APROJECTREPORT ON

Sign Language Translation into Text Using ML

Submitted in partial fulfilment for the award of the degree of

BACHELOR OFTECHNOLOGY

In

Computer Science and Engineering

By

D. LAHARI (21A81A0510)

K.S.L. SANDHYA (21A81A0524)

B. SOUDHAMANI (22A85A0501)

G.K.SAI SASHANK (22A85A0503)

M.NAVEEN (21A81A0532)

Under the Esteemed Supervision of

Dr. V.S. Naresh Ph.D.

Prof & Dean R&D

Department of Computer Science and Engineering (Accredited by N.B.A.)
SRI VASAVI ENGINEERING COLLEGE(Autonomous)
(Affiliated to JNTUK, Kakinada)
Pedatadepalli, Tadepalligudem-534101, A.P2023-24

SRI VASAVI ENGINEERING COLLEGE(Autonomous)

Department Of Computer Science and Engineering

Pedatadepalli, Tadepalligudem

Certificate

This is to certify that the Project Report entitled "**Sign Language Translation into Text Using ML**" submitted by **D. LAHARI (21A81A0510), K.S.L. SANDHYA (21A81A0524), B. SOUDHAMANI (22A85A0501), G.K.SAI SASHANK (22A85A0503), M. NAVEEN (21A81A0532)** for the award of the degree of Bachelor of Technology in the Department of Computer Science and Engineering during the academic year 2023-24.

Name of Project Guide
Dr. V.S Naresh Ph.D
Prof & Dean R&D

Head of the Department

Dr. D Jaya Kumari M. Tech., Ph.D.
Professor & HOD.

External Examiner

DECLARATION

We hereby declare that the project report entitled “**Sign Language Translation into Text using ML”** submitted by us to Sri Vasavi Engineering College(Autonomous), Tadepalligudem, affiliated to JNTUK Kakinada in partial fulfilment of the requirement for the award of the degree of B.Tech in Computer Science and Engineering is a record of Bonafide project work carried out by us under the guidance of **Dr.V.S.Naresh, Ph.D(Prof & Dean R&D)** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or University.

Project Associates

D. LAHARI (21A81A0510)
K.S.L. SANDHYA (21A81A0524)
B. SOUDHAMANI (22A85A0501)
G.K.SAI SASHANK (22A85A0503)
M. NAVEEN (21A81A0532)

ACKNOWLEDGEMENT

First and foremost, we sincerely salute to our esteemed institute **SRI VASAVI ENGINEERING COLLEGE,** for giving us this golden opportunity to fulfill our warm dream to become an engineer.

Our sincere gratitude to our project guide **Dr. V.S. Naresh, Ph. D (Prof &Dean R&D)** Department of Computer Science and Engineering**,** for his timely cooperat- -ion and valuable suggestions while carrying out this project.

We express our sincere thanks and heartful gratitude to **Dr. D. Jaya Kumari**, Professor & Head of the Department of Computer Science and Engineering, for permitting us to do our project.

We express our sincere thanks and heartful gratitude to **Dr. G.V.N.S.R. Ratnakara Rao**, Principal, for providing a favorable environment and supporting us during the development of this project.

Our special thanks to the management and all the teaching and non-teaching staff members, Department of Computer Science and Engineering, for their support and cooperation in various ways during our project work. It is our pleasure to acknowledge the help of all those respected individuals.

We would like to express our gratitude to our parents, friends who helped to complete this project.

Project Associates

D. LAHARI (21A81A0510)
K.S.L. SANDHYA (21A81A0524)
B. SOUDHAMANI (22A85A0501)
G.K.SAI SASHANK (22A85A0503)
M. NAVEEN (21A81A0532)

ABSTRACT

Nonverbal communication encompasses various methods of communication, including hand gestures or signs. Sign language is a collection of these hand gestures that enable communication, and it is commonly used by deaf-mute individuals to communicate with each other and others. However, it can be challenging for those who are not familiar with sign language to understand the gestures used.

As technology continues to advance, Gesture Recognition and Pattern Recognition have become popular fields of research. Hand gestures are an essential part of nonverbal communication and play a vital role in daily life.

To bridge the communication gap between deaf-mute individuals and others, we propose a project that involves creating a system for character classification in Indian Sign Language (ISL) or American Sign Language (ASL) using machine learning. This systemaims to utilize the power of machine learning to accurately identify and classify different sign language characters, enabling better communication between deaf-mute individuals and the general population.

TABLE OF CONTENTS

S.NO	TITLE	PAGE NO
	ABSTRACT	iii
1	**INTRODUCTION**	6-8
	1 Introduction	6
	1.2 Motivation	6-7
	1.3 Scope	8
	1.4 Project Outline	8
2	**LITERATURE SURVEY**	**9-10**
3	**SYSTEM STUDY AND ANALYSIS**	**11-15**
	3.1 Problem Statement	12
	3.2 Existing System	12
	3.3 Limitations of the Existing System	13
	3.4 Proposed System	13
	3.5 Advantages of Proposed System	13
	3.6 Functional Requirements	14
	3.7 Non-Functional Requirements	14-15
	3.8. System Requirements	15
	3.8.1 Software requirements	15
	3.8.2 Hardware Requirements	15
4	**SYSTEM DESIGN**	**16-20**
	4.1 System Architecture Design	17
	4.2 Design	18
	4.3 Diagrams using UML Approach	18-20

5	**TECHNOLOGIES**	**21-25**
	1. Python	22
	1. About Python	22
	2. History of Python	22-24
	3. Advantages of Python	24
	4. Disadvantages of Python	25
6	**IMPLEMENTATION**	**26-27**
	1. Implementation Steps	27
	2. Methodology	27
7	**TESTING**	**28-31**
	1. Test Objectives	29
	2. Testing Strategies	29
	3. Unit Testing	29-30
	4. Integration Testing	30
	5. Functional Testing	30-31
	6. System Testing	31
	7. Acceptance Testing	31
8	**SCREENSHOTS**	**32-42**
9	**CONCLUSION AND FUTURE WORK**	**43-44**
	REFERENCES	**45**

CHAPTER 1

INTRODUCTION

1. Introduction

Now a day’s sign language has much attention as the count of deaf and dumb was increasing over the years. Sign Language is the vision based natural language which is used for communication among people with low or no hearing sense and dumb that engages signs with the combination of hand movements, hand gestures. A proper recognition of these hand gestures is important while a non-speaking person is communicating with the person who able to speak as the normal person who don’t know the sign language. To have a smooth communication between people we can interpret the sign language translator. Finally, our project aims to take the hand gestures as an input through webcam and to produce text to have an effective communication between the deaf and dumb people and normal people.

2. Motivation

Communication is one of the basic requirements for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Extensive work has been done on American sign language recognition but Indian sign language differs significantly from American sign language. ISL uses two hands for communicating (20 out of 26) 7 whereas ASL uses single hand for communicating. Using both hands often lead to obscurity of features due to overlapping of hands. In addition to this, lack of datasets along with variance in sign language with locality has resulted in restrained efforts in ISL gesture detection. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using Indian sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with outer world, but also provide a boost in developing autonomous systems for understanding and aiding them.

3. Scope

Sign Language is a set of Languages that use predefined actions and movements to convey a message. These Languages are primarily developed to aid deaf and other verbally challenged people. They use a simultaneous and precise combination of movements of hands, orientation of hands, hand shapes etc. Different regions have different sign languages like American Sign Language, Indian Sign Language etc. We mainly focus on Indian Sign Language in this project.

4. Project Outline

Chapter 1: Introduction

Chapter 2: Literature Survey

Chapter 3: System Study and Analysis

Chapter 4: System Design

Chapter 5: Technologies

Chapter 6: Implementation

Chapter 7: Testing

Chapter 8: Screenshots

Chapter 9: Conclusion and Future Work

Chapter 10: References

CHAPTER 2

LITERATURE SURVEY

- **Title:** A Review on Indian Sign Language Recognition

  Author: Anuja Nair, Bindu V

  Mainly three steps are involved in sign language recognition-preprocessing, feature extraction and classification. The important classification methods used for recognition are Artificial Neural Networks (ANN), Support Vector Machine (SVM), Hidden Markov Models (HMM) etc.

- **Title:** Intelligent Sign Language Recognition Using Image Processing

  Author: Sawant Pramada, Archana Vaidya

  The project uses image processing system to identify, especially English alphabetic sign language used by the deaf people to communicate. The basic objective of this project is to develop a computer based intelligent system that will enable dumb people significantly to communicate with all other people using their natural hand gestures.

CHAPTER 3

SYSTEM STUDY AND ANALYSIS

3.1 Problem Statement

Generally Normal people face difficulty in understanding the language of Deaf and Dumb People. Hence there is a need of a system which recognizes the different signs, gestures and conveys the information to the normal people. Also, there is a need for voice impaired people to be able to understand normal people's language. Hence dual mode of communication is needed. It bridges the gap between physically challenged people and normal people. To design a real time software system that will be able to recognize ASL hand-gestures using Machine learning techniques. This project aims to predict the 'alphabetic' gesture of the ASL system and also convert the human speech into signs. But instead of using high-end technology like gloves or Kinect, we aim to solve this problem using state of the art computer vision and machine learning algorithms.

3.2 Existing System

In existing system, the module was developed for dumb person using flex sensor, there user hand is attached with the flex sensors. On this module the flex sensor reacts on the bend of each finger individually. By taking that value controller starts to react with speech, each flex sensor holds a unique voice stored in APR Kit and for each sign it will play a unique voice. And in other existing systems, the work is done only for some alphabets and not for the words or sentences, andaccuracy obtained is very low. Though existing systems identify sign language with sufficient accuracy, this proposal incorporates the recognition from a live video feed. As a result, it provides more interactivity than existing systems.

3. Limitations

- The system doesn’t give correct results in low light conditions.
- One of the major problems of the existing system is the user should always carry the hardware with him.
- Users cannot do any other work with flex sensors on fingers and also sensors should be placed straight.
- The system unable to identify the signs in case of movements in gestures.

4. Proposed System

In the proposed system of the sign language translation, firstly we need to perform the gestures in front of the webcam so that it can capture the images and the system processes those images. After that classification is performed. Finally, we will get the final text as output of sign language. The proposed system consists of a camera which captures an image.

5. Advantages of Proposed System

- When comparing with existing system user can give more signs.
- Easy to use Interface (Graphical User Interface)
- Flexible.
- The module provides dual mode of communication which helps in easy interaction between the normal people and disabled.

6. Functional Requirements

- **Image acquisition:** The image acquisition of signer, i.e., the person conveying in the sign language, can be obtained by using a camera. A camera sensor is needed in order to capture the features/ gestures of the signer.
- **Image Preprocessing:** As these images are not taken in a controlled lightening environment also images are taken with a digital camera, they have different sizes and different resolutions. So, in image pre-processing is required.
- **Classification:** gesture input that is made by the hand to be recognized and compared to the already available dataset which is sent to the model to train the system and classify the gesture accurately using various algorithms.

7. Non- Functional Requirements

Non-functional requirements of our project are:

- **Usability** is a quality attribute used to access how easy the interface is to use. GUI will be provided for each and every functionality through which navigation will be easier.
- **Flexibility** means new modules can be easily integrated to our system without disturbing the existing modules or modifying the logical database schema of the existing application.
- **Integrity** means doing the right thing in a reliable way. Data integrity is a fundamental component of security.

- **Performance** is also a major non-functional requirement. The performance constraints specify the timing characteristics of the software Performing ISL should be used by a single person.

3.8 System Requirements

In the making of this project, software and hardware tools are very important and essential. The system cannot be developed successfully without these software and hardware requirements. In order to complete the project, the facilities from hardware and software must be used.

1. Software Requirements

- Operating System: Windows 10
- SDK: Open CV, TensorFlow
- Language: Python

2. Hardware Requirements

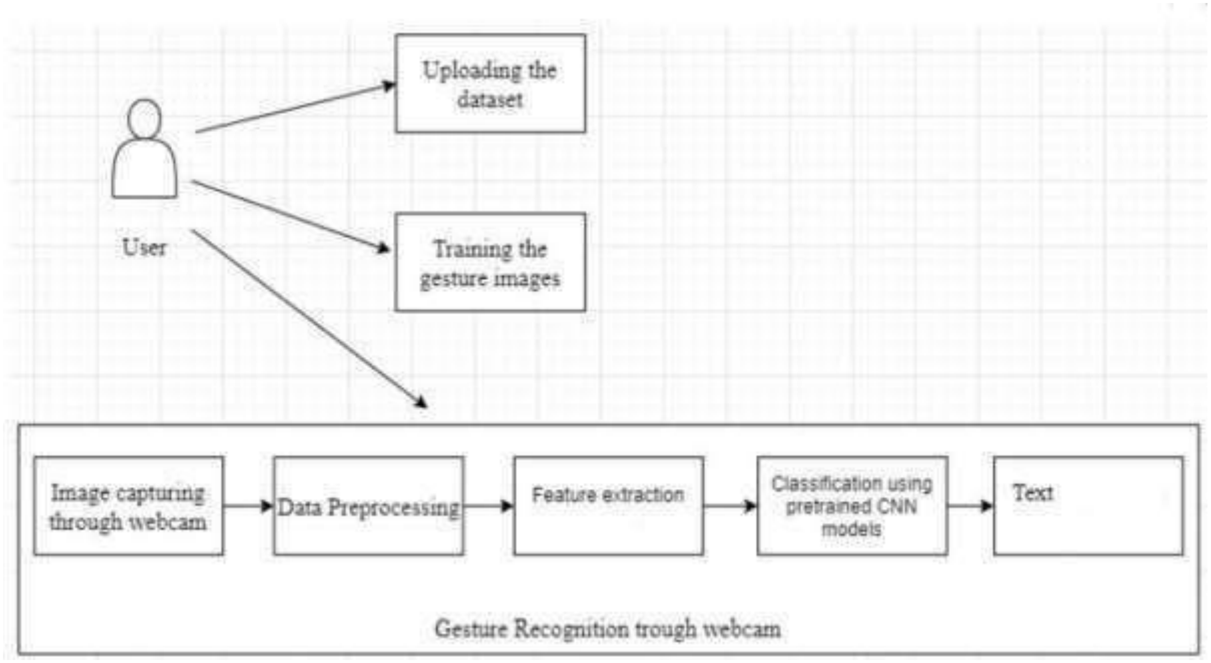
- Camera: Good quality (Minimum 5MP)
- Processor: core I3 and above
- RAM: Minimum 8GB or higher
- GPU: Minimum 2GB

CHAPTER 4

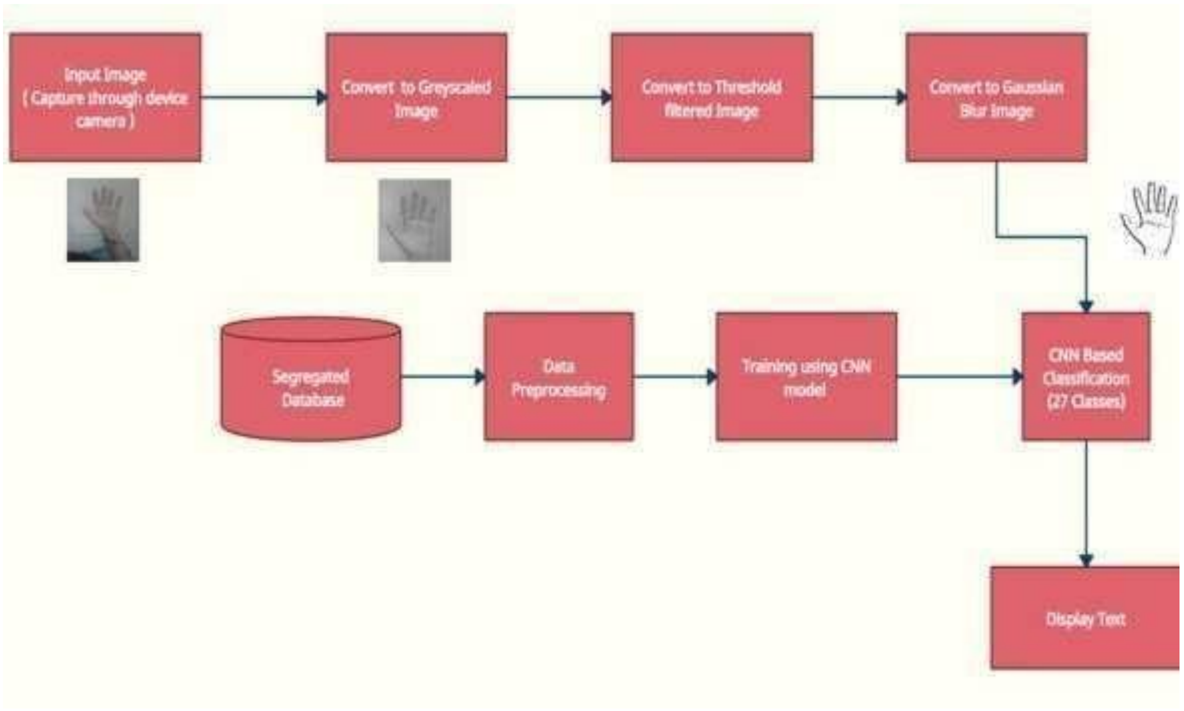
SYSTEM DESIGN

4.1 System Architecture Design

System Architecture diagrams helps developers and system designers to outline the software and to check whether the software meets their client needs. Architecture diagram also known as blueprint of the software which contains the components and their relationships. A system architecture description is a formal description and representation of a system, organized in a way that support reasoning about the structures and behaviors of the system. The below diagram represents the system architecture for sign language translation to text.

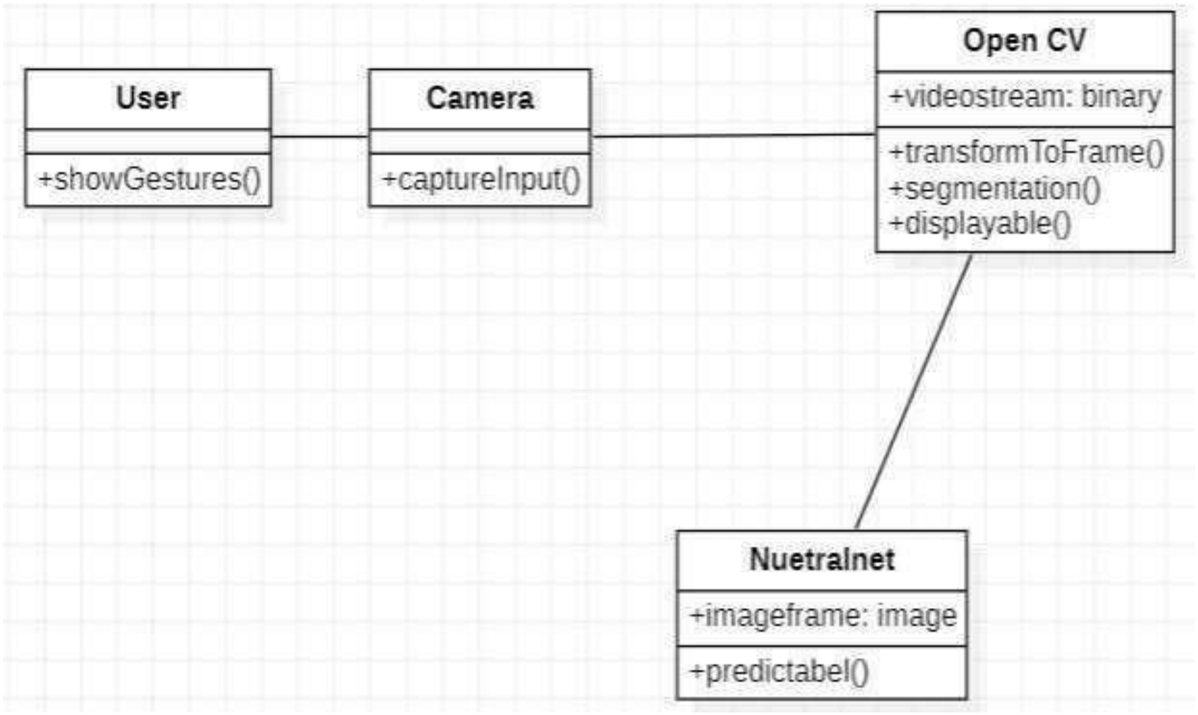

User
Uploading the dataset
Training the gesture images
Image capturing through webcam
Data Preprocessing
Feature extraction
Classification using pretrained CNN models
Text
Gesture Recognition trough webcam

4.2 Design

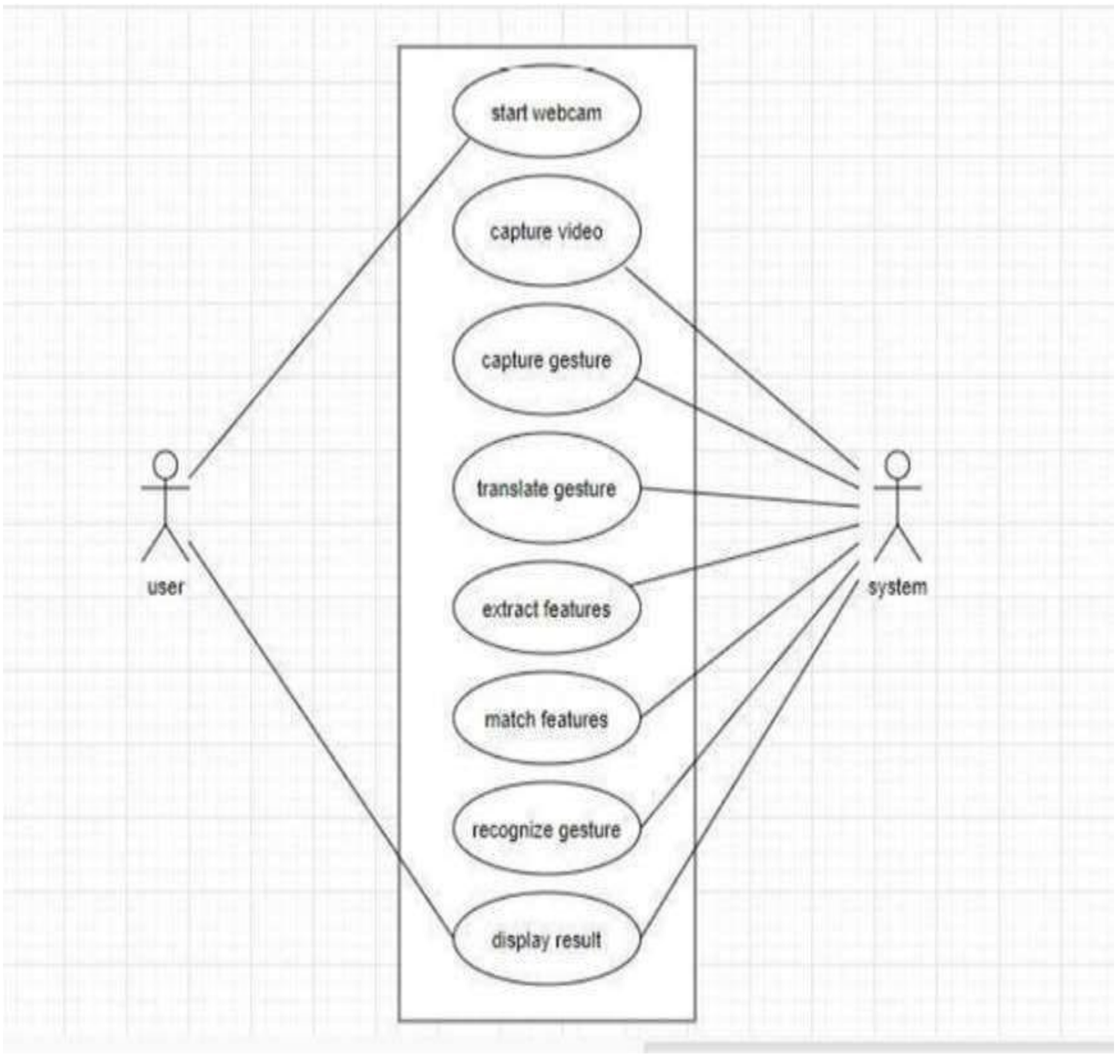

Input Image
(Capture through device camera)
Convert to Greyscaled Image
Convert to Threshold filtered Image
Convert to Gaussian Blur Image
Segregated Database
Data Preprocessing
Training using CNN model
CNN Based Classification (27 Classes)
Display Text

4.3 Diagrams using UML Approach

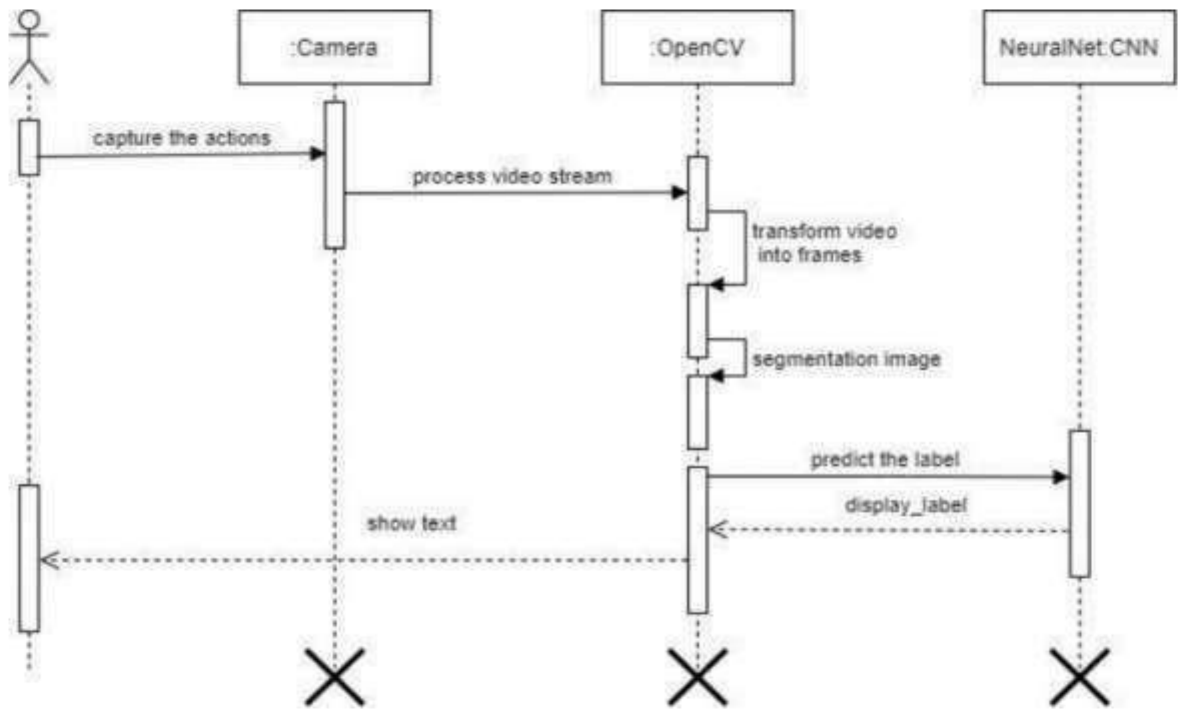
Class Diagram



Use case Diagram



Sequence Diagram



CHAPTER 5

TECHNOLOGIES

1. Python

1. About Python

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber…etc.
- Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.
- Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.
- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to Pearland PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

5.1.2 History of Python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general- purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operatingsystem. If we are talking aboutABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general- purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). I created a simply virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation

For statement grouping instead of curly braces or begin-end blocks. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum Wiskunde Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted Everything I learned during that project and to the people who workedon it. "Later on in the same Interview, Guido van Rossum continued: ‘I remembered all my experience and some of my frustration with ABC. I decidedto try to design a simple scripting language that possessed some of ABC's betterproperties, but without its problems’”.

So, I started typing. I created a simply virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers.

3. Advantages of Python

- **Less Coding:** Almost all the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is there as on that many people suggest learning Python to beginners.
- **Affordable:** Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support. The 2019 GitHub annual survey showed us that Python

  has overtaken Java in the most popular programming language category.
- **Python is for Everyone:** Python code can run on any machine whether it is Linux, Macor, Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.
- **Portable:** When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and

  you can run it anywhere. This is called Write Once Run Anywhere

(WORA). However, you need to be careful enough not to include any system-dependent features.

5.1.4 Disadvantages of Python

- **Speed Limitations:** We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.
- **Weak in Mobile Computing and Browsers:** While it serves as an excellent server- side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smart phone-based applications. One such application is called Carbon Nelle. There as on it is not so famous despite the existence python is that it isn’t that secure.
- **Design Restrictions:** As you know, Python is dynamically typed. This means that you don’t need to declare the type of variable while writing the code. It uses duck- typing. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.
- **Underdeveloped Database Access Layers:** Compared to more widely used technologies like JDBC (Java Data Base Connectivity) and ODBC (Open Data Base Connectivity), Python’s data base access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.
- **Simple:** No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

CHAPTER 6

IMPLEMENTATION STEPS

1. Implementation Steps

Implementation Steps

- Preprocessing images
- Making Datasets using images collected
- Feature extraction from images
- Train the model using various machine learning and deep learning algorithms
- Classify the user captured video images into signs (alphabets and digits)
- Capture user's voice and convert them into sign language sign.

6.2 Methodology

There are several steps involved in the methodology of sign language Translation to text like gesture recognition. To recognize the sign language, firstlywe need to take the input image through webcam then the image needs to be processed. Then we can extract the features from the processed image. Finally, the system recognizes the gestures, and we will get the needed result.

Methodology Modules

- Uploading and training the dataset
- Data Preprocessing
- Feature Extraction
- Classification
- Predicting the gestures

CHAPTER 7

TESTING

1. Test Objectives

- All field entries must work properly.
- The Entry screen, responses must not be delayed.

2 Testing Strategies

Different levels of testing are used in the test process; each level of testing aims to test different aspects of the system.

1.The first level is unit testing. In this testing, individual components are tested to ensure that they operate correctly. It focuses on verification efforts.

2.The second level is Integration testing. It is a systematic technique for constructing the program structure. In this testing, many tested modules are combined into the subsystem which is then tested. The good here is to see if the modules can be integrated properly.

3.Third level is Integration testing. System testing is actually a series of different tests whose primary purpose is to fully exercise computer-based system. These tests fall outside scope of software process and are not conducted solely by software engineer

1. Unit Testing

- Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application.
- It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a

specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

2. Integration Testing

- Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

3. Functional testing

- Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

- Functional testing is centered on the following items:
- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

- Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined**.**

4. System Testing

- System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points

5. Acceptance Testing

- Acceptance Testing is the last phase of Software testing performed after system Testing and before making the system available for actual use.

CHAPTER 8

SCREENSHOTS

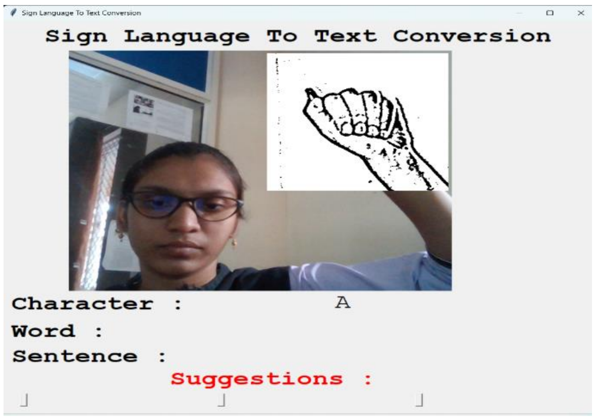
CODE:

```
# Importing Libraries

import numpy as np

import cv2
import os, sys
import time
import operator
from keras.models import model_from_json
from string import ascii_uppercase
from spellchecker import SpellChecker
import tkinter as tk
from PIL import Image, ImageTk



from keras.models import model_from_json

os.environ["THEANO_FLAGS"] = "device=cuda, assert_no_cpu_op=True"

#Application :

class Application:

    def __init__(self):

        spell=SpellChecker()
        self.hs=spell
        self.vs = cv2.VideoCapture(0)
        self.current_image = None
        self.current_image2 = None
        self.json_file = open("model_new.json", "r")
        self.model_json = self.json_file.read()
        self.json_file.close()
        self.loaded_model = model_from_json(self.model_json)
        self.loaded_model.load_weights("model_new.h5")
        self.json_file_dru = open("model-bw_dru.json" , "r")
        self.model_json_dru = self.json_file_dru.read()
        self.json_file_dru.close()
        self.loaded_model_dru = model_from_json(self.model_json_dru)
        self.loaded_model_dru.load_weights("model-bw_dru.h5")
        self.json_file_tkdi = open("model-bw_tkdi.json" , "r")
        self.model_json_tkdi = self.json_file_tkdi.read()
        self.json_file_tkdi.close()

        self.loaded_model_tkdi = model_from_json(self.model_json_tkdi)
        self.loaded_model_tkdi.load_weights("model-bw_tkdi.h5")
        self.json_file_smn = open("model-bw smn.json" , "r")
        self.model_json_smn = self.json_file_smn.read()
        self.json_file_smn.close()
        self.loaded_model_smn = model_from_json(self.model_json_smn)
        self.loaded_model_smn.load_weights("model-bw_smn.h5")
        self.ct = {}
        self.ct['blank'] = 0
        self.blank_flag = 0
        for i in ascii_uppercase:
          self.ct[i] = 0
        print("Loaded model from disk")

        self.root = tk.Tk()
        self.root.title("Sign Language To Text Conversion")
        self.root.protocol('WM_DELETE_WINDOW', self.destructor)
        self.root.geometry("900x900")

        self.panel = tk.Label(self.root)
        self.panel.place(x = 100, y = 10, width = 580, height = 580)

        self.panel2 = tk.Label(self.root) # initialize image panel
        self.panel2.place(x = 400, y = 65, width = 275, height = 275)

        self.T = tk.Label(self.root)
        self.T.place(x = 60, y = 5)
        self.T.config(text = "Sign Language To Text Conversion", font = ("Courier", 30,
"bold"))

        self.panel3 = tk.Label(self.root) # Current Symbol
        self.panel3.place(x = 500, y = 540)

        self.T1 = tk.Label(self.root)
        self.T1.place(x = 10, y = 540)
        self.T1.config(text = "Character :", font = ("Courier", 30, "bold"))

        self.panel4 = tk.Label(self.root) # Word
        self.panel4.place(x = 220, y = 595)

        self.T2 = tk.Label(self.root)
        self.T2.place(x = 10,y = 595)
        self.T2.config(text = "Word :", font = ("Courier", 30, "bold"))

        self.panel5 = tk.Label(self.root) # Sentence
        self.panel5.place(x = 350, y = 645)

        self.T3 = tk.Label(self.root)
        self.T3.place(x = 10, y = 645)
        self.T3.config(text = "Sentence :",font = ("Courier", 30, "bold"))

        self.T4 = tk.Label(self.root)
        self.T4.place(x = 250, y = 690)
        self.T4.config(text = "Suggestions :", fg = "red", font = ("Courier", 30, "bold"))

        self.bt1 = tk.Button(self.root, command = self.action1, height = 0, width = 0)
        self.bt1.place(x = 26, y = 745)

        self.bt2 = tk.Button(self.root, command = self.action2, height = 0, width = 0)
        self.bt2.place(x = 325, y = 745)

        self.bt3 = tk.Button(self.root, command = self.action3, height = 0, width = 0)
        self.bt3.place(x = 625, y = 745)


        self.str = ""
        self.word = " "
        self.current_symbol = "Empty"
        self.photo = "Empty"
        self.video_loop()


    def video_loop(self):
        ok, frame = self.vs.read()

        if ok:
            cv2image = cv2.flip(frame, 1)

            x1 = int(0.5 * frame.shape[1])
            y1 = 10
            x2 = frame.shape[1] - 10
            y2 = int(0.5 * frame.shape[1])

            cv2.rectangle(frame, (x1 - 1, y1 - 1), (x2 + 1, y2 + 1), (255, 0, 0) ,1)
            cv2image = cv2.cvtColor(cv2image, cv2.COLOR_BGR2RGBA)
        self.current_image = Image.fromarray(cv2image)
        imgtk = ImageTk.PhotoImage(image = self.current_image)

        self.panel.imgtk = imgtk
        self.panel.config(image = imgtk)

        cv2image = cv2image[y1 : y2, x1 : x2]

        gray = cv2.cvtColor(cv2image, cv2.COLOR_BGR2GRAY)

        blur = cv2.GaussianBlur(gray, (5, 5), 2)

        th3 = cv2.adaptiveThreshold(blur, 255
,cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 11, 2)

        ret, res = cv2.threshold(th3, 70, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)

        self.predict(res)

        self.current_image2 = Image.fromarray(res)

        imgtk = ImageTk.PhotoImage(image = self.current_image2)

        self.panel2.imgtk = imgtk
        self.panel2.config(image = imgtk)

        self.panel3.config(text = self.current_symbol, font = ("Courier", 30))

        self.panel4.config(text = self.word, font = ("Courier", 30))

        self.panel5.config(text = self.str,font = ("Courier", 30))

        predicts = self.hs.correction(self.word)#correction

        if(len(predicts) > 1):

            self.bt1.config(text = predicts[0], font = ("Courier", 20))

        else:

            self.bt1.config(text = "")
            if(len(predicts) > 2):
                self.bt2.config(text = predicts[1], font = ("Courier", 20))
            else:
                self.bt2.config(text = "")
            if(len(predicts) > 3):
                self.bt3.config(text = predicts[2], font = ("Courier", 20))
            else:
                self.bt3.config(text = "")

        self.root.after(5, self.video_loop)
    def predict(self, test_image):
        test_image = cv2.resize(test_image, (128, 128))
        result = self.loaded_model.predict(test_image.reshape(1, 128, 128, 1))

        result_dru = self.loaded_model_dru.predict(test_image.reshape(1 , 128 , 128 , 1))
        result_tkdi = self.loaded_model_tkdi.predict(test_image.reshape(1 , 128 , 128 , 1))
        result_smn = self.loaded_model_smn.predict(test_image.reshape(1 , 128 , 128 , 1))
        prediction = {}
        prediction['blank'] = result[0][0]
        inde = 1
        for i in ascii_uppercase:
            prediction[i] = result[0][inde]
            inde += 1
        #LAYER 1

        prediction = sorted(prediction.items(), key = operator.itemgetter(1), reverse =
True)

        self.current_symbol = prediction[0][0]


        #LAYER 2

        if(self.current_symbol == 'D' or self.current_symbol == 'R' or self.current_symbol
== 'U'):

          prediction = {}
          prediction['D'] = result_dru[0][0]
          prediction['R'] = result_dru[0][1]
          prediction['U'] = result_dru[0][2]

          prediction = sorted(prediction.items(), key = operator.itemgetter(1), reverse =
True)

          self.current_symbol = prediction[0][0]

        if(self.current_symbol == 'D' or self.current_symbol == 'I' or self.current_symbol
== 'K' or self.current_symbol == 'T'):

          prediction = {}

          prediction['D'] = result_tkdi[0][0]
          prediction['I'] = result_tkdi[0][1]
          prediction['K'] = result_tkdi[0][2]
          prediction['T'] = result_tkdi[0][3]

          prediction = sorted(prediction.items(), key = operator.itemgetter(1), reverse =
True)

          self.current_symbol = prediction[0][0]

        if(self.current_symbol == 'M' or self.current_symbol == 'N' or self.current_symbol
== 'S'):

          prediction1 = {}

          prediction1['M'] = result_smn[0][0]
          prediction1['N'] = result_smn[0][1]
          prediction1['S'] = result_smn[0][2]

          prediction1 = sorted(prediction1.items(), key = operator.itemgetter(1), reverse =
True)

          if(prediction1[0][0] == 'S'):

             self.current_symbol = prediction1[0][0]

          else:
             self.current_symbol = prediction[0][0]

        if(self.current_symbol == 'blank'):

            for i in ascii_uppercase:
                self.ct[i] = 0

        self.ct[self.current_symbol] += 1

        if(self.ct[self.current_symbol] > 60):

            for i in ascii_uppercase:
                if i == self.current_symbol:
                    continue

                tmp = self.ct[self.current_symbol] - self.ct[i]

                if tmp < 0:
                    tmp *= -1

                if tmp <= 20:
                    self.ct['blank'] = 0

                    for i in ascii_uppercase:
                        self.ct[i] = 0
                    return

            self.ct['blank'] = 0

            for i in ascii_uppercase:
                self.ct[i] = 0

            if self.current_symbol == 'blank':
```

```
          if self.blank_flag == 0:
            self.blank_flag = 1

            if len(self.str) > 0:
              self.str += " "

            self.str += self.word

            self.word = ""

        else:

          if(len(self.str) > 16):
            self.str = ""

          self.blank_flag = 0

          self.word += self.current_symbol

    def action1(self):

     predicts = self.hs.suggest(self.word)
     if(len(predicts) > 0):

        self.word = ""

        self.str += " "

        self.str += predicts[0]

    def action2(self):

     predicts = self.hs.suggest(self.word)

     if(len(predicts) > 1):
        self.word = ""
        self.str += " "
        self.str += predicts[1]

    def action3(self):

     predicts = self.hs.suggest(self.word)
```

```
        if(len(predicts) > 2):
            self.word = ""
            self.str += " "
            self.str += predicts[2]

    def action4(self):

        predicts = self.hs.suggest(self.word)

        if(len(predicts) > 3):
            self.word = ""
            self.str += " "
            self.str += predicts[3]

    def action5(self):

        predicts = self.hs.suggest(self.word)

        if(len(predicts) > 4):
            self.word = ""
            self.str += " "
            self.str += predicts[4]

    def destructor(self):

        print("Closing Application...")

        self.root.destroy()
        self.vs.release()
        cv2.destroyAllWindows()

print("Starting Application...")

(Application()).root.mainloop()
```

OUTPUT:



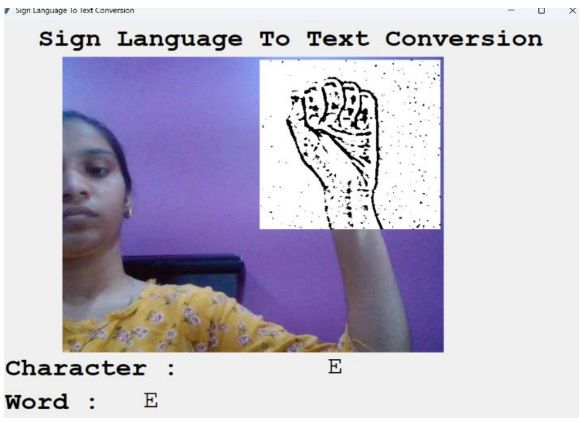
Sign Language To Text Conversion

Character : A

Word :

Sentence :

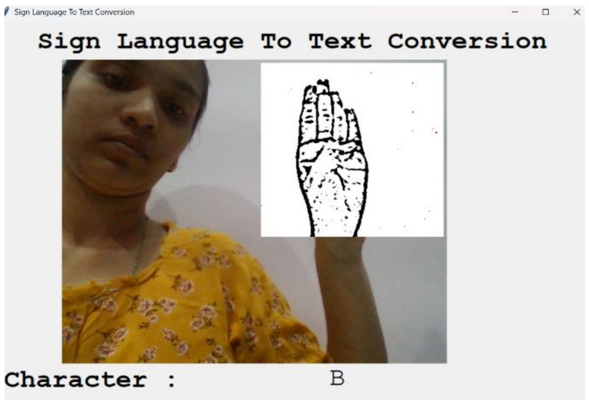
Suggestions :



Sign Language To Text Conversion

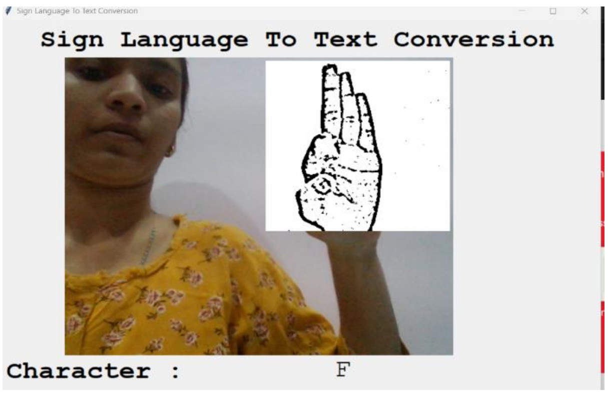
Character : E

Word : E



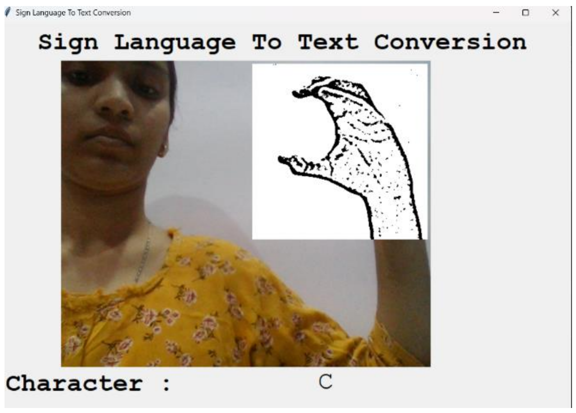
Sign Language To Text Conversion

Character : B



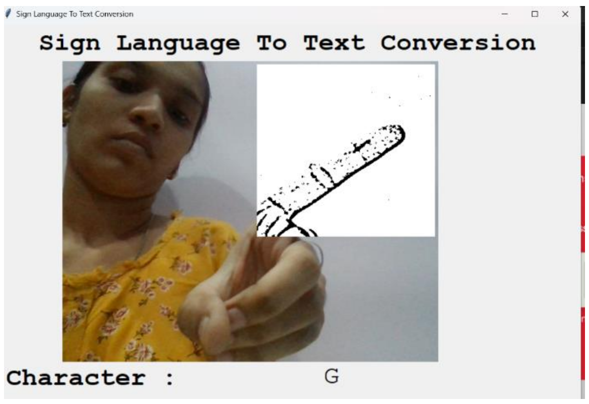
Sign Language To Text Conversion

Character : F



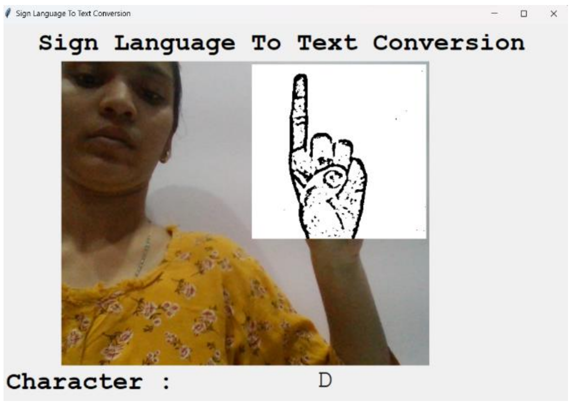
Sign Language To Text Conversion

Character : C



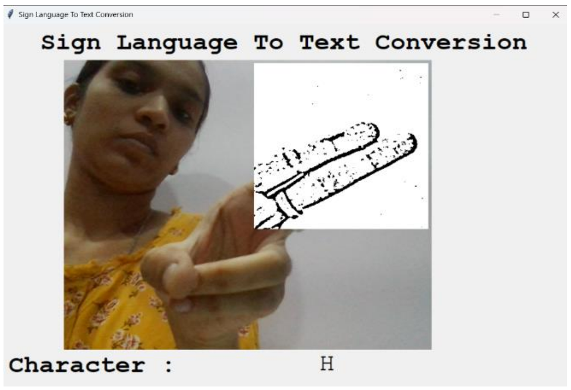
Sign Language To Text Conversion

Character : G



Sign Language To Text Conversion

Character : D



Sign Language To Text Conversion

Character : H

CHAPTER 9

CONCLUSION AND FUTURE WORK

Conclusion

To Conclude, this project is a simple recognition of sign language and convert them into text. This System Consists of Web camera which is used to capture the gestures. The Signs are processed for feature extraction. The features are compared with testing database to calculate the sign recognition. Eventually, recognized hand gesture is converted into text. This system provides an opportunity for deaf and dumb people to easily communicate with normal people.

Future work

- To increase the performance and accuracy, the quality of the training dataset used should be enhanced.
- This project did not focus on facial expressions although it is well known that facial expressions convey an important part of sign language.
- This project can also enhance and implement speech emotion recognitionas well as speech recognition.
- The project can be further extended by achieving better results even in darker environments where hand movements are clearly not visible.

References

- Anuja V Nair, V Bindu - A review on Indian sign language recognition Int J Compute Appl, 73 (22) (2013)
- K. Athira, C.J. Sruthi, A. Liliya - A signer independent sign language recognition with Co articulation elimination from live videos: an Indian scenario J King Saud Univ compute Inf Sci, 34 (3) (2022), pp. 771-778
- S.C. Agrawal, A.S. Jalal, C. Bhatnagar, Ieee - Recognition of Indian signlanguage using feature Fusion