FACE RECOGNITION GLASSES

A Project Report Submitted By

NIHAR RAVI MAJALIKAR	**4NM19EC103**
PRAKHAR MISHRA	**4NM19EC118**
ROVEL NAZARETH	**4NM19EC141**
SAMARTH GUPTA	**4NM19EC146**

Under the Guidance of

Dr. Mamatha Girish
Associate Professor

in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Engineering
in
Electronics and Communication Engineering

from
Visvesvaraya Technological University, Belagavi

NITTE EDUCATION TRUST | NMAM INSTITUTE OF TECHNOLOGY

Nitte-574110, Karnataka, India

May-2023

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
ISO 9001:2015 Certified, Accredited with 'A' Grade by NAAC
☎:08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Electronics and Communication Engineering

Certificate

Certified that the project work entitled
*"**Face Recognition Glasses**"*
is a bonafide work carried out by
Nihar Ravi Majalikar (4NM19EC103), Prakhar Mishra (4NM19EC118), Rovel Nazareth (4NM19EC141) & Samarth Gupta (4NM19EC146)
*in partial fulfillment of the requirements for the award of **Bachelor of Engineering Degree in Electronics and Communication Engineering***
*prescribed by **Visvesvaraya Technological University, Belagavi***
during the year 2022-2023.
It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library.
The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of the Guide **Signature of the HOD**

Signature of the Principal

Semester End Viva Voce Examination

	Name of the Examiners	Signature With Date
1.	_____	_____
2.	_____	_____

Abstract

According to the World Health Organization (WHO) results, 285 million people worldwide are visually impaired, among them 39 million people are blind and 246 million have low vision according to 2011 statistics. Therefore, this project intends to provide reliable and cost-efficient solution for blind people which would help them to travel independently without any major problem. In these glasses, the main component which integrates all the other components in it, is the Raspberry Pi. It is also responsible to run the code in the project including all the libraries which have been used. The face detection part is captured by the camera pi module, which captures the image then recognizes the image captured and compares it with an array of known faces. The ultrasonic sensor is used for measuring the distance between the glasses and the object and the final feedback is delivered in the form of audio output through the earphones to the user wearing it. The Raspberry Pi is back-up by a 5V battery/Power Bank.

Acknowledgement

Our project would not have been successful without the encouragement, guidance and support by various personalities. First and foremost, we would like to express our sincere gratitude towards our project guide **Dr. Mamatha Girish**, Associate Professor, Department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for her guidance, encouragement and inspiration throughout the project work.

We extend our gratitude to **Dr. K. V. S. S. S. S. Sairam**, Professor and Head, Department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for his encouragement and providing necessary facilities in the department.

We wish to acknowledge the support of **Dr. Niranjan N. Chiplunkar,** Principal, N. M. A. M. Institute of Technology, Nitte, for providing motivational academic environment.

We would like to express our sincere thanks to all the teaching and non-teaching staff of the department of Electronics and Communication Engineering, N. M. A. M. Institute of Technology, Nitte, for their patience and help during our project work.

Finally, we would like to thank all our friends and well-wishers who have helped us whenever needed and supported us.

Nihar Majalikar
Prakhar Mishra
Rovel Nazareth
Samarth Gupta

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 General Introduction

Human visual system plays an important role in recognizing information regarding surroundings. Since visual signals provide with more data than auditory information, visual signals are more effective than auditory signals for the human being to perceive information. However, in case of blind people the lack of visual information constrains them in recognizing information. For a blind person to recognize a subject around him is subjective to the idea put forth in oral format. In addition, even when the subject speaks, it is difficult for the blind person to recognize the subject. In a situation that the blind person meets people in the corridor, it is difficult for the blind person to recognize. If the blind person receives information in the form of auditory or tactile sense, they can recognize the person. Blind people face a lot of problems in their daily life to understand environmental conditions and identifying the people around. Obstacles which are not hazardous to ordinary people, may become fatal to them. In this context, the prototype proposes a solution for object detection and face recognition for visually impaired and blind people. The Smart eye which helps the blind and visually impaired people to commute freely by experiencing their surroundings. The working starts by absorbing the scenarios around the person and detecting them using camera pi module. The device thereafter recognizes the image captured and compares it with an array of known faces. Face detection and recognition uses algorithm, coded by python open CV. The contents of the text file are converted to voice using the Text to Speech Synthesizer (TTS) software eSpeak[1].

1.2 Aim

To measure distance between the user and the obstacle using the ultrasonic sensor and giving the audio output of the distance measured through the earphones. Also to recognize the image captured in the frame of the video and search through the array of known faces and return the audio feedback through the earphones.

1.3 Objective

The objective of this project is to develop a system that can perform both face recognition and object detection using a single Raspberry Pi Model 3 microcontroller. The use of the HC-SR04 ultrasonic sensor provides an additional functionality of distance measurement, which can be useful in various applications. The system aims to accurately detect the presence of a face, extract facial features and compare them with a database to identify a person. The use of OpenCV and facial recognition algorithms is crucial in achieving this objective. The audio output feature through the eSpeak library provides a convenient way to communicate the results to the user. Overall, the goal of this project is to demonstrate the capabilities of the Raspberry Pi Model 3 as a versatile microcontroller for image processing and object detection.

1.4 Problem Formulation

For this project is to develop a system that can assist people with visual impairments in identifying objects and recognizing faces. People with visual impairments face challenges in their daily lives, such as navigating unfamiliar environments, identifying objects, and recognizing people. The existing solutions for these challenges are often expensive and not easily accessible to everyone. The use of a Raspberry Pi microcontroller, along with the HC-SR04 ultrasonic sensor and the Pi camera module, provides an affordable and accessible solution to these challenges. The system aims to accurately detect the presence of objects, measure their distance, and identify them using the audio output feature. Similarly, the facial recognition feature provides a means of identifying people and announcing their name. Overall, the problem formulation for this project is to provide an affordable and accessible solution for people with visual impairments to overcome the challenges they face in their daily lives.

1.5 Proposed Method

Raspberry Pi Model 3 microcontroller to run Python code that employs Haar Cascade and Local Binary Patterns algorithms for face detection and a Principal Component Analysis algorithm for face recognition. The system also uses an ultrasonic sensor for object detection and provides audio output through the eSpeak library. The advanced algorithms used enable accurate face and object detection, providing valuable information to people with visual impairments.

1.6 Methodology

The system that we use has 2 operations one is object detection and the other is face detection.

1.6.1 Object Detection

The object detection involves using an HC-SR04 ultrasonic sensor to measure the distance of an object from the sensor. The system calculates the distance using sound waves and provides an audio output of the distance through the eSpeak library. This provides a portable and affordable solution for people with visual impairments to detect the distance of objects in their environment, helping them navigate and interact with their surroundings more effectively.

1.6.2 Face Detection

Face Recognition in this project involves using a Raspberry Pi camera module to capture an image of a person's face, which is then processed using Haar Cascade and Local Binary Patterns algorithms for face detection. Once the face is detected, facial features such as the distance between the eyes, the shape of the nose, and the size of the mouth are extracted from the image using a Principal Component Analysis algorithm in the form of a face template. The face template is then compared with a database of face templates to find a match. If a match is found, the person's identity is identified and an audio output of their name is provided through the eSpeak library. This provides people with visual impairments a convenient and reliable method for recognizing faces, enabling them to interact with others more easily.

1.7 Literature Survey

A unique smart glass for visually impaired people to overcome the traveling difficulties. It can detect the obstacle and measure the distance perfectly using the ultrasonic sensor and a micro-controller. After receiving information from the environment, it passes to the blind person through a headphone. The GSM/GPRS SIM900A module is used to collect the information from the internet. A switch is connected to the system which is used for an emergency task like sending SMS, including time, temperature and location to the subject's guardian when visually impaired people fall into any danger. By using the smart glass visually impaired people can walk in an indoor and outdoor environment. With the explosive popularity of smart glasses, new technologies capable of extending their functionality

and applications become extremely important. Face recognition is a promising application for smart glasses due to its potential to assist a user in recalling names of people, whom the user has met before. It enables the user to compare a captured face image against a database of faces, determine the best match and return the associated information. The application can benefit law enforcing forces in tracking criminals, as well as help old people in reminding names of friends, relatives, and caretakers. The proposed smart-glass face-recognition system uses a high-speed remote server and smart-glass processing device (client) with video camera and display allocated on glasses. The server has access to data-base that contains face images labelled with corresponding information of the person. We assume that the server is activated before the user initiates face recognition application (from the smart-glass device). The client–server connection can be wired/wireless. For wireless, It is based on TCP/IP protocol and controlled by operating system through a programming interface. The TCP connection is established before starting the data transmission and released after ending the data transfer [1].

Face detection and recognition are key components in multiple camera-based devices and applications. Smart glasses are a type of optical head mounted displays that integrate first person cameras and hands free displays with immediate access to processing power able to analyze first person images in real time with hands free operation. In this context, an application prototype that detects and recognizes faces in real-time, and runs independently on the device. A description of the embedded implementation at a system-level where we highlight the application development challenges and trade-offs that need to be dealt with battery powered wearable devices. The implementation includes a parallel pipeline that reduces the latencies of the application [2].

To overcome the travelling difficulty for the visually impaired group, this paper presents a novel ETA (Electronic Travel Aids)-smart guiding device in the shape of a pair of eyeglasses for giving these people guidance efficiently and safely. Different from existing works, a novel multi-sensor fusion based obstacle avoiding algorithm is proposed, which utilizes both the depth sensor and ultrasonic sensor to solve the problems of detecting small obstacles, and transparent obstacles, e.g. the French door. For totally blind people, three kinds of auditory cues were developed to inform the direction where they can go ahead. Whereas for weak sighted people, visual enhancement which leverages the AR (Augment Reality) technique and integrates the traversable direction is adopted. The prototype consisting of a pair of display glasses and several low-cost sensors is developed, and its efficiency and accuracy were tested by a number of users. The experimental results show that the smart guiding glasses can effectively improve the user's travelling experience in complicated indoor environment. Thus it serves as a consumer device for

helping the visually impaired people to travel safely [3].

Based on local binary pattern (LBP) texture features, this research introduces a new and effective method of representing face images. The face image is divided into several regions from which the LBP feature distributions are extracted and concatenated into an enhanced feature vector to be used as a face descriptor. The performance of the proposed method is assessed in the face recognition problem under different challenges. Other applications and several extensions are also discussed. For real-time face identification and recognition in backdrops with complicated geometry, this paper offers efficient and reliable techniques. Ada Boost, cascade classifier, local binary pattern (LBP), Haar-like feature, face image preprocessing, and principal component analysis are a few of the signal processing techniques used to build the algorithms (PCA). A cascade classifier using the Ada Boost method is used to train face and eye detectors with high detection precision. To quickly recognize faces, facial features are extracted using the LBP descriptor. The false face detection rate is decreased using the eye detection algorithm. In order to retain high facial recognition accuracy, the detected facial image is subsequently processed to correct the orientation and boost the contrast. Finally, effective face recognition is achieved using the PCA algorithm [4].

This paper describes an ultrasonic sensor that is able to measure the distance from the ground of selected points of a motor vehicle. The sensor is based on the measurement of the time of flight of an ultrasonic pulse, which is reflected by the ground. A constrained optimization technique is employed to obtain reflected pulses that are easily detectable by means of a threshold comparator. Such a technique, which takes the frequency response of the ultrasonic transducers into account, allows a sub-wavelength detection to be obtained. Experimental tests, performed with a 40 kHz piezoelectric-transducer based sensor, showed a standard uncertainty of 1 mm at rest or at low speeds; the sensor still works at speeds of up to 30 m/s, although at higher uncertainty. The sensor is composed of only low cost components, thus being apt for first car equipment in many cases, and is able to self-adapt to different conditions in order to give the best results [5].

1.8 Organization of the Report

- Chapter 1 gives the basic introduction about the project.

- Chapter 2 explains about the general system architecture of the project.

- Chapter 3 outlines the system architecture and the concepts used in the project.
- Chapter 4 details the hardware and software implementations in the project.
- Chapter 5 presents the algorithms used in the project.
- Chapter 6 presents the results and analysis of the project.
- Chapter 7 gives the concluding remarks.

Chapter 2
System Architecture

2.1 General system description

The system is designed to aid people with visual impairments by providing them with a portable and affordable solution for object detection and face recognition. The system consists of a Raspberry Pi Model 3 microcontroller running Python code that employs advanced algorithms for face detection and recognition, and an HC-SR04 ultrasonic sensor for object detection. The system uses a Raspberry Pi camera module to capture images of faces, which are then processed to extract facial features and compared to a database of face templates to provide audio output of the person's name. The system also provides audio output of the distance of objects detected by the ultrasonic sensor using the eSpeak library. Overall, the system provides a comprehensive solution for people with visual impairments to navigate and interact with their surroundings more effectively.

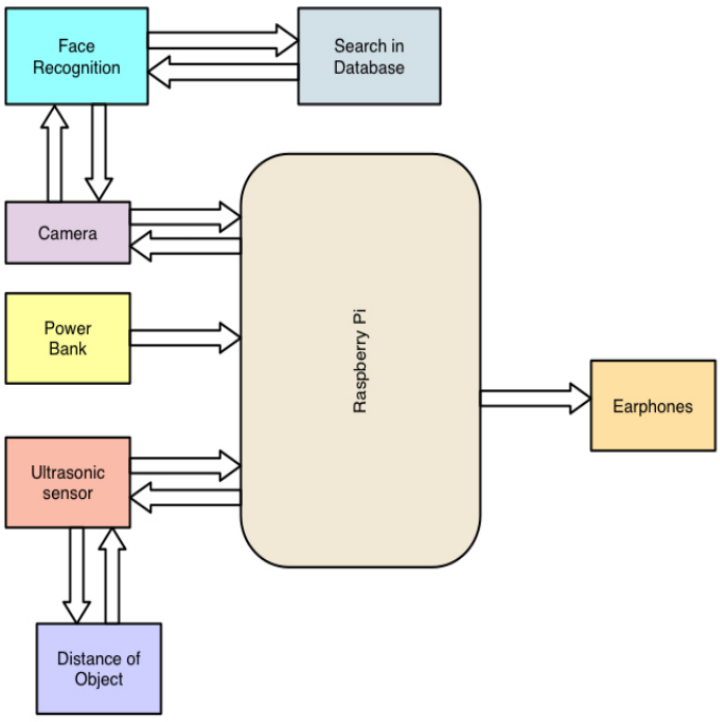

Figure 2.1: Block diagram for complete model

2.2 Block Diagram Description

2.2.1 Raspberry Pi 3:

Raspberry Pi 3 is a popular choice for building low-cost, embedded systems. It provides a powerful yet energy-efficient processor, high-resolution camera interface, and GPIO pins for interfacing with sensors such as ultrasonic sensors. With the help of OpenCV and other libraries such as NumPy and Dlib, Raspberry Pi 3 can be used to build real-time object detection and face recognition systems that can be integrated with other hardware components such as earphones and ultrasonic sensors. The Pi Camera module is also an ideal choice for capturing high-quality video and images for face recognition and object detection. Additionally, the compact size of the Raspberry Pi 3 makes it suitable for deployment in small spaces or on mobile robots [6].

2.2.2 Ultrasonic Sensor:

The ultrasonic sensor (HC-SR04) is used for object detection. The sensor measures the distance between the sensor and the object in front of it by emitting high-frequency sound waves and detecting the time it takes for the waves to bounce back to the sensor. This distance measurement is then used to detect whether an object is within a certain range. The ultrasonic sensor is a key component in this project as it allows the system to detect objects in the environment enabling the Raspberry Pi to respond appropriately [7].

2.2.3 Pi Camera

The Pi camera is used for face recognition and object detection by capturing images and video frames, which are then processed by algorithms to detect faces and objects. The camera's ability to capture high-quality images and video is crucial for accurate detection and recognition [8].

2.2.4 Face Recognition from existing Database:

Face recognition refers to the ability of the system to identify and recognize human faces in real-time using machine learning algorithms. The Raspberry Pi 3 can be used with libraries such as OpenCV and Dlib to perform facial recognition tasks. The Pi Camera can be used to capture images of faces, which are then processed by the face recognition algorithm to compare them to a database of known faces. The accuracy of the recognition system depends on factors such as lighting conditions, image quality, and the quality of the face recognition algorithm used. The output

of the face recognition system can be a simple text output indicating the name of the recognized person or a graphical user interface displaying the name and other relevant information [9].

2.2.5 Connections

The components are now connected after creating the code. After inserting the camera ribbon in camera module, solder the wires of ultrasonic sensor as illustrated below [10].

Table 2.1: Connection for GPIO

Raspberry Pi	Ultrasonic Sensor
5V pin	Vcc
Pin 6	GND
BCM Pin 27	TRIG
BCM Pin 22	ECHO

2.2.6 Testing the model

VNC (Virtual Network Computing) is used to remotely access and control your Raspberry Pi's desktop environment from another computer or device. This can be very helpful when testing and debugging your code for object detection and face recognition on the Raspberry Pi [11].

2.3 Theoretical background

2.3.1 Object Detection

Object detection is a field of computer vision that involves the detection of objects in an image or video stream. The goal is to identify and locate objects of interest within the image or video, and often to classify them as well. This is achieved by analyzing the pixel values in the image or video stream to identify regions that may contain objects, and then using machine learning algorithms to classify those regions as objects or non-objects. The most commonly used approach for object detection is the sliding window technique. This involves sliding a fixed-size window over the entire image or video stream, and classifying the contents of each window as either an object or not. However, this approach can be computationally expensive and time-consuming, especially for large images or videos. Another approach for object detection is using feature-based methods such as the Viola-Jones

algorithm, which uses Haar-like features and a cascade of classifiers to detect faces in an image. More advanced techniques involve using deep learning algorithms such as convolutional neural networks (CNNs), which have shown remarkable success in object detection tasks, especially with the advent of popular models such as YOLO (You Only Look Once) and Faster R-CNN (Region-based Convolutional Neural Network). Object detection has numerous practical applications, including self-driving cars, video surveillance, object tracking, and augmented reality [12].

2.3.2 Face Recognition

Face recognition involves various approaches, including geometric features-based, appearance-based, and hybrid methods. Geometric features-based methods use facial landmarks such as eyes, nose, and mouth to recognize the face, while appearance-based methods focus on the overall texture and appearance of the face. One of the most popular appearance-based methods is the Eigenface algorithm, which represents each face as a linear combination of eigenfaces, or principal components extracted from a set of training images. Another commonly used method is the Local Binary Pattern (LBP) algorithm, which extracts texture features from a face image using a local binary pattern operator. Recently, deep learning-based approaches, such as Convolutional Neural Networks (CNNs) and Deep Belief Networks (DBNs), have achieved state-of-the-art performance in face recognition tasks. These methods use multiple layers of neural networks to automatically learn features from raw face images, allowing for more accurate and robust recognition performance. Overall, face recognition methods have evolved significantly over the years, and continue to be an active area of research and development with various applications in security, surveillance, and biometric identification systems [13].

2.3.3 Object detection Evaluation Parameters

Distance: The Distance can be calculated by,

$$Distance = \frac{\text{Time X Speed Of Sound in Air(343m/s)}}{2} \tag{2.1}$$

Chapter 3
Software Description Requirements

The Software and the hardware requirements and its description is described below.

3.1 Software Description

3.1.1 Python

Python is a high-level programming language which includes simplicity and consistency, access to great libraries and frameworks for AI and machine learning (ML), flexibility, platform independence, and a wide community. Python is an open source interpreted, high level scripting language. It was used for Data Augmentation, creating a model, training and testing the model [14].

3.1.2 OpenCV

OpenCV is an open source computer vision library which is commonly used for image processing applications. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, clas- sify human actions in videos,track camera movements, track moving objects, ex- tract 3D models of objects, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc [15].

3.1.3 Face Recognition

Face recognition library is a collection of pre-trained models, tools, and algorithms used to detect and recognize faces from images or video feeds. It typically consists of several stages, including face detection, alignment, feature extraction, and face matching. The library uses deep learning techniques to learn the features of a face and recognize it accurately. Some popular face recognition libraries in Python include OpenCV, Dlib, and Face Recognition. These libraries can be used for a wide range of applications, including security systems, access control, and personalized advertising[13].

3.1.4 Dlib

Dlib is a C++ library with Python bindings that provides tools for developing complex software in C++. It includes tools for creating complex software such as machine learning models, data analysis tools, and deep learning algorithms. The Python bindings of Dlib make it easier to use this powerful library in Python. With Dlib in Python, you can perform a wide range of tasks including facial detection, facial recognition, pose estimation, and more. The library is widely used in the field of computer vision and is known for its high accuracy and performance [16].

3.1.5 NumPy

NumPy is a popular Python library for numerical computing that provides an efficient way to work with large, multi-dimensional arrays and matrices. It includes a wide range of mathematical functions for performing operations on these arrays, such as linear algebra, Fourier transforms, and random number generation. NumPy is widely used in scientific computing, data analysis, machine learning, and other fields that require fast and efficient numerical operations. Its syntax and functionality are similar to those of MATLAB, making it a popular choice for those who are familiar with MATLAB programming [17].

3.1.6 eSpeak

eSpeak is a compact open-source software speech synthesizer for Linux, Windows, and other platforms. It can speak text files, output spoken audio data to WAV files, and can also serve as a talking program for blind people or for those with a reading disability. It supports multiple languages, and its voice can be customized by adjusting pitch, speed, and volume. eSpeak is often used in robotics, home automation, and assistive technology projects. In Python, eSpeak can be used as a library to add speech synthesis capabilities to Python applications [18].

3.1.7 VNC

VNC (short for Virtual Network Computing) is a remote desktop software that allows users to access and control a remote computer over a network connection. It works by transmitting the keyboard and mouse input from the local computer to the remote computer, and sending the screen updates back to the local computer. This allows users to remotely access and control another computer as if they were sitting in front of it. VNC can be used on a variety of operating systems and hardware platforms, and supports various authentication and encryption methods to

ensure secure connections. It is commonly used for remote technical support, remote access to office computers, and remote access to servers and other computing resources[11].

3.2 Hardware Description

3.2.1 Raspberry Pi 3

Raspberry Pi 3 is a credit card-sized single-board computer that was released in 2016 by the Raspberry Pi Foundation. It is the third generation of Raspberry Pi and features a 1.2 GHz quad-core ARM Cortex-A53 processor, 1GB of RAM, and integrated Wi-Fi and Bluetooth capabilities. The board also has a full-size HDMI port, four USB 2.0 ports, an Ethernet port, a 3.5mm audio jack, a camera interface, and a microSD card slot for storage. It runs on a variety of operating systems, including Raspbian, a Linux-based operating system specifically designed for Raspberry Pi. The Raspberry Pi 3 is commonly used for a wide range of projects, including home automation, robotics, and education[6].

3.2.2 Pi Camera

The Pi Camera is a camera module specifically designed for the Raspberry Pi, developed by the Raspberry Pi Foundation. It is a small, lightweight camera that connects directly to the Raspberry Pi board through the camera serial interface (CSI) port, which is a high-speed serial interface that enables fast data transfer between the camera and the Pi. The Pi Camera comes in two versions, the regular Pi Camera and the Pi Camera Module V2, both of which offer high-quality image and video capture capabilities. The Pi Camera is widely used in a variety of projects, such as robotics, security systems, and surveillance cameras, due to its compact size, low cost, and ease of use[8].

3.2.3 Ultrasonic sensor(HC-SR04)

The HC-SR04 is a popular ultrasonic sensor module that is widely used in various electronics projects, including distance measurement applications. It consists of a transmitter and a receiver, which work together to emit and receive ultrasonic waves. When the transmitter sends out an ultrasonic wave, it bounces off an object and returns to the receiver. The time taken for the wave to travel back and forth is measured and used to calculate the distance to the object. The HC-SR04 sensor is simple to use and provides accurate distance measurements, making it an ideal

component for obstacle avoidance and navigation in robots and other electronics projects[7].

3.2.4 Battery/Power bank

A 2000mAh 5V battery is a type of rechargeable lithium-ion battery commonly used to power small electronics such as smartphones, tablets, and IoT devices. The 2000mAh rating indicates its capacity to store electrical energy, with higher numbers indicating a longer runtime before needing to be recharged. The 5V rating means it is compatible with devices that require a 5V power supply. This type of battery can be recharged via a USB cable or charging dock, and is often used in portable electronic projects that require a compact and reliable power source.

3.2.5 Earphones

Earphones are used as an output for the audio generated by the Raspberry Pi. The Raspberry Pi has a 3.5mm audio jack, which can be used to connect earphones or speakers. Alternatively, you can use USB speakers or Bluetooth speakers if they are compatible with the Raspberry Pi.

3.2.6 Headset

The hardware parts in this project including Raspberry Pi 3, Ultrasonic Sensor(HC-SR04), Pi camera and the power bank is attached in this headset which will worn by the user.

Chapter 4
Implementation

The implementation of the face recognition and object detection system using Raspberry Pi involves connecting the HC-SR04 ultrasonic sensor and Pi camera module, installing and importing the necessary libraries, and writing Python code for the algorithms. The code can be executed on the Raspberry Pi to detect faces and objects in real-time, and provide an audio output of the distance and identity. Enabling VNC allows for remote access and control of the system from another computer or mobile device. The project requires some technical skills and knowledge of programming and electronics, but can be a fun and rewarding project for enthusiasts and learners.

4.1 Object Detection

The steps for performing Object Detection are as follows:

1. Connect the HC-SR04 ultrasonic sensor to the Raspberry Pi and import the necessary libraries.
2. Write Python code to send a signal to the ultrasonic sensor and measure the time it takes for the echo to return.
3. Calculate the distance to the object using the formula: Distance = (Time taken by sound wave to travel to the obstacle and back x Speed of Sound in Air).
4. Set a threshold distance value for the detection range and check if the measured distance is within the range.
5. Draw a bounding box around the detected object and provide an audio output of the distance to the object using the eSpeak library.
6. Repeat the process for continuous object detection.

4.2 Face Recognition

The steps for performing Face Recognition are as follows:

1. Connect the Raspberry Pi camera module to the Raspberry Pi and import the necessary libraries.

2. Capture an image of the person's face using the Pi camera and save it.
3. Use a facial detection algorithm such as Haar cascades or Dlib to detect the face in the image.
4. Once a face is detected, use a facial recognition algorithm such as Eigenfaces or Fisherfaces to extract facial features from the image and compare them to a database of known faces.
5. If a match is found, identify the person's name and provide an audio output of their name using the eSpeak library.
6. Repeat the process for continuous face recognition.

Chapter 5
Algorithms

The Haar cascades algorithm for face detection and Eigenfaces or Fisherfaces for facial recognition are been used. However, there are many other algorithms available for face detection and recognition, such as Dlib, LBPH, and OpenFace. The choice of algorithm may depend on factors such as accuracy, speed, and resource constraints.

5.1 Introduction to Face Recognition

Face recognition is a biometric technology that has gained significant interest due to its wide range of applications, such as security systems, access control, and identification. The technology involves the analysis of facial features of an individual and comparing it with a database of known faces to identify the person. This identification process is based on the uniqueness of facial features, such as the distance between the eyes, the shape of the nose, and the size of the mouth. Various algorithms have been developed to recognize faces, including the popular Eigenface and Haar cascade algorithms. The success of face recognition technology depends on factors such as image quality, lighting conditions, and the algorithm used.

5.2 Haar Cascades Algorithm

Haar cascades algorithm is a popular technique used for object detection in computer vision. It is based on Haar wavelets and involves training a classifier to identify specific features of the object. The algorithm is capable of detecting objects of various sizes, orientations, and positions in an image or video. It works by sliding a window over the image and using a set of pre-trained classifiers to determine whether the object is present in the window or not. The classifiers are arranged in a cascading manner, with each stage progressively eliminating false positives. The Haar cascades algorithm has been successfully applied to a variety of object detection tasks, including face detection, pedestrian detection, and license plate recognition [19].

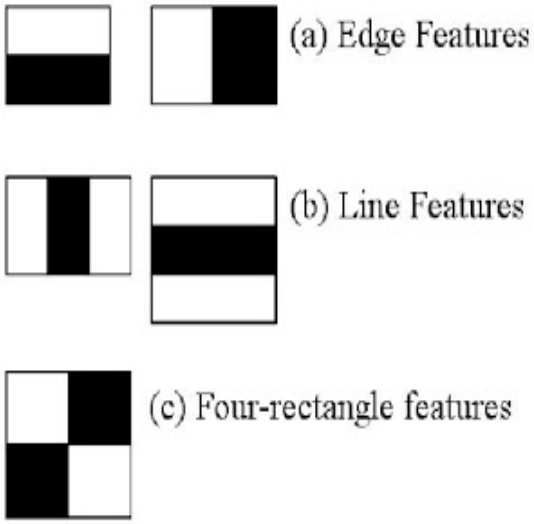

Figure 5.1: Haar-feature selection

5.2.1 Importance of Haar Cascades Algorithm

- Haar cascades are widely used in computer vision and image processing for object detection tasks, including face detection.
- They are based on the Haar wavelet, which is a mathematical function that can be used to detect patterns in images.
- The Haar cascade algorithm involves training a classifier on positive and negative examples of an object, which can then be used to detect the object in new images.
- Haar cascades are computationally efficient and can be used in real-time applications, making them suitable for tasks like face detection in video streams.
- However, they have some limitations, including a tendency to produce false positives and difficulties in detecting objects that vary in scale or orientation.

5.2.2 Working of Haar Cascades Algorithm

The Haar cascade algorithm is a type of object detection method that uses a set of features known as Haar features to detect objects within an image. The algorithm works by first training a classifier using a set of positive and negative images. The positive images are images that contain the object that we want to detect, while the negative images are images that do not contain the object. During the training process, the algorithm learns to distinguish between the positive and negative images by extracting Haar features from each image and using them to build a

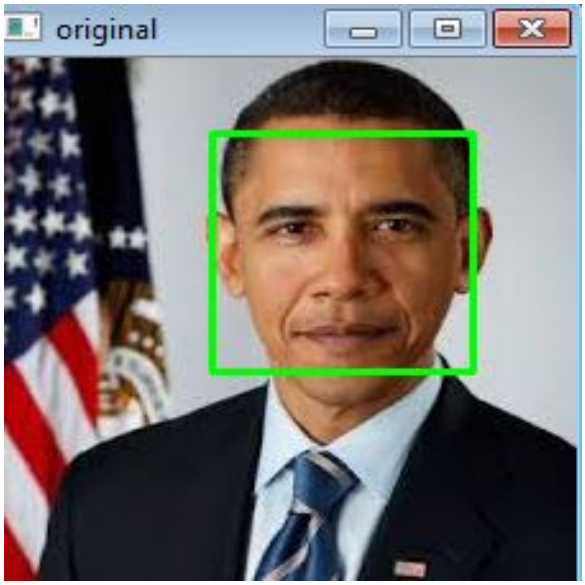

Figure 5.2: Face Detection

classifier. Once the classifier is trained, it can be used to detect the object in new images. To do this, the image is scanned at various scales and positions, and the Haar features are extracted at each location. The classifier then evaluates the features and determines whether the object is present or not. If the object is present, the algorithm returns the location and size of the object within the image. Overall, the Haar cascade algorithm is a powerful tool for object detection, particularly in situations where the object is well-defined and has distinctive features.

5.3 Eigenfaces Alogrithm

Eigenfaces is a popular algorithm for face recognition that utilizes principal component analysis (PCA) to extract the most important features of a face image. The algorithm works by creating a set of eigenfaces, which are the principal components of the image space, using a training dataset. Each eigenface is a vector that represents a specific feature of the face. When a new face image is presented to the algorithm, it is projected onto the eigenfaces to extract the relevant features, and then compared to the eigenfaces of the training dataset to find the closest match. Eigenfaces are widely used in face recognition applications due to their efficiency, accuracy, and ability to work with low-resolution images. However, they are sensitive to changes in lighting and facial expressions, and may require a large training dataset to achieve high accuracy [20].

Figure 5.3: Training Images

Chapter 6
Testing and Results

The project is divided into two parts:

- To use the Ultrasonic sensor to measure the distance of the object in front of the user and giving the feedback in the for audio output through the earphones
- To use the Pi camera to recognize the image captured in the frame of the video and search through the array of known faces and return the audio feedback through the earphones.

The Testing results are obtained and are as show below.

6.1 Testing

6.1.1 Object Detection Testing

The testing part for object detection involves running the system and evaluating its performance in detecting objects. The system should be tested using various scenarios and environments to ensure its robustness and accuracy. The following steps can be followed for testing the object detection system:

1. Set up the testing environment with appropriate lighting, background, and objects to be detected.
2. Run the object detection system and observe its performance in detecting the objects.
3. Record the detection accuracy, speed, and any false positives or false negatives.
4. Make adjustments to the system parameters, such as detection threshold or object size, to improve its performance.
5. Repeat the testing process for different scenarios and objects to ensure the system's robustness.
6. Evaluate the system's overall performance and determine if further improvements or optimizations are needed.

6.1.2 Face Recognition Testing

The testing phase for face recognition involves the following steps:

1. Acquiring face images: In this step, images of faces are acquired from the test dataset or from a live video feed.

2. Face detection: The acquired images are then passed through a face detection algorithm, such as Haar cascades, to locate the position and size of the face in the image.

3. Face alignment and normalization: The detected face is then aligned and normalized to a standard size and orientation to reduce variations caused by differences in pose, expression, and lighting.

4. Feature extraction: In this step, features are extracted from the normalized face using algorithms such as Eigenfaces, Fisherfaces, or Local Binary Patterns (LBP).

5. Matching: The extracted features are then compared with the features of the faces in the training dataset using algorithms such as k-Nearest Neighbors (k-NN) or Support Vector Machines (SVM) to determine the identity of the face.

6. Evaluation: The performance of the face recognition system is evaluated using metrics such as accuracy, precision, recall, and F1 score. The system may be retrained and fine-tuned based on the evaluation results to improve its performance.

6.2 Results

Results of Object detection and Image Recognition on underwater images obtained during the Testing are given below.

6.2.1 Object Detection Results

Figure 6.1 shows the results for Object detection which is a crucial task in computer vision, which involves identifying and locating objects within an image or video. In this particular project, the team utilized Python code to perform object detection on a Raspberry Pi 3, which is a small single-board computer that can run various operating systems, including Raspbian OS. To access the Raspberry Pi's graphical

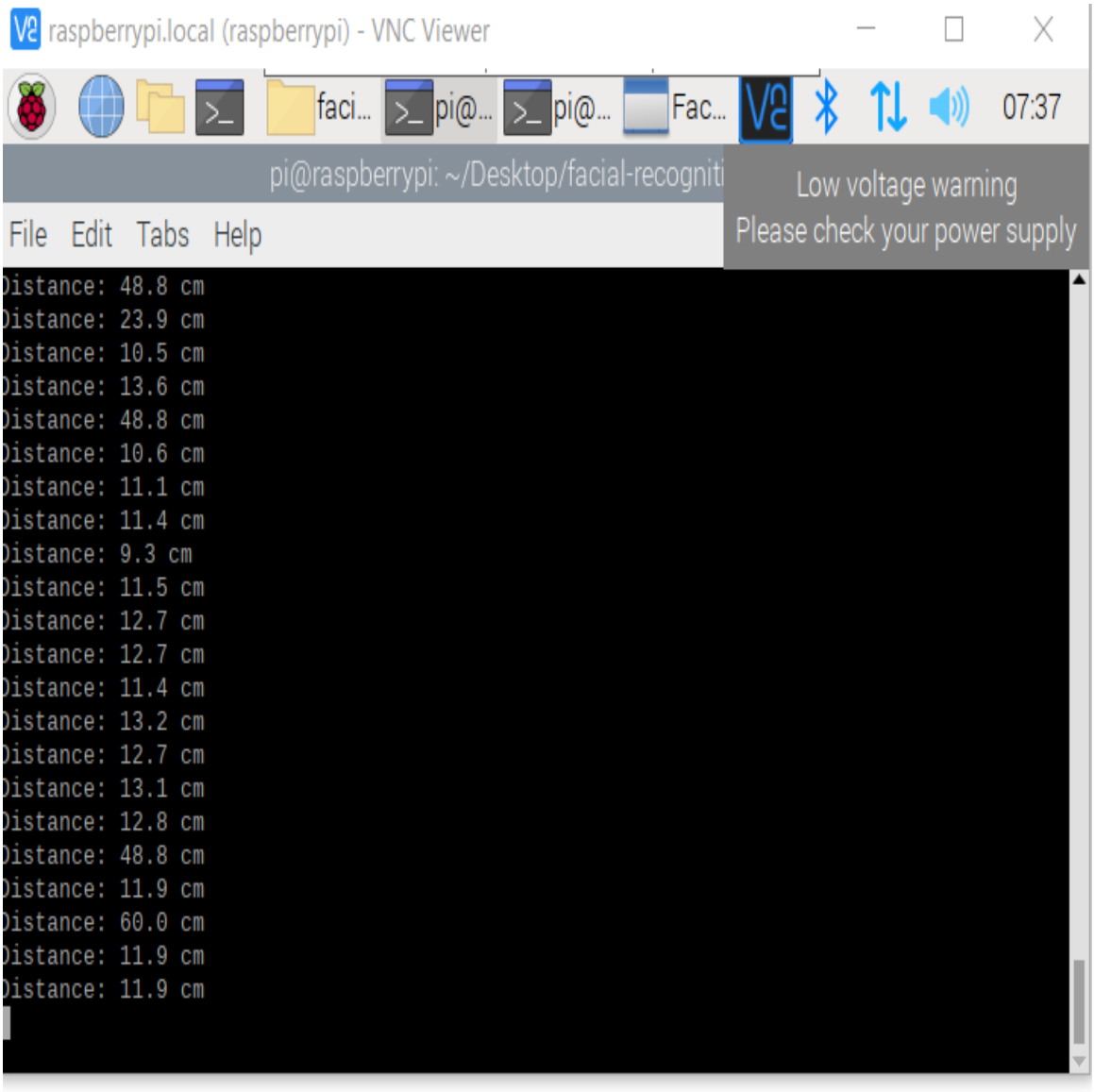

Figure 6.1: Results for the Object Detection code on VNC

user interface, the team used VNC (Virtual Network Computing) to remotely connect to the desktop. This allowed them to interact with the Raspberry Pi as if they were using it directly, using their own keyboard, mouse, and display.

Once the object detection was performed, the team calculated the output distance between the detected object and the Raspberry Pi's camera. This distance was then synthesized using the eSpeak library, which is a software speech synthesizer for Linux and other platforms. By using this library, the team was able to generate an audio output that provided information about the distance between the object and the camera.

To hear the audio output, the team used earphones connected to the Raspberry

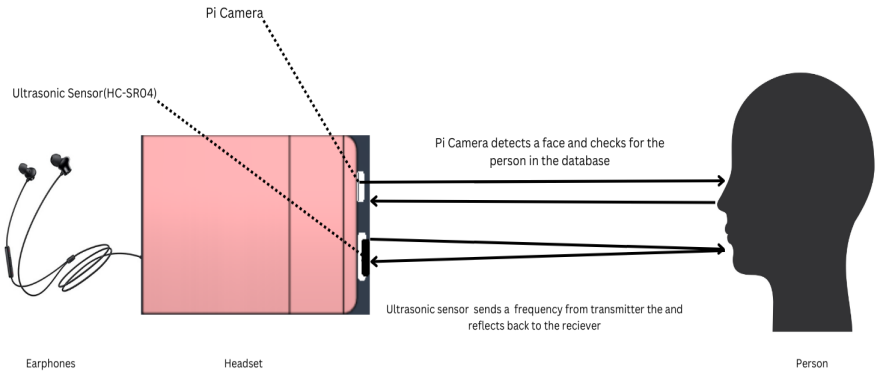

Figure 6.2: Working Diagram of Object Detection and Face Recognition

Pi's audio output jack. This allowed them to listen to the synthesized speech without disturbing others in the room. By combining object detection, distance calculation, and audio output, the team created a system that could provide useful information about objects in the Raspberry Pi's field of view, even without a visual display. This type of system could have a wide range of applications in fields such as robotics, surveillance, and assistive technology.

6.2.2 Face Recognition Results

In this particular project, the team implemented a face recognition system using a Raspberry Pi and a Pi Camera module. When a face is detected within the camera's field of view, the system first captures an image of the face. This image is then processed and analyzed using a face recognition algorithm.

The face recognition algorithm compares the features of the detected face with the features of the faces in a pre-existing dataset. This dataset contains images and corresponding labels for known faces that the algorithm has been trained on. If a match is found between the detected face and a face in the dataset, the system communicates the name associated with that face through the audio output. For example in Figure 6.3, if the detected face matches with the face labeled "Sam" in the dataset, the system will communicate the name "Sam" through the connected earphones.

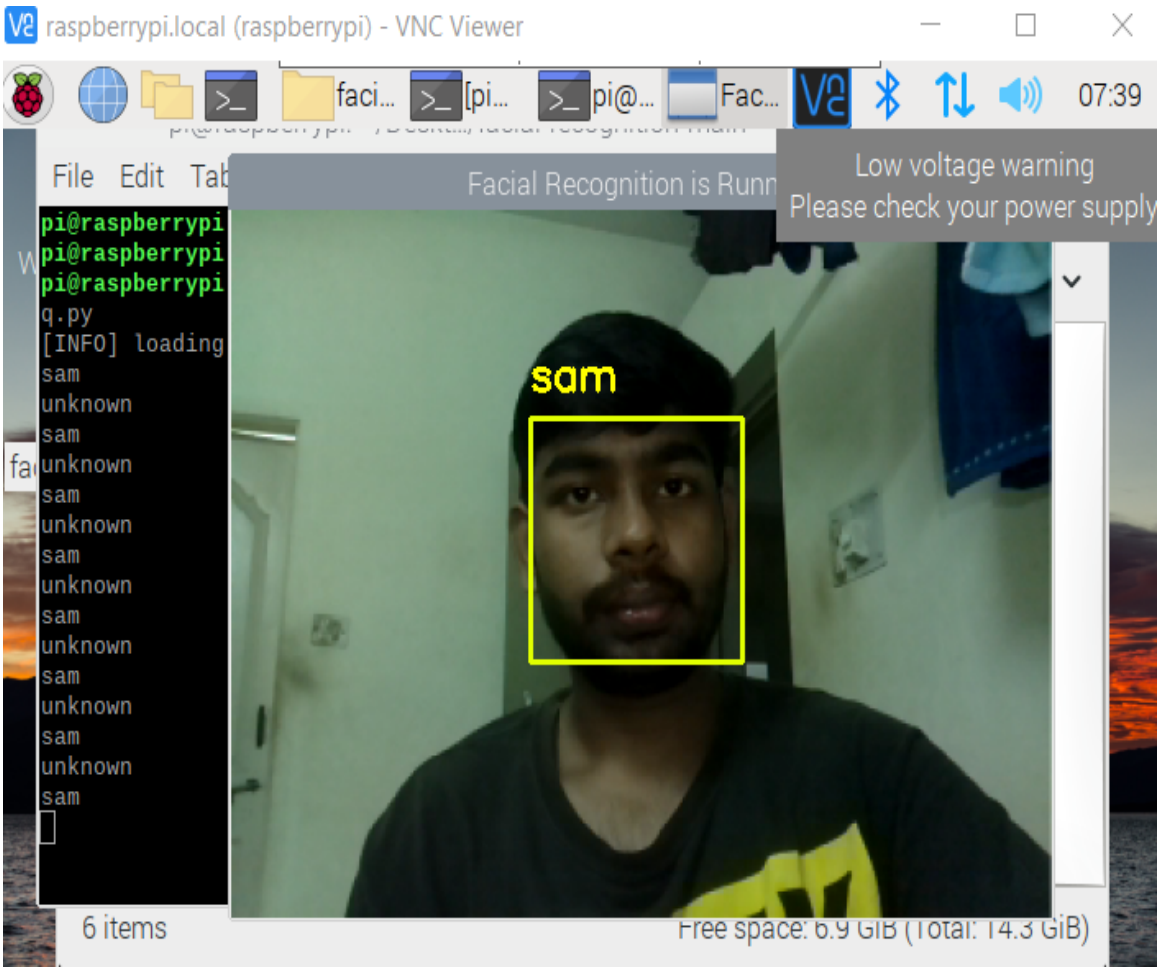

Figure 6.3: Known Face Recognized

However in Figure 6.4, if the detected face does not match with any of the faces in the dataset, the system will communicate a message called "Unknown" through the audio output via earphones. This is useful in scenarios where the system needs to identify unknown individuals. The audio output allows the user to receive information about the detected face without requiring a visual display.

The system's ability to recognize known faces and identify unknown faces has many potential applications in security, access control, and personalized user experiences. The system's flexibility and scalability also allow it to be trained on a wide variety of faces and easily integrated into different projects and environments.

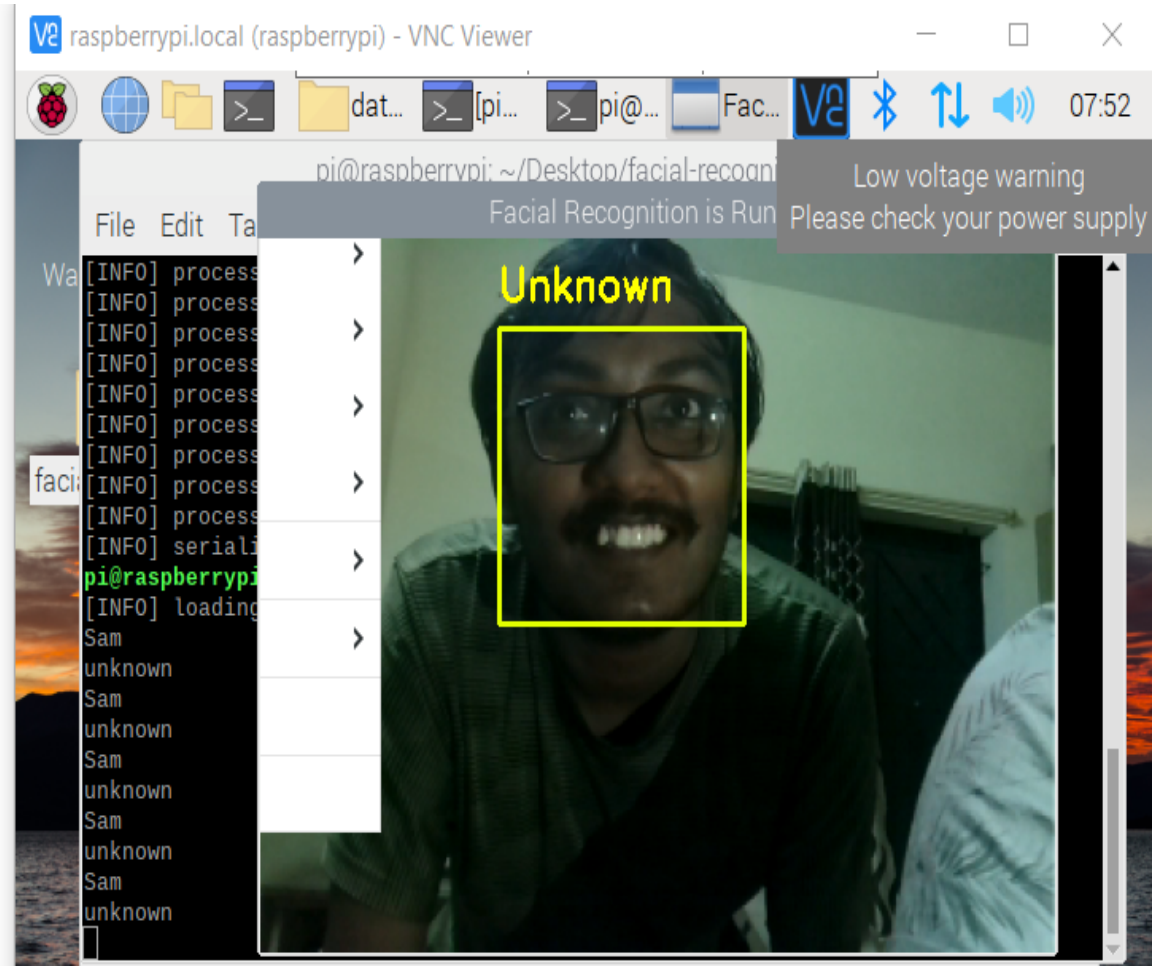

Figure 6.4: Unknown Face Recognized

Chapter 7
Conclusions and Future Work

7.1 Conclusion

1. The Raspberry Pi Model 3 was used as the main micro-controller to run Python code for face recognition and object detection.

2. The HC-SR04 ultrasonic sensor was used as a distance measuring sensor to detect the distance of an object.

3. The Raspberry Pi camera module was used for face recognition with OpenCV and a facial recognition algorithm

4. The system successfully enabled the measurement of object distance and face recognition through audio feedback. The system was able to capture an image of a person's face, detect the presence of a face, and extract facial features to identify the person's identity.

5. Using This smart glasses it can helps the blind and visually impaired people to commute freely by experiencing their surroundings.

7.2 Future Work

1. For better obstacle detection and faster response, an infrared sensor can be added to the existing ultrasonic sensor setup. This combination of sensors can provide more accurate and reliable distance measurements in various environmental conditions.

2. The performance and latency of the system can be improved by using a newer model of Raspberry Pi. Upgrading to a newer model can provide better processing power and memory, which can lead to faster and more efficient object detection and face recognition.

3. Multi-threading can be added to the code, allowing both the object detection and face recognition tasks to be executed in a single Python file, eliminating the need to run two separate files.

4. Integration of additional sensors: In addition to the ultrasonic sensor and infrared sensor, other sensors such as GPS, temperature sensors, and humidity

sensors can be integrated to provide more context and information about the environment.

5. Development of a mobile application: A mobile application can be developed to allow visually impaired individuals to control and customize the system settings and receive alerts and notifications on their smartphones.

Bibliography

[1] M. R. Miah and M. S. Hussain, "A unique smart eye glass for visually impaired people," in *2018 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, 2018, pp. 1–4.

[2] C. Alvarez Casado, M. Bordallo Lopez, J. Holappa, and M. Pietikäinen, "Face detection and recognition for smart glasses." United States: IEEE Institute of Electrical and Electronic Engineers, 2015, pp. 37–38, 2015 IEEE International Symposium on Consumer Electronics, ISCE 2015 ; Conference date: 24-06-2015 Through 26-06-2015.

[3] J. Bai, S. Lian, Z. Liu, K. Wang, and D. Liu, "Smart guiding glasses for visually impaired people in indoor environment," vol. 63, no. 3. IEEE Press, aug 2017, p. 258–266. [Online]. Available: https://doi.org/10.1109/TCE.2017.014980

[4] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Computer Vision - ECCV 2004*, T. Pajdla and J. Matas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 469–481.

[5] A. Carullo and M. Parvis, "An ultrasonic sensor for distance measurement in automotive applications," *Sensors Journal, IEEE*, vol. 1, pp. 143 – 147, 09 2001.

[6] "Raspberry pi 3 model b," https://www.raspberrypi.com/products/raspberry-pi-3-model-b/.

[7] "Ultrasonic sensor hc-sr04," https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/.

[8] "Pi camera," https://projects.raspberrypi.org/en/projects/getting-started-with-picamera.

[9] "Connections between raspberry pi and ultrasonic sensor," https://tutorials-raspberrypi.com/raspberry-pi-ultrasonic-sensor-hc-sr04/.

[10] "Face recognition," https://pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/.

[11] "Vnc(virtual network computing," https://www.realvnc.com/en/.

[12] "Object detection," https://www.instructables.com/Object-Detection-on-Raspberry-Pi/.

[13] "Face recognition," https://pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/.

[14] "Python," https://www.python.org/doc/.

[15] "Opencv," https://opencv.org/.

[16] "Dlib," https://pypi.org/project/dlib/.

[17] "Numpy," https://numpy.org/doc/stable/index.html.

[18] "espeak," https://pypi.org/project/python-espeak/.

[19] "Haar cascades algorithm," https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html.

[20] "Eigenfaces algorithm," https://pyimagesearch.com/2021/05/10/opencv-eigenfaces-for-face-recognition/.

Appendix A
Python code for Object Detection

```
   import time
import espeak as espeak
import os
def read_distance():
    import RPi.GPIO as GPIO
    GPIO.setmode(GPIO.BCM)
    TRIG = 27
    ECHO = 22
    GPIO.setup(TRIG, GPIO.OUT)
    GPIO.setup(ECHO, GPIO.IN)
    GPIO.output(TRIG, GPIO.LOW)
    GPIO.output(TRIG, GPIO.HIGH)
    time.sleep(0.00001)
    GPIO.output(TRIG, GPIO.LOW)
    pulse_start = time.time()
    while GPIO.input(ECHO) != GPIO.HIGH:
        pulse_start = time.time()
    pulse_end = pulse_start
    while time.time() < pulse_start + 0.1:
        if GPIO.input(ECHO) == GPIO.LOW:
            pulse_end = time.time()
            break
    GPIO.cleanup()
    pulse_duration = pulse_end - pulse_start
    distance = 34300 * pulse_duration / 2
    if distance <= 400:
        return distance
    else:
        return None

if __name__ == '__main__':
    print("Starting distance measurement! Press Ctrl+C to stop .")
    time.sleep(1)
    while True:
```

```
loop_start_time = time.time()
distance = read_distance()
if distance:
    print("Distance: %.1f cm" % (distance))
    if distance <= 18:
        os.system('espeak-ng  "Obstacle is nearby"')
time_to_wait = loop_start_time + 1 - time.time()
if time_to_wait > 0:
    time.sleep(time_to_wait)
```

Appendix B
Python code for Face Recognition

```
from imutils.video import VideoStream
from imutils.video import FPS
import face_recognition
import imutils
import pickle
import time
import cv2
import os

currentname = "unknown"
encodingsP = "encodings.pickle"

print("[INFO] loading encodings + face detector...")
data = pickle.loads(open(encodingsP, "rb").read())

vs = VideoStream(usePiCamera=True).start()
time.sleep(2.0)

fps = FPS().start()

while True:
frame = vs.read()
frame = imutils.resize(frame, width=500)
boxes = face_recognition.face_locations(frame)
encodings = face_recognition.face_encodings(frame, boxes)
names = []

for encoding in encodings:
matches = face_recognition.compare_faces(data["encodings"], encoding)
name = "Unknown"

if True in matches:
matchedIdxs = [i for (i, b) in enumerate(matches) if b]
counts = {}
```

```
for i in matchedIdxs:
name = data["names"][i]
counts[name] = counts.get(name, 0) + 1

name = max(counts, key=counts.get)

if currentname != name:
currentname = name
print(currentname)
os.system("espeak-ng '{}'".format(currentname))
else:
currentname = "unknown"
print(currentname)
os.system("espeak-ng '{}'".format(currentname))

names.append(name)

for ((top, right, bottom, left), name) in zip(boxes, names):
cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 225), 2)
y = top - 15 if top - 15 > 15 else top + 15
cv2.putText(frame, name, (left, y), cv2.FONT_HERSHEY_SIMPLEX, .8,
         (0, 255, 255), 2)

cv2.imshow("Facial Recognition is Running", frame)
key = cv2.waitKey(1) & 0xFF

if key == ord("q"):
break

fps.update()

fps.stop()
print("[INFO] elasped time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

cv2.destroyAllWindows()
vs.stop()
```