AI-based human tracking Robot

Inroduction to robotics &
Maths for computing-4

TEAM MEMBERS

Harish S.S. - CB.SC.U4AIE23230
Veluru Bhuvanesh - CB.SC.U4AIE23259
Gunda Harshavardhan - CB.SC.U4AIE23226
Avvaru Bharadwaj eswar - CB.SC.U4AIE23210

Introduction

Autonomous robots are increasingly used in airports, malls, and hospitals for assisting people with navigation and carrying loads.

- This project focuses on developing a 4-wheel robot that can follow a specific person from behind while maintaining a fixed distance.
- The system integrates AI-based person tracking (YOLO + DeepSORT) with LiDAR-based SLAM for real-time navigation and obstacle avoidance.
- A PID controller ensures smooth speed adjustments, while multi-sensor fusion (camera + LiDAR) improves tracking accuracy.
- The project combines robotics, AI, and mathematical models to enable intelligent human-following and path planning in dynamic environments.

- A live-streaming feature will allow remote monitoring of the robot’s camera feed via a web-based application

Problem Statement

In crowded environments like airports, malls, and hospitals, people often need assistance with carrying luggage or navigating spaces efficiently.

- A fully autonomous trolley should be able to follow a specific person without manual control.
- The robot must identify and track a target person in a group, ensuring it does not switch to another individual.
- The system should maintain a fixed distance from the person, dynamically adjusting speed based on movement.
- The robot must be capable of real-time obstacle detection and avoidance to ensure smooth navigation.
- The project requires advanced AI (YOLO, DeepSORT), and PID control for precise movement to achieve a reliable human-following system.

Objective

Develop an autonomous robot that can detect, identify, and follow a specific person in a dynamic environment.

- Implement AI-based tracking using YOLOv8 and DeepSORT to ensure accurate person re-identification.
- Integrate LiDAR-based model for real-time mapping and path planning, enabling the robot to navigate unknown environments.
- Maintain a fixed distance from the target using a PID controller, adjusting speed dynamically.
- Detect and avoid obstacles using LiDAR ensuring smooth movement in crowded areas.
- Synchronize all components (camera, LiDAR, motors, sensors) using multi-threading for real-time performance.
- Implement an alert system that notifies users in case of tracking failure, or low battery.
- Develop a web-based interface that provides real-time video streaming from the robot’s camera..

LITERATURE REVIEW

S NO	Title	Year	Authors	Observation
1	Human-following robot using infrared camera	Oct 2011	Quoc Khanh Dang 1 and Young Soo Suh2	Nintendo Wii infrared camera is used. low cost Can't be used under sunlight
2	A Novel Vision-Based Tracking Algorithm for a Human-Following Mobile Robot	2017	Meenakshi Gupta, Swagat Kumar,Laxmidhar Behera	Continuously updates the target appearance. Speeded-Up Robust Features (SURF) to detect humans in critical conditions. Not for multi-human tracking and more complex environment
3	Human following robot	2021	Amit kumar,Akash Pandey,Pankaj kumar	Laser Range Finder (LRF) for detecting human distance. Uses ultrasonic sensors, which are less accurate than LiDAR-based obstacle detection Limited Human Identification Capability
4	Single 2D lidar based follow-me of mobile robot on hilly terrains	2020	Jihoon Kim1 , Hyungjin Jeong2 and Donghun Lee2	follow a human on hilly terrains . Not efficient for crowd, and less perception about surroudings.

S NO	Title	Year	Authors	Observation
5	Moving Object Tracking Using Kalman Filter	2018	Pramod R. Gunjal; Bhagyashri R. Gunjal; Haribhau A. Shinde	Involves converting video sequences into individual frames . Predicting and correcting the estimated positions of moving objects, enhancing tracking accuracy. Assumption of Linear Motion
6	Unsupervised Person Re-Identification by Camera-Aware Similarity Consistency Learning	2019	Ancong Wu1 , Wei-Shi Zheng and Jian-Huang Lai	Proposed Solution – Camera-Aware Similarity Consistency (CASC) Learning. The paper tackles unsupervised person re-identification (Re-ID), aiming to match pedestrians across different camera views without labeled training data.
7	Motion planning and collision avoidance using navigation vector fields	2014	Dimitra Panagou	The designed vector fields inherently guide the robot around static circular obstacles, ensuring collision-free paths without the need for reactive obstacle avoidance strategies
8	Application of design of image tracking by combining SURF and SVM-based posture recognition system in robbery pre-alert system.	2023	Neng-Sheng Pai , Jiun-Hao Hong, Pi-Yun Chen,Jiun-Kai Wu	SURF improves feature matching, enabling tracking across complex backgrounds and slight pose variations.

Software Requirements

a)Computer Vision & Machine Learning

- OpenCV – Image processing for object tracking, edge detection, and filtering
- YOLOv8 – Real-time object detection to identify the human
- DeepSORT – Multi-object tracking to ensure continuous following
- HSV (Hue-Saturation-Value) – Color-based tracking for additional accuracy
- SURF-feature detection and matching algorithm
- Multi-threading – Ensures real-time synchronization of camera, LiDAR, motors, and sensors
- Flask + OpenCV + WebSockets – For real-time camera streaming on a web app

b. Speed control :

- PID Controller – Maintains a constant distance between the robot and the man.

c. Motor Control & Movement

- RPi.GPIO – Controls GPIO pins for motor interfacing.
- PWM (Pulse Width Modulation) – Adjusts motor speed for smooth motion.

d. Alert & Safety Mechanisms-Stop on vision failure

- Fail-Safe Mechanisms – Ensures robot stops and alert if tracking fails.

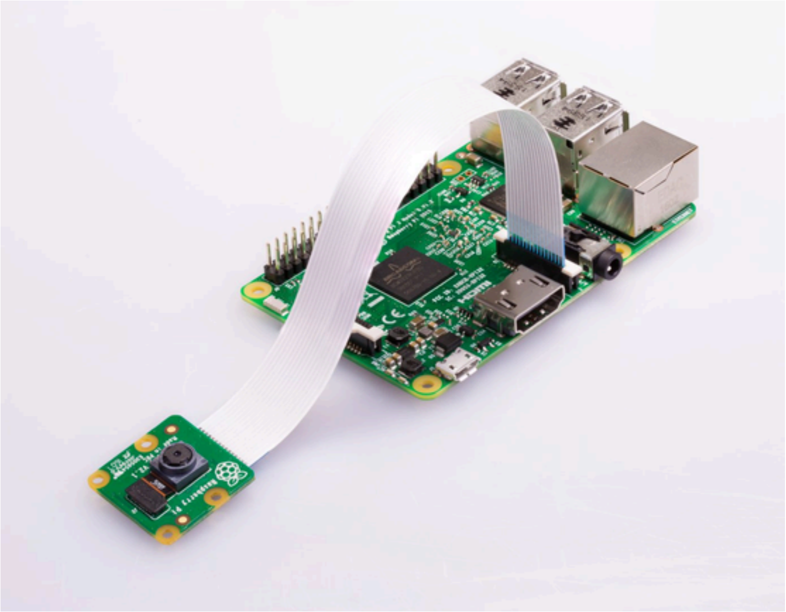
e. Remote Monitoring & Communication

- Flask (Python Web Framework) – Enables a web-based interface for live monitoring.
- OpenCV Video Streaming – Captures and transmits real-time video feed.
- WebSockets – Ensures smooth real-time data transmission.
- HTTP Requests (Flask API) – Allows remote control commands.
- Flask + OpenCV + WebSockets – For real-time camera streaming on a web app.

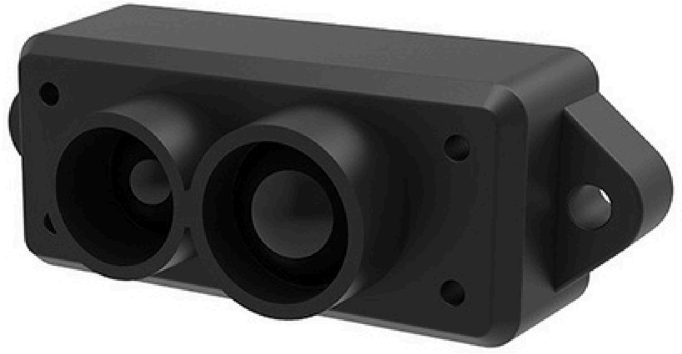
Hardware

- **Processing Unit:**
  - Raspberry Pi 4 – Runs AI models (YOLO, DeepSORT) and processes sensor data.
- **Vision & Tracking Sensors:**
  - Camera Module (Raspberry Pi Camera v2 /depth camera) – Captures real-time video for person detection.
- **Distance & Obstacle Detection:**
  - RPLIDAR A1 / TFmini-S (LiDAR Sensor) – Measures distance and builds a real-time map for navigation.
- **Motion :**
  - TB6612FNG Motor Driver – Controls motor speed and direction.
  - 4 x DC Motors with Encoders – Enables smooth movement and speed adjustments.
- **Power Supply:**
  - 12V Li-ion Battery Pack – Provides power to motors and sensors.
  - 5V Voltage Regulator – Ensures stable power for Raspberry Pi and camera module.

Hardware Requirements

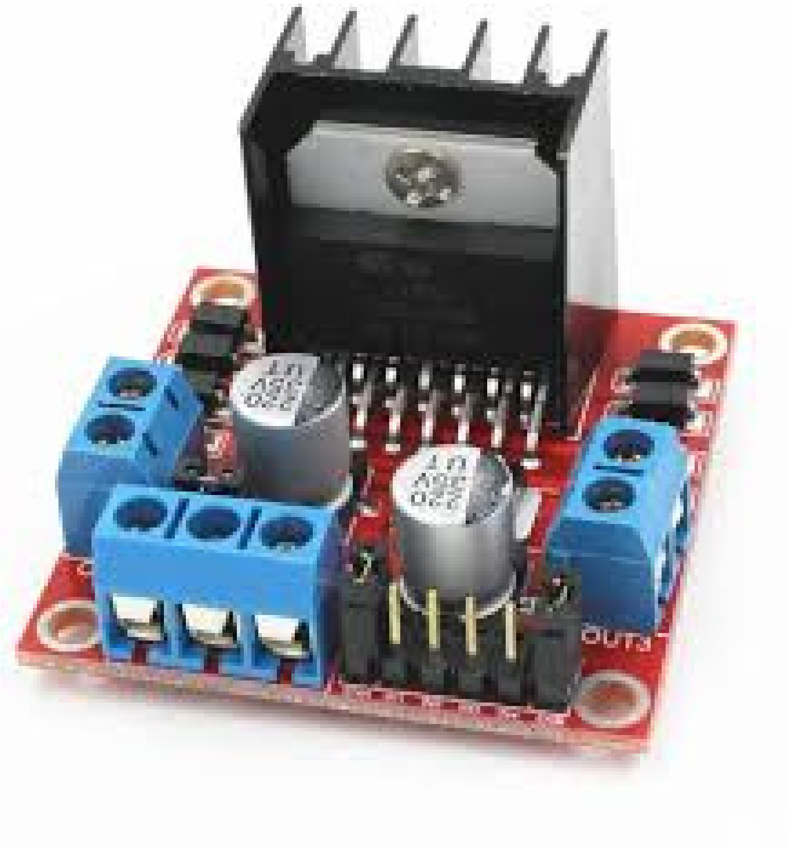


Raspberry pi camera v2

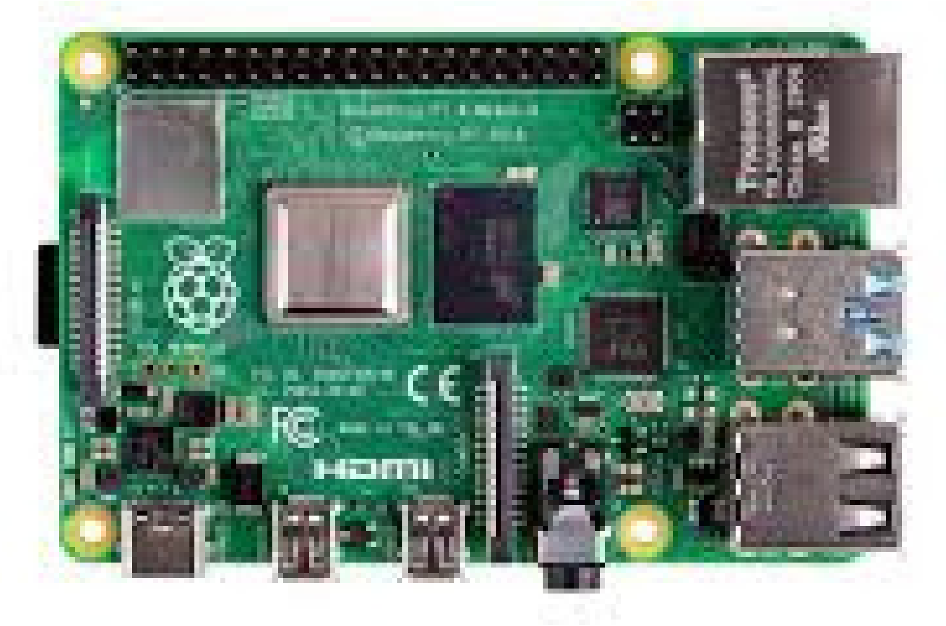


Single line LIDAR

Hardware Requirements

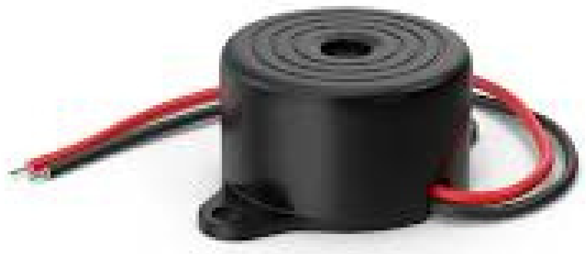


Motor Driver L298N

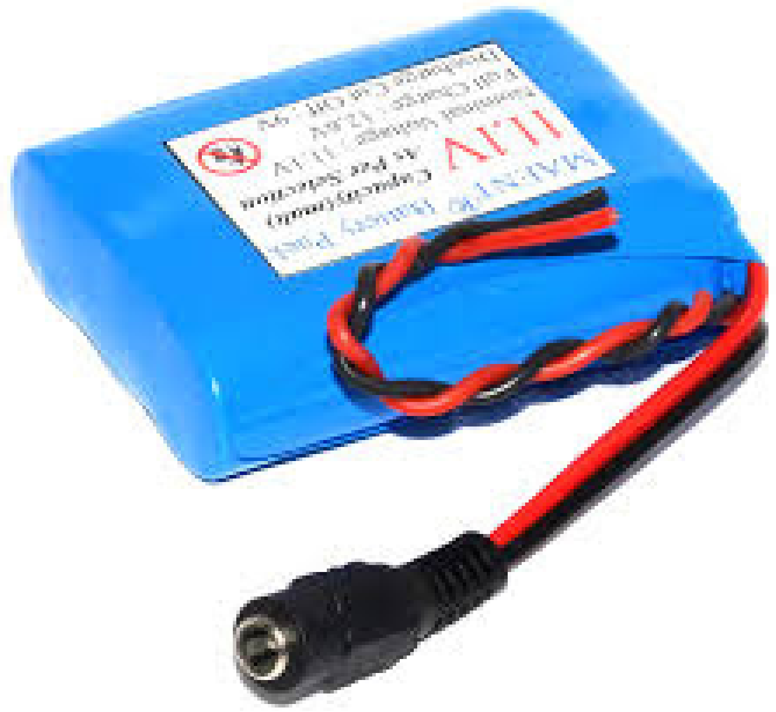


Rasberry Pi4

Hardware Requirements



Buzzer



Lithion ion battery

Methodology

01. Camera

Capture Video

02. Detect Human

Using SVM & SURF

03. Measure Distance (LiDAR)

To target ID person

04. PID Control

Maintain Fixed Distance

05. Obstacle Detection

LiDAR

06. ALERT SYSTEM

Buzzer alerts if tracking is lost or obstacle is too close

Synchronized Human Tracking Using SVM, SURF & Color Detection

Goal:

- Accurately detect, classify, and track a specific human using object detection, feature extraction, classification, and color tracking.

Working:

- YOLO (Object Detection) → Detects humans in real-time using bounding boxes.
- DeepSORT (Tracking Algorithm) → Assigns unique IDs to detected individuals for continuous tracking.
- SVM (Classification) → Determines if the detected object is a human or not.
- SURF (Feature Extraction) → Extracts body shape, movement, and texture for better recognition.
- Color Tracking → Ensures the robot follows the correct person by verifying predefined clothing color.

Synchronization:

- YOLO detects humans → DeepSORT tracks movement across frames.
- SURF extracts key features → SVM verifies if the detected object is a human.
- If classified as human → Color tracking ensures it is the correct target.
- If color matches → Robot continues tracking.
- If color doesn’t match → Robot ignores and searches for the correct person.

LiDAR-PID Synchronization for Distance Control

Goal: Maintain a constant distance between the robot and the human.

Working:

- Camera (YOLO/DeepSORT) detects & tracks the human.
- LiDAR measures the exact distance between the robot and the human.
- PID Controller takes LiDAR distance input and adjusts the robot's speed accordingly.

Synchronization:

- If LiDAR detects the human is too close → PID reduces robot speed.
- If LiDAR detects the human is too far → PID increases robot speed.
- Camera ensures tracking accuracy while LiDAR maintains the correct distance.

Camera & LiDAR Synchronization

Goal: Track the human while avoiding obstacles.

Working:

- Camera (YOLO Bounding Box) detects & tracks the human.
- LiDAR scans surroundings to detect obstacles and measure distance.

Synchronization:

- Human within bounding box → Camera keeps tracking.
- Obstacle detected by LiDAR → Robot decides whether to move left or right while maintaining tracking.
- LiDAR continuously updates obstacle positions → Robot dynamically adjusts movement to ensure smooth, collision-free tracking.

Web Streaming & Processing Synchronization

Goal:

To ensure seamless synchronization between real-time video capture and web-based viewing, allowing remote monitoring without affecting the robot's local processing.

Working:

Camera & Video Processing

- The robot's camera captures live video.
- Video frames are processed locally for tracking and movement decisions.

Web Streaming Integration

- The processed video is streamed via a Flask web server.
- The web interface continuously fetches and displays real-time video.

Parallel Execution

- The robot's movement and tracking run independently of web streaming.
- Web streaming is only for visualization, ensuring no delay in decision-making.

Mathematics

YOLO Bounding Box Calculation (For Person Detection)

$$B=(x,y,w,h)$$

- $x,y \rightarrow$ Top-left coordinates of the detected person
- $w,h \rightarrow$ Width & height of the bounding box

Cosine Similarity (For Re-Identification Using DeepSORT)

$$\cos(\theta)=A \cdot B/||A||.||B||$$

- Measures similarity between two feature vectors A and B

LiDAR Distance Calculation

$$d=c\times t/2$$

$d \rightarrow$ Distance to the person

$c \rightarrow$ Speed of light

$t \rightarrow$ Time taken for the LiDAR pulse to return

Mathematics

PID Controller for Distance Maintenance

Error Calculation (Distance Control)

e(t)=d_desired−d_current

- e(t)→ Error (difference between desired & actual distance)
- Robot increases/decreases speed based on this error

PID Control Equation (Adjusting Speed Dynamically)

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt}$$

-
- Kp → Proportional gain (reacts to current error)
- Ki → Integral gain (fixes accumulated errors)
- Kd → Derivative gain (prevents overshooting)

LiDAR Angle Mapping (Matching Camera & LiDAR Data)

$$\theta = \frac{x_p - x_{\text{center}}}{x_{\max}} \times \text{FOV}$$

- Converts camera's image coordinates to LiDAR angle

Mathematics

Mathematical Models for Obstacle Avoidance

Repulsion Force for Obstacle Avoidance (Vector Field Navigation)

$$F_{\text{repel}} = \frac{K}{(d_o^2 + \epsilon)}$$

- do → Distance to obstacle
- K → Repulsion constant

Kalman Filter (Motion Prediction for Tracking Stability)

- Predicts the next position of the tracked person
- Reduces noise in sensor readings (LiDAR + Camera Fusion)

$$X_t = AX_{t-1} + BU_t + W_t$$

Xt → Estimated position of the person at time t.
AXt−1 → Predicted position based on previous movement.
BUt→ Control input (robot's response to person’s movement).
Wt → Process noise (random movement variations).

Expected outcomes

- Accurate Real-Time Person Tracking
- Smooth Distance Maintenance Using PID Control
- Efficient Obstacle Detection & Avoidance
- Live Camera Streaming & Web-Based Monitoring
- Real-Time Alerts for Safety & Tracking Issues
- Scalable & Future-Ready Design

Timeline

1. RESEARCH & PLANNING
2. HARDWARE ASSEMBLY & SETUP
3. IMPLEMENT AI-BASED PERSON TRACKING
4. DISTANCE ESTIMATION & PID CONTROL
5. OBSTACLE DETECTION & AVOIDANCE
6. LIVE STREAMING & ALERT SYSTEM
7. SYSTEM INTEGRATIO N & TESTING
8. FINAL OPTIMIZATION & DEPLOYMENT

Thank you