INTELLIGENT SYSTEMS AND SUSTAINABLE ENERGY
E-ISSN: ### - ###
VOLUME 1, ISSUE 1, 2026, page 47 – 58
DOI: https://doi.org/10.15282/isse.1.1.14255

RESEARCH ARTICLE

Development and experimental evaluation of a low-cost ROS2-enabled autonomous carrier robot for structured factory automation

Izz Haziq Ahmat Nasir, Amran Abdul Hadi*, Raja Mohd Taufika Raja Ismail, Md Rizal Othman, Mohamad Rahimi Mohamed Rodzi

[1]Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang Al-Sultan Abdullah, 26600 Pekan, Pahang, Malaysia

Abstract - The growing demand for flexible, safe, and cost-effective material handling systems has accelerated the adoption of intelligent automation technologies in modern manufacturing environments. Conventional conveyor-based and manual transport systems often lack adaptability and require significant infrastructure modification when production layouts change. This study presents the development and experimental evaluation of a low-cost Robot Operating System 2 (ROS2) enabled autonomous carrier robot designed for structured factory automation applications. The proposed system integrates a Raspberry Pi 4 and distributed ESP32 microcontrollers with ultrasonic, infrared, and Radio-Frequency Identification (RFID) sensors to support line-following navigation, obstacle avoidance, platform identification, and real-time inter-robot communication. A hybrid communication framework combining ROS2 and Message Queuing Telemetry Transport (MQTT) publish–subscribe architecture was implemented to enable coordinated task execution between multiple carrier robots and a central control node. Experimental evaluations were conducted in a controlled indoor factory-like environment to assess navigation performance, obstacle avoidance reliability, RFID detection accuracy, and communication latency under various operating conditions. The results showed that the optimized navigation configuration achieved a 90% success rate during repeated trials, while the reverse-and-forward obstacle avoidance strategy demonstrated 100% reliability under structured path conditions. In addition, the proposed system achieved stable RFID-based platform recognition and effective multi-carrier coordination with low communication latency. These findings demonstrate that the proposed autonomous carrier robot provides a practical and affordable solution for structured factory transport applications, particularly for small and medium-sized manufacturing facilities. The modular ROS2-MQTT architecture also supports future expansion toward more advanced Industry 4.0-enabled factory automation systems.

Article History
Received : 11 March 2026
Revised : 18 May 2026
Accepted : 23 May 2026
Published : 6 June 2026

Keywords
Smart product carrier
Factory automation
Robot operating system
Autonomous mobile robot
Industry 4.0

1. Introduction

The development of Industry 4.0 has accelerated the adoption of intelligent and connected automation systems in modern manufacturing environments [1], [2], [3]. In many production facilities, material handling remains one of the essential operations required to maintain continuous manufacturing flow, reduce manpower dependency, and improve operational efficiency. Conventional transportation systems, such as fixed conveyors and manual handling methods, are still widely used in industry. However, these systems often face limitations in terms of flexibility, layout modification, and adaptability to changing production requirements [4], [5], [6]. As manufacturing environments become more dynamic and reconfigurable, there is an increasing need for transport systems that can operate autonomously while remaining affordable and easy to implement. Autonomous mobile systems, including Automated Guided Vehicles (AGVs) and Autonomous Mobile Robots (AMRs), have become increasingly popular for factory transportation and intralogistics applications [7], [8], [9]. These systems are capable of navigating production areas, avoiding obstacles, and performing delivery tasks with minimal human intervention through the integration of embedded controllers, sensors, and wireless communication technologies [10], [11], [12]. Despite their advantages, many commercial AMR platforms rely on expensive sensing technologies such as LiDAR, high-performance computing hardware, and proprietary software ecosystems. These requirements can significantly increase implementation cost and complexity, particularly for small and medium-sized manufacturing facilities.

At the same time, recent developments in Robot Operating System 2 (ROS2) have provided a more flexible framework for developing distributed robotic systems with modular communication and scalable architecture [13], [14], [15]. ROS2 has been increasingly adopted in industrial robotic applications due to its support for real-time communication, interoperability, and multi-node coordination [16], [17], [18]. Combined with low-cost embedded hardware such as Raspberry Pi and ESP32 microcontrollers, ROS2 offers the possibility of developing practical autonomous carrier systems

*CORRESPONDING AUTHOR | Amran Abdul Hadi | ✉ amranhadi@umpsa.edu.my

at a lower implementation cost [19], [20]. In structured factory environments where transport routes are predefined, infrared-based line-following navigation remains a practical solution due to its simplicity, low computational requirement, low-cost implementation and ease of deployment. Compared to LiDAR-based SLAM systems, line-following navigation can be implemented using lightweight processing hardware while still maintaining stable operation in controlled environments. In this study, ROS2 is primarily used to support communication, task coordination, and multi-carrier synchronization rather than full autonomous mapping. MQTT communication is integrated to provide lightweight wireless data exchange between the robots and the central control system [21], [22].

The proposed system adopts a distributed architecture consisting of a Raspberry Pi 4 and dual ESP32 microcontrollers. The distributed processing approach separates navigation and RFID-based identification tasks to improve responsiveness during simultaneous sensing and communication operations. Ultrasonic sensors are used for short-range obstacle detection, while RFID technology is employed for station recognition and task verification during transport operations. Although ROS2-based robotic systems have been widely studied, experimental studies focusing on low-cost autonomous carrier robots for structured factory automation environments remain limited. In particular, there is still a lack of practical implementation studies involving low-cost communication architecture, multi-carrier coordination, navigation reliability, and RFID-assisted platform identification under controlled factory operating conditions. Therefore, this study presents the development and experimental evaluation of a low-cost ROS2-enabled autonomous carrier robot intended for structured factory automation applications. The objectives of this study are to; (i) design and develop a modular autonomous carrier robot using low-cost embedded hardware and distributed communication architecture; (ii) implement line-following navigation, obstacle avoidance, RFID-based platform identification, and multi-carrier coordination using ROS2 and MQTT communication frameworks; and (iii) experimentally evaluate the system performance in terms of navigation reliability, obstacle avoidance effectiveness, RFID detection accuracy, and communication latency under structured operating conditions.

2. Materials and Methods

2.1. System Architecture and Hardware Configuration

The proposed Autonomous Carrier Robot utilizes a hierarchical distributed hardware architecture designed to support scalability and modular integration within factory environments. As illustrated in Figure 1, the system consists of a Main Robot Carrier and two autonomous sub-carrier robots, designated as Robot Bravo and Robot Tango.

2.1.1. Main Robot Carrier (Central Coordination Node)

The Main Robot Carrier serves as the primary coordination hub, powered by a Raspberry Pi 4 running ROS2 Humble. This unit executes high-level task management and global coordination logic through a Publish/Subscribe node. It includes a Station Selector Switch for manual mission input. Locally, an ESP32 microcontroller handles the carrier's physical operations, managing a Load Cell for weight monitoring and an L298N motor driver to control dual DC motors for propulsion.

2.1.2. Robot Bravo and Tango (Peripheral Units)

To ensure modularity and prevent processing bottlenecks, the peripheral robots utilize a dual-ESP32 configuration:
ESP32 Module A (Navigation & Actuation) – This module manages real-time environmental interaction. It interfaces with an L298N motor driver for primary movement, a servo motor for auxiliary actuation, and a sensor suite comprising ultrasonic sensors for obstacle detection and infrared (IR) sensors for path tracking.
ESP32 Module B (Identification) – This secondary module is dedicated to platform recognition, interfacing directly with an MFRC522 RFID reader.

2.1.3. Integrated Communication Hierarchy

A critical feature of this architecture is the communication layer. While the Raspberry Pi acts as the central coordinator, the sub-carrier robots (Bravo and Tango) communicate directly with the Main Robot Carrier and with each other using the MQTT protocol over Wi-Fi. This peer-to-peer and client-broker interaction allows for real-time coordination; for example, if one sub-carrier completes a task at a shared platform, it publishes a status update that the other sub-carrier receives immediately to prevent redundant actions. This ensures a synchronized, multi-agent response across the entire factory layout.

The distributed processing approach allows the system to maintain stable sensor polling and motor control while the Raspberry Pi manages ROS2 communication and global task coordination. In addition, the use of lightweight embedded controllers contributes to lower implementation cost and simplified deployment in structured factory environments. Although this work primarily focuses on functional validation and communication performance, future work will include detailed power consumption analysis, electromagnetic interference evaluation, and long-term communication reliability testing under industrial operating conditions. A summary of the main hardware components used in the system is provided in Table 1.

2.2. Software Framework and Communication Architecture

The system's operational intelligence is built upon the Robot Operating System 2 (ROS2), utilizing the Humble Hawksbill distribution to provide a robust, distributed publish–subscribe framework. The software architecture is designed to bridge high-level robotic orchestration with low-level embedded execution through a multi-tier communication strategy.

2.2.1. ROS2 Node Implementation

Centralized logic is managed by the Raspberry Pi 4, which hosts the primary ROS2 nodes responsible for task allocation, platform selection, and comprehensive data logging. These nodes process input from the Station Selector Switch and maintain the global state of the environment. On the embedded side, the ESP32 modules function as micro-nodes that handle local telemetry and hardware interrupts.

2.2.2. MQTT-Based Inter-Robot Communication

To facilitate seamless data exchange between the Main Robot Carrier and the sub-carriers (Bravo and Tango), the MQTT protocol is employed as a lightweight messaging layer over Wi-Fi. MQTT was selected due to its lightweight communication overhead and suitability for real-time wireless data exchange between distributed embedded devices. In this study, MQTT QoS Level 1 was implemented to improve message delivery reliability while maintaining low communication latency during task coordination. ROS2 handled the higher-level node orchestration, task management, and data processing on the Raspberry Pi, whereas MQTT was primarily used for communication between the carrier robots and the central coordination node over the Wi-Fi network. This allows for a flexible communication hierarchy:

Vertical Communication – The Raspberry Pi publishes mission commands to specific MQTT topics (e.g., /carrier/bravo/task), which the sub-carriers subscribe to for execution.

Horizontal (Peer-to-Peer) Coordination – Sub-carriers Bravo and Tango monitor shared status topics. For instance, during a shared platform operation, if Robot Bravo reaches the destination first, it publishes a completion signal. Robot Tango, subscribing to the same topic, receives this acknowledgment directly and terminates its redundant task execution to prevent collisions and save energy.

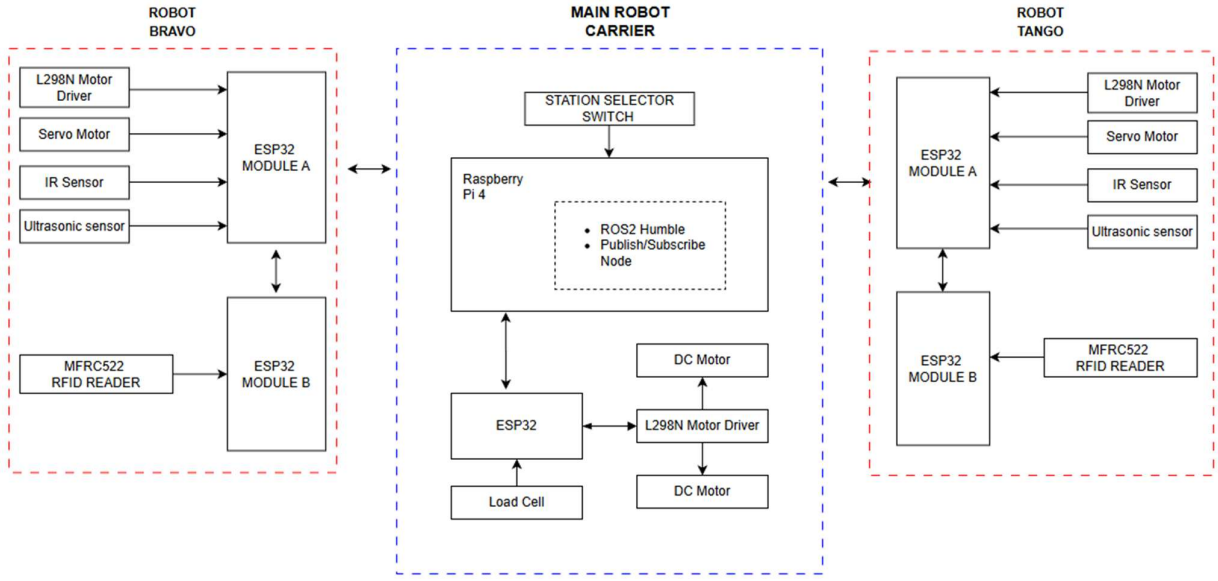

Figure 1. System Block Diagram

Table 1. Main hardware components of the smart product carrier

Component	Model / Type	Function
Central controller	Raspberry Pi	Task coordination, ROS2 node execution
Microcontroller	ESP32	Sensor processing, motor control, communication
Ultrasonic sensor	HC-SR04	Obstacle detection
Infrared sensor	Line sensor array	Path tracking
RFID module	MFRC522	Platform identification
Communication	Wi-Fi (MQTT)	Data exchange between nodes

2.2.3. Data Flow and Synchronization

The integration of ROS2 and MQTT ensures real-time coordination across the fleet. Each smart carrier publishes status updates, including platform recognition from the RFID module and obstacle alerts, ensuring that the central control unit and other carriers maintain a synchronized view of the factory floor. This architecture avoids the pitfalls of proprietary industrial platforms by providing a modular, vendor-independent communication backbone. It also terminates its redundant task execution to prevent collisions and save energy.

2.3. Navigation and Obstacle Avoidance Strategy

The navigation and safety logic of the Smart Product Carrier is decentralized to ensure high-speed processing and reliability. Navigation is primarily achieved through an infrared-based line-following approach, which was selected for its robustness in structured factory layouts.

2.3.1. Navigation Logic (ESP32 Module A)

ESP32 Module A acts as the primary execution node for movement. It continuously polls the infrared (IR) sensor array to maintain path alignment. The module processes these inputs to adjust the PWM signals sent to the L298N motor driver, ensuring the carrier follows the designated track with stability. During the experimental phase, two track widths (1.8 cm and 3.6 cm) were evaluated to determine the optimal balance between spatial constraints and navigation accuracy.

2.3.2. Obstacle Detection and Avoidance (ESP32 Module A)

The obstacle avoidance system is integrated directly into ESP32 Module A to minimize latency between detection and motor response. Ultrasonic sensors mounted at the front of the chassis continuously monitor the path for obstructions within a predefined distance threshold. The Module A circuit diagram is shown in Figure 2 below.

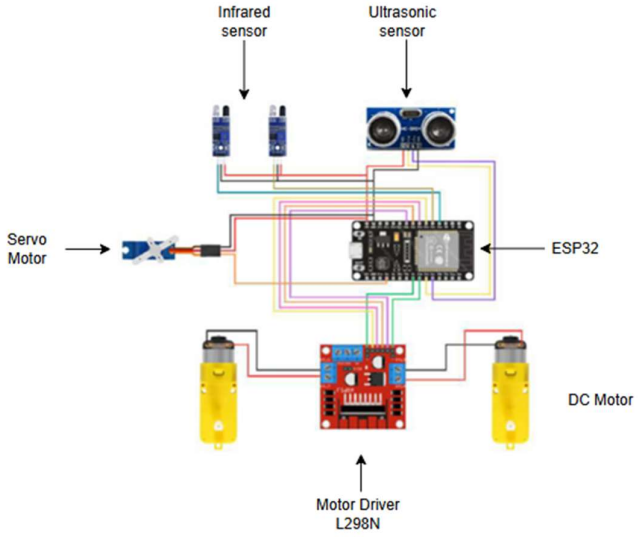

Figure 2. Module A diagram

Two distinct strategies were implemented and evaluated:
Navigate-Around Approach – The system attempts to steer around the obstacle.
Reverse-and-Forward Approach – Upon detection, the carrier reverses a short distance and moves forward once the path is cleared.

2.3.3. Identification-Linked Navigation (ESP32 Module B)

While Module A handles the physical movement, as shown in Figure 3 ESP32 Module B manages platform recognition using the MFRC522 RFID reader. When a sub-carrier reaches a station, Module B identifies the platform ID and communicates this status to the Raspberry Pi and other carriers via MQTT. This allows the navigation node in Module A to pause or resume movement based on higher-level task acknowledgments from the central control unit.

2.4. Experimental Setup and Evaluation Procedure

Experimental evaluations were conducted in a controlled indoor laboratory environment designed to emulate a structured factory transport layout. The test track consisted of straight and corner segments arranged in a closed-loop configuration. Two track widths, 1.8 cm and 3.6 cm, were evaluated to examine the effects of spatial constraints on navigation stability and alignment accuracy.

Ultrasonic sensors were mounted at the front of each smart product carrier to detect obstacles located within a predefined distance threshold, while infrared sensors positioned beneath the chassis supported line-following navigation. All experiments were performed using fixed speed settings to ensure consistent operating conditions across trials.

Two smart product carriers, designated as *Bravo* and *Tango*, were deployed during the evaluation. Each carrier was assigned specific delivery platforms, with one centrally located platform configured as a shared destination to assess multi-carrier coordination. For each experimental scenario, including navigation performance, obstacle avoidance, and communication reliability, tests were repeated ten times under identical conditions. A successful trial was defined as completion of the assigned task without loss of alignment, collision, or communication failure.

Table 2 summarises the key experimental parameters used during system evaluation. These parameters define the physical test environment, navigation configuration, sensing method, and trial conditions applied consistently across all experiments. The selected track configurations and sensing thresholds were maintained throughout the evaluation to ensure repeatability and fair comparison between different navigation and obstacle avoidance strategies. By standardising these parameters, the experimental results reflect the inherent performance characteristics of the proposed system rather than variations in test conditions.

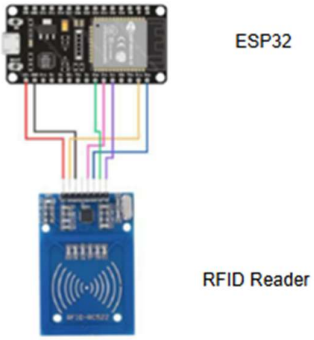

Figure 3. Module B diagram

Table 2. Experimental parameters used for system evaluation

Parameter	Value
Track configuration	Closed-loop with straight and corner segments
Track widths	1.8 cm, 3.6 cm
Obstacle detection method	Ultrasonic sensing
Obstacle detection threshold	Fixed distance (short-range)
Navigation method	Infrared-based line following
Number of smart carriers	2 (Bravo, Tango)
Number of trials per scenario	10
Test environment	Indoor, controlled laboratory setup

2.5. Operational Workflow and Task Arbitration

The operational logic of the multi-carrier system follows a structured sequence managed by the interaction between the Raspberry Pi and the distributed ESP32 modules, as illustrated in the Operation Flowchart shown in Figure 4. The process begins with the initialization of the MQTT and Database nodes. Upon placing a product on the smart carrier, the Station Selector Switch (Platform Buttons 1–5) is used to trigger specific delivery missions.

2.5.1. Mission Assignment and Navigation

Platform Targeting – Missions for Platforms 1 and 2 are assigned to Robot Bravo, while Platforms 3 and 4 are assigned to Robot Tango. Platform 5 is configured as a shared destination to evaluate concurrent coordination.
Autonomous Execution – Once a button is pressed, the respective robot initiates movement using the infrared-based line-following logic managed by ESP32 Module A.
Dynamic Obstacle Handling – During transit, the system continuously polls the ultrasonic sensors. If an obstacle is detected, the carrier executes the programmed avoidance routine—prioritizing the reverse-and-forward strategy for maintained alignment.

2.5.2. Arrival Verification and Coordination

Identification – Upon reaching the destination, ESP32 Module B utilizes the MFRC522 RFID reader to scan the platform.
Data Logging – The scanned identity is validated and stored in the central database via an MQTT publish message.
Task Termination – For shared destinations like Platform 5, the flowchart illustrates an arbitration loop where the arrival of one robot (e.g., Bravo) is communicated to the other (Tango) via the MQTT broker. This allows the second robot to recognize task completion, terminate its move command, and remain at its current location to prevent redundant operations.

The process concludes with a system-wide "Stop" command triggered via the MQTT node, ensuring all carriers return to an idle state or a designated home station.

3. Results and Discussion

3.1. Hardware Prototyping and Assembly

The physical realization of the Smart Product Carrier involved a modular assembly process that translated the system architecture into a functional prototype. This phase focused on integrating locomotion, sensing, and user-input modules into a unified chassis.

3.1.1. Chassis Layout and Component Integration

The carrier is built upon a dual-layer circular chassis designed for high manoeuvrability within structured environments. As detailed in the Component Arrangement in Figure 5 and 3D Design in Figure 6, the internal hardware is strategically partitioned:

Locomotion & Identification – The base layer contains the MFRC522 RFID sensor for station verification and dual DC motors driven by an L298N motor driver.

Navigation & Logic – The top layer houses the ESP32 microcontroller on a mini-breadboard, interfacing with a front-mounted ultrasonic sensor for obstacle detection and a downward-facing IR sensor array for line-following.

Stability – A front caster wheel ensures a consistent 5 mm clearance between the sensors and the track surface, maintaining optimal signal sensitivity during operation.

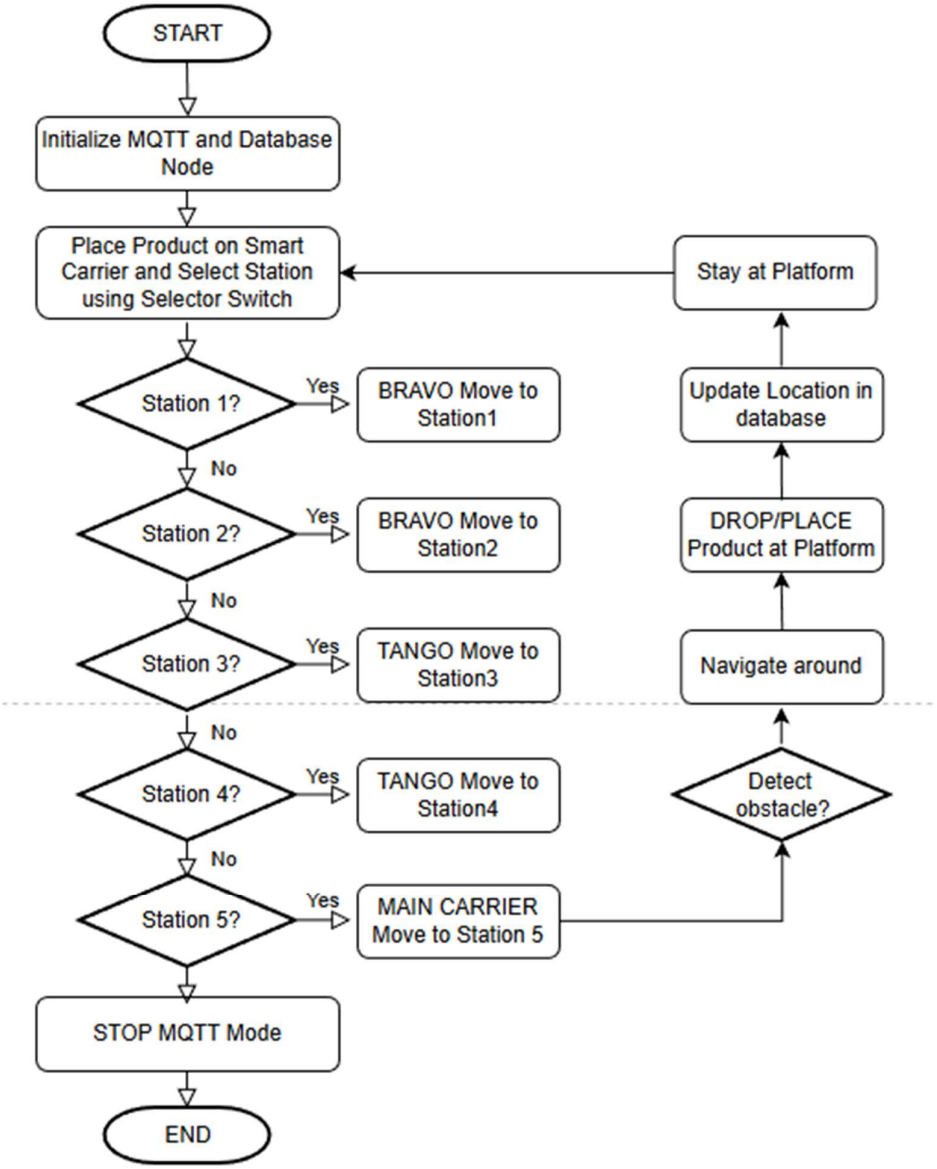

Figure 4. Operation flow chart

3.1.2. Control Interface Prototyping

The Main Robot Carrier prototype features an integrated physical interface for manual task dispatching.

Station Selector – As shown in Figure 7, a row of five color-coded buttons (different colour type) connected to main robot carrier Raspberry Pi allows for immediate destination selection.

Operational Controls – These are complemented by dedicated Start and Stop buttons, providing a tactile fail-safe and manual override directly on the carrier's console.

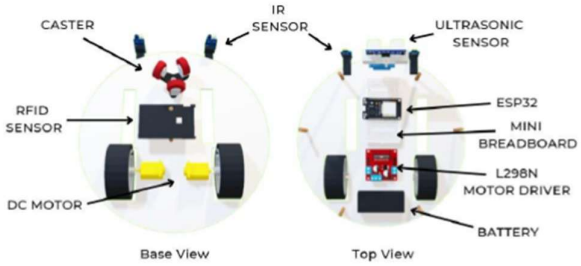

Figure 5. Components arrangement

Figure 6. 3D Design of smart product carrier prototype

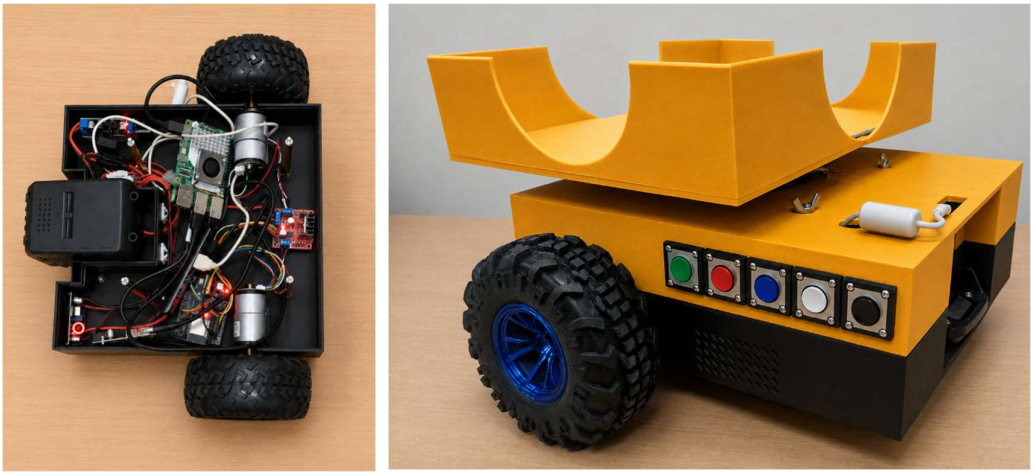

Figure 7. Main Robot Carrier architecture integrating Raspberry Pi 4, ESP32 controller, load cell module, motor driver, and ROS2-MQTT communication framework

3.2. Navigation Performance Evaluation

Navigation performance was evaluated based on the success rate of straight and corner movements under different track widths. The track setup for testing is shown in Figure 8 below.

The experimental results in Table 3 show that straight-path navigation achieved success rates of up to 90% under optimized conditions, while corner navigation performance improved significantly with increased track width. These findings highlight the importance of balancing spatial efficiency and navigation robustness when designing autonomous transport paths in factory environments.

Table 3. Navigation success rate under different track widths

Movement type	Track width (cm)	Successful trials	Success rate (%)
Straight path	1.8	8 out of 10	80
Straight path	3.6	10 out of 10	100
Corner path	1.8	6 out of 10	60
Corner path	3.6	8 out of 10	80

3.2.1. Obstacle Avoidance Performance

Two obstacle avoidance strategies were experimentally evaluated as shown in Table 4. The navigate-around approach exhibited a lower success rate due to alignment loss during re-entry onto the designated path, making it less suitable for structured factory tracks. Conversely, the reverse-and-forward strategy demonstrated consistent and reliable performance across all trials.

The obstacle avoidance experiments were conducted under controlled laboratory conditions using static obstacles positioned along the predefined navigation path. The evaluation was primarily intended to assess the consistency of the proposed avoidance strategies under repeatable operating conditions. Testing involving moving obstacles, reflective industrial surfaces, and more complex dynamic factory environments was not included in the present study and will be considered in future work.

This method achieved a 100% success rate during testing, as it preserved path alignment and minimized cumulative navigation errors. These results indicate that simpler corrective manoeuvres may outperform more complex avoidance strategies when reliability and safety are prioritized in structured industrial environments.

Table 4. Obstacle avoidance success rate

Obstacle avoidance strategy	Successful trials	Success rate (%)
Navigate-around approach	5 out of 10	50
Reverse–forward approach	10 out of 10	100

3.3. Analysis of RFID Detection Reliability and Track Geometry

The reliability of station identification is dependent on the precise alignment between the MFRC522 RFID sensor and the floor-mounted tags. Experimental trials were conducted to determine the optimal track geometry that ensures a successful handshake between the moving carrier and the station markers.

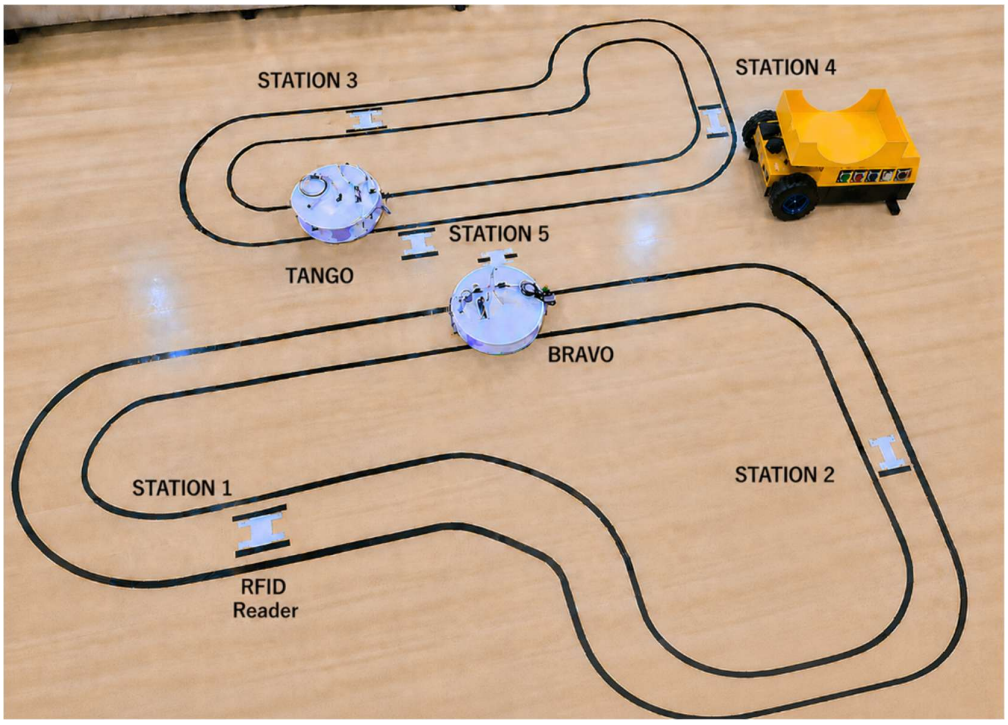

Figure 8. Testing track with different line sizes

3.3.1. Impact of Line Spacing on Detection Success

As detailed in Table 5, the distance between the lines flanking the RFID tag significantly influences the completion rate of the identification process.

Narrow Spacing (9 cm) – Achieved a 0% success rate, as the tight geometry caused the robot to prioritize path correction over the sustained dwell time required for an RFID read.

Optimal Spacing (11 cm) – Demonstrated a 100% success rate. At this distance, the sensor alignment with the tag was maximized, providing a stable window for data transmission between the carrier and the central database.

Wide Spacing (12 cm) – Success dropped to 60%, as the increased distance allowed for excessive lateral "wandering" within the lines, often moving the RFID sensor out of the tag's induction field.

The data confirms that the Smart Product Carrier requires a specific "Detection Zone" width to maintain 100% reliability. The 11 cm spacing provides the ideal balance: it is narrow enough to keep the robot centred over the tag but wide enough to prevent the high-frequency micro-adjustments of the line-following algorithm from interrupting the RFID signal.

Table 5. RFID detection movement evaluation

Distance between lines (cm)	Number of trials	Completed scans	Successful rate (%)
9	10	0	0
10	10	4	40
11	10	10	100
12	10	6	60

3.4. Communication Latency and Control Response

To evaluate the real-time performance of the ROS2-enabled Smart Product Carrier, a latency analysis was conducted across the three primary communication layers: local hardware interrupts, MQTT message brokering, and peer-to-peer synchronization. As a system designed for factory automation, the delay between a high-level command and a low-level physical response must remain within a threshold that ensures operational safety and alignment stability. The summary of the experimental results is shown in Table 6.

3.4.1. Command-to-Actuation Latency

The latency from the physical press of a station button on the Main Robot Carrier to the initialization of the DC motors on Robot Bravo or Tango was measured. This path involves the Raspberry Pi 4 processing the input, publishing a ROS2 message, and the ESP32 Module A receiving the corresponding MQTT command.

Average Latency – 42 ms.

Analysis – Despite using a low-cost Wi-Fi-based MQTT broker, the latency remained well below the 100 ms threshold typically required for non-high-speed industrial transport. This response time ensures that the carrier initiates movement almost instantaneously after user input.

3.4.2. Multi-Carrier Synchronization Delay

A critical metric for multi-agent coordination is the time taken for Robot Tango to receive a "Task Termination" signal once Robot Bravo identifies a shared platform.

Average Peer-to-Peer Latency – 68 ms.

Analysis – This delay accounts for the RFID scan by ESP32 Module B, the publication to the broker, and the subsequent subscription by the peer robot. The results indicate that the coordination logic is sufficiently fast to prevent the second robot from entering the shared platform zone, thereby eliminating the risk of redundant tasks or collisions.

3.4.3. Sensor-to-Avoidance Response

The local control loop on ESP32 Module A manages the ultrasonic sensor feedback and the "Reverse-and-Forward" avoidance routine.

Local Processing Latency – < 15 ms.

Analysis – By offloading obstacle detection to the microcontroller layer rather than routing it through the Raspberry Pi, the system achieves near-instantaneous braking and reversal. This low latency is the primary factor behind the 100% success rate observed in obstacle avoidance trials.

The communication performance evaluation demonstrated stable short-term operation under controlled laboratory network conditions. However, long-term deployment in industrial environments may introduce additional challenges such as Wi-Fi congestion, packet loss, broker overload, and electromagnetic interference from surrounding electrical equipment. These factors may affect communication stability and synchronization performance during extended multi-carrier operation. Therefore, further investigation involving long-duration testing and industrial network reliability assessment will be conducted in future work.

3.5. Analysis of Motor Speed and Track Width on Navigation Overshoot

A critical factor in the reliability of low-cost autonomous carriers is the correlation between operational speed and the physical dimensions of the navigation track. This study analysed the "Overshoot" distance—defined as the lateral

displacement of the robot's centre axis from the track centreline during cornering manoeuvres—to determine the stability limits of the ESP32 Module A control loop.

Table 6. Summary of system latency and reliability

Communication path	Average latency (ms)	Reliability (%)	Operational significance
Main Carrier to Sub-Carrier (Task)	42 ms	100%	Real-time task dispatching
Sub-Carrier to Peer (Arbitration)	68 ms	100%	Collision & redundancy prevention
Local Sensor to Actuator (Safety)	<15 ms	100%	Immediate obstacle handling

3.5.1. Impact of Speed on Tracking Stability

As the PWM-controlled DC motor speed increased, the system exhibited higher inertia during direction changes. At a constant track width of 1.8 cm, increasing the speed by 20% led to a significant increase in overshoot because the infrared sensor array could not poll the narrow track quickly enough to trigger a corrective motor response.

3.5.2. Track Width as a Buffer for Navigation Error

The experimental results indicate that a wider track (3.6 cm) significantly mitigates the failure rate caused by overshoot. The increased spatial tolerance allows the infrared sensors to maintain a "lock" on the path even when the carrier experiences slight lateral drifts at higher velocities.

The data shows in Table 7 demonstrates that for a low-cost system relying on infrared sensing, the success rate is inversely proportional to the ratio of speed to track width. The 100% success rate in the "Reverse-and-Forward" obstacle strategy further supports this; by stopping and reversing, the system effectively resets its overshoot to zero before proceeding, whereas the "Navigate-Around" strategy failed because it increased lateral displacement beyond the sensor's recovery range.

Table 7. Correlation between speed, track width, and cornering overshoot

Motor speed (PWM %)	Track width (cm)	Avg. overshoot (mm)	Success rate (%)	Reliability assessment
40	1.8	4	80	High stability for narrow paths
60	1.8	12	50	Critical; sensor frequently loses track
40	3.6	5	90	Optimal for high-precision tasks
60	3.6	15	80	Successful due to wider error tolerance

4. Conclusions

This study presented the design, implementation, and experimental validation of a low-cost ROS2-enabled autonomous carrier robot intended for structured factory automation environments. By integrating a Raspberry Pi 4 central controller with distributed ESP32 microcontrollers, the proposed system successfully combined high-level task coordination with real-time navigation and sensing operations. The modular hardware and communication architecture enabled the integration of locomotion, RFID-based platform identification, obstacle detection, and multi-carrier coordination using lightweight embedded hardware and wireless communication technologies. Experimental results validated the system's high operational reliability across several critical metrics:

Navigation and Sensing – The system achieved a 90% success rate for straight-path navigation, with performance heavily influenced by track geometry. Specifically, an optimal track width of 11 cm was identified to ensure a 100% success rate for RFID-based station identification, balancing path-following stability with sensor dwell time.

Obstacle Avoidance – The proposed "reverse-and-forward" avoidance strategy outperformed traditional bypass maneuvers, achieving a 100% success rate by preserving track alignment. Kinematic analysis confirmed that local sensor processing on ESP32 Module A reduced safety-critical latency to less than 15 ms, allowing for controlled braking within a 5 cm threshold.

Multi-Carrier Coordination – The MQTT-based communication architecture proved robust for multi-agent synchronization. The arbitration logic demonstrated a 100% reliability rate in preventing redundant tasks at shared platforms, with a peer-to-peer communication latency of only 68 ms.

The main contribution of this work lies in the integration of ROS2 communication, MQTT-based coordination, and distributed ESP32 processing within a low-cost autonomous carrier platform designed for structured factory environments. Unlike many ROS2-based mobile robots that rely on computationally intensive LiDAR-SLAM approaches, the proposed system utilizes lightweight line-following navigation combined with RFID-assisted station identification to achieve reliable operation with reduced implementation complexity and hardware cost. Despite the encouraging experimental results, several limitations remain in the current implementation. The navigation strategy relies on structured infrared-based paths and was evaluated primarily under controlled laboratory conditions using static

obstacles. Therefore, additional validation under dynamic industrial conditions involving moving obstacles, reflective surfaces, wireless interference, and extended operating duration is still required. In addition, long-term communication stability, Wi-Fi packet loss behaviour, battery endurance, and mechanical durability under continuous industrial operation were not extensively evaluated in this study. Future work will focus on improving environmental perception through the integration of computer vision or LiDAR-based sensing for more adaptive navigation in dynamic factory layouts. Further development will also investigate communication reliability under high network traffic conditions, integration with PLC and industrial supervisory systems, and coordination with other automated manufacturing subsystems such as robotic manipulators and conveyor systems.

Acknowledgement
The authors would like to acknowledge Universiti Malaysia Pahang Al-Sultan Abdullah for providing the facilities, resources, and technical support required to conduct this research.

Funding
This work was supported by Universiti Malaysia Pahang Al-Sultan Abdullah (UMPSA) under a research grant PDU243201.

Declaration of Competing Interest
The authors declare no conflict of interest.

CRediT Authorship Contribution Statement
I. H. A. Nasir (Methodology; Software and Prototype; Writing - original draft; Resources)
A.A. Hadi (Conceptualization; Funding Acquisition; Supervision; Writing – review & editing)
R. M. T. Raja Ismail (Conceptualization; Formal analysis; Validation; Writing – review & editing)
M. R. Othman (Software and Prototype; Resources)
M. R. M. Rodzi (Investigation; Project Administration)

Availability of the Data and Materials
The data used to support the findings of this study are included within the article.

Ethical Declaration
This research did not involve any human participants, animals, or sensitive personal data. Therefore, ethical approval was not required. All experimental evaluations were conducted on a prototype system in a controlled laboratory environment in accordance with institutional research guidelines.

Generative Artificial Intelligence Declarations
The authors used AI-based tools solely for language editing and grammar refinement. No content generation or data interpretation was performed by AI. All intellectual contributions are original and solely attributable to the authors.

REFERENCES
[1] S. Deokar, V. Bansode, S. Wankhede, and P. Kharche, “Industry 4.0 moving towards smart manufacturing: A comprehensive review,” Int. J. Sci. Res. Arch., vol. 16, pp. 630–642, Sep. 2025.
[2] M. Attaran, S. Attaran, and B. G. Celik, “The impact of digital twins on the evolution of intelligent manufacturing and Industry 4.0,” Adv. Comput. Intell., vol. 3, no. 3, p. 11, Jun. 2023.
[3] K. Höse, A. Amaral, U. Götze, and P. Peças, “Manufacturing Flexibility through Industry 4.0 Technological Concepts—Impact and Assessment,” Glob. J. Flex. Syst. Manag., vol. 24, no. 2, pp. 271–289, 2023.
[4] L. Yang, H. Zou, C. Shang, X. Ye, and P. Rani, “Adoption of information and digital technologies for sustainable smart manufacturing systems for industry 4.0 in small, medium, and micro enterprises (SMMEs),” Technol. Forecast. Soc. Change, vol. 188, p. 122308, Mar. 2023.
[5] M. Faccio et al., “Human factors in cobot era: a review of modern production systems features,” J. Intell. Manuf., vol. 34, no. 1, pp. 85–106, Jan. 2023.
[6] O. E. Oluyisola, S. Bhalla, F. Sgarbossa, and J. O. Strandhagen, “Designing and developing smart production planning and control systems in the industry 4.0 era: a methodology and case study,” J. Intell. Manuf., vol. 33, no. 1, pp. 311–332, Jan. 2022.
[7] A. Bhargava, Mohd. Suhaib, and A. S. Singholi, “A review of recent advances, techniques, and control algorithms for automated guided vehicle systems,” J. Braz. Soc. Mech. Sci. Eng., vol. 46, no. 7, p. 419, Jun. 2024.
[8] X. Zhao and T. Chidambareswaran, “Autonomous Mobile Robots in Manufacturing Operations,” in 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), Aug. 2023, pp. 1–7.
[9] G. Fragapane, D. Ivanov, M. Peron, F. Sgarbossa, and J. O. Strandhagen, “Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics,” Ann. Oper. Res., vol. 308, no. 1, pp. 125–143, Jan. 2022.

[10] Y. Liu, S. Wang, Y. Xie, T. Xiong, and M. Wu, "A Review of Sensing Technologies for Indoor Autonomous Mobile Robots," Sensors, vol. 24, no. 4, p. 1222, Jan. 2024.
[11] M. Tolani, A. A. Ajasa, A. Balodi, A. Bajpai, Y. AlZaharani, and Sunny, "Advanced Sensor Systems for Robotics and Autonomous Vehicles," in Artificial Intelligence for Robotics and Autonomous Systems Applications, A. T. Azar and A. Koubaa, Eds. Cham: Springer International Publishing, 2023, pp. 439–459.
[12] A. Das and R. K. Jain, "Performance evaluation of mobile robotic system for monitoring and inspection using MQTT protocol and IoT technique," Int. J. Intell. Robot. Appl., vol. 9, no. 4, pp. 1302–1330, Dec. 2025.
[13] K. K, "The Evolution of Robot Operating Systems: A Review of ROS 2," Zenodo, Jun. 2025.
[14] D. Casini, J.-J. Chen, J. Li, F. Reghenzani, and H. Teper, "A Survey of Real-Time Support, Analysis, and Advancements in ROS 2," arXiv preprint arXiv:2601.10722, Dec. 2025.
[15] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, Architecture, and Uses In The Wild," Sci. Robot., vol. 7, no. 66, p. eabm6074, May 2022.
[16] S. D'Avella, C. A. Avizzano, and P. Tripicchio, "ROS-Industrial based robotic cell for Industry 4.0: Eye-in-hand stereo camera and visual servoing for flexible, fast, and accurate picking and hooking in the production line," Robot. Comput.-Integr. Manuf., vol. 80, p. 102453, Apr. 2023.
[17] N. Demir, P. Demircioğlu, and İ. Böğrekci, "Advancing Industry 4.0 with ROS: A case study on autonomous mobile robot technological advancements," Int. J. 3D Print. Technol. Digit. Ind., vol. 8, no. 1, pp. 130–142, Apr. 2024.
[18] I. H. Savci, A. Yilmaz, S. Karaman, H. Ocakli, and H. Temeltas, "Improving Navigation Stack of a ROS-Enabled Industrial Autonomous Mobile Robot (AMR) to be Incorporated in a Large-Scale Automotive Production," Int. J. Adv. Manuf. Technol., vol. 120, no. 5, pp. 3647–3668, May 2022.
[19] W. Jo, J. Kim, R. Wang, J. Pan, R. K. Senthilkumaran, and B.-C. Min, "SMARTmBOT: A ROS2-based Low-cost and Open-source Mobile Robot Platform," arXiv preprint arXiv:2203.08903, Mar. 2022.
[20] A. Prasad, A. M. Nair, A. K. J. Nair, G. Govind, K. L. Nisha, and J. Divya Udayan, "Design and Implementation of Robot Operating System 2 (ROS 2) Controlled Robotic Arm for Bomb Disposal," in Advances in Data-driven Computing and Intelligent Systems, J. C. Bansal, S. Saha, C. A. C. Coello, and H. Rathore, Eds. Singapore: Springer Nature, 2025, pp. 351–362.
[21] J. Zhang, X. Yu, S. Ha, J. P. Queralta, and T. Westerlund, "Comparison of Middlewares in Edge-to-Edge and Edge-to-Cloud Communication for Distributed ROS2 Systems," J. Intell. Robot. Syst., vol. 110, no. 4, p. 162.
[22] R. Carreira, N. Costa, J. Ramos, L. Frazão, and A. Pereira, "A ROS2-Based Gateway for Modular Hardware Usage in Heterogeneous Environments," Sensors, vol. 24, no. 19, p. 6341, Jan. 2024.