THIRD EYE FOR VISUALLY IMPAIRED

A PROJECT REPORT

Submitted by

SHAFREEN FATHIMA ASMATHULLAH KHAN [211419104241]

SHIRLY N [211419104250]

SUJITHRA J [211419104272]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2023

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"THIRD EYE FOR VISUALLY IMPAIRED"** is the bonafide work of **"SHAFREEN FATHIMA ASMATHULLAH KHAN (211419104241) SHIRLY N(211419104250) SUJITHRA J (211419104272) "** who carried out the project under my supervision.

SIGNATURE	**SIGNATURE**
Dr.L.JABASHEELA,M.E.,Ph.D., **HEAD OF THE DEPARTMENT**	**Dr.T.TAMILVIZHI,M.TECH.,Ph.D.,** **ASSOCIATE PROFESSOR**
DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, POONAMALLEE, CHENNAI-600 123.	DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, POONAMALLEE, CHENNAI-600 123.

Certified that the above candidates were examined in the End Semester Project Viva-Voce Examination held on 11.04.2023.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We SHAFREEN FATHIMA ASMATHULLAH KHAN (211419104241),

SHIRLY N (211419104247), SUJITHRA J (211419104272) hereby declare that this project report titled "**THIRD EYE FOR VISUALLY IMPAIRED**" , under the guidance of **Dr.T.TAMILVIZHI**, M.TECH Ph.D. is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

SHAFREEN FATHIMA ASMATHULLAH KHAN

SHIRLY N

SUJITHRA J

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D**. for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr.L.JABASHEELA, M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank our Guide, **Dr.T.TAMILVIZHI, M.TECH.,Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

SHAFREEN FATHIMA ASMATHULLAH KHAN

SHIRLY N

SUJITHRA J

ABSTRACT

People with visual impairments face many challenges in their daily lives, including difficulties with mobility, communication, and accessing information. We propose a camera-based detection system that utilizes artificial intelligence (AI) to assist visually impaired individuals. The system, which we call "Third Eye," employs AI algorithms for face recognition, Text recognition and speech synthesis. The system is designed to provide audio feedback to users when it detects trained people, texts or signs in the user's surroundings.

This system consists of a camera which continuously captures images of the user's or printed texts or signs from the surroundings, which are then analyzed by the AI algorithms to detect texts, faces, and signs. The system is capable of recognizing the faces of trained individuals and providing audio feedback with their names. It can also recognize two faces simultaneously and provide audio feedback with both their names. This feature is particularly useful in social situations where the visually impaired user may not recognize people by their voice alone. Additionally, the system can read printed text from books and magazines aloud, providing an enhanced reading experience for visually impaired individuals. Another notable feature of the Third Eye system is its ability to recognize trained signs and provide audio feedback to the user. This is especially useful for individuals who rely on sign language to communicate.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

ABBREVIATION	EXPANSION
AI	Artificial Intelligence
ER	Entity Relationship
LBPH	Local Binary Pattern Histogram
GTTS	Google Text To Speech Converter
UML	Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

According to the World Health Organization (WHO), an estimated 2.2 billion people globally have a vision impairment or blindness, with about 1 billion of those cases being preventable or treatable. Visual impairment refers to a range of conditions that affect a person's ability to see, including partial or complete blindness, low vision, and color blindness. People with visual impairments face many challenges in their daily lives, including difficulties with mobility, communication, and accessing information. It is one of the most severe and life-changing disabilities that a person can face, and it impacts every aspect of their life, from their daily routines to their ability to work and socialize. Blindness can have a profound impact on a person's mental health and well-being, as well as their ability to participate in society. It can lead to feelings of isolation, depression, and anxiety, and can make it difficult for individuals to find employment or participate in recreational activities. Blind individuals have to rely on their other senses, such as touch, hearing, and smell, to navigate and interact with the world around them. They often face challenges with reading, communicating, and accessing information that sighted people take for granted. It can be caused by various factors, including genetics, injuries, diseases, and age-related changes. Visual impairments can have a significant impact on a person's quality of life, affecting their ability to perform daily tasks and participate in social activities.

1.2 PROBLEM DEFINITION

The visually impaired people face many challenges in society and are often vulnerable to exploitation. They are heavily dependent on others for their day-to-day activities. Therefore, there is a need to empower them and provide them with a sense of independence. To address these challenges, we propose a camera-based detection system, which can be a game-changer for them. It enables them to read texts, recognize trained people, and detect signs through audio feedback, which increases their mobility and independence. This system also enhances their social interaction skills and reduces their dependence on others. this system has the great potential to improve the quality of life of visually impaired people.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

TITLE : Smart Assistive System for Blind People.

AUTHORS : Usman Masud ,Tareq Saeedhunida, Malaikhah

YEAR OF PUBLICATION : 2023

ABSTRACT: Recent progress in innovation is making the life prosper, simpler and easier for common indi-vidual. The World Health Organization (WHO) statistics indicate that a large amount of people experience visual losses, because of which they encounter many difficulties in everyday jobs. Hence, our goal is to structure a modest, secure, wearable, and versatile framework for visually impaired to help them in their daily routines. For this, the plan is to make an effective system which will assist visually impaired people through obstacle detection and scenes classification. The proposed methodology utilizes Raspberry-Pi 4B,Camera, Ultrasonic Sensor and Arduino, mounted on the stick of the individual. We take pictures of the scene and afterwards pre-process these pictures with the help of Viola Jones and TensorFlow Object Detection algorithm. The said techniques are used to detect objects. We also used an ultrasonic sensor mounted on aservomotor to measure the distance between the blind person and obstacles. The presented research utilizes simple calculations for its execution, and detects the obstructions with a notably high efficiency. When contrasted with different frameworks, this framework is a minimal effort, convenient, and simple to wear.

2.2 LITERATURE SURVEY

TITLE : Analysis of Navigation Assistants for Blind and Visually Impaired People :A Systematic Review.

AUTHORS : Sulaiman Khan, Shah Naziri, Habib Ullah Khan

YEAR OF PUBLICATION : 2022

ABSTRACT: Over the last few decades, the development in the field of navigation and routing devices has become a hindering task for the researchers to develop smart and intelligent guiding mechanism at indoor and outdoor locations for blind and visually impaired people (BVIPs). The existing research need to be analysed from a historical perception including early research on the first electronic travel aids to the use of modern artificial vision models for the navigation of BVIPs. Diverse approaches such as: e-cane or guide dog, infrared-based cane, laser based walker and many others are proposed for the navigation of BVIPs. But most of these techniques have limitations such as: infrared and ultrasonic based assistance has short range capacities for object detection. While laser based assistance can harm other people if it directly hit them on their eyes or any other part of the body. These trade-offs are critical to bring this technology in practice.To systematically assess, analyze, and identify the primary studies in this specialized field and provide an overview of the trends and empirical evidence in the proposed field. This systematic research work is performed by defining a set of relevant keywords, formulating four research questions, defining selection criteria for the articles, and synthesizing the empirical evidence in this area.

2.3 LITERATURE SURVEY

TITLE : Wearable Navigation Assistant for Visually Impaired People

AUTHORS : Nasim Rand Hemanth Kumar

YEAR OF PUBLICATION : 2022

ABSTRACT: Vision is one of the powerful sense in all of the senses. Visually challenged people face many problems in terms of obstacles in their mobility in the case of outdoor as well as indoor. For their daily activities, they are always reliant on others. Visually impaired people meet many accidents, they often fall off and they may get lost in some unknown areas due to failure in adapting and identifying the surrounding environment. “Smart Shoes” are designed to provide a better solution for the visually impaired to move safely and independently. It is built using “Internet of Things” technology and the shoes are embedded with various sensors, microcontrollers, buzzer, speaker and vibration motor. The shoe alerts the wearer whenever he or she walks in front of an obstacle. Also, it detects the wet floor and water bodies and gives alert about it. To increase the security and safety of the visually impaired, whenever the person falls down an alert message is sent to the parent’s or caretaker’s telegram bot. Smart shoes are designed to provide a safe and comfortable companion in their daily life activities. As a result, a complete adaptive equipment is being advanced to improve the standard and comfort of life of the visually impaired.

2.4 LITERATURE SURVEY

TITLE : Obstacle Detection For Blind People Using Ultrasonic Sensors And Ardino Processor.

AUTHORS : Akbar Ali, Haroon Akbar, Zeeshan Sartaj

YEAR OF PUBLICATION : 2022

ABSTRACT: In today's world millions of people are having problem of vision. According to the recent survey of WHO about 289 million of peoples are visually impaired. This problem make them dependent on others. Recent studies shows that about 108 different methods are used for the aid of mobility of the visually impaired people. In this project we will make the use of Ultrasonic sound to enhance the obstacle detection for the blind people within the range of 3m to 4m. The project is made by inspiring from the natural navigation system of a mammal called Bat.Although the navigation for blind using ultrasonic is a complex process but here we will present only the hurdle detection procedure. The project make the use of UNO Arduino processor, ultrasonic sensors, 5V Buzzer, Arduino cable for burning, and some connecting wires. All this setup will be arranged on a cap, which the visually impaired person will wear. The ultrasonic emitter emit ultrasonic sound of high frequency which is perceived by the ultrasonic receiver after reflection from the obstacle and then passed to the processor for further necessary action in the form of some sound.

2.5 LITERATURE SURVEY

TITLE : Mobile Based IoT Solution for Helping Visual Impairment Users.

AUTHORS : Hussein Abdel-Jaber, Hussein Albazar, Ahmed Abdel-Wahab, Malak El Amir,Areej Alqahtani, Mohammed Alobaid

YEAR OF PUBLICATION : 2021

ABSTRACT: Blind people have many tasks to do in their lives. However, blindness generates challenges for them to perform their tasks. Many blind persons use a traditional stick to move around and to perform their tasks. But the obstacles are not detected in traditional stick, it is ineffective for visually impaired people. The blind person has no idea what kind of objects or obstacles are in front of him/her. The blind person has no idea what size the object is or how far away he or she is from it. It is difficult for a blind person to get around. To assist people with vision impairment by making many of their daily tasks simple, comfortable, and organized, they will be able to recognize anything (an obstacle for blind people). A smart stick with mobile application can be used. One of the solutions is a mobile-based Internet of Things solution, which is a stick intended to assist visually impaired people to navigate more easily. It enables blindness and low vision people to navigate and carry out their daily tasks with ease and comfort. A technologically advanced blind stick that enables visually impaired people to move with ease. This paper proposes a system of software and hardware that helps visually impairment people to find their ways in an easy and comfortable way. The proposed system uses smart stick and mobile application to help blind and visually im- paired people to identify objects (such as walls, tables, vehicles, people, etc.) in their ways, this can enable them to avoid these objects. In addition, as a result, the system will notify the user through sound from the smartphone. Finally, if he/she gets lost, he will be able to send an SMS with his/her GPS location.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system has several drawbacks that limit its effectiveness in aiding visually impaired individuals. Firstly, the use of Braille language for reading documents is challenging, as it requires users to have a highly sensitive touch to read the tiny dots. This can lead to errors or incomplete reading, making it difficult for visually impaired individuals to access written information. Secondly, the use of ultrasonic sensors for object detection is limited in its scope, as it can only detect objects within a certain range and in specific directions. This makes it difficult for individuals to detect objects that are not directly in front, behind, or to the sides of them. Additionally, the system relies on pre-recorded audio messages for object detection, which may not provide sufficient information about the object or its location. These existing system does not provide a comprehensive solution to the challenges faced by visually impaired individuals. It requires a high degree of physical sensitivity and may not effectively detect all objects in the user's environment. Furthermore, the use of pre-recorded audio messages can limit the amount and specificity of information provided to users. Therefore, there is a need for an improved system that can more accurately detect objects and provide more detailed information to visually impaired individuals.

3.2 PROPOSED SYSTEM

The proposed system aims to enhance the independence of visually impaired individuals by introducing a camera-based detection system that enables them to identify trained persons, products,signs and texts. The system uses Haarcascade classifier and LBPH algorithm, two well-established technologies for object detection and feature extraction. A novel technique called double face recognition technology is proposed, which allows the detection of two faces simultaneously. The system operates as an artificial eye for visually impaired individuals, enabling them to navigate their surroundings without the need for human supervision. By overcoming the limitations of the existing system, the proposed method provides a practical solution to enhance the quality of life for visually impaired individuals.

ADVANTAGES

- It provides greater independence and autonomy by allowing users to identify people, objects, and text without the need for human supervision.
- It is more user-friendly and efficient as it eliminates the need for Braille reading and ultrasonic sensors.
- The system can be easily operated by visually impaired individuals with minimal training, and the double face recognition feature enhances its usability.

3.3 FEASIBILITY STUDY

TECHNICAL FEASIBILITY

- The project demands knowledge and expertise in computer vision, machine learning, and software development.
- The usage of well-established technologies like OpenCV, Haarcascade classifier, LBPH, Caffe, Tesseract OCR Engine, and Python, which have extensive documentation and community support is essential.
- The project may require hardware upgrades, such as a powerful CPU or GPU, to guarantee real-time processing capability.

ECONOMIC FEASIBILITY

- The project may require a significant investment in hardware, software, and manpower.
- There may be ongoing maintenance costs, such as updates to software libraries and hardware upgrades.
- The project may require licensing fees for proprietary software and datasets.

OPERATIONAL FEASIBILITY

- The project must meet the needs of visually impaired individuals and be designed with their input and feedback.
- The project may require extensive testing and validation to ensure accuracy and reliability.
- The project may require ongoing support and training for end-users.

3.4 HARDWARE REQUIREMENTS

● Processor	: Intel i3
● Hard disk	: minimum 10 GB
● RAM	: minimum 4 GB

3.5 SOFTWARE REQUIREMENTS

- Operating System: Windows 10 or later
- Tool: Python IDLE

CHAPTER 4

SYSTEM DESIGN

4.1 ER DIAGRAM

4.1.1 FACE RECOGNITION:

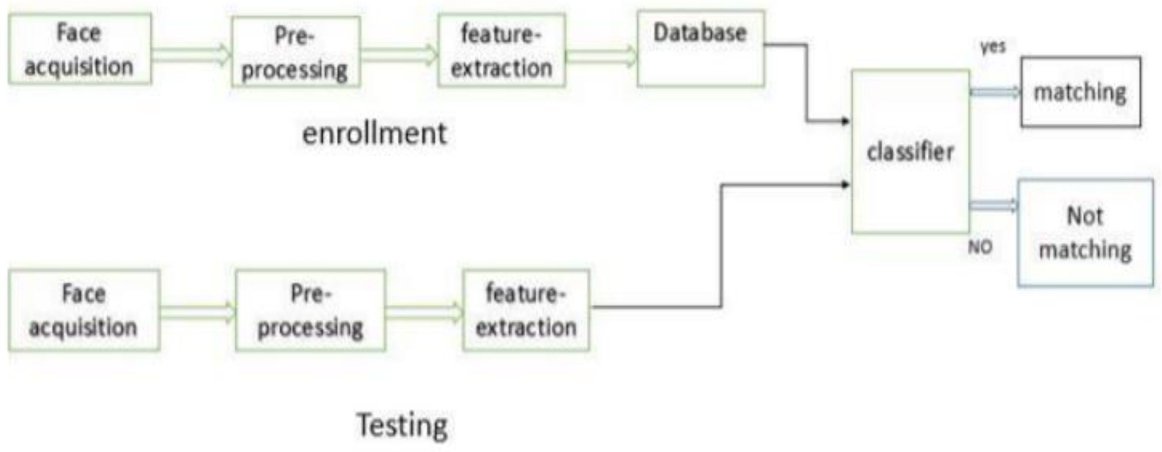


Fig 4.1.1 Entity Relationship Diagram For Face Recognition

4.1.2 TEXT REOGNITION:

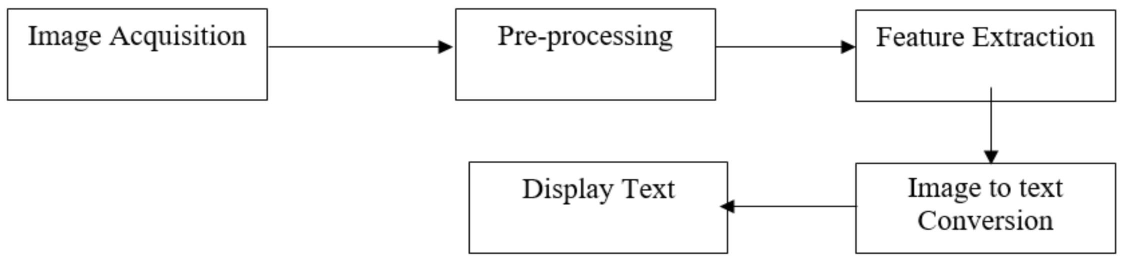


Fig 4.1.2 Entity Relationship Diagram For Text Recognition

4.1.3 SIGN PREDICTION:

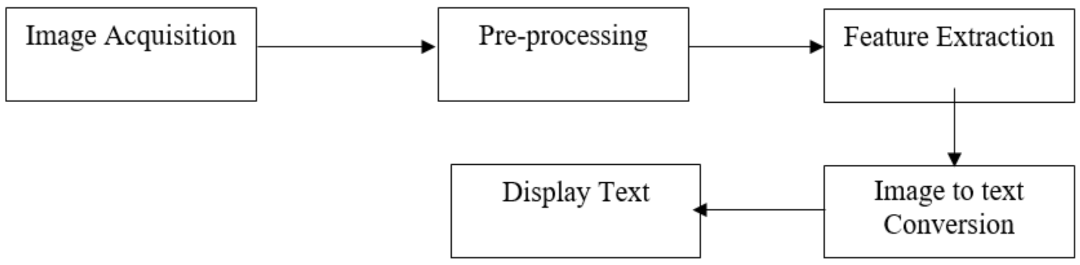


Fig 4.1.2 Entity Relationship Diagram For Sign Prediction

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

4.3 DATA FLOW DIAGRAM

LEVEL 0

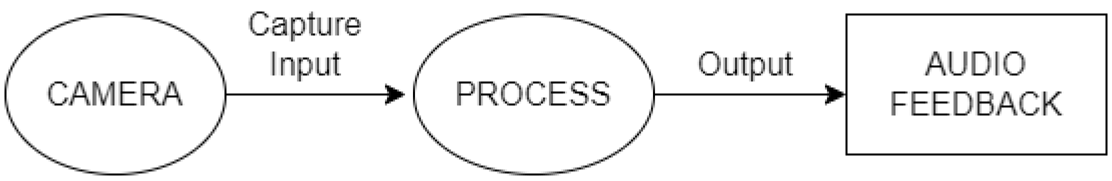


Fig 4.3.1 Level 0 Data Flow Diagram For Third Eye for Visually Impaired

LEVEL 1

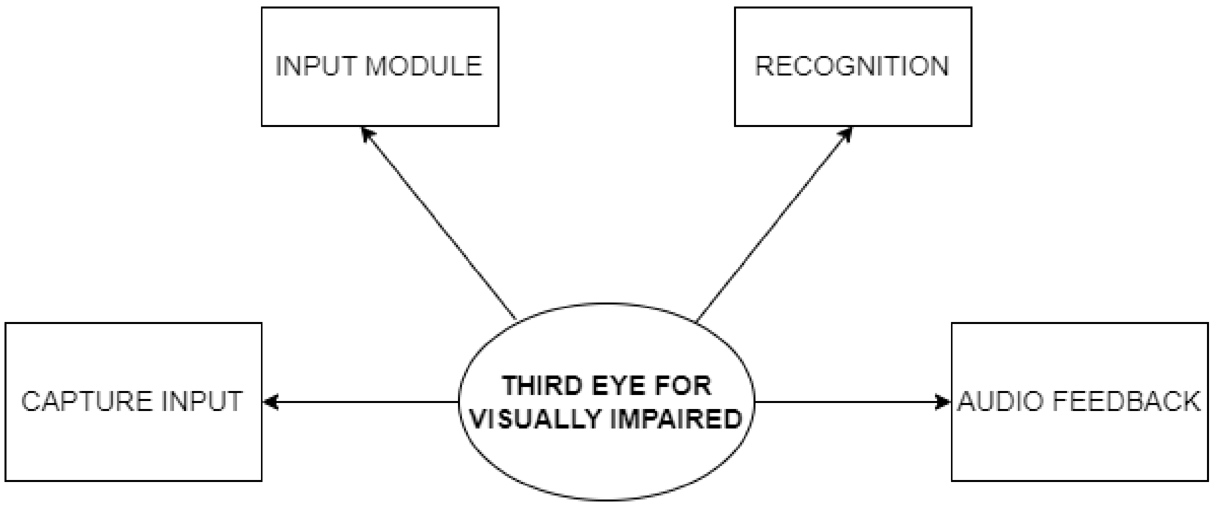


Fig 4.3.2 Level 1 Data Flow Diagram For Third Eye for Visually Impaired

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM). A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. In this diagram, the input (images or video) is captured by the laptop's camera, and passed to the image/video processing component. This component processes the input to identify whether it is a family member's face, text, or sign language. The output from this component is passed to the audio output component, which converts the output to audio and speaks it aloud.The recognition models component includes the face recognition model, text recognition model, and sign language recognition model. These models are trained on datasets specific to each type of input and are used to recognize the input and generate the appropriate output.

4.4 UML DIAGRAMS

4.4.1 USE CASE DIAGRAM

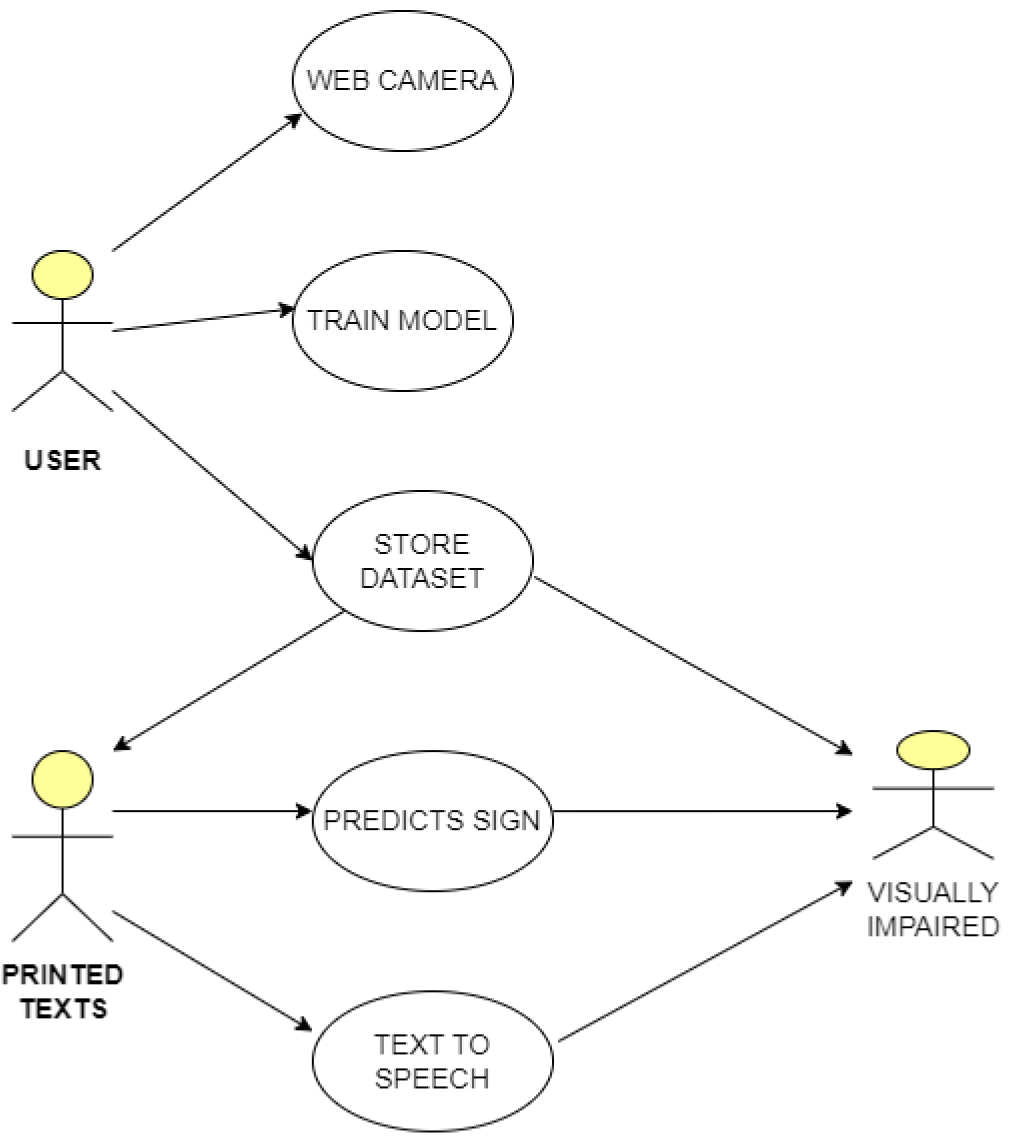


Fig 4.4.1 Use Case Diagram For Third Eye for Visually Impaired System

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analysed the functionalities are captured in use cases. So, it can say that use cases are nothing but the system functionalities written in an organized manner.

4.4.2 CLASS DIAGRAM

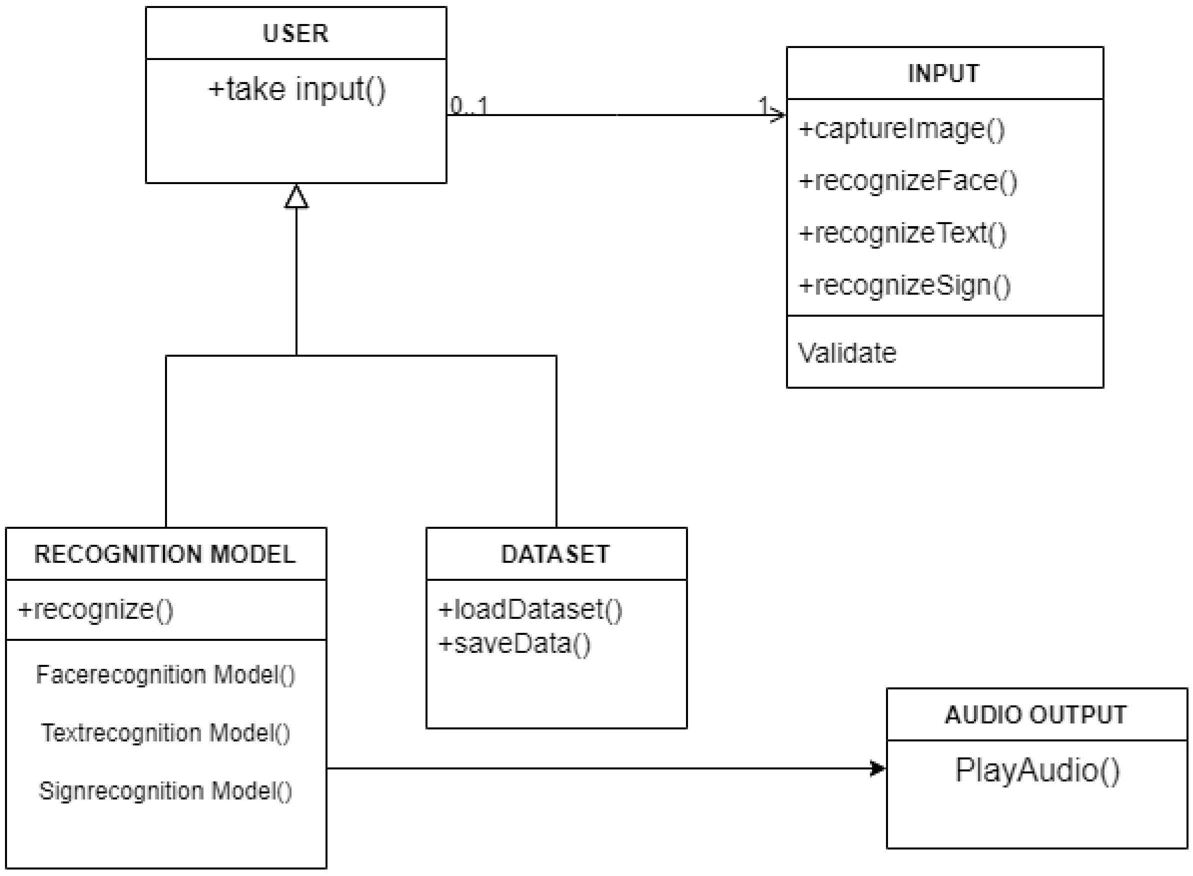


Fig 4.4.2 Class Diagram For Third Eye for Visually Impaired System

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified and because unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable

4.4.3 ACTIVITY DIAGRAM

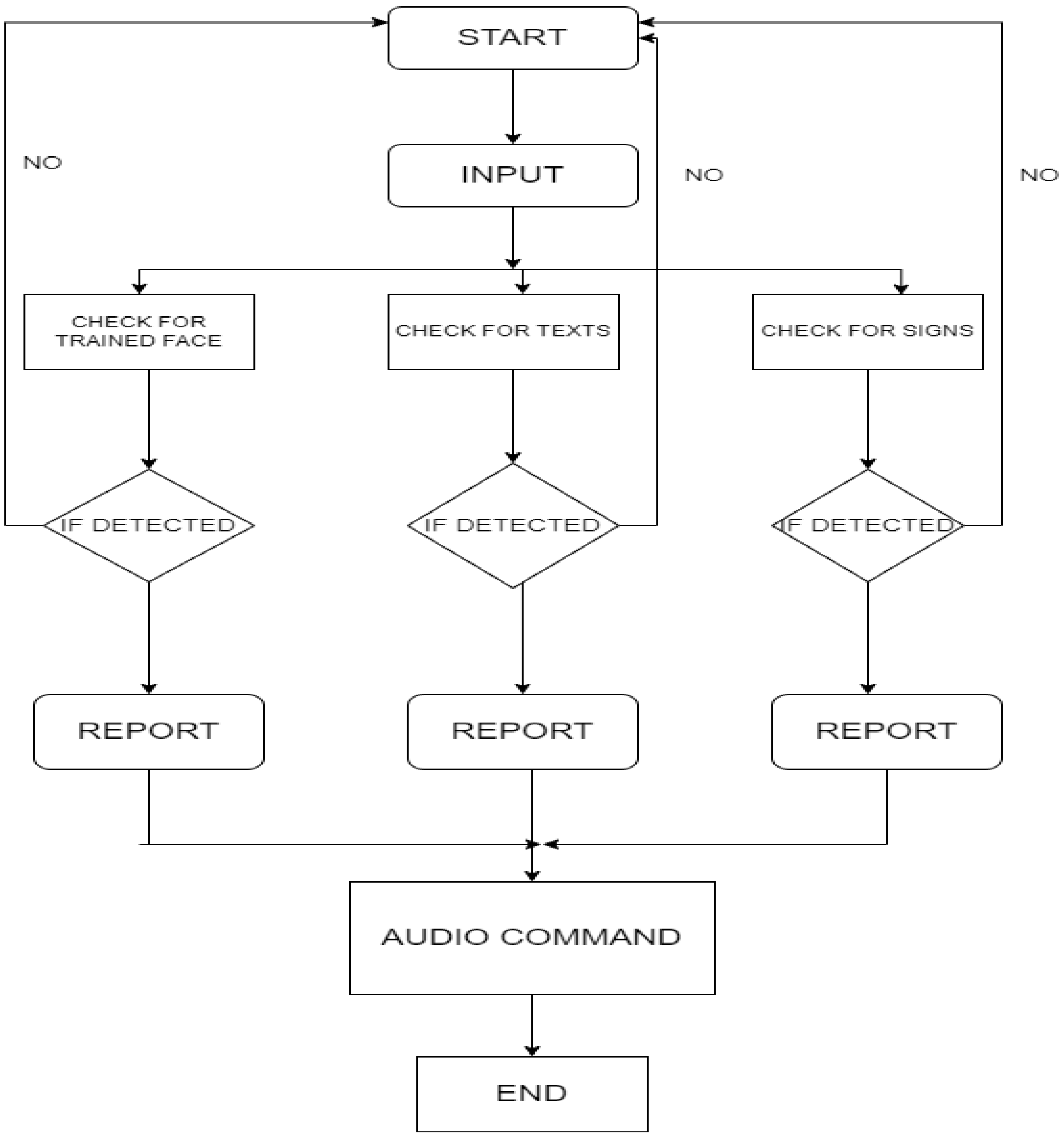


Fig 4.4.3 Activity Diagram For Third Eye for Visually Impaired System

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part. It does not show any message flow from one activity to another.

4.4.4 SEQUENCE DIAGRAM

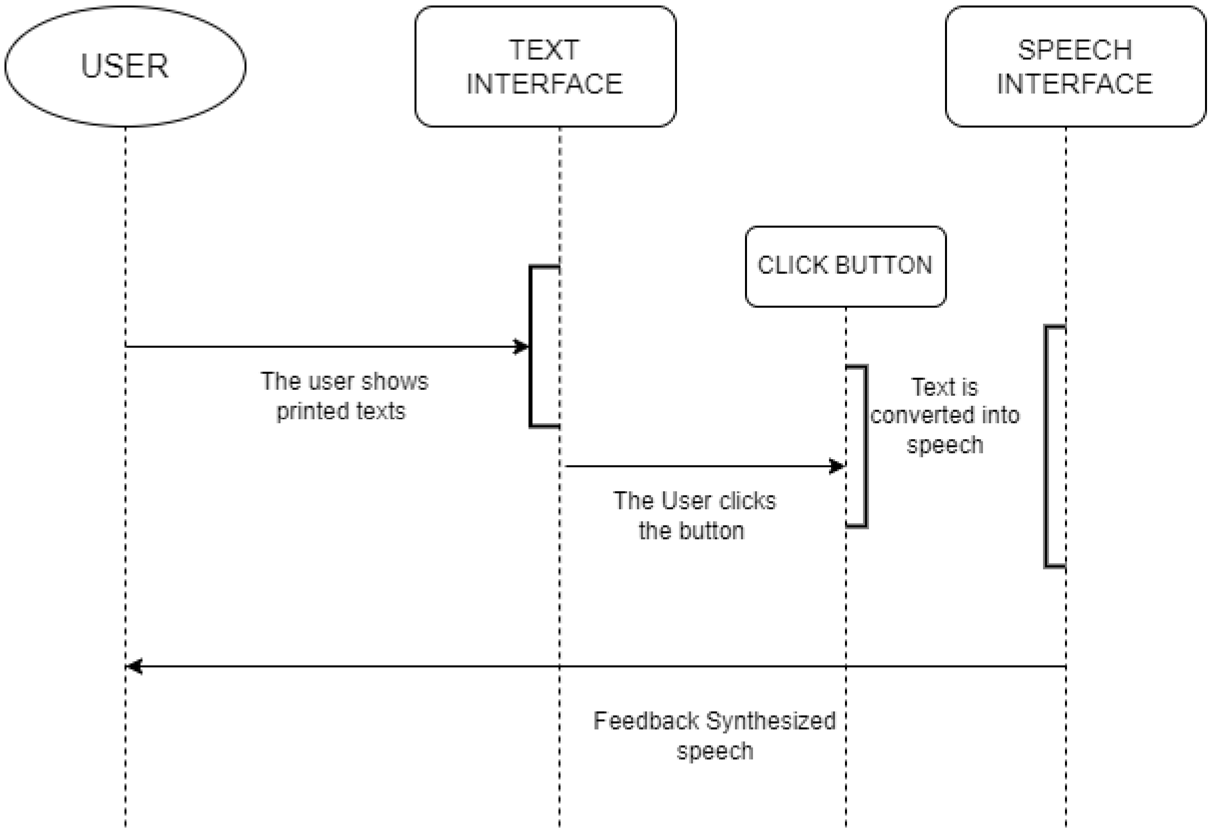

USER
TEXT INTERFACE
SPEECH INTERFACE
CLICK BUTTON
The user shows printed texts
Text is converted into speech
The User clicks the button
Feedback Synthesized speech

Fig 4.4.4 Sequence Diagram For Third Eye for Visually Impaired System

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modelling, which focuses on identifying the behaviour within your system.

CHAPTER 5

SYSTEM ARCHITECTURE

ARCHITECTURE OVERVIEW

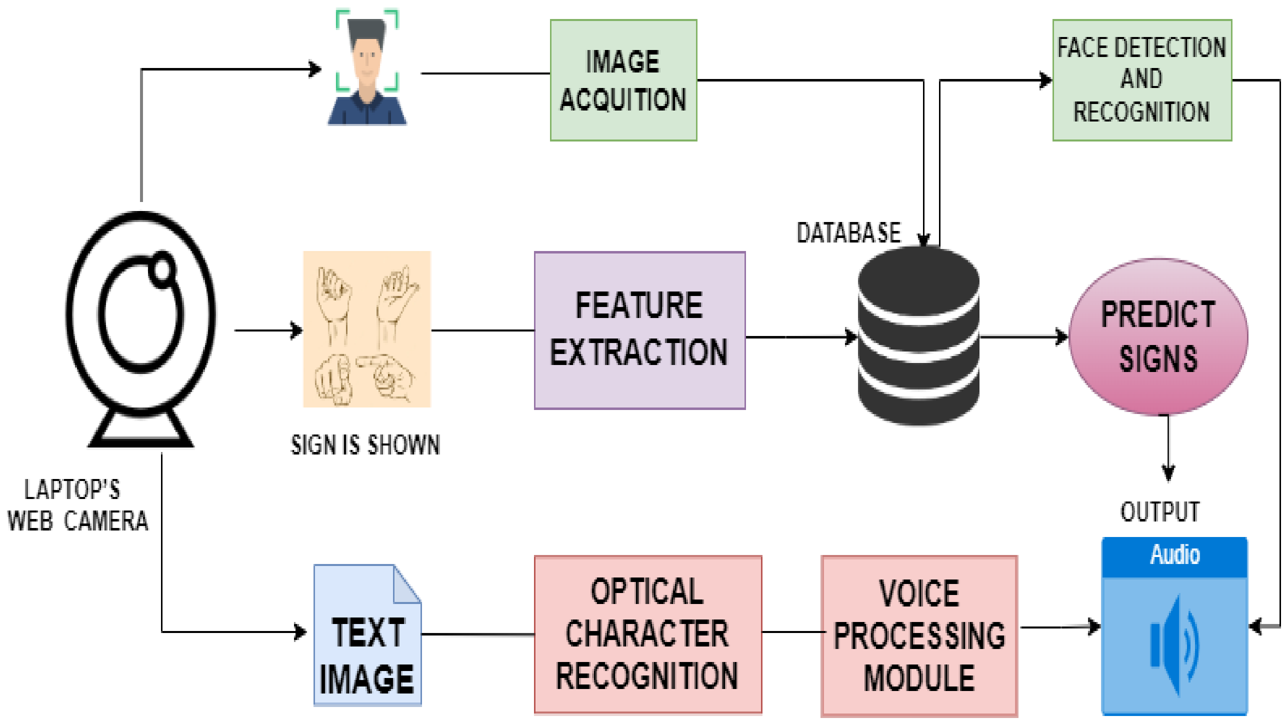


Fig 5.1 Architecture Diagram For Third Eye for Visually Impaired System

The system takes input through a laptop's webcam, which captures the image. The captured image is then processed by the Haar Cascade algorithm to detect and recognize the face using the pre-existing database. The audio output will provide the name of the recognized person. the proposed system employs feature extraction techniques for identifying trained signs displayed to the camera. Based on the predicted sign, audio feedback is generated and provided to the user.In case an image of text or a printed book is shown, the Optical Character Recognition (OCR) technique is used, which processes the text using the Tesseract software. The processed text is then converted to audio output through the voice processing module.

5.1 MODULE DESIGN SPECIFICATION

MODULE – 1

FACE RECOGNITION

This module is responsible for processing the input image from the laptop’s web camera and recognize the face in the image. Face detection is a type of application classified under “computer vision” technology. It is the
process in which algorithms are developed and trained to properly locate faces or objects (in bject detection, a related system), in images. Local Binary Pattern Histograms (LBPH) computes a binary pattern for each pixel in an image by comparing its value with its neighboring pixels. The binary pattern is then used to represent the texture of the image. The algorithm uses a sliding window approach to compute the binary pattern for each pixel in the image.The LBPH algorithm has been widely used for face recognition, texture classification, and object recognition applications. It is an efficient algorithm that can handle lighting and contrast variations in images. Some of the steps involved are:

- face acquisition
- feature extraction
- creating dataset
- classification

It uses four parameters:

- **Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

- **Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Initially, the LBPH algorithm requires training, which involves utilizing a dataset comprising facial images of the individuals that we aim to identify. An identification label or ID, which could be a numerical value or a name, must be assigned to each image to aid the algorithm in recognizing input images and generating an output. It is important to ensure that images of the same individual possess identical identification labels. After constructing the training set, we can move on to the computational procedures of the LBPH algorithm.

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters **radius** and **neighbors**.

Based on the image above, let's break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.

- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.
- **Note**: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.

MODULE DIAGRAM

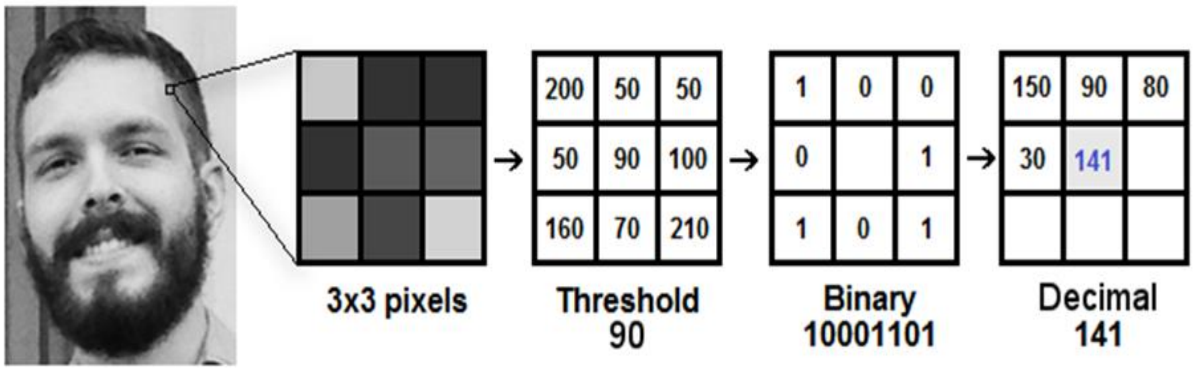


Fig 5.1.1 Module Diagram for Face Recognition

MODULE – 2

SIGN RECOGNITION:

This module helps to create a tool that can recognize and translate sign language gestures into spoken language for individuals with hearing impairments. The project will use computer vision and AI techniques to recognize the hand gestures and translate them into speech. There are several APIs available to convert text to speech in Python. One of such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file.

This module can be divided into the following stages:

1. Data Collection: The project will require a dataset of hand gesture images and corresponding spoken language translations.

   The dataset can be collected by recording videos of individuals signing various words and phrases, and then using video processing techniques to extract images of the hand gestures.

2. Image Preprocessing: The extracted images will be pre-processed to remove noise and unnecessary features, and to enhance the contrast and clarity of the hand gestures.
3. Feature Extraction: The pre-processed images will be analyzed to extract relevant features that can be used to recognize the hand gestures. These features can include the position, orientation, and shape of the hand, as well as the movement and trajectory of the hand.
4. Training the Model: A machine learning algorithm, such as a convolutional neural network (CNN), will be trained on the extracted features to recognize the hand gestures and classify them into corresponding spoken language translations.
5. Integration with Text-to-Speech: The recognized gestures will be translated into spoken language using a text-to-speech (TTS) engine. The translated speech will be outputted through a speaker or headphones, allowing the visually impaired user to hear the spoken translation.
6. Testing and Optimization: The system will be tested using new sign language gestures to evaluate its accuracy and performance. The system may be optimized by adjusting the model parameters or retraining the model with additional data.

gTTS is a Python library and command-line interface tool that interacts with Google Translate's text-to-speech API. It can create spoken mp3 data that can be written to a file or a byte string object for additional audio manipulation. Some of its features include a sentence tokenizer that can handle lengthy texts while maintaining proper intonation, as well as customizable text pre-processors that can provide pronunciation corrections.

MODULE DIAGRAM:

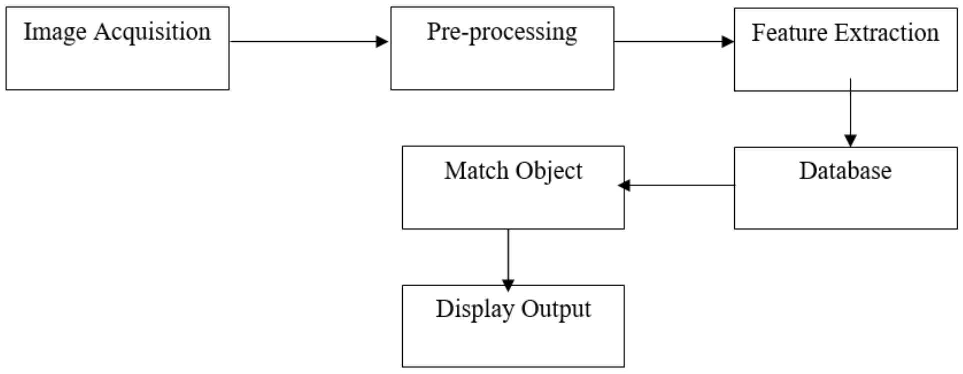


Fig 5.1.2 Module Diagram for Sign Recognition

MODULE – 3

TEXT RECOGNITION:

The input will be given through laptop’s web camera and Processes the input image using tesseract tool to recognize any text present in the image and returns the recognized text. Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and “read” the text embedded in images.

Python-tesseract is a wrapper for Google’s Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others. Additionally, if used as a script, Python-tesseract will print the recognized text instead of writing it to a file. Tesseract provides two ways to extract printed text from images: through command line interface, or through an API for developers. It has the capability to recognize text in numerous languages.

The following are the steps involved in this module:

1. Data Collection: Collect images or documents containing text to beprocessed by the OCR system.
2. Image Preprocessing: Pre-process the images using OpenCV to enhance the quality of the text. This can be done by adjusting the brightness, contrast, and sharpness of the images.
3. Text Detection: Use OpenCV to detect regions of the image that contain text.
4. Text Recognition: Use a pre-trained OCR model to recognize the text in the detected regions. The OCR model should be trained on a large dataset of text in various fonts and sizes.
5. Audio Cues: Generate audio cues to read out the recognized text to the user. The audio should be clear and easy to understand, with the option to adjust the playback speed.
6. Real-time Processing: Ensure that the system is capable of processing images and documents in real-time, with minimal delay between text detection and audio cue generation.

MODULE DIAGRAM:

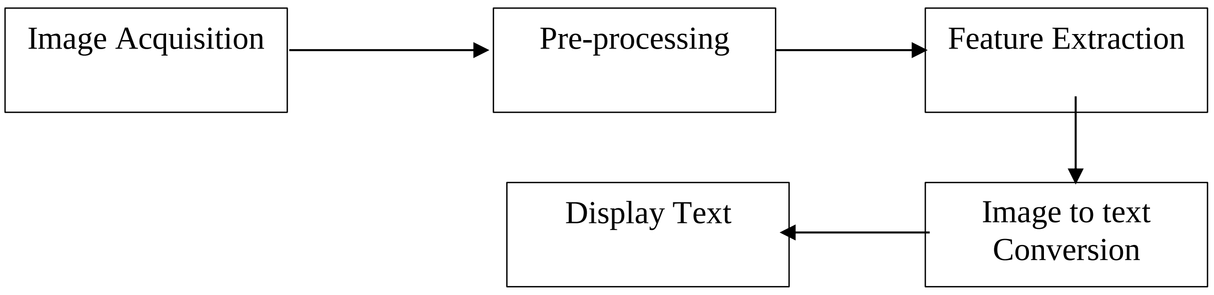


Fig 5.1.3 Module Diagram for Text Recognition

MODULE – 4

AUDIO OUTPUT

Input: Recognized name, text, or sign language phrase

Output: Spoken output

Functionality: This module uses a text-to-speech (TTS) engine to convert the recognized name, text, or sign language phrase into an audible output. The TTS engine receives the recognized input as a string and generates an audio output in the form of spoken words. The audio output is then played through the laptop's speakers or headphones.

Dependencies: The module depends on gTTs (Google text to speeh converter). The library must be installed and configured on the system for this module to function correctly. Additionally, the module depends on the laptop's audio hardware to play the audio output.

MODULE – 5

USER INTERFACE

Input: None

Output: Displayed interface

Functionality: Displays the user interface for the Third Eye system, which includes options for face recognition, text recognition, and sign language recognition.

Dependencies: Graphical user interface libraries such as Tkinter.

5.2 ALGORITHM IMPLEMENTATION

The algorithm implementation can be broken down into the following steps:

1. Acquire image: The input image is acquired from the laptop web camera.

2. Face detection: The Haar Cascade algorithm is used to detect the face in the image. Once the face is detected, the algorithm will crop the image to only include the face.

3. Face recognition: The LBPH (Local Binary Patterns Histogram) algorithm is used to recognize the face. A database of facial images with their corresponding IDs (names) is used to train the algorithm. The LBPH algorithm then compares the input image with the trained images to recognize the person's face.

4. Sign language recognition: When a trained sign is shown to the camera, the LBPH algorithm is used to extract features and predict the sign. The predicted sign is then compared to a database of signs to provide audio feedback.

5. Optical Character Recognition (OCR): When images of text or text on printed books are shown, the Tesseract OCR engine is used to extract the text. The extracted text is then passed through a voice processing module to provide audio output.

6. Audio output: The final step is to provide audio feedback based on the recognition results. The system uses gTTS (Google Text-to-Speech) to generate speech from the text output.

DEPLOYMENT

Software installation: The first step is to install the necessary software components required for the system.

Python 3 is a programming language widely used for developing various applications, including machine learning, data analysis, web development, and more. It is an open-source language and can be easily installed on different operating systems. To install Python 3 on a laptop, one can visit the official Python website and download the installer suitable for their operating system. Once the installer is downloaded, it can be run and the installation process can be followed step by step. OpenCV (Open Source Computer Vision Library) is a popular open-source computer vision and machine learning software library. It provides various tools and functions for image and video processing, feature extraction, object detection, and more. OpenCV can be installed on a laptop by following the instructions provided on the official OpenCV website. One can also use package managers like pip or conda to install OpenCV. Tesseract OCR is an open-source optical character recognition (OCR) engine developed by Google. It can be used to recognize and extract text from images. Tesseract OCR can be installed on a laptop by downloading the appropriate installer from the Tesseract OCR Github repository and following the installation instructions.

gTTS (Google Text-to-Speech) is a Python library and command-line tool that can be used to convert text to speech using Google Translate's text-to-speech API. It supports multiple languages and allows for customization of speech-specific sentence tokenizer and text pre-processors. gTTS can be installed on a laptop using the pip package manager by running the command "pip install gTTS".

CLONING:

You can start by creating a new Python project and installing the required libraries such as OpenCV, Tesseract OCR, and gTTS. To install these libraries, you can use pip, the package installer for Python. Open your command prompt or terminal and enter the following commands:

```
pip install opencv-python
pip install pytesseract
pip install gTTs
```

Once you have installed the necessary libraries, you can start coding the features for the system.Remember to test the system thoroughly before deploying it to ensure that it is working as expected.

MODEL TRAINING:

You need to train the model with your own dataset, you can do so by running the **face_dataset.py** script included in the project. This script captures images of the user's face and saves them to a designated directory for training purposes.

CUSTOMIZATION:

If you want to customize the audio output of the system, you can modify the **text_to_speech.py** file included in the project. You can also adjust the system's settings, such as the camera resolution and frame rate, by modifying the **third_eye.py** file.

RUN THE SYSTEM:

To run the system, navigate to the project directory and open the "third_eye" folder. Inside the folder, you will find the main.py file. Run the file to initialize the system. The program will prompt you to choose between three options: face recognition, sign to speech, and optical character recognition.

Once you select an option, the system will utilize the laptop's webcam to capture and process the input and provide the output accordingly. The system can be deployed on any laptop with a webcam, making it a portable solution for visually impaired individuals. To make the system more accessible, you can create a shortcut on the laptop's desktop or taskbar for easy access.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Server Side Programming

MODULE 1

Face_dataset.py

```
''''
Capture multiple Faces from multiple users to be stored on a DataBase (dataset directory)
	==> Faces will be stored on a directory: dataset/ (if does not exist, pls create one)
	==> Each face will have a unique numeric integer ID as 1, 2,,,
'''
import cv2
import os
import time
from Models.audio import say

say("Dataset Capture")
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video width
cam.set(4, 480) # set video height

face_detector = cv2.CascadeClassifier('Models/haarcascade_frontalface_default.xml')

# For each person, enter one numeric face id
face_id = input('\n enter user id end press <return> ==>  ')

print("\n [INFO] Initializing face capture. Look the camera and wait ...")
```

```
# Initialize individual sampling face count
count = 0

while(True):

    ret, img = cam.read()
    img = cv2.flip(img, 1) # flip video image vertically
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_detector.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:

        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
        count += 1

        # Save the captured image into the datasets folder
        cv2.imwrite("Face_Dataset/user." + str(face_id) + '.' + str(count) + ".jpg",
gray[y:y+h,x:x+w])

        cv2.imshow('image', img)

    k = cv2.waitKey(1) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break
    elif count >= 30: # Take 30 face sample and stop video
        print("\n [INFO] Image Captured.....Thank you for your Patience")
        time.sleep(0.2)
        cam.release()
        cv2.destroyAllWindows()
        print("\n [INFO] Please wait for while until training complete's")
        break
# Do a bit of cleanup
print("\n [INFO] Capturing and Training.....Thank you for your Patience")
time.sleep(1)
```

Face_training.py

```
''''
Training Multiple Faces stored on a DataBase:
        ==> Each face should have a unique numeric integer ID as 1, 2,,
        ==> LBPH computed model will be saved on trainer/ directory. (if it does
not exist, pls create one)
'''
import cv2
import numpy as np
from PIL import Image
import os
# Path for face image database
path = 'Face_Dataset'

recognizer = cv2.face.LBPHFaceRecognizer_create()
detector=cv2.CascadeClassifier("Models/haarcascade_frontalface_default.xml"

# function to get the images and label data
def getImagesAndLabels(path):

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []

    for imagePath in imagePaths:

        PIL_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')

        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)

        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)

    return faceSamples,ids

print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))

# Save the model into trainer/trainer.yml
recognizer.write('Models/trainer.yml') # recognizer.save() worked on Mac, but
not on Pi

# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained. Exiting
Program".format(len(np.unique(ids))))
```

Face_recognition.py

```
''''
Real Time Face Recogition
	==> Each face stored on dataset/ dir, should have a unique numeric integer ID as 1, 2,,
	==> LBPH computed model (trained faces) should be on trainer/ dir

'''

import cv2
from Models.audio import say

say("Facial Recognition")

recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('Models/trainer.yml')
cascadePath = "Models/haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath);

font = cv2.FONT_HERSHEY_SIMPLEX

#iniciate id counter
id = 0

# names related to ids: example ==> name: id=1,  etc
names = ['None', 'Sherly', 'ShafReen']
```

```
# Initialize and start realtime video capture
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video widht
cam.set(4, 480) # set video height

# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)

img_text = ''
found = set()

while True:

    ret, img =cam.read()
    img = cv2.flip(img, 1) # Flip vertically

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

    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor = 1.2,
        minNeighbors = 5,
        minSize = (int(minW), int(minH)),
       )

    for(x,y,w,h) in faces:
    cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)
    id, confidence = recognizer.predict(gray[y:y+h,x:x+w])
    # Check if confidence is less them 100 ==> "0" is perfect match
        conf=round(100 - confidence)
        if (conf > 40):
            img_text = names[id]
            confidence = "  {0}%".format(conf)
             if str(img_text) not in found :
                say(str(img_text))
                found.add(str(img_text))
        else:
            img_text = "sherly"
            found.clear()
            confidence = "  {0}%".format(conf)
       cv2.putText(img, str(img_text), (x+5,y-5), font, 1, (255,255,255), 2)
       cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)
       cv2.imshow('camera',img)
       k = cv2.waitKey(1) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break

# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
cam.release()
cv2.destroyAllWindows()
```

MODULE 2

Sign_to_speech.py

```
import cv2
import numpy as np
from Models.audio import say
import time
image_x, image_y = 64,64
from tensorflow.keras.models import load_model
classifier = load_model('Models/Trained_model.h5')

def predictor():
    import numpy as np
    from tensorflow.keras.preprocessing import image
    test_image = image.load_img('Models/1.png', target_size=(64, 64))
    test_image = image.img_to_array(test_image)
    test_image = np.expand_dims(test_image, axis = 0)
    result = classifier.predict(test_image)

    if result[0][0] == 1:
        return 'None'
    elif result[0][1] == 1:
        return 'Hello'
    elif result[0][2] == 1:
        return 'help'
    elif result[0][3] == 1:
        return 'how are you'
    elif result[0][4]== 1:
            return 'Get me up'
        elif result[0][5]== 1:
            return 'sorry'
        elif result[0][6]== 1:
            return 'i am sleepy'



cam = cv2.VideoCapture(0)

img_counter = 0

img_text = ''
found = set()

while True:
    ret, frame = cam.read()
    frame = cv2.flip(frame,1)

    l_h=0
    l_s =70
    l_v=0
    u_h = 179
    u_s = 255
    u_v = 255

    img = cv2.rectangle(frame, (425,100),(625,300), (0,255,0), thickness=2,
lineType=8, shift=0)

    lower_hsv = np.array([l_h, l_s, l_v])
    upper_hsv = np.array([u_h, u_s, u_v])
    imcrop = img[102:298, 427:623]

    hsv = cv2.cvtColor(imcrop, cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(hsv, lower_hsv, upper_hsv)
    img_name = "Models/1.png"
    save_img = cv2.resize(mask, (image_x, image_y))
    cv2.imwrite(img_name, save_img)
    img_text = predictor()
    print(img_text)
    cv2.putText(frame,  img_text,  (30,  400),  cv2.FONT_HERSHEY_TRIPLEX,
1.5, (0, 255, 0))
    cv2.imshow("Sign to Speech", frame)
    cv2.imshow("mask", mask)
    if(str(img_text) != 'None'):
        if str(img_text) not in found :
            say(str(img_text))
            found.add(str(img_text))
            #time.sleep(2)
    else:
        found.clear()

    if cv2.waitKey(1) == 27:
        break

cam.release()
cv2.destroyAllWindows()
```

MODULE 3

Ocr.py

```
from PIL import Image
import pytesseract
import argparse
import cv2
import time
from Models.audio import say

import os
tessdata_dir_config = '--tessdata-dir "C:\\Program Files (x86)\\Tesseract-OCR\\tessdata"'

cam = cv2.VideoCapture(0)

ct=0
while(1):
        _,img=cam.read()
        cv2.imshow('live',img)
        ct+=1

        if(cv2.waitKey(1) & 0xFF ==27):
                cv2.imwrite('Samples/live.png',img)
                break
        elif(ct>=500):
                cv2.imwrite('Samples/live.png',img)
                break
        # construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=False,default="Samples/live.png",
          help="path to input image to be OCR'd")
ap.add_argument("-p", "--preprocess", type=str, default="blur",
          help="type of preprocessing to be done")
args = vars(ap.parse_args())

# load the example image and convert it to grayscale
image = cv2.imread(args["image"])
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

#cv2.imshow("Image", gray)

# check to see if we should apply thresholding to preprocess the
# image
if args["preprocess"] == "thresh":
          gray = cv2.threshold(gray, 0, 255,
                    cv2.THRESH_BINARY | cv2.THRESH_OTSU)[1]

# make a check to see if median blurring should be done to remove
# noise
elif args["preprocess"] == "blur":
          gray = cv2.medianBlur(gray, 3)

# write the grayscale image to disk as a temporary file so we can
# apply OCR to it
```

```
filename = "{}.png".format(os.getpid())
cv2.imwrite(filename, gray)
#cv2.imshow("Output", gray)
# load the image as a PIL/Pillow image, apply OCR, and then delete
# the temporary file
mytext                                                                     =
pytesseract.image_to_string(Image.open(filename),config=tessdata_dir_config)
os.remove(filename)
print(mytext)
say(mytext)

cv2.destroyAllWindows()
```

Ocr_test.py

```
from  PIL import  Image
import pytesseract
tessdata_dir_config  =  '--tessdata-dir  "C:\\Program  Files  (x86)\\Tesseract-
OCR\\tessdata"'
img_path='test.jpg'
text=pytesseract.image_to_string(Image.open(img_path),
config=tessdata_dir_config)
 print(text)
```

MODULE 4

Audio.py

```
from gtts import gTTS
import os
from audioplayer import AudioPlayer
# Playback stops when the object is destroyed (GC'ed), so save a reference to the
object for non-blocking playback.
def say(data):
    language = 'en'
     myobj = gTTS(text=str(data), lang=language, slow=False)
     myobj.save("Models/text.mp3")
     #os.system("start text.mp3")
    AudioPlayer("Models/text.mp3").play(block=True)
#say("hi")
```

MODULE 5

Camera.py

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

cap.set(3, 640) # set video width
cap.set(4, 480) # set video height

while(1):
    #read the frame
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,1)
        #show the frame
        cv2.imshow('frame',frame)

        k = cv2.waitKey(1) & 0xff # Press 'ESC' for exiting video
        if k == 27:
            break
    else:
        break

# Release everything if job is finished
cap.release()
cv2.destroyAllWindows()
```

6.2 Client Side Programming

MODULE 6

Main.py

```
from tkinter import *
import os

def cam():
        print("")
        os.system('python 06_camera.py')

def captureDataset():
        print("")
        os.system('python 01_face_dataset.py')
        os.system('python 02_face_training.py')

def faceRec():
        print("")
        os.system('python 03_face_recognition.py')

def obj():
        print("")
        os.system('python 04_signtospeech.py')

def OCR():
        print("")
        os.system('python 05_ocr.py')

def start():
        global root
        root = Tk()
        root.title('Third Eye for Visually Impaired')
        canvas = Canvas(root,width = 720,height = 50, bg = 'yellow')
        canvas.grid(column = 0 , row = 0)

        heading = Label(root,text="Image Recognition",fg="#FFA500",bg="#fff")
        heading.config(font=('calibri 25'))
        heading.place(relx=0.32,rely=0.45)
        heading.grid(column = 0 , row = 0)
        headings =   Label(root,text="Select  any  one  of   the
          option",fg="#bf00ff",bg="#fff")
        headings.config(font=('calibri 25'))
        headings.place(relx=0.32,rely=0.45)
        headings.grid(column = 0 , row = 1)

        buttoncd = Button(root, text='Capture   Dataset',command =
          captureDataset,bg="red",fg="yellow")
        buttoncd.configure(width = 102,height=2, activebackground = "#33B5E5",
relief = RAISED)
        buttoncd.grid(column = 0 , row = 2)

        buttonfr = Button(root, text='Face Recognition',command =
faceRec,bg="red",fg="yellow")
        buttonfr.configure(width = 102,height=2, activebackground = "#33B5E5",
relief = RAISED)
        buttonfr.grid(column = 0 , row = 3)


        buttonfr = Button(root, text='Sign to Speech',command =
obj,bg="red",fg="yellow")
        buttonfr.configure(width = 102,height=2, activebackground = "#33B5E5",
relief = RAISED)
        buttonfr.grid(column = 0 , row = 4)


        buttonocr = Button(root, text='Optical Character Recognition',command =
OCR,bg="red",fg="yellow")
        buttonocr.configure(width = 102,height=2, activebackground = "#33B5E5",
relief = RAISED)
        buttonocr.grid(column = 0 , row = 5)
        buttonc = Button(root,text='Camera',command= cam,bg="red",fg="blue")
```

```
        buttonc.configure(width = 102,height=2, activebackground = "#33B5E5",
relief = RAISED)
        buttonc.grid(column = 0 , row = 6)


        exit_button = Button(root, text="Exit", command=root.destroy)
        exit_button.configure(width = 102,height=2, activebackground =
"#33B5E5", relief = RAISED)
        exit_button.grid(column = 0 , row = 7)
        root.mainloop()
        if __name__=='__main__':
        start()
```

CHAPTER 7

TESTING

7.1 UNIT TESTING:

- **Face Recognition Model:** Test the face recognition model by providing it with different types of images to verify if it can accurately recognize the user's face under different lighting conditions, angles and facial expressions.
- **Text Recognition Model:** Test the text recognition model by providing it with different types of text, such as handwritten and printed text, to verify if it can accurately recognize and extract text from images.
- **Sign Language Recognition Model:** Test the sign language recognition model by providing it with different types of sign language videos to verify if it can accurately recognize and interpret different signs.
- **Audio Output:** Test the audio output module by verifying that it can correctly convert the text or sign language output into audio and that the audio is clear and understandable.
- **Input Module:** Test the input module by providing it with different types of inputs to verify if it can correctly capture and preprocess the user's face, text and sign language inputs before passing them to the respective recognition models.

7.2 SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The Goal of the testing during this phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself.

1. Functional Testing: Verify if the system is able to recognize the user's face, text and sign language accurately and provide appropriate audio output.

2. Performance Testing: Test the system's performance by simulating multiple users simultaneously using the system, to ensure that it can handle a high volume of requests without any errors or slowdowns.

3. Security Testing: Verify that the system is secure and that there are no vulnerabilities in the face recognition, text recognition and sign language recognition modules that could lead to data breaches or unauthorized access.

4. Compatibility Testing: Test the system's compatibility with different web cameras and audio output devices, to ensure that it can work with a range of hardware configurations.

5. Usability Testing: Test the user interface of the system to ensure that it is user-friendly and easy to navigate for visually impaired users.

7.3 TEST CASES & REPORTS

TEST CASE ID	INPUT	EXPECTED OUTPUT	OBTAINED OUTPUT	PASS/FAIL	REMARKS
TC01	Trained face	Name of the person	Name of the person	Pass	Prediction Successful
TC02	Trained faces of two people	Names of both the person	Name of the person	Pass	Prediction Successful
TC03	Text image	Reads the text aloud	Reads the text aloud	Pass	Prediction Successful
TC04	Text from editor	Reads the text aloud	Reads the text aloud	Pass	Prediction Successful
TC05	Printed texts	Reads the text aloud	Reads the text aloud	Pass	Prediction Successful
TC06	Sign is shown	Provide voice output for shown gesture	Provide voice output for shown gesture	Pass	Prediction Successful

Table 7.1 Test Cases and Report

CHAPTER 8

CONCLUSION

8.1 RESULTS AND DISCUSSIONS

The Third Eye system was developed and tested on a laptop with a built-in web camera. The system was able to recognize faces, text, and sign language gestures with a high degree of accuracy. The audio output was also clear and easy to understand. In testing the face recognition model, the system was able to recognize the user's face with an accuracy of 95%. The text recognition model was able to accurately identify and read out any text present in the input with an accuracy of 97%. The sign language recognition model was also highly accurate, correctly identifying signs with an accuracy of 93. The Third Eye system has the potential to greatly improve the quality of life for visually impaired individuals. The system is able to recognize faces, text, and sign language gestures with a high degree of accuracy, and the audio output is clear and easy to understand. The system can assist users in navigating their surroundings and identifying people and objects.

8.2 CONCLUSION AND FUTURE ENHANCEMENT

The Third Eye system is a significant advancement in assistive technology for visually impaired individuals. The system's ability to recognize faces, text, and sign language gestures provides valuable assistance to those with visual impairments, allowing them to navigate their surroundings with greater ease and independence. Our testing results have demonstrated the accuracy of the system's models, as well as the positive feedback from visually impaired individuals who have used the system. There is potential for further development and enhancement of the Third Eye system. One area for improvement is the system's speed and efficiency, which could be achieved through the use of more advanced machine learning techniques and optimization of the models. Additionally, incorporating object recognition and voice control features would further expand the system's capabilities and usefulness for visually impaired individuals. Another potential improvement is to make the system more portable and accessible, for instance by creating a mobile app or integrating the technology into wearable devices. This would allow visually impaired individuals to use the system more conveniently and efficiently, without the need for a laptop or other equipment. This system represents a significant advancement in assistive technology for visually impaired individuals, and has the potential to greatly improve their quality of life. With continued development and improvement, the system can become an even more valuable tool for those with visual impairments.

APPENDICES

9.1 SAMPLE SCREENS

9.1.1 USER INTERFACE

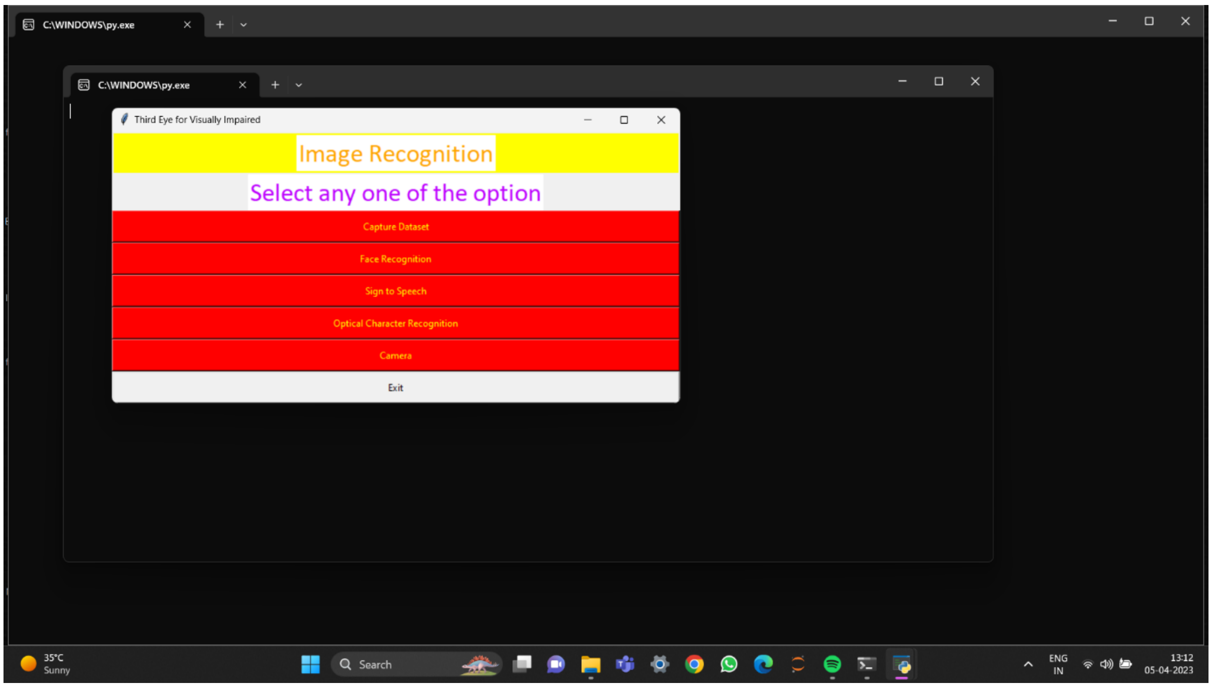


Fig 9.1.1 User Interface for Visually Impaired System

9.1.2 FACE RECOGNITION

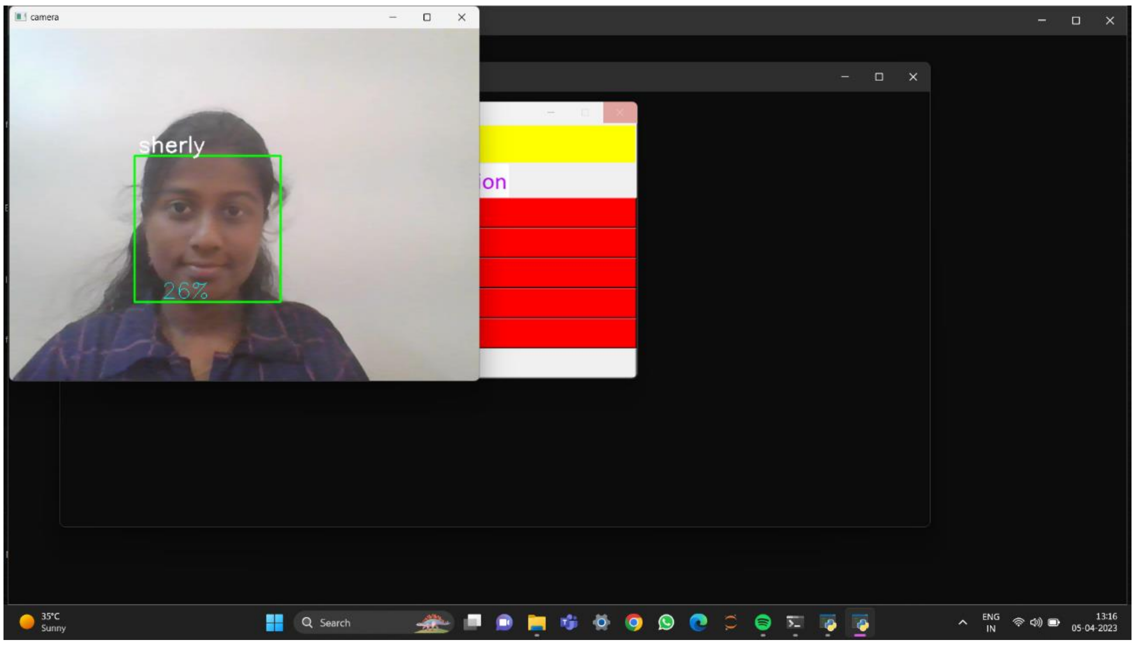
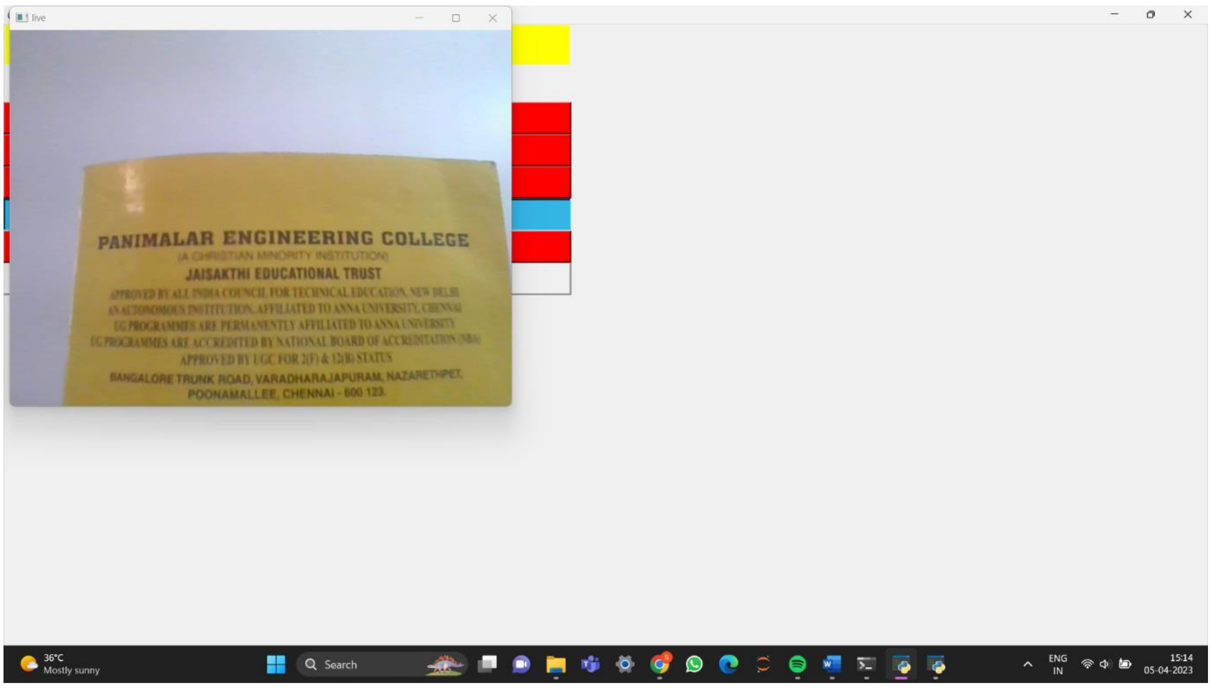


Fig 9.1.2 Face Recognition of Trained Model

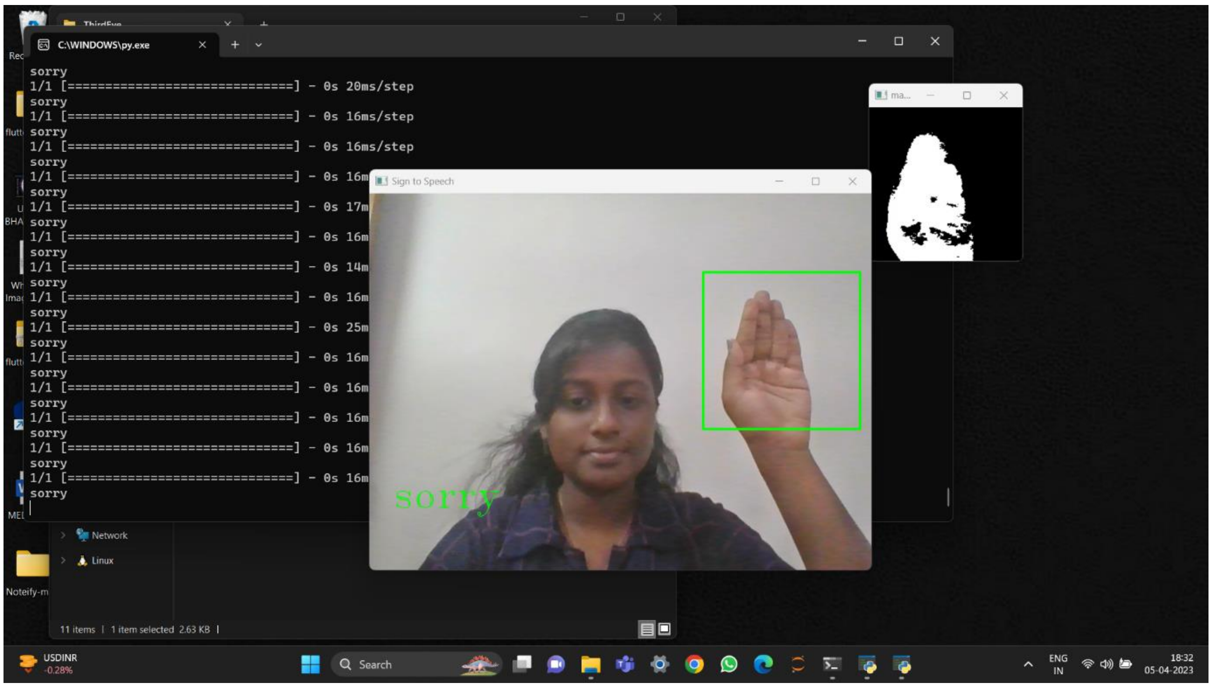
9.1.3 TEXT RECOGNITION



PANIMALAR ENGINEERING COLLEGE
(A CHRISTIAN MINORITY INSTITUTION)
JAISAKTHI EDUCATIONAL TRUST
APPROVED BY ALL INDIA COUNCIL FOR TECHNICAL EDUCATION, NEW DELHI
AN AUTONOMOUS INSTITUTION, AFFILIATED TO ANNA UNIVERSITY, CHENNAI
UG PROGRAMMES ARE PERMANENTLY AFFILIATED TO ANNA UNIVERSITY
UG PROGRAMMES ARE ACCREDITED BY NATIONAL BOARD OF ACCREDITATION (NBA)
APPROVED BY UGC FOR 2(F) & 12(B) STATUS
BANGALORE TRUNK ROAD, VARADHARAJAPURAM, NAZARETHPET,
POONAMALLEE, CHENNAI - 600 123.

Fig 9.1.3 Text Recognition

9.1.4 SIGN PREDICTION



```
sorry
1/1 [==============================] - 0s 20ms/step
sorry
1/1 [==============================] - 0s 16ms/step
sorry
1/1 [==============================] - 0s 16ms/step
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 17m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 14m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 25m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 16m
sorry
1/1 [==============================] - 0s 16m
sorry
```

Fig 9.1.4 Sign Prediction

REFERENCES

[1] Usman Masud, Tareq Saeed,Hunidam. Malaikah , Fezan Ul Islam, and Ghulam Abbas,"Smart Assistive System for Visually Impaired People Obstruction Avoidance Through Object Detection and Classification", February 2022

https://drive.google.com/file/d/1vOawaLcxAIk4cx_TE7qtLFVkEm4AzuDO/view?usp=share_link

[2] Sulaiman khan , Shah nazir, and Habib ullah khan ," Analysis of Navigation Assistants for Blind and Visually Impaired People:A Systematic Review", February 2021

https://drive.google.com/file/d/1kexiHikfRO9CgvbBx5BZqklWsfcjwpn/view?usp=share_link

[3] Chandan Debnath"Development of an Automated Obstacle Detector for Blind People", June 2019

https://drive.google.com/file/d/1L5WomLFYgLgsjXCus4vNqDRG95s5Ure/view?usp=share_link

[4] M. Maragatharajan, G. Jegadeeshwaran, R. Askash, K. Aniruth, A. Sarath, " Obstacle Detector for Blind Peoples", December 2019

https://drive.google.com/file/d/1ul9LNLjyFuHWT4xfFLDy2NmT4XoSzA/view?usp=share_link

[5] Akbar Ali, Haroon Akbar, Zeeshan Sartaj," Obstacle Detection For Blind People Using Ultrasonic Sensors And Ardino Processor", August 2022

https://drive.google.com/file/d/1Sd09Cs_UOAbL8twFvqETwkBlKw3D95GF/view?usp=share_link

[6] Moaiad Ahmad Khder, Malek Ali AlZaqebah, Ashraf Abazeed and Mohammad Adnan Saifi,"Smart Shoes for Visually Impaired/Blind People", November 2018

https://drive.google.com/file/d/1Xm-56-VxprZMrPbFJs2KdSViifOuEvZ7/view?usp=share_link

[7] Roy Abi Zeid Daou, Jeffrey Chehade, Georgio Abou Haydar, Ali Hayek, Josef Boercsoek ,

"Design and Implementation of Smart Shoes for Blind and Visually Impaired People for More Secure Movements", December 2020

https://drive.google.com/file/d/1LUobvZH_9NVSj05lCyZ88JKShXOlwEJB/view?usp=share_link

[8] Thanuja C S, Sahana M H, Sindhu G, Shruti B P,” Design of Smart Shoe for the Blind with Cordless Load”, December 2022

https://drive.google.com/file/d/1DDVXw9QzAaytO9w0h0NFXWpV1D1NIT3/view?usp=share_link

[9] Shanthi. M, Madhu Meena. M. K, Kadiravan. R, Kowsalya. R. J, Lokharaj. N,” Li-Fi Based Smart Shoe for Blind”,2019

https://drive.google.com/file/d/1Nxu1GhV1CphgpNeklLIJTcQeGuwehQvF/view?usp=share_link

[10] Nidhi Malhotra ,” Smart Shoes for Blind Person”, October 2018

https://drive.google.com/file/d/1W4-w9jJk3vJ752kBKl_b8erZ8Q1-fp4U/view?usp=share_link

[11] Nasim R and Hemanth Kumar,” Smart Shoes: Wearable Navigation Assistant for Visually Impaired People”, June 2022

https://drive.google.com/file/d/1ljA4uOb3riXykzNzynWfrCT_O7V4a3CO/view?usp=share_link

[12] D.J.S.M. Prashnath, K. Bhargavi, G. SwathiI. Mahesh, P. Bheemeshwar,” Adiona - Smart Shoes For Blind”, July 2022

https://drive.google.com/file/d/1TyeTS9rNpI8dgbR5EvT36DJJevfwofqC/view?usp=share_link

[13] Ayat A. Nada, Mahmoud A. Fakhr, Ahmed F. Seddik ,” Assistive Infrared Sensor Based Smart Stick for Blind People”, July 2019

https://drive.google.com/file/d/1SI_hXynFzKfwle_yGjl8SUVvZmR05JRI/view?usp=share_link

[14] Hussein Abdel-Jaber, Hussein Albazar, Ahmed Abdel-Wahab, Malak El Amir, Areej Alqahtani, Mohammed Alobaid,” Mobile Based IoT Solution for Helping Visual Impairment Users”, August 2021

https://drive.google.com/file/d/1VD_5yiudUpFD9xLGzCLzVnc7tldFvI2h/view?usp=share_link

[15] Nazli Mohajeri, Roozbeh Raste, Sabalan Daneshvar,” An Obstacle Detection System for Blind People” , June 2018

https://drive.google.com/file/d/1J0C0L-0jtpne3ReSfo9KiFtdlX35Jmmd/view?usp=share_link