



**UNIVERSITY OF ILORIN, ILORIN, KWARA STATE**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**DEVELOPMENT OF AN ASSISTIVE SMART DEVICE:**

**REAL-TIME OBJECT DETECTION, OBSTACLE**

**AVOIDANCE AND NAVIGATION ASSISTANCE**

**BY**

**IDAJILI JOHN OJOCHEGBE**

**(19/30GR028)**

**July, 2025**

**DEVELOPMENT OF AN ASSISTIVE SMART DEVICE:**

**REAL-TIME OBJECT DETECTION, OBSTACLE  
AVOIDANCE AND NAVIGATION ASSISTANCE**

**IDAJILI, JOHN OJOCHEGBE**

**19/30GR028**

**A REPORT SUBMITTED TO THE DEPARTMENT OF  
COMPUTER ENGINEERING, UNIVERSITY OF  
ILORIN, NIGERIA, IN PARTIAL FULFILLMENT OF THE  
REQUIREMENT FOR THE AWARD OF B.ENGR. (HONS) IN  
COMPUTER ENGINEERING**

**SUPERVISED BY:**

**ENGR. H. O. MAHMUD**

**JULY, 2025**

## **DECLARATION**

I hereby attest that, under the supervision of Engr. H. O. Mahmud, I performed all the works detailed in this report, submitted to the Department of Computer Engineering, Faculty of Engineering and Technology, University of Ilorin, Ilorin, Nigeria. All of my sources of research are properly cited and acknowledged.

.....

**IDAJILI, JOHN OJOCHEGBE**

**19/30GR028**

**Signature & Date**

## CERTIFICATION

The undersigned hereby attest that the project report titled "Development of an assistive Smart Device: Object detection, Obstacle Avoidance and Navigation Assistance" implemented by IDAJILI, John Ojochegbe with matriculation number "19/30GR028" has been read, accepted, and deemed to satisfy the requirements of the Department of Computer Engineering, University of Ilorin, Nigeria for the award of a Bachelor of Engineering (B.Eng.) Degree in Computer Engineering.

.....

**Engr. H. O. Mahmud**  
**Project Supervisor**

.....

**Date**

.....

**Prof. J.F. Opadiji**  
**Head of Department**

.....

**Date**

.....

**External Supervisor**

.....

**Date**

## **DEDICATION**

I dedicate this project and the success of my undergraduate program to the Almighty God, my ever and present helper, and unlimited provider in times of need. The one who saw me through from the very beginning and continually watches over me. To whom I give all the praises. To my lovely parents, Mr. Stephen Idajili and Mrs. Blessing Idajili, who have been my unwavering support both financially and mentally. To my siblings Mrs. Joy Idachaba, Miss Gloria Stephen, Master Samuel Idajili, Miss Naomi Aaron, Master Emmanuel Idajili and our last born, Miss Salome Idajili. Lastly, I dedicate this project to all visually impaired individuals, trying to cope and survive through life hurdles.

## **ACKNOWLEDGEMENT**

I want to thank my parents, Mr. Idajili Stephen and Mrs. Aladi Blessing Idajili, my brothers Samuel Idajili and Emmanuel Idajili, and my sisters Mrs. Joy Idachaba, Miss Gloria Stephen, Mrs. Naomi Aaron and Miss Salome Idajili for their unmistakable and unquantifiable love, support and assistance during my undergraduate Programme over the years. These individuals helped me identify my desired path and inspired me to remain focused with steady encouragement and financial support, aiding my sustenance. May God continue to provide for you, ease your affairs, and reward you in abundance.

I want to express my deepest gratitude to my supervisor, Engr. H.O. Mahmud, for his guidance, subtle messages of wisdom and encouragement. His support has enabled me to handle all my work efficiently and effectively with utmost diligence and perseverance. It was and will remain a privilege to work with a man of his caliber. To the Computer Engineering Academia, my highly esteemed HOD Prof. Engr. J.F. Opadiji, and my various academic parents; Prof. Engr. A.T. Ajiboye, Dr. Mrs. Olawole, Engr A.R. Ajayi, Engr. Adeshina who happened to be my Level Adviser and Engr. O.F Adebayo and to all staff who have impacted my life during my course.

May God continue to shower his blessings on every one of you. Amen.

# ABSTRACT

Visually impaired individuals face daily challenges in navigating unfamiliar environments due to limited spatial awareness and the lack of affordable, real-time assistive technologies. Traditional solutions often depend on internet connectivity or are too expensive for widespread use in developing regions.

This project aims to develop an affordable, offline, AI-powered wearable system that enhances navigation and object awareness for visually impaired users. The objectives include implementing real-time object detection, short-range obstacle avoidance, and offline GPS-based voice navigation—all integrated into a lightweight, wearable prototype.

The system was developed using a Raspberry Pi 4 as the core processing unit. A custom-trained YOLOv5-nano model was deployed for real-time object detection, with output filtered by confidence score and announced via text-to-speech. Ultrasonic sensors were integrated to provide adaptive feedback for close-range obstacles using speech and variable beeping patterns. Offline navigation was enabled using the NEO-6M GPS module and localized map data extracted from OpenStreetMap, specifically covering the University of Ilorin campus.

Implementation involved configuring the Raspberry Pi environment, integrating the camera, sensors, and GPS module, and developing Python scripts for

synchronized data processing and audio output. The entire system was assembled into a wearable form and tested in real-world settings.

Results showed that the system successfully identified objects with reasonable accuracy, provided responsive obstacle alerts, and generated real-time voice-guided navigation without internet access. These outcomes validate the project's objectives and demonstrate the system's potential as a low-cost assistive tool for visually impaired users, especially in low-resource environments.



## Table of Contents

DECLARATION .....	iii
CERTIFICATION.....	iv
DEDICATION .....	v
ACKNOWLEDGEMENT .....	vi
ABSTRACT.....	vii
TABLE OF CONTENTS .....	<b>Error! Bookmark not defined.</b>
LIST OF FIGURES .....	xi
LIST OF TABLES .....	xii
CHAPTER ONE .....	1
INTRODUCTION .....	1
Project Background .....	1
1.2 Problem Statement .....	2
Aim & Objectives.....	3
Project Significance.....	4
Project Scope .....	5
Project Organization.....	7
CHAPTER TWO .....	9
LITERATURE REVIEW .....	9
2.1. Introduction .....	9

2.2. Components Review.....	9
2.2.1 Raspberry Pi 4 Model B .....	9
2.2.2 Camera Module .....	11
2.2.3 Ultrasonic Sensor (HC-SR04) .....	12
2.2.4 GPS module.....	13
2.2.5. Audio output (speaker) .....	14
2.2.6. Microphone Module .....	15
2.2.7 SD Card .....	17
2.3 Power supply (Battery selection) .....	18
2.4 Related Works. ....	19
2.5. Appraisal of the related works.....	23
CHAPTER THREE .....	27
METHODOLOGY .....	27
3.1 System Overview .....	27
3.2 Component Review .....	30
3.2.1 Raspberry Pi 4 .....	30
3.2.2 Pi Camera Module.....	33
3.2.3 Ultrasonic Sensors (HC-SR04) .....	36
3.2.4 GPS Module (GY-GPS6MV2).....	39
3.2.5 Power Supply .....	41
3.2.6 Power Calculation .....	42

3.2.7 Power Supply Design and Calculation .....	43
3.2.8 Circuit Diagram .....	45
3.2.8.1 Camera Module Connection.....	46
3.2.8.2 Ultrasonic Sensor Interface (HC-SR04).....	46
3.2.8.3 GPS Module Connection (GY-GPS6MV2) .....	46
3.3 Software Development .....	47
3.3.1 AI Model Development and Training Process .....	50
3.3.2 Component-Level Software Implementation .....	51
3.3.2.1 Ultrasonic Sensor .....	51
3.3.2.2 Camera and Object Detection.....	51
3.3.2.3 GPS Module and Navigation.....	52
CHAPTER FOUR.....	53
IMPLEMENTATION, TESTING AND RESULTS .....	53
4.1 Overview .....	53
4.2 Raspberry Pi Setup .....	53
4.3 Model Training and Optimization.....	54
4.4 Object Detection Model Testing (PC Simulation) .....	55
4.5 Object Detection Deployment on Raspberry Pi. ....	57
4.6 Proximity-Based Audio Feedback Using Ultrasonic Sensor .	59
4.7 Navigation Assistance .....	61
4.8 Full System Integration and Field Testing .....	63

4.9 Result and Summary .....	66
CHAPTER FIVE .....	68
CONCLUSION AND RECOMMENDATIONS .....	68
5.1 Conclusion.....	68
5.2 Limitations.....	69
5.3 Recommendations .....	70
REFERNCES .....	73
APPENDICES .....	75
APPENDIX A: Bill of Engineering and Materials Evaluation (BEME) .....	75
APPENDIX B: Raspberry Pi Codes .....	76

# LIST OF FIGURES

Figure 1.1: Diagram of the Proposed Methodology .....	7
Figure 2.1: Raspberry Pi 4 Model B Specifications .....	10
Figure 2.2: Raspberry Pi Camera Module .....	12
Figure 2.3: Typical Ultrasonic Sensor Module HC-SR04 .....	13
Figure 2.4: NEO-6M GPS Module for Location Tracking .....	14
Figure 2.5: Compact Audio Speaker Module .....	15
Figure 2.6: Typical Electret Microphone Module .....	16
Figure 2.7: Lithium-Polymer Battery Pack for Power Supply .....	17
Figure 2.8: Micro-SD Card for Data Storage .....	18
Figure 3.1: Block Diagram of the AI-Powered Smart Glasses System .....	27
Figure 3.2: Raspberry Pi 4 Tech Specs .....	31
Figure 3.3: Mechanical Dimension of the Raspberry Pi 4B .....	32
Figure 3.4: Raspberry Pi NoIR Camera v2 .....	34
Figure 3.5: Schematics of the Raspberry Pi CSI Camera Connector.....	35
Figure 3.6: HC-SR04 Ultrasonic Sensor .....	37
Figure 3.7: HC-SR04 Electric Parameter .....	38
Figure 3.8: GPS Module Interfacing with Raspberry Pi .....	39
Figure 3.9: Li-Po Battery Specification .....	41
Figure 3.10: Circuit Diagram of the AI-Powered Smart Glasses .....	44

Figure 3.11: Flowchart of the AI-Powered Smart Glasses .....	46
Figure 4.1: Raspberry Pi OS Desktop Environment .....	52
Figure 4.2: YOLOv5-nano Training Graph Showing Performance Metrics Over Epochs .....	53
Figure 4.3: Object Detection Running on PC .....	54
Figure 4.4: Real-Time Object Detection on Raspberry Pi .....	56
Figure 4.5: Ultrasonic Sensor Code Snippet.....	58
Figure 4.6: Ultrasonic Sensor Implemented with Raspberry Pi .....	59
Figure 4.7: Offline Navigation Path Generated from GPS Data .....	61
Figure 4.8: Standalone View of the Integrated AI-Powered Smart Glasses System .....	63
Figure 4.9: Live Testing of the Smart Glasses Being Worn .....	64

## **LIST OF TABLES**

Table 3.1: Power Rating .....	42
Table 3.2: Summary of GPIO Pin Assignment .....	45
Appendix A: Bill of Engineering and Materials Evaluation (BEME) .....	72

# CHAPTER ONE

## INTRODUCTION

### Project Background

Visual impairment affects more than 2.2 billion people worldwide, significantly limiting their ability to interact with their environment and maintain independence (WHO, 2023). Despite advancements in assistive technologies, many individuals with partial or complete vision loss continue to face challenges in mobility, safety, and access to information—especially in low- and middle-income regions where affordability and infrastructure remain barriers.

Visual impairments are typically categorized as near or distance vision impairments. Near vision impairment refers to difficulty seeing objects up close, while distance vision impairment ranges from mild to complete blindness, depending on visual acuity thresholds (Bourne et al., 2017). The causes are diverse and include uncorrected refractive errors, cataracts, glaucoma, diabetic retinopathy, and age-related conditions (Joshi *et al.*, 2020).

Individuals with visual impairments often experience reduced independence, difficulty in learning and communication, and an increased reliance on caregivers. These limitations highlight the need for assistive tools that go beyond simple obstacle detection, offering meaningful interaction with the user's environment in real-time.

Traditional aids such as white canes and guide dogs provide some level of support but come with significant limitations. Canes are effective only for nearby obstacles and cannot detect overhanging or dynamic objects. Guide dogs, while helpful, are



expensive to train and maintain, and their availability is limited. Furthermore, many modern assistive devices rely on internet connectivity and high-end hardware, making them inaccessible to users in rural or economically disadvantaged communities (Sarkar *et al.*, 2020; Mukhiddinov & Cho, 2021).

Recent developments in artificial intelligence and computer vision present new opportunities for assistive device innovation. Object recognition and real-time feedback systems have transformed fields such as autonomous vehicles, yet their application in mobility aids for the visually impaired remains limited. Some devices focus solely on either object detection or obstacle avoidance, but few integrate both functionalities in a portable, affordable, and offline-capable solution.

The proposed system aims to address this gap by developing a pair of AI-powered smart glasses that provide real-time object detection and navigation support without requiring internet connectivity. The device combines deep learning models for object recognition with ultrasonic sensors for obstacle detection. Information is delivered through auditory feedback, enabling users to understand their environment and navigate safely.

Designed for affordability and efficiency, this system has the potential to empower visually impaired individuals with greater independence and confidence in their daily activities.

## **1.2 Problem Statement**

Visually impaired individuals face significant challenges in navigating public environments safely and independently due to the limitations of existing assistive tools such as white canes, guide dogs, and GPS-based applications, which

either lack comprehensive obstacle detection, are unaffordable, or rely on constant internet connectivity. These limitations are especially pronounced in low- and middle-income regions, where access to reliable, cost-effective assistive technology is minimal. The core problem this project addresses is the absence of an affordable, offline-capable, wearable device that integrates real-time object detection and obstacle awareness to support the mobility and situational awareness of visually impaired users.

## **Aim & Objectives**

To develop an affordable, AI-powered wearable device that enables real-time object detection, obstacle awareness, and offline navigation to support the mobility of visually impaired individuals.

The objective of this project is To:

- i. develop a low-cost wearable device for real-time AI model object detection and obstacle avoidance that runs offline on the edge system.
- ii. integrate multiple sensors (camera, ultrasonic sensors, GPS) into a unified system for enhanced environmental awareness.
- iii. incorporate an audio-based feedback system for obstacle detection and haptic alerts.

- iv. enable voice input for navigation, using offline maps for real-time route guidance.

## **Project Significance**

The AI-Powered Smart Glasses aim to bridge the accessibility gap for visually impaired individuals by providing an affordable, offline-capable, real-time assistive device. Existing tools such as white canes, guide dogs, and expensive smart devices fall short due to limited functionality, high costs, or dependency on internet access. This project addresses these limitations by integrating object detection, obstacle awareness, and sensory feedback into a single wearable system powered by embedded AI and edge computing.

Affordability is a central objective, making the device accessible to users in low- and middle-income regions where most visually impaired individuals reside. The glasses are designed to operate without internet connectivity, using a lightweight AI model on a Raspberry Pi-based system, making them suitable for deployment in rural and underdeveloped areas. This directly supports Sustainable Development Goals (SDGs), particularly SDG 3 (good health and well-being) and SDG 10 (reduced inequalities).

The project promotes independence and safety by delivering real-time auditory and haptic alerts, reducing reliance on caregivers and enhancing confidence in daily navigation. It also considers sustainability through energy-efficient components and durable materials to extend product life and reduce environmental impact.

Beyond individual users, the solution can benefit institutions such as rehabilitation centers, educational bodies, and NGOs supporting visually impaired individuals. By encouraging strategic partnerships and community-based distribution, the project has the potential for large-scale societal impact. It also contributes to research in wearable AI and assistive technology, paving the way for future enhancements in human-centered design and inclusive innovation.

## **Project Scope**

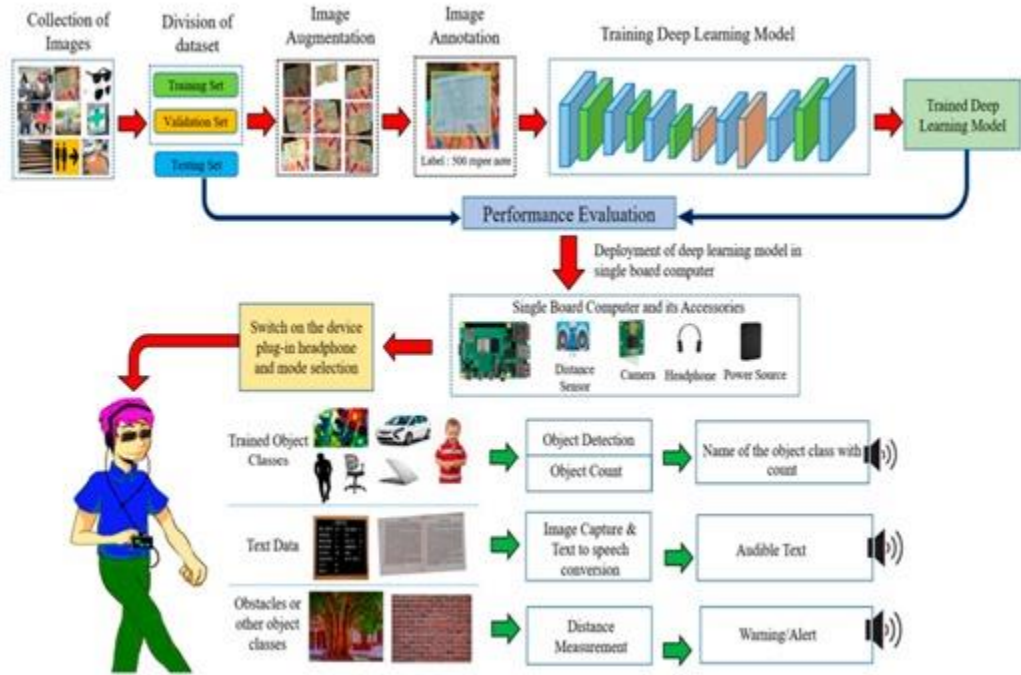
This project is focused on the design and development of AI-powered Smart Glasses to enhance mobility and environmental awareness for visually impaired individuals. The device combines real-time object detection using a lightweight deep learning model with ultrasonic sensors for obstacle detection, all operating offline on a Raspberry Pi-based embedded system. Key features include audio feedback via text-to-speech and optional haptic alerts, enabling intuitive interaction without the need for internet connectivity.

The system is designed to be affordable, portable, and scalable, targeting a production cost that ensures accessibility in low-resource settings. Hardware components include a high-resolution camera, ultrasonic sensors, GPS module, microphone, and speaker, all integrated into a wearable form factor. The software architecture supports local processing for object recognition and sensor fusion, ensuring low latency and real-time responsiveness.

The scope of this project specifically covers the **University of Ilorin**, where development, initial deployment, and performance evaluation will be conducted. The university campus provides a diverse environment for real-world testing—such as walkways, staircases, and indoor corridors—allowing for assessment of obstacle detection accuracy, voice feedback clarity, and user experience.

This scope excludes integration with cloud-based services, large-scale manufacturing, and support for languages beyond English during the initial phase.

The overall structure and functionality of the proposed system are illustrated in **Figure 1.1**. It shows how the core components—camera, ultrasonic sensors, AI model, and audio feedback—work together to provide real-time environmental awareness for visually impaired users.



**Figure 1.1: Diagram of the Proposed Methodology (Makanyadevi K et al., 2025 )**

## Project Organization

To ensure a systematic and thorough approach, the project is organized into five main chapters:

### Chapter 1: Introduction

This chapter outlines the project background, problem statement, aims, objectives, and significance, providing a foundation for the research and development.

## **Chapter 2: Literature Review**

This section delves into existing assistive technologies, comparing their advantages and limitations to establish the AI-Powered Smart Glasses. It explores related work on object detection, edge computing, and assistive devices.

## **Chapter 3: Design and Methodology**

This chapter details the hardware and software architecture of the system, including sensor integration, AI model development, and system testing protocols.

## **Chapter 4: Implementation and Testing**

This chapter presents the implementation process, showcasing the integration of components, system performance, and feedback from real-world testing.

## **Chapter 5: Conclusion and Recommendations**

The final chapter summarizes the project's achievements and offers recommendations for future improvements and scaling opportunities.

# **CHAPTER TWO**

## **LITERATURE REVIEW**

### **2.1. Introduction**

In recent years, the integration of artificial intelligence (AI) in assistive devices has gained considerable attention, offering innovative solutions to improve accessibility for visually impaired individuals. Various studies have explored different approaches to enhancing mobility, object recognition, and real-time assistance through wearable technology. However, despite these advancements, challenges such as accuracy, affordability, and user-friendliness persist.

### **2.2. Components Review**

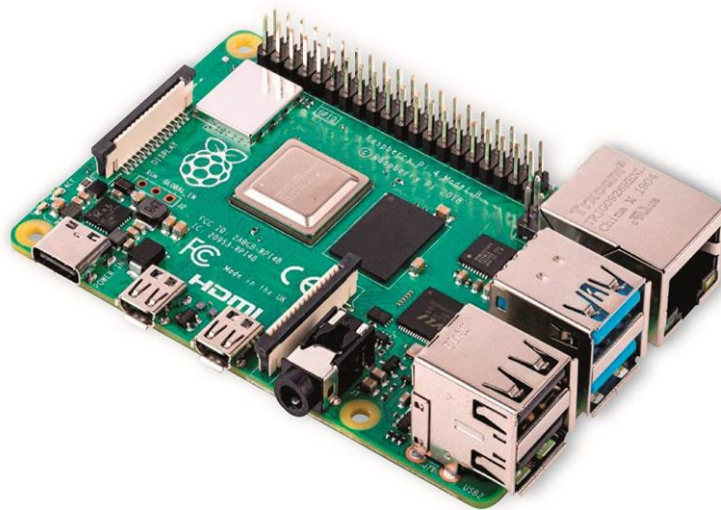
The selected components for this project are analyzed as follows. Their details and applications are explained.

#### **2.2.1 Raspberry Pi 4 Model B**

The Raspberry Pi 4 is a compact, credit-card-sized single-board computer developed by Raspberry Pi Holdings and released in June 2019. It features a quad-core ARM Cortex-A72 CPU, supports up to 8 GB of RAM, dual micro-HDMI ports with 4K display capability, and both USB 3.0 and Gigabit Ethernet connectivity (Raspberry Pi Documentation, 2025). Since its introduction, it has found widespread use in education, makerspaces, IoT systems, media centers, robotics,



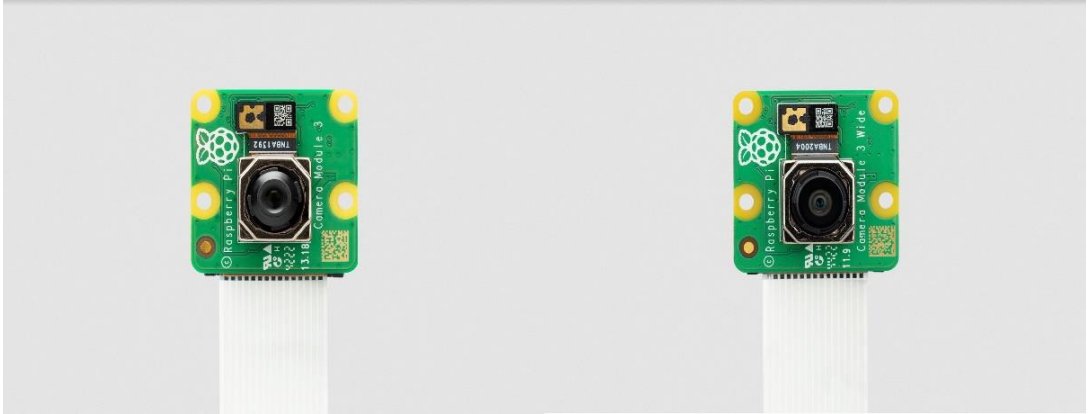
and even small-scale industrial applications due to its versatile I/O options and strong software ecosystem (DeepSea Developments, 2025). Its affordability starting at ₦53,000 Naira and robust community support make it a popular tool for learning computing and building innovative prototypes. Industry reviews highlight its performance upgrade, describing it as “world’s best single-board computer” thanks to significantly faster processing and expanded memory (Tom's Hardware, 2022), though users should note that it runs warm under heavy loads (Wired, 2019). This makes the Raspberry Pi 4 an ideal platform for affordable, compact, and high-performance embedded computing tasks. An image of the Raspberry Pi 4 Model B, showing its core interfaces and layout, is provided in **Figure 2.1**.



**Figure 2.1: Raspberry Pi 4 Model B specifications. From Raspberry Pi**  
([Raspberry pi 4B Datasheet,2024](#))

### 2.2.2 Camera Module

The Pi Camera is an official accessory from the Raspberry Pi Foundation that connects directly via the CSI interface, designed to deliver high-quality photos and videos while minimizing CPU load (Raspberry Pi, 2025). Current editions, such as the Camera Module 3, feature a 12 MP Sony IMX708 sensor, autofocus, HDR, and special NoIR (infrared) options (Tom's Hardware,2023). Compared to typical USB webcams, these camera modules offer cleaner integration, compact design, and access to hardware-accelerated video encoding—reducing latency and CPU usage (Tom's Hardware, 2023). They are created to serve a variety of projects, including surveillance, robotics, computer vision, and scientific monitoring, with strong software compatibility in Raspberry Pi OS and libraries like Python's Pi camera and OpenCV (Sinoseen, 2024). Its compact size, compatibility, and high-resolution imaging capabilities make it a preferred choice for projects involving real-time vision processing. **Figure 2.2** displays the standard Pi Camera Module, which integrates seamlessly with Raspberry Pi boards.



**Figure 2.2: Raspberry Pi Camera Module. From Raspberry Pi ([Pi Camera - Documentation,2022](#))**

### **2.2.3 Ultrasonic Sensor (HC-SR04)**

Ultrasonic sensors use high-frequency sound waves (typically between 20–40 kHz) for non-contact distance measurement by emitting pulses and analyzing the time it takes for the echoes to return. They are widely employed in robotics, industrial automation, vehicle parking aid systems, and liquid level monitoring. These sensors offer advantages such as accuracy across various target materials, immunity to lighting conditions, and reliable performance in dusty, dark, or humid environments (Renkeer, 2024). They are low-cost, easy to interface with microcontrollers, and resistant to electromagnetic interference. However, they cannot function in a vacuum, may struggle with soft or angled surfaces, and their precision can be influenced by extreme temperature changes (DigiKey, 2021). Despite these limitations, their flexibility and affordability make ultrasonic sensors

well-suited for proximity detection and obstacle avoidance tasks. A visual representation of a typical ultrasonic sensor, commonly used in embedded systems like this project, is shown in **Figure 2.3**.

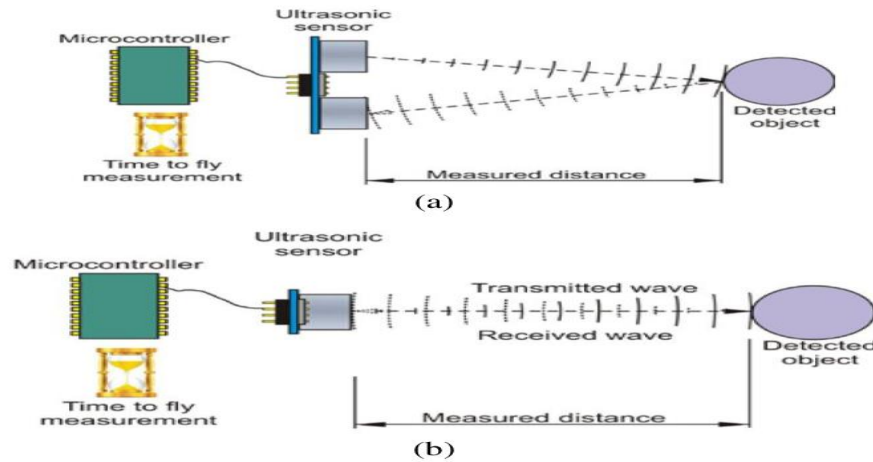


Fig. 2: The basic working principle of the ultrasonic sensor: (a). top view and (b). side view.



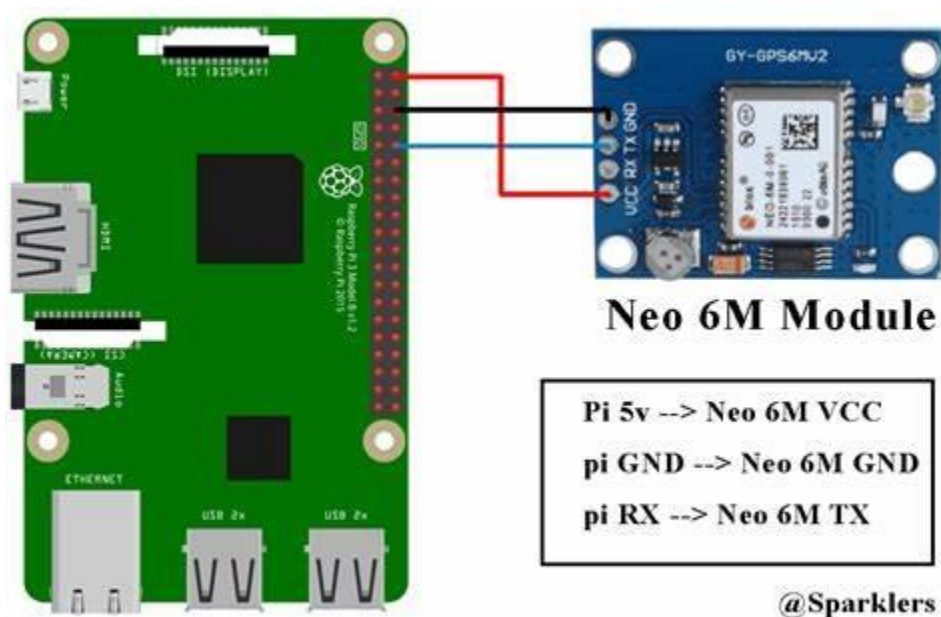
**Figure 2.3: Typical Ultrasonic Sensor Module HC-SR04**

([Ultrasonic Sensor datasheet, 2021](#))

## 2.2.4 GPS module

GPS modules, such as the ubiquitous u-blox NEO-6M, are compact units capable of receiving satellite signals from up to 22 channels to determine geographic coordinates with approximately 2.5 m accuracy. They feature fast

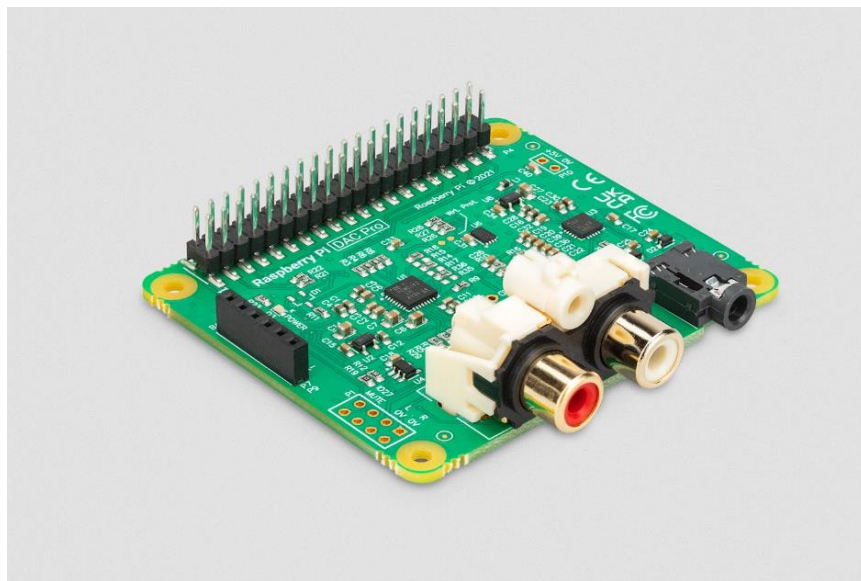
“Time-to-First-Fix” performance—typically under one second for hot starts—and include power-saving modes that reduce current consumption to around 11 mA, making them suitable for battery-powered wearable devices (LastMinuteEngineers, 2023). Commonly used in navigation, asset tracking, and timing systems, these modules are produced by companies like u-blox and are valued for their small size and low power draw. **Figure 2.4** shows a typical GPS breakout board used in embedded systems.



**Figure 2.4 NEO-6M GPS Module for Location Tracking (Neo-6M datasheet, 2023)**

## 2.2.5. Audio output (speaker)

It conveys alerts and real-time feedback, wearable systems typically include compact speakers or earbuds. These tiny audio transducers deliver sound over short distances and allow voice prompts or auditory warnings to reach the user clearly while maintaining portability. Often sourced from electronics manufacturers like Adafruit, CUI, or small OEM suppliers, these speakers offer decent fidelity, low power consumption, and easy integration through standard audio interfaces (LastminuteEngineers, 2021). **Figure 2.5** illustrates a typical small speaker module suitable for use in smart glasses.



**Figure 2.5 Compact Audio Speaker Module**

[\(compact audio output for raspberry pi,2023\)](#)

## 2.2.6. Microphone Module

Embedded microphone modules in wearable systems most often use electret condenser technology. These microphones incorporate a permanently charged diaphragm and often include a built-in JFET preamplifier, enabling them to produce clean audio signals with moderate power usage (Rose, 2019). Originating from Bell Labs and employed by companies such as Knowles and CUI, electret microphones are favored in consumer electronics for voice capture, audio processing, and voice control applications due to their compact size, reliability, and ease of integration. **Figure 2.6** provides an example of a common microphone breakout used in embedded electronics.



## Figure 2.6 Typical Electret Microphone Module

([I2S sph0645 datasheet, 2022](#))

### 2.2.7 SD Card

A micro-SD card serves as the low-cost, removable storage medium for embedded systems like Raspberry Pi, storing the operating system, applications, and any collected data. Brands such as SanDisk, Samsung, and Kingston dominate this market. Users must consider durability and speed, as many recommend industrial-grade cards—like SanDisk Max Endurance—for reliability in long-term applications (Reddit, 2022). **Figure 2.8** displays a high-quality micro-SD card typically used in Raspberry Pi projects.



**Figure 2.8: Micro-SD Card for Data Storage**

([sd card datasheet, 2019](#))



## 2.3 Power supply (Battery selection)

Lithium-polymer (LiPo) batteries are the preferred power source for portable electronics due to their high energy density, lightweight form factor, and flexible packaging options. Widely used in applications such as drones, RC vehicles, and mobile devices, these batteries have a relatively low self-discharge rate (around 5% per month) but require careful handling to avoid safety issues (Wikipedia, 2025). Manufacturers like Panasonic, Samsung, and Tattu produce LiPo batteries in various capacities designed for embedded systems where space efficiency and runtime are crucial. **Figure 2.7** shows a typical LiPo battery pack suited for wearable electronics.



**Figure 2.7: Lithium-Polymer Battery Pack for Power Supply**

[\(Lipo battery datasheet, 2018\)](#)

## **2.4 Related Works.**

This literature review examines selected studies by analyzing their objectives, methodologies, key findings, and limitations. By reviewing these aspects, we gain a comprehensive understanding of each project's purpose, the approaches used, the results obtained, and any challenges or gaps identified. This process helps highlight existing knowledge, uncover limitations in previous work, and identify opportunities for further research and improvement.

(Makanyadevi K et al., 2025 ) developed a wearable system that integrates AI-powered smart glasses with specialized shoes to enhance navigation and safety for visually impaired individuals. The system employs deep learning algorithms for real-time object detection and ultrasonic sensors for obstacle avoidance. It provides feedback through vibration motors embedded in both the glasses and shoes, along with auditory alerts to inform users of potential hazards. While the proposed methodology presents an innovative and comprehensive approach, the study lacks concrete experimental results or discussions on system limitations, leaving its real-world applicability uncertain.

(Shree Lakshmi et al., 2022) introduced a multifunctional smart glasses system designed to assist individuals who are blind, deaf, or mute. The device acts as an assistive tool, utilizing YOLO (You Only Look Once) for object detection,

Automatic Speech Recognition (ASR), and Google's speech-to-text API to provide real-time guidance. The methodology incorporates Python, OpenCV, CNN, and various APIs for object identification and navigation. The results indicate that YOLO performed well on test datasets, effectively detecting objects in real-time. For deaf users, the system converts spoken language into text, which is displayed on a HUD (head-up display). However, a significant limitation is the dependency on an active internet connection for speech-to-text translation, making it less accessible for users in areas with limited connectivity.

(Birambole et al., 2022) proposed an object detection-based assistant for visually impaired individuals. The system utilized TensorFlow and the SSD (Single Shot MultiBox Detector) algorithm for object recognition, along with text-to-speech conversion via Pyttsx3 to communicate information to the user. The methodology focused on using a microcontroller with built-in Wi-Fi to support these features. However, the study primarily detailed the theoretical framework and proposed methodologies rather than providing experimental results. Furthermore, no specific limitations were discussed, making it difficult to assess the system's accuracy and usability in real-world scenarios.

(Dematti et al., 2023) developed smart glasses aimed at enhancing mobility for blind and partially sighted individuals. The system integrated a camera, ultrasonic sensors, GPS, and facial recognition technology to assist users. A Raspberry Pi

served as the core processing unit, enabling image processing and real-time navigation assistance. The study successfully produced a low-cost prototype capable of identifying faces and providing auditory feedback, making it a cost-effective and practical solution. However, while the system demonstrated promising results, the study did not delve into challenges such as environmental adaptability, power efficiency, or processing speed, which are crucial for real-world applications.

Sweatha and Priya (2024) developed smart glasses integrating YOLOv5 for real-time object detection, OCR for text recognition, and multilingual voice command support. Their goal was to create an inclusive tool that enhances accessibility and autonomy for visually impaired users. The methodology involved AI-driven detection and voice feedback, demonstrating accurate recognition of both static and moving objects. The results, supported by loss function graphs and training tables, confirmed the system's effectiveness. However, the study emphasized the need for further optimization, particularly through the integration of Natural Language Processing (NLP) for more natural interactions.

Although Yu and Chen (2021) focused on general motion target detection rather than specifically assisting the visually impaired, their work on YOLOv4 contributes valuable insights into AI-based object recognition. The study employed YOLOv4 for detecting and identifying moving objects, presenting results through video

analysis and tabulated performance metrics such as accuracy and recall rate. The research confirmed YOLOv4's ability to meet basic detection requirements but noted that enhancements in image and data processing could improve recognition accuracy. These advancements could be applied to assistive technologies for visually impaired individuals by refining object detection in dynamic environments.

(Boobalan *et al.*, 2023) proposed an object detection system that combines YOLOv7 with Google Text-to-Speech (GTTS) to provide real-time audio guidance for visually impaired users. The methodology centered on convolutional neural networks for object recognition and audio output for navigation assistance. Experimental evaluations demonstrated the feasibility of this approach, highlighting its potential for real-world applications. However, testing was limited to webcam-based evaluations in controlled settings, raising concerns about its performance in complex environments. Additionally, the study did not address challenges related to audio guidance in noisy or fast-changing surroundings, limiting its applicability without further refinement.

Sarkar et al. (2020) introduced a smart eyewear system that utilizes the Mask R-CNN model for object detection and segmentation, converting image data into audio descriptions. The device was designed to process visual information, generate text descriptions through image captioning, and convert text into speech via a

UDA1334A audio module. The model was trained on 5000 images and achieved an impressive accuracy of 96%. Additionally, the system exhibited low latency and reduced power consumption due to the use of a Raspberry Pi Zero board. While the results were promising, the authors acknowledged that increasing the dataset could further enhance accuracy, suggesting potential future refinements.

Mandhala et al. (2020) explored machine learning-based object detection to assist visually impaired individuals in navigation by providing spatial information about obstacles. The study utilized Region-based Proposal Networks (RPN) to extract image data and vanishing points to assign depth values to segmented regions. A comparison between estimated depth maps and reference depth maps was used to gather spatial details. The results demonstrated the system's capability in estimating obstacle depth, with Resort and Dense Net models achieving average precision rates of 83.1% and 79.8%, respectively. However, the study focused primarily on methodology, lacking details on the final integrated system or its real-world application constraints.

## **2.5. Appraisal of the related works**

Recent research in assistive technologies has demonstrated significant progress in applying artificial intelligence (AI) and computer vision to support visually impaired individuals. While various systems have implemented real-time

object detection and audio feedback, many of them are limited by high computational demands, reliance on cloud services, and a narrow feedback mechanism. This project distinguishes itself by focusing on **offline operation**, **cost-effectiveness**, and **multimodal feedback**, addressing gaps commonly observed in prior works.

For instance, **Arsalwad et al. (2024)** proposed YOLOInsight, a real-time assistive system integrating NLP and OCR for personalized assistance. Although it emphasizes affordability and user interaction, it still considers future integration with cloud computing for extended functionalities, which may introduce hosting costs and potential latency. In contrast, the present project is designed to run entirely offline on a Raspberry Pi 4, thereby eliminating the need for paid cloud-based model hosting and ensuring faster local inference with reduced latency. This architectural choice contributes directly to both **affordability** and **real-time responsiveness**, making the system more practical for deployment in low-resource settings.

Similarly, **Laad et al. (2023)** developed a smart glasses prototype using YOLOv3 for real-time object detection. However, their system lacked depth estimation and was primarily reliant on visual input. While effective in object identification, it did not integrate alternative sensing modalities for spatial awareness. The current project addresses this limitation by incorporating **ultrasonic sensors** alongside the

camera module, enabling the detection of nearby obstacles that may be visually ambiguous or outside the camera's frame. This integration supports a **multimodal feedback mechanism**, providing users with both contextual object recognition and spatial alerts.

**Joshi et al. (2020)** introduced a system combining object detection with obstacle avoidance and gesture control. Although comprehensive, their work lacked emphasis on edge-based processing or system speed. Additionally, voice feedback in their system appeared basic and not closely tied to environmental sensing. In contrast, the proposed smart glasses system provides **layered feedback** through synthesized speech, triggered contextually by both object detection and ultrasonic range measurements. This ensures that the user receives accurate, timely, and relevant alerts based on real-world spatial dynamics.

Projects such as **Singh and Muthukumarswamy (2023)**, which used MobileNet SSD on a Raspberry Pi 3, also highlighted hardware constraints, as lower processing power resulted in delayed responses. This project circumvents such limitations by adopting a **lightweight YOLOv5s model** optimized for real-time performance on embedded hardware, further enhanced by eliminating network dependence.



In summary, existing literature underscores the growing sophistication of assistive AI systems but also reveals recurring limitations related to **cost, latency, overreliance on cloud processing**, and **restricted feedback modalities**. This project directly responds to these gaps by prioritizing **offline capability, hardware efficiency**, and a **multimodal feedback framework**, offering a more inclusive and scalable solution for visually impaired users in diverse environments.

## CHAPTER THREE

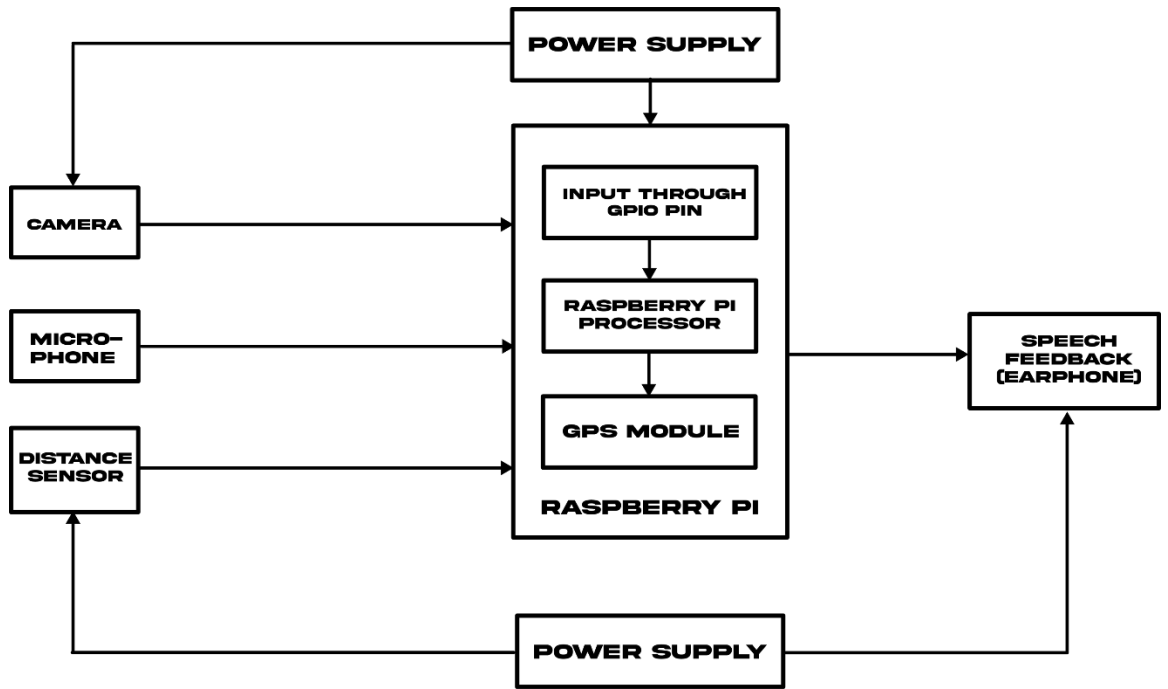
### METHODOLOGY

#### 3.1 System Overview

The AI-powered Smart Glasses are designed as a wearable assistive device that provides real-time object recognition and obstacle detection to support the mobility and situational awareness of visually impaired users. The system operates entirely offline, ensuring usability in environments with limited or no internet access. At the core of the device is a Raspberry Pi 4, which performs all local computation required for object detection, sensor data processing, and audio output.

By integrating a lightweight deep learning model with peripheral components such as a camera, ultrasonic sensors, a GPS module, and audio interfaces, the system is able to perceive the user's surroundings and communicate relevant information through voice feedback. The smart glasses are optimized for portability, cost-effectiveness, and usability, making them a practical solution for real-world application, particularly in low-resource settings. The overall architecture of the smart glasses system is illustrated in **Figure 3.1**. It demonstrates the interaction between the core components, including Input devices, processing unit, and

feedback mechanism, all working together to provide real-time assistance to visually impaired users.



**Figure 3.1: Block Diagram of the AI-Powered Smart Glasses System**

The block diagram in **Figure 3.1** presents a high-level view of the system architecture, highlighting the flow of data and control between key hardware components. The system is built around a Raspberry Pi, which serves as the central processing unit responsible for data acquisition, analysis, and feedback generation.

**Raspberry Pi Processor:** At the core of the system, the Raspberry Pi processes all incoming data—both visual and spatial—and performs object detection, spatial

analysis, and feedback generation. It also controls data fusion between camera input and sensor readings.

**Camera:** Positioned at the front of the glasses, the camera captures continuous video input of the user's environment. This visual data is sent to the Raspberry Pi for object detection processing using a pre-trained deep learning model.

**Microphone:** The microphone allows the system to capture ambient sound and, in future iterations, user voice commands. It transmits audio input to the Raspberry Pi for optional voice interaction or noise detection capabilities.

**Distance Sensor (Ultrasonic):** This sensor detects obstacles in close proximity that may not be easily identified by the camera (e.g., transparent or low-contrast objects). It calculates the distance using sound wave reflection and sends the measurements to the Raspberry Pi.

**GPS Module:** This component provides location data that can be used for orientation and navigation assistance. It connects to the Raspberry Pi through its serial interface or GPIO pins.

- **Speech Feedback (Earphone):** Once objects or obstacles are identified, the Raspberry Pi converts the information into natural language descriptions

using a text-to-speech engine. This audio is relayed to the user via an earphone or speaker.

- **Power Supply:** The entire system is powered by a rechargeable lithium-polymer battery. The power supply supports both the Raspberry Pi and peripheral devices, ensuring mobile, untethered operation.

This block diagram reinforces the system's design principles: **portability, real-time processing, offline operation, and intuitive feedback delivery**. Each component plays a crucial role in supporting visually impaired users with enhanced situational awareness and mobility assistance.

## 3.2 Component Review

### 3.2.1 Raspberry Pi 4

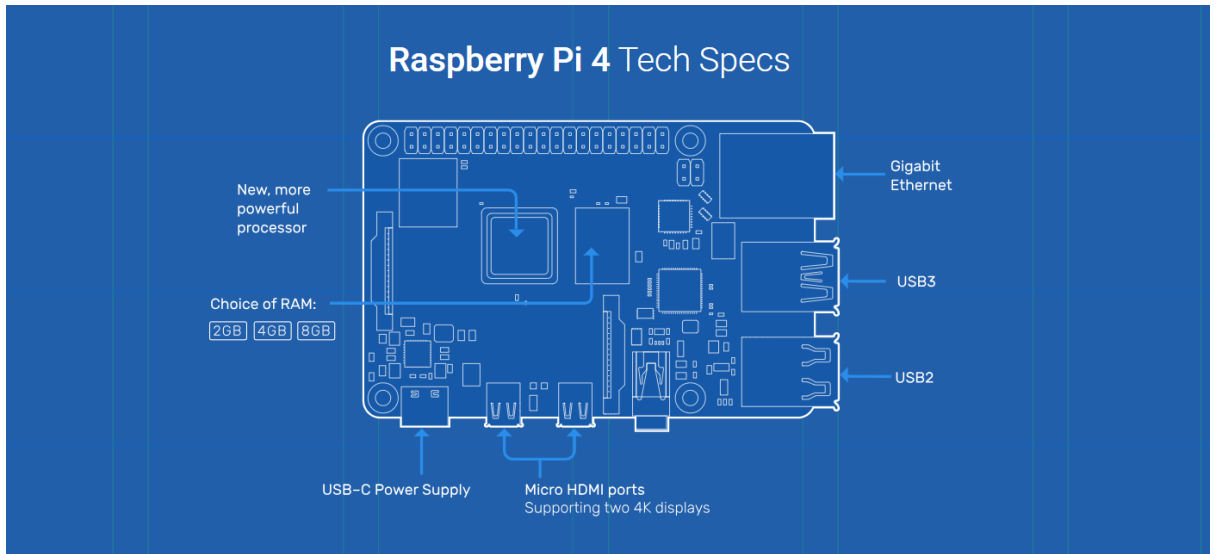
The Raspberry Pi 4 is powered by a Broadcom BCM2711 system-on-chip (SoC), featuring a quad-core 64-bit ARM Cortex-A72 processor clocked at 1.8 GHz. It comes in multiple memory configurations, offering 1GB, 2GB, 4GB, or 8GB of LPDDR4-3200 SDRAM depending on the model variant. For connectivity, the device supports dual-band wireless networking over 2.4 GHz and 5.0 GHz using IEEE 802.11ac Wi-Fi standards, along with Bluetooth 5.0 and BLE (Bluetooth Low Energy). It also includes Gigabit Ethernet for high-speed wired communication. The board is equipped with two USB 3.0 ports and two USB 2.0 ports, as well as

the standard 40-pin GPIO header for peripheral integration, fully backward-compatible with earlier Raspberry Pi models.

Display and video capabilities are supported through two micro-HDMI ports capable of 4Kp60 output, along with a 2-lane MIPI DSI display port and a 2-lane MIPI CSI camera interface. Multimedia support includes H.265 video decoding at 4Kp60, H.264 decoding at 1080p60, and encoding at 1080p30. The board also supports OpenGL ES 3.1 and Vulkan 1.0 for advanced graphics rendering. Audio and composite video output are provided via a 4-pole stereo jack.

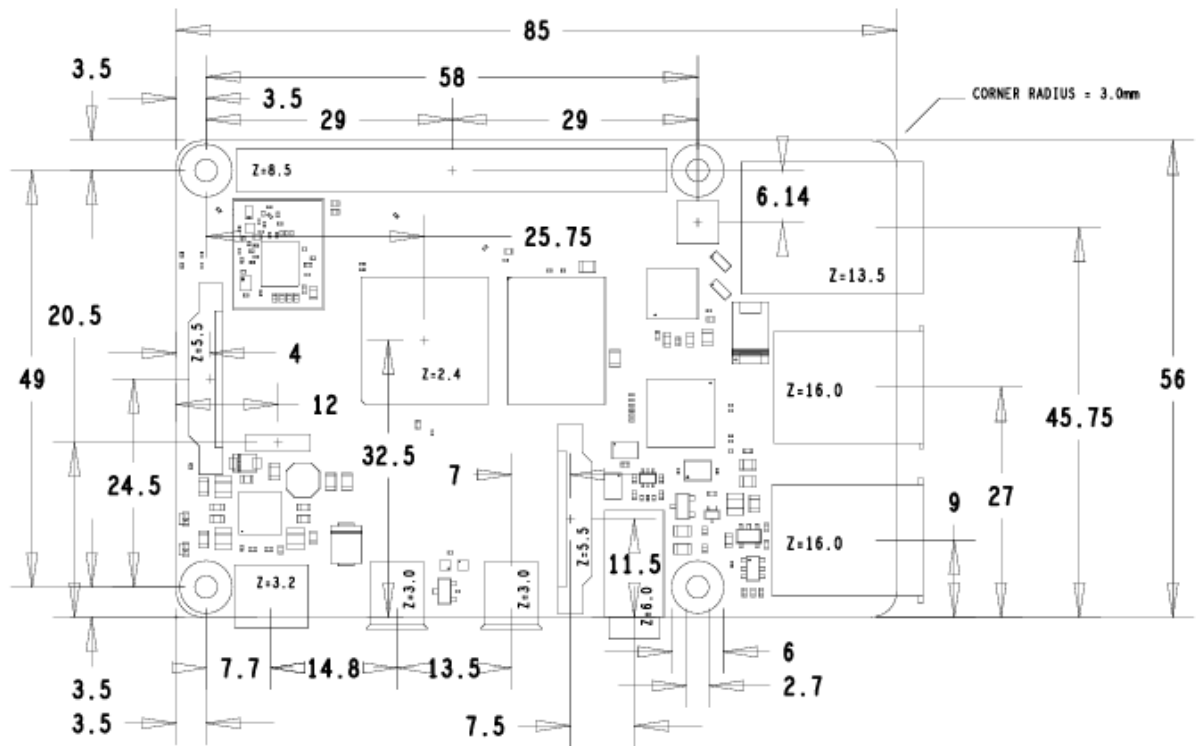
For storage, the Raspberry Pi 4 uses a micro-SD card slot, which is also the medium for booting the operating system and storing data. It can be powered either via a 5V USB-C connector or through the GPIO header, with a minimum current supply of 3A. Additionally, it supports Power over Ethernet (PoE) when used with an optional PoE HAT. The board operates within an ambient temperature range of 0 to 50 degrees Celsius, making it suitable for most indoor embedded applications.

**Fig 3.2** shows the specifications of the Raspberry pi 4B.



**Fig 3.2: Raspberry Pi 4 Tech Specs** (Raspberry pi 4 Datasheet,2024)

The Mechanical Dimensions of the Raspberry Pi 4 can be seen in **Fig 3.3**



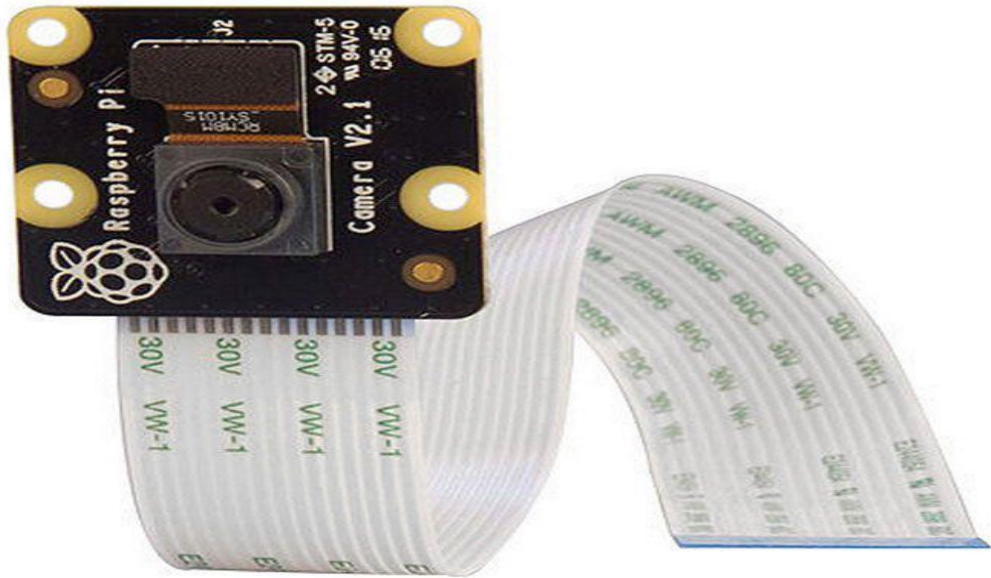
**Fig 3.3: Mechanical Dimension of the Raspberry Pi 4B**

### 3.2.2 Pi Camera Module

The Raspberry Pi NoIR Camera Module v2 is a high quality 8 megapixel Sony IMX219 image sensor custom designed add-on board for Raspberry Pi, featuring a fixed focus lens. It's capable of 3280 x 2464 pixel static images, and also supports 1080p30, 720p60 and 640x480p60/90 video. It attaches to Pi by way of one of the small sockets on the board upper surface and uses the dedicated CSI interface, designed especially for interfacing to cameras.



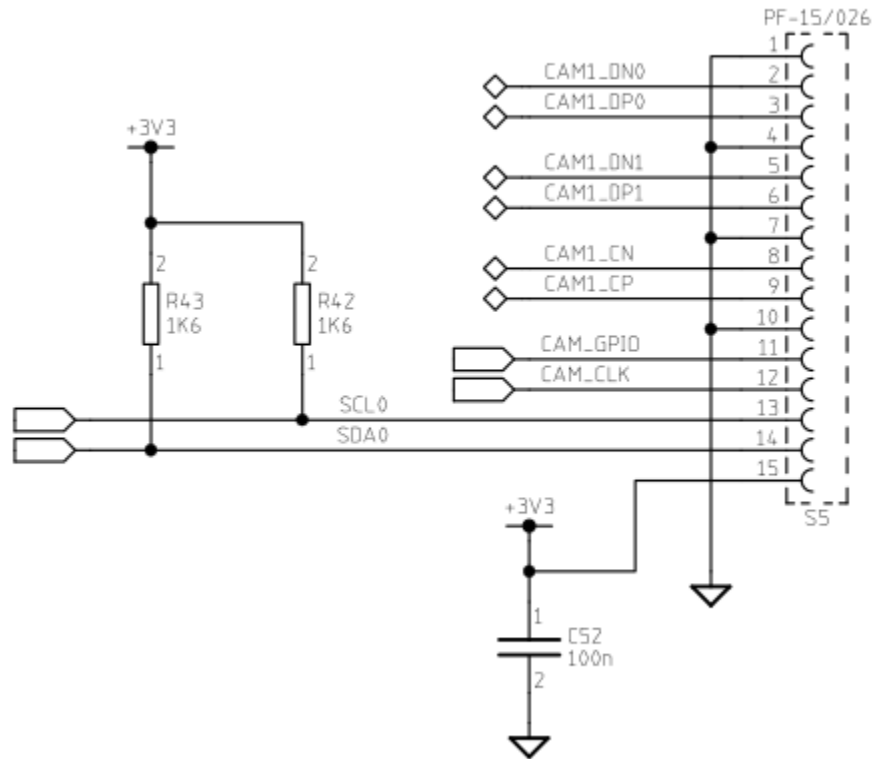
- 8 megapixel native resolution sensor-capable of 3280 x 2464 pixel static images
- Supports 1080p30, 720p60 and 640x480p90 video
- Camera is supported in the latest version of Raspbian, Raspberry Pi's preferred operating system The board itself is tiny, at around 25mm x 23mm x 9mm. It also weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. It connects to Raspberry Pi by way of a short ribbon cable. The high quality Sony sensor itself has a native resolution of 8 megapixel, and has a fixed focus lens on-board. In terms of still images, the camera is capable of 3280 x 2464 pixel static images, and also supports 1080p30, 720p60 and 640x480p90 video. The NoIR Camera has No InfraRed (NoIR) filter on the lens which makes it perfect for doing Infrared photography and taking pictures in low light (twilight) environments. It can be used for Infrared photography, Low light photography, Monitoring plant growth, CCTV security camera.( Raspberry Pi NoIR Camera v2 Datasheet,2023). An image of the Raspberry Pi NoIR Camera v2 is seen in **Fig 3.4**



**Fig 3.4: Raspberry Pi NoIR Camera v2(Raspberry pi Datasheet,2023).**

The Raspberry pi CSI camera connector schematics for easy connection is seen in

**Fig 3.5**



**Fig 3.5: Schematics of the Raspberry pi CSI Camera connector (Raspberry pi Datasheet,2023).)**

### 3.2.3 Ultrasonic Sensors (HC-SR04)

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules

includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:

- (1) Using IO trigger for at least 10us high level signal
- (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
- (3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.

Test distance = (high level time  $\times$  velocity of sound (340M/S) / 2.

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground. (Elec Freaks,2022).

**Fig 3.6:** shows the image of an ultrasonic sensor and how it measures distance by sending and receiving signals.

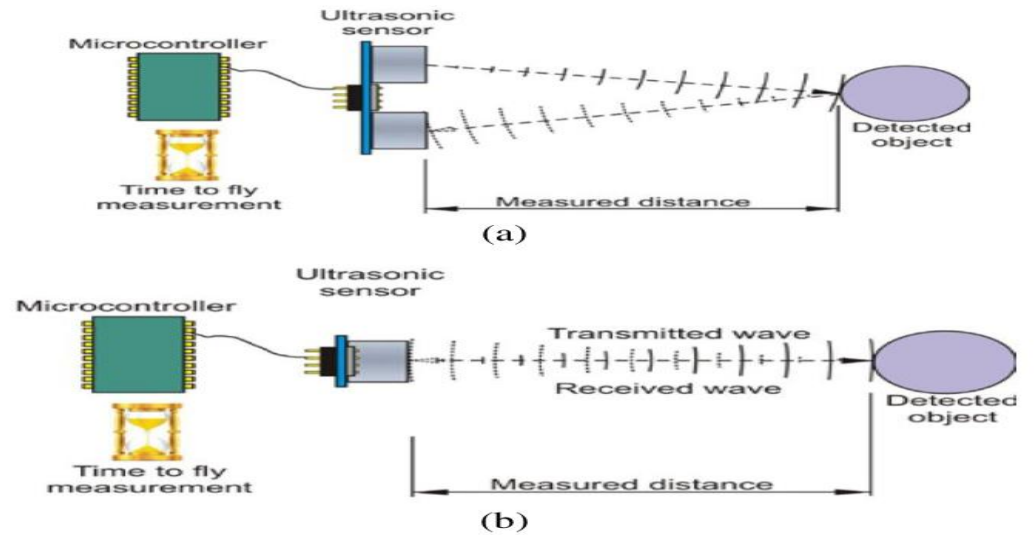


Fig. 2: The basic working principle of the ultrasonic sensor: (a). top view and (b). side view.



Fig 3.6: HC-SR04 Ultrasonic Sensor ([Ultrasonic sensor datasheet](#))

**Fig 3.7** shows the electric parameter of the ultrasonic sensor, dimension and it's power consumption.

## Electric Parameter

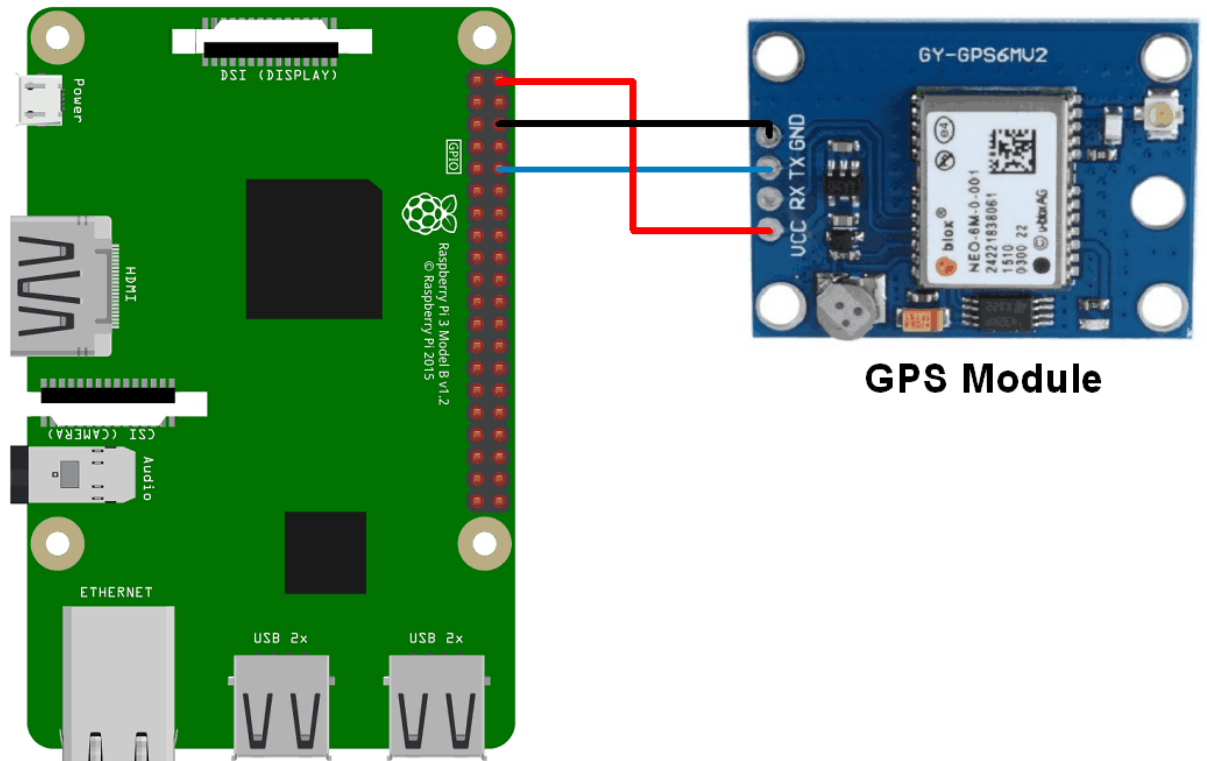
<b>Working Voltage</b>	<b>DC 5 V</b>
<b>Working Current</b>	<b>15mA</b>
<b>Working Frequency</b>	<b>40Hz</b>
<b>Max Range</b>	<b>4m</b>
<b>Min Range</b>	<b>2cm</b>
<b>MeasuringAngle</b>	<b>15 degree</b>
<b>Trigger Input Signal</b>	<b>10uS TTL pulse</b>
<b>Echo Output Signal</b>	<b>Input TTL lever signal and the range in proportion</b>
<b>Dimension</b>	<b>45*20*15mm</b>

**Fig 3.7: HC-SR04 Electric Parameter**

### 3.2.4 GPS Module (GY-GPS6MV2)

The Global Positioning System (GPS) determines its position on Earth by receiving radio frequency signals sent by satellites in space and ground stations on Earth. These signals contain time stamps indicating when they were transmitted. By calculating the difference between the transmission and reception times and knowing the speed at which the signals travel, the GPS receiver can determine the distance to each satellite. Using this distance data from three or more satellites, the GPS can triangulate its exact position. Notably, the GPS itself does not need to transmit any information; it only receives signals. The GPS receiver module typically communicates with a controller or PC terminal using UART

communication (electronicWings,2025). **Fig 3.8** shows the GPS Module interfacing with Raspberry pi.



**Fig. 3.8: GPS Module Interfacing with Raspberry Pi**  
([Raspberry Pi Gps Module Interfacing With Raspberry Pi](#))

### 3.2.5 Power Supply

The DNK 503450 is a 3.7V Lithium Polymer (Li-Po) battery featuring a nominal capacity of 1000mAh and a nominal voltage of 3.7V. It is designed with a charge voltage of 4.2V and an internal impedance of less than 60m $\Omega$ . The battery supports a recommended charge and discharge current of 0.2C, with a maximum continuous discharge current of 1C and a pulse discharge capability of 3C for 10 milliseconds. It has a discharge cut-off voltage of 2.5V  $\pm$  0.05V and delivers an output voltage range between 4.2V and 2.5V. This battery offers a cycle life of approximately 500 cycles at 0.2C under 25°C conditions. Its environmental operating conditions include a charging temperature range of 0–45°C and a discharging range of -20–60°C. For storage, it can be kept safely for up to 6 months within 0–35°C, while shorter periods allow wider ranges. Dimensionally, the battery measures 5mm in thickness (T), 34mm in width (W), and 50mm in length (L), with a lead length (L<sub>1</sub>) of 52mm and a connector pin height (M) of 10mm. The DNK 503450 also comes with a 1-year warranty. **Fig 3.9** shows the battery specification and environmental specification of the Li-Po battery 3.7V.

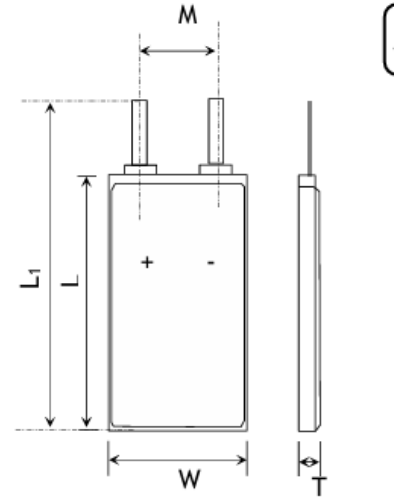


### BATTERY SPECIFICATION

Part Number	DNK 503450
Nominal Voltage	3.7 V
Nominal Capacity	1000 mAh
Internal Impedence	<60mΩ
Charge Voltage	4.2V
Recommended Charge Current	0.2 C
Allowed Max Charge Current	0.5 C
Output Voltage Range	4.2 - 2.5 V
Recommended Discharge Current	0.2 C
Max Continuous Discharge Current	1 C
Pulse Discharge Current	3 C (10 mS)
Discharge Cut-off Voltage	2.5± 0.05V
Cycle Life to 80% Health	500 (0.2C, 25°C)

### ENVIRONMENTAL SPECIFICATIONS

Charging Temperature Range	0 - 45° C
Discharging Temperature Range	-20 - 60° C
Storage Period 1 Week	-20 - 45° C
Storage Period 1 Month	-10 - 45° C
Storage Period 6 Months	0 - 35° C



Item	Parameter (mm)
T	5
W	34
M	10
L	50
L <sub>1</sub>	52
BL	52 ± 1

**Fig 3.9: Li-Po Battery Specification** ([battery-503450-3.7V-1000mAh-Lithium-Polymer-Battery-Specification.pdf](#))

### 3.2.6 Power Calculation

To effectively power the AI-powered smart glasses, it is important to understand the voltage, current, and power requirements of each hardware component used in the system. **Table 3.1** below outlines the key electrical specifications of all major

components, including their operating voltage, typical current draw, and power consumption. It also calculates the total power and current ratings based on the quantity of each component used. These values are essential for selecting an appropriate power supply and ensuring the system operates efficiently and reliably under typical working conditions.

**Table 3.1: Power Rating**

S/N	Component	Voltage (V)	Current (A)	Power (W)	Quantity	Total Power (W)	Total Current (A)
1	Raspberry Pi 4 Model B	5	2.5	12.5	1	12.5	2.5
2	Pi Camera Module v2 (NoIR)	3.3	0.25	0.825	1	0.825	0.25
3	HC-SR04 Ultrasonic Sensor	5	0.015	0.075	2	0.15	0.03
4	GPS Module (NEO-6M)	3.3	0.045	0.149	1	0.149	0.045
5	Speaker (Mini 0.5W 8Ω)	5	0.1	0.5	1	0.5	0.1
6	Microphone (Electret)	3.3	0.0005	0.00165	1	0.00165	0.0005
7	microSD Card (Class 10)	3.3	0.1	0.33	1	0.33	0.1
8	<b>Total</b>					<b>14.46</b>	<b>3.06</b>

### 3.2.7 Power Supply Design and Calculation

To power the AI-powered smart device system efficiently, a stable 5V power source with sufficient current capacity is essential. The system comprises

components like the Raspberry Pi 4, Pi Camera, GPS module, ultrasonic sensors, microphone, speaker, and microSD card. These components collectively draw a total current of approximately **3.06A** and consume about **14.46 watts** of power.

$$\text{Energy per hour} = \text{Power} \times \text{Time} = 14.46\text{W} \times 1\text{hr} = 14.46\text{Wh}$$

So the system needs approximately **14.5Wh per hour** of operation.

To meet the power demands, a **USB power bank** with the following specifications is proposed:

- **Output Voltage:** 5V
- **Output Current:** Minimum 3A (15W output)
- **Capacity:** 20,000mAh @ 3.7V (industry standard for rating)
- **Energy Content:**

$$\text{Energy} = 3.7\text{ V} \times 20,000\text{ mAh} = 74\text{ Wh}$$

Taking into account the **voltage step-up conversion losses (typically ~85% efficient)** when boosting 3.7V to 5V:

$$\text{Usable Energy} = 74\text{ Wh} \times 0.85 = 62.9\text{ Wh.}$$

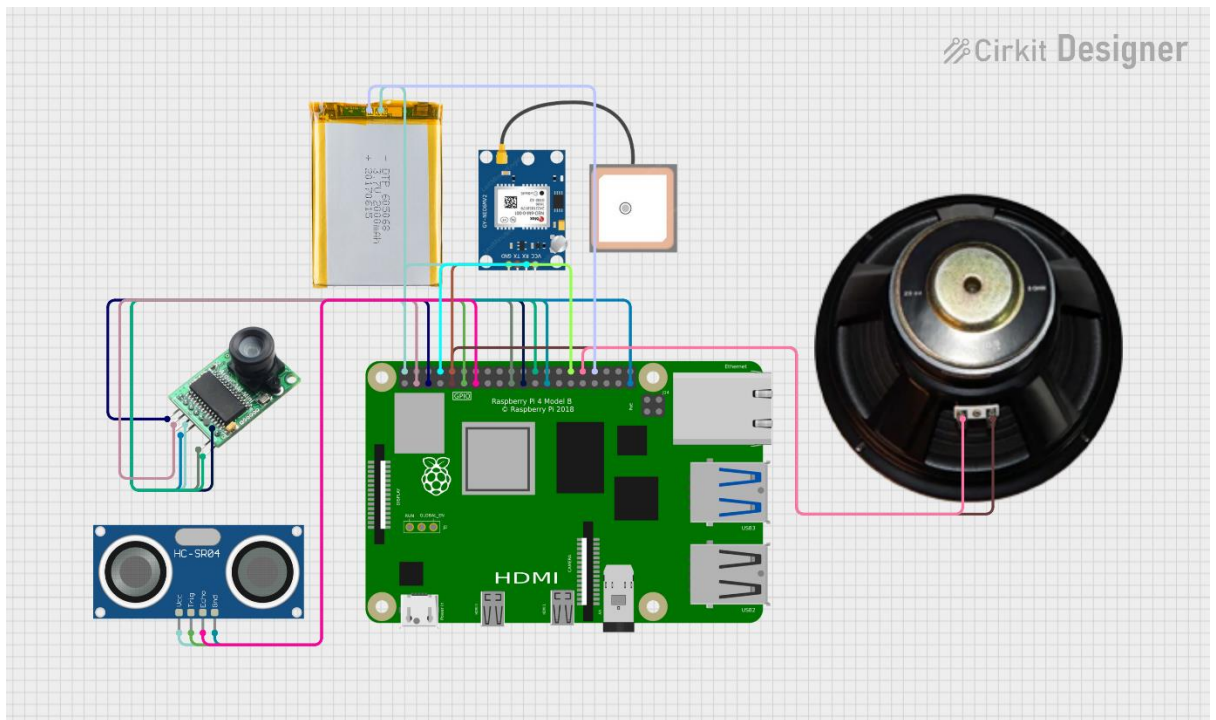
### **Estimated Runtime**

$$\text{Runtime} \frac{62.9\text{ Wh}}{14.5\text{ W}} \approx 4.34\text{ hours}$$

So a **20,000mAh power bank** will be able to power the system for **approximately 4 hours** of continuous operation.

### 3.2.8 Circuit Diagram

The circuit diagram in **Figure 3.10** illustrates the complete electrical interconnection of all components in the AI-Powered Smart Glasses system. The Raspberry Pi 4 Model B serves as the central processing unit, coordinating data flow between all peripheral devices through various communication interfaces.



**Fig 3.10: Circuit Diagram of the AI Powered Smart Glasses**

### 3.2.8.1 Camera Module Connection

The Raspberry Pi NoIR Camera v2 connects directly to the Raspberry Pi through the dedicated CSI (Camera Serial Interface) connector.

### 3.2.8.2 Ultrasonic Sensor Interface (HC-SR04)

The ultrasonic sensor connects to the Raspberry Pi GPIO pins as follows:

- **VCC:** Connected to 5V GPIO pin for power supply
- **GND:** Connected to ground GPIO pin
- **Trigger Pin:** Connected to GPIO pin (e.g., GPIO 18) for sending trigger pulses
- **Echo Pin:** Connected to GPIO pin (e.g., GPIO 24) for receiving echo signals

### 3.2.8.3 GPS Module Connection (GY-GPS6MV2)

The GPS module interfaces with the Raspberry Pi through UART communication:

- **VCC:** Connected to 3.3V GPIO pin
- **GND:** Connected to ground GPIO pin
- **TX:** Connected to UART RX pin (GPIO 15)
- **RX:** Connected to UART TX pin (GPIO 14)

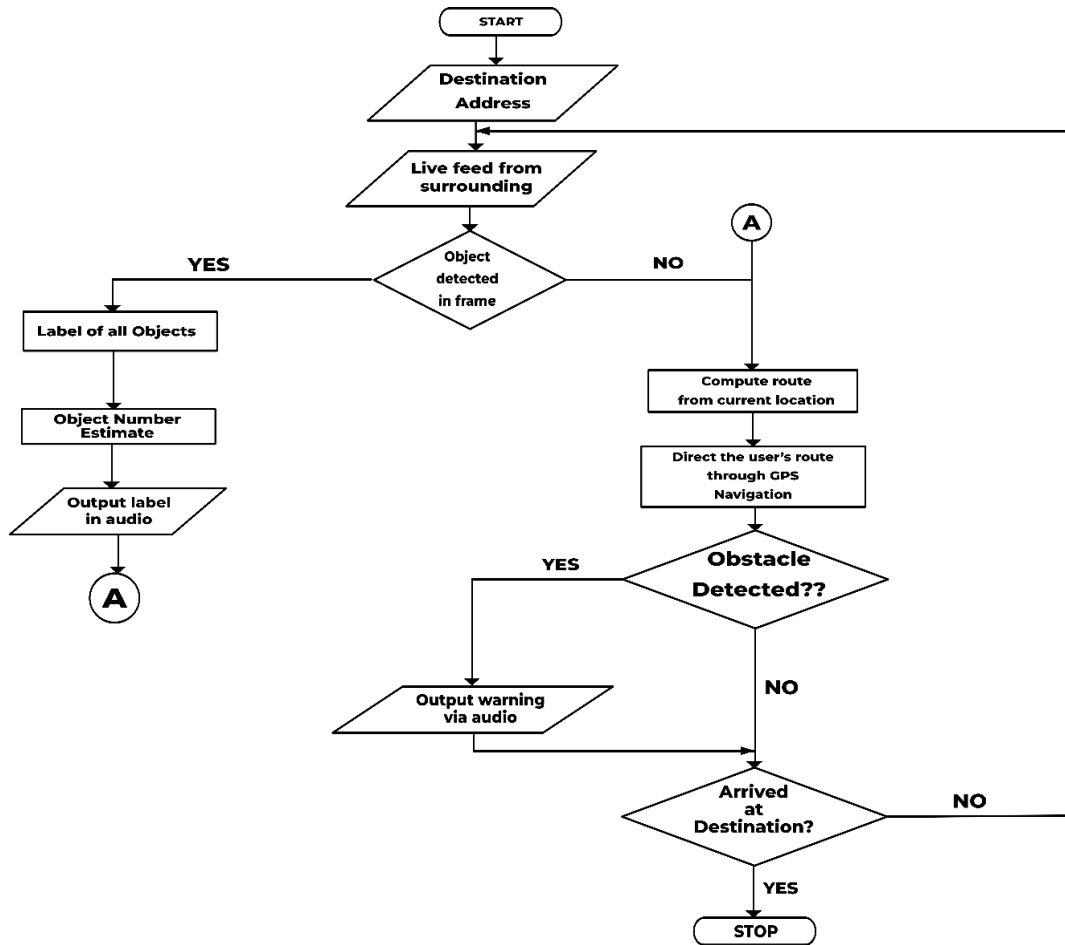
**Table 3.2: Summary of GPIO Pin Assignment**

Component	Pin Number	GPIO Function	Purpose
Ultrasonic Trigger	Pin 12	GPIO 18	Send trigger pulse
Ultrasonic Echo	Pin 18	GPIO 24	Receive echo signal
GPS TX	Pin 8	GPIO 14	UART transmission
GPS RX	Pin 10	GPIO 15	UART reception
Power LED	Pin 16	GPIO 23	System status indicator
Audio Ground	Pin 6	GND	Audio reference

**Table 3.2** above is the summary of the GPIO pins used in developing the smart glasses with each purpose of the GPIO pin stated accordingly.

### 3.3 Software Development

The software development phase of the AI-powered smart glasses system focuses on integrating object detection, obstacle avoidance, and audio feedback in a cohesive and efficient manner. To provide a clear understanding of how the system processes data and responds to the environment, the flowchart in **Fig 3.11** below illustrates the logical sequence of operations—from capturing input to delivering real-time navigation assistance to the user.



**Fig 3.11: Flowchart of the AI powered Smart Device**

The flowchart below illustrates the overall operational logic of the AI-powered smart glasses system designed for visually impaired users. It outlines the sequence of decision-making and data processing that enables the device to perform real-time object detection, obstacle avoidance, and route guidance using audio feedback.

The process begins with capturing a **live feed from the surrounding environment** via the onboard Pi Camera. The system then analyzes the video input to determine

whether any objects are present within the frame. If objects are detected, the system proceeds to label all identifiable objects using a pre-trained deep learning model (YOLOv5n), estimate their number, and then provide descriptive **audio feedback** to the user through the speaker or earphones. This supports situational awareness and helps the user navigate dynamic environments in real time.

On the other hand, if no object is detected and the user intends to navigate to a specific location, the system prompts for a **destination address**. Using the GPS module, the system calculates a route from the user's current position and begins guiding them using **GPS-based navigation**. As the user moves, the ultrasonic sensors continuously monitor for nearby **obstacles** that may not be captured visually (e.g., glass doors or low-contrast objects). If an obstacle is detected, an **audio warning** is immediately issued to prevent collision.

The system checks periodically whether the user has reached their destination. Once the GPS confirms arrival, the system halts its operations, concluding the navigation task.

This flowchart ties directly into the system's architecture and component functionalities discussed in previous sections—highlighting the synergy between computer vision, sensor fusion, and audio feedback that enables robust, **offline-capable**, and **user-friendly** assistive navigation.



### **3.3.1 AI Model Development and Training Process**

The software development phase began with the creation of a custom dataset specifically tailored to the University of Ilorin environment, which aligns with the project's defined scope. This dataset was processed through Roboflow, a comprehensive platform that provided essential data augmentation capabilities and object detection preprocessing tools to enhance the training data quality and diversity. The augmented dataset was then utilized to fine-tune a YOLOv5 Nano model, chosen for its optimal balance between detection accuracy and computational efficiency suitable for embedded systems.

The actual model training was conducted on Google Colab, leveraging its GPU acceleration capabilities to handle the computationally intensive deep learning training process. This cloud-based approach enabled efficient model development while generating comprehensive training graphs and performance metrics that validated the model's learning progress. Upon successful training completion, the resulting model was converted to a lightweight version specifically optimized for embedded system deployment, ensuring reduced computational overhead and faster inference times while maintaining detection accuracy.

### 3.3.2 Component-Level Software Implementation

The software aspect of the system involved writing and integrating code for each hardware component to ensure coordinated and responsive behavior. Below is a breakdown of how each module was implemented programmatically to achieve the intended assistive functionalities:

#### 3.3.2.1 Ultrasonic Sensor

The ultrasonic sensor (HC-SR04) was coded to continuously monitor the space directly ahead of the user. A threshold distance of **1 meter** was set using a conditional statement in the code. If an object is detected **within this range**, the system triggers an **audio alert** to inform the user of a nearby obstacle. However, if the object is **beyond 1 meter**, the system ignores it to reduce unnecessary distractions and ensure the user can still hear audio alerts from object detection through the camera. This improves clarity and enhances the overall user experience.

#### 3.3.2.2 Camera and Object Detection

The Pi Camera was initialized to perform **real-time object detection** using a custom-trained YOLOv5-nano model. The model was fine-tuned to detect only relevant objects that pose a risk to visually impaired individuals—such as **moving vehicles or people**. In the detection loop, a **maximum of 3 objects per frame** was allowed to limit information overload and prioritize critical obstacles. Additionally,

object **counting logic** was implemented to keep track of how many times a particular object appeared, which can later be used to determine movement or persistence in the user's environment. The system outputs spoken warnings via TTS for any hazardous object identified in the frame.

### 3.3.2.3 GPS Module and Navigation

For navigation, the **GPS module** was used to track the user's **latitude and longitude** coordinates. To maintain offline functionality, a **local map of the University of Ilorin** was pre-downloaded and stored in the system. Routing algorithms (based on shortest path logic) were implemented using this local map data to guide the user from their **current location to a voice-specified destination**. The GPS updates in real time, and voice instructions are provided as the user moves along the route. This allows seamless, internet-independent navigation.

To make destination input user-friendly, we integrated **voice recognition using OpenAI's Whisper model** to interpret the user's spoken commands. Once the desired location is recognized, the system processes it, maps it on the local data, and initiates navigation. Throughout the journey, feedback is provided using **eSpeak for text-to-speech conversion**, allowing the system to speak out directions and warnings.

# CHAPTER FOUR

## IMPLEMENTATION, TESTING AND RESULTS

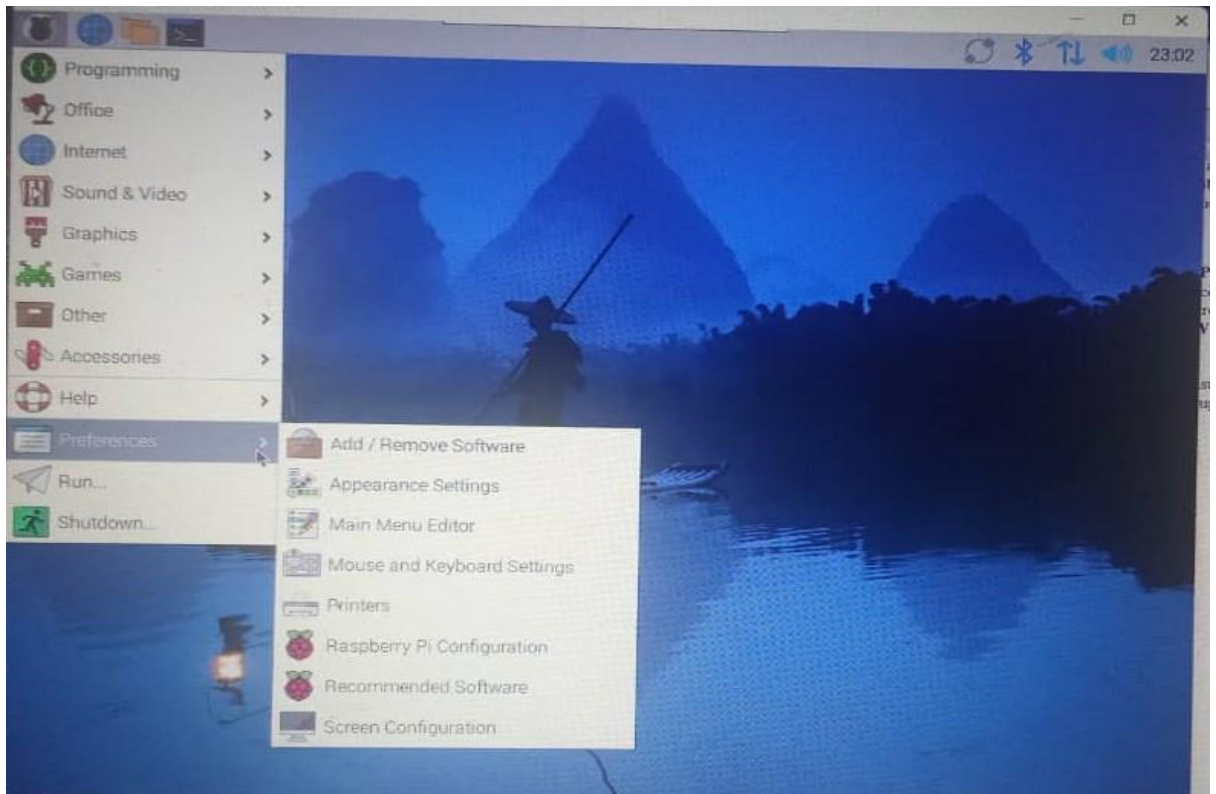
### 4.1 Overview

The system architecture consists of a Raspberry Pi 4 as the central processing unit, connected to a Pi Camera, ultrasonic sensors, a GPS module, a microphone, and a speaker or earphone. The Pi runs a lightweight object detection model (YOLOv5-nano) offline, while peripheral sensors handle obstacle and location awareness. All output is delivered to the user through real-time audio feedback.

### 4.2 Raspberry Pi Setup

The Raspberry Pi 4 was configured using the **Raspberry Pi Imager**, through which the official Raspberry Pi OS was flashed onto a high-endurance microSD card. After initial setup, a project-specific Python virtual environment was created, and all required libraries—including **PyTorch**, **OpenCV**, **RPi.GPIO**, **eSpeak**, and **Whisper** dependencies—were installed.

The Pi is powered by a **20,000mAh USB power bank**, ensuring stable power delivery through its USB-C input port. This portable power setup supports untethered field testing, especially in outdoor environments. **Fig 4.1** shows the pictorial Raspberry Pi OS desktop environment.

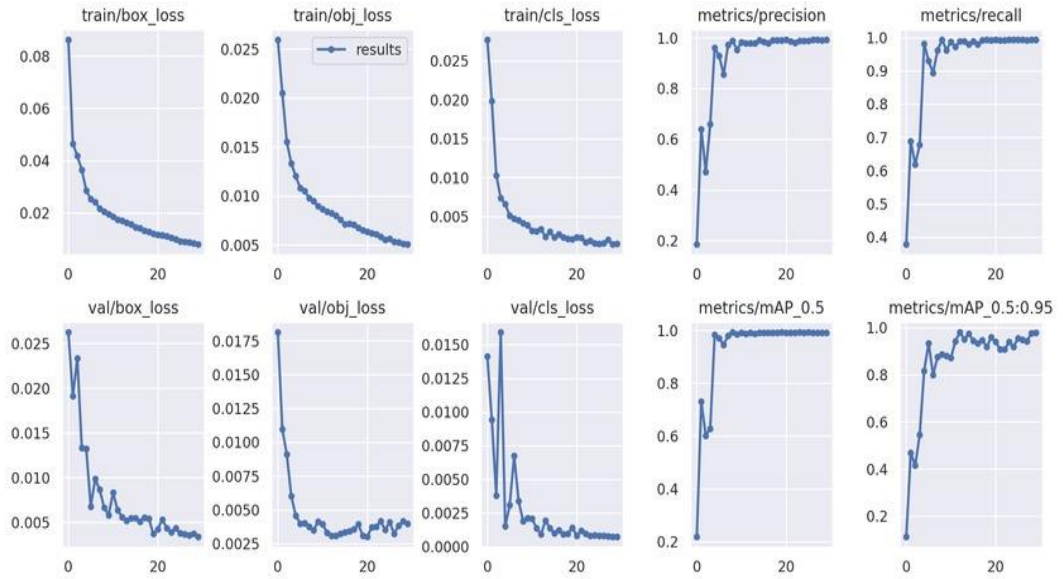


**Fig 4.1: Raspberry Pi OS desktop environment**

### **4.3 Model Training and Optimization**

The object detection model used in this project is a custom-trained version of **YOLOv5-nano**, optimized for embedded deployment. Training was carried out on **Google Colab** using a dataset collected from the University of Ilorin, with **Roboflow** handling annotation, preprocessing, and data augmentation. After training, the model was exported and converted to a lightweight format compatible with the Raspberry Pi.

The training process yielded strong results in terms of loss reduction and model accuracy. The graph below as seen in **Fig 4.2** shows the training performance, including classification loss, objectness loss, and mAP (mean Average Precision) metrics across epochs.



**Figure 4.2: YOLOv5-nano Training Graph showing performance metrics over epochs**

#### 4.4 Object Detection Model Testing (PC Simulation)

Before deploying the object detection model to the Raspberry Pi, it was first tested successfully on a personal computer to validate functionality and performance. The model, trained using YOLOv5-nano on a custom dataset collected from the University of Ilorin, was designed to detect only critical obstacles such as people and moving vehicles. To reduce the cognitive load on users, the detection logic was

limited to a **maximum of three objects per frame**. This ensures that the most relevant threats are prioritized in real-time audio feedback.

The model was tested using a webcam connected to the PC, simulating the live input that will later come from the Pi Camera. Detection results were visually displayed along with confidence scores and bounding boxes as seen in **Fig 4.3**



**Figure 4.3: Object detection running on PC showing real-time bounding boxes and class labels.**

## 4.5 Object Detection Deployment on Raspberry Pi.

Following successful testing on a personal computer, the custom-trained YOLOv5-nano model was deployed on the Raspberry Pi 4 for real-time object detection using the onboard Pi Camera. As shown in **Figure 4.4**, the model accurately identifies and labels objects in the user's environment, displaying bounding boxes and class names on the captured video frames.

To ensure the reliability of feedback and reduce cognitive load on the user, multiple optimizations were implemented. First, the system was configured to **filter objects by a confidence threshold of 50%**, meaning only detections with a high likelihood were processed and announced. This reduced false positives and helped focus the system's attention on clearly identifiable objects.

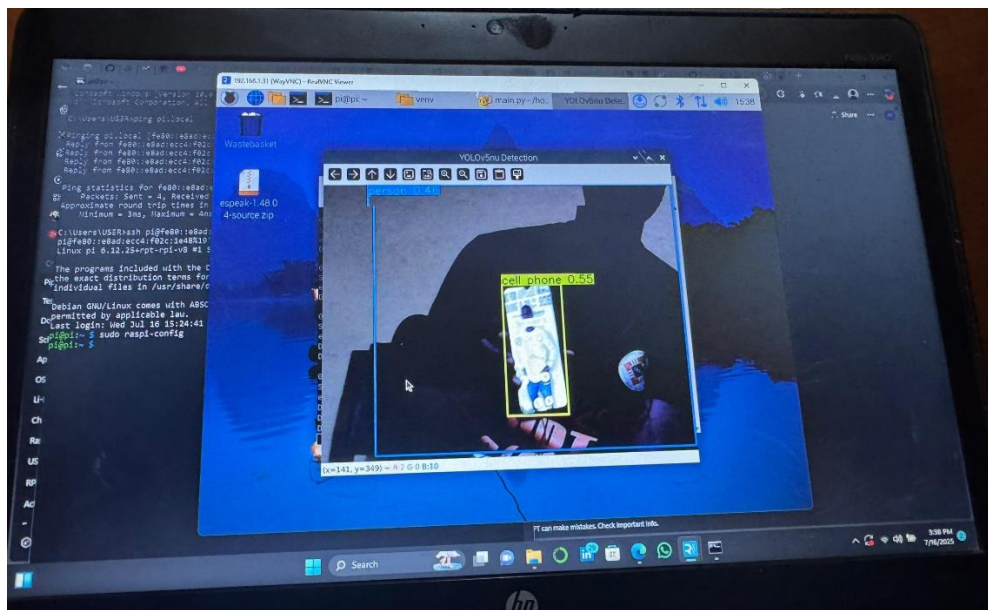
Additionally, to avoid overwhelming the user with too much information per frame, the system was limited to **a maximum of three objects per frame**. This ensured that only the most relevant objects—typically the closest or most prominent—were communicated.

To further enhance clarity and timing of spoken feedback, a **frame-skipping strategy** was implemented. This technique allowed the system enough time to **complete text-to-speech output** before processing and announcing new objects from subsequent frames. By doing so, it prevented speech overlap and ensured that



the user could fully hear and understand each spoken alert. The result is a more natural, steady flow of audio feedback that aligns with the user's walking pace and surroundings.

This real-time deployment on the Raspberry Pi demonstrated the feasibility of running a lightweight AI model for object detection directly on embedded hardware without cloud dependency. The performance remained stable during extended use, with detections being accurate and voice feedback delivered promptly.



**Figure 4.4: Real-time object detection running on Raspberry Pi with bounding boxes and class labels displayed on screen.**

## 4.6 Proximity-Based Audio Feedback Using Ultrasonic Sensor

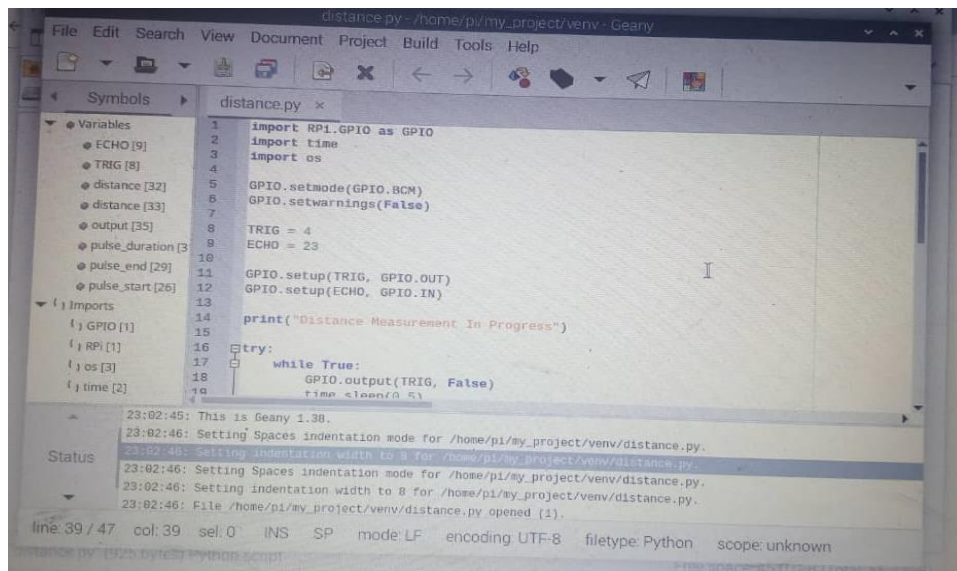
A key functionality implemented in this project is the use of an ultrasonic sensor (HC-SR04) to detect nearby obstacles and provide adaptive audio feedback based on proximity. The system is designed to alert the user using both spoken distance updates and variable-frequency beeps, depending on how close the user is to an object.

When an obstacle is detected at a distance between **50 cm and 100 cm**, the system communicates the **exact distance** to the user through speech feedback—for example, "Obstacle ahead at 85 centimeters." This provides the user with informative, non-intrusive spatial awareness while still allowing freedom of movement.

As the user gets closer—specifically within the **30 cm to 50 cm** range—the system switches to a **slow, periodic beep**, indicating increased proximity and prompting the user to proceed cautiously.

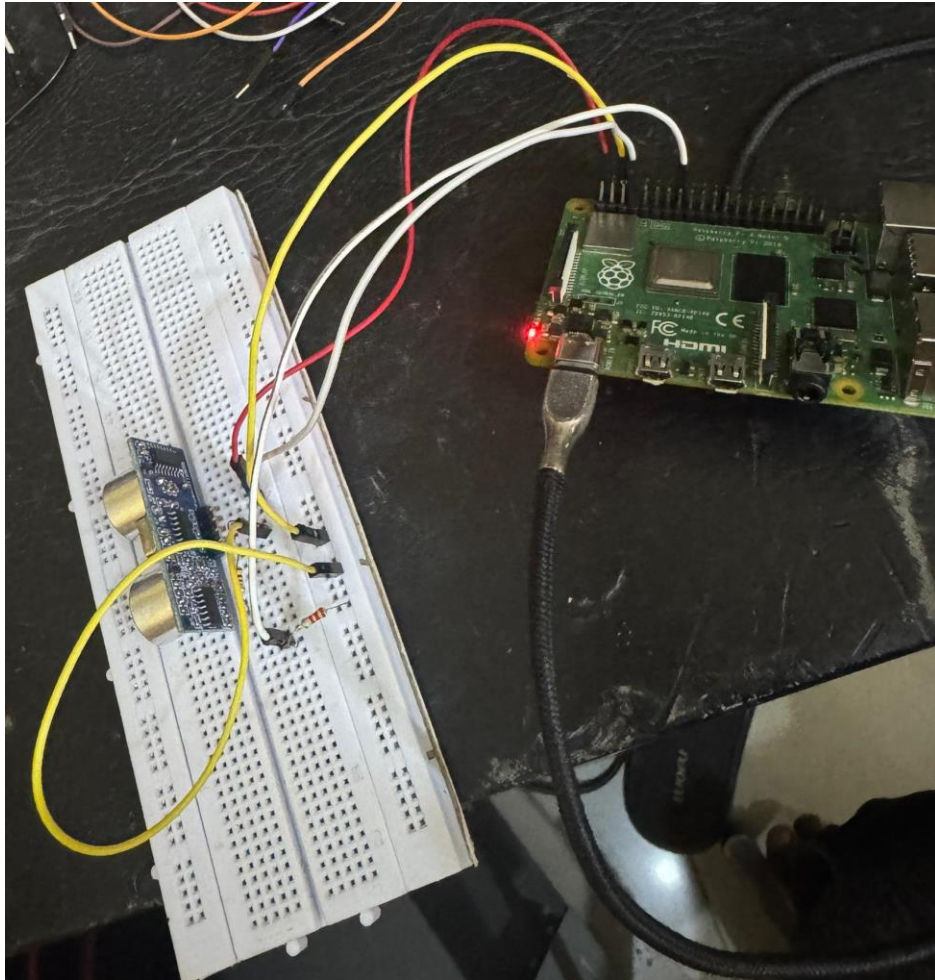
For **distances of 20 cm or less**, the system emits **rapid beeps**, signaling immediate proximity to an obstacle and warning the user to stop or adjust direction to avoid a collision.

This dynamic audio feedback mechanism enhances the system's usability by offering intuitive, real-time obstacle alerts. It supplements the camera-based object detection by covering scenarios where objects may be transparent, low-contrast, or outside the camera's field of view. The sensor was tested in various settings, and results confirmed its reliability and responsiveness across different distances and surface types. **Fig. 4.4** shows the code snippet while **Fig 4.5** shows the ultrasonic sensor implemented with the Pi



```
1 import RPi.GPIO as GPIO
2 import time
3 import os
4
5 GPIO.setmode(GPIO.BCM)
6 GPIO.setwarnings(False)
7
8 TRIG = 4
9 ECHO = 23
10
11 GPIO.setup(TRIG, GPIO.OUT)
12 GPIO.setup(ECHO, GPIO.IN)
13
14 print("Distance Measurement In Progress")
15
16 try:
17     while True:
18         GPIO.output(TRIG, False)
19         time.sleep(0.5)
```

**Fig 4.4: Ultrasonic Sensor code snippet**



**Fig 4.5: Ultrasonic Sensor implemented with the Raspberry pi.**

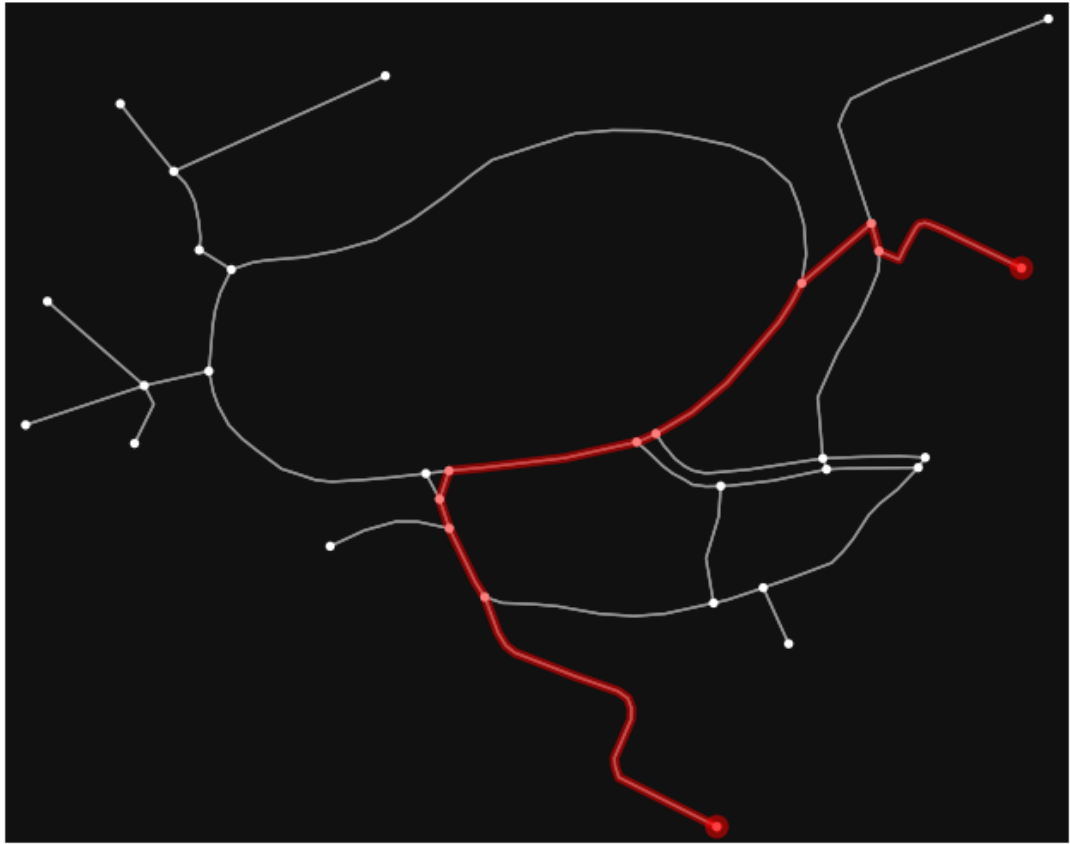
## **4.7 Navigation Assistance**

Navigation functionality has now been fully implemented and tested directly on the Raspberry Pi. The system successfully obtains real-time GPS coordinates from the NEO-6M GPS module, enabling live tracking of the user's location.

For offline navigation, map data was downloaded from OpenStreetMap, with the full African dataset used to extract and isolate the University of Ilorin region. This localized map was then integrated into the system and used to compute walking routes entirely offline.

When a destination is selected, the system calculates an optimal path based on the user's current GPS coordinates and provides voice-based step-by-step guidance using the eSpeak text-to-speech engine. This guidance is dynamically updated as the user moves, allowing for continuous direction assistance.

The navigation system was tested on the mapped region of the University of Ilorin, and the visual output confirmed accurate route generation from the starting point to the destination. This real-time navigation path is shown in **Figure 4.6**, demonstrating the route a visually impaired user would follow based on GPS updates.



**Figure 4.6: Offline navigation path generated from GPS data on the Raspberry Pi.**

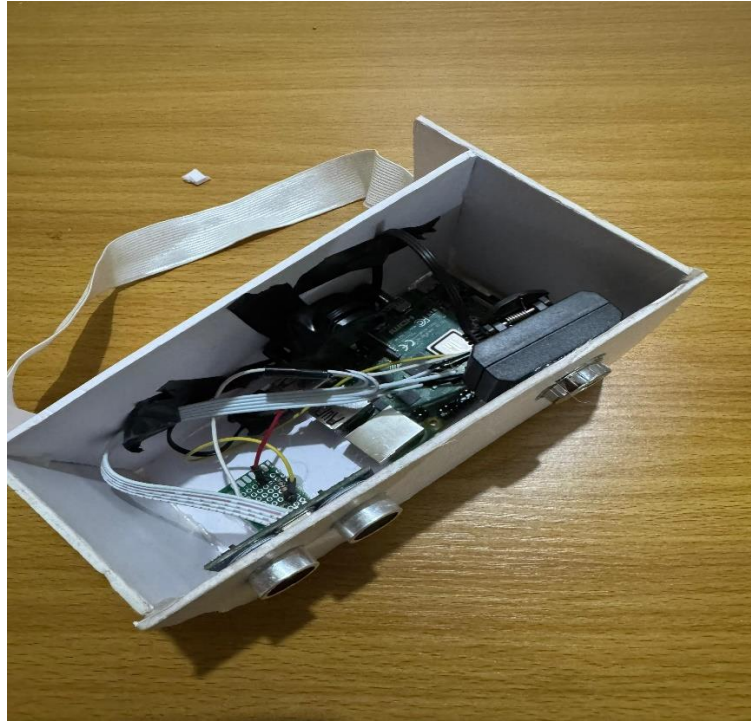
## **4.8 Full System Integration and Field Testing**

After completing the individual modules for object detection, ultrasonic-based proximity alerts, and offline GPS navigation, all components were successfully integrated into a single, wearable system. The final prototype is designed to operate

completely offline, combining real-time object recognition, adaptive audio feedback, and GPS-guided navigation in one cohesive solution.

As shown in **Figure 4.7**, the **standalone device** demonstrates the complete setup running independently, with all sensors and processing units active. The system was then worn and tested in real-world environments around the University of Ilorin campus to validate its usability and performance. **Figure 4.8** shows the system being worn during one of these live tests.

During field evaluation, the system performed as expected: it detected and announced nearby objects with high confidence, alerted the user to obstacles through adaptive beeping and speech, and guided the user along predefined routes using voice-based directions from the offline navigation module. The testing confirmed the system's real-time responsiveness, accuracy, and potential to support safe and independent mobility for visually impaired users.



**Figure 4.7: Standalone view of the fully integrated AI-powered smart glasses system.**





**Figure 4.7: Live testing of the smart glasses being worn during navigation in a real-world environment.**

## **4.9 Result and Summary**

This chapter presented the full implementation, testing, and evaluation of the AI-powered smart device system designed for visually impaired users. The Raspberry Pi environment was successfully configured with all necessary libraries and tools, including the lightweight YOLOv5-nano object detection model, text-to-speech engine (eSpeak), and supporting Python modules for sensor control and navigation.

Each core component of the system—object detection, ultrasonic obstacle sensing, and GPS-based offline navigation—was tested independently and later integrated into a cohesive, wearable solution. The object detection module accurately

identified relevant objects in real time, while the ultrasonic sensor provided adaptive feedback based on proximity. The GPS module, combined with offline OpenStreetMap data of the University of Ilorin, enabled voice-guided navigation without internet connectivity.

Field testing of the fully assembled device confirmed that the system operates reliably, providing timely audio alerts and navigational instructions. The integration of all components into a functional, standalone unit demonstrates the system's potential as an affordable, offline-capable assistive device for enhancing mobility and environmental awareness for visually impaired users.

These results validate the project's objectives and establish a strong foundation for further optimization, real-world deployment, and future enhancements in accessibility-focused wearable technology.

## **CHAPTER FIVE**

### **CONCLUSION AND RECOMMENDATIONS**

#### **5.1 Conclusion**

This project successfully developed a functional prototype of an affordable, AI-powered smart device aimed at improving mobility and spatial awareness for visually impaired users. The system integrates a fine-tuned YOLOv5-nano object detection model, ultrasonic sensors, and a NEO-6M GPS module, all running on a Raspberry Pi 4 to provide real-time, offline object detection, obstacle avoidance, and voice-guided navigation.

All core modules were implemented and tested—both individually and as a fully integrated wearable device. Live field trials at the University of Ilorin demonstrated the device’s potential to assist users in navigating real-world environments safely and independently. The object detection system provided auditory alerts based on object confidence levels, while the ultrasonic module offered adaptive proximity feedback. The GPS module successfully tracked the user’s location and computed offline navigation paths using localized OpenStreetMap data.

Although the system presents some performance constraints, the initial outcomes confirm that it aligns with the project’s core goals of accessibility, offline

functionality, affordability, and usability in low-resource environments. This work lays a solid foundation for future iterations and advancements in AI-based assistive technology.

## **5.2 Limitations**

Despite the functional success of the prototype, a number of limitations were identified during implementation and testing:

- 1. Latency in Object Detection:**

The current YOLOv5-nano model, though optimized for Raspberry Pi, still causes noticeable lag (about 10–15 seconds) between object capture and voice output. This reduces real-time responsiveness and may hinder performance in fast-changing environments.

- 2. Offline Navigation Constraints:**

Offline navigation support is limited due to the availability of routing tools and data formats compatible with embedded systems. Routing accuracy and granularity were constrained by the limited offline mapping libraries available.

- 3. Model Accuracy on Edge Hardware:**

The object detection model performs with acceptable accuracy but not at the same level as when run on more powerful systems. This is due to

hardware limitations of the Raspberry Pi and may affect detection in complex scenes.

**4. Camera Field of View and Quality:**

The current USB Camera module provides a limited field of view, which may not be sufficient to capture a wide enough scene in real-world use.

**5. Power Integration and Weight:**

While the system runs on a power bank, integrating a battery directly into the wearable form factor without adding excessive weight remains a challenge.

### **5.3 Recommendations**

To enhance the performance, usability, and scalability of the system in future iterations, the following recommendations are proposed:

**1. Model Optimization for Embedded Deployment:**

Convert the trained YOLOv5-nano model to more lightweight formats such as TensorFlow Lite, ONNX, or NCNN. These versions are better optimized for real-time inference on edge devices and may reduce latency significantly.

**2. Upgrade Camera Module:**

Use a wide-angle or higher-resolution camera module to improve environmental coverage and object detection accuracy.

**3. Improved Power Integration:**

Design and test a custom battery housing that integrates directly into the glasses frame, maintaining comfort and wearability while improving portability.

**4. Internet-Enabled Option:**

Include optional internet connectivity so that the system can download or access more powerful AI models in environments where reliable internet is available.

**5. Multi-language and Accessibility Features:**

Expand support for multiple languages to accommodate a broader user base. Include accessibility features such as adjustable speech speed, vibration feedback, or gesture input.

**6. Expanded Testing and Feedback:**

Conduct user testing across a wider range of environments—including crowded urban areas, indoor facilities, and rural settings. Gather feedback from visually impaired individuals to guide usability improvements.

**7. Future Hardware Upgrade:**

Consider porting the system to more efficient hardware platforms like

NVIDIA Jetson Nano, Google Coral, or Raspberry Pi 5 for better performance while maintaining portability.

## REFERENCES

- Arsalwad, G., Dabhade, S., Shaikh, K., & D'silva, S. (2024). YOLOInsight: Artificial intelligence-powered assistive device for visually impaired using Internet of Things and real-time object detection. *Cureus Journal of Computer Science*. <https://cureusjournals.com/articles/2353>
- Birambole, A., Bhagat, P., Mhatre, B., & Abhyankar, A. (2022). Blind person assistant: Object detection. *International Journal for Research in Applied Science and Engineering Technology*, 10(3), 1168–1172.
- Boobalan, P., Bhuvanikha, S., Sivapriya, M., & Sivakumar, R. (2023). Object detection with voice guidance to assist visually impaired using YOLOv7. *International Journal for Research in Applied Science and Engineering Technology*, 11(4), 764–768.
- Bourne, R. R. A., Flaxman, S. R., Braithwaite, T., Cicinelli, M. V., Das, A., Jonas, J. B., ... Zheng, Y. (2017). Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: A systematic review and meta-analysis. *The Lancet Global Health*, 5(9), e888–e897. [https://doi.org/10.1016/S2214-109X\(17\)30293-0](https://doi.org/10.1016/S2214-109X(17)30293-0)
- Dematti, G., Teggi, A., Naik, L., & Guddadamani, D. (2023). Smart glasses for visually impaired persons. *International Journal of Research Publication and Reviews*, 4, 2778–2782. [www.ijrpr.com](http://www.ijrpr.com)
- Joshi, R. C., Yadav, S., Dutta, M. K., & Travieso-Gonzalez, C. M. (2020). Efficient multi-object detection and smart navigation using artificial intelligence for visually impaired people. *Entropy*, 22(9). <https://doi.org/10.3390/e22090987>
- K, M., MV, A., S V, C., S, K., S A, B. K., & S, D. (2025). AI-powered smart glasses and shoes for the visually impaired. In *2025 6th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI)* (pp. 1595–1603). IEEE. <https://ieeexplore.ieee.org/document/10883641/>
- Mukhiddinov, M., & Cho, J. (2021). *Electronics article*. <https://doi.org/10.3390/electronics>



- Ramya Priyatharsini, T. G., Arif, A. A., Ahamed, A. A., Kesavan, P., & Senguttuvan, K. (2024). Smart glass for visually impaired people with facial recognition using IoT and machine learning. *International Journal of Research Publication and Reviews*, 5, 12010–12013. [www.ijrpr.com](http://www.ijrpr.com)
- Salau, H. B. (n.d.). An Artificial Intelligence of Things (AIOT) based assistive smart glass for the visually impaired. <https://www.vanguardngr.com/2015/10/42-out-of-every-1000->
- Sarkar, T., Patel, A., & Arjunan, S. P. (2020). Design and development of a smart eye wearable for the visually impaired. In *Communications in Computer and Information Science* (Vol. 1170, pp. 208–221). Springer.
- Shree Lakshmi, R., Sneha, M., Swetha, K., & Thilagavathy, A. (2022). AI-powered smart glasses for blind, deaf, and dumb. In *5th IEEE International Conference on Advances in Science and Technology, ICAST 2022* (pp. 280–285). IEEE.
- Soham Ganesh, J., Chandrakant, D. A., Ramesh, W. S., & Gangadhar, B. B. (2023). Smart glasses with blind assistance system: Real-time object detection with voice alerts. *International Journal of Research Publication and Reviews*, 9(6), 353–374. [www.ijariie.com](http://www.ijariie.com)
- Sweatha, R., & Sathiya Priya, S. (2024). YOLOv5 driven smart glasses for visually impaired. *International Journal of Science and Research Archive*, 12(1), 353–374.
- Yu, H., & Chen, W. (2021). Motion target detection and recognition based on YOLOv4 algorithm. *Journal of Physics: Conference Series*, 2025(1). <https://doi.org/10.1088/1742-6596/2025/1/012004>

## APPENDICES

### APPENDIX A: Bill of Engineering and Materials Evaluation (BEME)

S/N	Component	Quantity	Unit cost (N)	Total cost (N)
1	Raspberry pi 4 (4GB RAM)	1	120,000	120,000
2	USB Camera	1	13,000	13,000
3	HYSRF05 5pin Ultrasonic sensor	1	4,300	4,300
4	Raspberry pi SD card 64gb	1	13,500	13,500
5	Resistors	2	300	600
6	Jumper wires and connectors	1	2,500	2,500
7	Development kit (Vero board)	1	1,000	1,000
8	Power Bank (10,000mAh)	1	15,000	15,000
9	Airpod	1	7,500	7,500
10	Lightweight Frame Material(foam board)	1	2,500	2,500
11	NEO-6M V2 GPS module	1	9,000	9,000
12	INMP441 omnidirectional microphone module	1	5,300	5,300
13	Installation tools and accessories	1	10,000	10,000
14	Miscellaneous		15,000	15,000
	<b>Total</b>			<b>206,200</b>

## APPENDIX B: Raspberry Pi Codes

```
import cv2
from ultralytics import YOLO
import RPi.GPIO as GPIO
import time
import os
import subprocess
import threading

# === GPIO Setup for Ultrasonic Sensor ===
TRIG = 4
ECHO = 23

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

# === Helper Functions ===

def measure_distance():
    GPIO.output(TRIG, False)
    time.sleep(0.05)

    GPIO.output(TRIG, True)
```

```

time.sleep(0.00001)
GPIO.output(TRIG, False)

pulse_start = time.time()
timeout = pulse_start + 0.04

while GPIO.input(ECHO) == 0 and time.time() < timeout:
    pulse_start = time.time()

pulse_end = pulse_start
while GPIO.input(ECHO) == 1 and time.time() < timeout:
    pulse_end = time.time()

try:
    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration * 17150
    return round(distance, 2)
except Exception as e:
    print("Distance measurement error:", e)
    return -1

def speak(text):
    print(f"Speaking: {text}")
    safe_text = ".join(c for c in text if c.isalnum() or c.isspace())
    subprocess.run(['espeak', safe_text], stdout=subprocess.DEVNULL,
stderr=subprocess.DEVNULL)

```

```

def beep(pattern='slow'):
    beep_path = '/home/pi/my_project/venv/censor-beep-1-372459.wav'
    if not os.path.exists(beep_path):
        print("Beep file not found.")
        return

    if pattern == 'slow':
        os.system(f'aplay {beep_path}')
        time.sleep(0.5)
    elif pattern == 'fast':
        for _ in range(3):
            os.system(f'aplay {beep_path}')
            time.sleep(0.1)
    elif pattern == 'very_fast':

        for _ in range(6):
            os.system(f'aplay {beep_path}')
            time.sleep(0.05)

# === YOLO Model Setup ===
model_path = '/home/pi/my_project/venv/yolov5nu.pt'
model = YOLO(model_path)

cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)

```

```

# Shared flag to pause distance speech during object announcements
object_detected_flag = threading.Event()

running = True

# Distance speaking loop running in a separate thread
def distance_loop():
    last_spoken_time = 0
    distance_speaking_interval = 2 # seconds

    while running:
        distance = measure_distance()

        if distance == -1:
            time.sleep(0.1)
            continue

        if distance > 100:
            # Do NOT speak or beep if distance is greater than 100cm
            time.sleep(0.1)
            continue

        # Pause distance announcements if object is detected and being spoken
        if object_detected_flag.is_set():
            time.sleep(0.1)
            continue

```

```

current_time = time.time()
if current_time - last_spoken_time >= distance_speaking_interval:
    if 50 < distance <= 100:
        speak(f"The object is {int(distance)} centimeters away")
    elif 30 < distance <= 50:
        beep('slow')
    elif 20 < distance <= 30:
        beep('fast')
    elif distance <= 20:
        beep('very_fast')

    last_spoken_time = current_time

time.sleep(0.1)

# Start distance measurement thread
distance_thread = threading.Thread(target=distance_loop)
distance_thread.daemon = True
distance_thread.start()

print("Starting smart vision system. Press 'q' to quit.")
speak("Smart vision system started")

spoken_objects = { } # track last spoken time per label

try:

```

```

while True:
    ret, frame = cap.read()
    if not ret:
        print("Failed to grab frame.")
        break

    # YOLO Object Detection
    results = model(frame)

    annotated_frame = results[0].plot()

    # Process detections
    boxes = results[0].boxes
    if len(boxes) > 0:
        current_time = time.time()
        labels_to_speak = []

        # Set flag to pause distance speech
        object_detected_flag.set()

        # Sort boxes by confidence descending, get top 3
        confs = boxes.conf.cpu().numpy()
        sorted_indices = confs.argsort()[::-1][:3]

        for idx in sorted_indices:
            conf = float(confs[idx])

```



```

        if conf < 0.5: # Confidence threshold 50%
            continue
        cls_id = int(boxes.cls[idx])
        label = model.names[cls_id]

        # Speak all detected labels every frame (no delay)
        if label not in labels_to_speak:
            labels_to_speak.append(label)

    if labels_to_speak:
        speak("I see " + ", ".join(labels_to_speak))

    # Clear spoken_objects after speaking all labels
    spoken_objects.clear()

    # Clear flag to resume distance speech
    object_detected_flag.clear()

    cv2.imshow("Smart Vision", annotated_frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

except KeyboardInterrupt:
    print("Interrupted by user.")

finally:

```

```
running = False
distance_thread.join(timeout=1)
cap.release()
cv2.destroyAllWindows()
GPIO.cleanup()
speak("Smart vision system stop
ped")
print("Resources released.")
```

