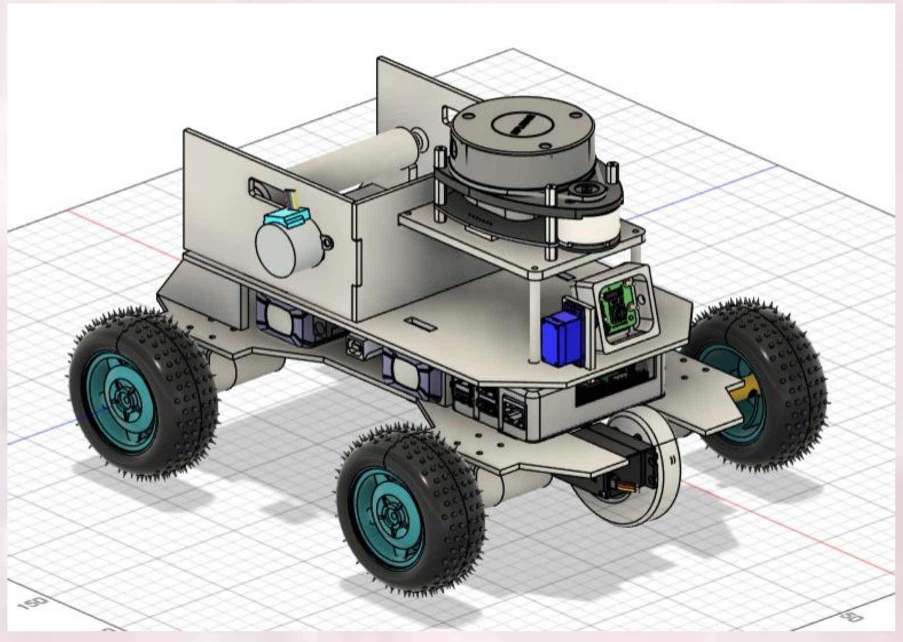
Tetsuzakura Knights ✿ Technical Poster

John Khaemba, Prudence Njoroge, Glenn Gatiba, Allen Wachio

Robotics Dojo 2025 Competition

ABSTRACT

This poster 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 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.



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.

The perception system integrates an RPLiDAR sensor for Simultaneous Localization and Mapping (SLAM), enabling real-time localization in changing environments .

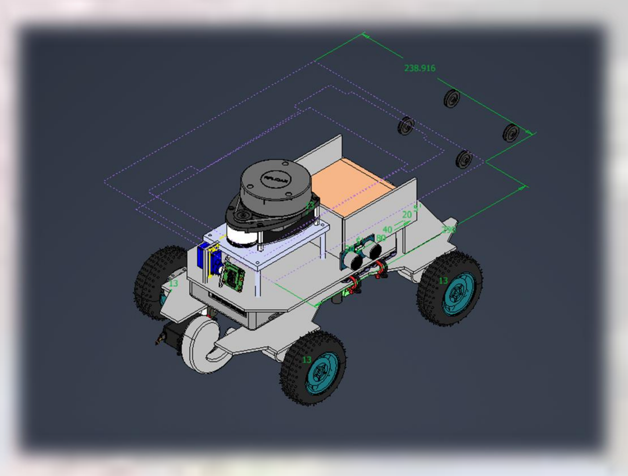
Complementary ultrasonic sensors, mounted laterally, together with an onboard IMU, provide diagnostic feedback for drift detection, immobilization monitoring, and LiDAR frame filtering to ensure map integrity. A color sensor mounted near the conveyor belt identifies the color of transported loads; this information is cross-referenced with input from a forward-facing Pi Camera to locate designated deposition zones. 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.

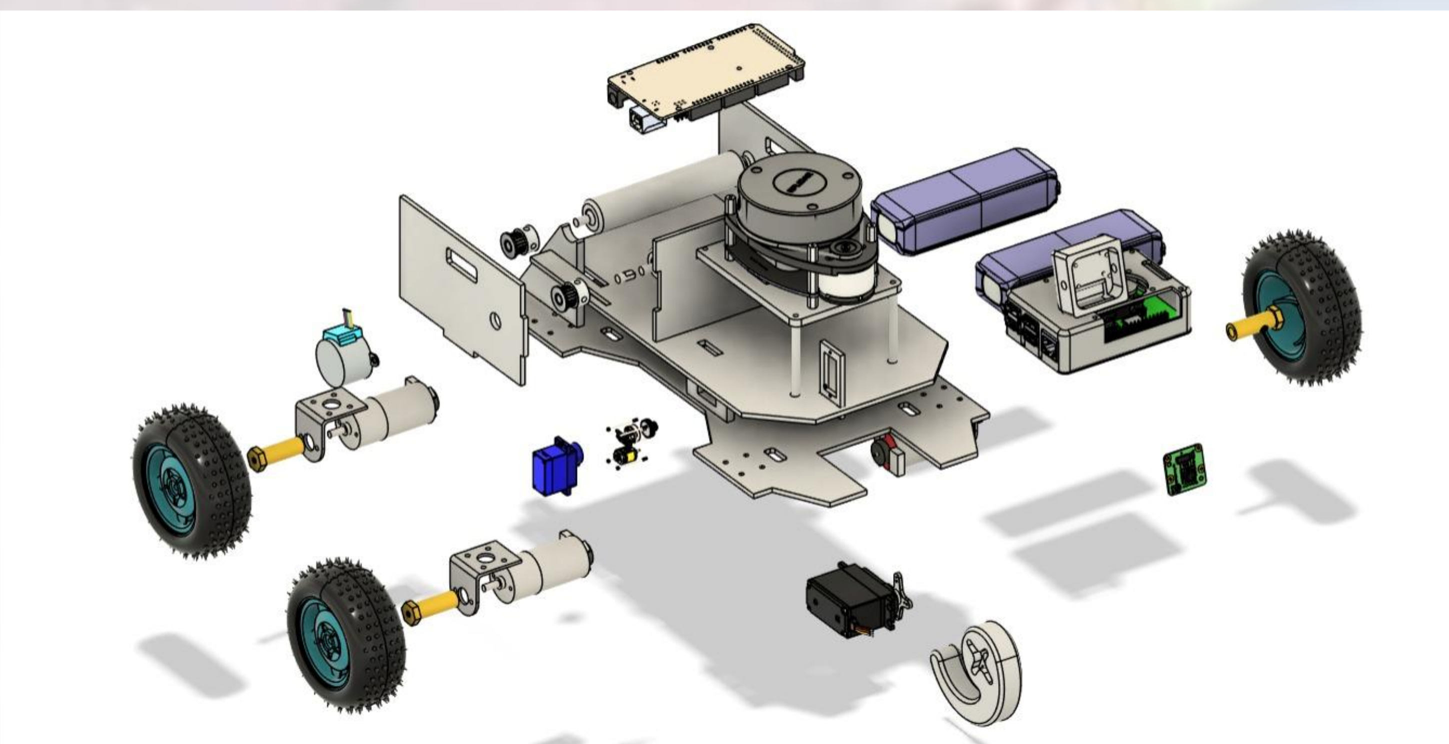
OBJECTIVES

- To design and fabricate a compact four-wheeled robot (300 × 300 × 300 mm) capable of reliably traversing rough and uneven terrain in competition environments.
- To implement LiDAR-based SLAM supported by multi-sensor fusion (IMU, ultrasonic sensors, and wheel encoders) for accurate mapping, localization, and orientation tracking.
- To integrate a color sensor for load identification, cross-referenced with a forward-facing Pi Camera for zone detection, enabling accurate deposition of loads in designated areas.
- To employ the Pi Camera, running an OpenCV pipeline with a pre-trained dataset, for plant health analysis (healthy/unhealthy leaf classification) at the operator start point.
- To implement a stepper-driven conveyor belt and a servo-actuated tipper mechanism for reliable object handling, delivery, and self-recovery when the robot becomes immobilized.
- To enable dual operational modes: teleoperation for subsystem validation and debugging, and autonomous navigation for competition tasks.
- To utilize the Nav2 stack within ROS2 for dynamic path planning, allowing the robot to autonomously re-route if its original path becomes blocked.
- To develop a modular hardware–software architecture combining Arduino Mega, Raspberry Pi, motor drivers, and sensors, powered by a LiPo/Lithium-ion battery system with a Battery Management System (BMS) for safe, reliable, and enduring operation.
- To leverage ROS2, URDF, Gazebo, and RViz for simulation, visualization, and testing, ensuring a smooth transition from virtual validation to physical deployment.
- To enhance system resilience and maintainability through modular hardware design, ultrasonic-assisted fault detection, quick recovery mechanisms, and structured software practices for efficient debugging.

VEHICLE DESIGN

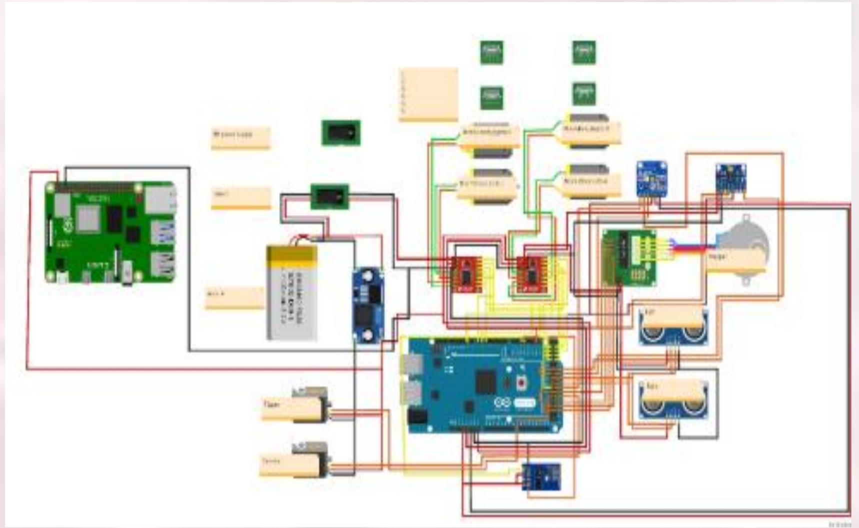
The design strategy of the proposed robotic vehicle, was guided by competition constraints and mission objectives. The robot was required to fit within a 300 × 300 × 250 mm bounding box, while being capable of traversing rough terrain, mapping the dynamic game field, and handling objects based on color recognition.

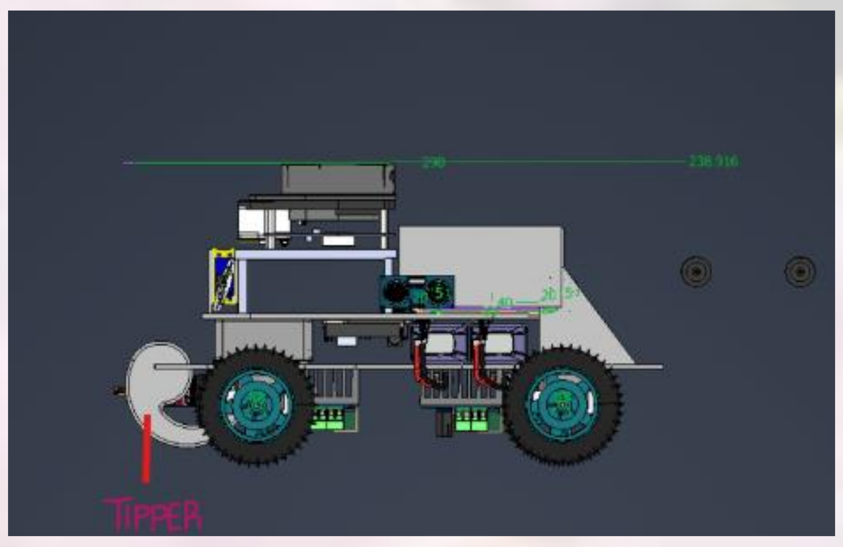




Object handling is performed by a 28BYJ-48 stepper-motor-driven conveyor belt that reliably transports and deposits loads; a high torque servo-actuated tipper provides recovery capability by lifting the front wheels to extricate the robot if it becomes immobilized.

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

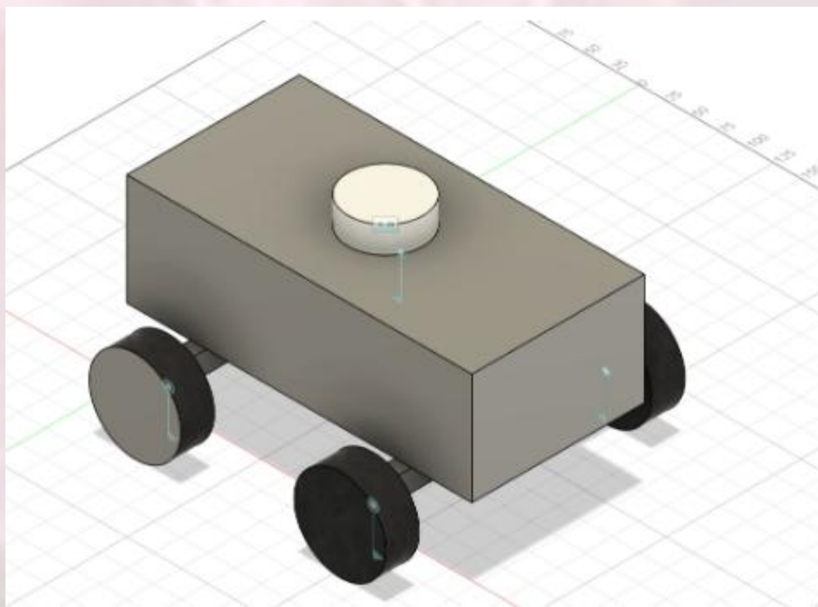




CONCLUSIONS

This project presented the design and development of a compact four-wheeled autonomous robot (300 × 257 × 207 mm) for the Robotics Dojo Competition 2025. The system combined mechanical robustness, reliable power management, and modular sensing to deliver autonomous operation in challenging environments. Core functions such as LiDAR-based SLAM, color-based load handling, and IMU–ultrasonic fusion for orientation monitoring enabled the robot to map, localize, and interact with the dynamic competition field. A stepper-driven conveyor belt and servo-actuated tipper mechanism provided reliable object handling and recovery from immobilization, while the Pi Camera extended the platform's capability with both color-zone detection and plant health analysis using a pre-trained dataset.

The adoption of ROS2 as the middleware, together with URDF modeling, RViz visualization, and Gazebo simulation, facilitated a streamlined development cycle and ensured smoother transition from virtual prototypes to physical implementation. Integration of the Nav2 stack further enhanced autonomy by enabling dynamic path planning and rerouting in response to obstacles or blocked paths.



Despite hardware challenges such as damaged motor drivers and charging failures, and software setbacks including instability of Gazebo Fortress on WSL, the team developed effective debugging workflows and rapid recovery strategies. These experiences strengthened our problem-solving skills, improved our understanding of multi-sensor integration, and highlighted the importance of resilience in both hardware and software design.

Ultimately, the project delivered a robust, versatile, and competition-ready robotic platform that successfully demonstrated autonomy, adaptability, and task-specific intelligence under real-world constraints.

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

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.