IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

AUTONOMOUS ROBOTIC SURVEILLANCE AND SECURITY SYSTEM

Tanay Chaudhari, Aadarsh Singh, Atharva Dabke, Deepak Mahajan, Prof. Vinod Chede

Department of Electronics and Telecommunication Engineering, Terna Engineering College, Plot no. 12, sector -22 opp. Nerul Railway Station, phase-11 Nerul (W), Navi Mumbai 400706, University of Mumbai ,Maharashtra , India

Abstract: This paper presents the design and development of an autonomous indoor surveillance and security robot capable of reliable operation in GPS-denied environments such as offices, warehouses, residential buildings, and industrial facilities. The proposed system addresses the limitations of traditional fixed-camera surveillance by integrating autonomous navigation, real-time perception, and multi-sensor threat detection into a mobile robotic platform capable of continuous operation without constant human supervision. The core navigation capability is achieved through visual-inertial odometry, which fuses visual data from a USB webcam with inertial measurements from an MPU6050 six-axis inertial measurement unit comprising a three-axis accelerometer and a three-axis gyroscope. This sensor fusion approach mitigates the scale ambiguity limitation of monocular visual odometry and reduces cumulative drift errors associated with wheel encoder-based odometry. The fused pose output drives a continuous simultaneous localization and mapping loop, implemented using ORB-SLAM3 operating in monocular-inertial mode within the ROS 2 Jazzy Jalisco framework, that generates a persistent map of the environment and maintains self-localization throughout autonomous operation. The robotic platform employs a six-wheel drive configuration powered by six 25GA370 DC geared motors rated at twelve volts. Motor control is implemented using three TB6612FNG dual H-bridge motor driver modules. High-level computation is performed on a Raspberry Pi 4 Model B single-board computer, while an ESP32 microcontroller handles exclusively low-level real-time motor PWM control. The system integrates an HC-SR04 ultrasonic sensor mounted on an SG90 servo for dynamic obstacle scanning, an MQ-2 smoke and gas sensor for environmental hazard detection, and robotic arms implemented in single-servo and double-servo configurations using MG90S micro-servo motors controlled via a PCA9685 PWM driver. The software architecture is built on the Robot Operating System 2 Jazzy Jalisco framework, incorporating visual-inertial odometry, SLAM-based mapping, computer vision-based threat detection using OpenCV with a MobileNet-SSD v2 model deployed via TensorFlow Lite, and sensor-fusion-based obstacle avoidance. Component-level and integrated system testing was conducted in real indoor environments. The system successfully demonstrates stable multi-waypoint autonomous navigation, obstacle avoidance without halting, environmental hazard detection with hardware-level emergency motor stop, and real-time computer vision processing on embedded hardware. Navigation accuracy is expected to be within fifteen to twenty centimetres under favorable indoor conditions, consistent with reference literature on comparable low-cost sensor configurations, though this figure represents a hardware-specification-based estimate rather than a directly measured experimental result. These outcomes demonstrate the practical feasibility of an integrated, affordable, and intelligent indoor surveillance solution using commercially available embedded components.

Index Terms— Autonomous Surveillance, Indoor Navigation, Visual-Inertial Odometry, SLAM, Sensor Fusion, Mobile Robotics, GPS-Denied Environments, Obstacle Avoidance, Threat Detection, Computer Vision, ROS, Embedded Systems, Gas Detection, Inertial Measurement Unit (IMU), Embedded Computing

I. INTRODUCTION

The rapid advancement of robotics, embedded systems, and artificial intelligence has fundamentally transformed the design and implementation of security and surveillance operations. Growing concerns related to safety, asset protection, unauthorized access, and emergency response across residential, commercial, and industrial environments have created strong demand for intelligent surveillance systems capable of operating continuously with minimal human intervention. Traditional surveillance solutions have largely relied on fixed closed-circuit television systems that provide real-time video feeds monitored by human operators. While widely adopted, such systems suffer from inherent limitations including restricted coverage due to fixed viewpoints, dependence on continuous human supervision, susceptibility to operator fatigue, and inability to actively respond to detected threats or dynamically reposition for improved situational awareness. To overcome these limitations, mobile robotic surveillance systems have emerged as a compelling alternative. Unlike fixed cameras, mobile robots can actively patrol an environment, reposition themselves to investigate suspicious activity, and cover large areas using a single platform. The convergence of mobility with sensing and intelligence enables surveillance robots to provide dynamic coverage, reduce blind spots, and improve overall situational awareness. Early implementations were constrained by limited computational power, unreliable navigation, and high system costs. However, recent developments in low-cost embedded computing, compact sensors, and efficient algorithms have made capable surveillance robots feasible using commercially available components. Navigation remains one of the most critical challenges in autonomous mobile robotics. Outdoor robotic systems often rely on Global Positioning System receivers for absolute position information. However, GPS signals are significantly attenuated or entirely unavailable indoors due to attenuation caused by walls, ceilings, and metallic surfaces, making indoor environments commonly referred to as GPS-denied environments. Wheel encoder-based odometry estimates robot motion by integrating wheel rotation measurements over time but suffers from cumulative errors caused by wheel slippage, uneven surfaces, and mechanical wear. Visual odometry offers a promising alternative by leveraging camera data to estimate motion by tracking visual features across consecutive frames, making it inherently immune to wheel slippage effects. However, monocular visual odometry faces the challenge of scale ambiguity, which limits long-term accuracy. Visual-inertial odometry addresses this challenge by fusing visual data from a camera with inertial measurements from accelerometers and gyroscopes, exploiting the complementary characteristics of both sensor modalities. Visual measurements provide long-term stability by anchoring motion estimates to environmental features, while inertial measurements improve shortterm accuracy and help resolve scale ambiguity. When combined with a simultaneous localization and mapping pipeline, visualinertial odometry enables a robot to build a persistent map of its surroundings while maintaining real-time self-localization throughout extended autonomous operation. In parallel, significant progress in computer vision and artificial intelligence enables machines to interpret visual information and make intelligent decisions. Modern computer vision frameworks allow embedded systems to detect objects, recognize human presence, and identify potentially hazardous situations in real time. When integrated with robotic mobility, these capabilities transform surveillance robots from passive monitoring devices into intelligent agents capable of understanding their environment and responding to detected events. The convergence of these technologies provides the foundation for developing an autonomous indoor surveillance robot capable of reliable navigation, intelligent perception, environmental hazard detection, and effective threat response. This paper presents an integrated framework for autonomous indoor surveillance that combines visual-inertial odometry with continuous SLAM-based localization and mapping, sensor-fusion-based obstacle avoidance using a scanning ultrasonic sensor, computer vision-based threat detection, smoke and gas sensing for environmental hazard monitoring, and robotic arm actuation for physical response. The complete system is implemented within a ROS-based software architecture and deployed on a physical mobile robot platform. The framework is validated through component-level testing

and integrated system evaluation, demonstrating reliable autonomous navigation, effective surveillance, and real-time performance within the computational constraints of a Raspberry Pi-class embedded platform.

II. SYSTEM ARCHITECTURE AND HARDWARE DESIGN

The proposed autonomous surveillance system is implemented on a compact six-wheeled mobile robot designed for reliable operation in indoor environments. The platform follows a modular architecture in which sensing, localization, planning, perception, and motion control are integrated through the Robot Operating System middleware. This design enables flexible development while ensuring real-time interaction between all functional modules.

A. Mobile Robot Platform

The robotic platform uses a six-wheel drive configuration mounted on a 3D printed chassis to provide mechanical stability, improved traction, and load distribution on indoor surfaces. Six 25GA370 DC geared motors rated at twelve volts and one hundred thirty revolutions per minute independently drive the wheels, providing consistent forward propulsion across all operating conditions. Directional control is achieved through a servo-driven gear steering mechanism mounted on both sides of the chassis. MG90S micro-servo motors drive a spur gear train that physically rotates the wheel assemblies to the commanded steering angle, allowing the rover to turn left or right while all drive motors continue operating at a consistent forward speed. This architecture completely decouples propulsion from steering — the DC motors are responsible exclusively for forward drive, while the MG90S servo and gear assembly are responsible exclusively for directional control. The drivetrain is designed to prioritize torque consistency and smooth low-speed operation, essential for indoor robotic systems operating in confined environments. The sixwheel configuration also provides improved redundancy and fault tolerance, ensuring that even under partial motor degradation, mobility is maintained.

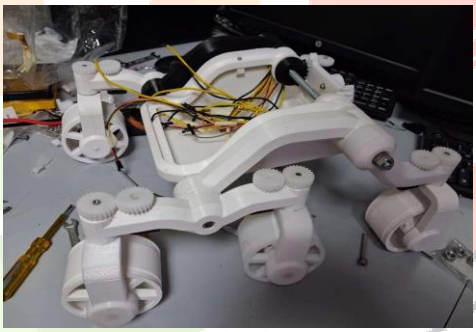

Fig. 1: Hardware assembly of the autonomous surveillance robot showing the six-wheel drive chassis, DC geared motor mounting, and initial electronic component placement during the fabrication phase.

B. Motor Control and Drive Electronics

Motor actuation is handled using three TB6612FNG dual H-bridge motor driver modules, each capable of independently controlling two DC motors. The TB6612FNG supports a motor supply voltage of 2.5V to 13.5V and delivers a continuous output current of 1.2A per channel with a peak of 3.2A. It incorporates built-in protection features including thermal shutdown, overcurrent protection, and under-voltage lockout, ensuring reliable operation under varying load conditions. The motor drivers receive pulsewidth modulation signals and direction control inputs from the ESP32 microcontroller. The use of dedicated motor driver modules protects the microcontroller from high current loads and voltage transients generated by motors during operation and enables fine control over robot motion with support for braking functionality.

C. Dual Controller Architecture

The system adopts a dual controller architecture that separates high-level computational tasks from low-level real-time control responsibilities. The Raspberry Pi 4 Model B serves as the primary processing unit running Ubuntu 24.04 LTS with the Robot Operating System 2 Jazzy Jalisco framework. It executes all computationally demanding tasks including visual-inertial odometry computation via ORB-SLAM3 in monocular-inertial mode, SLAM-based mapping and localization, computer vision-based threat detection using OpenCV with a MobileNet-SSD v2 model deployed under TensorFlow Lite, sensor fusion for

navigation decisions, overall system coordination, direct acquisition of HC-SR04 ultrasonic ranging data via GPIO, MQ-2 smoke and gas sensor monitoring, SG90 scanning servo control via GPIO PWM, and servo actuation through the PCA9685 PWM driver via I²C. It is acknowledged that this aggregate workload places significant demand on a single Raspberry Pi 4B. In practice, CPU utilization reaches high levels during simultaneous VIO computation, SLAM update, and vision inference. This can cause frame rate reduction and increased latency in the vision pipeline during peak load conditions. Accordingly, the system is tuned for moderate patrol speeds and controlled indoor environments where peak load events are manageable. The VIO pipeline is prioritised in the ROS 2 node scheduling configuration to maintain navigation continuity during periods of high compute demand.

The PCA9685 sixteen-channel PWM driver manages two functionally distinct categories of servo actuation, addressed as separate channel groups and commanded independently by the Raspberry Pi based on navigation state and alert state respectively. The first channel group drives the MG90S steering servos connected to the chassis spur gear mechanism, which physically rotate the wheel assemblies left or right in response to obstacle avoidance and waypoint navigation heading corrections. The second channel group drives the MG90S robotic arm servos, which execute predefined motion sequences in response to computer vision threat detections or MQ-2 hazard alerts. These two groups operate completely independently — a steering command issued during obstacle avoidance does not interfere with arm servo positions, and an arm actuation command triggered by a threat detection does not affect the steering angle.

The ESP32 microcontroller serves as the secondary processing unit responsible exclusively for time-sensitive low-level motor actuation, including PWM signal generation to the three TB6612FNG dual H-bridge motor driver modules and execution of hardware-level emergency motor shutdown upon receiving a halt command transmitted from the Raspberry Pi over the UART serial link. This strict separation of responsibilities ensures that time-critical motor control operations are never disrupted by the computational load on the primary processing unit. Communication between the two controllers is established using UART serial communication at 115200 baud in the 8N1 configuration, through which high-level forward drive speed commands and emergency halt instructions are transmitted from the Raspberry Pi to the ESP32. Upon detection of a hazardous gas concentration by the MQ2 sensor, the Raspberry Pi transmits an emergency halt command to the ESP32 over the UART serial link, upon which the ESP32 immediately cuts all PWM signals to the TB6612FNG motor drivers, stopping all motor activity at the hardware driver level without software intervention on the primary controller.

D. Sensing and Perception Components

Localization, navigation, and surveillance perception are achieved through a combination of onboard sensing modalities. Visual perception is provided by a USB webcam mounted at the front of the robot and interfaced directly with the Raspberry Pi. The camera captures continuous image frames that serve a dual purpose: visual-inertial odometry for motion estimation, and computer vision-based surveillance for threat detection. Inertial measurements are provided by an MPU6050 six-axis inertial measurement unit integrating a three-axis accelerometer and a three-axis gyroscope. The MPU6050 communicates with the Raspberry Pi via the I²C interface and provides high-frequency motion data used to improve orientation estimation and supplement visual odometry during periods of visual degradation such as lighting changes or featureless surfaces. Obstacle detection is implemented using an HC-SR04 ultrasonic distance sensor mounted on an SG90 micro-servo motor. The SG90 servo is commanded directly by the Raspberry Pi via a dedicated GPIO PWM pin to perform continuous left-to-right sweeps of the forward arc. For each angular step, the corresponding distance reading from the HC-SR04 is tagged with its angle and processed directly on the Raspberry Pi, effectively producing a low-cost two-dimensional proximity fan consumed immediately by the sensor fusion and navigation layers without any intermediate forwarding. Environmental hazard detection is achieved using an MQ-2 smoke and gas sensor interfaced directly with the Raspberry Pi, capable of detecting smoke and combustible gases including methane and butane, providing an analog output signal proportional to detected concentration and monitored continuously within the Raspberry Pi sensor acquisition loop.

E. Robotic Arms and Servo System

MG90S micro-servo motors serve two distinct mechanical roles in the system, both managed through the PCA9685 sixteenchannel twelve-bit PWM driver module communicating with the Raspberry Pi via the I²C protocol. The first role is steering actuation. A dedicated MG90S servo on each side of the chassis drives a spur gear train that physically pivots the wheel assemblies to the commanded steering angle, translating navigation heading correction commands from the Raspberry Pi into physical directional control of the rover. These steering servos are addressed as the first channel group within the PCA9685 output and are commanded continuously by the navigation controller in response to obstacle avoidance and waypoint heading corrections. The second role is robotic arm actuation. To extend the system's capability beyond passive surveillance, robotic arms are mounted on both sides of the chassis in two configurations. The single-servo configuration provides basic rotational motion suitable for simple response actions, while the double-servo configuration offers an additional degree of freedom enabling more complex movements. The MG90S micro-servo motors used in both configurations provide a torque of 1.8 kg·cm at 4.8V and 2.2 kg·cm at 6V with an operating speed of 0.1 seconds per 60 degrees. These robotic arm servos are addressed as the second channel group within the PCA9685 output and are commanded independently of the steering servo group based on alert state rather than navigation state. The robotic arms execute predefined motion sequences in response to detected threats identified by the computer vision module or environmental hazard alerts generated by the MQ-2 sensor, with actuation commands generated directly by the Raspberry Pi as part of its overall system coordination and decision-making responsibilities. Because the two channel groups within the PCA9685 are addressed and commanded independently, steering corrections issued during active obstacle avoidance never interfere with arm servo positions, and arm actuation commands triggered by threat or hazard events never affect the active steering angle.

F. Power Supply System

The Raspberry Pi 4B is powered independently via USB from a dedicated power bank, providing a clean and isolated supply to the primary controller. The drive motors, motor drivers, and all remaining onboard electronic components are powered by an elevenpoint-one volt three-cell lithium polymer battery with a capacity of five thousand two hundred milliampere hours. Lithium polymer batteries are selected due to their high energy density, lightweight construction, and ability to deliver the high discharge currents required by the drive motors and servo actuators. Voltage regulation is achieved through a single buck converter module that steps down the battery voltage to a stable five-volt supply rail. The ESP32 microcontroller and all onboard sensors and peripherals, including the MPU6050 inertial measurement unit, HC-SR04 ultrasonic sensor, MQ-2 smoke and gas sensor, SG90 scanning servo, and PCA9685 PWM driver, are powered from this five-volt regulated rail via their respective VIN or power supply pins. The DC motors are powered directly from the lithium polymer battery through the TB6612FNG motor driver modules, allowing them to utilize the full available voltage for improved torque performance. Separate power paths are maintained for actuators and control electronics to prevent electrical noise and transient voltage spikes generated by motors from interfering with sensitive digital components.

III. NAVIGATION ALGORITHMS AND PERCEPTION METHODOLOGY

The proposed navigation and perception framework enables autonomous indoor operation by combining visual-inertial motion estimation, continuous SLAM-based mapping and localization, sensor-fusion-based obstacle avoidance, computer vision-based threat detection, and environmental hazard monitoring. Rather than relying on computationally demanding infrastructure-dependent systems or expensive LiDAR-based approaches, the system adopts a lightweight hybrid strategy that provides practical performance within the computational constraints of embedded hardware.

A. Visual-Inertial Odometry for Motion Estimation

The primary navigation strategy relies on visual-inertial odometry to estimate the robot’s position and orientation continuously during indoor operation. This approach fuses visual information obtained from the USB webcam with inertial measurements from the MPU6050 IMU to overcome the limitations of each individual sensing modality. Visual odometry is implemented by detecting and tracking distinctive visual features across consecutive image frames captured by the camera. Feature extraction uses oriented FAST

corner detection combined with BRIEF descriptors, while Hamming-distance-based matching identifies correspondences across frames: $d_H(f_1, f_2) = \sum_{i=1}^{n} f_1(i) \oplus f_2(i)$

where f_1 and f_2 are binary descriptors, $\oplus$ denotes the XOR operation, and n is the descriptor length. Relative robot motion is estimated by computing the rigid body transformation between consecutive frames: $T_k = [R_k \ t_k ; 0 \ 1] \in SE(3)$

where R_k represents the rotation matrix and t_k represents the translation vector. Since the system uses a monocular camera, absolute depth is not directly measurable; instead, relative depth and scale-consistent spatial structure are inferred through feature triangulation within the monocular-inertial SLAM framework. Inertial measurements from the MPU6050 provide high-frequency angular velocity and linear acceleration data, which are integrated to estimate motion dynamics between frames: $p_{k+1} = p_k + v_k\Delta t + \frac{1}{2}(R_k a_k - g)\Delta t^2$

where p_k is position, v_k is velocity, R_k is orientation, a_k is accelerometer measurement, g is gravitational acceleration, and Δt is the IMU sampling interval. The fusion of visual and inertial data is performed using an Extended Kalman Filter operating in prediction and update stages:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k)$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \quad \hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - H_k \hat{x}_{k|k-1})$$

where $\hat{x}$ is the state estimate, P is covariance, F_k and H_k are Jacobians, Q_k and R_k are noise covariances, and K_k is the Kalman gain. This fusion ensures observability of scale and improves robustness by leveraging the complementary characteristics of visual and inertial measurements within the SE(3) motion estimation space. The fused pose estimate is continuously published within the ROS 2 framework, providing real-time localization to the SLAM, navigation, and decision-making modules. B. SLAM-Based Continuous Mapping and Localization

The SLAM module operates as a continuous background process that consumes fused pose estimates from the visual-inertial odometry system and maintains a persistent map of the environment. The environment is represented as a pose graph in which each node corresponds to a robot pose and each edge represents a spatial constraint between poses. At each iteration, the current pose is inserted into the graph, followed by feature matching against previously stored keyframes to identify potential loop closures. Global consistency of the map is maintained through pose graph optimization, which minimizes the total error across all constraints:

$$F^* = \arg\min \sum\nolimits_{(i,j)\in E} \| h(x_i, x_j) - z_{ij} \|^2_{\Sigma_{ij}}$$

where x represents robot poses, z_{ij} represents measured relative transformations, $h(x_i, x_j)$ is the predicted transformation, Σ_{ij} is the information matrix, and E is the set of edges. Loop closure detection is performed by computing descriptor similarity scores:

$$S(q, d) = \left(\sum_i w_i \, v_i^{\wedge}q \, v_i^{\wedge}d\right) / \left(\sqrt{\sum_i (v_i^{\wedge}q)^2} \cdot \sqrt{\sum_i (v_i^{\wedge}d)^2} \right)$$

where v_i denotes descriptor components and w_i represents feature weights. When the similarity score exceeds a predefined threshold, a loop closure is confirmed and global optimization is triggered, reducing accumulated drift across the map. The SLAM pipeline operates continuously and remains uninterrupted across all behavioral states, including obstacle avoidance, threat detection, and hazard response, ensuring that localization and mapping remain active at all times. The module outputs a continuously refined pose estimate and an updated map that directly support navigation and decision-making.

C. Sensor-Fusion-Based Obstacle Avoidance

Real-time obstacle avoidance relies on a continuously fused distance field that combines geometric information derived from camera frames with the angular distance readings produced by the scanning HC-SR04 ultrasonic sensor, both of which are acquired and processed directly on the Raspberry Pi. The ultrasonic sensor provides direct distance measurements through active ranging, while the vision system contributes relative depth and scale-consistent spatial structure inferred through feature triangulation within the visual-inertial SLAM pipeline. Since the system employs a monocular camera, absolute depth is not directly measured; instead, the camera provides relative geometric structure derived from motion and feature correspondence. The fusion logic compares the two distance estimates for a given direction and determines

their reliability based on a disagreement threshold computed as the absolute difference between camera-derived and ultrasonic distance measurements: $\Delta d = | d_{ultrasonic} - d_{camera} |$

An important temporal constraint governs the angular coverage provided by the scanning HC-SR04 system. The SG90 servo advances step by step across the forward arc, pausing at each angular position while the HC-SR04 performs an ultrasonic ranging cycle consisting of pulse transmission and echo reception. Each measurement requires approximately 60 milliseconds at the maximum effective range. For a sweep arc of 120 degrees with a 5-degree angular step, a complete sweep cycle comprises 24 measurement positions and takes approximately 1.4 to 1.5 seconds to complete. During this interval, the robot continues moving forward at its cruise velocity. At a representative cruise speed of 0.2 metres per second, the robot travels approximately 28 to 30 centimetres between the start and end of a single sweep. Objects entering the robot's path between sweep cycles, or located at angular positions not yet sampled, may not appear in the proximity field until the sweep reaches their corresponding angle. To mitigate this limitation, the cruise velocity is constrained such that the forward distance travelled during one sweep cycle remains below the minimum reliable obstacle detection distance. Specifically, the cruise velocity v_{cruise} is bounded such that: $v_{cruise} \times t_{sweep} < d_{min\_detection}$

where t_{sweep} is the sweep duration and $d_{min\_detection}$ is the effective detection threshold required for safe avoidance. Additionally, the vision-based geometric channel provides supplementary forward coverage between ultrasonic measurements, partially compensating for temporal gaps in angular sensing.

The fused distance estimate is determined using a condition-based arbitration strategy. When the disagreement Δd is below a predefined threshold δ_{thresh}, both sensing modalities are considered reliable and combined using weighted averaging with weights w_1 and w_2 such that $w_1 + w_2 = 1$. When the disagreement exceeds δ_{thresh}, the system selects the more reliable sensing modality based on environmental conditions. Ultrasonic sensing remains robust under low-light and textureless conditions but may degrade on smooth, curved, or acoustically reflective surfaces. Vision-based estimation performs reliably in well-lit, textured environments but degrades under poor lighting or featureless scenes. By exploiting this complementary behavior, the system ensures that the most physically appropriate sensing modality is used at any given time, resulting in a robust fused distance field.

When an obstacle is detected in the fused distance field, the navigation controller computes a heading correction using a proportional-derivative control law: $\theta_{correction} = K_p \cdot \theta_{error} + K_d \cdot \dot{\theta}_{error}$

where K_p and K_d are the proportional and derivative gains, θ_{error} is the angular deviation toward the safer direction, and $\dot{\theta}_{error}$ is the rate of change of the heading error. This correction is converted into a steering command: $\delta_{steering} = \arctan(\theta_{correction})$

The Raspberry Pi transmits $\delta_{steering}$ as a servo command to the MG90S steering servo via the PCA9685 PWM driver over the I²C interface. The PCA9685 generates the corresponding PWM signal, causing the servo to rotate the spur gear mechanism and physically pivot the wheel assemblies to the commanded angle. The rover steers left when an obstacle is detected on the right, steers right when detected on the left, and steers toward the clearer direction when an obstacle appears directly ahead. All six DC motors continue operating simultaneously at the computed forward velocity: $v_{linear} = v_{cruise} \cdot (1 - \delta_{obs} / d_{max})$

where v_{cruise} is the nominal velocity, δ_{obs} represents obstacle proximity, and d_{max} is the maximum sensing range. As obstacle proximity increases, forward velocity decreases proportionally while steering correction is applied, ensuring safe and smooth avoidance without halting motion. Once the obstacle is cleared, the steering angle returns to neutral and the rover resumes its nominal trajectory based on the SLAM-generated path. This non-halting navigation strategy enables continuous motion while maintaining collision-free operation.

The forward velocity command is transmitted to the ESP32 over UART as a structured control message. The ESP32 operates as a dedicated real-time motor control unit, translating this command into PWM duty cycles applied uniformly across all three TB6612FNG motor driver modules, ensuring consistent propulsion across all six wheels. In parallel, the Raspberry Pi continuously monitors the MQ-2 gas sensor within its acquisition loop. When the sensor reading exceeds the predefined hazard threshold, an emergency halt command is immediately transmitted to the ESP32 via UART. Upon receiving this command, the ESP32 drives the standby pins of all motor drivers low, halting all motor activity at the hardware level. The hazard

event is simultaneously logged by the alert management module for system-level monitoring and external notification.

D. Computer Vision-Based Threat Detection

The computer vision module enables intelligent surveillance by processing continuous image streams from the onboard camera to detect relevant objects and events. Image frames are preprocessed using OpenCV for noise reduction and normalization, after which a lightweight MobileNet-SSD v2 model deployed via TensorFlow Lite performs object detection. The model generates bounding boxes, class labels, and confidence scores, which are filtered using predefined thresholds to reduce false positives. Detected objects are evaluated within contextual constraints to improve reliability during long-term operation. Upon detection of a potential threat, an alert is generated and propagated to the navigation and response modules. This integration transforms the system from a passive monitoring platform into an active surveillance agent capable of real-time perception and response.

E. Environmental Hazard Detection and Response

The MQ-2 smoke and gas sensor provides continuous environmental monitoring capabilities beyond conventional visual surveillance, enabling detection of combustible gases and smoke within indoor environments. The sensor is interfaced directly with the Raspberry Pi and produces an analog output proportional to the detected gas concentration, which is continuously sampled within the Raspberry Pi sensor acquisition loop. To ensure stable and reliable operation, the sensor undergoes an initial burn-in period of 24 to 48 hours to stabilize its sensing characteristics, followed by a session-level warm-up and baseline calibration phase prior to deployment. During calibration, the system measures ambient conditions to establish baseline statistical parameters, allowing robust differentiation between normal environmental variations and hazardous conditions.

The hazard detection threshold is dynamically defined relative to the baseline as:

$$V_{threshold} = V_{baseline} + k \cdot \sigma_{baseline}$$

where $V_{baseline}$ represents the mean sensor output under normal atmospheric conditions, $\sigma_{baseline}$ denotes the standard deviation capturing ambient fluctuations, and k is an empirically tuned sensitivity coefficient used to balance detection responsiveness against false alarm rates. During operation, each real-time analog reading V_{MQ2} sampled by the Raspberry Pi is continuously compared against $V_{threshold}$, and a hazard condition is immediately detected when $V_{MQ2} \geq V_{threshold}$. This continuous sample-by-sample evaluation ensures minimal detection latency and avoids delays associated with periodic polling mechanisms.

Upon detection of a hazardous condition, the Raspberry Pi immediately transmits an emergency halt command to the ESP32 via the UART serial interface. The ESP32, operating as a dedicated real-time motor control unit, responds by driving the standby pins of all three TB6612FNG motor drivers low, thereby halting all motor activity at the hardware level. This hardware-level intervention ensures rapid and reliable system shutdown independent of higher-level software execution delays. Simultaneously, the Raspberry Pi propagates the hazard alert through the ROS-based alert management module, recording contextual information such as timestamp, sensor readings, and estimated robot pose for system-level logging and potential external notification. This coordinated response mechanism ensures safe, deterministic, and timely handling of hazardous environmental conditions.

F. Navigation and Decision-Making Integration

The navigation and decision-making module integrates information from the visual-inertial odometry system, SLAM-based localization, sensor-fusion-based obstacle perception, and computer vision–based threat detection to determine the rover's behavior in real time. Patrol routes are defined as ordered sequences of target waypoints derived from the SLAM-generated map. The rover navigates toward each successive waypoint by computing the required heading correction relative to the current pose estimate and converting this correction into a steering angle command transmitted to the MG90S steering servo via the PCA9685 PWM driver over the I²C interface. In parallel, the forward drive velocity command is transmitted to the ESP32 over UART, where it is converted into PWM signals for the motor drivers. This separation of steering

and propulsion ensures stable and continuous motion control. Obstacle avoidance is integrated seamlessly into the navigation process through continuous modulation of both steering and velocity. When an obstruction is detected in the fused distance field, the heading correction is dynamically adjusted toward the safer direction, the steering servo physically pivots the wheel assemblies accordingly, and the forward velocity is proportionally reduced based on obstacle proximity. This results in a non-halting navigation strategy in which the rover maintains forward motion while continuously adapting its trajectory, eliminating the need for discrete stop-and-replan cycles. Upon reaching a waypoint within a predefined acceptance radius, the navigation module transitions to the next target, and the steering angle is updated to align with the new waypoint direction.

The decision-making framework also incorporates higher-level responses to threat detection and environmental hazards. When a potential threat is identified by the computer vision module, the system transitions to a response state that may include reducing forward velocity, commanding the steering servo to a neutral position to stabilize the platform, activating the robotic arm servos through the PCA9685 secondary channel group, or issuing alerts via the ROS 2 alert management module. In the event of a hazardous gas condition detected by the MQ-2 sensor, the Raspberry Pi immediately transmits an emergency halt command to the ESP32, which disables all motor drivers at the hardware level to ensure rapid system shutdown. Simultaneously, the steering servo is returned to the neutral position, and the robotic arm servos execute a predefined hazard response sequence. This rule-based decision-making approach ensures deterministic, safe, and verifiable system behavior while maintaining computational efficiency and reliability on resource-constrained embedded hardware.

IV. SOFTWARE ARCHITECTURE

The software architecture of the autonomous indoor surveillance robot is organized as a modular framework based on the Robot Operating System 2 (ROS 2), running on the Raspberry Pi 4 Model B. Each major system functionality is implemented as an independent ROS node that communicates through standardized topics, services, and message interfaces. This modular design ensures clear separation of concerns, facilitates debugging and testing at the component level, and supports scalability and future system extensions without requiring structural redesign.

The sensor interface layer is responsible for acquiring raw data from all onboard sensors, including the USB webcam, MPU6050 inertial measurement unit, MQ-2 gas sensor, HC-SR04 ultrasonic sensor, and SG90 scanning servo. All sensors are interfaced directly with the Raspberry Pi. Camera data is captured and published as a continuous image stream, which is consumed in parallel by both the visual-inertial odometry module and the computer vision–based threat detection module. Inertial data from the MPU6050 is acquired via the I²C interface, filtered to compensate for noise and bias, and published as standardized IMU messages. Ultrasonic distance measurements from the HC-SR04 sensor are obtained via GPIO, tagged with their corresponding SG90 servo angle, and published as a directional proximity fan representing the local obstacle field. MQ-2 gas sensor readings are continuously sampled within the sensor acquisition loop and evaluated against a dynamically defined hazard threshold.

Upon threshold exceedance, the Raspberry Pi immediately transmits an emergency halt command to the ESP32 over the UART serial interface. The ESP32, acting as a dedicated real-time control unit, responds by disabling all PWM outputs to the TB6612FNG motor drivers, thereby stopping all motor activity at the hardware level.

The visual-inertial odometry module processes synchronized camera and IMU data to generate a continuous real-time estimate of the robot's pose. This pose estimate is consumed by the SLAM module, which maintains a globally consistent map and provides refined localization. The sensor-fusion-based obstacle avoidance module processes both the proximity fan from the ultrasonic sensor and vision-derived geometric information to construct a fused obstacle distance field for navigation.

The navigation and decision-making module integrates outputs from localization, mapping, obstacle perception, and threat detection to generate motion commands. Two independent command streams are produced. The first is a forward velocity command transmitted to the ESP32 via UART as a structured control message, which is converted into PWM duty cycles applied uniformly across all three TB6612FNG motor driver modules, driving the six DC motors. The second is a steering angle command transmitted

directly from the Raspberry Pi to the PCA9685 PWM driver via the I²C interface. The PCA9685 generates the corresponding PWM signal to actuate the MG90S steering servo, which drives the spur gear mechanism to physically pivot the wheel assemblies to the commanded angle.

The Raspberry Pi also manages a second, independent channel group on the PCA9685 for robotic arm actuation using MG90S servos, enabling execution of predefined response sequences triggered by threat detection or hazard events. Additionally, the SG90 scanning servo is driven directly via a dedicated GPIO PWM signal for continuous ultrasonic scanning.

This architecture enforces a strict separation between propulsion and steering control pathways, ensuring that steering adjustments during obstacle avoidance do not affect motor drive consistency, and that robotic arm actuation does not interfere with navigation control. This decoupled design is critical for maintaining stable, continuous, and non-halting navigation behavior.

The computer vision and threat detection module operates as a separate ROS node, processing the camera image stream in real time using OpenCV-based preprocessing and lightweight machine learning models optimized for embedded execution. Detected events are published as alert messages and handled by the alert management module, which logs contextual information including timestamp, sensor readings, and the current SLAM-estimated robot pose. This centralized alert handling mechanism ensures consistent system-level monitoring and facilitates integration with external notification systems.

V. EXPERIMENTAL VALIDATION

To evaluate the performance of the proposed autonomous surveillance framework, systematic testing was conducted at both component and integrated system levels. The testing methodology was designed to verify that the system meets its intended objectives and operates reliably under representative indoor conditions.

A. Navigation and Localization Performance

Navigation testing evaluated the performance of the visual-inertial odometry system in estimating robot position and orientation during indoor movement. The robot was commanded to follow predefined motion patterns including straight-line movement, turning maneuvers, and closed-loop paths within an indoor environment. The visual-inertial odometry module continuously estimated the robot's trajectory based on fused camera and inertial sensor data. Based on the component specifications and reference literature on similar low-cost camera and IMU configurations, the expected navigation accuracy of the proposed system is within fifteen to twenty centimeters under favorable indoor conditions with adequate lighting and sufficient visual texture. The SLAM module successfully maintained a globally consistent map throughout navigation tasks including periods of active obstacle avoidance, demonstrating the effectiveness of the non-halting SLAM strategy.

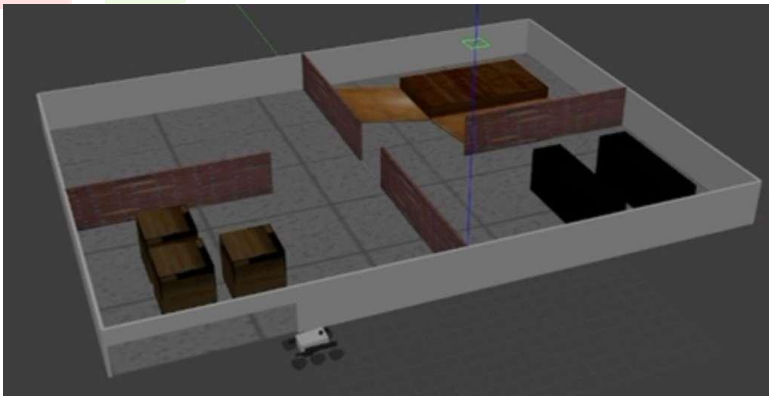

Fig. 2: Visualization of SLAM-generated map during autonomous navigation

B. Obstacle Detection and Avoidance Performance

Obstacle detection and avoidance testing evaluated the rover's ability to detect and safely navigate around objects using the scanning ultrasonic sensor combined with camera-derived depth information. Various obstacles including walls, furniture, and stationary objects were placed within the rover's operating environment. The rover was allowed to patrol autonomously while the obstacle sensor fusion module

processed continuous scanning data in real time. Testing confirmed that obstacles are detected within the effective range of the ultrasonic sensor and that avoidance is achieved without halting the rover at any point. Upon obstacle detection, the navigation controller computes a heading correction and transmits the corresponding steering angle command to the MG90S steering servo via the PCA9685 PWM driver over I²C. The PCA9685 generates the PWM signal that drives the MG90S servo to rotate the spur gear train, physically pivoting the wheel assemblies toward the clearer direction while all six DC drive motors continue operating at the simultaneously reduced forward velocity. The rover steers left when an obstacle is detected on the right, steers right when detected on the left, and steers toward the clearer direction when an obstacle appears directly ahead. Once the obstacle is cleared and the fused distance field confirms the forward path is unobstructed, the steering angle command returns to neutral and cruise velocity resumes. The HC-SR04 ultrasonic sensor provides reliable distance measurements within an effective range of two to four meters. The scanning SG90 servo provides angular coverage across the forward arc, producing a twodimensional proximity fan that supplements camera-derived depth data. The sensor-fusion disagreement threshold logic successfully selected the more reliable sensor in scenarios involving reflective surfaces and varying lighting conditions. Successful collision-free navigation through obstacle-laden corridors and laboratory spaces was demonstrated during integrated system testing.

C. Computer Vision and Threat Detection Performance

Testing of the computer vision module focused on verifying the system's ability to process visual data in real time and identify predefined threat categories under normal indoor lighting conditions. The system's response to visual input was observed to confirm detections are generated with appropriate confidence and without excessive latency. Based on existing research and similar embedded implementations using lightweight vision models on Raspberry Pi-class hardware, the expected object detection accuracy exceeds eighty-five percent for trained categories under favorable conditions. The latency between visual detection and alert generation was confirmed to be within acceptable bounds for real-time surveillance response. The confidence-threshold filtering strategy effectively suppressed false positives during extended monitoring tests.

D. Environmental Hazard Detection Performance

Environmental hazard detection testing evaluated the MQ-2 smoke and gas sensor's responsiveness to simulated hazardous conditions. Sensor output was monitored to confirm that readings exceed predefined thresholds when hazardous conditions are present and that the emergency stop is triggered immediately upon threshold detection. The hazard response was confirmed to halt all motor activity rapidly upon the Raspberry Pi detecting threshold crossing and transmitting the emergency halt command to the ESP32 over the UART serial link. Alert logging with contextual data was verified through the ROS alert management module. The system reliably distinguished hazardous conditions from normal background readings across multiple test exposures.

E. Robotic Arm Response Testing

Robotic arm testing verified correct actuation and response behavior for both single-servo and double-servo configurations. Predefined motion sequences were executed repeatedly to evaluate positioning accuracy, repeatability, and mechanical stability. Servo motors responded correctly to commands issued by the PCA9685 driver under Raspberry Pi control via the I²C interface, with smooth and accurate positioning across the available range of motion. Robotic arms successfully executed response motions upon trigger commands without interfering with robot mobility or sensor operation.

F. Power System and Endurance Testing

Power system testing evaluated voltage stability under different load conditions including idle operation, continuous movement, and combined actuation scenarios involving all drive motors and robotic arm servos simultaneously. The regulated five-volt supply rail maintained stable output within acceptable margins across all tested load conditions, while the power bank sustained consistent voltage supply to the Raspberry Pi throughout extended operation. Based on measured current consumption of individual subsystems and the nominal capacity of the five thousand two hundred milliampere-hour lithium polymer battery, the

estimated operational duration is sufficient to support extended indoor patrol sessions. No unexpected voltage drops, component resets, or performance degradation were observed during combined load testing.

G. Comparison with Existing Surveillance Approaches

Traditional fixed-camera surveillance systems are limited to static viewpoints and require continuous human monitoring. The proposed autonomous surveillance robot overcomes these limitations through dynamic coverage, real-time threat response, and continuous environmental monitoring. Compared to surveillance systems based purely on wheel encoder odometry, the visualinertial odometry approach eliminates cumulative drift errors caused by wheel slippage and mechanical tolerances. Compared to LiDAR-based systems, the proposed framework achieves competitive localization accuracy at significantly lower hardware cost, demonstrating that reliable indoor navigation can be achieved using a standard USB webcam and a low-cost IMU when combined with appropriate sensor fusion and SLAM techniques. The hardware-level gas sensor emergency stop provides a safety response capability not commonly found in camera-only surveillance platforms, extending the system's utility to environmental safety monitoring applications.

VI. APPLICATIONS AND DEPLOYMENT CONTEXTS

The proposed autonomous surveillance framework is designed to support intelligent security monitoring in environments where GPS signals are unavailable or unreliable. Such conditions are common in indoor facilities, underground structures, and complex built environments where traditional satellite-based localization cannot be used. The ability to navigate, perceive, and respond using
onboard sensing and computation makes the system suitable for a variety of operational scenarios requiring reliable indoor mobility and continuous surveillance. In defense and security contexts, autonomous mobile robots can be deployed for routine surveillance and patrol operations within secured installations. The robot can follow predefined or dynamically generated inspection routes through corridors, rooms, and restricted areas while continuously monitoring the environment for unauthorized presence or suspicious activity. The system can also assist in reconnaissance missions where hazardous or access-restricted locations must be observed before human personnel can safely enter. Emergency response and disaster management represent another critical application domain. Following natural disasters or industrial accidents, indoor environments may become unsafe due to structural damage, toxic gases, or fire hazards. The integrated smoke and gas detection capability enables the surveillance robot to assist in such scenarios by detecting hazardous atmospheric conditions and providing early warning before human responders enter the affected area. Autonomous navigation allows the robot to explore damaged indoor spaces and collect environmental data while keeping human personnel at a safe distance. Industrial and commercial inspection represents a further deployment context. Autonomous robots can execute periodic inspection routes in warehouses, power plants, transportation hubs, and industrial plants, detecting anomalies, monitoring environmental conditions, and collecting operational data while navigating indoor spaces that may not be easily accessible or safe for continuous human occupation. Beyond operational deployment, the modular ROS-based architecture makes the system a flexible research and development platform for studying autonomous navigation, sensor fusion, computer vision, and embedded robotics algorithms. New perception, localization, or planning modules can be integrated and evaluated without major modifications to the overall system structure, providing a scalable foundation for future research in intelligent autonomous systems.

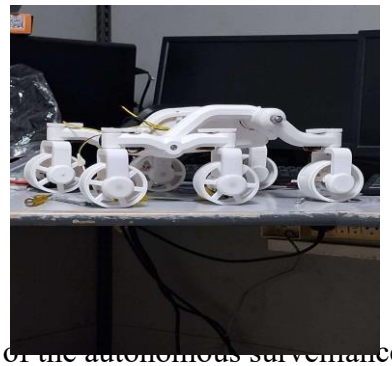

Fig. 3: Advanced assembly stage of the autonomous surveillance robot platform showing onboard computing unit, sensor.

VII. Real World Deployment And System Validation

The proposed navigation and surveillance framework was deployed on the developed physical mobile robot platform for evaluation under real-world indoor operating conditions. The deployment process involved system integration, sensor calibration, controller tuning, and progressive functional validation to ensure reliable autonomous operation. Accurate sensor calibration was performed prior to experimentation. The MPU6050 IMU was calibrated to compensate for sensor bias and noise by collecting stationary data and computing offset corrections that minimize drift in accelerometer and gyroscope readings. Camera calibration was carried out using reference patterns to determine intrinsic parameters including focal length, optical center, and lens distortion coefficients, enabling accurate image rectification for feature extraction and visual odometry. The SG90 servo carrying the HCSR04 ultrasonic sensor was calibrated to establish the relationship between servo angle commands and actual measurement direction, ensuring accurate angular coverage mapping. The MG90S steering servos were calibrated to establish the precise relationship between commanded PWM pulse width and the resulting physical wheel assembly pivot angle, ensuring that heading correction commands from the navigation controller produce accurate and repeatable steering responses through the spur gear mechanism. Neutral steering position — corresponding to straight-ahead wheel alignment — was verified and recorded as the reference zero-angle command. The MQ-2 gas sensor, after undergoing an initial 24–48 hour burn-in period, was allowed to warm up for a minimum of five minutes before baseline calibration under normal atmospheric conditions, with hazard detection thresholds set relative to the established baseline. Controller parameters initially developed and tuned for expected operating conditions were experimentally refined to account for real-world effects including floor friction, motor nonlinearities, actuator delays, surface irregularities, and steering servo gear backlash. Velocity control parameters were adjusted to achieve smooth motion and stable trajectory tracking. Steering servo gain parameters were tuned to achieve responsive yet stable heading correction without overshoot during obstacle avoidance maneuvers. Sensor fusion gains for the obstacle avoidance disagreement threshold logic were refined based on observed behavior across different indoor surface types and lighting conditions

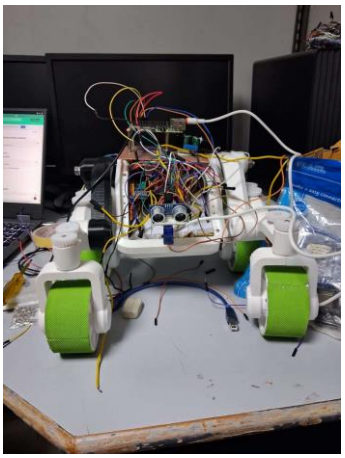

Fig. 4: Fully assembled autonomous surveillance and security robot

VIII. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

This work presented an integrated framework for autonomous indoor surveillance and security using a mobile robotic platform designed for GPS-denied environments. The implemented system combines visual-inertial odometry with continuous SLAM-based localization and mapping, sensor-fusion-based non-halting obstacle avoidance, computer vision-based threat detection, MQ-2 smoke and gas sensing for environmental hazard monitoring, and robotic arm actuation for physical response. The complete system is implemented within a ROS-based software framework deployed on a dual-controller embedded architecture consisting of a Raspberry Pi 4 Model B and an ESP32 microcontroller. The proposed framework was validated through component-level and integrated system testing conducted in real indoor environments. The robot successfully demonstrated autonomous multi-waypoint patrol navigation with real-time obstacle avoidance achieved through physical servo-driven wheel steering — where the MG90S steering servo and spur gear mechanism pivot the wheel assemblies to the commanded angle in direct response to the fused obstacle distance field — without halting forward motion at any point during avoidance. The system also demonstrated detection of environmental hazards with hardware-level emergency motor stop, identification of potential security threats through real-time computer vision processing using a MobileNet-SSD v2 model deployed via TensorFlow Lite, and execution of physical response actions through robotic arm actuation via the second PCA9685 channel group. Navigation accuracy is expected within fifteen to twenty centimetres for the monocular-inertial ORB-SLAM3 configuration under favorable indoor conditions, consistent with published literature on comparable low-cost sensor hardware. This figure is a literature-derived estimate and has not been independently confirmed through direct experimental measurement on this system. Real-time performance of the combined navigation, SLAM, and vision inference pipeline is demonstrated to be achievable within the tested operating conditions on a Raspberry Pi 4B running Ubuntu 24.04 LTS with the ROS 2 Jazzy Jalisco framework, though it is acknowledged that simultaneous peak load across all modules can cause frame rate reduction in the vision pipeline and that the system is tuned for moderate patrol speeds accordingly. The implementation identifies several performance constraints that bound the operational envelope: the ORBSLAM3 tracking quality is dependent on visual texture availability and degrades in featureless environments; the HC-SR04 sweep cycle introduces a temporal blind spot whose extent is bounded by enforcing a maximum cruise velocity relative to the sweep rate; the MQ-2 sensor requires a 24-to-48-hour initial burn-in and session-level baseline recalibration to maintain consistent threshold behavior; and camera-derived depth estimates become unreliable during near-stationary and pure-rotation maneuvers, during which the IMU integration branch provides short-term pose continuity at the cost of accumulated drift. Future experimental work should address quantitative position error benchmarking, holdout-set accuracy evaluation of the vision model, and characterization of CPU utilization and pipeline latency under worst-case concurrent load conditions.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics, MIT Press, Cambridge, MA, USA, 2005.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2nd ed., MIT Press, Cambridge, MA, USA, 2011.

[3] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," IEEE Robotics and Automation Magazine, vol. 18, no. 4, pp. 80-92, Dec. 2011.

[4] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On Manifold Preintegration for Real-Time Visual-Inertial Odometry," IEEE Transactions on Robotics, vol. 33, no. 1, pp. 1-21, Feb. 2017.

[5] M. Quigley et al., "ROS: An Open-Source Robot Operating System," in Proc. IEEE Int. Conf. on Robotics and Automation (ICRA) Workshop, Kobe, Japan, 2009.

[6] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[7] H. Durrant-Whyte and T. Bailey, "Simultaneous Localization and Mapping: Part I," IEEE Robotics and Automation Magazine, vol. 13, no. 2, pp. 99-110, June 2006.

[8] T. Bailey and H. Durrant-Whyte, "Simultaneous Localization and Mapping (SLAM): Part II," IEEE Robotics and Automation Magazine, vol. 13, no. 3, pp. 108-117, Sept. 2006.

[9] D. Fox, W. Burgard, and S. Thrun, "The Dynamic Window Approach to Obstacle Avoidance," IEEE Robotics and Automation Magazine, vol. 4, no. 1, pp. 23-33, 1997.

[10] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," Numerische Mathematik, vol. 1, no. 1, pp. 269-271, 1959.

[11] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Technical Report, University of North Carolina at Chapel Hill, 2006.

[12] K. P. Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, Cambridge, MA, USA, 2012.

[13] J. Borenstein, H. R. Everett, and L. Feng, Where Am I? Sensors and Methods for Mobile Robot Positioning, University of Michigan, 1996.

[14] Y. Wang, D. Mulvaney, I. Sillitoe, and E. Swere, "Robot Navigation by Waypoints," Journal of Intelligent Robotic Systems, vol. 52, pp. 175-207, 2008.

[15] F. A. Moreno et al., "Automatic Waypoint Generation to Improve Robot Navigation Through Narrow Spaces," Sensors, vol. 20, no. 1, p. 240, 2019.

[16] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion Planning and Control for Mobile Robot Navigation: A Survey," arXiv preprint arXiv:2011.13112, 2021.

[17] Open Robotics, ROS 2 Navigation Framework, 2024. Retrieved from https://docs.ros.org

[18] Raspberry Pi Foundation, Raspberry Pi 4 Model B Technical Specifications, 2024. Retrieved from https://www.raspberrypi.com

[19] InvenSense, MPU-6050 Product Specification, Revision 3.4, 2013.

[20] STMicroelectronics / Toshiba, TB6612FNG Motor Driver Datasheet, 2014.

[21] NXP Semiconductors, PCA9685 16-Channel, 12-Bit PWM Fm+ I2C-Bus LED Controller Datasheet, 2015.

[22] L. Liu, X. Chen, S. Zhu, and P. Tan, "Path Planning Techniques for Mobile Robots: Review and Challenges in Autonomous Navigation," Robotics and Computer-Integrated Manufacturing, vol. 79, pp. 102-126, 2023.

[23] A. Raj et al., "Deep Q-Network Approach for Intelligent Mobile Robot Navigation," Nature Scientific Reports, 2024. [24] M. T. Tariq, C. Wang, and Y. Hussain, "Robust Mobile Robot Path Planning via LLM-Based Dynamic Waypoint Generation," arXiv preprint arXiv:2501.15901, 2025.

[25] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," Proceedings of IEEE International Conference on Robotics and Automation, pp. 500-505, 1985.