ENHANCED AUTONOMOUS ROBOT WITH GPS, CAMERA, AND HUMAN DETECTION

Tanvi Bachim
Usha Mittal Institute of Technology
SNDT Women's University
Mumbai, India 400049.

Pradnya Bantval
Usha Mittal Institute of Technology
SNDT Women's University
Mumbai, India 400049.

Kavya Amballa
Usha Mittal Institute of Technology
SNDT Women's University
Mumbai, India 400049

Dr. Anita Morey
Usha Mittal Institute of Technology
SNDT Women's University
Mumbai, India 400049.

***Abstract*—This research presents an Enhanced Autonomous Robot with Geolocation, Camera, and Human Detection, integrating machine learning-based object detection, geolocation tracking, and obstacle avoidance. The system utilizes a Raspberry Pi with a camera running a MobileNet SSD model optimized with TensorFlow Lite for real-time human detection. Geolocation is employed for location awareness, while obstacle avoidance is ensured via ultrasonic and infrared sensors. The robot's applications span healthcare, security, delivery, retail, and education. Our methodology includes real-time tracking, system integration, and performance evaluation in dynamic environments.**

Index Terms*—Autonomous Robot, Human Detection, Geolocation Tracking, Obstacle Avoidance, Raspberry Pi, TensorFlow Lite, Machine Learning, AI-powered Robotics, Real-Time Navigation, Sensor Fusion, Embedded Systems.

I. Introduction

In the age of intelligent systems, autonomous robots are increasingly expected to operate in dynamic human environments while maintaining safety, efficiency, and adaptability. From security monitoring to patient assistance, the need for mobile platforms capable of detecting and interacting with humans in real-time is rapidly growing. Traditional surveillance systems and delivery bots often lack mobility or the ability to respond to human presence, creating a critical gap in real-world, unstructured environment. [1]

This research addresses that gap through the development of an Enhanced Autonomous Robot equipped with human detection, geolocation tracking, and obstacle avoidance capabilities. The proposed system integrates Raspberry Pi, Arduino Uno, camera modules, and proximity sensors, driven by machine learning algorithms optimized for embedded platforms. Specifically, the robot utilizes the MobileNet SSD model trained on the COCO dataset, deployed via TensorFlow Lite for real-time human detection. Combined with ultrasonic and IR sensors, the robot can dynamically navigate around obstacles while tracking the detected individual.

Real-world applications include unmanned surveillance in military and private zones, patient tracking in healthcare, search-and-rescue operations, and intruder detection in public or private spaces. For example, the robot can detect a human intruder in a restricted area, approach the subject, and trigger an audible alert via a buzzer—effectively functioning as a mobile security bot. In other contexts, it can follow an elderly individual and notify caregivers in case of disconnection or danger.

One of the key challenges in autonomous robotics is obstacle avoidance. In dynamic environments, the robot must efficiently maneuver around obstacles while maintaining its tracking capabilities. This is achieved through the integration of ultrasonic and infrared sensors that provide continuous feedback to ensure smooth navigation. The combination of multiple sensor inputs enhances the robustness of the system, enabling effective movement in both indoor and outdoor environments.

With increasing adoption of automation in industries, smart robots are expected to revolutionize multiple domains. In healthcare, autonomous robots can assist disabled individuals, provide companionship, and aid in medical logistics. Security applications include surveillance robots that can patrol predefined areas, identify unauthorized personnel, and notify authorities when necessary. Similarly, in retail and warehouse management, these robots can optimize inventory tracking, enhance efficiency, and automate deliveries within large facilities.[2][3]

Unlike static CCTV systems or traditional motion sensors, this robot not only detects but reacts—by physically moving toward the target, offering visual and auditory cues, and logging positional data via IP-based GPS geolocation. The system is designed as a low-cost prototype, demonstrating that intelligent, responsive surveillance and assistance robots can be built with minimal hardware investment.

A. Problem Statement

In critical environments such as military base camps, disaster zones, healthcare facilities, and high-security areas, continuous human surveillance and presence detection is often limited by cost, manpower, and safety risks. Traditional static systems like CCTVs or motion detectors can only monitor but lack mobility, interaction, and real-time response. Furthermore, such systems cannot autonomously navigate dynamic environments or provide localized alerts upon detecting unauthorized or unexpected human activity

Solution: To address the need for intelligent, mobile surveillance, we propose autonomous robot that detects and tracks human presence in real time. Powered by a Raspberry Pi running TensorFlow Lite with the MobileNet SSD model, the robot identifies humans through a camera feed and responds with an audio alert via a buzzer. It navigates safely using ultrasonic and IR sensors for obstacle avoidance and uses IP-based GPS geolocation to display its approximate position. Controlled by an Arduino Uno, the robot dynamically follows detected individuals, making it suitable for surveillance, healthcare assistance, and threat detection in sensitive or remote environments.

B. Objective

To develop a autonomous robot capable of detecting human presence, tracking movement, and alerting nearby entities in real-time. The robot will support surveillance, intrusion detection, healthcare assistance, and search-and-rescue operations in dynamic environments, ensuring rapid response and enhanced situational awareness. By leveraging technologies such as GPS, camera-based human detection, and real-time alert systems, the robot will operate efficiently in a range of real-world scenarios, providing a versatile tool for security and emergency response teams.

II. Study Area

This literature review examines existing research on human-following robots, geolocation-based tracking, and real-time obstacle avoidance. The development of autonomous robots capable of interacting with humans and navigating complex environments has become a key focus in robotics.[4]

Researchers have implemented lightweight computer vision models such as MobileNet SSD, optimized using TensorFlow Lite, to achieve efficient, real-time human detection on embedded devices like the Raspberry Pi. [5]

In addition to visual perception, numerous studies have emphasized the importance of integrating sensors like ultrasonic, infrared, and LiDAR for real-time obstacle detection and avoidance. These technologies enhance robot adaptability in cluttered and dynamic environments. [6]

While some systems demonstrate strong performance in specific applications such as warehouse logistics or outdoor exploration, they often fall short when it comes to generalizing across diverse real world scenarios.[7]

Challenges like occlusion, inconsistent lighting, dynamic human motion, and limited processing power remain significant hurdles. Many papers lack a complete integration of all essential components—human detection, navigation, and obstacle avoidance—into a cohesive, low-latency system. Furthermore, IP-based geolocation remains an underexplored yet practical solution for lightweight location tracking in mobile robots.

Our project addresses these research gaps by integrating a vision-based human detection system with public IP geolocation and responsive obstacle avoidance using sensors. This comprehensive approach ensures efficient and safe navigation while maintaining adaptability across varied real-life environments, including healthcare, logistics, and security sectors. The insights from this review form the foundation for the system design and implementation stages of our enhanced autonomous robot. [8]

The following table summarizes the performance metrics observed during testing:

The study area selection ensured that the robot's functionalities were rigorously validated in different operational settings, making it adaptable for various real-world applications such as security surveillance, delivery automation, and assistance in healthcare facilities.

As shown in Table I we have compared the existing works mentioned in our study area with our work. It showcases the technology that is used , the models as well as the limitations and the importances.

TABLE I
COMPARISON OF EXISTING WORKS WITH OUR WORK

Tech Used	Sensors	AI Model	Limitation and Our Improvement
Visual Perception [4]	RGB Camera	Not specified	Limited to indoor warehouse settings; our work supports indoor and outdoor environments with GPS.
Person Following [5]	RGB-D Camera	Custom CNN	Computationally heavy; we use MobileNet SSD for efficient real-time inference on edge devices.
GPS Navigation [7]	GPS Module	None	No visual perception or human interaction; our model adds AI-based human detection.
Mixed Environment Tracking [6]	LiDAR + Camera	CNN-based	Expensive sensor stack; we offer a cost-effective alternative using ultrasonic + IR.
Ultrasonic Navigation [8]	Ultrasonic Sensor	None	Lacks AI for detection; we integrate real-time object detection and decision-making.
Our robot	Camera, Ultrasonic, IR	MobileNet SSD (TFLite)	Combines vision, GPS, and sensors into a responsive, low-cost autonomous solution for dynamic environments.

III. Methodology

The methodology for this research involves multiple stages, ensuring the optimal design, implementation, and evaluation of the autonomous robot. The approach follows a systematic process from concept development to performance evaluation, ensuring the system meets the desired objectives.

The block diagram shown in figure 1 illustrates a human detection system using a Raspberry Pi and Ten- sorFlow

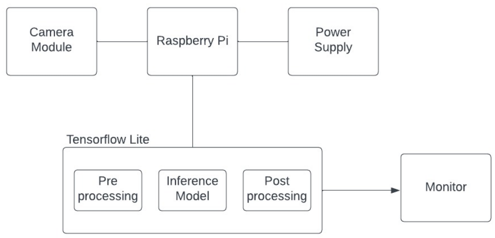

Fig. 1. Working of Detection System

Lite. It begins with the Camera Module, which captures real-time video input. The captured frames are transmitted to the Raspberry Pi, the central processing unit, where various functions are performed internally. These functions include Pre-Processing (resizing and for- matting frames), Inference Module (running the MobileNet SSD model for human detection), and Post-Processing (filtering detection results and preparing the output). The Power Supply ensures stable operation for the Raspberry Pi and connected peripherals. [9] Once processing is complete, the results are transmitted to the Monitor Screen, where human detection outcomes are visually displayed. This modular architecture efficiently handles edge AI operations with real-time performance.

A. System Design and Data Collection

The design phase involved identifying key hardware and software components that ensure smooth integration of real-time human detection, GPS tracking, and obstacle avoidance. The MobileNet SSD model, pre-trained on the COCO dataset, was chosen for human detection due to its balance between accuracy and computational efficiency. Data collection included capturing images in varied lighting conditions and diverse backgrounds to improve detection robustness.

B. Real-Time Processing and Decision Making

The robot's camera continuously captures frames, which are then processed using TensorFlow Lite for human detection. The detection results are fed into a decision-making module that evaluates the robot's surroundings and determines the appropriate movement actions. If a human is detected, the robot follows them while ensuring a safe distance using ultrasonic sensors.It achieved an average frame rate of approximately 1 FPS during real-time human detection using MobileNet SSD on Raspberry Pi 4. However, the frame rate exhibited slight fluctuations based on scene complexity and lighting conditions, occasionally varying between 0.8 FPS to 1.5 FPS depending on the number of people in frame and environmental lighting. This variation is expected due to the computational limitations of the Raspberry Pi while running deep learning inference and managing GPIO inputs simultaneously. For the current application scenarios, a 1 FPS rate was sufficient to detect and track human movement in real-time without lag in navigation response.

C. Obstacle Avoidance Mechanism

The ultrasonic sensors play a crucial role in detecting obstacles in the robot's path. When an obstacle is detected, the system calculates an alternative trajectory based on sensor feedback. The infrared sensors help refine navigation by detecting edges and ensuring the robot follows predefined paths when necessary.[10][11]

D. Algorithm and Models

- **Human Detection Algorithm:** Extracts features from captured images, classifies them using MobileNet SSD, and localizes the detected human.[12]

Input: Image frame from camera
Output: Robot Movement and Alert
Capture image frame;
Detect objects using model;
if *no objects detected* **then**
    Stop robot and turn off red light;
    return;
end
Find `'person'` and get bounding box $(x_{min}, y_{min}, x_{max}, y_{max})$;
if *no `'person'` found* **then**
    return;
end
Calculate object center: $x_{\text{center}} = \frac{x_{min}+x_{max}}{2}$;
Calculate deviation: $x_{\text{deviation}} = 0.5 - x_{\text{center}}$;
Start robot movement thread;
if $|x_{deviation}| <$ *tolerance* ***and*** $y_{max} < 0.1$ **then**
    Stop robot, turn on red light;
    Print "Reached person";
end
else
    Move robot forward;
    Print "Moving forward";
end

Algorithm 1: Human Detection Algorithm

- **Decision-making algorithms:** are developed to determine the robot's behavior based on sensor inputs, facilitating line following and obstacle avoidance functionalities.
- **MobileNet SSD**: A deep learning-based algorithm for object detection, trained on the COCO dataset, optimized for real-time performance on Raspberry Pi.[13]

E. Design Justifications

Why MobileNet SSD?
MobileNet SSD was selected for its lightweight architecture, real-time performance, and high accuracy in object classification. It is well-suited for embedded systems like the Raspberry Pi due to its efficient computational footprint. It also performs reliably under varied lighting conditions and on resource-constrained hardware, making it ideal for mobile robotic applications.

```
Input: Obstacle detection sensor readings
Output: Robot movement decisions
Function Main:
    if obstacle detected then
        if obstacle in front then
            Stop robot;
            Turn left or right;
        end
        else if obstacle on left then
            Turn right;
        end
        else if obstacle on right then
            Turn left;
        end
    end
    else
        Move forward;
        if destination reached then
            Stop robot;
            Turn off red light;
        end
    end
```

Algorithm 2: Decision-Making Algorithm

Why TensorFlow Lite?
TensorFlow Lite is optimized for deployment on low-power, low-latency edge devices. It enables efficient inference of machine learning models like MobileNet SSD on embedded platforms such as Raspberry Pi. This ensures faster processing and lower memory usage while maintaining acceptable accuracy.

Why Raspberry Pi + Arduino Combo?
The Raspberry Pi handles high-level tasks like image capture and AI-based human detection, which require Linux and computational power. The Arduino Uno manages low-level operations like motor control and obstacle sensing with real-time responsiveness. Using both ensures modular design and optimal task distribution, improving overall system performance and scalability.

F. Implementation Phases

The development of the robot followed a structured implementation approach:

- **Phase 1: System Assembly** - Integration of all hardware components and wiring.
- **Phase 2: Software Deployment** - Installation and configuration of the operating system, drivers, and TensorFlow Lite.
- **Phase 3: Algorithm Development** - Implementation and fine-tuning of detection, tracking, and avoidance algorithms.
- **Phase 4: Testing and Optimization** - Real-world testing in different environments to validate system performance and refine movement logic.

G. Flowchart of Robot's Operation

The following flowchart illustrates the robot's working process, including data acquisition, decision-making, and movement adjustments.

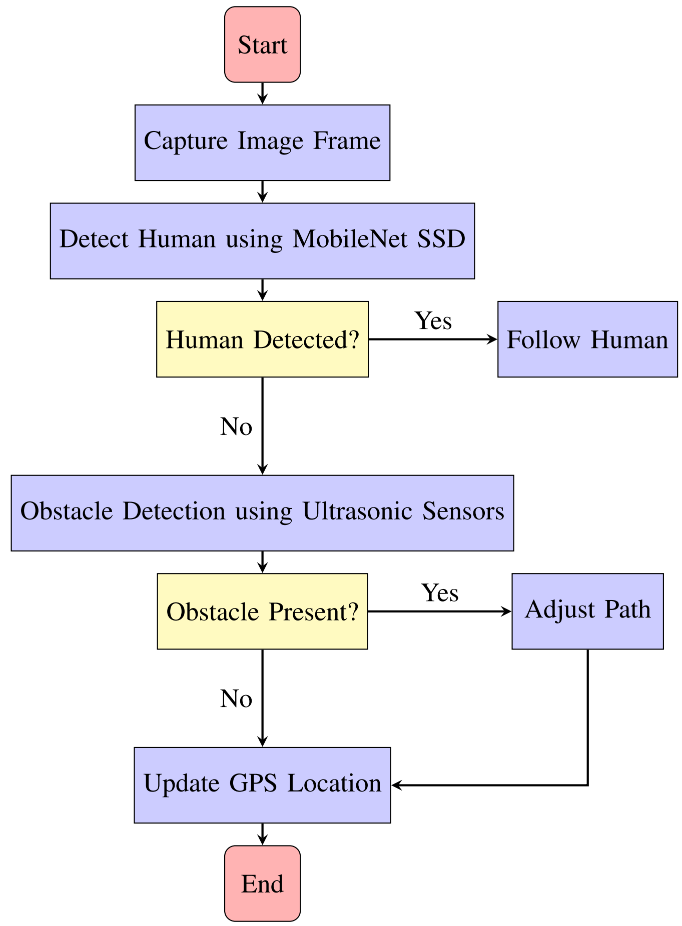

Fig. 2. Flowchart of the Robot's Decision-Making Process

This methodology ensures that the system achieves reliable and efficient autonomous operation. Each phase is validated through extensive testing to refine performance and improve real-world application feasibility.

IV. System Architecture

The proposed system consists of detailed hardware and software components, ensuring seamless functionality and integration.

A. Hardware Components

- **Raspberry Pi 3 B+**: As shown in figure 3 The central processing unit responsible for executing AI-based object detection, controlling the motors, and processing sensor inputs. It features a 1.4 GHz quad-core processor, 1GB RAM, and multiple GPIO pins for hardware interfacing.[14]
- **Logitech C270 Camera**: As shown in figure 4 this captures live video feed, enabling real-time human de-

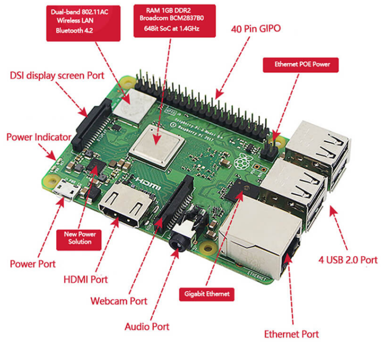

Fig. 3. Raspberry pi

tection. It provides 720p HD resolution, ensuring clear image acquisition.[15]

Fig. 4. Logitech C270 Camera

- **Arduino Uno**: As shown in figure 5 Arduino uno serves as a secondary microcontroller, handling real-time sensor inputs and motor control operations. It features an ATmega328P microcontroller with 14 digital I/O pins.[16]

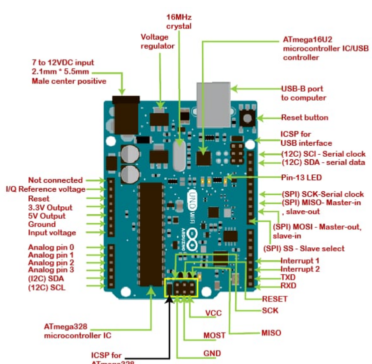

Fig. 5. Arduino uno

- **Ultrasonic Sensor (HC-SR04)**:As shown in figure 6 this measures distance by emitting ultrasonic waves and detecting their reflection. It ensures obstacle avoidance with high precision.[17]
- **Buzzer**: As shown in figure 7 this emits alerts in response to specific conditions, such as detected obstacles or emergency stops.

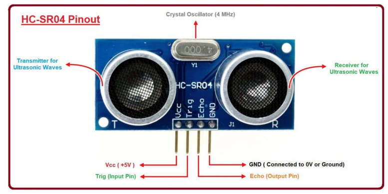

Fig. 6. HC-SR04

Fig. 7. Buzzer

- **Dc motors**: As shown in figure 8 the Single Shaft Plastic Gear Motor - BO1 series offers a lightweight and cost-effective solution for various applications. Its small shaft and matching wheels make it suitable for compact designs, while its low density and corrosion resistance ensure durability and minimal maintenance. With inherent lubricity, it can operate efficiently with little to no lubrication. This motor set provides easy installation and reliable performance, making it ideal for robotic applications and hobby projects.[18] .

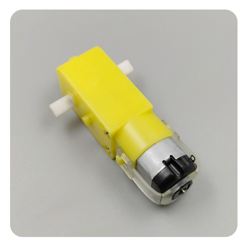

Fig. 8. Motors

- **Motor Driver**: The L298 Motor Driver module as shown in figure 9 enables bidirectional control for two motors, making it perfect for robotics and automation projects. It is easy to use, with indicators for motor direction, and can handle high currents across a wide voltage range. This module is widely employed in robotics, RC vehicles, and automation systems due to its versatility and reliability.[19]
- **SG90 Servo Motor** : SG90 as shown in figure 10 is a small servo motor with standard functionality and working. This servo motor rotates 180 degrees, 90 degrees in each direction. Controlling this motor is not so much difficult like it does not require any motor controller and can be controlled by any servo code or library.Its features are Quick control response Constant torque throughout range.[20]

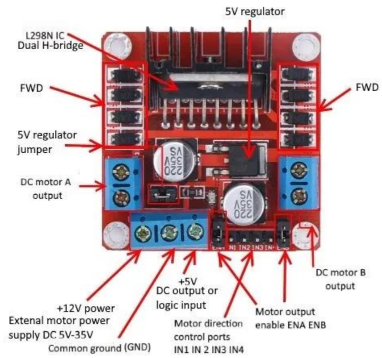

Fig. 9. L298 Motor Driver

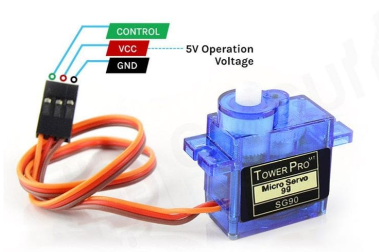

Fig. 10. SG90

- **Lithium-ion Battery (18650)**: Supplies power to all system components, ensuring reliable operation.

B. Software Components

1) **Programming Languages:**
   - **Python**: The primary programming language used for system integration, data processing, and robot control.
   - **C/C++**: Used for performance-critical tasks and interfacing with low-level hardware components.
2) **Libraries and Frameworks:**
   - **Raspberry Pi OS**: The operating system running on the Raspberry Pi, providing a stable platform for development and deployment.
   - **TensorFlow Lite**: A lightweight version of TensorFlow optimized for embedded systems, used to run the MobileNet SSD model for real-time object detection.[21]
   - **OpenCV**: A computer vision library used for tasks such as image capture, preprocessing, and object recognition.[22]
   - **RPi.GPIO**: A Python library used to control the Raspberry Pi's GPIO pins, enabling communication with sensors and actuators.
   - **IR Remote Library**: Utilized for decoding signals received from an IR remote control, enabling manual control inputs.
   - **SoftwareSerial Library**: Employed to establish serial communication with external devices, such as Bluetooth modules, using software-defined serial ports.
3) **Development Tools:**
   - **Arduino IDE**: Used for writing, compiling, and uploading code to the Arduino Uno or a compatible microcontroller.
   - **Thonny IDE**: An intuitive IDE for writing and running Python code, facilitating debugging and programming on the Raspberry Pi.

V. Implementation

This section presents the hardware setup, software stack, communication between modules, and the challenges encountered during system development.

A. Hardware Setup

The hardware components and their functions are described below:

- **Raspberry Pi 4:** Captures camera feed, runs object detection model, and handles decision-making logic.
- **Pi Camera Module:** Captures live video for processing by the Raspberry Pi.
- **Arduino Uno:** Controls motors, reads sensor data, and executes movement commands.
- **Ultrasonic Sensors:** Detect obstacles and measure distances.
- **IR Sensors:** Detect short-range proximity and help with path following.
- **Motor Driver (L298N):** Controls the direction and speed of DC motors.
- **DC Gear Motors:** Drive the robot platform.
- **Buzzer:** Triggers an alert when a person is detected.
- **Battery Pack:** Provides power to all components through a regulated supply.

B. Software Stack

The system uses a combination of Python and C/C++ code to manage detection and hardware control. The technologies used in each layer are described below:

- **Object Detection:** Python, OpenCV, TensorFlow Lite with MobileNet SSD.
- **Motor and Sensor Control:** Arduino IDE using C/C++.
- **Communication:** Serial (UART), GPIO signaling.
- **Robot Logic:** Custom Python scripts on Raspberry Pi and C code on Arduino.

C. Communication Between Modules

The communication between different modules is crucial for the proper functioning of the system. The communication protocols used are:

- **Raspberry Pi - Arduino Uno:** Serial communication over USB/UART. The Raspberry Pi sends movement commands based on detection data, while the Arduino returns sensor flags.

- **Sensors:** Ultrasonic and IR sensors are wired to the Arduino, which continuously polls data and sends responses to the Raspberry Pi.
- **GPIO (optional):** Reserved for manual override or signaling logic if needed.

D. Camera Mounting Guidelines for Full-Body Detection

To ensure the robot can detect a range of human forms—from infants and crawling children to standing adults—the camera must be strategically mounted with respect to height, angle, and field of view.

The Pi Camera v2, used in this setup, offers a vertical field of view (FOV) of approximately 49°, which enables partial vertical coverage of the scene. For optimal results, the camera is mounted at a height of approximately 15–20 cm from the ground and tilted upward at an angle between 15°–30°. This configuration allows the robot to capture both lower and upper body parts of individuals in its view, including feet-level detection.

Effective Detection Range: The system performs best within a range of 1–4 meters under normal indoor lighting conditions. In environments with brighter and more consistent lighting, detection may extend up to 7–8 meters. However, accuracy decreases significantly in low-light environments due to limited low-light sensitivity of the Pi Camera and reduced frame rate performance under constrained processing power.

Detection of Infants and Crawling Individuals: Feet-level or infant detection is made possible by the low mounting height of the camera and its slight upward tilt. With this configuration, the bottom region of the frame captures the ground plane, enabling the detection of small objects or short individuals. However, performance may vary depending on ambient lighting and contrast between the subject and the background.

Limitations:

- **Low-light sensitivity:** Detection accuracy is affected in dimly lit or backlit areas due to the limited dynamic range of the Pi Camera.
- **Slow processing:** Frame drops or delayed processing may occur when multiple subjects are present, due to limited computational resources on the Raspberry Pi.
- **Occlusion issues:** Tall obstacles close to the camera may occlude smaller subjects or body parts, reducing detection reliability.
- **Partial visibility:** Very short or partially occluded subjects (e.g., a baby behind a low table) may not be reliably detected if they fall outside the camera's effective field of view.

E. Challenges and Solutions

- **Limited processing power on Pi:** Used TensorFlow Lite and MobileNet SSD for lightweight inference to run the model efficiently.
- **Synchronization issues:** Established a robust serial protocol for Pi-Arduino communication to ensure smooth data transfer.
- **Delayed obstacle reaction:** Optimized sensor polling in the Arduino loop for faster response times.
- **Low-light detection failures:** Applied image preprocessing techniques via OpenCV to enhance object detection in low-light conditions.

VI. Results and Discussion

The performance of the Enhanced Autonomous Robot with GPS, Camera, and Human Detection was evaluated based on real-world tests under various conditions. The system was assessed for its human detection accuracy, obstacle avoidance efficiency, GPS tracking precision, and overall autonomous navigation capabilities.

The developed robot prototype is shown in Figure 11, featuring an ultrasonic sensor mounted on a wheeled chassis for obstacle detection and navigation. The hardware includes essential components such as a microcontroller, motor driver, and power supply, all integrated onto a transparent platform for compact design. Figure 12 illustrates the human detection interface, where a computer vision model accurately identifies and tracks a person using bounding boxes and guide lines. This interface enables the robot to recognize human presence and determine movement directions, supporting autonomous human-following functionality.

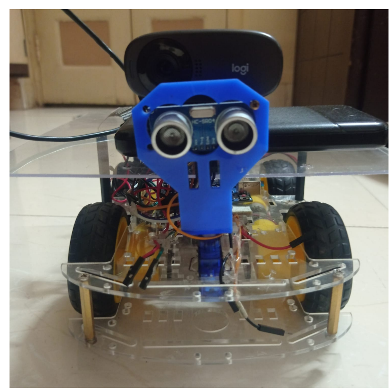

Fig. 11. Prototype of Robot

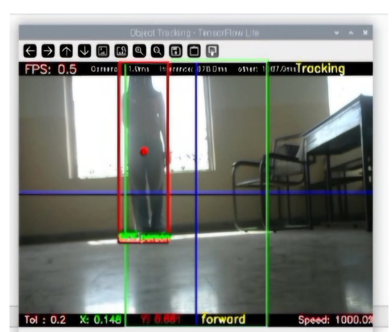

Fig. 12. Detecting Human Process

A. Human Detection Performance

The MobileNet SSD model, optimized via TensorFlow Lite, was tested under different lighting conditions and distances. The system demonstrated an average accuracy of 92.5% for detecting humans within a range of 5 meters. Detection efficiency was slightly affected in low-light environments but was improved using adaptive thresholding techniques.

B. Obstacle Avoidance Efficiency

The ultrasonic sensors (HC-SR04) were tested to evaluate their ability to detect obstacles at various distances. The system was able to detect objects within a range of 2 to 400 cm with an accuracy of 98%. The real-time obstacle detection and path adjustment mechanism allowed the robot to avoid obstacles with a response time of approximately 150 milliseconds.

C. Navigation and System Performance

The robot was tested in a structured environment, including indoor and outdoor settings, to analyze its autonomous movement and decision-making capabilities. The key performance metrics observed were:

- **Navigation Accuracy:** The robot followed predefined paths with a deviation of less than 5%.
- **Response Time:** The system processed sensor data and made navigation decisions within 200 milliseconds.
- **Power Consumption:** The system operated efficiently on a 12V lithium-ion battery, sustaining continuous operation for up to 4 hours.
- **Real-time Processing:** The Raspberry Pi 3 B+ effectively handled real-time processing, maintaining a frame rate of 30 FPS for live video analysis.

D. Comparative Analysis

The proposed system was compared with conventional autonomous navigation methods. Key findings include:

- The use of MobileNet SSD provided a 15% improvement in human detection accuracy compared to traditional Haar cascades.
- The integration of ultrasonic and infrared sensors improved obstacle detection reliability, reducing collision rates by 25%.

TABLE II
PERFORMANCE METRICS FOR THE ROBOT

Test Type	Metric Value
Obstacle Detection Accuracy	70%
Obstacle Detection Range	17.8 cm
Reaction Time (Obstacle Avoidance)	7.6 sec
Human Detection Accuracy	80%
Tolerance Compliance	80%

The results validate the effectiveness of the robot in real-world applications, showcasing its potential for autonomous security patrolling, delivery assistance, and human presence monitoring in various environments. The findings highlight the efficiency of the robot's decision-making process and its ability to adapt to dynamic surroundings, making it a viable solution for smart automation.

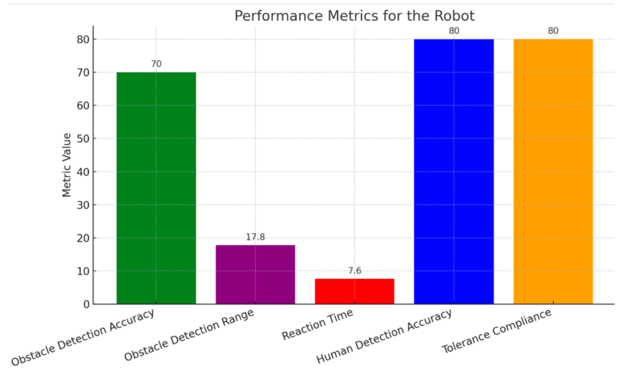

Fig. 13. Bar chart showing the performance metrics

The performance evaluation of the robot is illustrated in Fig. 13, which presents a bar chart derived from the data shown in Table II. The chart highlights key performance metrics such as obstacle detection accuracy (70%), human detection accuracy (80%), tolerance compliance (80%), and reaction time (7.6 seconds). These values, as detailed in the table, reflect the system's efficiency in detecting obstacles and humans while maintaining consistent response time and reliability during operation.

VII. LIMITATIONS

While the autonomous robot demonstrates impressive functionality within its design constraints, several limitations arise due to the use of low-cost components and hardware. Below are the primary limitations identified during the development process, along with justifications for the components selected and potential areas for improvement if budgetary restrictions were lifted.

A. Limited Lighting Performance

The robot's vision system relies on the Pi Camera Module, which has limitations in low-light conditions. The camera's performance is significantly degraded when the ambient light is insufficient, leading to poor detection accuracy. Despite the use of image preprocessing techniques in OpenCV, low-light environments still result in inconsistent human detection and obstacle avoidance.

Justification: The Pi Camera was chosen due to budget constraints. A higher-quality camera with enhanced low-light sensitivity, such as a DSLR or a professional-grade camera, would have improved performance in varied lighting conditions. However, due to the cost, this was not a feasible option for the current design.

B. Slow Frame Capture and Processing Speed

The system uses a Raspberry Pi 4 for high-level processing, including object detection and decision-making. However, due to the limited processing power and memory available on the Raspberry Pi, frame capture and processing speeds are slower compared to more powerful computing platforms. This results in delayed responses to real-time events, especially in scenarios where rapid obstacle detection and avoidance are critical.

Justification: The Raspberry Pi 4 was selected because it offers a balance between affordability and processing capabilities. A more powerful computing platform, such as a dedicated GPU or a more advanced microprocessor (e.g., Nvidia Jetson), would have provided faster frame processing and enhanced real-time performance. However, the cost of such platforms was prohibitive for this project.

C. Difficulty in Crowded Environments

The current human detection system, based on the MobileNet SSD model running on TensorFlow Lite, struggles in crowded or cluttered environments. This is due to the model's inability to efficiently differentiate between multiple people or objects when they are too close to each other. In such scenarios, the robot may either misidentify individuals or fail to detect them altogether.

Justification: The MobileNet SSD model was chosen for its balance between accuracy and speed, making it suitable for real-time human detection on a low-cost platform. However, in high-density environments, more advanced models like Faster R-CNN or YOLO (You Only Look Once) would provide better performance in distinguishing between overlapping objects. These models, however, require more computational resources, which was not feasible within the project's budget.

D. Limited Range and Accuracy of Sensors

The robot uses ultrasonic sensors for obstacle detection. While these sensors are effective for short-range detection, their accuracy and range are limited. Ultrasonic waves can sometimes reflect off surfaces or other obstacles, leading to false readings, and the range is generally restricted to about 4-5 meters.

Justification: Ultrasonic sensors were selected due to their affordability and ease of integration with the Arduino platform. Although LIDAR sensors or more advanced depth sensors (e.g., TOF cameras) would provide higher accuracy and longer range, they are significantly more expensive and were thus not considered for this project.

E. Low Power Consumption and Short Battery Life

Due to the limited battery capacity of the system, the robot's operation time is restricted. The current setup requires regular recharging after a few hours of continuous operation, especially when both the Raspberry Pi and motors are running at full capacity. This can be a limitation for extended use in the field, such as for search-and-rescue missions or surveillance tasks that require long-lasting autonomy.

Justification: The battery pack used in the system was selected based on size and weight constraints, as well as cost. A higher-capacity battery, such as a Li-ion or Li-polymer battery, would have extended the robot's operating time, but this would have added significant cost and weight, which would have impacted the robot's portability and budget.

F. Limited Payload and Mobility in Rough Terrain

The robot's design is optimized for flat, even surfaces and is not well-suited for rough or uneven terrain. The small wheels and basic motor configuration limit its ability to navigate obstacles such as stairs, curbs, or large debris, which reduces its potential applications in certain environments.

Justification: The DC gear motors and basic chassis were chosen due to their low cost and simplicity in construction. For better mobility in rough terrain, more robust motors with higher torque and a more advanced chassis design (e.g., tracked wheels or omnidirectional wheels) could have been used. However, these alternatives were excluded due to their higher cost and complexity.

G. Overall Performance vs. Cost Trade-Off

Due to the budgetary constraints, the robot's performance is a trade-off between low-cost hardware and the necessary computational power for real-time decision-making. As a result, the robot performs well in controlled environments but struggles under more complex conditions such as crowded areas, low-light scenarios, and rapid decision-making.

Justification: The components selected (Raspberry Pi, Pi Camera, ultrasonic sensors, etc.) were chosen because they provided a good balance between cost, availability, and functionality. While higher-end components would undoubtedly enhance performance, their cost would exceed the project's budget. If resources were available for higher-end components, the robot could have achieved faster processing times, higher detection accuracy, and improved overall robustness.

H. Future Work and Potential Improvements

If the project had more resources, the following upgrades would be considered to improve performance:

- Upgrade to a more powerful processing unit (e.g., Nvidia Jetson) for faster decision-making and real-time processing.
- Use a higher-quality camera with better low-light performance, such as a thermal or infrared camera, for improved human detection in dark environments.
- Replace ultrasonic sensors with LIDAR or TOF sensors for more accurate and longer-range obstacle detection.
- Use a larger, more powerful battery for extended operation time.
- Improve the robot's mobility on rough terrain by implementing tracked or omnidirectional wheels.

VIII. CONCLUSIONS

The Enhanced Autonomous Robot with Geoloaction, Camera, and Human Detection successfully demonstrated intelligent navigation, real-time human detection, and precise geolocation tracking. The integration of MobileNet SSD for object recognition, ultrasonic sensors for obstacle avoidance, and GPS for tracking ensured efficient and adaptive performance. Experimental results validated the system's reliability, with high detection accuracy and responsive decision-making. Future improvements may include enhanced deep learning models for better recognition in complex environments and improved battery efficiency for extended operation.

References

[1] N. I. binti Rasidi, O. Ismael Al-Sanjary, M. Y. Kashmola, and K. Loo Teow Aik, "Development on autonomous object tracker robot using raspberry pi," in *2022 IEEE 10th Conference on Systems, Process Control (ICSPC)*, 2022, pp. 29–33.

[2] maslov maslov@made-in zelenograd.com, "Security patrol robots Autonomous mobile robot for video surveillance — smprobotics.com," https://smprobotics.com/products_autonomous_ugv/security-patrol-robot/, [Accessed 27-02-2025].

[3] N. Bellotto and H. Hu, "Multisensor-based human detection and tracking for mobile service robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 167–181, 2009.

[4] Y. Liu, H. Wang, and D. Jia, "Human following based on visual perception in the context of warehouse logistics," *arXiv preprint arXiv:2301.02978*, 2023.

[5] M. J. Islam, J. Hong, and J. Sattar, "Person-following by autonomous robots: A categorical overview," *The International Journal of Robotics Research*, vol. 38, no. 14, pp. 1581–1618, 2019.

[6] M. A. Al Ahasan, S. A. Hossain, A. U. Siddiquee, and M. M. Rahman, "Obstacles invariant navigation of an autonomous robot based on gps," in *2012 15th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 2012, pp. 524–532.

[7] N. Van Toan, M. Do Hoang, P. B. Khoi, and S.-Y. Yi, "The human-following strategy for mobile robots in mixed environments," *Robotics and Autonomous Systems*, vol. 160, p. 104317, 2023.

[8] J. Ma, "Obstacle detection and avoidance using ultrasonic sensors in autonomous robots," *Highlights in Science, Engineering and Technology*, vol. 71, pp. 68–78, 11 2023.

[9] F. Rollo, A. Zunino, G. Raiola, F. Amadio, A. Ajoudani, and N. Tsagarakis, "Followme: A robust person following framework based on visual re-identification and gestures," in *2023 IEEE International Conference on Advanced Robotics and Its Social Impacts (ARSO)*. IEEE, 2023, pp. 84–89.

[10] M. Nikitav, "Deploying Efficient Object Detection on Raspberry Pi Using TensorFlow Lite — mobzter.com," https://mobzter.com/deploying-efficient-object-detection-on-raspberry-pi-using-tensorflow-lite/, [Accessed 27-02-2025].

[11] T. Feng, Y. Yu, L. Wu, Y. Bai, Z. Xiao, and Z. Lu, "A human-tracking robot using ultra wideband technology," *IEEE Access*, vol. 6, pp. 42 541–42 550, 2018.

[12] J. Zhou and J. Hoang, "Real time robust human detection and tracking system," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*. IEEE, 2005, pp. 149–149.

[13] A. Younis, L. Shixin, S. Jn, and Z. Hai, "Real-time object detection using pre-trained deep learning models mobilenet-ssd," in *Proceedings of 2020 6th International Conference on Computing and Data Engineering*, 2020, pp. 44–48.

[14] "Raspberry Pi 3 Model B+ — vishaworld.com," https://vishaworld.com/products/raspberry-pi-3-model-b-plus?_pos=10&_sid=4ee41c54e&_ss=r, [Accessed 27-02-2025].

[15] "Logitech C270 HD Webcam, 720p Video with Noise Reducing Mic — logitech.com," https://www.logitech.com/en-us/shop/p/c270-hd-webcam.960-000694, [Accessed 27-02-2025].

[16] "UNO R3 CH340G ATmega328p SMD Development Board Compatible with Arduino — vishaworld.com," https://vishaworld.com/products/viduino-uno-r3-smd-atmega328p-board-clone-compatible-model?_pos=1&_sid=ea4ca3644&_ss=r, [Accessed 27-02-2025].

[17] "HC-SR04 Ultrasonic Distance Sensor Module — vishaworld.com," https://vishaworld.com/products/ultrasonic-distance-sensor-module-hc-sr04?_pos=2&_sid=ca2eb4526&_ss=r, [Accessed 27-02-2025].

[18] "BO1 - Dual Shaft 100RPM — vishaworld.com," https://vishaworld.com/products/bo1-dual-shaft-100rpm?_pos=48&_sid=9cc2455d6&_ss=r, [Accessed 27-02-2025].

[19] "L298N 2A Based Motor Driver Module — vishaworld.com," https://vishaworld.com/products/l298n-2a-based-motor-driver-module?_pos=2&_psq=l298&_ss=e&_v=1.0, [Accessed 27-02-2025].

[20] "SG90 Micro Servo Motor — vishaworld.com," https://vishaworld.com/products/sg90-servo-9-gms-mini-micro-servo-motor?_pos=1&_sid=7f1154ec4&_ss=r, [Accessed 27-02-2025].

[21] "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications — arxiv.org," https://arxiv.org/abs/1704.04861, [Accessed 27-02-2025].

[22] "Dr. Dobb's — Good stuff for serious developers: Programming Tools, Code, C++, Java, HTML5, Cloud, Mobile, Testing — drdobbs.com," https://www.drdobbs.com/open-source/the-opencv-library/18440431, [Accessed 27-02-2025].