KLE Technological University
Creating Value, Leveraging Knowledge
Dr. M. S. Sheshgiri Campus, Belagavi

Department of Electronics and Communication Engineering

Report on Minor Project

Smart Object Detection Using Multimodal Data for Assistance SYSTEM

By:

1. **Abubakarsiddik Desai** USN: 02FE22BEC002
2. **Bhaskar Kamble** USN: 02FE22BEC017
3. **Mayuri Hegade** USN: 02FE22BEC037
4. **Rakshita Mathapati** USN: 02FE22BEC065

Semester: VI, 2024-2025

Under the guidance of
Pooja Mahajan

KLE Technological University
Creating Value, Leveraging Knowledge

Dr. M. S. Sheshgiri Campus, Belagavi

2024-2025

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled **"Smart Object Detection Using Multimodal Data for Assistance"** is a bonafide work carried out by the student team of " **Abubakarsiddik Desai (02FE22BEC002), Bhaskar Kamble (02FE22BEC017), Mayuri Hegade (02FE22BEC037), Rakshita Mathapati (02FE22BEC065)"**. The project report has been approved as it satisfies the requirements with respect to the Minor Project work prescribed by the university curriculum for B.E. (VI Semester) in Department of Electronics and Communication Engineering of KLE Technological University Dr. M. S. Sheshgiri CET Belagavi campus for the academic year 2024-2025.

Pooja Mahajan	**Dr. Dattaprasad A. Torse**	**Dr. S. F. Patil**
Guide	**Head of Department**	**Principal**

Name of Examiners **Signature with date**

1.

2.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to the following individuals and organizations who have played a significant role in the development of the Smart Object Detection Using Multimodal Data for Assistance SYSTEM.

We would like to express our sincere gratitude to our guide, Prof.Pooja Mahajan, whose invaluable guidance and unwavering support have been instrumental throughout the entirety of our project.

We take the opportunity to thank our project coordinator Prof.Ashwini Desai, for providing us motivation and encouragement.

We are grateful to our Head of the Department Dr. Dattaprasad Torse, for granting us the privilege to work on this project and for his valuable support throughout.

We take the opportunity to thank our Principal Dr. S. F. Patil, for providing us the opportunity to undergo the project.

We acknowledge the contributions of all the team members who participated in this project. Each team member brought unique skills and perspectives, contributing to the development of different elements of the project.

We extend our thanks to KLE Dr. MSSCET for providing the necessary resources and infrastructure to carry out this project. Their support was vital in enabling us to conduct the required research, acquire relevant knowledge, and access computing resources.

We take this opportunity to thank all the teaching and non-teaching staff of Electronics and Communication Department for their cooperation and suggestions during the project.

-The project team

ABSTRACT

Object detection and distance estimation are crucial for various applications, including surveillance, automation, and embedded systems. However, traditional object detection systems often require high computational power, making them unsuitable for low-resource devices. This project aims to implement an efficient object detection system using multimodal data on a Raspberry Pi.The system is designed to detect objects and estimate their distances, displaying the results on an OLED screen.By Fine-tuning the YOLO algorithm for Raspberry Pi, this project ensures real-time object detection and distance estimation while maintaining low computational overhead. This enables the system to operate independently, without reliance on cloud services, offering advantages such as lower latency and enhanced data privacy.This approach is particularly useful for smart, portable assistive devices, demonstrating the potential to deploy intelligent and cost-effective solutions on resource-constrained hardware.

Contents

List of Figures

1 Motivation

Object detection enable machines to understand and respond to their surroundings. However, most modern deep learning models require powerful hardware and constant internet access, making them unsuitable for small or remote devices. This project demonstrates that a streamlined YOLO-based model can run on a Raspberry Pi to detect and segment objects, estimate their distance, and display the results on an OLED screen in real time. By fitting all processing onto a compact, low-cost platform, we make intelligent vision systems accessible without expensive GPUs or cloud servers. Such a solution can support applications like simple security cameras, mobile robots, and assistive devices in areas with limited power or connectivity.

2 Objectives

- **On-Device Multimodal Object Detection:** Deploy a lightweight YOLO model on the Raspberry Pi to perform real-time detection using multiple input modalities (e.g., RGB camera + depth data), all processed locally without cloud reliance.
- **Instant Display of Results:** Show detected objects along with their distances on an attached OLED screen for quick and clear visual feedback.
- **Privacy and Low Latency:** Process all sensory data entirely on-device to minimize delay and protect user privacy by avoiding data transfer to external servers.
- **Optimized for Limited Resources:** Use model compression methods such as pruning and quantization to optimize the YOLO network, ensuring smooth real-time operation on the Raspberry Pi's constrained hardware.
- **IoT Integration for Remote Monitoring:** Enable the Raspberry Pi to send detection and distance information over a network to IoT platforms, facilitating scalable monitoring and automated control systems.
- **Affordable and Accessible Hardware:** Utilize low-cost, off-the-shelf components (Raspberry Pi, compatible camera, OLED display) to create an accessible solution for advanced multimodal object detection and distance measurement.

3 Literature Survey

3.1 Object Detection Learning for Intelligent Self Automated Vehicles

The paper provides a comparative evaluation of three popular object detection algorithms—YOLO, Faster R-CNN, and R-CNN—highlighting their strengths and weaknesses in terms of performance metrics like mAP (mean Average Precision) and FPS (Frames Per Second). It also introduces an intelligent system for real-time object detection and tracking, utilizing an pi-camera module to capture input images. This system applies machine learning algorithms to identify and localize objects, with the added capability of tracking and following moving objects by calculating their position coordinates. The study emphasizes the importance of object detection in modern technology, showcasing how it can be integrated into robotic systems to automate tasks and enhance awareness of visual data in diverse applications.

3.2 Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges

This comprehensive survey explores deep learning methods for multi-modal object detection and semantic segmentation in autonomous driving. It discusses the integration of various sensors like cameras, LiDAR, and radar, emphasizing the complementary nature of these modalities. The paper categorizes fusion strategies into early, late, and hybrid fusion, analyzing their effectiveness and challenges. It also reviews available datasets and highlights open research questions in the field.

3.3 3D Object Detection via Modality Interaction

This paper introduces "DeepInteraction," a novel framework for 3D object detection that emphasizes modality interaction. Unlike traditional fusion methods, it maintains separate representations for each modality (e.g., camera and LiDAR) throughout the network, allowing for more effective exploitation of their unique characteristics. The approach includes a multi-modal representational interaction encoder and a predictive interaction decoder. Evaluated on the nuScenes dataset, DeepInteraction achieved state-of-the-art performance, ranking first on the nuScenes object detection leaderboard.

3.4 Multi-modal Object Detection via Transformer Network

The paper exploresThis paper presents a transformer-based approach for multi-modal object detection, focusing on the fusion of intra-modal and inter-modal features. The method leverages the complementary information from different modalities to improve detection performance. Additionally, a contrastive loss function is employed to enhance the utilization of label information during training. Extensive experiments on various datasets validate the effectiveness of the proposed approach

3.5 Object Recognition Using Automotive Ultrasonic Sensor and Raspberry Pi

The paper focuses on object recognition by combining ultrasonic sensing with machine learning and deep learning techniques. Object recognition using automotive ultrasonic sensors and Raspberry Pi involves integrating these sensors to detect and identify objects in the environment. The Raspberry Pi processes the data collected from the ultrasonic sensors, which measure distances by emitting sound waves and analyzing their echoes. This setup can be enhanced with machine learning algorithms to improve recognition accuracy. By training the system on various object types, it can distinguish between them based on distance and size. The project typically includes coding in Python and utilizing libraries like OpenCV for image processing. This combination allows for real-time object detection and tracking, making it suitable for applications in robotics and autonomous vehicles. Overall, it provides a cost-effective solution for developing smart sensing systems.

3.6 What is YOLOv5: A Deep Look into the Internal Features of the Popular Object Detector

This comprehensive study delves into the architecture and functionalities of the YOLOv5 object detection model. Key components such as the Cross Stage Partial (CSP) backbone and

Path Aggregation Network (PANet) are explored in detail, highlighting their roles in enhancing feature extraction and fusion. The paper also discusses the transition from the Darknet framework to PyTorch, emphasizing improvements in development efficiency and deployment capabilities. Performance evaluations across various datasets and hardware platforms are presented, demonstrating YOLOv5's balance between speed and accuracy. Additionally, the study examines training methodologies, including data augmentation techniques and loss function optimizations, providing insights into the model's robustness and generalization capabilities.

3.7 Real-Time Multiple Object Detection Using Raspberry Pi and Tiny-ML Approach

The study presents an enhanced Single Shot MultiBox Detector (SSD) approach tailored for deployment on resource-constrained devices like the Raspberry Pi. By leveraging TinyML techniques, the model is optimized to reduce computational complexity and memory usage while maintaining detection accuracy. Key optimizations include model pruning, quantization, and efficient architecture design, enabling real-time object detection on low-power hardware. The approach addresses challenges such as limited processing power and energy efficiency, which are critical for embedded applications. Experimental results demonstrate that the optimized SSD model achieves a good balance between speed and accuracy on the Raspberry Pi. This work highlights the potential of TinyML to bring advanced AI capabilities to edge devices with strict resource limits.

3.8 Object Detection Learning for Intelligent Self Automated Vehicles

Ahtsham Alam et al. (2022) presents a self-driving car prototype that integrates artificial intelligence, image processing, and robotics. Utilizing ultrasonic and camera sensors, the vehicle autonomously detects and recognizes obstacles, calculates their distance, and makes real-time decisions to avoid collisions. The system also features GPS-based tracking and Bluetooth voice control for enhanced user interaction. Implemented on a Raspberry Pi and Arduino UNO, the design emphasizes cost-effectiveness and scalability for smart city applications. This research contributes to the development of intelligent, crash-avoidant vehicles suitable for urban environments.

3.9 An Autonomous Vehicle using Raspberry Pi and Ultrasonic Sensors

An autonomous vehicle using Raspberry Pi and ultrasonic sensors combines affordable hardware with real-time obstacle detection capabilities. The Raspberry Pi acts as the central processing unit, running control algorithms to navigate the vehicle. Ultrasonic sensors measure the distance to nearby objects by emitting sound waves and calculating their echo return time. This data helps the vehicle detect obstacles and avoid collisions effectively. The system processes sensor input to control motor speed and steering direction dynamically. It supports real-time decision-making for smooth and safe autonomous navigation. This setup is ideal for beginners aiming to understand self-driving technology fundamentals. The use of open-source software enhances flexibility and customization in autonomous vehicle projects.

3.10 Real-Time Object Detection and Tracking for Autonomous Vehicles

Real-time object detection and tracking for autonomous vehicles is crucial for safe navigation and decision-making. This involves utilizing advanced algorithms to identify and track various objects in the vehicle's environment. Techniques like YOLO and sensor fusion are commonly employed to improve detection accuracy. The integration of 3D object detection enhances spatial awareness, aiding in path planning. Additionally, adaptive systems are developed to function effectively under varying conditions, such as nighttime or adverse weather. The tracking-by-detection paradigm is often used, leveraging bounding boxes and class information. Overall, these technologies are essential for the reliability and safety of autonomous driving systems. Continuous research aims to refine these methods for better performance in real-world scenarios.

4 Problem Statement

Develop a multimodal system using Raspberry Pi to detect vehicles and estimate their distance in real-time. Combine Raspberrypi camera input with ultrasonic sensor to enhance detection accuracy and spatial awareness.

5 Application In Societal Context

1. Smart Home Automation: A compact vision module installed in homes identifies and segments people, pets, and objects (e.g., appliances or packages) in real time. It can trigger custom actions—like turning lights on when someone enters a room or notifying you when a pet approaches a restricted area—while keeping all video data on-device for privacy.
2. Precision Agriculture:Deployed across greenhouses or open fields, the system detects and segments crops, weeds, and pests. Distance estimation helps an autonomous weeding robot navigate rows accurately and apply targeted treatments only where needed, reducing chemical usage and improving yield.
3. Assistive Robotics for Elderly Care: A wearable or wheelchair-mounted camera identifies and segments obstacles, furniture, and people. The system estimates distances to these objects and provides real-time audio or haptic alerts to help users navigate safely indoors, reducing fall risk and enhancing independence.
4. Traffic Monitoring and Management: Installed at intersections, the system segments vehicles, cyclists, and pedestrians, estimating their distances from the camera. Real-time data can feed into edge-based traffic controllers to adjust signal timings dynamically, improving flow and safety without sending video to the cloud.

Project planning

Project planning is critical for ensuring project success by clearly defining objectives, identifying tasks, and setting timelines. It enables efficient resource allocation, risk management, and helps to maintain focus and alignment among team members. Additionally, it enhances communication, allowing for smooth coordination, and ensures that milestones are met within budget and scope.

6 Gantt Chart

A Gantt chart is a visual project management tool that represents the timeline of a project. It displays tasks or activities as horizontal bars along a timeline, with the length of each bar indicating the duration of a task. The Gantt chart helps project managers track progress, manage deadlines, and visualize how tasks overlap or depend on one another. It is widely used for scheduling, resource allocation, and ensuring timely completion of projects. The timeline or Gantt chart for this project is given in below Figure 2.2. Gantt chart outlines the project phases for developing a smart object detection system. Phase 1 focuses on understanding the problem and conducting a literature survey. Subsequent phases include data selection, block diagram development, and system demonstration. The final phase covers optimization, report writing in LaTeX, and project presentation by May.

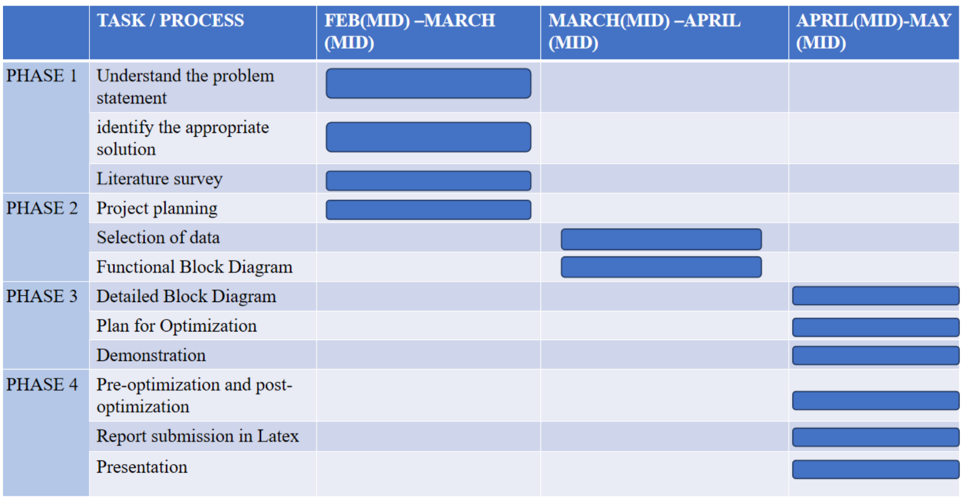

Figure 1: Gantt Chart For Smart Object Detection

7 Work Breakdown Structure(WBS)

A Work Breakdown Structure (WBS) is a project management tool that divides a project into smaller, manageable components or tasks. It organizes the work into a hierarchy, making it easier to plan, assign resources, and track progress. The WBS helps clarify the scope of the project, ensuring that all required work is identified. It also enhances communication among team members and stakeholders, assigns responsibilities, and improves time and resource management. By breaking down complex projects into manageable parts, it makes it easier to estimate costs, schedule timelines, and monitor progress. The WBS for this object detection system is shown in Figure 2.2.

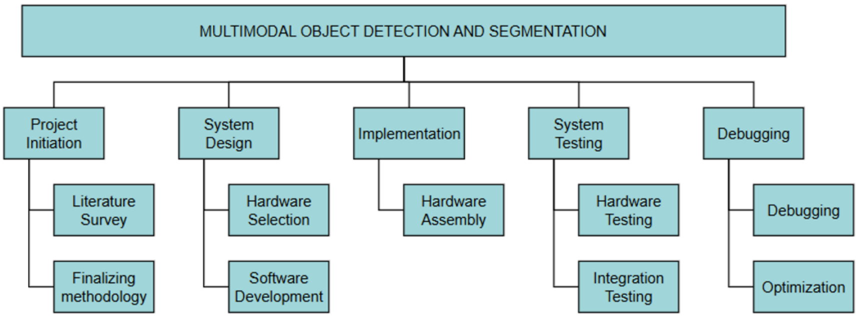

Figure 2: Work Breakdown Structure For Smart Object detection

This flowchart outlines the development process for a Smart Object detection using multimodal data system. It begins with project initiation, including literature survey and methodology planning. The system design phase covers hardware selection and software development, followed by implementation and testing. Finally, debugging and optimization ensure the system runs efficiently and accurately on embedded platforms like Raspberry Pi.

Design Specifications

The system ingests a continuous real-time video stream—either live capture from the Raspberry Pi Camera Module (Raspberry Pi-CAM-MB) or pre-recorded footage from a public dataset—at a resolution of 640×480 pixels. Each incoming frame is processed on-device by a lightweight YOLO model optimized for the Raspberry Pi's Tensilica Xtensa dual-core processor. Detection results are converted into simple text labels (e.g., "Person: 2.3 m") that include both the object class and its estimated distance. These labels are then transmitted in under 20–30 seconds per frame to an SSD1306 OLED display via I^2C, and simultaneously streamed to the Raspberry Pi's serial monitor for logging or debugging purposes.

8 Input Specifications

8.1 Raspberry-Pi-CAM Module

The Raspberry Pi–Camera setup is shown in Figure 3.1. The object detection workflow begins with the Raspberry Pi Camera Module capturing live video frames at 640 × 480 resolution. These raw frames serve as the system's input data, encompassing a wide range of environments, lighting conditions, and viewing angles to ensure robustness. As each frame arrives, it is fed directly into the optimized YOLO model running on the Pi's dual-core processor, and the resulting detections—including object class and distance estimates—are output as text labels. Because the Raspberry Pi performs all pre-processing and inference on-device, the overall detection accuracy is heavily influenced by the quality and diversity of the captured frames, making the camera's performance and configuration critical to the system's success.

8.2 Ultrasonic Sensor

The ultrasonic sensor acts as an input device to measure the distance between the detected object and the system. It emits sound waves and calculates the time taken for the echo to return, providing accurate distance measurements. This data is used alongside the YOLO model to enhance object detection by adding proximity awareness. The measured distance is then displayed on the OLED screen for real-time feedback.

9 Output Specifications

9.1 Displaying object name with distance on OLED

The output of our Smart Object detection using multimodal data system running on a Raspberry Pi with YOLO comprises real-time detections annotated with bounding boxes, segmentation masks, class labels, and estimated distances for each object in the scene. These results are transmitted in two forms: as concise text messages (e.g. "Car: 2.3 m", "Divider: 1.8 m") to an SSD1306 OLED display, and as a live overlay—showing bounding boxes, masks, labels, and confidence scores—on the video feed accessible via a local web interface hosted on the Pi. To ensure usability in interactive and safety-critical scenarios, the system enforces a confidence threshold that filters out low-probability detections, and delivers end-to-end detection-to-display latency of under one second per frame.

10 Software specifications

10.1 YOLO Based MODEL

All software components are deployed on Raspberry Pi OS and implemented in Thony Python. Video frames at 1280X780 resolution are captured and preprocessed using OpenCV, while the optimized YOLOv5n model is executed via Thony python. Detection outputs—object class and estimated distance—are transmitted to an SSD1306 OLED display. Code is developed directly on the Raspberry Pi using editors such as Thonny and run from the command line. For debugging and analysis, detailed detection overlays (bounding boxes, segmentation masks, and confidence scores) can be streamed to a local web interface or logged to the serial console, ensuring a fully self-contained, edge-only deployment.

10.2 Google colab

Google colab is central to our Smart Object detection using multimodal data project on the Raspberry Pi. Here Google colab is used to train lightweight YOLO models on custom datasets. After training, models are converted to PyTorch file.

10.3 Thonny Python

In Thonny Python on Raspberry Pi, you can begin by importing necessary libraries such as 'torch', 'cv2', and 'RPi.GPIO' for model handling, image capture, and GPIO control. Set up GPIO pins using 'RPi.GPIO.setmode(GPIO.BCM)' and define input/output pins with 'GPIO.setup()'. Load the lightweight YOLOv5n model using by loading a custom .pt file with 'torch.load()', followed by 'model.eval()' for inference. Capture frames from the Pi Camera or USB webcam using OpenCV ('cv2.VideoCapture') and pass them to the model for detection. Based on detection results, you can trigger GPIO outputs like OLEDs for real-time response.

10.4 List of tools used

1. Raspberry Pi 4B
2. Raspberry Pi-CAM
3. OLED

10.5 Resource specification

1. **Operating System:** Compatible with Windows, macOS, and Linux. Here Windows 11 is used for development and running the software. Raspberry pi OS is required.
2. **Power Supply:** Continuous power supply of 5V for the Raspberry Pi.
3. **Camera Module:** Raspberry-pi-CAM module with a low-resolution camera for optimized memory usage.

Methodology

The multimodal object detection system on Raspberry Pi captures real-time video from the Pi Camera Module, processing it with OpenCV for resizing and normalization. The lightweight YOLOv5-nano model is trained in Google colab, with data augmentation to improve accuracy. The trained model is then converted to PyTorch file. Inference is performed directly on the device, detecting objects, calculating distances, and sending results to an OLED display. The system's performance is evaluated based on accuracy to ensure real-time operation on the constrained hardware.

11 Functional block diagram for Smart object detection

The functional block diagram for Smart object detection as shows in Figure 4.1.

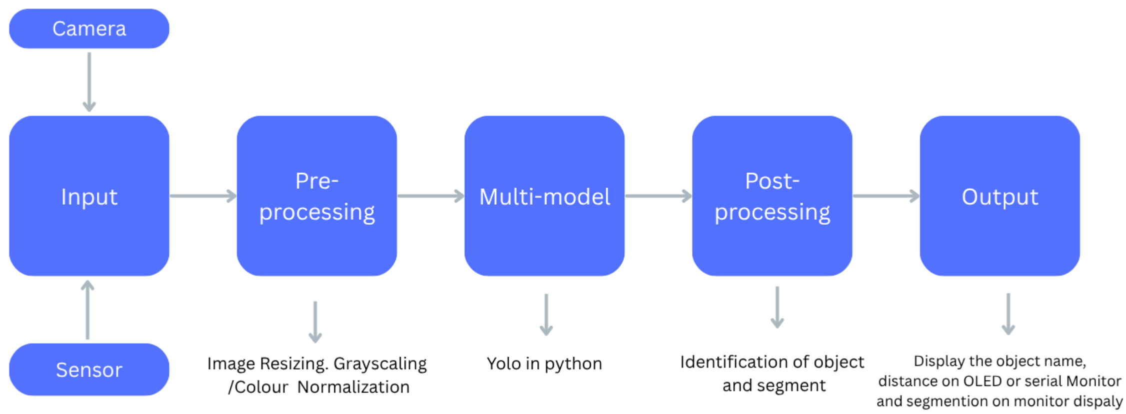

Figure 3: Functional block diagram for Smart object detection

11.1 Input

The system begins by capturing live images or video frames using the Raspberry Pi Camera Module. This camera serves as the primary sensor, providing raw visual data for object detection. The captured frames are in a resolution of 640x480, optimized for processing on the Raspberry Pi. These raw frames are then passed through preprocessing steps, such as resizing, normalization, and filtering, to prepare them for the next stage of analysis. This step ensures that real-time data is consistently fed into the system for accurate and timely object detection.

11.2 Preprocessing Stage

In the preprocessing stage, the captured image is prepared for the YOLOv5 model by following several steps. First, the image is resized to 640x480 pixels to match the resolution of the Raspberry Pi Camera Module and ensure consistency in processing. The pixel values are then scaled down from 0-255 to 0-1, making the data easier for the model to process. The color channels are rearranged to match the format expected by the YOLOv5 model. Finally, the image is wrapped into a batch format to ensure compatibility with the model, which processes images in batches. These preprocessing steps ensure the image is optimized for accurate and efficient object detection and segmentation.

11.3 YOLO based model

The YOLO-based model, optimized for deployment on the Raspberry Pi, performs real-time object detection and segmentation. It processes the preprocessed frames captured by the Pi Camera and accurately identifies and segments multiple objects within the image.It predicts bounding box coordinates, assigns class labels, and calculates confidence scores for each detection.This model is trained in Python and quantized to run efficiently on the Raspi-cam The inference results are then passed to the post-processing stage for further refinement.

11.4 Post-Processing:

The raw outputs of the YOLO-based model are refined through post-processing to ensure accurate and meaningful results. Predictions with low confidence scores are filtered out to reduce false positives, and the remaining bounding boxes are adjusted for precision. Each detected object is labeled with its class name and confidence score. The processed results are then formatted for display on the OLED screen or serial monitor, enabling real-time interpretation and user-friendly visualization.

11.5 Output

The final results, including class labels and object distance, are displayed on the OLED screen for easy, real-time interpretation. The system provides live updates, highlighting detected objects as they appear in the camera's view. These outputs offer a clear and intuitive representation of the surroundings, assisting users in identifying objects and their relative positions. The results can also be viewed via the serial monitor for monitoring or debugging purposes. This ensures the system remains efficient, user-friendly, and suitable for practical deployment.

12 Bill Of Material

The Bill of Materials (BOM) is a list of all the components, their specifications, quantities, and prices required for the assembly and implementation of the system. It ensures accurate cost estimation and procurement planning, with the total project cost 5,780. For this model, the BOM is given in Table 4.1.

No.	Component	Specifications	Qty	Price (rupee)
1	Raspberry Pi 4B	1.5GHz Quad-core 2GB/4GB/8GB RAM Gigabit Ethernet Wi-Fi, Bluetooth 5.0	1	5,000
2	Pi Camera	8MP/12MP sensor 1080p@30fps video 720p@60fps video	1	450
3	Ultrasonic Sensor	5V operation 2cm-400cm range $\pm$3mm accuracy 40kHz frequency	1	130
4	OLED-Display	128$\times$64 resolution Monochrome display I2C interface	1	200
5	**Total**		**4**	**5,780**

Table 1: Bill of Materials

13 Circuit Diagram

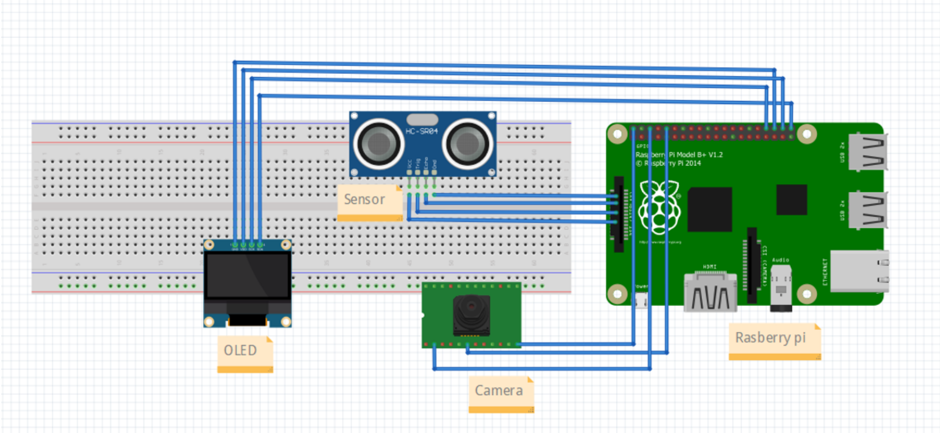

Figure 4: Circuit diagram for Smart object detection

This is the circuit diagram for the object detection system. The Raspberry Pi serves as the central controller, managing all components. The camera module captures images of the object for analysis. The ultrasonic sensor measures the distance to ensure the object is within detection range. The OLED display shows relevant output like distance or disease type. All components are connected via a breadboard using GPIO and I2C interfaces. The HC-SR04 ultrasonic sensor is connected to GPIO pins for accurate range detection. The OLED uses the I2C protocol for real-time display updates. The system integrates sensing, imaging, and display efficiently for driver assistance.

Results And Discussion

This section outlines the outcomes and insights gained from the development and testing phases of the Smart object detection system using YOLO Based Model and Raspberry Pi. The discussion is categorized into simulation results, experimental setup, test cases, and real-time implementation.

14 Results Of Smart object detection system

14.1 Optimization

YOLOv5 is the lightweight model, is used as a Optimization technique improve the object detection model for efficient deployment on the Raspberry Pi. In the post-optimization phase, the system achieved an improved accuracy, as shown in Figure 5.2. The optimized YOLOv5 model enabled the Raspberry Pi to detect a greater number of objects with higher precision in real-time. Additionally, integration with an ultrasonic sensor allowed for accurate distance estimation of detected objects, and the results—including object names and their distances—were displayed clearly on an OLED screen as shown in Figure 5.1. This optimization significantly enhanced the system's performance, enabling smooth real-time detection and display with reduced latency and better resource management.

14.2 Pre-Optimization

Pre-optimization in this project focuses on enhancing both hardware and software efficiency before final deployment. Here 5 classes of objects used for training. Initially the model trained with nearly 75 images, means 15 images of each class. Beacause of this the model was giving less accuracy of 71% as shown in Figure 5.1.

Figure 5: Pre-optimization for Smart Object Detection

14.3 Post-Optimization

Post-optimization, the system showed increased accuracy. The model again trained with nearly 130 images, means 25 images of each class. Hence the object detection accuracy is increased to 81%. This is shown in Figure 5.2.

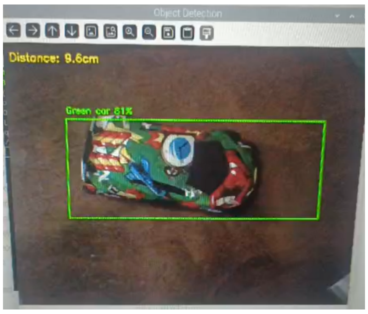

Figure 6: Post optimization for Object Detection

Conclusion and Future scope

15 Conclusion

In conclusion, the multi-model object detection system implemented on the Raspberry Pi using a YOLO-based model delivers an effective, real-time solution for detecting and identifying objects with high accuracy. The Raspberry Pi captures live video through its camera module, while the YOLO algorithm performs object detection locally. An ultrasonic sensor measures the distance to detected objects, and the results—such as object names and distances—are displayed clearly on an OLED screen. This edge-based approach ensures data privacy, low latency, and power efficiency, making it suitable for practical applications such as smart surveillance, home automation, and robotic vision. The system effectively combines hardware simplicity with algorithmic precision, demonstrating robust performance on a cost-effective platform.

16 Future scope

The Multi Model object detection system implemented on Raspberry Pi using a YOLO-based model has vast potential in real-world applications such as agriculture, disaster rescue, community surveillance, and home automation. In agriculture, the system can monitor fields to detect unwanted objects, track crop growth, or identify animals entering the farm area. During rescue operations, especially in flood-prone areas, it can detect people or obstacles and display their distance using the ultrasonic sensor, aiding navigation and alert generation. For community safety and surveillance, the system can detect suspicious movements or objects in real time, displaying relevant information like object name and distance on an OLED screen. By integrating with mobile or portable setups, it can be used for on-site monitoring without relying on cloud infrastructure. Future enhancements may include GPS tracking, solar-powered operation, and voice alerts, making the system even more effective in real-time, low-power, and remote applications.analysis.

References

1. P. Agrawal et al., "YOLO Algorithm Implementation for Real Time Object Detection and Tracking," 2022 IEEE Students Conference on Engineering and Systems (SCES), Prayagraj, India, 2022, pp. 01-06, doi: 10.1109/SCES55490.2022.9887678.
2. Di Feng, Christian Haase-Schuetz, Lars Rosenbaum, Heinz Hertlein, Fabian Duffhauss, Claudius Gläser, W. Wiesbeck, K. Dietmayer et al. ,"Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges" IEEE transactions on intelligent transportation systems India, 2022, pp. 01-06,10.1109/SCES55490.2022.9887678.
3. A. Carullo, Senior Member Ieee Marco Parvis, "An ultrasonic sensor for distance measurement in automotive applications," IEEE Sensors Journal 2021 Chennai, India, , pp. 01-04, doi: 10.1109/ICSSS54381.2022.9782169.
4. Jordan Filteau; Suk Jin Lee; Andrew Jung , "Real-Time Streaming Application for IoT Using Raspberry Pi and Handheld Devices 2018 IEEE Global Conference on Internet of Things (GCIoT), DOI: 10.1109/GCIoT.2018.8620141
5. Srećko Kunić, Zoran Šego "OLED technology and displays" Proceedings ELMAR-2012 Publisher: IEEE
6. Tasnim Sorwar, Sabbir Bin Azad, Sayed Rizban Hussain, Azfar Isa Mahmood , et al. "Real-time Vehicle monitoring for traffic surveillance and adaptive change detection using Raspberry Pi camera module." International Conference on Emerging Research in Computing, Information, Communication and Applications. Singapore: Springer Nature Singapore, 2023.
7. Xingkun Li, Guangyu Tian,Zhenghong Lu,Guojun Zhang "YOLO V5-MAX: A Multi-object Detection Algorithm in Complex Scenes." : 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC) DOI: 10.1109/R10-HTC.2017.8289003
8. Katsidimas, Li Zhang, et al. "A Model-Level Fusion-Based Multi-Modal Object Detection and Recognition Method" 2023 7th Asian Conference on Artificial Intelligence Technology (ACAIT) Jiaxing, China DOI: 10.1109/ACAIT60137.2023.1052838
9. Keying Jia, "Comparative Experimental Study of Infrared Distance Sensor and Ultrasonic Distance Sensor," 2024 3rd International Symposium on Sensor Technology and Control (ISSTC),Zhuhai, China DOI: 10.1109/ISSTC63573.2024