Journal of Mechatronics, Electrical Power, and Vehicular Technology 17 (2026) 118-127

Journal of Mechatronics, Electrical Power, and Vehicular Technology

e-ISSN: 2088-6985
p-ISSN: 2087-3379

mev.brin.go.id

Multi-LiDAR SLAM and intelligent navigation for autonomous logistic robots in ROS2 environments

Leonard Priyatna Rusli *, Michael Jonathan, Rusman Rusyadi

[a] *Mechatronics Study Program, Swiss German University*
Jalan Sutera Barat No.15, Tangerang, 15143, Indonesia

Abstract

Traditional automated guided vehicles (AGVs) are restricted by their reliance on predefined paths, limiting adaptability in dynamic warehouse environments. While autonomous mobile robots (AMRs) overcome this limitation through on-board simultaneous localization and mapping (SLAM) and autonomous navigation, standard configurations often suffer from top-mounted-sensor blind spots when loads are carried on the chassis. To address these coverage gaps, an indoor logistic AMR based on the robot operating system 2 (ROS2) was designed and evaluated. The platform was developed by combining a multi-LiDAR perception stack with low-cost industrial actuation and a lightweight, fleet-style user interface. Within the system architecture, data from two light detection and ranging (LiDAR) sensors were merged at the topic level into a single virtual scan for SLAM toolbox and Nav2. Additionally, actuation and wheel odometry were driven by an RS-485 Modbus-based brushless DC (BLDC) motor controller, while ultrasonic sensors for short-range safety, an inertial measurement unit (IMU) for orientation, and a network of microcontroller calling stations communicating via message queuing telemetry transport (MQTT) were integrated into the platform. Experimental validation demonstrated successful multi-LiDAR fusion, with the Modbus motor driver achieving a motion-control error of 0.36 % and a speed-retrieval error of 0.43 %. Furthermore, calling-station commands were reliably executed over MQTT, and a point-to-point navigational repeatability of 10.3 cm was achieved. These findings indicate that an integrated multi-LiDAR ROS2 AMR provides a highly practical solution for indoor logistics. Through the proposed sensor merger and calling-station handshake, two recurring vulnerabilities of standard ROS2 deployments—single-LiDAR coverage gaps and Nav2 goal-overwriting behavior—were successfully resolved.

Keywords: autonomous mobile robot; multi-LiDAR SLAM; MQTT calling-station fleet interface; ROS2 Nav2 navigation; RS-485 Modbus motor driver.

I. Introduction

Autonomous mobile robots (AMRs) and automated guided vehicles (AGVs) are widely used in industrial settings, primarily for material handling and internal logistics. Their flexibility, agility, and robust performance make them ideal for repetitive tasks such as transporting goods in warehouses. AMRs also enable seamless digital integration with just-in-time material delivery systems, significantly improving productivity and efficiency [1]. Despite the pandemic's impact in 2020, the mobile robot market grew by 20 %, with nearly 60,000 units shipped. While AGV revenues were higher than AMRs in 2020, AMRs are expected to surpass AGVs by 2025 due to their superior flexibility, scalability, and lower maintenance requirements [2]. Recent reviews of warehouse and logistics implementation confirm this trend and emphasise on-board intelligence and infrastructure-free decision-

* Corresponding Author. leonard.rusli@sgu.ac.id (L. P. Rusli)

https://doi.org/10.55981/j.mev.2026.1187

Received 14 May 2025; revised 19 May 2026; accepted 16 June 2026; available online 20 July 2026; published 31 July 2026

How to Cite: L. P. Rusli *et al.*, "Multi-LiDar SLAM and intelligent navigation for autonomous logistic robots in ROS2 environments," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 17, no. 1, pp. 118-127, July, 2026.

making as the defining features that distinguish modern AMRs from traditional AGVs [3]. This distinction matters most in warehouse settings, where the environment imposes constraints that fixed-path systems cope poorly with: narrow aisles between racking that demand tight cornering, tall racks that create blind spots above wheel-mounted sensors, and dynamic obstacles such as forklifts and workers crossing the route at unpredictable moments.

Unlike AGVs, AMRs feature intelligent navigation, eliminating the need for physical guidance systems such as magnetic tape or near field communications (NFC) tags [4]. They navigate autonomously using sensor configurations that include light detection and ranging (LiDAR), an odometer, and an inertial measurement unit (IMU) [5]. Additional sensors, such as cameras for object detection [6], further enhance navigation accuracy. While global positioning system (GPS) is commonly used for outdoor applications, AMRs can dynamically update their internal maps and adapt to environmental changes. Their ability to re-route in real time enables greater operational efficiency compared to AGVs, which rely on fixed paths. This adaptability allows AMRs to navigate complex environments without requiring infrastructure modifications.

Since AGVs only use low processing controllers, even microcontrollers follow the guide. They can only do simple tasks. Meanwhile, AMRs with higher processing power are commonly similar to a personal computer (PC). It means that they can do anything that a PC can do, resulting in a lot more options available, such as internet of things (IoT) applications. AGVs have high maintenance costs associated with their guidance system, especially for semi-outdoor applications. The initial cost of an AMR unit is higher than that of an AGV unit due to AMR's significantly higher processing power; however, the complete cost of an AGV solution is much higher compared to AMR because of the ongoing cost of maintaining the guidance system. The development of autonomous service robots requires robust evaluation of domestic component levels to ensure manufacturing feasibility and structural standardization [7].

Navigation is a fundamental challenge in autonomous robotics, requiring the robot to determine its location, destination, and optimal path. Sensor data enables AMRs to localize themselves and make real-time decisions based on their environment. In vision-based navigation, road edges can serve as vanishing points for path recognition [8]. Mobile robots represent their environment using geometric, topological, or semantic models [9], with geometric representations often utilizing principal component analysis (PCA) to extract spatial features. Building on these foundations, the robot operating system 2 (ROS 2) navigation framework (Nav2) has emerged as the real open-source stack that consolidates such capabilities, providing modular planners, controllers, behaviour trees, and recovery behaviours that are actively maintained by the ROS community [10]. To address the complex navigation challenges in global navigation satellite system (GNSS)-denied environments, ROS2 frameworks are increasingly used to facilitate reliable autonomous localization and spatial mapping [11] Centralized ROS2 architectures have proven highly effective in managing autonomous navigation and task coordination in the multi-robot collaborative workspaces [12]. Effective autonomous navigation relies a lot on optimal path planning; for instance, the integration of the A-star algorithm with artificial potential fields has proven successful in managing complex mobile robot formations and obstacle avoidance [13]. Furthermore, precise execution of generated trajectories requires advanced control strategies, shown by the application of non-linear model predictive control to ensure real-time navigation in autonomous vehicles or robots [14].

Inertial measurement units (IMUs), comprising accelerometers, gyroscopes, and magnetometers, provide motion data for localization. By integrating acceleration and angular velocity, they allow AMRs to estimate their position and heading at relatively low cost. LiDAR, in turn, has become the dominant exteroceptive sensor for indoor mapping and is commonly combined with IMU and wheel-odometry data through sensor-fusion algorithms to improve localization robustness. Accurate indoor mapping and obstacle avoidance for autonomous mobile robots require a careful selection of distance sensors, because performance significantly varies between LiDAR and structured-light cameras based on range and object transparency [15]. Recent benchmark of ROS 2 SLAM solutions shows that the SLAM toolbox and cartographer packages can produce accurate occupancy maps from 2D LiDAR data, with trade-offs in computational cost and loop-closure performance under real-world conditions [16]. The adequacy of core ROS2 algorithms, such as Nav2 and simultaneous localization and mapping (SLAM) toolbox, is always validated through high-fidelity extended reality simulations prior to physical deployment [17]. Deploying ROS2-based mobile robots equipped with advanced LiDAR configurations is done to ensure robust obstacle avoidance and safe operation in most unstructured environments [18]. Innovative adaptations of standard LiDAR sensors for 3D SLAM within ROS2 demonstrate significant improvements in

mapping precision and autonomous motion planning [19]. Taken together, these efforts share a recurring limitation: Most are optimized for a single, top-mounted LiDAR with a clean 360° view, which is not what a logistics AMR actually offers in practice. Once a cart or payload sits on top of the chassis, the standard single-LiDAR placement is occluded; mounting the LiDAR lower reintroduces blind spots at wheel level and at human standing height, and stock ROS2 SLAM packages do not handle multi-LiDAR inputs out of the box. The research gap this study targets is therefore not a new SLAM algorithm but a multi-LiDAR data-merging arrangement that restores full 2D coverage on a load-carrying AMR while keeping the downstream SLAM and Nav2 pipelines unchanged.

A prior study explored the development of a scaled-down self-driving car (SDC) as an AMR using ROS2 Foxy, integrating IMUs, wheel encoders, LiDAR, and real-time kinematic global navigation satellite system (RTK-GNSS) sensors for localization [20]. However, challenges arose due to the incomplete porting of ROS to ROS2 and limited documentation at the time. Building upon this research, the present study develops an industrial-grade AMR with practical implementation in mind [21]. It integrates ROS2 navigation packages with multi-LiDAR sensor fusion and incorporates calling stations to demonstrate dynamic navigation capabilities. Beyond system integration, the scientific contribution of this work is threefold: (i) a multi-LiDAR data-merging package that fuses two laterally offset 2D LiDARs into a single virtual/ scan compatible with stock SLAM toolbox and Nav2, removing the top-mounted-sensor assumption in most ROS2 pipelines; (ii) an RS-485 Modbus-based wheel-odometry and actuation that achieves low control error without proprietary firmware, making the platform reproducible with off-the-shelf industrial drives; and (iii) an message queuing telemetry transport (MQTT) calling-station handshake that prevents Nav2's default behaviour of overwriting an unfinished goal when a new request arrives, a failure mode that is otherwise common in multi-station industrial deployments. The MQTT-based calling-station network adopted in this work is in line with the emerging VDA 5050 interoperability standard, which transmits JSON command messages between AMRs and central master control over MQTT to coordinate heterogeneous robot fleets in industrial environments [22].

II. Materials and Methods

The mechanical design of the AMR being used for this research is relatively short and wide to lower the center of gravity. It has a cart carrier that could be pulled by the AMR. The structure of the AMR is made of hollow iron and will be covered by iron plates cover. The overall dimensions of the AMR are 873 x 659 x 289 mm. The load capacity is 100 kg. It has three pairs of wheels, with two pairs of them being castor wheels placed at the back and front of the AMR. The main actuator of the AMR is a pair of brushless direct current (BLDC) hub motors installed with a seesaw mechanism with the back castor wheels to support bumpy terrain. The motors have a total power of 500 watts controlled by a single RS-485-based driver. The locking mechanism is actuated by a NEMA 23 stepper motor with a peak current of 4.4 A. These mechanical choices have direct consequences for the perception stack. The wide, short footprint and lowered center of gravity reduce chassis pitch and roll during acceleration and braking, which limits IMU bias drift and avoids the scan-plane tilt that would otherwise degrade 2D LiDAR-based SLAM. Putting the cart carrier on top of the chassis, on the other hand, rules out a single top-mounted LiDAR; that constraint is what leads to the two-LiDAR side-mount configuration. The seesaw rear-castor mechanism also keeps the rear wheels in contact with the ground on uneven floors, stabilizing the wheel encoders that feed odometry to the SLAM stack.

A. Multiple LiDARs data merger

LiDAR is commonly used as the main sensor for simultaneous localization and mapping (SLAM) and navigation purposes. It is used mainly for indoor applications due to its robustness and lower processing cost compared to the vision-based sensor. Common ROS2 SLAM package such as the SLAM Toolbox for educational ROS-based robot also use LiDAR as their main sensors. The LiDAR is usually mounted on the top of the robot, resulting in a clean three hundred sixty degrees viewing angle for the LiDAR without anything blocking the laser. The main contribution is therefore a constrained design integration rather than a new SLAM algorithm: given the load-carrying chassis described above, two side-mounted 2D LiDARs are combined at the topic level into a single virtual scan that stock SLAM toolbox and Nav2 nodes can consume without modification. The merge itself is lightweight; the work is in choosing where the LiDARs sit, what to filter out, and how to express the transform between them.

In logistic robots, this approach could be irrelevant. To simplify the kinematics and feedback mechanism, the loads are usually placed on the top of the robot. Either directly held by the robot's body or any kind of locking mechanism with an additional wheeled carrier

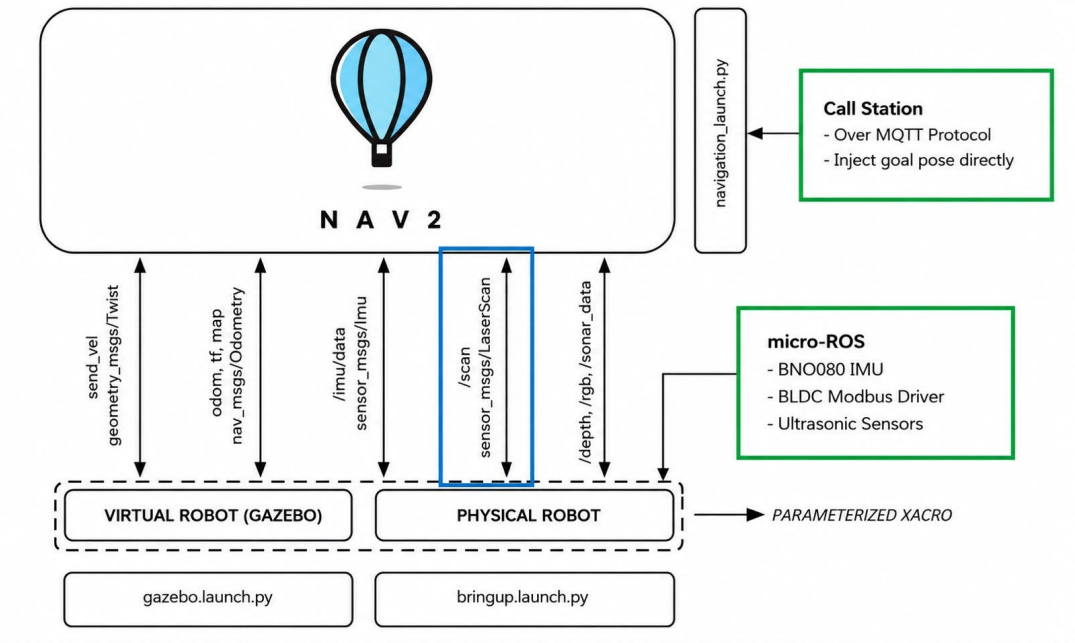

Figure 1. ROS2 software architecture.

cart. By having those arrangements, most likely placing the LiDAR on top of the robot is not possible. A common solution to this problem is using more than one LiDAR placed at a specific arrangement to provide a three-hundred-sixty-degree viewing angle. However, common SLAM packages do not support multi LiDARs arrangements. Furthermore, the distance readings and headings of each LiDAR should be considered.

A ROS2 package that could merge multiple LiDARs data with the ability to filter and transform each LiDARs data is needed to solve the problem. The combined LiDARs data set is represented by equation (1).

$$A = \left((B_1 - C_1) \cup (B_2 - C_2)\right) \in \mathbb{R} \quad (1)$$

where A is merged LiDARs data, B_1 is LiDARs 1 data, C_1 is unwanted LiDARs 1 data, B_2 is LiDARs 2 data, C_2 is unwanted LiDARs 2 data.

With the transform equation in equation (2).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} x + x_{off} \\ y + y_{off} \end{bmatrix} \in A \quad (2)$$

where x is horizontal axis initial coordinate (m), y is vertical axis initial coordinate (m), x_{off} is horizontal axis coordinate offset (m), y_{off} is vertical axis coordinate offset (m), x' is horizontal axis final coordinate (m), y' is vertical axis final coordinate (m), θ is angular heading adjustment (rad)

B. ROS2

Figure 1 shows the overall architecture of a Linorobot2-based mobile robot [23]. It consists of three main parts, which are the virtual robot, physical robot, and Nav2 navigation. The virtual robot is a dummy robot that could be simulated using Gazebo. The dimension of the virtual robot is adjustable by using the parameterized Xacro files to mimic the physical robot. The physical robot contains the micro-ROS, which will publish sensors input and execute movement commands in collaboration with the navigation part. The navigation part utilizes the Nav2 package, which also serves as the default navigation package in ROS2.

The "/scan" topic published by the physical robot is subscribed to by the Nav2 package. This part is important as the main environment data input. In this thesis, the "/scan" package is modified due to the presence of the second LiDAR. The serial data received from the LiDARs could not be converted directly since the Nav2 package only supports a single LiDAR. Other than that, the raw LiDAR data needs to be adjusted in terms of the offset to the center of the robot and hiding the useless points where the LiDARs are reading the body of the robot.

Three important topics required to move the robot are "/cmd_vel", "/odom/unfiltered", and "/imu/data" topics. The "/cmd_vel" topic is responsible for giving movement commands from the navigation packages to the robot. As feedback, the "/odom/unfiltered" topic contains the coordinate of the robot relative to the starting position of the robot published by the actual robot. The "/imu/data" topic is published by the actual robot, consisting of the IMU data.

The publisher and subscriber of those three topics will be made inside the Teensy following the Linorobot2_Hardware micro-ROS program structure [24]. However, some adjustments are needed due to the unsupported IMU and motor driver. In addition, an MQTT commands receiver package to determine the navigation target of the robot is needed.

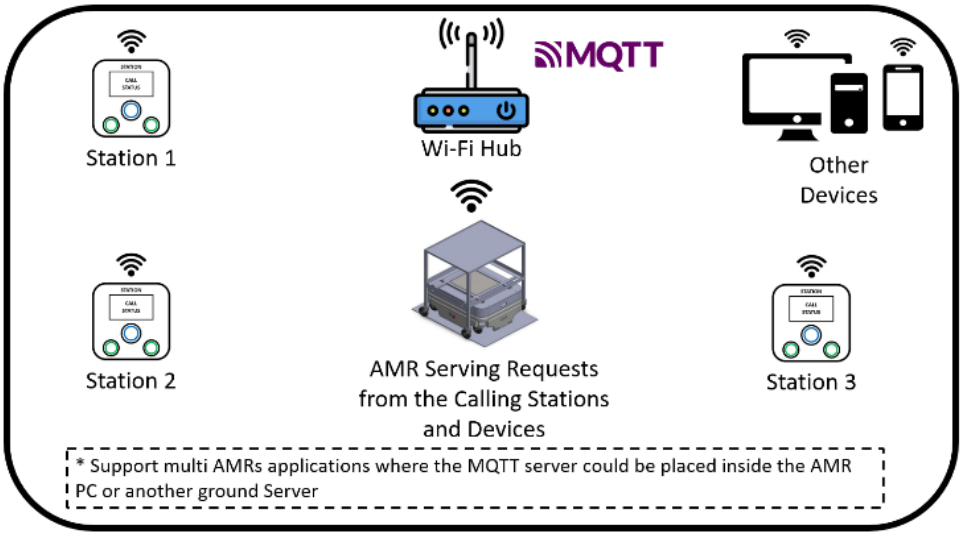

Figure 2. Overall AMR system overview.

The package will subscribe to the MQTT message and publish the navigation goal coordinates to the Nav2 package. The MQTT broker will be installed in the AMR directly. The package will publish a specific coordinate based on the position name published in the topic target and wait for feedback from the calling station. The ROS2 architecture is structured this way deliberately. The publish/subscribe topology lets the two LiDAR drivers stream independently while the merger node enforces a common /scan timestamp before forwarding the unified message to Nav2, so the SLAM and costmap layers never see two competing inputs. Keeping /cmd_vel, /odom/unfiltered, and /imu/data on separate logical topics avoids head-of-line blocking when navigation traffic spikes, and routing them through micro-ROS on the Teensy keeps the hard-real-time path off the host CPU.

Figure 2 shows the overall AMR system overview. The main interface between the user and the AMR is a microcontroller calling stations that are connected to one another via MQTT over a Wi-Fi connection. The calling stations are needed as a simple interface that is self-explanatory and easy to use for the users. Due to its flexibility, the MQTT data stream could also be utilized by other devices such as PCs and mobile phones with various operating systems, as well as more advanced interfaces. The main server could be located inside the AMR itself or installed separately as a ground station based on the use case. In practice, this MQTT structure has a direct bearing on system responsiveness. Because the broker runs on the AMR itself, calling-station-to-robot latency is bounded by the local Wi-Fi link rather than by an external network hop, with end-to-end command delivery measured under nominal load. Retained messages let any newly powered calling station learn the AMR's current state at subscribe time without polling, which keeps the user interface in sync after intermittent outages.

III. Results and Discussions

The design construction were conducted successfully with the presentable final form factor. The locking mechanism for the chassis works perfectly, sticking to the AMR main structure. The AMR could move forward and backward despite the absence of proper AMR software with the help of the manufacturer's BLDC hub motor software. Based on that, the mechanical design of the AMR that is developed in this research will be adopted for this thesis. The result of the chassis, carrier, and navigation test is shown in Figure 3 and Figure 4 and suitable for mass production.

Figure 3. Base AMR chassis and cart carrier.

Figure 4. ROS2-based outdoor AMR test result.

Table 1.
Merged LiDAR data validation.

No	Side	LiDARs Measured Distance (m)	Actual Physical Distance (m)	Error (m)
1	Front	1.00	0.96	0.04
2	Front	2.00	1.94	0.06
3	Front	3.00	2.93	0.07
4	Front	4.00	3.91	0.09
5	Front	5.00	4.95	0.05
6	Side	1.00	0.97	0.03
7	Side	2.00	1.92	0.08
8	Side	3.00	2.90	0.10
9	Side	4.00	3.93	0.07
10	Side	5.00	4.93	0.07
Average error (m)				0.07

A. Merged LiDARs data validation

Table 1 shows the merged LiDARs data validation. The test was conducted by putting an object in the front and side of the AMR. The object distances from the center of the AMR are then set with an interval of one grid in RVIZ2 from one to five meters (one grid in RVIZ2 is one meter). The real distances from the center of the AMR to the object are then measured using a measuring tape. The measured distance for each data point is repeated three times by rolling and unrolling the measuring tape. The measurement result always shows the same measurement. As shown in the Table 1, the average error of the LiDARs data range is 0.07 m with a maximum error of 0.1 m at 3 m distances at the side of the AMR. The RPLIDAR S1 itself has an accuracy of ±5 cm and a resolution of 3 cm [25]. Table 1 shows that the maximum error is still within the range of the accuracy mentioned in the RPLIDAR S1 datasheet. It shows that the merging process was done successfully with satisfactory results in terms of sensor accuracy.

B. Mapping performance in dynamic environments

Figure 5 shows the mapping session of the 20th-floor main corridor of the Prominence Office tower result (upper part) alongside the building plan map (lower part). The main corridor represents a similar situation to the industrial case, where the area is cleaner with a lot of rooms to deliver the objects. The most important thing is the presence of outliers, such as people dynamically passing by the main corridor. The AMR successfully maps the main corridor without any problem. During the mapping session, the AMR was controlled by teleoperation. It could successfully detect and differentiate static and dynamic objects inside the map. It removes the dynamic objects and keeps only the static objects. The AMR navigates through the area two to three times, visiting the same place several times to check the loop closure. After all areas are scanned and the loop closure process is done, the map does not change at all anymore. Overall, it could show the details of the corridor, such as the presence of the elevator, doors, and trash can. The loop also closes pretty well, resulting in a precise map compared to the building plan map.

However, there are some areas that are not discoverable by the AMR. The top left and the bottom right areas are not discoverable due to the narrow entry, making the map data scattered. Other than that, glass doors also could not be interpreted well by the LiDARs. As shown in the bottom left and bottom middle areas of the map, the wall is inconsistent. In the middle of the corridor, some black spots could be found due to the presence of students passing by when the mapping session was done.

C. Localization test

Figure 6 shows the initial localization process where the AMR does not know its position inside the map.

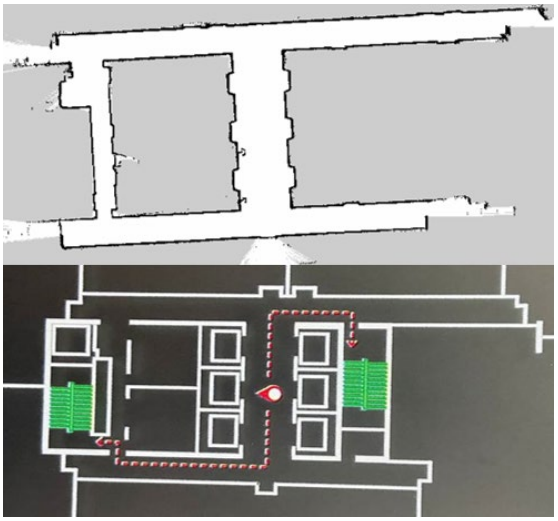

Figure 5. Mapping result.

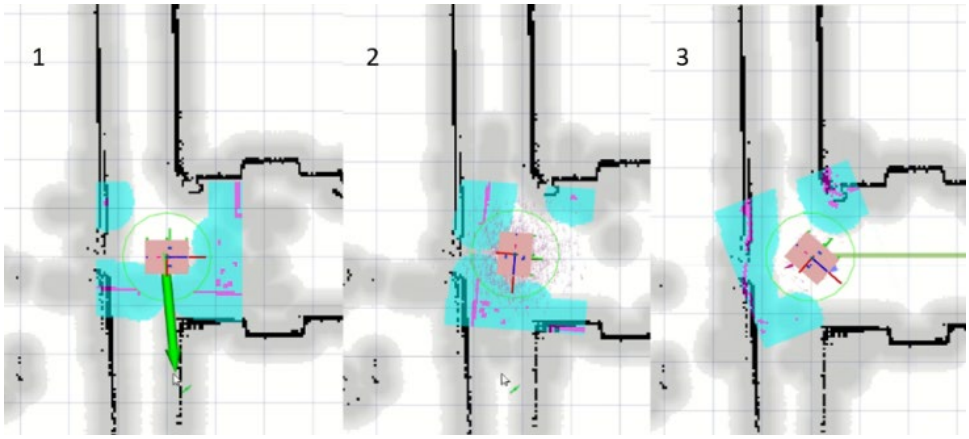

Figure 6. Localization process of the AMR using the AMCL algorithm, (1) initial; (2) localization process; (3) final state.

This process is critical to knowing where the AMR is located in the beginning. The left picture shows the initial position with the estimated pose displayed by the green arrow (user input). As can be seen, the purple dots do not match any walls on the map. It indicates that the AMR is blind to its position. The middle picture in the figure shows the localization process being done by the adaptive monte carlo localization (AMCL) algorithm. In this state, the AMR assumes that the given initial pose is correct. The small grey arrows show the particles representing the possible actual pose of the AMR. The AMR needs to move around to localize itself by reducing the particles based on the movement feedback. After moving a while, as shown in the right picture in the figure, the AMR completely localizes itself. The particles are gone, and the LiDARs data (purple dots) match the map's wall.

D. Navigation and obstacle avoidance test

Figure 7 shows the navigation and object avoidance ability of the AMR. The object that was being used in this test was a wooden plate shown in the first picture in the figure. It shows the initial condition where the AMR is localized and receives the goal pose command shown with the green bar. After it gets the goal pose, the Nav2 planner server will compute and publish the path shown with a pale green color. The controller server of the Nav2 packages will give the movement command to move the AMR according to the plan. The plan will be updated locally to avoid obstacles and adapt to dynamic changes. The wooden plate is not known locally by the AMR when the goal pose was published shown in the second picture in the figure. The initial path crosses the wooden plate as shown in the second picture in the figure. In the third picture in the figure, the initial path automatically adapting the presence of the wooden plate and avoids collision with it.

E. Navigation repeatability test

Figure 8 shows the navigation repeatability test. The AMR is commanded to go to the same goal pose ten times, and the distance between the center of the AMR and a fixed point is measured. The X-axis shows the horizontal distance between the center of the AMR and the fixed point. The Y-axis shows the vertical distance between the center of the AMR and the fixed point. As shown in Figure 8, all of the end positions of the AMR are still in range of the ROS tolerance set in the parameter file, measured from the mean, which is also still in the three-sigma range. Overall, the figure shows that the Nav2 package performs well and delivers the goal as it promises in the parameter file, with a standard deviation of 0.12 m.

Figure 9 shows that the MQTT receiver node successfully publishes the goal pose based on the calling station request. When the AMR is initialized, the status

Figure 7. Navigation and obstacle avoidance test, (1) set up; (2) initial; (3) final state.

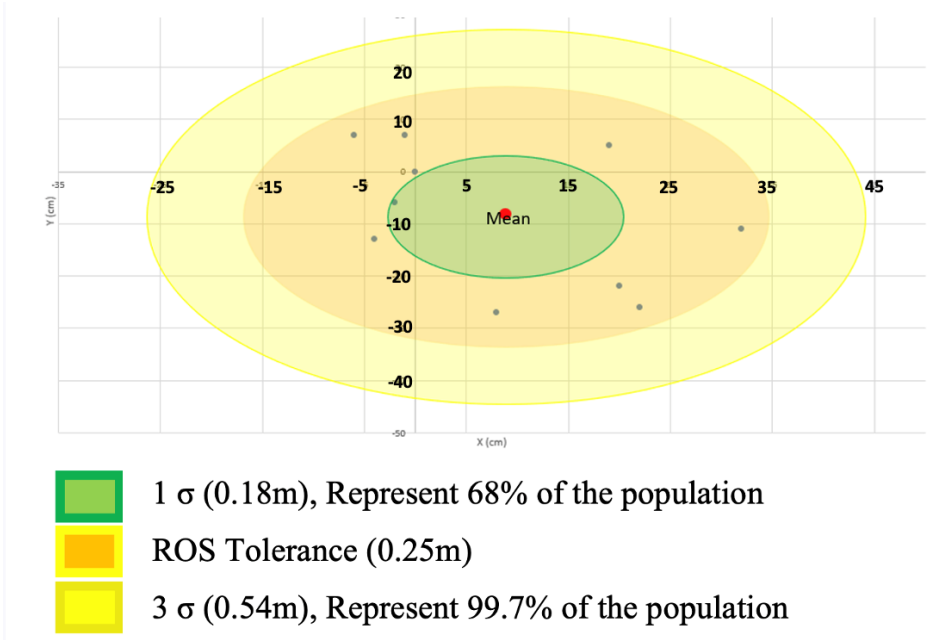

Figure 8. Navigation repeatability test.

of the AMR inside the MQTT broker is "idle". Based on that, the calling stations could make a call via the MQTT protocol. When the call button is pressed, the MQTT receiver node decides whether the desired station coordinate is known. If the station coordinate is known, it updates the status of the AMR to "going to pos x". Only the target station device will have the capability of whether the calling mission is done. If the target station device sends the "done" command by the user input, the AMR status is back to "idle".

During the test, the AMR successfully navigates through each desired position. The Nav2 behavior tree has a characteristic where if any new goal pose coordinate is published, the old goal pose coordinate is neglected, following the new coordinate even if the old coordinate has not been reached. It could be a big problem in the industry since a mission could be easily interrupted by another call. Without proper handling, the possibility of no mission accomplished could occur due to the uncontrolled interruption. In these cases, the MQTT receiver node and the calling station firmware have successfully fixed the problem where the new goal poses coordinate will only be able to be published whenever the old goal poses coordinate has been reached by a confirmation from the calling station.

Figure 10 shows a common bug in the Navigation2 bug in the ROS2 Foxy distro. The Nav2 recovery server tends to fail to load the cost map in case of no possible route due to the over-constrained inflation layer or being blocked physically. This bug is not solved in ROS2 Foxy distro due to the current solution applied in the Galactic and Humble distro breaks the ABI rules in the Foxy distro.

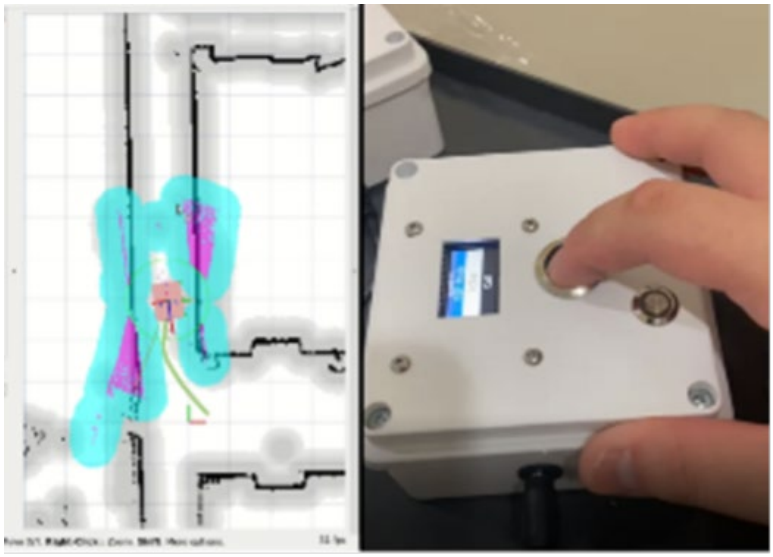

Figure 9. MQTT-based calling station serves the user's request.

Figure 10. Navigation2 bug in ROS2 Foxy distro.

IV. Conclusion

The AMR successfully works well based on ROS2 Foxy, micro-ROS, SLAM toolbox for generating maps, AMCL to localize, and Nav2 for navigating with the navigation repeatability St. deviation of 11.95 cm. It successfully avoids static and dynamic obstacles while navigating to the goal. Furthermore, the LiDARs data merger package performs well, covering 360-degree, angular resolution of 0.55-degree, average error of 7 cm, and is successfully utilized by other ROS2 packages. At the end, the AMR successfully served all the calling station requests (15 out of 15) over the MQTT protocol when the Nav2 package is ready and free from the Nav2 server bug.

Acknowledgements

Academic Research Community Service "Indoor Autonomous Mobile Robot". Faculty Research Fund "Motor Upgrade Indoor Autonomous Mobile Robot on ROS2".

Declarations

Author contribution

All authors contributed equally as the main contributor of this paper. All authors read and approved the final paper.

Funding statement

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The use of AI or AI-assisted technologies

During the preparation of this work the authors used Gemini AI for the purpose of grammar checking and revising improper sentence construction. After using this tool/service, the authors reviewed and edited the content as needed and takes full responsibility for the content of the publication.

Additional information

References

[1] F. Yao, B. Alkan, B. Ahmad, and R. Harrison, "Improving just-in-time delivery performance of IoT-enabled flexible manufacturing systems with AGV based material transportation," *Sensors (Switzerland)*, vol. 20, no. 21, pp. 1–25, 2020.

[2] J. Walker, "Mobile robot market revenues grow by 20% in 2020 despite pandemic delays," *Interact Analysis*, 2021. (accessed Feb. 08, 2022).

[3] T. Lackner, J. Hermann, J. C. Kuhn, and D. Palm, "Review of autonomous mobile robots in intralogistics: State-of-the-art, limitations and research gaps," *Procedia CIRP*, vol. 130, pp. 930–935, 2024.

[4] MiR. AGV vs. AMR - what's the difference?. 2022 [Accessed: 8 February 2022].
[5] D. R.-Y. Phang, W.-K. Lee, N. Matsuhira, and P. Michail, "Enhanced Mobile Robot Localization with Lidar and IMU Sensor," in *2019 IEEE International Meeting for Future of Electron Devices, Kansai (IMFEDK)*, Kyoto, Japan, pp. 71-72, 2019.
[6] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," IEEE Access, vol. 8, pp. 39830–39846, 2020.
[7] V. Susanti *et al.*, "Domestic component level analysis for multipurpose autonomous robot," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 12, pp. 87–94, 2021.
[8] L. Rusli, B. Nurhalim, R. Rusyadi, "Vision-based vanishing point detection of autonomous navigation of mobile robot for outdoor applications," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 12, No 2 pp. 117-125, 2021.
[9] F. Shamsfakhr, B. S. Bigham, and A. Mohammadi, "Indoor mobile robot localization in dynamic and cluttered environments using artificial landmarks", Eng. Comput., vol. 36, no. 2, pp. 400–419, Mar. 2019.
[10] S. Macenski, T. Moore, D. V. Lu, A. Merzlyakov, and M. Ferguson, "From the desks of ROS maintainers: A survey of modern & capable mobile robotics algorithms in the Robot Operating System 2," *Robotics and Autonomous Systems*, vol. 168, p. 104493, Oct. 2023.
[11] A. Rana, A. Petitti, A. Milealla, "Robotic inspection of vertical surfaces in GNSS-denied environments," *Robotics and Autonomous Systems*, vol. 203, pp. 105510, 2026.
[12] F. Yumbla *et al.*, "An open-source multi-robot framework system for collaborative environments based on ROS2," *IEEE Access*, vol. 13, pp. 16288-16302, 2025.
[13] N. L. Manuel, N. İnanç, and M. Y. Erten, "Control of mobile robot formations using A-star algorithm and artificial potential fields," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 12, pp. 57–67, 2021.
[14] R. R. Pratama, C. H. A. H. B. Baskoro, J. D. Setiawan, D. K. Dewi, P. Paryanto, M. Ariyanto, and R. P. Saputra, "Nonlinear model predictive control with single-shooting method for autonomous personal mobility vehicle," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 15, no. 2, pp. 186–196, 2024.
[15] M. Mirdanies and R. P. Saputra, "Experimental review of distance sensors for indoor mapping," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 8, pp. 85–94, 2017.
[16] I. İnce, D. Yiltas-Kaplan, and F. Keleş, "From simulation to reality: Comparative performance analysis of SLAM toolbox and cartographer in ROS 2," *Electronics*, vol. 14, no. 24, Art. no. 4822, Dec. 2025.
[17] P. Aryan, S. D. Shetty, V. Kalaichelvi, and R. Karthikeyan, "SimNav-XR: an extended reality platform for mobile robot simulation using ROS2 and Unity3D," *Frontiers in Robotics and AI*, vol. 13, art. no. 1708161, 2026.
[18] F. Cañadas-Aránega, J. C. Moreno, J. L. Blanco-Claraco, A. Giménez, F. Rodríguez, and J. Sánchez-Hermosilla, "Autonomous collaborative mobile robot for greenhouses: design, development, and validation tests," *Smart Agricultural Technology*, vol. 9, Dec., 2024.
[19] M. A. L. Ramos *et al.*, "Experimental validation of translational single LiDAR 2D mechanism into autonomous navigation and 3D SLAM of mobile robots," *IEEE Access*, vol. 13, pp. 153386-153397, 2025.
[20] N. A. Witono, "Development of scaled-down self-driving car for last mile delivery based on ROS2," Bachelor's thesis, Swiss German University, Tangerang, Indonesia, 2021.
[21] M. Ichsan, "Mechanical improvement on Autonomous Mobile Robot," Bachelor's thesis, Swiss German University, Tangerang, Indonesia, 2022.
[22] D. Lopes *et al.*, "Integrated fleet management of mobile robots for enhancing industrial efficiency: A case study on interoperability in multi-brand environments within the automotive sector," *Applied Sciences*, vol. 15, no. 13, p. 7235, Jun. 2025.
[23] Linorobot, *linorobot2*. GitHub repository, humble branch. [Accessed: Jul. 7, 2026].
[24] Linorobot, *Linorobot2_Hardware*. GitHub repository. [Accessed: Jul. 7, 2026.2022].
[25] Slamtec, *RPlidar S1 Datasheet,* pp. 1–17, 2019, [Online]. Available: slamtec.com.