FACULTY OF ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

# A “Eye”
# AI-Powered cane vision for the blind

A graduation project

Submitted by

| | |
|---|---|
| YOUCEF BOUKENNA | 220207340 |
| ABDULBARI KHABULI | 200207331 |

in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

May 2024

Program: Mechatronics Engineering

# A “Eye”

# AI-Powered cane vision for the blind

A graduation project

Submitted by

YOUCEF BOUKENNA 220207340

ABDULBARI KHABULI 200207331

submitted to the Department of Mechatronics Engineering of

OKAN UNIVERSITY

in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

Approved by:

Advisor: Assist. Prof. Dr. Sina Alp

May 2024

Program: Mechatronics Engineering

# ABSTRACT

A “Eye”

Visual impairment is affecting around 40 million people worldwide, Governments in developed countries have implemented many solutions to make everyday life for visually impaired people easier such as spreading the use of Braille code in public areas and building tactile paving in most areas just to name a few, Unfortunately this does not apply in 3rd world countries which have the majority of visually impaired people in numbers and percentage. This project is aiming to give both real-time auditory and haptic feedback for the visually impaired combining both computer vision and sensors, implementing the state of the art “YOLOv” algorithm and OpenCV library for fast and accurate object detection and providing audio feedback of the distance and the class of the object , combined with a haptic vibration alert to ensure timely collision avoidance from a vibration motor based on the measurement of an ultrasonic sensor. We believe that this design if adapted widely it should grant visually impaired people more confidence and self-reliance to their day-to-day life.

# KISA ÖZET

A"Eye"

Görme bozukluğu dünya çapında yaklaşık 40 milyon insanı etkilemektedir, Gelişmiş ülkelerdeki hükümetler, görme engelliler için günlük yaşamı kolaylaştırmak için halka açık alanlarda Braille kodunun kullanımını yaygınlaştırmak ve çoğu alanda dokunsal kaldırım inşa etmek gibi birçok çözüm uygulamıştır. Bu proje, hem bilgisayarla görmeyi hem de sensörleri birleştirerek görme engelliler için hem gerçek zamanlı işitsel hem de dokunsal geri bildirim vermeyi, hızlı ve doğru nesne algılama için son teknoloji ürünü "YOLOv" algoritmasını ve OpenCV kitaplığını uygulamayı ve mesafenin sesli geri bildirimini sağlamayı amaçlamaktadır. Bu tasarımın geniş çapta uyarlanması halinde görme engelli insanlara günlük yaşamlarında daha fazla güven ve özgüven kazandırması gerektiğine inanıyoruz.

# ACKNOWLEDGMENT

we would like to express my sincere gratitude to my supervisor, Assist.Prof.Dr.Sina Alp for his invaluable guidance, support, and patience throughout the course of this research. His expertise and insightful feedback have been helpful shaping this thesis.

we are also thankful to my committee members for their constructive criticism and valuable suggestions that have greatly improved the quality of this work.

we extend my appreciation to my colleagues for their encouragement, discussions, and assistance during the research process. Their input has been invaluable.

We are grateful to our family for their unwavering support, understanding, and encouragement throughout our academic journey. Their love and belief in us have been our source of strength.

## Contents:

# TABLES

# Figures

# ABBREVIATIONS

**AI** artifice intelligence

**YOLO** you only look once

**OpenCV** Open Computer Vision

**GB** Gigabytes

**SMS** short message service

**COCO** Common objects in context

**OCR** Optical character recognition

**mAP** Mean Average Precision

**mAR** mean Average Recall

**LIDAR** light detection and ranging

# INTRODUCTION

Visual impairment poses significant challenges to individuals' mobility and safety, with approximately 253 million people worldwide affected, according to the World Health Organization (2018). Traditional white canes offer limited assistance, prompting the need for advanced solutions that provide real-time feedback and enhance environmental perception.

we introduce a groundbreaking smart cane that integrates cutting-edge technologies to address these challenges. Our smart cane utilizes YOLOv (You Only Look Once) object detection algorithm, a camera, and an ultrasonic sensor to detect obstacles. It also features a vibrating mechanism and sound warning system to alert users about obstacles in their path.

The primary objective of this project is to design, develop, and evaluate a prototype that significantly enhances the mobility and safety of visually impaired individuals. By leveraging YOLOv and other technologies, our smart cane aims to provide a comprehensive and intuitive solution for obstacle detection and navigation.

Brambring's travel model (Brambring, 1985) as shown in Figure 1 demonstrates that locomotion, also known as mobility, is realized by perception of objects and perception of orientation. Visually impaired people use the identification of landmarks to determine their specific locations on a route for determining orientation and navigation.

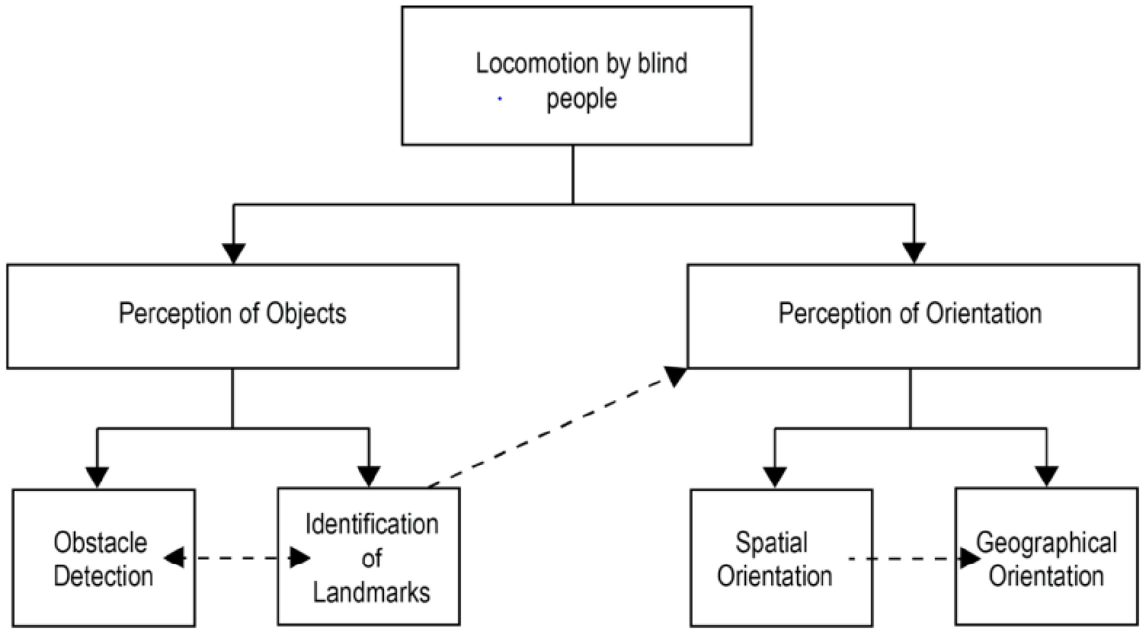


*Figure1: locomotion model*

## TERMINOLOGY

-Smart Cane

A cane equipped with advanced technologies such as sensors, cameras, and artificial intelligence algorithms to aid visually impaired individuals in detecting obstacles and navigating their environment safely.

-YOLOv (You Only Look Once)

An object detection algorithm used in the smart cane to identify objects quickly and accurately in the user's path.

-Ultrasonic Sensor

A sensor that uses sound waves to detect objects in the environment and provide feedback to the user about their proximity.

-Object Detection

The process of identifying and locating objects in images or video frames, used in the smart cane to detect obstacles.

-Navigation System

The system within the smart cane that provides guidance and direction to the user, helping them navigate their surroundings effectively.

-Haptic Feedback

Feedback provided to the user through vibrations, used in the smart cane to alert the user of obstacles or changes in their environment.

-Real-time

Refers to the ability of the smart cane to provide feedback and information instantly, as events are happening.

-Prototype

A preliminary version of the smart cane used for testing and evaluation before final production.

-Mobility

The ability of visually impaired individuals to move around independently and safely in their environment.

## PURPOSES

The goal of this project is to create a smart cane that helps blind people move around safely. We'll use object detection technology to detect obstacles and warn the user. The aim is to make it easier for visually impaired individuals to navigate their surroundings and improve their independence and safety.

## RESEARCH QUESTIONS

In order to design a technological aiding system that supports visually impaired and blind people for independent mobility, I will need to look into the following research areas:

1. Investigate the validity of previous findings on the needs of visually impaired and blind individuals in independent mobility.

2. Uncover additional requirements for supporting visually impaired and blind individuals to complete daily mobility tasks.

3. Find feasible and robust technologies for supporting independent mobility cost-effectively.

4. Evaluate conceptual designs.

## Methodology

This methodology outlines the systematic approach to develop a smart cane for visually impaired individuals, integrating advanced object detection algorithms like YOLOv5 from Ultralytics, Raspberry Pi, and 3D printing for the prototype.

The key phases include:

<u>Algorithm Selection and Training:</u> Choosing and optimizing the YOLOv5 algorithm, trained on datasets like COCO for object detection.

<u>Hardware Integration:</u> Configuring Raspberry Pi 4 Model B with necessary components like a camera module, power source, and vibrating motors.

<u>Prototyping and Fabrication:</u> Using 3D printing with Creality3D CR-10 to create a functional prototype, iterating design for user comfort.

<u>Sensor Integration and Calibration:</u> Incorporating ultrasonic sensors for environmental awareness, ensuring accuracy through calibration.

<u>Software Implementation and UI:</u> Developing software for algorithm deployment and user interface, enabling intuitive interaction.

<u>Evaluation and Validation:</u> Rigorously testing the prototype in real-world scenarios, refining based on user feedback for usability and functionality.

By following this structured methodology, the aim is to create a user-friendly and efficient smart cane, empowering visually impaired individuals with enhanced mobility and safety.

# Literature review

Enhancing navigation for visually impaired individuals has become a pivotal area of research, leveraging technological advancements to design intelligent solutions. In this literature review, we explore three notable contributions in the realm of smart blind sticks utilizing AI-driven object detection. The Vision Navigator framework, developed by Hindawi, introduces a multi-sensory approach with the Smart-fold Cane and Smart-alert Walker. This hybrid model employs the SSD-RNN obstacle recognition framework, achieving exceptional accuracies both indoors and outdoors. The model's versatility, coupled with a minimum latency delay, positions it as a well-equipped and generic framework, promising enhanced usability for the visually impaired.

In a parallel effort, the Department of Computer Science and Technology at Hohai University proposes a multi-sensory guidance system integrating ORB-SLAM and YOLO techniques. This system, implemented as a prototype cane, offers tactile and auditory advice, incorporating real-time target detection and a prompt voice system. Experiments showcase accurate real-time detection of obstacles, affirming its competency as an auxiliary aid for secure navigation.

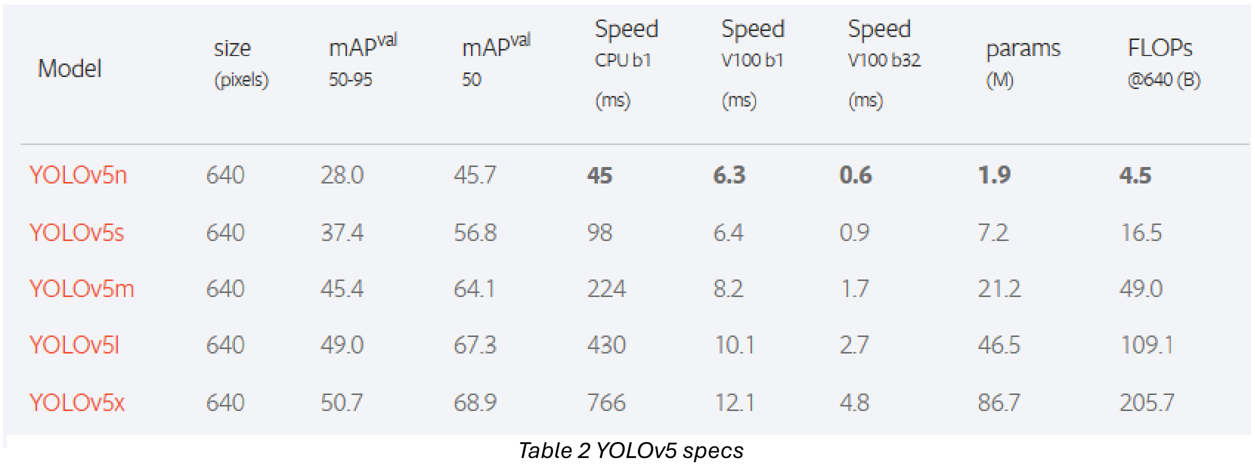
Addressing the need for robust and fast algorithms, a different perspective is presented in a study emphasizing the challenges of simultaneous robustness and speed. The paper explores alternatives to template matching, suggesting the application of machine learning techniques such as Artificial Neural Networks (ANN), Support Vector Machines (SVM), and the Wavelet Transform. These techniques, notably SIFT and SURF, demonstrate the potential to enhance feature detection in images.

In conclusion, the literature presents promising strides in smart blind stick technology. These advancements underscore the potential to significantly improve the mobility and independence of visually impaired individuals through AI-driven object detection in navigation aids.

## YOLOv

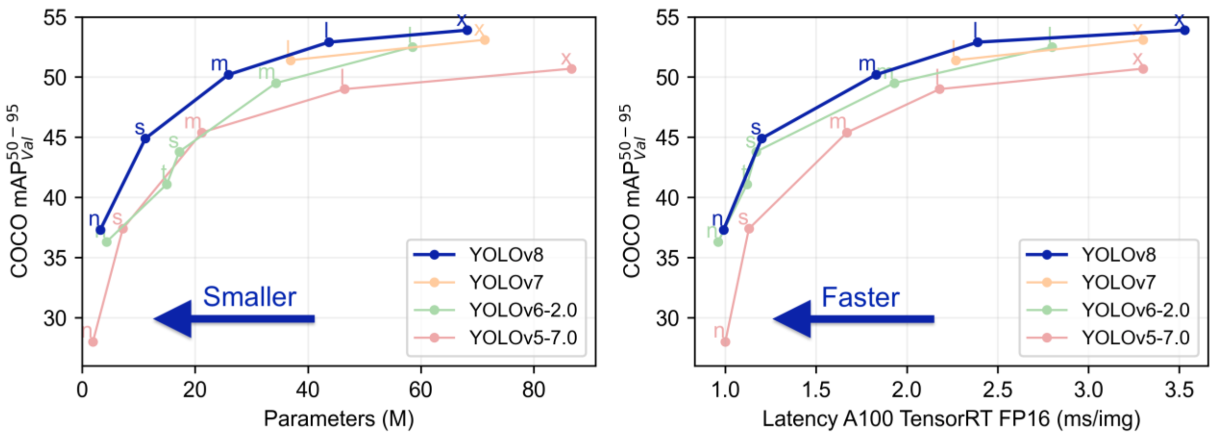
In the field of computer vision, the You Only Look Once (YOLO) algorithm has revolutionized the landscape. It offers real-time object detection with exceptional accuracy, rendering it a formidable tool for various applications including surveillance, autonomous vehicles, as well as image and video analysis. Many version of YOLOv were released through out the years in this project we are using YOLOv5 from Ultralytics, this model is in the goldilock zone between speed and accuracy while having fair requirements for computing power.

The Ultralytics YOLOv5 model represents the forefront of object detection technology, building upon the achievements of earlier YOLO versions while introducing new features and enhancements to enhance performance and versatility. YOLOv5 prioritizes speed, accuracy, and user-friendliness, making it an ideal solution for various tasks including object detection, instance segmentation, and image classification.



| Model | size (pixels) | $mAP^{val}$ 50-95 | $mAP^{val}$ 50 | Speed CPU b1 (ms) | Speed V100 b1 (ms) | Speed V100 b32 (ms) | params (M) | FLOPs @640 (B) |
|---|---|---|---|---|---|---|---|---|
| YOLOv5n | 640 | 28.0 | 45.7 | **45** | **6.3** | **0.6** | **1.9** | **4.5** |
| YOLOv5s | 640 | 37.4 | 56.8 | 98 | 6.4 | 0.9 | 7.2 | 16.5 |
| YOLOv5m | 640 | 45.4 | 64.1 | 224 | 8.2 | 1.7 | 21.2 | 49.0 |
| YOLOv5l | 640 | 49.0 | 67.3 | 430 | 10.1 | 2.7 | 46.5 | 109.1 |
| YOLOv5x | 640 | 50.7 | 68.9 | 766 | 12.1 | 4.8 | 86.7 | 205.7 |

*Table 2 YOLOv5 specs*



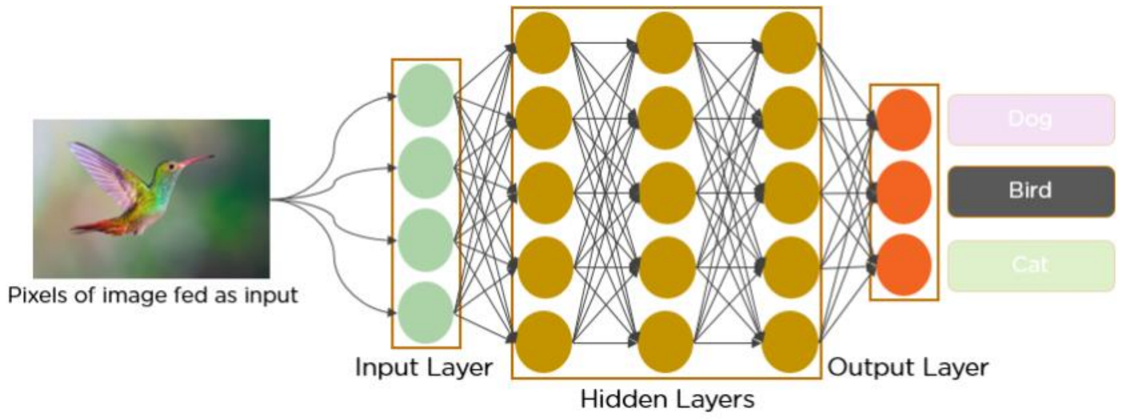
*YOLOv5 performance graphs*

## AI Model YOLOv Algorithm:

The YOLO (You Only Look Once) algorithm, initially implemented using the Darknet framework, employs a Convolutional Neural Network (CNN) to predict bounding boxes and class probabilities of objects within input images. YOLO operates by partitioning the input image into a grid of cells, wherein each cell is tasked with predicting the presence of objects, their bounding box coordinates, and respective class labels. Unlike two-stage object detection methods like R-CNN, YOLO processes the entire image in a single pass, leading to superior efficiency and speed. The algorithm has evolved through multiple iterations, including YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, and YOLOv7, each introducing refinements aimed at enhancing accuracy, processing speed, and the ability to detect smaller objects.

## <u>Convolutional Neural Networks (CNN):</u>

Convolutional Neural Networks (CNNs) are a type of deep neural network primarily used for analyzing visual data. Unlike traditional neural networks, CNNs employ convolution, a mathematical operation that modifies one function based on another. However, understanding the mathematics behind CNNs is not necessary to grasp their functionality. Essentially, CNNs reduce images into a more manageable format while retaining important features for accurate predictions.



*CNN LAYERS*

YOLOv5, introduced in 2020, integrates the EfficientDet architecture for enhanced efficiency and accuracy. Unlike prior versions, it adopts anchor-free detection, replacing anchor boxes with a single convolutional layer for bounding box prediction, ensuring adaptability to diverse object shapes and sizes. Additionally, it incorporates Cross mini-batch normalization (CmBN), a variant of batch normalization, to refine model accuracy. YOLOv5 leverages transfer learning, initially training on a large dataset and fine-tuning on a smaller one, facilitating improved generalization to new data.

## COCO Dataset:

COCO
Common Objects in Context

*COCO LOGO*

The COCO (Common Objects in Context) dataset serves as a cornerstone in computer vision research, specifically tailored for object detection, segmentation, and captioning tasks. It stands as a pivotal benchmarking resource, facilitating exploration into diverse object categories.

Key Features:

Boasting a vast repository of over 330K images, COCO includes annotations for 200K images, spanning object detection, segmentation, and captioning. Encompassing 80 object categories, ranging from commonplace entities like automobiles and fauna to specialized items such as parasols and athletic gear, it provides annotations inclusive of object bounding boxes, segmentation masks, and textual descriptors. Standardized evaluation metrics such as mean Average Precision (mAP) and mean Average Recall (mAR) ensure consistent model assessment across tasks.

Dataset Structure:

COCO's architecture is stratified into three distinct subsets:

- Train2017: Constituting 118K images, this segment serves as the training corpus for model development.

- Val2017: With a contingent of 5K images, this subset operates as the validation set during model training.

- Test2017: Comprising 20K images, this division is designated for model benchmarking. Devoid of publicly accessible ground truth annotations, model performance is assessed through submissions to the COCO evaluation server.

Applications:

The COCO dataset finds extensive utility in training and evaluating a spectrum of deep learning models across manifold applications, including object detection (e.g., YOLO, Faster R-CNN, SSD), instance segmentation (e.g., Mask R-CNN), and key point detection (e.g., OpenPose). Its comprehensive repertoire of object categories, exhaustive annotations, and standardized evaluation metrics solidify its status as an indispensable asset within the domain of computer vision research and application.

# Prototype

## Hardware & Software

### RASPBERRY PI

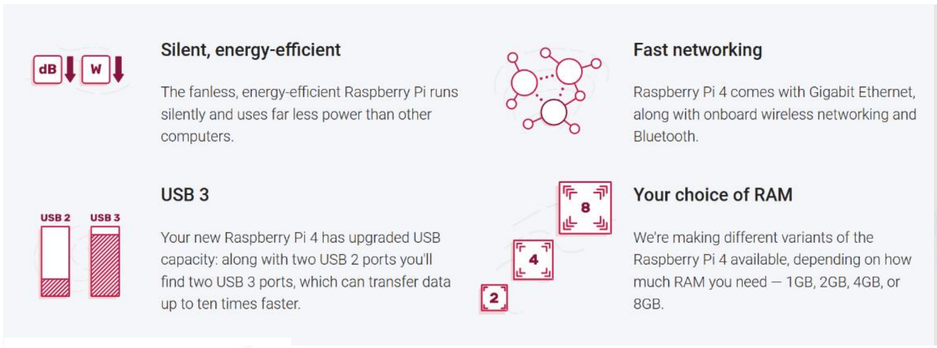
*raspberry pi 4*

The **Raspberry Pi** is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It does not include peripherals (such as keyboards and mice) or cases.

However, some accessories have been included in several official and unofficial bundles.

The Raspberry Pi is a credit-card-sized computer that plugs into your TV and a keyboard. It is a capable little device that enables people of all ages to explore computing and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

Furthermore, In the prototype of this project a Raspberry pi 4 model B 4GB ram is being utilized with a cooling case and a camera module and a 16 GB memory card for storage alongside a power source, either two batteries or a power bank.



**Silent, energy-efficient**

The fanless, energy-efficient Raspberry Pi runs silently and uses far less power than other computers.

**Fast networking**

Raspberry Pi 4 comes with Gigabit Ethernet, along with onboard wireless networking and Bluetooth.

**USB 3**

Your new Raspberry Pi 4 has upgraded USB capacity: along with two USB 2 ports you'll find two USB 3 ports, which can transfer data up to ten times faster.

**Your choice of RAM**

We're making different variants of the Raspberry Pi 4 available, depending on how much RAM you need — 1GB, 2GB, 4GB, or 8GB.

*Rasbperry pi 4 properties.*

*raspberry pi cooling case.*

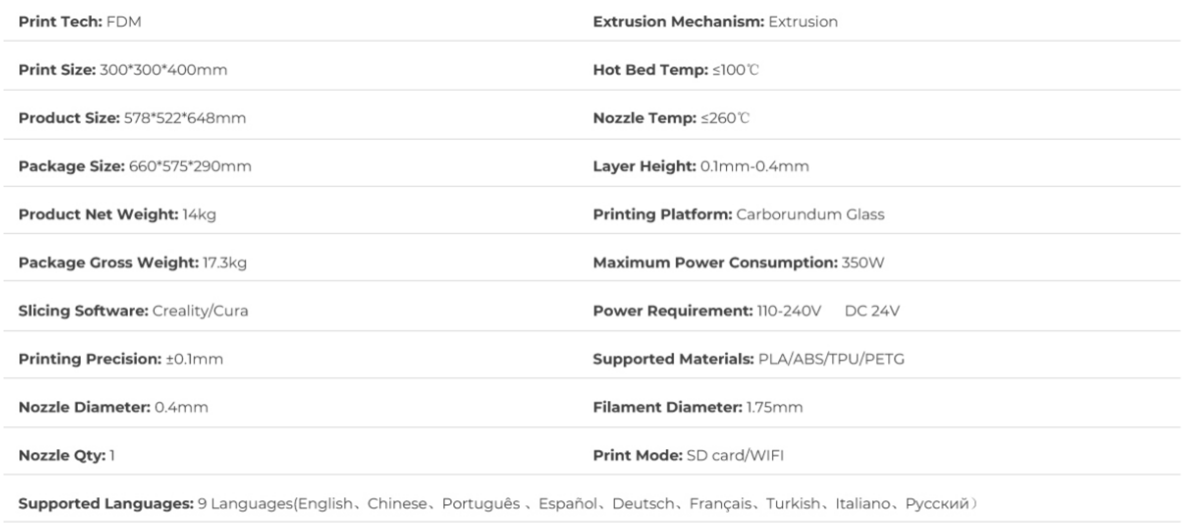
*raspberry pi camera module.*

## 3D printer

3D Printing is perhaps one of the most exciting advancements in technology that a lot of people are now interested in learning.

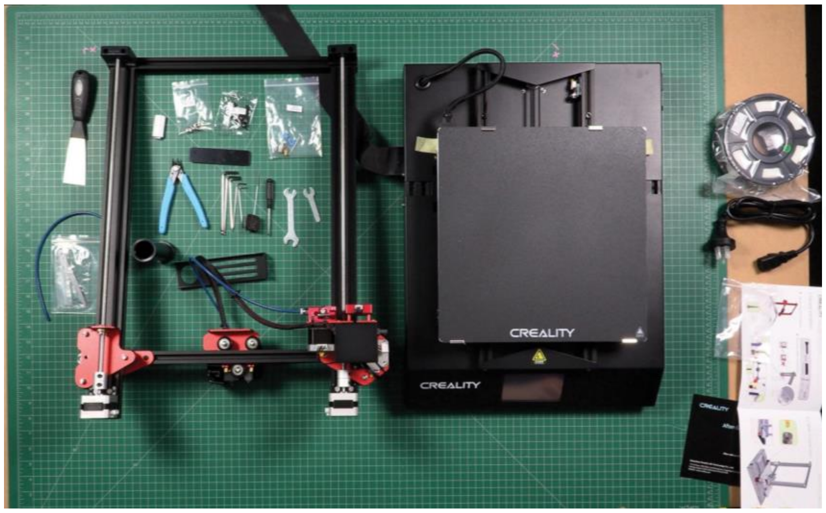
For the prototype we are using the Creality3D CR-10 printer.

The Creality3D CR-10 is an FDM 3D printer that made a significant impact on the 3D printing community. Quickly rising to popularity, it became one of the most sought-after mid-range printers, alongside Anet's A8 and A6



| | |
|---|---|
| **Print Tech:** FDM | **Extrusion Mechanism:** Extrusion |
| **Print Size:** 300*300*400mm | **Hot Bed Temp:** ≤100℃ |
| **Product Size:** 578*522*648mm | **Nozzle Temp:** ≤260℃ |
| **Package Size:** 660*575*290mm | **Layer Height:** 0.1mm-0.4mm |
| **Product Net Weight:** 14kg | **Printing Platform:** Carborundum Glass |
| **Package Gross Weight:** 17.3kg | **Maximum Power Consumption:** 350W |
| **Slicing Software:** Creality/Cura | **Power Requirement:** 110-240V DC 24V |
| **Printing Precision:** ±0.1mm | **Supported Materials:** PLA/ABS/TPU/PETG |
| **Nozzle Diameter:** 0.4mm | **Filament Diameter:** 1.75mm |
| **Nozzle Qty:** 1 | **Print Mode:** SD card/WIFI |
| **Supported Languages:** 9 Languages(English、Chinese、Português 、Español、Deutsch、Français、Turkish、Italiano、Русский) | |

*Table 5 CR-10 printer specs*



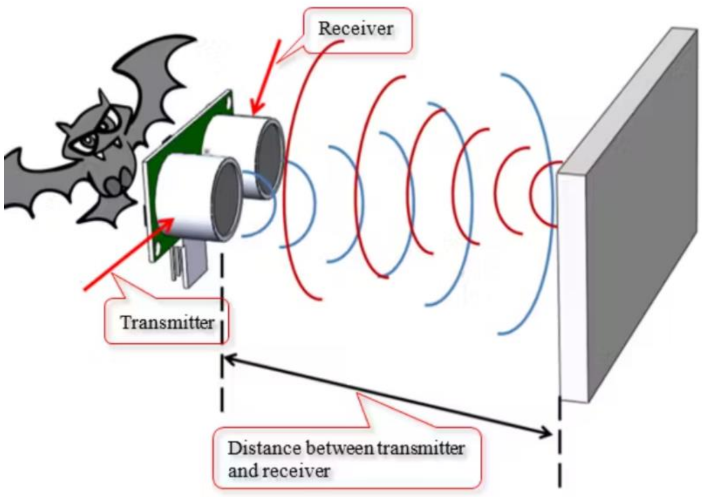
*CR-10 printer parts*

*CR-10 printer*

## Ultrasonic sensor

The Ultrasonic Sensor is a cost-effective proximity and distance sensor widely employed for object avoidance in robotics projects. Its versatility extends to applications such as turret control, water level sensing, and even parking assistance. Power source

It operates by emitting sound waves at a frequency beyond human hearing. The sensor's transducer serves as both a transmitter and receiver of these ultrasonic signals. Like other ultrasonic sensors, ours utilizes a single transducer to emit a pulse and detect the echo. By measuring the time interval between transmission and reception, the sensor calculates the distance to the target.



*ultrasonic sensor working principle*

*ultrasonic sensor*

## Mini vibrating motor

That's your little buzzing motor, and for any haptic feedback project you'll want to pick up a few of them. These vibe motors are tiny discs, completely sealed up so they're easy to use and embed.

Two wires are used to control/power the vibe. Simply provide power from a battery or microcontroller pin (red is positive, blue is negative) and it will buzz away. The rated voltage is 2.5 to 3.8V and for many projects, we found it vibrates from 2V up to 5V, higher voltages result in more current draw but also a stronger vibration.

*Vibrating motor*

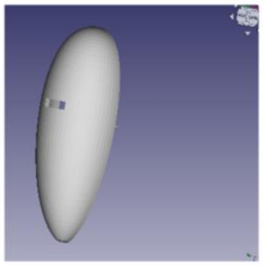
Technical Details

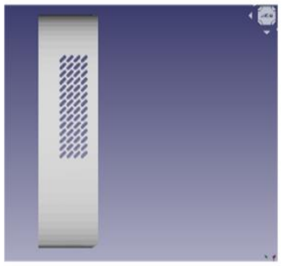
Dimension: 10mm diameter, 2.7mm thick

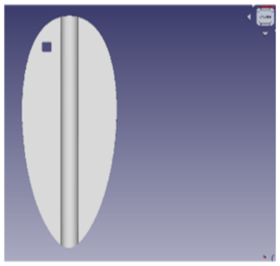
Voltage: 2V - 5V/5V current draw: 100mA, 4V current draw: 80mA, 3V current draw: 60mA, 2V current draw: 40mA/11000 RPM at 5V/Weight: 0.9 gram

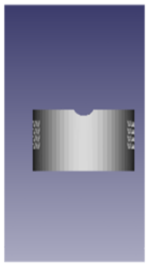
# Prototype Visual representation

## 3D Design

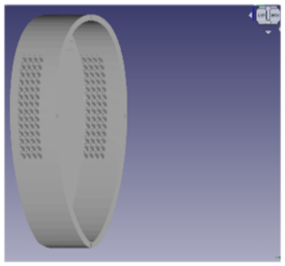

*Front view*

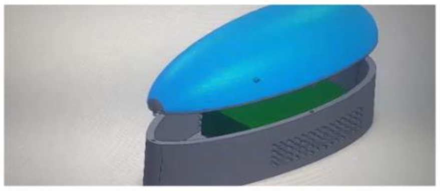

*Side view*


*Back view*


*Top view*

*view1*

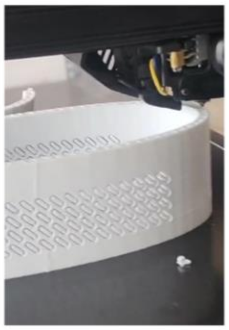

*view 2*


*3-D view*

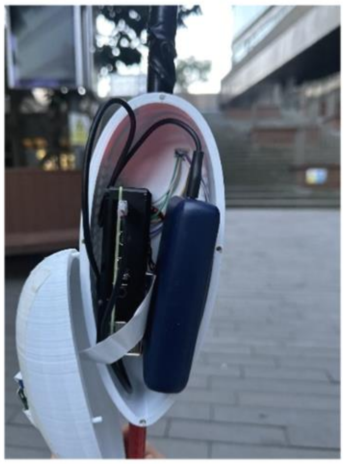
## 3D-printing the casing

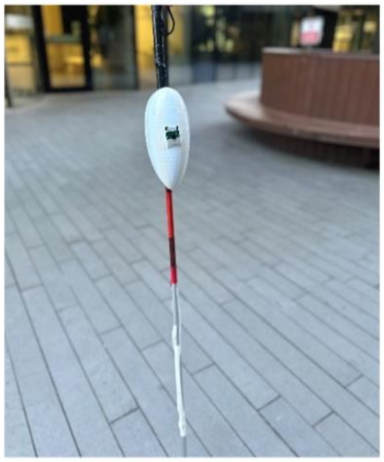
*printing stage*

*printing stage 2*


*printing stage 3*

## Finale assembly

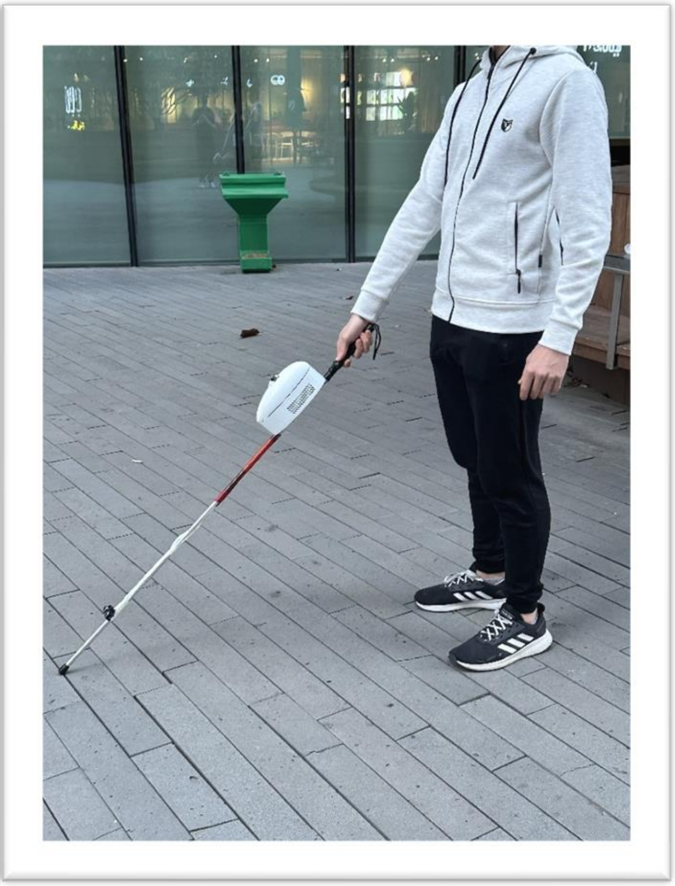

*Tear down*

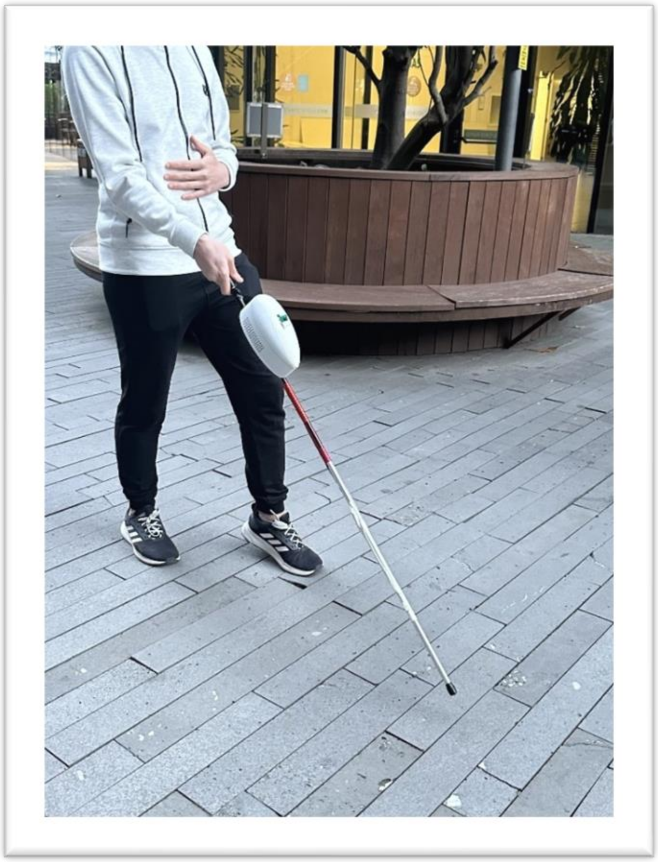

*Complete view*

*case view 1*

*case view 2*

Real life test 1


Real life test 2


Real life test 3

# Future Scope

In future implementations, we hope to add many features we see very effective in achieving the essential purpose of this project and improve life quality alongside it:

-panic button pressing which will notify the relatives of the person about the GPS co-ordinates via SMS message

-train the object detection algorithm to detect other objects that are not present on the COCO dataset the current model is trained on, objects that will improve the quality of life of the user such as, Food recognition, currency identification, face recognition, color identifier and OCR (text recognition).

- make a smaller version for kids and have a live video feed transmission for parents.

- make the design more compact and shrink it enough to fit in a glasses format to be used indoor, glasses format will give the camera more stability and enabling the user to benefit from the same features without having to hold a cane on one hand so the user will have both his hands for other tasks.

- add an AI voice recognition assistant such as SIRI or ALEXA to enable the user to ask about directions, weather situation, time, make phone calls, get info about transportation and the battery life left in the cane.

- train the model on night vision footage so the cane could be used at low light situations as well.

-add a LIDAR instead of the ultrasonic sensor for more accurate distance sensing.

-add a camera with wide angle so it detects the environment on a wider scale.

-train the model to understand and turn sign language into voice output so the visually impaired person could communicate with a deaf person if needed.

- add multiple languages to the voice output so that non-English speakers could use it.

- Ensuring user privacy and compliance with data protection regulations.

-make it detect when an accident happens, it should notify relatives through SMS or voice message and include medical ID and make the cane say it to the specialized authorities.

- pair it with a phone app that stores stats and improve overall experience.

# Conclusion

This project showed us the potential of current object detection algorithms including the YOLOv, the one used in this project, these algorithms are  sophisticated enough to be used for a general purpose such as detecting the common objects in everyday life , all this while being easy to learn and easy to use and deploy from any one with a decent literacy in technology, we were quite surprised that such and algorithm could run on a device the size of a credit card with decent performance even though the raspberry pi is not very powerful in terms of computation power, so this operation could be conducted on a smaller form factor design in the future and could be way more powerful and useful.

We also saw the simple audio feedback which is text to speech in python how it seemed to be quite smart, so if more work is put on it, it could become an AI assistant for the visually impaired not just an alerting or navigation assistance system, but also enhance their quality of life and give them more independency and self-confidence.

# REFERENCES

Websites:

1. Creality. (n.d.). Creality CR-10 Smart 3D Printer. Retrieved from https://www.creality.com/products/creality-cr-10-smart-3d-printer
2. Pevly. (n.d.). CR-10 Review: 3D Print Like a Pro with This Printer. Retrieved from https://pevly.com/cr-10-review/
3. Raspberry Pi Foundation. (n.d.). Raspberry Pi. Retrieved from https://www.raspberrypi.com/
4. PyTorch. (n.d.). Ultralytics YOLOv5. Retrieved from https://pytorch.org/hub/ultralytics_yolov5/
5. Ultralytics. (n.d.). YOLOv5 Documentation. Retrieved from https://docs.ultralytics.com/
6. COCO Dataset. (n.d.). Retrieved from https://cocodataset.org/#home
7. Kili Technology. (n.d.). YOLO Algorithm: Real-Time Object Detection from A to Z. Retrieved from https://kili-technology.com/data-labeling/machine-learning/yolo-algorithm-real-time-object-detection-from-a-to-z
8. Analytics Vidhya. (2021, May). Convolutional Neural Networks (CNN). Retrieved from https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
9. Papers with Code. (n.d.). COCO Dataset. Retrieved from https://paperswithcode.com/dataset/coco
10. GitHub. (n.d.). GitHub Copilot. Retrieved from

    https://github.com/features/copilot

articles:

1- F. Scioscia, S. Suman, S. Mishra, K. S. Sahoo, and A. Nayyar, "Vision Navigator: A Smart and Intelligent Obstacle Recognition Model for Visually Impaired Users," Mobile Information Systems, vol. 2022, Article ID 9715891, Mar. 11, 2022. DOI: 10.1155/2022/9715891.

2- Fernandes, H., Costa, P., Paredes, H., Filipe, V., Barroso, J. (2014). Integrating Computer Vision Object Recognition with Location Based Services for the Blind. In: Stephanidis, C., Antona, M. (eds) Universal Access in Human-Computer Interaction. Aging and Assistive Environments. UAHCI 2014. Lecture Notes in Computer Science, vol 8515. Springer, Cham. https://doi.org/10.1007/978-3-319-07446-7_48

3- Xie, Z. Li, Y. Zhang, J. Zhang, F. Liu, and W. Chen, "A Multi-Sensory Guidance System for the Visually Impaired Using YOLO and ORB-SLAM," Information, vol. 13, no. 7, p. 343, 2022. DOI: 10.3390/info13070343.

# APPENDIX

```
import RPi.GPIO as GPIO
import time
import cv2
import numpy as np
import pyttsx3
from imutils.video import VideoStream
import threading
import speech_recognition as sr
# GPIO Mode (BOARD / BCM)
GPIO.setmode(GPIO.BCM)
# Set GPIO Pins
GPIO_TRIGGER = 18
GPIO_ECHO = 24
BUZZER_PIN = 20
# Set GPIO direction (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)
GPIO.setup(BUZZER_PIN, GPIO.OUT)
# Initialize PWM
buzzer_pwm = GPIO.PWM(BUZZER_PIN, 100)
# Initialize the video stream
frameSize = (320, 240)
vs = VideoStream(src=0, resolution=frameSize, framerate=32).start()
time.sleep(2.0)

# Load the COCO class labels our YOLO model was trained on
```

```
LABELS = open("/home/grad_project/Desktop/GitHub/Realtime-ObjectDetection-Direction-for-Blind-Raspberry-Pi/model2-files/coco.names").read().strip().split("\n")

# Load YOLOv5 model
print("[INFO] loading YOLOv5 from disk...")
net = cv2.dnn.readNet(r"/home/grad_project/Desktop/GitHub/Realtime-ObjectDetection-Direction-for-Blind-Raspberry-Pi/model2-files/yolov5s.torchscript.pt")

# Initialize text-to-speech engine
engine = pyttsx3.init()

# Function to speak text
def speak(voice_prompt):
    if not engine._inLoop:
        engine.say(voice_prompt)
        engine.runAndWait()

# Function to calculate distance
def calculate_distance(known_width, focal_length, pixel_width):
    return (known_width * focal_length) / pixel_width

# Function to measure distance with ultrasonic sensor
def distance():
    # Measurement logic
    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)
    StartTime = time.time()
    StopTime = time.time()
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()
    TimeElapsed = StopTime - StartTime
    distance = (TimeElapsed * 34300) / 2
    return distance
# Function to control buzzer
def control_buzzer():
    try:
        while True:
            distance_val = distance()
            distance_val = min(200, distance_val)  # Limit distance to 200 cm
            distance_val = max(2, distance_val)  # Limit distance to 2 cm
            if distance_val < 50:
                duty_cycle = int((50 - distance_val) / 50 * 100)
                buzzer_pwm.start(duty_cycle)
            else:
                buzzer_pwm.stop()
            time.sleep(0.5)
    except KeyboardInterrupt:
        print("Buzzer control stopped by User")

# Function to recognize speech
def recognize_speech():
    recognizer = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        recognizer.adjust_for_ambient_noise(source)
        audio = recognizer.listen(source)
    try:
        print("Recognizing...")
        command = recognizer.recognize_google(audio)
        print("You said:", command)
        return command.lower()
    except sr.UnknownValueError:
        print("Could not understand audio")
        return None
    except sr.RequestError as e:
        print("Could not request results; {0}".format(e))
        return None
# Function to check if object is detected
def check_object(object_name, boxes, classIDs):
    for i in range(len(boxes)):
        if LABELS[classIDs[i]] == object_name:
            return True
    return False

# Function to detect object
def detect_object(object_name, boxes, classIDs):
    if check_object(object_name, boxes, classIDs):
        speak(f"Yes, there is a {object_name} in front of you.")
    else:
        speak(f"No, there is no {object_name} in front of you.")
# Main function
def main():
    threading.Thread(target=control_buzzer).start()
    detect_flag = False
    try:
        while True:
            if not detect_flag:
                command = recognize_speech()
                if command and "detect" in command:
                    detect_flag = True
            else:
                frame = vs.read()
                (H, W) = frame.shape[:2]
                blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416), swapRB=True,
crop=False)
                net.setInput(blob)
                outputs = net.forward()
                boxes = []
                confidences = []
                classIDs = []
                for output in outputs:
                    for detection in output:
                        scores = detection[5:]
                        classID = np.argmax(scores)
                        confidence = scores[classID]
                        if confidence > 0.3:
```

```
                    box = detection[0:4] * np.array([W, H, W, H])
                    (centerX, centerY, width, height) = box.astype("int")
                    x = int(centerX - (width / 2))
                    y = int(centerY - (height / 2))
                    boxes.append([x, y, int(width), int(height)])
                    confidences.append(float(confidence))
                    classIDs.append(classID)
        idxs = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)
        voice_prompt = ""
        if len(idxs) > 0:
            for i in idxs.flatten():
                centerX = boxes[i][0] + boxes[i][2] / 2
                object_name = LABELS[classIDs[i]]
                pixel_width = boxes[i][2]
                distance_val = distance()
                distance_val = min(200, distance_val)  # Limit distance to 200 cm
                distance_val = max(2, distance_val)  # Limit distance to 2 cm
                distance_val = round(distance_val, 2)
                if centerX <= W / 3:
                    voice_prompt = f"{object_name} to your left. {distance_val} centimeters away
. "
                elif centerX <= (W / 3 * 2):
                    voice_prompt = f"{object_name} in front of you . {distance_val} centimeters
away . "
                else:
                    voice_prompt = f"{object_name} to your right. {distance_val} centimeters
away . "
                if object_name in ["cup", "fork", "umbrella"]:
                     voice_prompt += f"Don't worry about it."
                    (x, y, w, h) = boxes[i]
                    color = (0, 255, 0)
                    cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
                    text = f"{object_name}: {confidences[i]:.2f}"
                    cv2.putText(frame, text, (x, y - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color,
                    cv2.putText(frame, f"Distance: {distance_val} cm", (x, y - 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
                cv2.imshow("video", frame)
                if voice_prompt:
                    threading.Thread(target=speak, args=(voice_prompt,)).start()
                key = cv2.waitKey(1) & 0xFF
                if key == ord("q"):
                    break
                command = recognize_speech()
                if command:
                    if "how are you" in command:
                        speak("I'm fine, thank you. And you?")
                        response = recognize_speech()
                        if response:
                            if "fine" in response or "good" in response:
                                speak("That's great!")
                            else:
                                speak("I hope you feel better soon.")
                    elif any(obj in command for obj in LABELS):
                        object_name = next((obj for obj in LABELS if obj in command), None)
                        detect_object(object_name, boxes, classIDs)
except KeyboardInterrupt:
        print("Measurement stopped by User")
    finally:
        GPIO.cleanup()
        cv2.destroyAllWindows()
if __name__ == "__main__":
    main()
```