Tetsuzakura Knights🌸 Technical Paper Robotics Dojo Competition 2025

Glenn Gatiba, Prudence Njoroge, John Khaemba, Allen Wachio

Abstract

This paper presents the design and development of a four-wheeled autonomous off-road robot engineered for the 2025 Robotics Dojo Competition. The platform integrates mechanical, electrical, and computational subsystems into a cohesive architecture optimized for reliable operation in unstructured and dynamic environments. ROS2 serves as the middleware layer, enabling modular communication across perception, planning, and control nodes. Mapping and localization are achieved through LiDAR-based SLAM, while each wheel is independently driven by a 25GA370 12 V DC motor to maximize traction, stability, and maneuverability on uneven terrain. The development process emphasized rapid iteration, employing 3D printing and laser cutting for custom structural components. Prior to physical deployment, URDF modeling, RPLidar integration, and simulation in RViz and Gazebo were used to validate navigation strategies, sensor placement, and control pipelines, reducing design risks and accelerating system refinement.

Keywords

ROS2, SLAM, LiDAR

I. Introduction

Robotics has become an essential field in advancing automation, mobility, and intelligent systems across diverse industries. In particular, autonomous ground vehicles are increasingly deployed for applications ranging from agriculture to exploration, where reliable navigation in unstructured terrains is a critical challenge. Off-road robots, unlike those designed for controlled indoor environments, must contend with irregular surfaces, unpredictable obstacles, and limited sensor visibility. Designing such systems requires careful integration of mechanical robustness, efficient electronics, and intelligent software.

The Robotics Dojo Competition 2025 provides a platform for addressing the challenges of constrained, real-world autonomy by engaging teams in the design and implementation of robots capable of executing diverse tasks under strict size, power, and time limitations [1]. To meet these requirements, our team developed a four-wheeled autonomous robot with compact dimensions of 300 × 257 × 207 mm, optimized for maneuverability in the dynamic game field. Each wheel is powered by an independent 12 V DC motor, providing enhanced traction and mobility across rough terrain [2].

The perception system integrates an RPLiDAR sensor for Simultaneous Localization and Mapping (SLAM), enabling real-time localization in changing environments [3]. Complementary ultrasonic sensors, mounted laterally, together with an onboard IMU, provide diagnostic feedback for drift detection, immobilization monitoring, and LiDAR frame filtering to ensure map integrity [4]. A color sensor mounted near the conveyor belt identifies the color of transported loads; this information is cross-referenced with input from a forward-facing Pi Camera to locate designated deposition zones [5]. Beyond object handling, the Pi Camera also supports plant health monitoring by running an OpenCV pipeline with a pre-trained classification model, allowing the robot to distinguish between healthy and unhealthy leaves at the start of operation [6].

The robot supports two operational modes: teleoperation for subsystem debugging and validation, and autonomy for competition tasks [7]. ROS2 middleware coordinates perception, planning, and control nodes, while the Nav2 stack enables dynamic path planning and obstacle avoidance [8]. This allows the robot to re-route autonomously if the originally mapped path becomes blocked, improving robustness in cluttered or uncertain environments [9]. Object handling is achieved via a 28BYJ-48 stepper-motor-driven conveyor belt for load collection and deposition, complemented by a servo-actuated tipper mechanism that enables self-recovery if the robot becomes stuck [10]. The CAD drawing for the tipper is illustrated in figure 1 below.

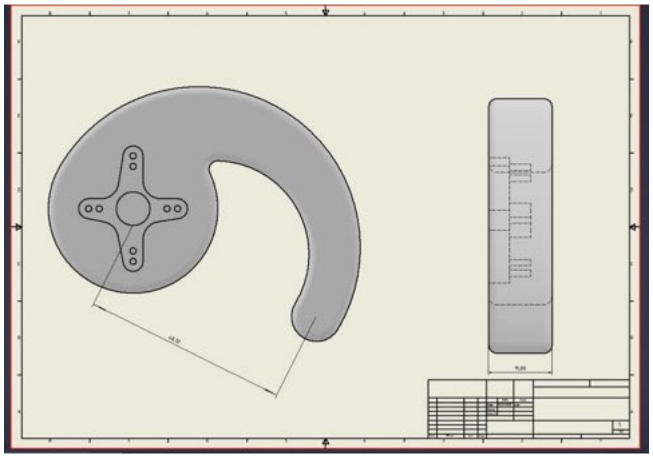


Figure 1. CAD drawing for the tipper

The contributions of this work include: (1) the development of a compact, low-cost autonomous robot tailored for off-road operation, (2) the integration of ROS2 middleware with LiDAR-based mapping, multi-sensor fusion, and plant-health analysis for task-specific perception, (3) the implementation of dynamic path-planning strategies for resilience in unpredictable environments, and (4) the design of a conveyor-based object handling and deposition system within a constrained 30 cm cubic form factor.

II. Related Work

Autonomous mobile robots have been widely studied in both academic and industrial contexts, with applications spanning warehouse logistics, planetary exploration, agriculture, and defense. Small-scale ground robots provide an effective platform for testing navigation, mapping, and perception algorithms in controlled settings before scaling to larger systems. Platforms such as the TurtleBot series [1] have been instrumental in popularizing the use of the Robot Operating System (ROS) for education and research, enabling the development and validation of algorithms including SLAM and autonomous navigation. Similarly, competition-focused robots such as those used in RoboCup [2] and Eurobot [3] emphasize the integration of perception, planning, and actuation within constrained design requirements.

Despite these advances, most existing small robots are designed primarily for indoor use, with limited capability to traverse rough or uneven terrain. Off-road navigation at small scales presents unique challenges due to reduced ground clearance, limited wheel traction, and the sensitivity of sensors to vibration. Furthermore, while manipulation tasks such as object pick-and-place are well explored in industrial robotics, the integration of lightweight object handling mechanisms into compact ground robots remains underexplored.

The Robotics Dojo Competition 2025 introduces a dynamic game field with both rough terrain and object interaction tasks, which necessitates a design that balances mobility, perception, and manipulation. Our robot addresses this gap by combining a four-wheel independent drive system for enhanced traction, LiDAR-based SLAM for navigation in dynamic environments, and a conveyor-based mechanism guided by a color sensor for object sorting and placement. This integration contributes to the development of low-cost, multifunctional off-road robots capable of operating in constrained environments.

III. Design Strategy

The design strategy of the proposed robotic vehicle, as seen in figure 2, was guided by competition constraints and mission objectives. The robot was required to fit within a 300 × 300 × 250 mm bounding box, while being capable of traversing rough terrain, mapping the dynamic game field, and handling objects based on color recognition. These requirements necessitated a compact yet robust design approach that balanced mechanical strength, sensing accuracy, and control flexibility.

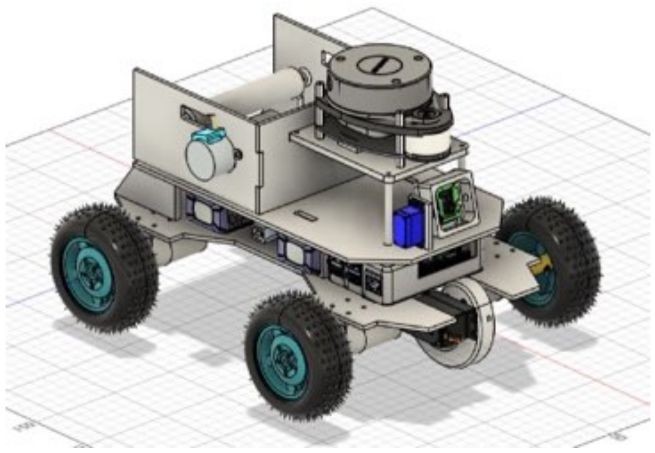


Figure 2. CAD model of the robot showing mechanical layout, conveyor system, and LiDAR placement.

A modular design partitioned the system into mechanical, electrical, and computational subsystems, following conventions used in platforms such as TurtleBot and Eurobot [1], [2]. The vehicle employs four independently driven off-road wheels to maximize traction and maneuverability on uneven terrain [3]. Object handling is performed by a 28BYJ-48 stepper-motor-driven conveyor belt that reliably transports and deposits loads; a high torque servo-actuated tipper provides recovery capability by lifting the front wheels to extricate the robot if it becomes immobilized.

Vision is performed by a forward-facing Pi Camera housed in a servo-actuated casing. A color sensor mounted at the conveyor interface inspects each load's color; the Raspberry Pi cross-references this reading with the Pi Camera's detected zone to route loads to their designated areas. Sensor fusion underpins navigation and data integrity: a LiDAR unit mounted on the mast provides primary object detection and SLAM for global mapping, while ultrasonic sensors mounted on both sides of the robot, together with an onboard IMU mounted at the bottom but center of the robot , monitor vehicle orientation and detect drifting or immobilization—feedback that is used to flag or discard LiDAR frames that could corrupt the map [5].

Computationally, a two-tier architecture separates perception from control: the Raspberry Pi handles vision, SLAM, color–zone matching, and high-level decision making, while an Arduino Mega handles low-level, real-time actuation of motors, servos, and the conveyor via dedicated drivers. Power is supplied by a rechargeable battery pack with buck regulation to provide stable rails for the Pi, microcontroller, sensors, and actuators. Together, these subsystems form a fault-aware, field-ready platform for robust navigation, color-based object handling[6].

The control architecture balances autonomy with reliability. A Raspberry Pi was designated as the high-level processing unit for tasks such as SLAM, object recognition, and teleoperation control, while an Arduino Mega microcontroller handles low-level motor actuation and sensor interfacing. The Arduino Mega was chosen due to: Sufficient number of PWM-capable pins for 4 DC motors and servos, multiple UARTs (Serial0–3), critical for debugging, ESP-01, and Pi interface, enough interrupt-capable pins for quadrature encoder inputs and sensor interrupts, and larger flash and RAM for integration with ROS node communication. This layered approach, common in distributed robotic systems, reduces computational bottlenecks and improves fault tolerance [7].

Finally, trade-offs were carefully considered in power management, weight distribution, and wiring complexity. Lithium-based batteries combined with a Battery Management System (BMS) were selected to provide sufficient endurance without exceeding the mass limitations [8]. The wiring harness was designed to accommodate multiple actuators and sensors, acknowledging the challenge of increased complexity in exchange for improved functionality.

This strategy ensured that the final design remained compact, robust, and capable of fulfilling the competition's operational requirements. Figure 3 shows the wiring schematic of the electrical system, highlighting the separation between high-current motor drivers and low-voltage sensor lines.

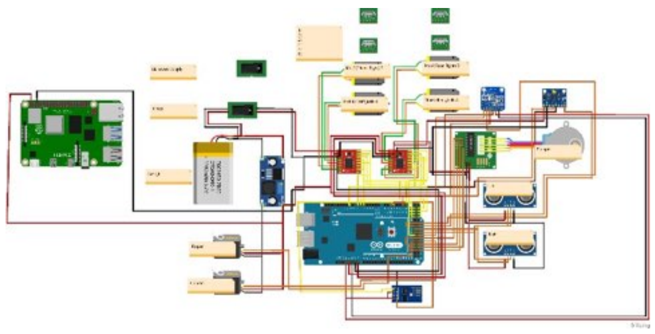


Figure 3. Electrical system schematic highlighting Arduino Mega, Raspberry Pi, sensors, and power distribution.

IV. Vehicle Design

A. Mechanical Design

The robot was designed within the competition-imposed volume constraint of 300 × 300 × 250 mm, which guided many early design decisions. CAD modeling (Fig. 2) was employed to optimize the allocation of internal space, balancing the requirements for electronic subsystems, actuation mechanisms, and sufficient ground clearance to traverse uneven terrain. Initial CAD iterations revealed conflicts between the conveyor assembly and sensor placement, which informed the final configuration.

The locomotion system employs off-road wheels, chosen after testing multiple alternatives to ensure traction and stability on rough surfaces. These wheels are each driven by 200rpm 12DC motors. Early prototypes with smaller, smooth wheels experienced excessive slippage on the competition terrain, underscoring the necessity of the final selection [1].

The object handling system consists of a stepper-motor-driven conveyor belt that deposits the load. A LiDAR sensor was mounted on the top platform to enable 360° field scanning. [4].

The tipper, illustrated in figure 1, payload mechanism was actuated using a 28BYJ-48 unipolar stepper motor coupled with a ULN2003 driver board. This motor was chosen due to its fine step resolution (≈4096 steps/rev with internal gearing), compact size, and sufficient torque output (~0.03–0.04 Nm), which makes it suitable for precise angular control of the tipping motion. While its maximum safe speed is limited to ~15 RPM, the application prioritizes controlled motion over rapid actuation, making the trade-off acceptable. The ULN2003 driver was selected to interface the motor with the Arduino Mega, providing reliable coil sequencing with built-in LED indicators for debugging. This configuration ensures stable, incremental actuation of the tipper while minimizing power consumption from the 5V supply line.

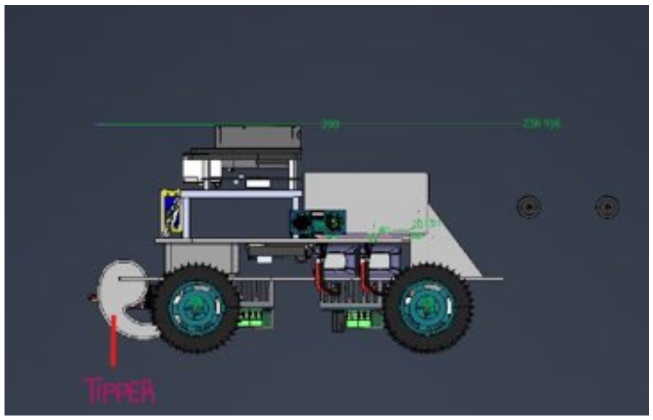


FIGURE 4. SIDE VIEW OF THE TIPPER ATTACHED TO THE ROBOT

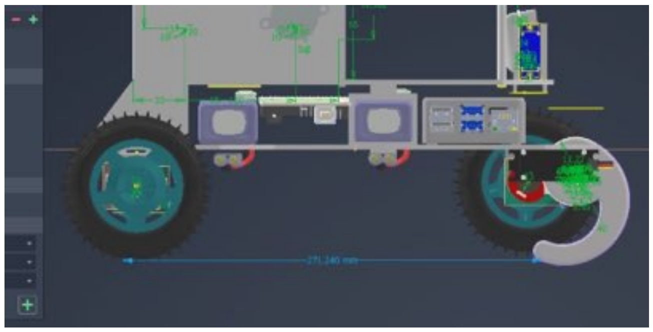


FIGURE 5. CAD GEOMETRY OF THE TIPPER AND WHEELBASE DISTANCE

The following schematic shows the torque calculation setup, including the wheelbase $L_{wheelbase}$, center of mass r_{com}, and tipper arm r_{arm} at different rotation angles θ

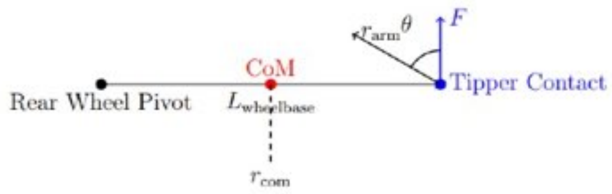


FIGURE 6. MATHEMATICAL TORQUE DIAGRAM: WHEELBASE $L_{WHEELBASE}$, CENTER OF MASS R_{COM}, TIPPER ARM R_{ARM}, AND LIFT ANGLE Θ OF THE SERVO

Mathematical torque diagram: wheelbase Lwheelbase, center of mass rcom, tipper arm rarm, and lift angle θ of the servo

Lesson learned: Compact robot dimensions magnify the impact of component placement. Elevating the LiDAR and modularizing the conveyor supports reduced interference and simplified maintenance.

B. Electrical and Power System

The electrical system follows a modular architecture separating low-level actuation from high-level computation. An Arduino Mega was selected as the primary microcontroller, offering abundant I/O pins for motor and sensor integration. Higher-level perception and decision-making tasks, including SLAM and teleoperation, were delegated to a Raspberry Pi, ensuring efficient task partitioning [2].

Initial wiring trials highlighted cable congestion and noise interference, which led to inconsistent sensor readings. To mitigate this, the wiring layout was redesigned, grouping sensor signal lines away from motor power lines. This separation significantly reduced electrical noise and improved system reliability.

The team selected the TB6612FNG motor driver instead of the more commonly used L298N. The decision was based on key performance trade-offs: the TB6612FNG offers lower power loss due to MOSFET outputs, supports higher efficiency, and delivers up to 1.2 A continuous current per channel with a peak of 3.2 A, compared to the L298N's ~0.6 A continuous output. Additionally, the TB6612FNG has a smaller footprint, making it more suitable for compact integration. While the L298N is widely available and inexpensive, its higher heat dissipation and lower efficiency were identified as limiting factors for sustained operation. Figure 6 shows the functional block diagram of the TB6612FNG

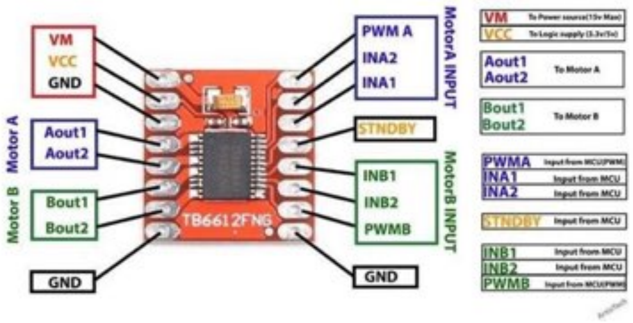
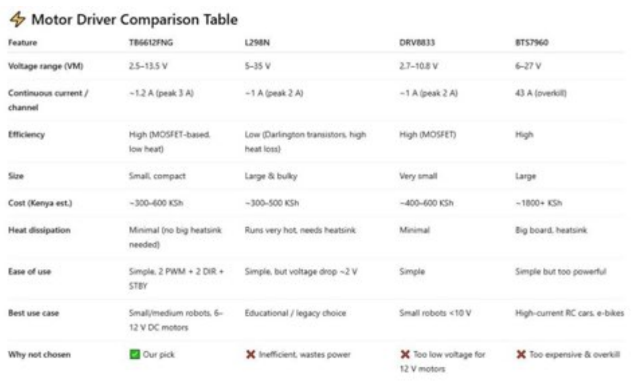


FIGURE 7. TB6612FNG MOTOR DRIVER USED IN THE ROBOT SHOWING DUAL H-BRIDGE CONFIGURATION FOR DC MOTOR CONTROL.

In order to determine the most suitable motor driver for our four-wheel drive configuration, we compared multiple options including the TB6612FNG, L298N, DRV8833, and BTS7960. The results are summarized in Table I, which highlights trade-offs in efficiency, size, cost, and current handling. Based on this analysis, the TB6612FNG was selected due to its balance of efficiency, compact form factor, and compatibility with our 12 V DC motors.

Table 1. Motor Driver Comparison Across Different Options



Motor Driver Comparison Table

Feature	TB6612FNG	L298N	DRV8833	BTS7960
Voltage range (VM)	2.5–13.5 V	5–35 V	2.7–10.8 V	6–27 V
Continuous current / channel	~1.2 A (peak 3 A)	~1 A (peak 2 A)	~1 A (peak 2 A)	43 A (overkill)
Efficiency	High (MOSFET-based, low heat)	Low (Darlington transistors, high heat loss)	High (MOSFET)	High
Size	Small, compact	Large & bulky	Very small	Large
Cost (Kenya est.)	~300–600 KSh	~300–500 KSh	~400–600 KSh	~1800+ KSh
Heat dissipation	Minimal (no big heatsink needed)	Runs very hot, needs heatsink	Minimal	Big board, heatsink
Ease of use	Simple, 2 PWM + 2 DIR + STBY	Simple, but voltage drop ~2 V	Simple	Simple but too powerful
Best use case	Small/medium robots, 6–12 V DC motors	Educational / legacy choice	Small robots <10 V	High-current RC cars, e-bikes
Why not chosen	✅ Our pick	❌ Inefficient, wastes power	❌ Too low voltage for 12 V motors	❌ Too expensive & overkill

Four 12 DC motors independently drive the wheels through motor drivers, enabling skid-steer motion. Stepper motor actuates the conveyor belt, while additional peripherals include ultrasonic sensors for lateral obstacle detection and a color sensor for object classification. Power is supplied via a 7.4V 2200mAh LiPo battery to the Raspberry pi and Arduino, and three lithium-ion batteries, to power motors, are monitored by a Battery Management System (BMS) to prevent over-discharge and ensure safe operation [7]. A servo motor is used to control the casing, which holds the pi camera.

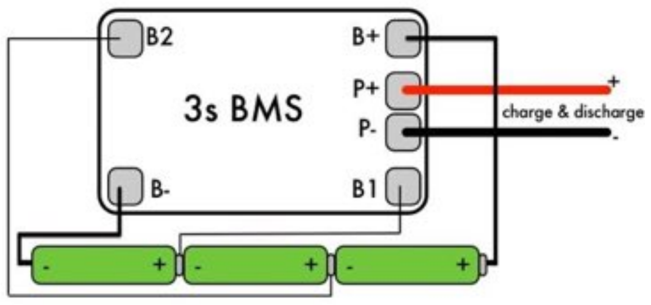


Figure 8. BMS Circuit used

C. Control and Software Architecture

The robot supports two operational modes: teleoperation, which facilitates early debugging and validation of subsystems, and autonomy, which integrates mapping, perception, and actuation for competition tasks. Teleoperation is available via a joystick and a remote GUI (web/ROS2-based), enabling live camera feed, manual drive, and on-screen diagnostics during testing. ROS2 serves as the middleware framework, enabling modular communication between nodes handling LiDAR-based SLAM, motor control, vision, planner/behavior, and sensor fusion (e.g., dedicated nodes for slam, motor_control, vision, planner, safety). ROS2 actions and topics are used for high-level navigation and coordinated actuation, improving scalability and maintainability relative to earlier monolithic Arduino sketches [2], [4].

The autonomous workflow consists of mapping the dynamic game field with LiDAR, detecting and classifying objects using the color sensor and Pi Camera, and executing path-planning strategies for collection and deposition [5], [6]. The Pi Camera runs an OpenCV pipeline with a pre-trained plant-health model; at the operator start point a display shows captured leaf images and instant healthy/unhealthy classification for quick verification. Color readings from a conveyor-mounted color sensor are passed to the Pi, which cross-checks camera-detected zones to determine the correct deposition target. The system implements a state machine that sequences perception → pick/transport → zone-verify → deposit → recovery actions to ensure deterministic behavior in competition scenarios.

Safety, robustness, and data integrity were addressed through multiple complementary features. Ultrasonic sensors positioned on both sides, together with the IMU, primarily monitor vehicle orientation and detect drifting or immobilization; they can also be configured as a short-range safety stop to halt motion if immediate lateral obstruction is detected, preventing collisions or entrapment. Wheel encoders provide odometry feedback for closed-loop speed control and improved pose estimation, and current-sensing on motor drivers enables stall detection and overcurrent protection. An emergency-stop button, a firmware watchdog, and a software-level failsafe in ROS2 (preemption of motion commands on error conditions) provide layered protection against faults.

Operational readiness and maintainability were further supported by calibration and diagnostics routines: automated startup checks verify sensor connectivity and sanity (LiDAR, camera, color sensor, IMU, encoders), an initial sensor calibration (camera intrinsics, color calibration, IMU bias) runs before autonomy is enabled, and runtime health telemetry (battery voltage, BMS status, motor currents, CPU load) is published and logged to local storage for post-run analysis. Communications and telemetry are available over Wi-Fi, with local SD logging as a fallback. These additions—modular teleoperation, ROS2-based node separation, explicit state sequencing, layered safety mechanisms, and systematic calibration/logging—help ensure dependable autonomous operation in the competition environment.

Navigation is implemented through the ROS2 Navigation Stack (Nav2), which integrates LiDAR-based SLAM, odometry feedback from wheel encoders, and IMU data to provide continuous localization and mapping. Nav2's planner and behavior tree architecture enable dynamic path planning, allowing the robot to autonomously re-route if obstacles block the originally mapped trajectory, ensuring task completion even in partially obstructed fields. Teleoperation is integrated into the same framework via the teleop_twist_joy package, allowing seamless operator control during debugging or fallback scenarios. Low-level actuation of motors is managed through ros2_control, which standardizes the hardware interface and provides velocity and position controllers for consistent execution of navigation commands. This layered navigation architecture—teleop, ROS2 control, and Nav2—ensures that the robot balances autonomy with operator oversight, while dynamic re-planning guarantees adaptability in unpredictable environments.

Lesson learned: Over-reliance on teleoperation during early development delayed integration of autonomous behaviors. Future iterations should emphasize parallel development of autonomy in simulation environments to reduce integration bottlenecks [1], [3].

V. Experimental Results

The robot was tested under both simulated and real-world conditions to evaluate mechanical robustness, sensor integration, and system performance. The development environment consisted of ROS2 as the middleware for modular node communication, URDF to describe the robot in XML, as illustrated in figure 9, Gazebo Fortress for physics-based simulation, and RViz for visualization of sensor data and system state. Fusion 360 was employed for CAD modeling, while code editing was performed in Visual Studio Code.

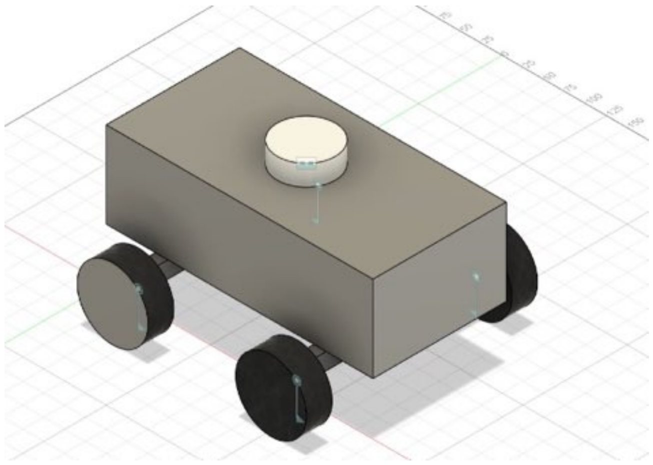


Figure 9. URDF describing robot in xml format

A custom fusion 360 URDF-Exporter script was developed to generate a URDF model, which was later adapted from Gazebo Classic to Gazebo Fortress due to compatibility issues. On the hardware side, the conveyor belt subsystem, powered by a stepper motor, demonstrated smooth and reliable object handling, the Pi Camera was configured with OpenCV, incorporating image-processing pipelines for feature extraction and classification. The LiDAR sensor code functioned as intended, enabling consistent mapping of the surrounding environment.

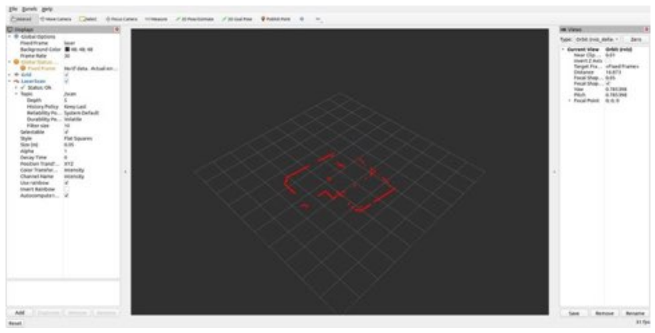


Figure 10. Raw LiDAR scan data being visualised in RViz

The power system was based on dual battery types: Lithium-ion packs with a Battery Management System (BMS), and LiPo batteries for high-current demands. Runtime tests were conducted by discharging the batteries until safe limits were reached. Faults encountered during integration included the failure of two motor drivers due to short circuits, and a LiPo charger malfunction.

Ultrasonic sensors were installed on both sides of the robot to work in conjunction with the IMU for monitoring vehicle orientation; however, initial code execution failed, and debugging is ongoing. Additionally, teleoperation was affected by buffering issues caused by multiple parallel processes in WSL, with occasional crashes under Gazebo Fortress. To address this, the team restructured the full codebase overnight, optimizing node execution and process scheduling.

In addition to hardware and integration testing, several trade-offs were encountered during the development of the robot's navigation and perception systems. A critical design decision involved the choice of mapping resolution in SLAM. A finer resolution (e.g., 0.05 m per grid cell) provided highly accurate navigation but significantly increased CPU load and slowed real-time updates. Conversely, coarser resolutions reduced computational demand but compromised navigation performance in tight spaces. After multiple trials, the team selected an intermediate resolution of 0.05–0.1 m per grid cell, which offered sufficient detail for reliable navigation while maintaining computational efficiency.

Another major trade-off was the sensor update rate. High-frequency lidar and odometry updates (20–40 Hz) produced smoother motion and improved localization but placed excessive demands on processing resources, occasionally leading to lag in path planning. Lower frequencies (5–10 Hz) eased the CPU load but reduced responsiveness. Through iterative tuning, the team stabilized SLAM by setting the lidar publish rate to approximately 10–15 Hz, which allowed timely map updates without overloading the system. This adjustment also addressed initial challenges with LiDAR, where low publish frequencies had previously caused delayed map updates and unstable navigation.

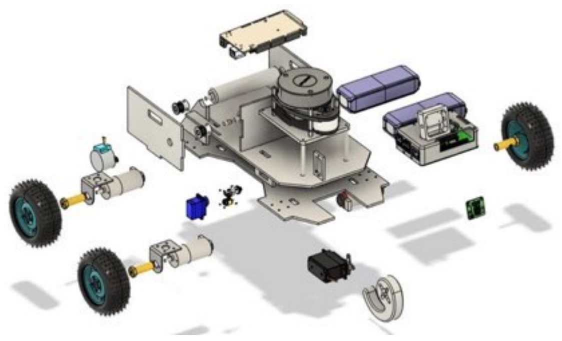


FIGURE 11. EXPLODED VIEW OF THE ROBOT CAD MODEL

VI. ACKNOWLEDGEMENTS

The authors would like to acknowledge the contributions of the broader robotics community, particularly the engineers and researchers whose published works on ROS2, Gazebo, RViz, SLAM, and the Nav2 stack provided invaluable technical guidance during system design and implementation. Widely referenced texts such as *Mastering ROS2* and *Programming Robots with ROS* served as foundational resources in developing and validating the robot's navigation framework.

Special thanks also go to Dr. Shohei Aoki for his technical advice and tutorial preparation, particularly his guidance on integrating navigation and task execution through PyTree-based state machine transitions, which informed the architecture of our autonomy pipeline.

We extend our gratitude to Mr. Lenny Ng'ang'a for his mentorship and continuous guidance throughout the project. Mr. Billy for his assistance in sourcing critical hardware components.

Finally, we recognize the trainers who provided us with the stepping stones to understand and apply much of the new content we encountered, ensuring that the team was able to build on a solid learning foundation.

VII. References

1. ROS 2 Tutorials | Husarion. (n.d.). https://husarion.com/tutorials/ros2-tutorials/ros2/
2. Wambua, C. (2025, August 20). Getting Started with Gazebo Fortress: Installation, Setup, and ROS 2 Integration. Robotics Dojo. https://roboticsdojo.substack.com/p/getting-started-with-gazebo-fortress
3. Rotich, V. (2025, August 20). Introduction to worlds in Robotics Simulation. Robotics Dojo. https://roboticsdojo.substack.com/p/introduction-to-worlds-in-robotics
4. Olumo, R. (2025, August 20). RQT setup in ROS2 Humble. Robotics Dojo. https://roboticsdojo.substack.com/p/rqt-setup-in-ros2-humble
5. Farida, M. M. (2025, August 20). Troubleshooting tips when setting up the RPLIDAR. Robotics Dojo. https://roboticsdojo.substack.com/p/troubleshooting-tips-when-setting
6. Farida, M. M. (2025b, August 20). Understanding URDF: building the blueprint of your robot. Robotics Dojo. https://roboticsdojo.substack.com/p/understanding-urdf-building-the-blueprint
7. Kimani, S. (2025, August 20). Getting Started with RViz in ROS 2 Humble. Robotics Dojo. https://roboticsdojo.substack.com/p/getting-started-with-rviz-in-ros
8. Opiyo, F. (2025, August 20). INTRODUCTION TO SIMULTANEOUS LOCALIZATION AND MAPPING. Robotics Dojo. https://roboticsdojo.substack.com/p/introduction-to-simultaneous-localization
9. BEST Arduino Mega 2560 MICROCONTROLLER | BluePack Nova. (2025, September 11). bluepackventures.com. https://bluepacknova.com/product/arduino-mega-2560-microcontroller/
10. Off-Road Wheels - 85x38mm. (2025). Pixel Electric Company Limited. https://www.pixelelectric.com/products/robots/wheels/rc-wheels/off-road-wheels-85x38mm/
11. DRV8833 TB6612FNG Step Motor Driver Module | BluePack Nova. (2025, September 11). bluepackventures.com. https://bluepacknova.com/product/drv8833-tb6612fng-step-motor-driver-module/
12. DC Motor 12V 200rpm with Encoder Module. (2025). Pixel Electric Company Limited. https://www.pixelelectric.com/more-categories/motors-drivers-actuator/brushed-dc-motor/geared-motor/dc-motor-12v-200rpm-with-encoder-module/
13. MG996R(Metal Gear) High Torque Servo Motor | BluePack Nova. (2025, September 14). bluepackventures.com. https://bluepacknova.com/product/servo-motor-mg996r%ef%bc%88metal-gear%ef%bc%89/
14. GT2 Precision Timing Belt pitch 2MM x 6MM wide. (n.d.). Nerokas Online Store. https://store.nerokas.co.ke/SKU-1305
15. ALUMINIUM GT2 TIMING PULLEY - 6MM BELT - 20 TOOTH - 5MM BORE. (n.d.). Nerokas Online Store. https://store.nerokas.co.ke/SKU-1526?search=pulley&description=true
16. OV7670 Camera module module. (2025). Pixel Electric Company Limited. https://www.pixelelectric.com/sensors/distance-vision/camera-sensor/ov7670-camera-module-module/
17. Male-Male Jumper Wire 40pcs Dupont. (2025). Pixel Electric Company Limited. https://www.pixelelectric.com/instruments-tools/wire-and-cables/cables/dupont-jumper-cable/male-male-jumper-wire-40pcs-dupont/
18. Buck Converter XL4015 | BluePack Nova. (2025, September 18). bluepackventures.com. https://bluepacknova.com/product/buck-converter-x4015/

Appendix

[1] M. Quigley, B. Gerkey, K. Conley, et al., "ROS: an open-source Robot Operating System," ICRA Workshop on Open Source Software, Kobe, Japan, 2009.

[2] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. MIT Press, 2005.

[3] M. J. Mataric, "Designing emergent behaviors: From local interactions to collective intelligence," in Proceedings of the International Conference on Simulation of Adaptive Behavior, 1991, pp. 432–441.

[4] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, Jun. 2006, doi: 10.1109/MRA.2006.1638022.

[5] RoboCup Federation, "RoboCup: International Competitions and Conferences on Robotics and AI," 2025. [Online]. Available: https://www.robocup.org

[6] K. Nonaka, T. Hasegawa, and Y. Kanayama, "A distributed control architecture for intelligent mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 51, no. 1, pp. 48–55, Feb. 2004, doi: 10.1109/TIE.2003.822042.

[7] M. Chen, Y. Zhang, and V. Leung, "Battery management system and SOC development for electrical vehicles," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 1, pp. 76–88, Jan. 2011, doi: 10.1109/TVT.2010.2089647.