Technical Design Paper for the Robotics Dojo Competition 2026

Autonomous Waste Management Robot

Team 6, JNNP Robotics Innovators Club

Jeremiah Nyaga National Polytechnic, Kenya

Jane Kabuchi Maina (Team Leader; Software Lead) Munyao Nthenya (Team Member; Hardware Lead)
Ephantus Kariuki Kihara (Team Member; Electronics Lead) Francisco Mbugua Ngigi (Trainer; Robotics and Assistant Research Coordinator, Technical Documentation)

November 2026 · Source code: github.com/jnnproboticsinnovators/jnnp-robodojo-robot

***Abstract*—This paper presents the autonomous waste management robot of the JNNP Robotics Innovators Club for the Robotics Dojo Competition 2026. The robot uses a lightweight acrylic base, a four-wheel skid-steer drive with encoder gearmotors, and a two-tier controller: a Raspberry Pi 4 running Ubuntu 24.04 and ROS 2 Jazzy for perception and communication, and an Arduino Mega 2560 with a Monster Moto Shield for real-time motor control. A rear servo lifts and tips a bin to offload waste. A custom YOLO11s model, trained on 9,670 images, detects metal, organic and plastic waste on the Pi with an overall mAP50 of 0.77. Bench testing traced a turning failure to a reversed motor channel, which was corrected. The robot is teleoperated with automatic safety stops and simulated in Gazebo on the official competition field, as a base for autonomous waste collection.**

***Index Terms*—autonomous mobile robot, waste management, skid-steer drive, ROS 2, YOLO, edge AI, Gazebo, Raspberry Pi, Arduino.**

I. Introduction

Robotics Dojo is a robotics competition and training programme hosted at Jomo Kenyatta University of Agriculture and Technology (JKUAT) that encourages students across Kenya to design, build and program autonomous robots. This year's challenge centres on waste management: a mobile robot must move around a competition field that includes ramps and floor markings, find waste items, carry them and offload them at a dumping point, with as little human control as possible.

This paper presents the robot built by the JNNP Robotics Innovators Club of Jeremiah Nyaga National Polytechnic. It describes the design strategy, the mechanical, electrical and software design, the computer vision system that recognises metal, organic and plastic waste, and the tests carried out so far. This paper records the robot as rebuilt after the RoboDojo mini competition at JKUAT in October 2026.

II. Design Strategy

The team set four requirements for the robot:

- Move reliably over the competition field, including turning on the spot, under teleoperation now and autonomously as the software matures.
- Detect waste items with a camera, classify them as metal, organic or plastic, and locate them relative to the robot.
- Carry collected waste and offload it at a dumping point.
- Stop safely on its own when communication is lost or an obstacle is too close.

The robot was divided into the following subsystems:

Robot frame

An acrylic base and top deck. The first build used a welded steel chassis; it was replaced by acrylic to reduce mass, halving the robot from 4 kg to 2 kg, because a skid-steer robot needs more wheel torque to turn the heavier it is.

Offloading mechanism

A bin at the rear of the robot, lifted and tipped by a single TowerPro MG995 servo to empty collected waste, then lowered again.

Drivetrain and propulsion

Four 12 V, 400 rpm GA25-370 encoder gearmotors in a skid-steer layout: the two motors on each side are wired in parallel and driven by one H-bridge channel, and the robot turns by running the two sides in opposite directions, so the wheels skid sideways in every turn [10].

Wheel mounts and wheels

Metal motor brackets bolted to the base, with 85 mm × 38 mm rubber wheels for grip and ground clearance.

Control scheme

Two tiers: a Raspberry Pi 4 for perception, decisions and communication, and an Arduino Mega 2560 [1] with a Monster Moto Shield [2] for real-time motor control, encoder counting and the servo.

Environment detection devices

An RPLIDAR A1M8 [13] for obstacle detection and mapping, a Raspberry Pi Camera Module 3 [14] for waste detection, and encoders on the two rear motors for wheel speed and distance.

Power supply

A single 4S lithium-polymer battery (14.8 V nominal) feeding four buck converters, so motor current cannot disturb the computers.

Mapping, localisation and navigation

- **Communication:** the Pi runs Ubuntu Server 24.04 LTS with ROS 2 Jazzy [5], [7] and publishes the LiDAR scans, wheel odometry and drive commands as standard ROS 2 topics; the laptop joins over Wi-Fi through Zenoh [16].
- **Simulation:** the robot is modelled in Gazebo [6] on the official competition field, for safe testing of driving and code.

- **Navigation:** the real robot has mapped the field with SLAM Toolbox [8]; Nav2 [9] is installed on the Pi for the next stage, autonomous navigation.

The general design steps were:

1) Define the requirements and divide the robot into subsystems.
2) Select the frame material, motors, controllers and sensors.
3) Build the frame and mount the drivetrain.
4) Design the power distribution and wiring, and allocate microcontroller pins.
5) Write and test the motor control firmware on the Mega.
6) Set up the Raspberry Pi, wireless access and the serial link to the Mega.
7) Build arrow-key teleoperation with automatic safety stops.
8) Integrate the LiDAR and camera.
9) Train and deploy the waste-detection model.
10) Model the robot in ROS 2 and simulate it in Gazebo.
11) Test each subsystem, then the whole robot, and record the results.

III. Vehicle Design

The vehicle design covers three areas: the mobile platform, the navigation and control system, and the computer vision system. Compared with the robot taken to the mini competition, the rebuilt robot has a lighter base, one battery instead of two, corrected motor wiring, a dedicated offloading servo, ROS 2 on the Pi and an AI waste detector.

1. Mobile Platform

Frame

Steel was chosen for the first build for its strength and welded joints. In the rebuild, mass became the deciding factor: acrylic is light, which lowers the torque needed to turn and reduces battery drain, easy to cut and drill, and strong enough for a small robot when holes are drilled carefully and bolts use washers.

The battery, the heaviest part, sits near the centre of the base so all four wheels share the load and skid evenly when the robot turns.

Drivetrain and motor wiring

The front and rear motor on each side are wired in parallel to one channel of the Monster Moto Shield, which carries two VNH2SP30-E H-bridges [3]. With 85 mm wheels at 400 rpm, the theoretical no-load top speed is about 1.78 m/s (π × 0.085 m × 400 rpm ÷ 60).

TABLE I
MOTOR CONNECTIONS TO THE MONSTER MOTO SHIELD

Side	Red (+) lead	White (−) lead
Left (LF + LR)	OUT A1	OUT B1
Right (RF + RR)	OUT B2	OUT A2

The right pair is wired opposite to the left because the right motors are mounted as a mirror image of the left ones (see Section IV).

Main components

- Motor driver: Monster Moto Shield (2 × VNH2SP30-E)
- Microcontroller: Arduino Mega 2560
- Computer: Raspberry Pi 4
- Motors: 4 × GA25-370 12 V, 400 rpm encoder gearmotors
- Servo: TowerPro MG995, metal gear, on its own 6 V rail
- IMU: MPU6050 on I2C (firmware v11)
- LiDAR: RPLIDAR A1M8; camera: Raspberry Pi Camera Module 3
- Battery: 1 × 4S LiPo, 14.8 V nominal, 16.8 V full
- Buck converters: 4, set to 5.11 V, 6.0 V, 5.0 V and 12.0 V
- Wheels: 4 × 85 mm × 38 mm rubber

Power distribution

The battery feeds one switched bus. Each converter supplies one group of loads, and all negatives join a single ground bus.

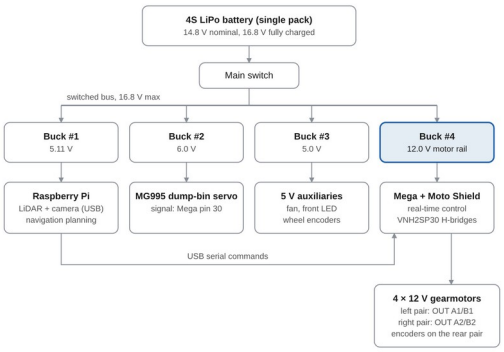


Fig. 1. Power and control architecture.

TABLE II
CONVERTER RAILS

Converter	Output	Load
Buck #1	5.11 V	Raspberry Pi
Buck #2	6.0 V	Dump-bin servo
Buck #3	5.0 V	Encoders, fan, front LED
Buck #4	12.0 V	Motor shield (all four motors)

Pin allocation

TABLE III
ARDUINO MEGA 2560 PINS

Pin	Connected to	Function
4, 7, 8, 9	Shield INA/INB	Motor direction
5, 6	Shield PWM1, PWM2	Motor speed
A0, A1	Shield EN/DIAG	Driver enable, fault
A2, A3	Shield CS	Motor current sense
18 / 22	Left-rear encoder A / B	Interrupt / direction
19 / 24	Right-rear encoder A / B	Interrupt / direction
30	Servo signal	Servo pulse
26	Status LED	Steady moving, blink stopped
28, 34	Reverse LEDs	Alternate blink in reverse
20 / 21	MPU6050 SDA / SCL	Gyro (firmware v11)
USB	Raspberry Pi	Serial link, 115200 baud

Hardware challenges

- **Robot drove straight but would not turn.** Both motor pairs were first wired alike. Bench tests with the wheels off the ground showed the right side ran backwards to every command; swapping the right pair's leads fixed it (Section IV).
- **Wiring fault before the mini-competition dry run.** A connection loosened in transport and stopped the robot starting. All joints are now crimped or soldered, heat-shrunk, labelled and tug-tested, and the robot is checked before travel and on arrival.
- **LiDAR drop-outs.** The Pi was under-powered (throttled = 0x50005), which made the LiDAR drop out. A stable 5 V, 3 A supply fixed it (throttled = 0x0). ModemManager was also disabled so it cannot claim the LiDAR's USB port.

2. Navigation and Control System

The laptop reaches the Raspberry Pi over Wi-Fi by SSH, and the Pi commands the Mega over USB serial at 115200 baud. The robot runs in two modes: direct teleoperation, in which the Pi talks straight to the Mega and the LiDAR with no ROS in between, and ROS 2 mode, used for mapping and navigation. In both, the Mega has its own watchdog: if commands stop, for example when the Wi-Fi drops, it stops the motors within 0.5 s.

Single-letter serial commands drive the Mega: w/s forward and backward, a/d spin, x brake, +/− speed, u/n bin up and down, and r to reset the encoders.

The drive firmware adds a short kick-start burst to overcome static friction, boosts the slower side using encoder counts so the robot tracks straight, brakes actively on stop, and spins on the spot by running the two sides at full power in opposite directions.

A Python teleoperation program on the Pi drives the robot with the arrow keys (↑ forward, ↓ backward, ← left, → right, Space stop), with U and N for the bin and C to toggle the camera. Two safety stops are built in: the robot stops if no movement key arrives for 0.6 s, and an optional LiDAR stop blocks forward motion when an obstacle is too close.

For autonomy, the real robot runs as a ROS 2 robot. One program on the Pi owns the serial link and the LiDAR: it publishes LiDAR scans on /scan and rear-wheel odometry on /odom, and converts /cmd_vel velocity commands into left and right wheel speeds, using an exact wheel-speed command added in Mega firmware version 10 alongside the letter commands. The laptop and Pi exchange ROS 2 messages through Zenoh (rmw_zenoh) [16], with a router on the Pi that starts at power-on and both clocks synchronised to internet time. In testing, the arrow-key teleoperation on the laptop drove the real robot through ROS 2 and the odometry reported correctly, so any ROS 2 tool, from keyboard teleoperation to Nav2, can drive the robot without firmware changes.

The robot's URDF model runs in Gazebo Harmonic on the official competition field mesh, with a differential-drive plugin and a camera sensor, and uses the same arrow-key layout as the real robot.

3. Computer Vision System

Waste is detected with a YOLO11s object-detection model trained by the team. YOLO detectors find and classify every object in a single pass of the network, which suits real-time use on small computers [11], and litter datasets such as TACO show that waste in real scenes varies widely in shape and appearance [12]. It was trained with Ultralytics 8.3.40 [4] from the pretrained yolo11s.pt weights, on a Tesla T4 GPU in Google Colab, for 30 epochs (1.54 hours). The RDJ2026 Roboflow dataset (version 3) has 9,670 labelled images at 640 × 640 pixels in three classes (metal, organic and plastic); 8,703 were used for training and 580 for validation. The best weights were exported to ONNX (36 MB) for the Pi.

On the robot, the model runs with ONNX Runtime [15], which needs no PyTorch and is light enough for the Pi's processor. The pipeline is:

1) The Pi Camera Module 3 is read through libcamera at 640 × 480 pixels; a background thread always keeps the newest frame, which is scaled to 640 × 640.
2) The model returns a box, class and confidence for each item; detections above 40% confidence are kept and duplicates removed by non-maximum suppression (IoU 0.45).
3) OpenCV finds each box's centre and marks the item as LEFT, CENTRE or RIGHT of the view.
4) Box size estimates distance: a larger box means a closer item.
5) The annotated video is streamed to a browser on the laptop.

A frame takes about 0.6 to 1.4 s on the Pi (about 94 ms on a laptop CPU). The detector runs as a system service that is off at power-on, because it uses most of the Pi's processor and slows the controls; the operator switches it on and off with the C key during teleoperation. The planned autonomous behaviour follows from it: turn towards an item that is LEFT or RIGHT, drive forward while it is CENTRE, stop when the box is large or the LiDAR reports the item close, then drive to the dumping point and tip the bin.

IV. Experimental Results

Testing so far has covered the drivetrain, the offloading servo, the waste-detection model, driving the robot through ROS 2, and dry runs on the competition field.

Drivetrain

Each movement was tested on blocks, with the wheels off the ground. At first every turn command drove the robot straight and every straight command made it spin: the right channel was inverted. After swapping the right pair's leads, forward, backward and both spins were correct on blocks and on the floor. Straight driving alone does not prove the wiring; every movement must be tested.

Offloading servo

The servo moved the bin between its home and dump angles on command, lifting and tipping a wooden cuboid test load in about 1 s; it lifts loads of up to 0.5 kg.

Waste detection

TABLE IV
DETECTION ACCURACY ON THE VALIDATION SET (580 IMAGES, 1,011 ITEMS)

Class	P	R	mAP50	mAP50-95
Metal	0.84	0.86	0.90	0.73
Plastic	0.74	0.75	0.82	0.55
Organic	0.73	0.48	0.59	0.34
All classes	0.77	0.70	0.77	0.54

Metal and plastic are detected reliably (P = precision, R = recall). Organic waste varies widely in shape and colour and is the weakest class, with about half of organic items missed (recall 0.48); more organic images, taken under the arena's lighting, will be added before retraining.

Driving through ROS 2

With the ROS 2 program on the Pi, /scan, /odom and /cmd_vel were visible from the laptop, and the robot drove through ROS 2 over Wi-Fi with correct odometry.

Dry runs on the competition field

TABLE V
DRY-RUN RESULTS

Test	Result
Robot mass	2 kg (first build: 4 kg)
Ground speed	0.15–0.60 m/s
360° spin on the spot	13 s (mean of two runs)
Drive speed setting	PWM 120 of 255 (47%)
Field runs, teleoperation	74 s, 42 s, 78 s (mean 65 s)
Field runs, SLAM mapping	122 s, 135 s
Terrain	Ramp, sand, runway and obstacles crossed
Bin lift and dump	About 1 s; lifts loads up to 0.5 kg
Battery, 4-hour session	16.3 V to 15.13 V (full: 16.8 V)

Under teleoperation the robot performed well on the ramp, the sand, the runway and around the obstacles. Mapping runs took longer because the robot is driven slowly so that SLAM can match each LiDAR scan. During the first mapping runs, turns swung about 30° away from the intended heading; running the turn-calibration routine corrected this, and adding the MPU6050 gyro improved the quality of the SLAM map. Over a 4-hour session the battery fell from 16.3 V to 15.13 V, or 3.78 V per cell, well above the 3.3–3.5 V per cell at which LiPo cells are damaged.

PWM 120 of 255 gave good control on the field. In tests with real items on the field, the detector found all the test objects.

V. Acknowledgement

The team thanks the Chief Principal of Jeremiah Nyaga National Polytechnic, Mr. Silvano Mureithi, and the JNNP administration for their support; the Robotics Dojo organisers at JKUAT for the competition and training; Dedan Kimathi University of Technology for training in IoT and TinyML; and TME for the technology days that connected the club to industry.

VI. References

[1] Arduino, "Arduino Mega 2560 Rev3," Arduino Documentation. [Online]. Available: https://docs.arduino.cc/hardware/mega-2560

[2] SparkFun Electronics, "Monster Moto Shield," GitHub. [Online]. Available: https://github.com/sparkfun/Monster_Moto_Shield

[3] STMicroelectronics, "VNH2SP30-E automotive fully integrated H-bridge motor driver," datasheet. [Online]. Available: https://www.st.com/

[4] Ultralytics, "Ultralytics YOLO documentation." [Online]. Available: https://docs.ultralytics.com/

[5] Open Robotics, "ROS 2 Jazzy Jalisco documentation." [Online]. Available: https://docs.ros.org/en/jazzy/

[6] Open Robotics, "Gazebo Harmonic documentation." [Online]. Available: https://gazebosim.org/docs/harmonic/

[7] S. Macenski, T. Foote, B. Gerkey, C. Lalancette and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," Science Robotics, vol. 7, no. 66, 2022.

[8] S. Macenski and I. Jambrecic, "SLAM Toolbox: SLAM for the dynamic world," Journal of Open Source Software, vol. 6, no. 61, p. 2783, 2021, doi: 10.21105/joss.02783.

[9] S. Macenski, F. Martín, R. White and J. Ginés Clavero, "The Marathon 2: A navigation system," in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), 2020.

[10] A. Mandow, J. L. Martínez, J. Morales, J. L. Blanco, A. García-Cerezo and J. González, "Experimental kinematics for wheeled skid-steer mobile robots," in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), 2007, doi: 10.1109/IROS.2007.4399139.

[11] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.

[12] P. F. Proença and P. Simões, "TACO: Trash annotations in context for litter detection," arXiv:2003.06975, 2020.

[13] Slamtec, "RPLIDAR A1M8 datasheet and user manual." [Online]. Available: https://www.slamtec.com/

[14] Raspberry Pi Ltd., "Camera documentation." [Online]. Available: https://www.raspberrypi.com/documentation/accessories/camera.html

[15] Microsoft, "ONNX Runtime documentation." [Online]. Available: https://onnxruntime.ai/docs/

[16] ROS 2, "rmw_zenoh: ROS 2 middleware based on Zenoh." [Online]. Available: https://github.com/ros2/rmw_zenoh

VII. Appendix – Situational Awareness

The team commits to the following so that operators and judges understand and can trust the robot:

- **Autonomy:** the robot is currently driven by teleoperation with automatic safety stops; autonomous waste collection is being developed on the detection and ROS 2 systems described above.
- **System components:** all hardware, wiring and software are documented in this paper and the team's build guide.
- **User role:** an operator supervises every run, can stop the robot at once with the Space key or by quitting teleoperation, and can cut all power with the main switch.
- **Validation data:** detection accuracy is reported per class with its confidence level (Table IV); organic waste is the least reliable class.
- **Fault conditions:** loss of communication stops the robot within 0.5 s (Arduino watchdog); the LiDAR stop blocks forward motion near obstacles; a fuse and battery-voltage monitor are being added.
- **Corner cases:** poor lighting, cluttered backgrounds and unusual organic items can cause missed or wrong detections; ramps and floor markings are tested in simulation and on the field.
- **Nature of the robot:** it is a machine with no awareness of people, and must not be run near people without supervision.