such as opencv, mediapipe, etc. and flask framework[2].

ACKNOWLEDGEMENTS

The making of the Android Application “The Career Guide” involves contribution of many people. We owe a great thanks to many people who have helped and supported us during the course of our project. We express our gratitude to Dr. S. M. Khot, Principal of Fr. C. Rodrigues Institute of Technology, H.O.D. of IT department Dr. Vaishali Bodade, the Project Coordinator of IT department Prof. Chetana Badgujar and Prof. Kalpana Wani for extending their inevitable and valuable support to us.

We would take this opportunity to thank our guide Mrs. Smita Rukhande and Prof. Poonam Bari for guiding and correcting various documents of ours with attention and care. This synopsis could not have been completed without their knowledge and expertise.

We would also thank our Institution and our faculty members without whom this project would have been a distant reality. We also extend our heartfelt thanks to our families and well-wishers.

Yours sincerely,

Anvisha H. Pathak(5020136)
Prashant Padhy(5020134)
Niraj Patil(5020138)
Adarsh Jadhav(5020118)

LIST OF FIGURES

INDEX

Chapter - 1

INTRODUCTION

1.1 Introduction

The aim of this system is to elevate people with hearing disability and help them socialize with common people. It is a form of non-verbal communication. Sign language is the structured form as each gesture represents a unique element or a character. With the advent of advancement in science and engineering many researchers are working on different methodologies that could take the event of human computer interaction to a much higher extent. The computer is trained in such a way that it could translate the sign to text for static as well as dynamic frames.

The system is designed and implemented to recognize sign language gestures. The signs are captured using web cam and are pre-processed. In pre-processing stage, we use background subtraction to eliminate the background which makes this system to adapt to any dynamic background. The main difficulty while implementing in software based is that the image must be properly captured and filtered.

1.2 Motivation

Dumb people are usually deprived of normal communication with other people in the society. It has been observed that they find it really difficult at times to interact with normal people with their gestures, as only a very few of those are recognized by most people. Since people with hearing impairment or deaf people cannot talk like normal people so they have to depend on some sort of visual communication in most of the time.

The 2019 Indian census cites roughly 1.3 million people with “hearingimpairment”. In contrast to that numbers from India’s National Association of the Deaf estimates that 18 million people –roughly 1 per cent of Indian population are deaf. These statistics formed the motivation for our project. As these speech impairment and deaf people need a proper channel to communicate with normal people there is a need for a system . Not all normal people can understand sign language of impaired people. Our project hence is aimed at converting the sign language gestures into text that is readable for normal people.

1.3 Problem Statement and Objectives

- **Problem Statement :**

Speech impaired people use hand signs and gestures to communicate. Normal people face difficulty in understanding their language. Hence there is a need of a system which recognizes the different signs, gestures and conveys the information to the normal people. It bridges the gap between physically challenged people and normal people.

- **Objectives :**

The foremost aim of our system is to provide communication between common people and those with hearing aid without need of any specific color background, hand gloves or any sensors. Other systems used image dataset as such in '.jpg' format. But in our system the pixel values of each images are stored in csv file which reduces the memory requirement of the system. Also, the accuracy of prediction is high when csv dataset is used. The four modules of this system are Image capturing, Preprocessing, Classification and Prediction.

Chapter - 2

LITERATURE SURVEY

This chapter gives the basic information about the languages which were used to develop the project and the existing systems which were refereed before making the project. It also elaborates on the features and modules which are present in the application.

2.1 Survey of Existing System

The related work on this project shows that there have been several methods of implementing the system under different domains namely vision-based approach, glovebased approach, fuzzy logics, soft computing like neural network or using MATLAB etc. Vision-based approach requires camera to capture image in 2D or 3D format.

Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi [3] :
In this proposed system canny edge detection algorithm was used as the accuracy of this algorithm is better and the time consumed is also less. This algorithm is efficient in removing noise as well as detecting a clear image from the input for further processing. This algorithm results comparatively a low rate of error, edge points that are localized, and single edge point response. This system has been implemented using Python.

Indian Sign Language Translator Using Gesture Recognition Algorithm [4] :
This system is designed and built in such a way that the gestures are detected from the obtained input images and are converted to its standard format i.e. the converted hand gestures are represented in English language. The input that is in the form of video is segmented into single frames and each frame is then given to its pre-processing function. In order to reduce the unwanted regions and to enhance its performance each frame travels through multiple filters. Fourier description technique is adapted to extract hand gestures from the pre-processed frames and are finally stored in its desired repository.

Sign Language Recognition System Using Deep Neural Network [5] :
The proposed system was implemented using a 2- layer convolutional neural network (CNN) to predict sign language gestures made by the users. Two different models were used to classify and compare the accuracy of prediction. The two optimizers used for optimizing the output were SGD optimizer and Adam optimizer, in which Categorical Cross entropy cost function was used. The model was found to predict the gestures well even with noisy images and in different lighting conditions of datasets. Using SGD and Adam as two different optimizers, this system has

classified 6 different sign languages with an accuracy of 99.12% and 99.51% respectively. More accuracy is obtained when the Adam optimizer is used.

2.2 Limitation Existing System or Research gap

1. Hand gloves are used to predict the signs in the existing systems, but in our system, there is no need of hand gloves.
2. Also, sensors are required to detect the hand landmarks in existing systems, but our system doesn’t need those.

Chapter - 3

PROPOSED SYSTEM

Speech impaired people use hand signs and gestures to communicate. Normal people face difficulty in understanding their language. Hence there is a need of a system which recognizes the different signs, gestures and conveys the information to the normal people. It bridges the gap between physically challenged people and normal people. For this purpose, we have used OpenCV and mediapipe python libraries.

Following hand gestures are recognized by our system :

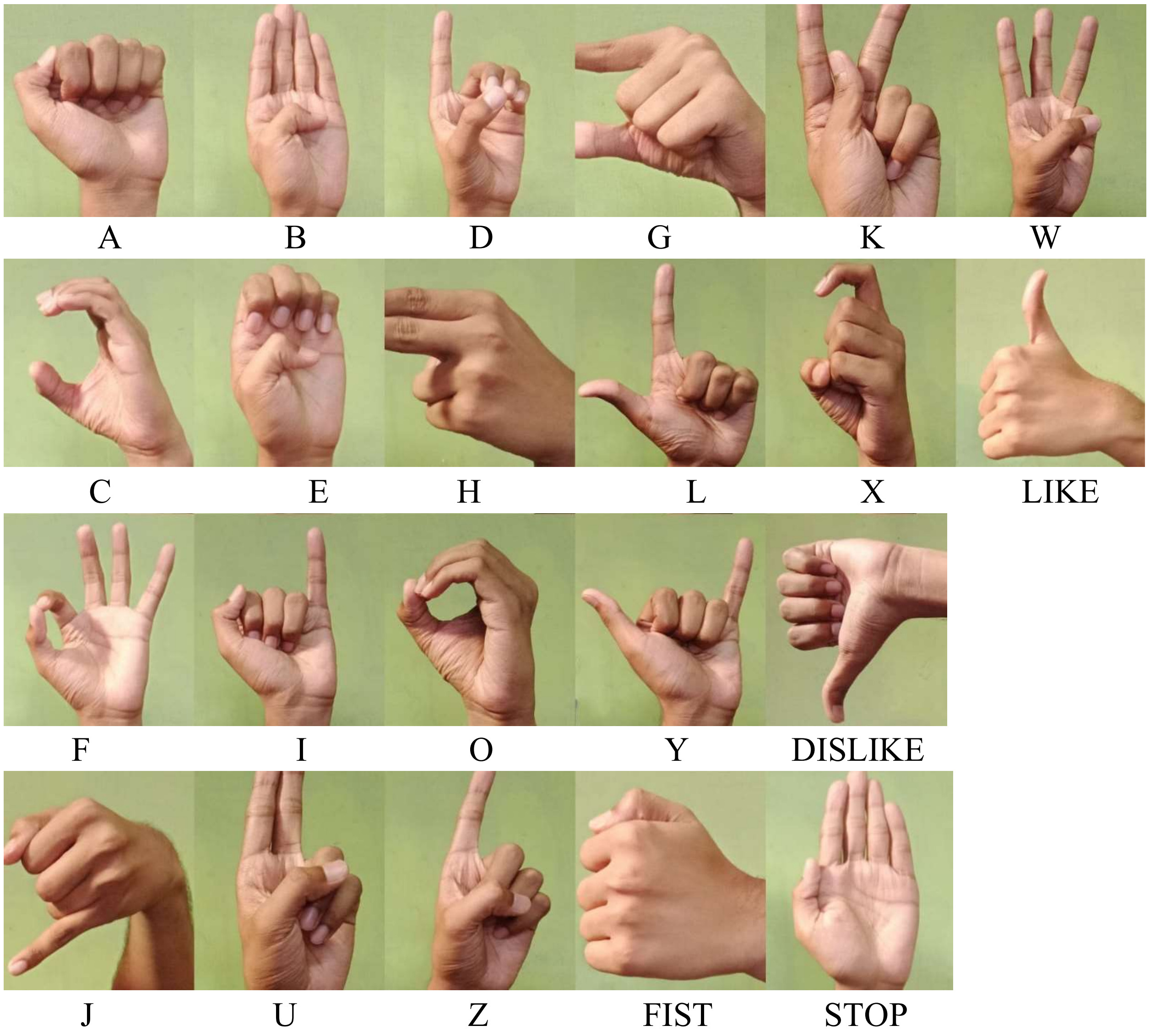


Figure 3 – Hand gestures for each sign

3.1 Architecture/Framework

Figure 3.2 shows the working of the system. User needs to show hand gestures in front of web cam. The hand gestures are then captured with the help of image capturing and processed with the help of image preprocessing. With the help of hand tracking algorithm, the landmarks of our hand are traced and then the name of the hand gesture is reflected on the screen.

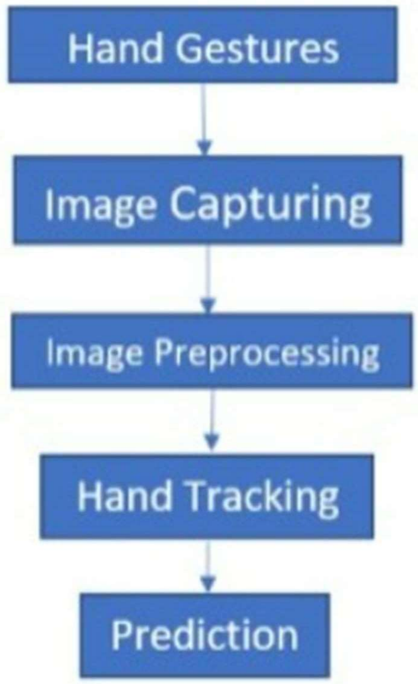


Figure 3.1 - Flowchart

3.2 Details of Hardware and Software

Following are the system requirements for implementing Sign language to speech converter :

- Hardware requirements:
    - Processor: Intel Core i3
    - RAM: 4 GB RAM or more
    - Display: having fps 30-40
    - Hard disk Drive: 5 GB of available disk space minimum

- Software requirements :

- Operating system: 64-bit Microsoft Windows 8/10/11
- Python (3.10)
- IDE (Pycharm)
- cv2 (openCV) (version 4.5.5.62)
- Mediapipe (version 0.8.9.1)

Chapter - 4

IMPLEMENTATION DETAILS

Module 1 : Image capturing:

Open Camera (Open CV):

```
import cv2
cap=cv2.VideoCapture(0)
```

cv2 is an Open CV package which gives access to Camera.If we want to use the default camera of Laptop we assign VideoCapture(0).For using Secondary Camera we assign VideoCapture(1).And if we wanna use any IP Address based external camera we have to provide Full link or IP address of that Camera.

```
import cv2

cap=cv2.VideoCapture(0)

while True:
   success,img=cap.read()

   img=cv2.flip(img,1)

   cv2.imshow('Image',img)

   cv2.waitKey(1)
```

cap.read() helps to read and display the capture Image.To acess our Video camera we use cv2.imshow() Method.It will show images in camera.
cv2.waitkey () is for wait.cv2.flip() method is used to flip the camera view.

Module 2 : Image processing:

Mediapipe:

```
import mediapipe as mp

mpHands=mp.solutions.hands

hands=mpHands.Hands()
mpDraw=mp.solutions.drawing_utils
```

```
while True:
  results=hands.process(img)
```

Mediapipe helps to apply Handtracking to our Image.mp.solutions.hands stored in mpHands to access the handsolutions from mediapipe.Inside of Hand solutions there is actual class Hands to access it mpHands.Hands().

Module 3 : Handtracking:

Hand Class:

```
def __init__(self,
           static_image_mode: bool = False,
           max_num_hands: int = 2,
           model_complexity: int = 1,
           min_detection_confidence: float = 0.5,
           min_tracking_confidence: float = 0.5)
```

To track our Hand use Hand algorithm with hands.process() method and store it in results.Drawing_utils is one of the drawing module.

```
if results.multi_hand_landmarks:
   for handlandmark in results.multi_hand_landmarks:

mpDraw.draw_landmarks(img,handlandmark,mpHands.HAND_CONNECT
IONS,
               mpDraw.DrawingSpec((255,0,0),2,2),
               mpDraw.DrawingSpec((0,255,0),4,2))
```

Now we try to access handlandmark of each hand from multi_Handlandmark.Drawing module is used to Highlight our handlandmarks on our image.draw_landmarks() module will detect all hand and from result we will get all multihandlandmarks and we will highlight tha handlandmarks on image.HAND_CONNECTIONS object joins or coonects all landmarks.Drawing Spec used to change color mp_draw.DrawingSpec((255,0,0),5,2).

```
finger_tips=[8,12,16,20]
thumb_tip=4

While True:
 if results.multi_hand_landmarks:
    for handlandmark in results.multi_hand_landmarks:
       lm_list=[]
       for id,lm in enumerate(handlandmark.landmark):
          lm_list.append(lm)
       for tip in finger_tips:
          x,y=int(lm_list[tip].x*w),int(lm_list[tip].y*h)
          cv2.circle(img,(x,y),15,(255,0,0),cv2.FILLED)
```

Now from handlandmarks to access actual landmarks create unique ID and landmarks from all this handlandmarks and enumerate this so it will provide all details of each landmarks.
Declare one array like finger_tips and store fingerstips we need.Store landmarks coordinates x*w and y*h as int value in x and y variable.
Cv2.circle() method is used to create circle and highlight specific tip pass position,radius and color.

Module 4 : Prediction:

Folded Finger Status:

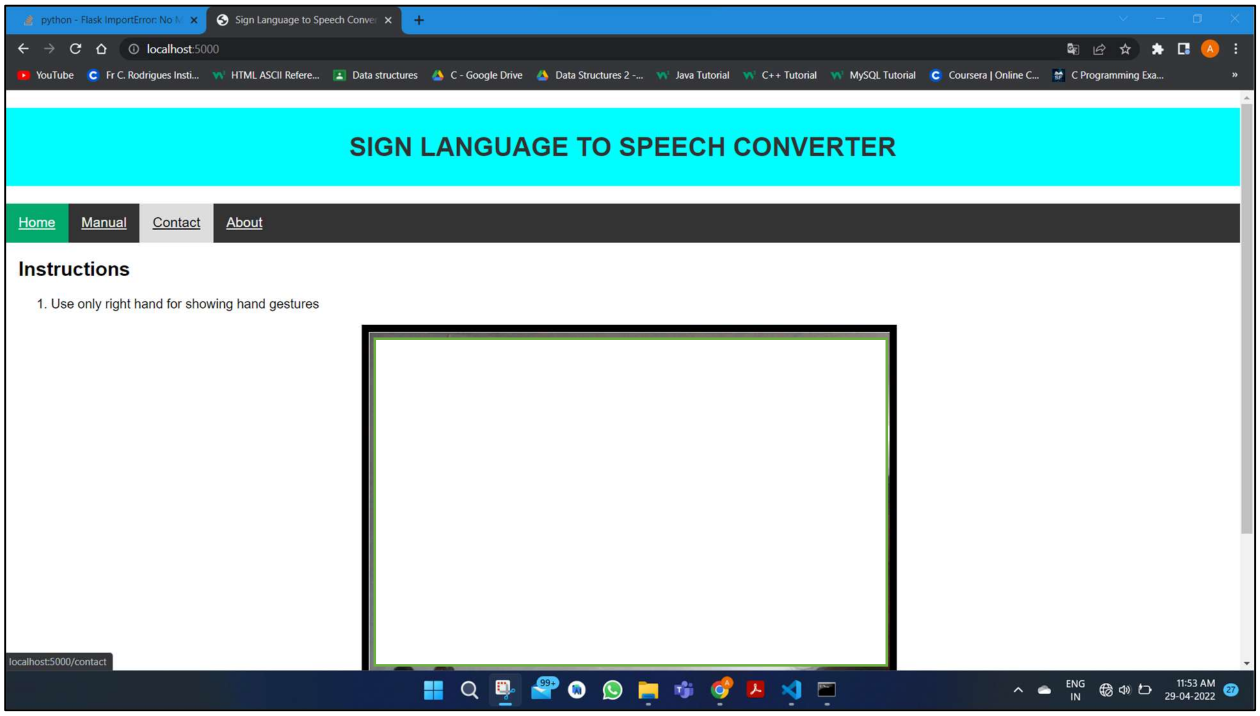
```
finger_fold_status = []
for tip in finger_tips:
   x,y=int(lm_list[tip].x*w),int(lm_list[tip].y*h)
   cv2.circle(img,(x,y),15,(255,0,0),cv2.FILLED)

   if lm_list[tip].x>lm_list[tip-3].x:
      cv2.circle(img, (x, y), 15, (255, 0, 0), cv2.FILLED)
      finger_fold_status.append(True)
   else:
      finger_fold_status.append(False)
print(finger_fold_status)
```

Now tips we assigned in an array. If tip along x-axis is greater than tip-3 then it will print the finger fold status as True else fingers are not folded.

Chapter - 5

EXPERIMENTAL RESULTS



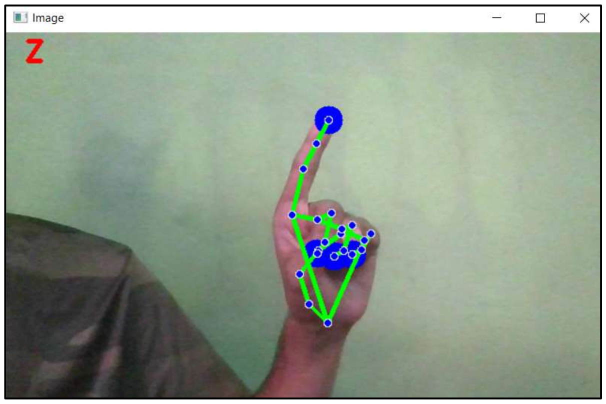
SIGN LANGUAGE TO SPEECH CONVERTER

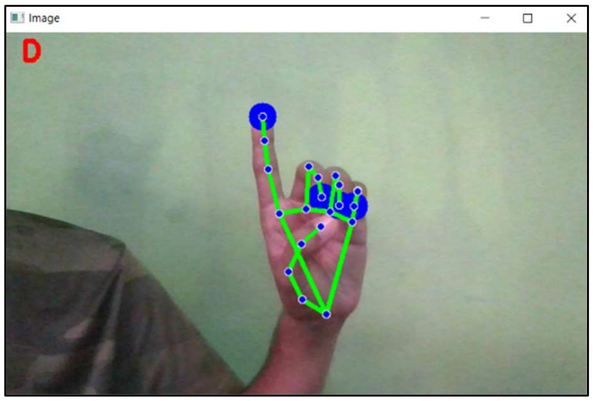
Instructions

1. Use only right hand for showing hand gestures

Figure – 5.1 : Front-end

Here are some of the results of our system:





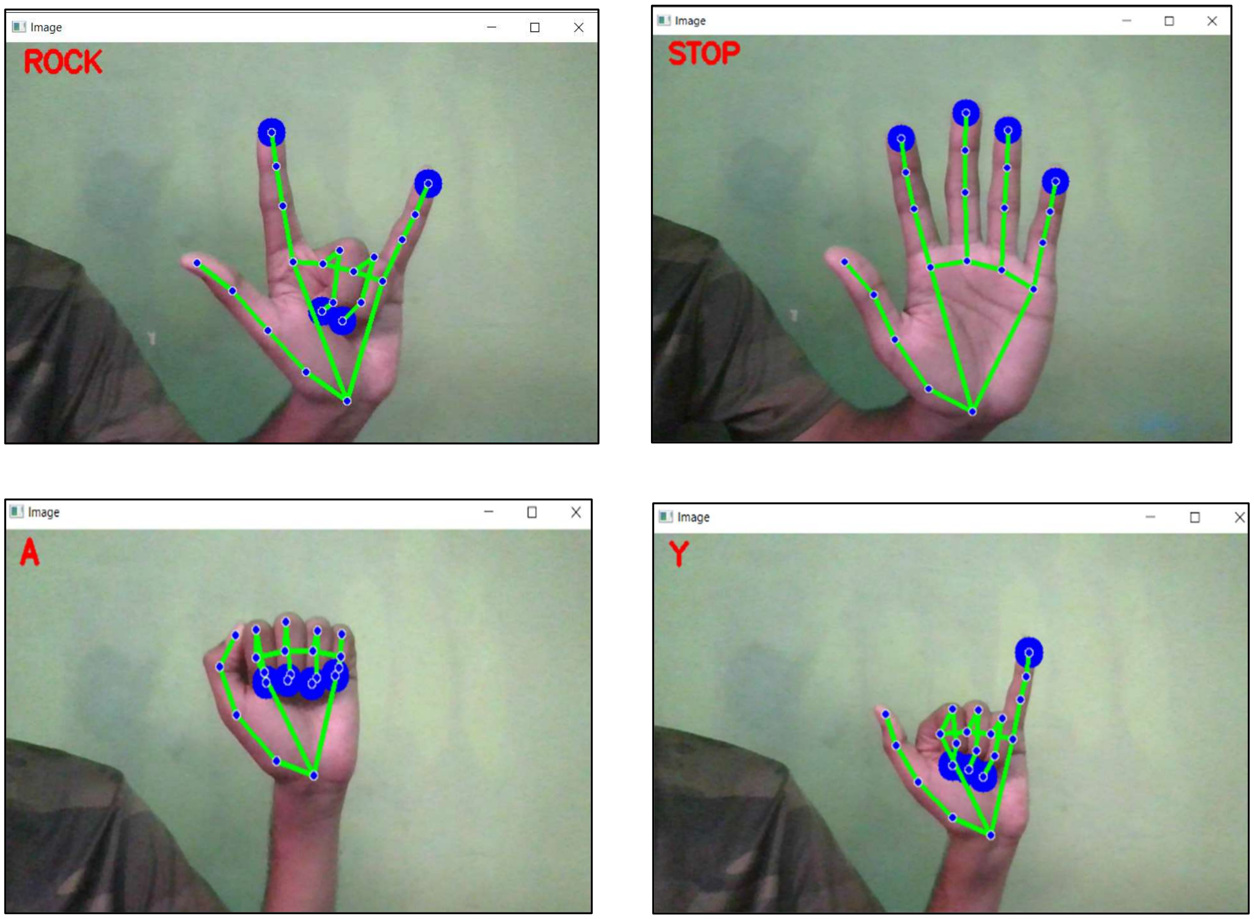


Figure 5.2 – Experimental results

Chapter - 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

- A Sign language to text converter is developed with the use of wireless technologies for people with hearing disabilities will improve the quality of life and the interaction of people with such disabilities in society.
- This system can be applied both in education, in enterprises, and in the daily life of people with disabilities as a real-time translator.
- On the whole, the solution aims to provide aid to those in need thus ensuring social relevance. The people can easily communicate with each other.
- The user-friendly nature of the system ensure that people can use it without any difficulty and complexity.
- The system is cost efficient and eliminates the usage of expensive technology.

6.2 Future Scope

- We can develop a model for ISL word and sentence level recognition.
- We are planning to achieve higher accuracy even in case of complex backgrounds.
- We are also thinking of improving the preprocessing to predict gestures in low light conditions with a higher accuracy.
- We are also planning to convert the text to audio to bridge the communication gap between blind, dumb and deaf people.

Citation	Reference link
[1]	https://www.javatpoint.com/python-tutorial
[2]	https://flask.palletsprojects.com/en/2.1.x/
[3]	Amitkumar Shinde, Ramesh Kagalkar, “ Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi,” National Conference on Advances in Computing.2017
[4]	Purva C. Badhe, Vaishali Kulkarni, “Indian Sign Language Translator Using Gesture Recognition Algorithm,” IEEE International Conference on Computer Graphics, Vision and Information Security.2018
[5]	Surejya Suresh, Mithun Haridas.T.P, Supriya M.H. “ Sign Language Recognition System Using Deep Neural Network,”5th International Conference on Advanced Computing & Communication Systems (ICACCS) 2019

Appendix : Code

Backend : Python code

```
import cv2
import mediapipe as mp

mpHands=mp.solutions.hands
hands=mpHands.Hands()
mpDraw=mp.solutions.drawing_utils
cap=cv2.VideoCapture(0)

finger_tips=[8,12,16,20]
thumb_tip=4

while True:
    success,img=cap.read()
    img=cv2.flip(img,1)
    h, w, c = img.shape
    results=hands.process(img)

    if results.multi_hand_landmarks:
        for handlandmark in results.multi_hand_landmarks:
            lm_list=[]
            for id,lm in enumerate(handlandmark.landmark):
                lm_list.append(lm)
            finger_fold_status = []
            for tip in finger_tips:
                x,y=int(lm_list[tip].x*w),int(lm_list[tip].y*h)
                cv2.circle(img,(x,y),15,(255,0,0),cv2.FILLED)

                if lm_list[tip].x>lm_list[tip-3].x:
                    cv2.circle(img, (x, y), 15, (255, 0, 0), cv2.FILLED)
                    finger_fold_status.append(True)
                else:
                    finger_fold_status.append(False)
            print(finger_fold_status)

            if all(finger_fold_status):
                if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y:
                    print("LIKE")
                    cv2.putText(img,"LIKE",(20,30),cv2.FONT_HERSHEY_SIMPLEX,1,(0, 0, 
255),3)
                if lm_list[thumb_tip].y > lm_list[thumb_tip-1].y > lm_list[thumb_tip-2].y:
                    print("DISLIKE")
                    cv2.putText(img, "DISLIKE", (20, 30), cv2.FONT_HERSHEY_SIMPLEX,
1, (0, 0, 255), 3)
                if lm_list[thumb_tip].y > lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y:
                    print("FIST")
                    cv2.putText(img, "FIST", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(0, 0, 255), 3)
            else:
                if lm_list[thumb_tip].x < lm_list[thumb_tip-1].x < lm_list[thumb_tip-2].x and
lm_list[8].y < lm_list[7].y and lm_list[12].y < lm_list[11].y and lm_list[16].y <
lm_list[15].y and lm_list[20].y < lm_list[19].y:
                    print("STOP")
                    cv2.putText(img, "STOP", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(0, 0, 255), 3)
                if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y < lm_list[7].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y < lm_list[19].y:
                    print("ROCK")
                    cv2.putText(img, "ROCK", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(0, 0, 255), 3)
                if lm_list[8].y > lm_list[5].y and lm_list[12].y > lm_list[9].y and lm_list[16].y
> lm_list[13].y and lm_list[20].y > lm_list[17].y and lm_list[thumb_tip].y <
lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y:
                    print("A")
                    cv2.putText(img, "A", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
                if lm_list[thumb_tip].x > lm_list[thumb_tip-1].x > lm_list[thumb_tip-2].x and
lm_list[8].y < lm_list[7].y and lm_list[12].y < lm_list[11].y and lm_list[16].y <
lm_list[15].y and lm_list[20].y < lm_list[19].y:
                    print("B")
                    cv2.putText(img, "B", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
                if lm_list[thumb_tip].y > lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y > lm_list[7].y < lm_list[5].y and lm_list[12].y > lm_list[11].y < lm_list[9].y
and lm_list[16].y > lm_list[15].y < lm_list[13].y and lm_list[20].y > lm_list[19].y <
lm_list[17].y:
                    print("C")
                    cv2.putText(img, "C", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
                if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y < lm_list[7].y and lm_list[12].y > lm_list[11].y < lm_list[9].y and
lm_list[16].y > lm_list[15].y < lm_list[13].y and lm_list[20].y > lm_list[19].y <
lm_list[17].y:
                    print("D")
                    cv2.putText(img, "D", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
```

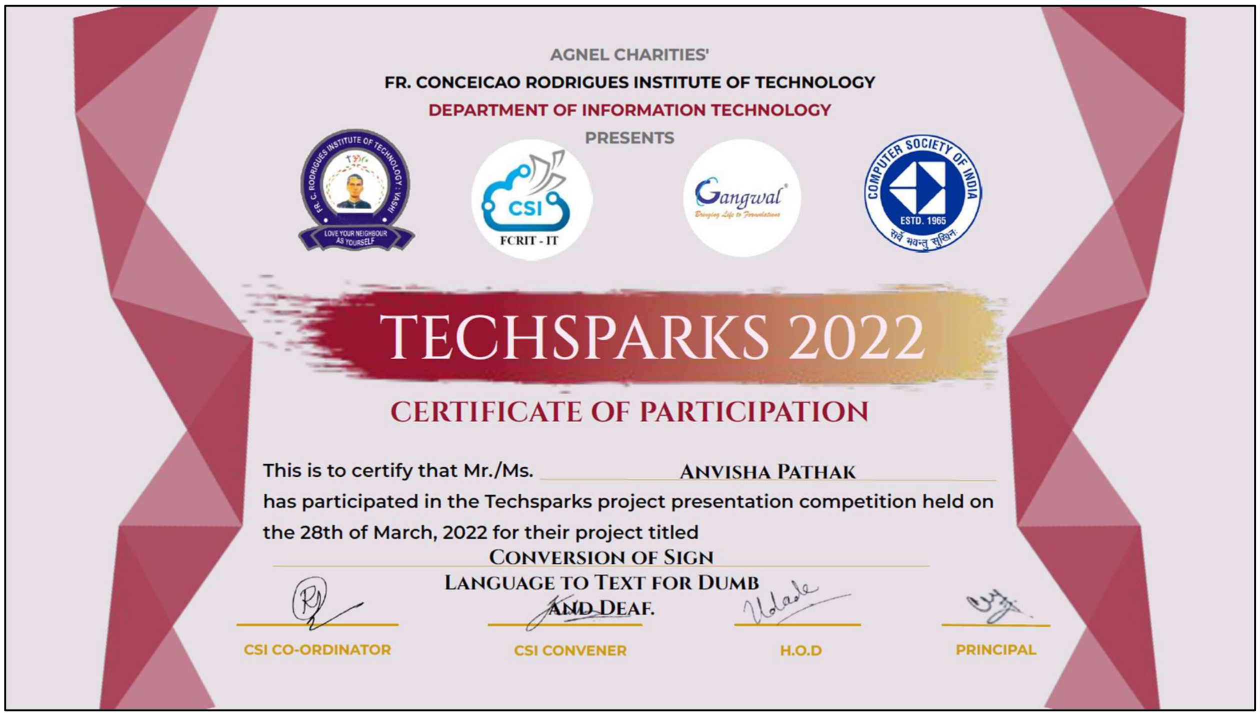
```
            if lm_list[thumb_tip].x > lm_list[thumb_tip-1].x > lm_list[thumb_tip-2].x and
lm_list[8].y > lm_list[7].y < lm_list[5].y and lm_list[12].y > lm_list[11].y < lm_list[9].y
and lm_list[16].y > lm_list[15].y < lm_list[13].y and lm_list[20].y > lm_list[18].y <
lm_list[16].y:
                print("E")
                cv2.putText(img, "E", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y > lm_list[6].y and lm_list[12].y < lm_list[11].y and lm_list[16].y <
lm_list[15].y and lm_list[20].y < lm_list[19].y:
                print("F")
                cv2.putText(img, "F", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x < lm_list[thumb_tip-1].x < lm_list[thumb_tip-2].x and
lm_list[8].x < lm_list[7].x and lm_list[12].x > lm_list[9].x and lm_list[16].x >
lm_list[13].x and lm_list[20].x > lm_list[17].x:
                print("G")
                cv2.putText(img, "G", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y > lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].x < lm_list[7].x and lm_list[12].x < lm_list[11].x and lm_list[16].x >
lm_list[13].x and lm_list[20].x > lm_list[17].x:
                print("H")
                cv2.putText(img, "H", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x > lm_list[thumb_tip-1].x > lm_list[thumb_tip-2].x and
lm_list[8].y > lm_list[5].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y < lm_list[19].y:
                print("I")
                cv2.putText(img, "I", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y > lm_list[thumb_tip-1].y > lm_list[thumb_tip-2].y and
lm_list[8].y > lm_list[5].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y > lm_list[19].y:
                print("J")
                cv2.putText(img, "J", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y < lm_list[7].y and lm_list[12].y < lm_list[11].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y > lm_list[17].y:
                print("K")
                cv2.putText(img, "K", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x < lm_list[thumb_tip-1].x < lm_list[thumb_tip-2].x and
lm_list[8].y < lm_list[7].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y > lm_list[17].y:
```

```
                print("L")
                cv2.putText(img, "L", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y > lm_list[7].y < lm_list[6].y and lm_list[12].y > lm_list[11].y < lm_list[10].y
and lm_list[16].y > lm_list[15].y < lm_list[14].y and lm_list[20].y > lm_list[19].y <
lm_list[18].y:
                print("O")
                cv2.putText(img, "O", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x > lm_list[thumb_tip-1].x > lm_list[thumb_tip-2].x and
lm_list[8].y < lm_list[7].y and lm_list[12].y < lm_list[11].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y > lm_list[17].y:
                print("U")
                cv2.putText(img, "U", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y < lm_list[7].y and lm_list[12].y < lm_list[11].y and lm_list[16].y <
lm_list[15].y and lm_list[20].y > lm_list[17].y:
                print("W")
                cv2.putText(img, "W", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x < lm_list[thumb_tip-1].x < lm_list[thumb_tip-2].x and
lm_list[8].y > lm_list[7].y < lm_list[6].y and lm_list[12].y > lm_list[11].y < lm_list[9].y
and lm_list[16].y > lm_list[13].y and lm_list[20].y > lm_list[17].y:
                print("X")
                cv2.putText(img, "X", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].y < lm_list[thumb_tip-1].y < lm_list[thumb_tip-2].y and
lm_list[8].y > lm_list[5].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y < lm_list[19].y:
                print("Y")
                cv2.putText(img, "Y", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)
            if lm_list[thumb_tip].x > lm_list[thumb_tip-1].x > lm_list[thumb_tip-2].x and
lm_list[8].y < lm_list[7].y and lm_list[12].y > lm_list[9].y and lm_list[16].y >
lm_list[13].y and lm_list[20].y > lm_list[17].y:
                print("Z")
                cv2.putText(img, "Z", (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0,
255), 3)

mpDraw.draw_landmarks(img,handlandmark,mpHands.HAND_CONNECTIONS,mpDr
aw.DrawingSpec((255,0,0),2,2),mpDraw.DrawingSpec((0,255,0),4,2))
    cv2.imshow('Image',img)
    cv2.waitKey(1)
```

Participation in Techsparks competition

1. Anvisha H. Pathak (5020136)



AGNEL CHARITIES'

FR. CONCEICAO RODRIGUES INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

PRESENTS

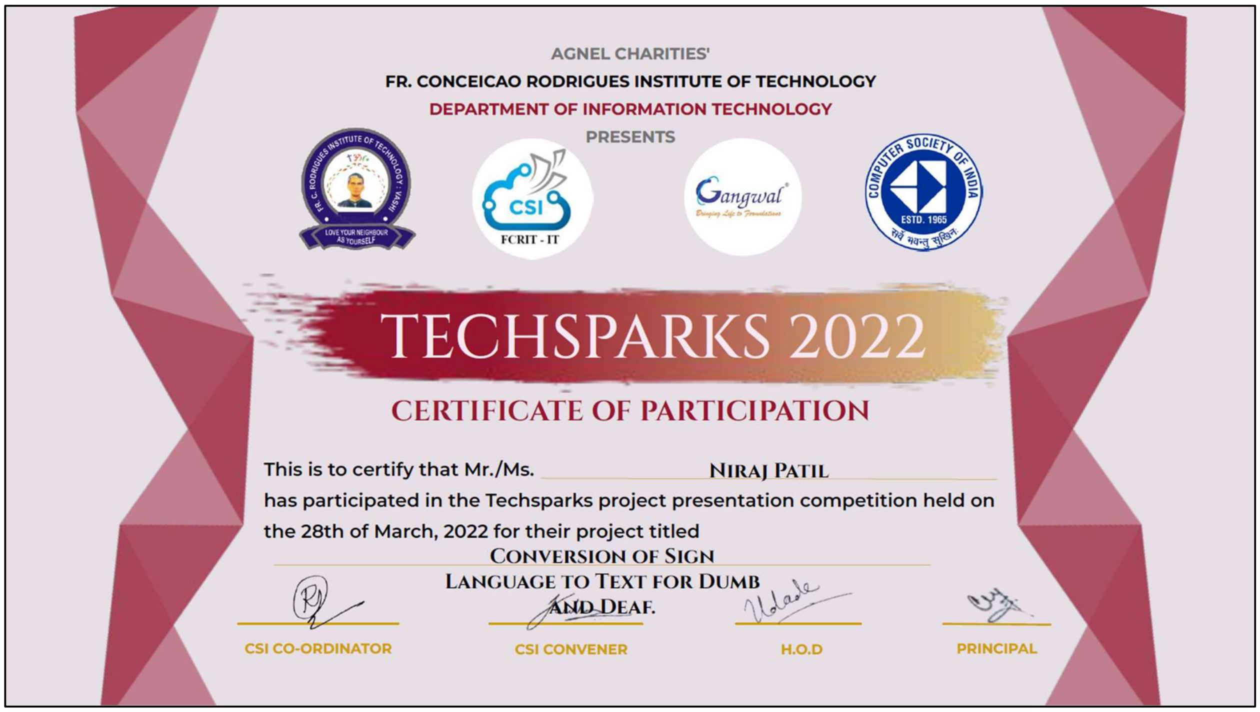
TECHSPARKS 2022

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./Ms. ANVISHA PATHAK has participated in the Techsparks project presentation competition held on the 28th of March, 2022 for their project titled CONVERSION OF SIGN LANGUAGE TO TEXT FOR DUMB AND DEAF.

CSI CO-ORDINATOR CSI CONVENER H.O.D PRINCIPAL

2. Niraj Patil (5020138)



AGNEL CHARITIES'

FR. CONCEICAO RODRIGUES INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

PRESENTS

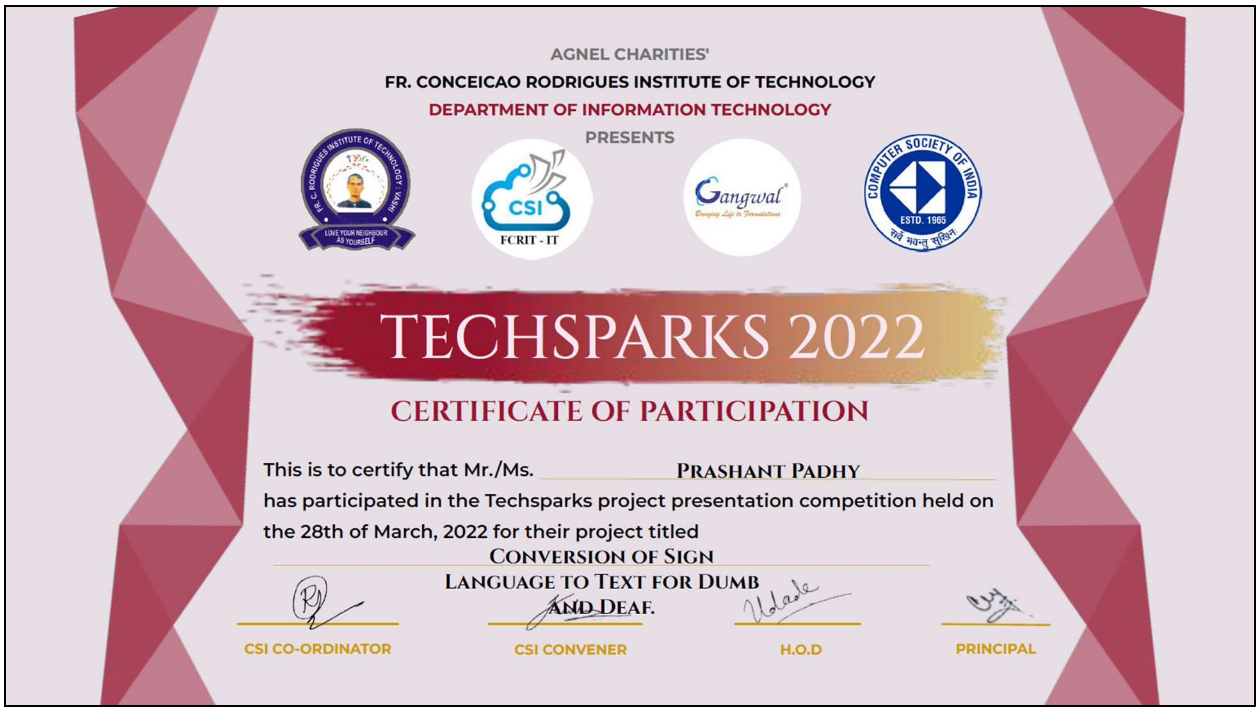
TECHSPARKS 2022

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./Ms. NIRAJ PATIL has participated in the Techsparks project presentation competition held on the 28th of March, 2022 for their project titled CONVERSION OF SIGN LANGUAGE TO TEXT FOR DUMB AND DEAF.

CSI CO-ORDINATOR CSI CONVENER H.O.D PRINCIPAL

3. Prashant Padhy (5020134)



AGNEL CHARITIES'

FR. CONCEICAO RODRIGUES INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

PRESENTS

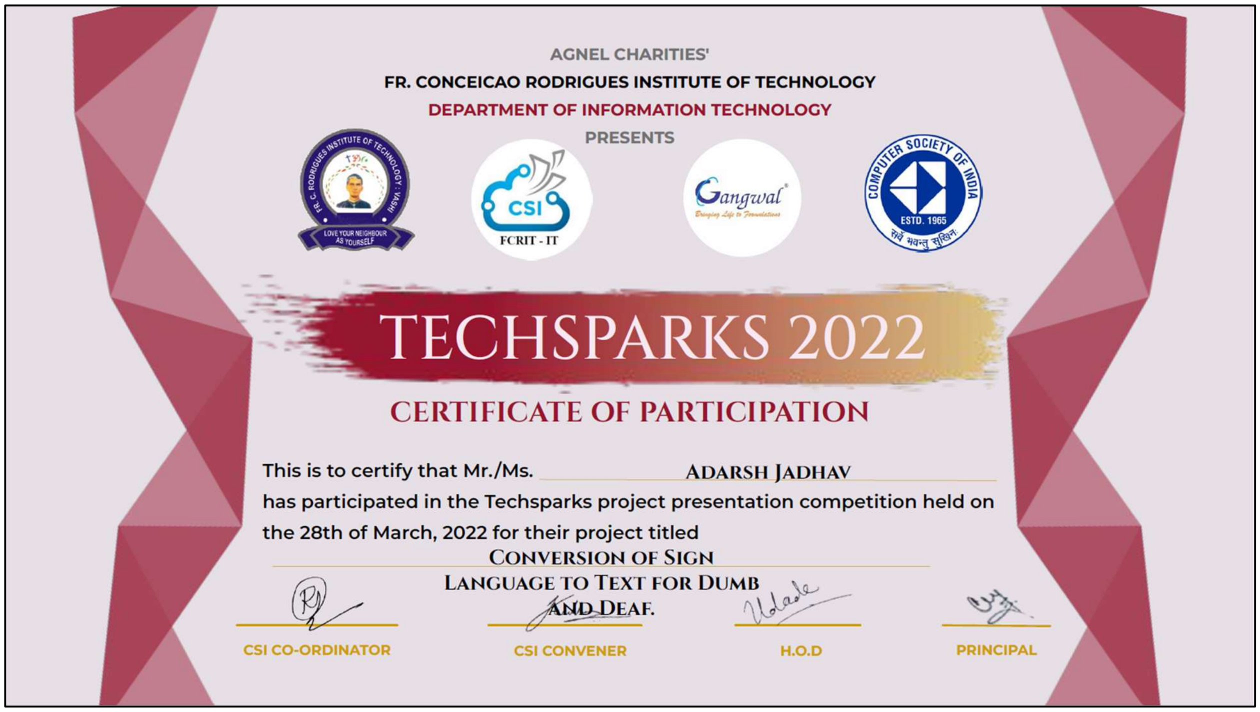
TECHSPARKS 2022

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./Ms. PRASHANT PADHY has participated in the Techsparks project presentation competition held on the 28th of March, 2022 for their project titled CONVERSION OF SIGN LANGUAGE TO TEXT FOR DUMB AND DEAF.

CSI CO-ORDINATOR CSI CONVENER H.O.D PRINCIPAL

4. Adarsh Jadhav (5020118)



AGNEL CHARITIES'

FR. CONCEICAO RODRIGUES INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

PRESENTS

TECHSPARKS 2022

CERTIFICATE OF PARTICIPATION

This is to certify that Mr./Ms. ADARSH JADHAV has participated in the Techsparks project presentation competition held on the 28th of March, 2022 for their project titled CONVERSION OF SIGN LANGUAGE TO TEXT FOR DUMB AND DEAF.

CSI CO-ORDINATOR CSI CONVENER H.O.D PRINCIPAL