EVOLUTION IN ELECTRICAL AND ELECTRONIC ENGINEERING
e-ISSN: 2756-8458

EEEE

Vol. 7 No. 1 (2026) 26-32
https://publisher.uthm.edu.my/periodicals/index.php/eeee

Application of ROS 2-Based Linorobot2 Framework on A Boxy Mobile Robot

Iman Hakimi Shah[1], Mohamad Fauzi Zakaria[1]*

[1] *Faculty of Electrical and Electronic Engineering*
Universiti Tun Hussein Onn Malaysia, Parit Raja, 86400, MALAYSIA

*Corresponding Author: mfauzi@uthm.edu.my
DOI: https://doi.org/10.30880/eeee.2026.07.01.004

Article Info

Received: 18 February 2026
Accepted: 1 April 2026
Available online: 30 April 2026

Keywords

ROS 2, Linorobot2, Raspberry Pi, Micro-ROS, Raspberry Pi Pico, Mobile Robot, Differential Drive, LiDAR.

Abstract

This project implements the linorobot2 framework, integrated with Robot Operating System 2 (ROS 2), on a custom-built Boxy mobile robot. The main goal is to create a reliable system for manually controlling this differential-drive robot indoors. A Raspberry Pi 4 running ROS 2 Humble serves as the main computer, managing high-level ROS 2 nodes for communication and control. Simultaneously, a Raspberry Pi Pico microcontroller handles low-level motor control and processes data from wheel encoders, communicating with the ROS 2 system using micro-ROS. The robot utilises a YD LiDAR sensor for environmental data acquisition, supporting future mapping and localisation. While the current focus is on manual control via keyboard teleoperation, the project lays the foundation for fully autonomous navigation using the Navigation2 (Nav2) framework. The results demonstrate that the Linorobot2 framework provides a modular and scalable platform for low-cost mobile robotics in education and research.

1. Introduction

The field of mobile robotics has undergone a significant transformation with the emergence of the Robot Operating System (ROS), which provides a standardised middleware for complex robotic applications [1]. This project focuses on the practical application of the ROS 2-based Linorobot2 framework on a custom-built "Boxy" mobile robot, addressing the critical need for accessible, low-cost platforms in educational and research settings [2]-[6]. A primary challenge in developing such systems is seamlessly integrating high-level computational tasks, such as environmental perception and path planning, with low-level hardware actuation. Traditional architectures often suffer from high latency and rigid communication protocols that hinder real-time responsiveness. By leveraging ROS 2 Humble on a Raspberry Pi 4 and micro-ROS on a Raspberry Pi Pico, this project establishes a modular, scalable architecture that bridges the gap between high-level processing and embedded hardware control.

The primary objectives of this research involve the systematic development and validation of a functional differential-drive mobile robot platform. This process includes integrating the Linorobot2 framework with ROS 2 to manage high-level control nodes and implementing micro-ROS to facilitate efficient, first-class communication between the microcontroller and the ROS graph [7]. Furthermore, the project aims to validate the system's operational reliability through extensive manual teleoperation testing and rigorous sensor data verification, with a specific focus on the fusion of wheel encoder and IMU data via an Extended Kalman Filter (EKF). By achieving these goals, the project not only demonstrates a reliable manual control system but also provides a robust, pre-configured foundation for the future deployment of autonomous navigation using the Navigation2 (Nav2) stack and SLAM Toolbox [8].

Evolution in Electrical and Electronic Engineering Vol. 7 No. 1 (2026) p. 26-32

The scope of this project is specifically tailored to indoor environments, utilising a YDLIDAR X2 for 360-degree environmental scanning and an MPU6050 IMU for precise motion tracking. The "Boxy" robot serves as a physical prototype to test the Linorobot2 hardware abstraction layer, ensuring that velocity commands (cmd_vel) are accurately translated into motor PWM signals while maintaining stable odometry feedback. This work contributes to the growing ecosystem of open-source robotics by providing a reproducible reference for students and researchers to build and experiment with sophisticated ROS 2-based systems at a fraction of the cost of commercial alternatives.

2. Methodology

The methodology for this project is structured around a modular system architecture that separates high-level intelligence from low-level hardware execution. This approach ensures that the robot can handle complex sensor data while maintaining precise motor control.

2.1 System Architecture

The system architecture follows the Linorobot2 framework, which integrates ROS 2 and micro-ROS. The Raspberry Pi 4 serves as the central processing unit, hosting the ROS 2 Humble environment. It manages high-level nodes, including the robot_state_publisher , ydlidar_ros2_driver , and the robot_localization EKF node. The Raspberry Pi Pico acts as the hardware interface layer, running micro-ROS to communicate with the main controller via a serial link. This architecture enables real-time processing of LiDAR scans and IMU data, while the Pico handles time-critical tasks such as PWM generation and encoder pulse counting.

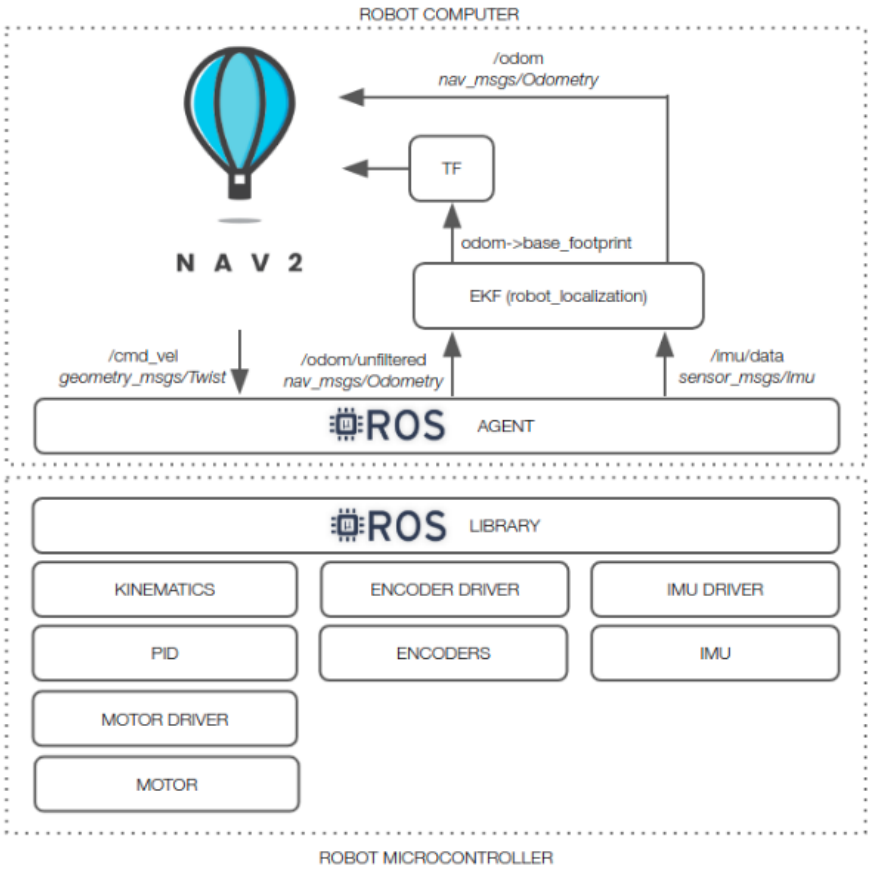


Fig. 1 *Linorobot2 System Architecture with ROS 2 and Micro-ROS Integration*

2.2 Circuit Connection and Hardware Integration

The hardware integration involves a complex network of components powered by a distributed power system. The Raspberry Pi 4 is powered via a dedicated 5V supply, while the motors and L298N driver are powered by a separate battery source to prevent electrical noise from affecting the logic circuits. The Raspberry Pi Pico is connected to the Pi 4 via USB for both power and data communication.

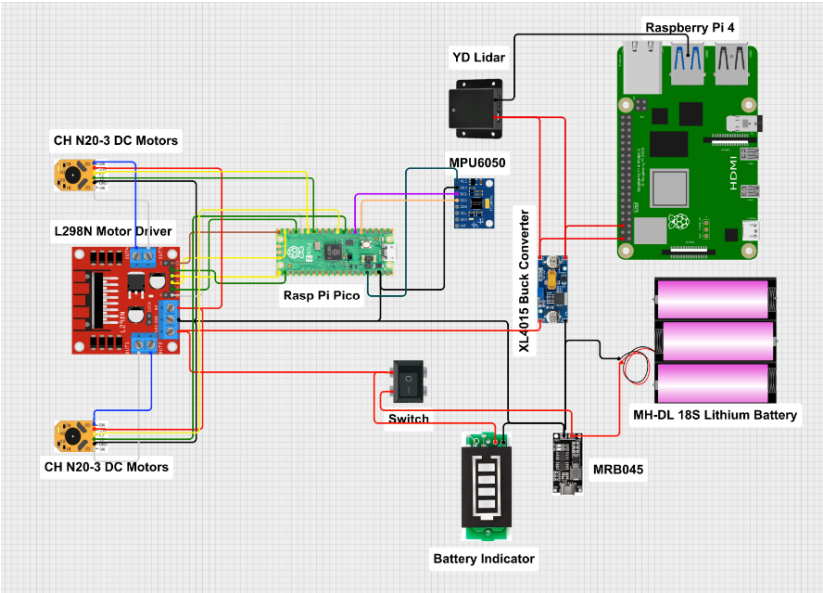
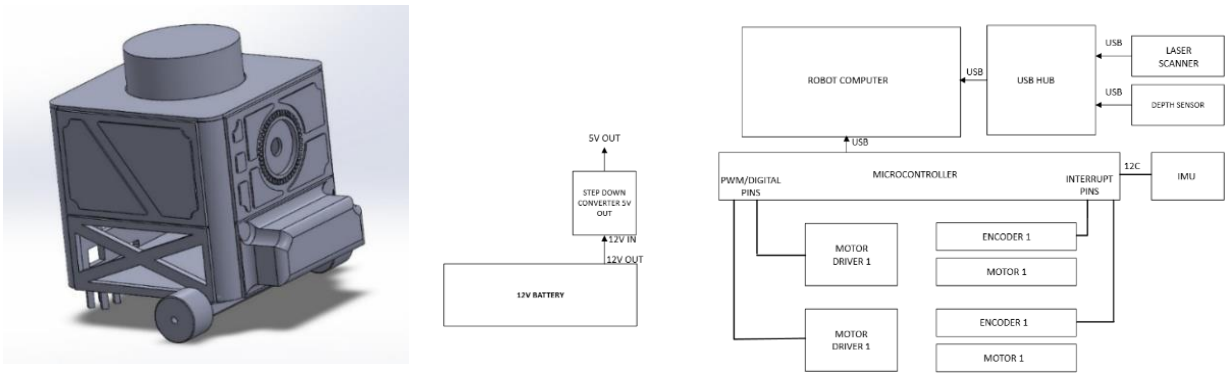


Fig. 2 *Complete wiring diagram of the Boxy mobile robot*

2.3 Boxy Mobile Robot Hardware Architecture and 3D Design

The physical development of the Boxy mobile robot began with a comprehensive 3D design phase to ensure structural integrity and optimal component placement. The robot's chassis was designed using SolidWorks, focusing on a multi-tiered "boxy" structure that provides sample space for the internal electronics while maintaining a compact footprint for indoor navigation. The design features a differential-drive configuration with two primary drive wheels located at the centre axis and two caster wheels for stability. This 3D model enabled precise positioning of the Raspberry Pi 4, Raspberry Pi Pico, and the L298N motor driver, ensuring the centre of gravity remains low to prevent tipping during rapid acceleration or turning.

The hardware architecture is built upon this physical design, integrating the mechanical components with the electronic control system. The chassis houses the DC gear motors equipped with Hall-effect encoders, which are directly coupled to the drive wheels. The YDLIDAR is mounted on the topmost tier to provide an unobstructed 360-degree field of view for environmental scanning. Internally, the hardware is organised into a logic layer and a power layer; the logic layer contains the microcontrollers and sensors, while the power layer consists of the motor drivers and battery packs. This physical and architectural organisation ensures that the robot is not only robust in its construction but also efficient in its operational performance, providing a stable platform for the ROS 2-based control framework.



(a) **(b)**

Fig 3 (a) 3D *model of the original Boxy mobile robot design (b) Hardware Architecture Diagram*

Evolution in Electrical and Electronic Engineering Vol. 7 No. 1 (2026) p. 26-32

3. Results and Discussion

The performance of the Boxy mobile robot was evaluated through a series of systematic tests, ranging from low-level firmware verification to high-level navigation tasks. The results confirm the successful integration of the ROS 2-based Linorobot2 framework with the custom hardware platform.

3.1 Firmware and Communication Performance

The integration of micro-ROS on the Raspberry Pi Pico was validated by monitoring the communication between the microcontroller and the Raspberry Pi 4. The /pico node was successfully discovered in the ROS 2 network, establishing stable subscriptions to the /cmd_vel topic and publishing sensor data to the /imu/data and /odom topics. Data rate analysis showed that the MPU6050 IMU maintained a consistent publication frequency of 50 Hz, with a minimal timing jitter of less than 0.00056s. This stability is crucial for the Extended Kalman Filter (EKF) to provide accurate real-time pose estimation.

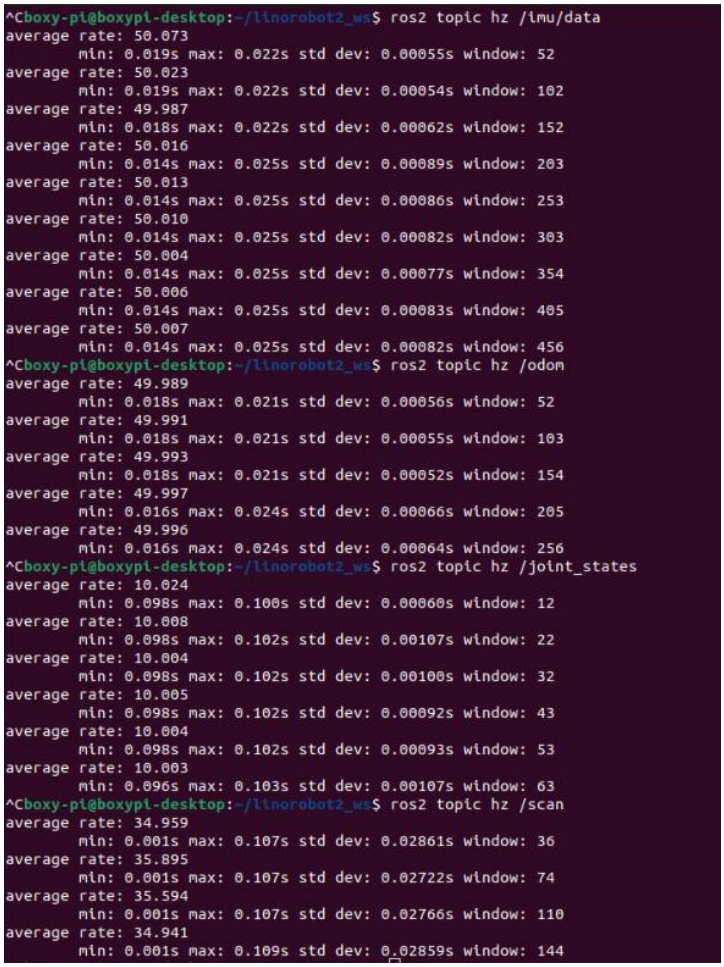


Fig. 4 *ROS 2 topic publication rate measurements for /imu/data, /odom, /joint_states, and /scan topics.*

3.2 Sensor Validation and Environmental Perception

The YDLIDAR was tested for its ability to map indoor environments. The sensor achieved a stable 360-degree scanning rate of 7 Hz, providing high-resolution distance measurements. During testing, the LiDAR successfully detected obstacles within its range, and the data was visualised in RViz2, showing a clear representation of the surrounding walls and objects. The IMU's accuracy was also verified; when stationary, the Z-axis acceleration measured approximately 9.77 m/s^2, aligning with gravitational norms, while the gyroscope responded accurately to rotational movements during manual handling.

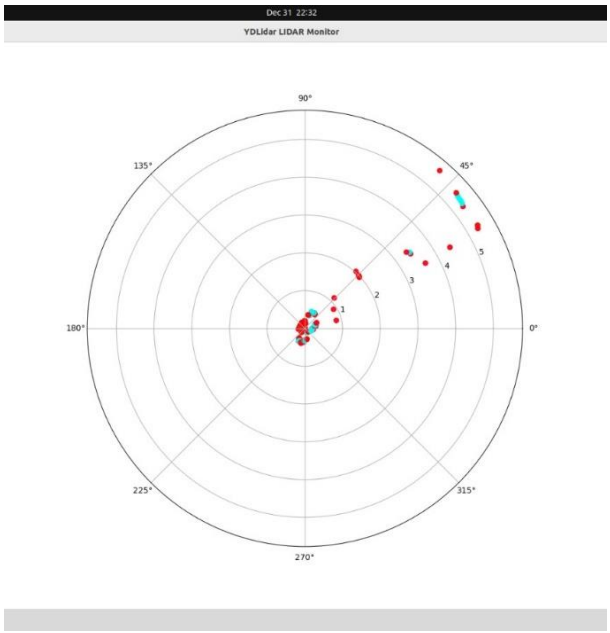


Fig. 5 *LiDAR PLOT*

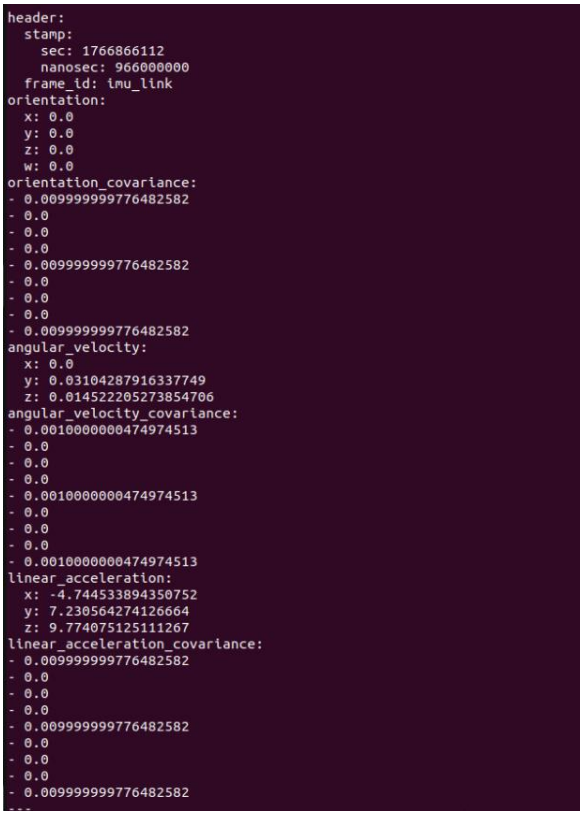


Fig. 6 *ROS 2 topic output for /imu/data showing IMU sensor data in YAML format.*

3.3 Motor Performance and Odometry Accuracy

The robot's mobility was assessed under both no-load and full-load conditions. The maximum linear velocity was recorded at 0.17 m/s, with a peak acceleration of 2.76 m/s^2. For rotational maneuvers, the robot reached a maximum angular velocity of 2.62 rad/s.

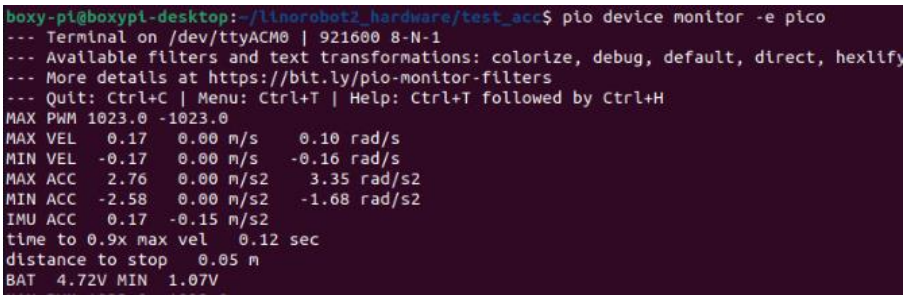


Fig. 7 *Acceleration test results for linear motion*

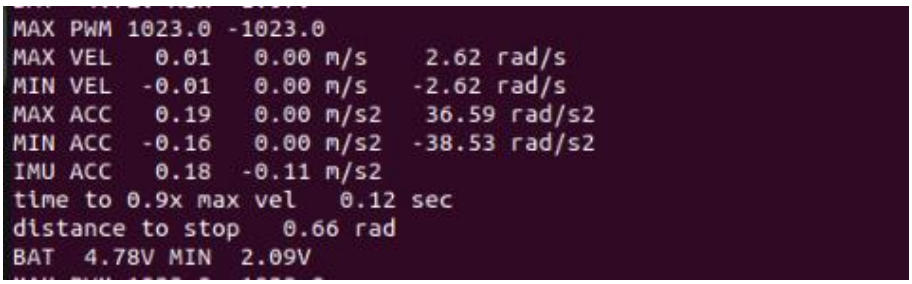


Fig. 8 *Rotational performance test results*

Evolution in Electrical and Electronic Engineering Vol. 7 No. 1 (2026) p. 26-32

Odometry accuracy was specifically tested on a controlled white paper surface to minimise environmental variables. The robot was commanded to travel a set distance, and the EKF-fused odometry-estimated position was compared with physical measurements. The results, summarised in Table 1, show a 93.4% positional accuracy. The minor 6.6% discrepancy (approximately 1.5 cm over a 22.8 cm run) is attributed to wheel slippage on the smooth paper surface and slight variations in the wheel-diameter parameters within the firmware.

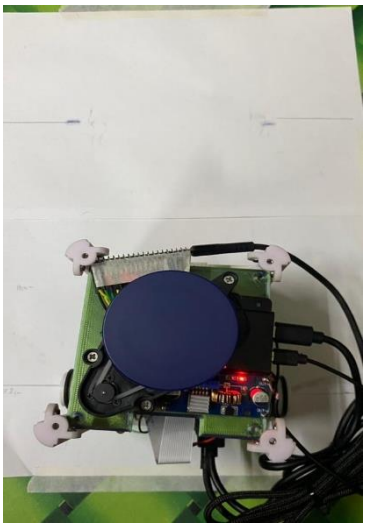


Fig. 9 *Boxy mobile robot at final position after forward movement on white paper test surface.*

Table 1 *Comparison between Physical Measurement and Odometry Estimation*

Trial	Physical Distance (cm)	Odometry Estimation (cm)	Accuracy (%)
Forward Movement	22.8	24.3	93.4%

3.4 Manual Teleoperation and SLAM Integration

Manual control via the teleop_twist_keyboard node demonstrated high responsiveness. The control pipeline from keyboard input to motor actuation functioned with negligible latency. Furthermore, preliminary SLAM (Simultaneous Localisation and Mapping) tests were conducted using the slam_toolbox. The robot successfully generated a detailed occupancy grid map of the test environment while simultaneously tracking its pose. The RViz visualisation confirmed that the system could handle real-time sensor fusion and mapping, proving that the platform is fully prepared for future autonomous navigation tasks using the Nav2 stack.

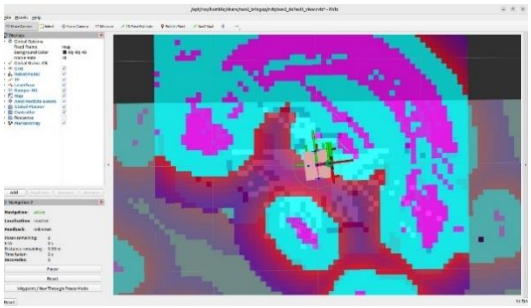


Fig. 10 *RViz visualisation during SLAM showing the generated occupancy grid map and robot pose.*

The results collectively demonstrate that the Boxy mobile robot, powered by ROS 2 and Linorobot2, is a robust and reliable platform. The high accuracy of odometry and the stability of the communication network meet the requirements for a research-grade mobile robot.

4. Future Work and Conclusion

The successful implementation of manual teleoperation and the preliminary SLAM testing have paved the way for several exciting avenues for future work. The primary focus will be on achieving fully autonomous navigation by leveraging the Navigation2 (Nav2) stack. This will involve configuring the various Nav2 servers, such as the planner, controller, and recovery servers, to work with the Boxy robot's specific characteristics. Further research will also be conducted on sensor fusion techniques, combining data from the wheel encoders, IMU, and LiDAR to improve the robot's localisation accuracy.

This paper has presented the successful application of the ROS 2-based linorobot2 framework on a custom-built Boxy mobile robot. The project achieved its goal of creating a reliable manual teleoperation system while laying the groundwork for future autonomous navigation. The use of a distributed architecture with a Raspberry Pi 4 and a Raspberry Pi Pico running micro-ROS proved to be an effective and low-cost solution for mobile robot development. Results from hardware integration and sensor testing have validated the system's functionality and performance. The successful teleoperation and preliminary SLAM tests demonstrate the robot's readiness for more advanced autonomous capabilities. This work contributes to the growing body of research in ROS 2-based robotics and provides a valuable platform for education and further development in the field of autonomous systems

Acknowledgement

The authors would like to thank the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, for its support.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***Study conception and design:*** *Iman Hakimi Shah Bin Ally Basira, Mohamad Fauzi Bin Zakaria;* ***Data collection:*** *Iman Hakimi Shah Bin Ally Basira;* ***Analysis and interpretation of results:*** *Iman Hakimi Shah Bin Ally Basira, Mohamad Fauzi Bin Zakaria;* ***Draft manuscript preparation:*** *Iman Hakimi Shah Bin Ally Basira, Mohamad Fauzi Bin Zakaraia. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2nd ed. MIT Press, 2011.
[2] S. Macenski et al., "Robot Operating System 2: Design, architecture, and uses in the wild," Science Robotics, vol. 7, no. 66, 2022.
[3] S. Macenski et al., "The Marathon 2: A Navigation System," in IROS, 2020, pp. 2718– 2725.
[4] Linorobot, "linorobot2: Autonomous mobile robots," GitHub. [Online]. Available: https://github.com/linorobot/linorobot2
[5] M. Quigley et al., "ROS: an open-source Robot Operating System," in ICRA Workshop, 2009.
[6] hippo5329, "linorobot2_hardware Wiki," GitHub. [Online]. Available: https://github.com/hippo5329/linorobot2_hardware/wiki
[7] B. Stadnik, "Overview Analysis of Micro-ROS System," PRZEGLĄD ELEKTROTECHNICZNY, vol. 1, no. 12, pp. 219–224, 2024.
[8] Nav2 Documentation, "Navigating while Mapping (SLAM)," [Online]. Available: https://docs.nav2.org