AN-NAJAH NATIONAL UNIVERSITY

Computer Engineering Department

Hardware Graduation Project

Automatic Hall Guide Robot
- Igris -

Students
Omar Maher Khatib and Osaid Islam Raddad

Under the supervision of
Dr. Manar Qamhieh

A report submitted in partial fulfillment of the requirements for
bachelor's degree in computer engineering in the Faculty of Engineering
Information Technology - Hardware Project
September 2025

Acknowledgements

We would like to express our sincere gratitude to Dr. Manar Qamhieh for her invaluable guidance and continuous support throughout this project, our heartfelt thanks go to our families and friends for their love, encouragement, and unwavering support across the past four years, We also extend our appreciation to everyone who contributed to our learning journey, special thanks to Iyad Ayyash for his hands-on help and contributions to the robot’s chassis design.

Disclaimer

This report was written by Osaid Raddad and Omar Khatib at the Computer Engineering Department, Faculty of Engineering, An-Najah National University. It has not been altered or corrected, other than editorial corrections, as a result of assessment and it may contain language as well as content errors. The views expressed in it together with any outcomes and recommendations are solely those of the student(s). An-Najah National University accepts no responsibility or liability for the consequences of this report being used for a purpose other than the purpose for which it was commissioned.

Contents

List Of Figures

Abstract

This project presents an autonomous mobile robot designed to guide students and visitors to specific classrooms or labs within a single floor of a university building. The importance of this project lies in enhancing student experience, especially for new students who may struggle to locate rooms in complex campus buildings. The robot aims to reduce confusion, save time, and demonstrate the integration of robotics in educational spaces.

Key aspects of the project include autonomous navigation using a lidar, real-time obstacle avoidance, room recognition, user interaction through a touchscreen, and voice feedback via a speaker. The robot also features tilt detection for safety and a small internal bin for plastic bottle collection to promote environmental awareness. It can be remotely monitored and send alerts via Telegram, such as notifications about falls or tilts, or reaches its destination.

The primary objective is to build a functional, cost-effective, and interactive robot that receives a room number from the user and autonomously guides them to the correct location. Upon arrival, it confirms the destination through voice and display.

The methodology involves system design, component selection (Raspberry Pi, Arduino Mega, Arduino UNO, NEMA 23 motors, TB6600 drivers, RPLiDAR-A1, ultrasonic sensors, MPU6050, 12V Battery, Camera), and software development using ROS2 (Robot Operating System), Python and Arduino C++. Slam_toolbox will create a map that will be used for navigation, with real-time obstacle detection and dynamic path updates handled by ROS2, onboard sensors and high-quality camera.

The robot was successfully implemented and tested in the university's basement floor corridors, where it achieved reliable navigation and guided users accurately to selected rooms.

While similar robots exist in commercial sectors such as shop robot "Navii", this implementation is unique in its educational focus, simplicity, and other features.

Chapter 1 — Introduction

1.1 Background

Wayfinding across large university buildings is challenging for newcomers, visitors, and even returning students when rooms and labs are dispersed along long corridors with complex numbering. Traditional signs and ad-hoc assistance are not always available during peak hours. Recent advances in mobile robotics and cost-effective 2D LiDAR/ultrasonic sensing now enable indoor assistive systems that can guide users in real time with minimal infrastructure changes. **This project addresses that gap by delivering an on-demand, corridor-safe guidance robot for single-floor deployments.**

1.2 Problem Statement

Students and visitors often waste time searching for rooms, causing delays, confusion, and congestion near information desks—especially at semester start and during exams. A practical solution should provide on-demand guidance inside corridors, operate safely around people and obstacles, and avoid building modifications.

1.3 Project Objectives

The Hall Guide Robot aims to:

- Provide indoor guidance to a user-specified room **within a single floor**.
- Perform **autonomous navigation** using 2D LiDAR with **ultrasonic** for safety.
- Achieve **real-time obstacle detection/avoidance** with smooth path updates.
- Offer **kiosk-based touchscreen** interaction and **voice feedback**.
- Support **remote monitoring/alerts** (e.g., tilt/fall, destination reached).
- Include a **bottle-collection** feature to promote environmental awareness.
- Maintain **cost-effectiveness and reliability** using COTS hardware and open-source software.
- Allow **dynamic mapping** when deployed on a different floor (no re-engineering).

1.4 Scope and Limitations

- **Scope:** Single-floor operation; the robot accepts a room number, plans a corridor route, and accompanies the user to the destination with audio confirmation and on-screen prompts.

- **Limitations:** No elevator/stair handling; performance may vary with corridor width, people density, and glossy/reflective surfaces. Vision-based recognition is lighting-dependent. Battery runtime constrains operation without scheduled charging.

1.5 Significance and Contributions

This system improves on-campus experience by reducing confusion and saving time for freshmen and visitors, while demonstrating a practical integration of mobile robotics, perception, and human–robot interaction on commodity hardware. **Contributions** include: (i) a replicable mechanical design, (ii) a ROS 2–based navigation stack tailored to single-floor guidance with dynamic mapping for new floors, and (iii) a user-friendly kiosk interface with multilingual options and a sustainability-oriented bottle bin. The approach emphasizes **low cost**, **safety in human environments**, and **ease of deployment** by avoiding building changes.

1.6 Report Organization

- **Chapter 1** introduces the background, problem, objectives, scope, and significance.
- **Chapter 2** reviews foundational concepts (indoor localization, SLAM, LiDAR sensing, path planning) and related work.
- **Chapter 3** details hardware choices, software architecture, safety, and constraints (economy, environment, ethics/privacy, health & safety, manufacturability, sustainability).
- **Chapter 4** presents the experimental setup and results (key metrics, error estimates, figures/tables).
- **Chapter 5** interprets the results, discusses limitations, and relates them to prior work.
- **Chapter 6** concludes with recommendations and future directions.

Chapter 2 — Theoretical Background & Previous Work

2.1 Indoor Wayfinding and Service Robotics

Wayfinding in public buildings (universities, hospitals, malls) requires reliable local perception, robust localization under occlusions, and human-aware motion. Service robots in such settings should be **legible** (clear intent), **predictable** (smooth, low-jerk paths), and **safe** (fail-safe stops, conservative speeds near people). For single-floor guidance, corridor geometry, door density, junctions, and dynamic obstacles (people, carts) dominate design choices.

Commercial deployments (e.g., in retail concourses) indicate feasibility, while many academic systems adopt open-source middleware (ROS/ROS 2) with 2D LiDAR as the primary range sensor. Compared to logistics AMRs, a hall-guide system optimizes **human interaction and clarity of motion** rather than payload or fleet coordination.

2.2 Sensing Fundamentals (LiDAR, IMU, Ultrasonic, Camera)

2D LiDAR (e.g., RPLIDAR-A1). Emits laser pulses and measures time-of-flight to form polar scans (range, optional intensity). Strengths: angular precision, ambient-light immunity, a thin vertical slice that cleanly observes corridor edges/doors. Limitations: specular/absorptive surfaces (glass, glossy paint), occlusion by legs/furniture. Mounting height should capture wheel-level and knee-level obstacles; a slight forward offset reduces self-hits.

IMU (6-axis). Short-term yaw stabilization and tilt/fall detection. Gyro helps bridge LiDAR dropouts; accelerometer thresholds enable inclination alarms and E-stop triggers.

Ultrasonic. Short-range cones complement LiDAR in the **near field** (stair lips, table edges) and on low-reflectance materials. Requires debouncing/outlier rejection to avoid spurious zeros/inf.

Camera. Used for room-number OCR or fiducials (AprilTag/ArUco) and for UI feedback. In guidance-first designs, vision is typically **secondary for navigation** but **primary for destination recognition**.

2.3 Mapping, Localization, and SLAM (2D)

Indoor navigation can rely on: (i) **pre-built static maps** with online localization, (ii) **online SLAM** (build + navigate), or (iii) **hybrid** updates. A 2D **occupancy grid** encodes free/occupied/unknown cells at fixed resolution (≈0.02–0.05 m). Localization aligns live scans to the map via scan matching (e.g., correlative/ICP) optionally fused with odometry/IMU. SLAM adds a **pose-graph** to refine map/trajectory.

For single-floor university corridors with largely stable layouts, **static map + robust localization** yields predictable behavior with lower compute and fewer drift issues, while dynamic obstacles are handled in the **local costmap**.

2.4 Navigation in ROS 2 (Nav2 Overview)

Nav2 decomposes navigation into: **Map Server**, **Localization**, **Global Planner**, **Local Controller** (e.g., DWB or Regulated Pure Pursuit), **layered costmaps** (static + obstacle + inflation), and **Behavior Trees** (task orchestration/recoveries). Parameters such as inflation radius, obstacle persistence time, controller gains, and planner tolerances should match corridor widths and turning radii. Legible motion typically caps accelerations, limits curvature, and reduces speed near crowds/intersections.

2.5 Sensor Fusion and Odometry

With stepper-driven wheels, **open-loop step counts** can serve as proxy odometry; drift accumulates under wheel slip or calibration error (wheel radius/track width). Fusion approaches (e.g., complementary/EKF of odom + IMU + scan-matching) reduce drift; periodic **scan-to-map** corrections re-anchor pose. Good practice: calibrate wheel geometry, limit accelerations, and use traction-aware motion (slower turns, S-curves) on smooth tiles.

2.6 Human–Robot Interaction (HRI)

A campus guide must communicate intent clearly:

- **Touchscreen UI** with large bilingual buttons, room search, and progress feedback (e.g., “Guiding to B1021…”).
- **Audio prompts** for start/arrival and safety notices.
- **Visible state cues** for ready/paused/error.
- **Remote supervision** via a light-weight web dashboard. Accessibility: screen height, readable fonts, and safe stopping distances.

2.7 Safety, Compliance, and Operational Boundaries

Safety spans hardware and software:

- Physical and software **E-stop** (zero velocity immediately).
- **Edge/step protection**; conservative speed near stairs.
- Speed limits based on corridor occupancy; reduced speed at junctions.
- **Geofencing** within the known floor map.

- **Tilt/fall detection** with automatic stop and alert. Operational procedures: daily pre-flight, supervised peak-time trials, periodic map validation.

2.8 Prior and Related Work (Brief Survey)

- **Commercial service robots** in public spaces commonly employ 2D LiDAR, human-aware speeds, and large display UIs.
- **Academic guide robots** often use static maps with AMCL-style localization; some adopt fiducial tags to avoid OCR failures on door signs.
- **Hospital/office delivery robots** handle coexistence with people and sometimes elevators—offering lessons for costmaps and recovery behaviors. **Key gaps addressed here:** low-cost BOM; single-floor corridor focus; bilingual, touch-first UI; sustainability via an integrated bottle-bin.

2.9 Design Implications for This Project

- **Mapping strategy:** static map + online localization; allow limited SLAM updates when layout changes.
- **Sensor roles:** 2D LiDAR as primary; ultrasonics for edges/cliffs; IMU for yaw/tilt; camera for room signs/fiducials.
- **Nav2 configuration:** tuned inflation/persistence; smooth, low-jerk controller; BT-managed recoveries.
- **Human-aware kinematics:** clear, legible motion with audio/visual cues.
- **Safety envelope:** geofencing, dual E-stops, conservative speeds, and alerts.
- **Maintainability:** parameterized YAMLs; modular microcontroller code; documented wiring; diagnostics.

2.10 Chapter Summary

This chapter outlined the theoretical foundations and briefly surveyed related work, motivating the design selections applied later in **Chapter 3 (Methodology)**.

Chapter 3 — Methodology

This chapter documents the materials, standards, procedures, and safety practices used to build the Hall Guide Robot so that an experienced practitioner can reproduce the system and obtain comparable results.

3.1 System Overview

The robot comprises a Raspberry Pi 4 (ROS 2; initial bring-up on Humble, later migrated to **Jazzy** on Raspberry Pi), an Arduino Mega for motor control, and an Arduino UNO for ultrasonic sensing. Mobility uses four stepper motors (**final choice: NEMA-23**) driven by TB6600 drivers. Perception includes an **RPLIDAR A1** for 2D range scanning, a 6-axis IMU for tilt/yaw, an **OAK-D** camera for room-number OCR, and **four ultrasonic sensors** for edge/stair safety. A **7-inch touchscreen** and a lightweight web panel provide user interaction; voice prompts are played through a speaker. Power is supplied by a **12 V, 8.4 Ah lead-acid battery** with a **12→5 V buck** for logic, and an **external power bank** that feeds the Pi, OAK-D, and speakers (kept in the final build to offload the lead-acid battery).

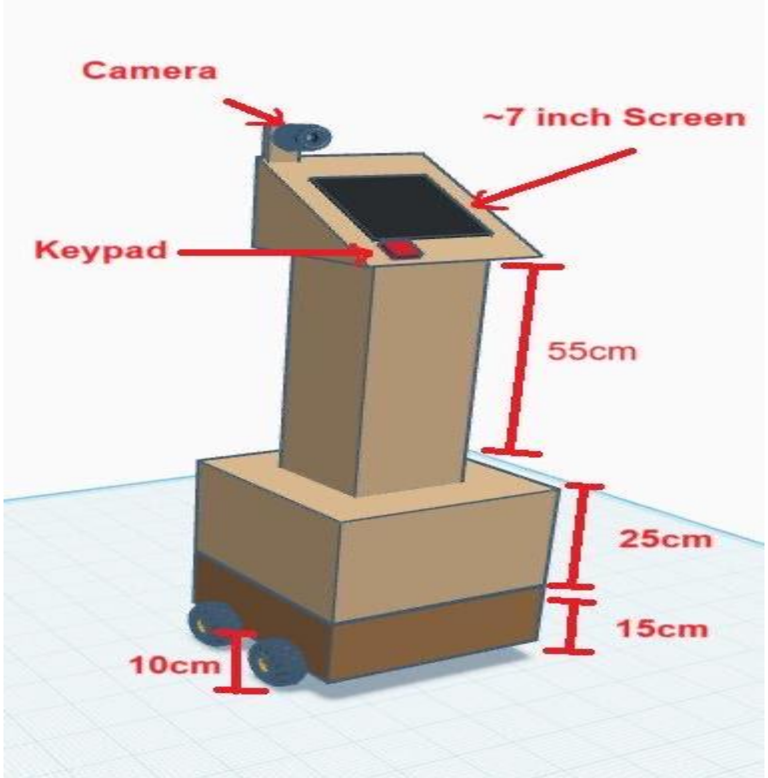


Figure 3-1 Early 3D Concept of the Hall Guide Robot

3.1.1 Compute & Control

- **High-level compute:** Raspberry Pi 4 running ROS 2 (Humble → Jazzy), hosting Nav2, slam_toolbox, UI, and telemetry services.
- **Motor control:** Arduino Mega generating step/dir to TB6600 drivers; serial protocol bridges ROS `cmd_vel` to wheel commands.
- **Ultrasonic bridge:** Arduino UNO dedicated to polling four ultrasonic sensors and publishing safety flags.
- **Communication:** USB CDC links (Pi↔Mega/UNO); ROS topics/services exposed on the Pi.

3.1.2 Mobility Subsystem

- **Actuators:** 4× NEMA-23 steppers with 4× TB6600 drivers (micro-step).
- **Drive:** differential (left/right); traction-aware limits for smooth tiles and turns.
- **Geometry:** wheel radius r = 12 cm; effective track width dynamic
- **Chassis:** MDF 0.5 cm walls; reduced mass vs. early 25 cm box, rigid mounts for drivetrain and mast.

Figure 3-2 Final mobility hardware layout

3.2 Bill of Materials (BOM) and Rationale (key items)

- **Raspberry Pi 4** (ROS 2 host): runs navigation, UI, and perception nodes.
- **Arduino Mega + 4× TB6600 + 4× NEMA-23:** deterministic step/dir motor control with sufficient torque under load (NEMA-17 proved insufficient in early tests; see ***Figure 5-2***).
- **Arduino UNO:** handles ultrasonic sensors and safety gating.
- **RPLIDAR A1:** primary 2D range sensor for mapping, localization, and obstacle avoidance.
- **OAK-D camera:** door-plate OCR (room numbers). Bottle detection was prototyped but excluded due to CPU load on the Pi; OCR retained.
- **IMU (6-axis):** yaw aiding and tilt/fall detection.
- **Ultrasonic sensors (×4):** near-field edge/stair protection (front and rear on the main box).
- **7″ touchscreen + speaker:** on-robot kiosk and audio feedback.
- **2× servo motors:** one operate the bottle-bin door the second operate camera direction(right/left).
- **12 V 8.4 A lead-acid + buck 12→5 V + power bank:** traction drivers and logic rails (power bank supplies Pi/OAK-D/speakers).

3.3 Standards and Specifications (Codes)

- **Wireless:** IEEE 802.11 via mobile phone hotspot (tethering) for telemetry and remote supervision.
- **Robotics conventions:** ROS 2 REP-105 coordinate frames for `map → odom → base_link → {laser, imu_link, camera_link}`.
- **Electrical:** fused low-voltage DC distribution; color-coded harnesses; common ground; isolation between logic and motor rails.
- **Operational safety:** capped speeds in crowded corridors; geofenced operation to approved areas; emergency stop behavior.

3.4 Mechanical Design and Revisions

The frame is **MDF**. The initial base used two stacked boxes (25 cm and 15 cm). Load testing showed excessive mass, so the 25 cm box was removed and **MDF thickness changed to 0.5 cm** to reduce weight while keeping stiffness via corner brackets.
Figure 3-2 shows the final internal layout with NEMA-23 motors and TB6600 drivers and fused distribution inside the base; wiring grouped via lever terminals. This replaced earlier NEMA-17 units to meet torque requirements.

See **Figure 3-13** for the pre-final chassis and **Figure 3-14** for the final look.

3.5 Electronics and Wiring

- **Power path:** Battery → **main ON/OFF switch** → fuse block → TB6600 drivers; Battery → buck 12→5 V → Raspberry Pi & peripherals; power bank → Pi/OAK-D/speakers.

- **Main power switch:** latching **ON/OFF toggle** mounted on the base exterior, **in series with the battery positive** before the fuse block. The switch provides a single-action hardware isolation for service and emergencies. ***(Figure 3-3)***.

- **Signal path:** Pi USB ↔ Arduino Mega/UNO; Mega step/dir/enable → TB6600; RPLIDAR via USB; IMU via I²C.
  Harnessing uses twisted pairs for step/dir lines, a shielded route for LiDAR, and strain relief at bulkhead pass-throughs.

Figure 3-3 Main On/Off switch ***(external view) and placement on the base***

3.6 Sensing and Perception

- **LiDAR (RPLIDAR A1):** publishes `/scan`; used by SLAM/localization and obstacle layers.
- **IMU:** tilt/fall detection and yaw aiding.
- **Ultrasonics: four sensors** mounted on the main box (**front ×2, rear ×2**) for stairs/edge protection (see ***Figure 3-4***).
- **OAK-D camera:** OCR for door labels (room numbers). A dedicated node returns the recognized room ID to the UI.(see **Figure 3-5).**

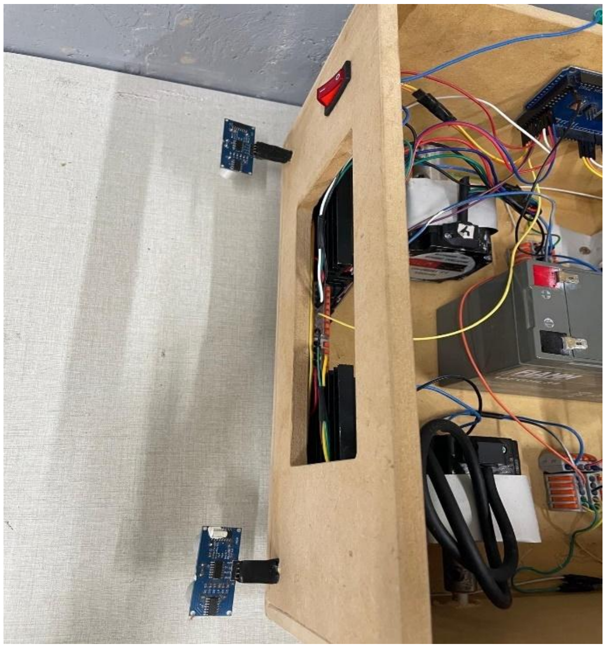
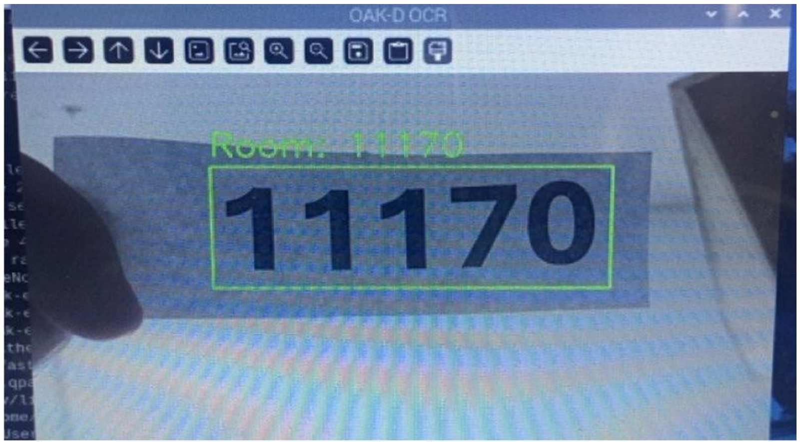


Figure 3-4 Ultrasonic sensor placement on the chassis (front/back).



- **Figure 3-5 OAK-D reading a door-plate number (OCR overlay).**

3.7 Software Workflow and Procedures

1. **Bring-up:** connect LiDAR to the Pi and verify the `/scan` topic.
2. **Odometry:** compute wheel-based odometry from **step counts only** (open-loop steppers). With **wheel radius r = 12 cm**, and **effective track width b** estimated dynamically (calibrated during tests):
   $\Delta s\_R/L = steps \times (2\pi \cdot r / steps\_per\_rev)$;
   $\Delta\theta = (\Delta s\_R - \Delta s\_L) / b$;
   $\Delta x = ((\Delta s\_R + \Delta s\_L)/2) \cdot \cos(\theta)$, $\Delta y = ((\Delta s\_R + \Delta s\_L)/2) \cdot \sin(\theta)$.
   Publish on `/odom` and validate against tape-measure distances.
3. **Mapping:** run **slam_toolbox** to build the occupancy grid; verify `/map` publication. **Scan-matcher update rate = 10 Hz** (map resolution).
4. **Localization & Navigation:** use **Nav2** on the built map with global planning and a local controller (DWB or RPP); confirm that goals at room locations are reachable with obstacle avoidance.
5. **UI Integration:** connect the touchscreen kiosk (Arabic/English selection) to ROS services for destination selection; confirm voice prompts and Telegram alerts.

3.7.1 Differential-Drive Odometry from Step Counts (with IMU yaw option)

Symbols

N_steps: motor full-steps per revolution (typically **200**).
μ: micro-step factor (e.g., 4 for 1/4-step) — **TBD**.
G: gear ratio (if any; **1** otherwise).
D: wheel diameter (**0.24 m**; radius *r* = **0.12 m**).
b: track width (distance between wheel centers), **dynamic**; nominal ≈ **0.19 m**.
x, y, θ: robot pose in *odom* frame.
Δsteps_L, Δsteps_R: step count deltas per sample.
dt: sample period.

(1) Steps ↔ Distance constants

```
steps_per_rev = N_steps · µ · G
C = π · D (wheel circumference)
steps_per_meter = steps_per_rev / C
meters_per_step = C / steps_per_rev
```

(2) Wheel travel per sample

```
Δs_L = Δsteps_L · meters_per_step
Δs_R = Δsteps_R · meters_per_step
```

(3) Differential kinematics

```
Δs = (Δs_R + Δs_L) / 2
Δθ = (Δs_R − Δs_L) / b
```

(4) Pose update

Simple:

`θ_new = θ_old + Δθ` *(or use* `θ_IMU` *if available)*

```
x_new = x_old + Δs · cos(θ_new)
y_new = y_old + Δs · sin(θ_new)
```

Mid-point (more accurate without IMU):

```
θ_mid = θ_old + Δθ/2
x_new = x_old + Δs · cos(θ_mid)
y_new = y_old + Δs · sin(θ_mid)
θ_new = θ_old + Δθ
```

(5) Velocities for `/twist`

`v = Δs / dt`

`ω = Δθ / dt` *(or* `ω = d(θ_IMU)/dt` *if using IMU yaw)*

(6) Wheel speeds from body twist (feed-forward/control)

```
v_R = v + (b/2)·ω
v_L = v − (b/2)·ω
steps_per_sec_R = v_R · steps_per_meter
steps_per_sec_L = v_L · steps_per_meter
```

(7) Quaternion for ROS *(yaw-only)*

```
qx = 0, qy = 0, qz = sin(θ/2), qw = cos(θ/2)
```

(8) Calibration notes

• **Straight-line:** drive 2 m, compare measured vs. odom; adjust **D** (or `steps_per_meter`).

• **In-place 360° turn:** expected wheel travel per side = `π · b`; compute expected steps = `π · b · steps_per_meter`; if error persists, tune **b**.

• Ensure sign conventions (positive wheel directions) match the formulas above.

3.8 Human–Machine Interface (HMI)

- **Touchscreen (7″, kiosk mode):** four main options:
  1. **Hall Guide** (list/search of rooms scanned by the robot),
  2. **FAQs** (pre-stored questions; answers via speaker),
  3. **Dispose Bottle** (servo-driven bin door after confirmation),
  4. **Support** (form sends a message through a Telegram bot see **Figure 3-12**).
     *(Screens for each option will be inserted **Figure 3-6**… **Figure 3-10**.)*
- **Remote control:** web panel used during testing and supervision (see ***Figure 3-11***).

Figure 3-6 Main Interface for the screen

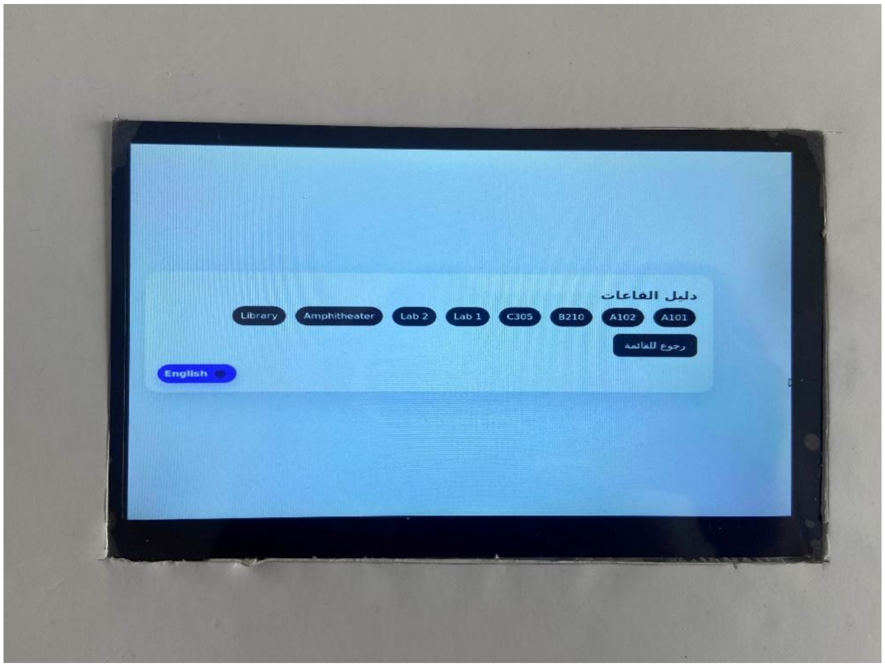
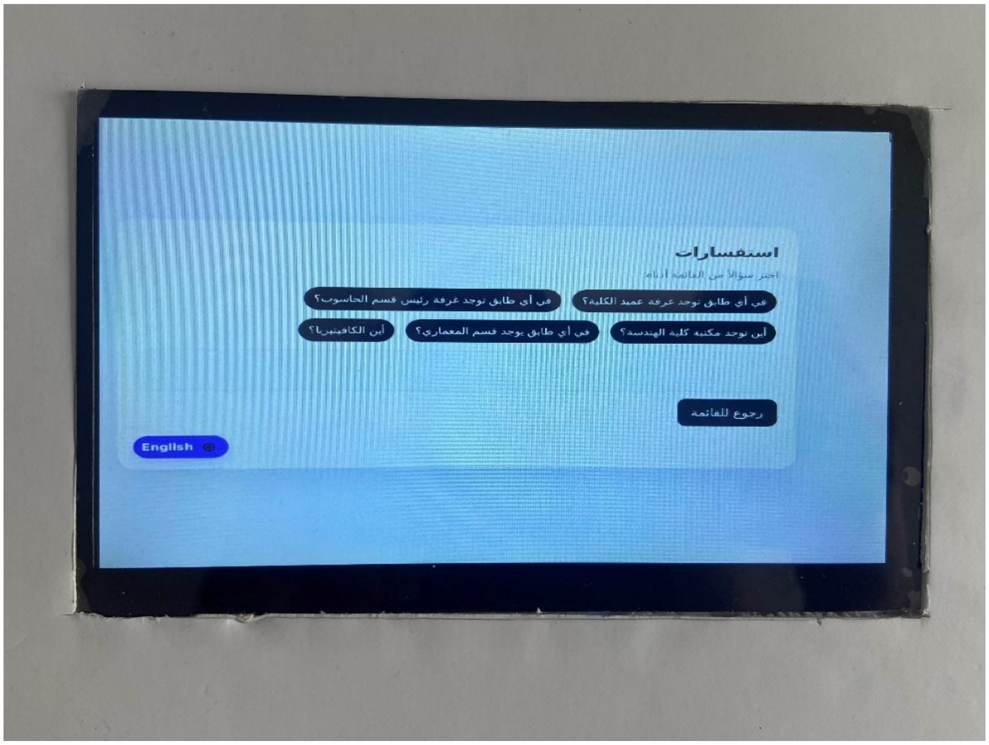


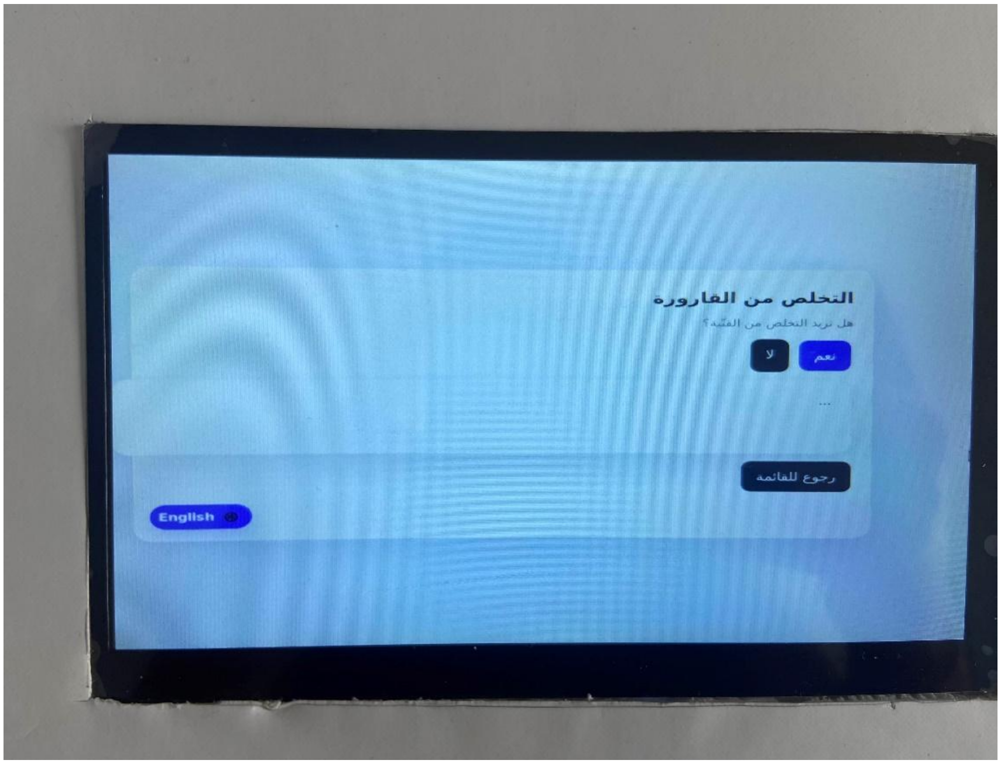
Figure 3-7 Hall Guide Interface



استفسارات

اختر سؤالاً من القائمة أدناه:

Figure 3-8 Inquiries Interface



التخلص من القارورة

هل تريد التخلص من العلبة؟

...

Figure 3-9 Bottle Recycling Interface

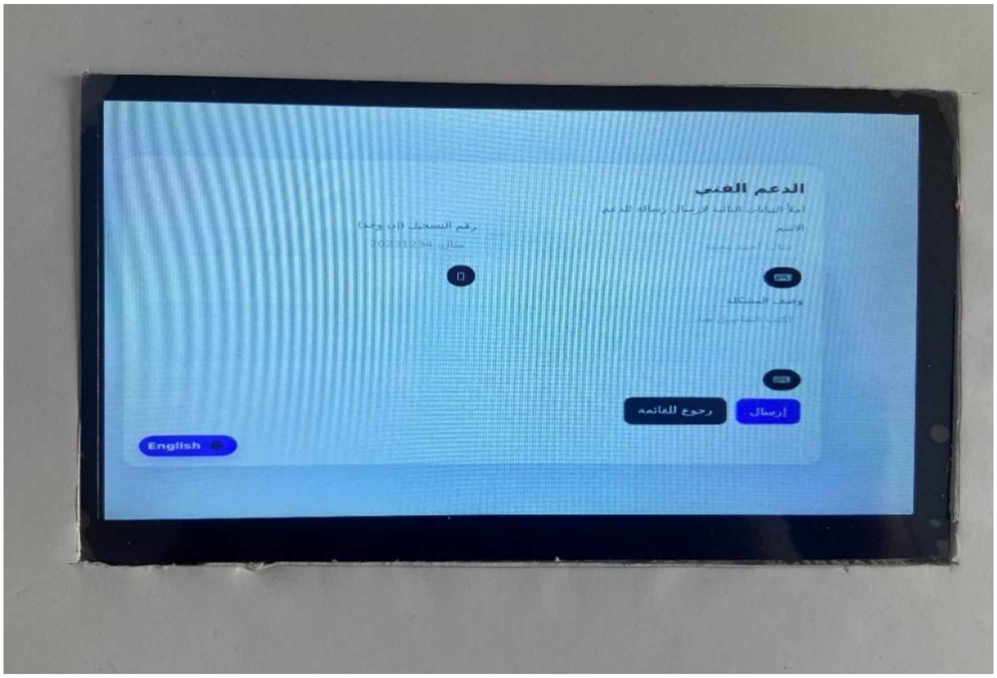


Figure 3-10 Technical Support Interface

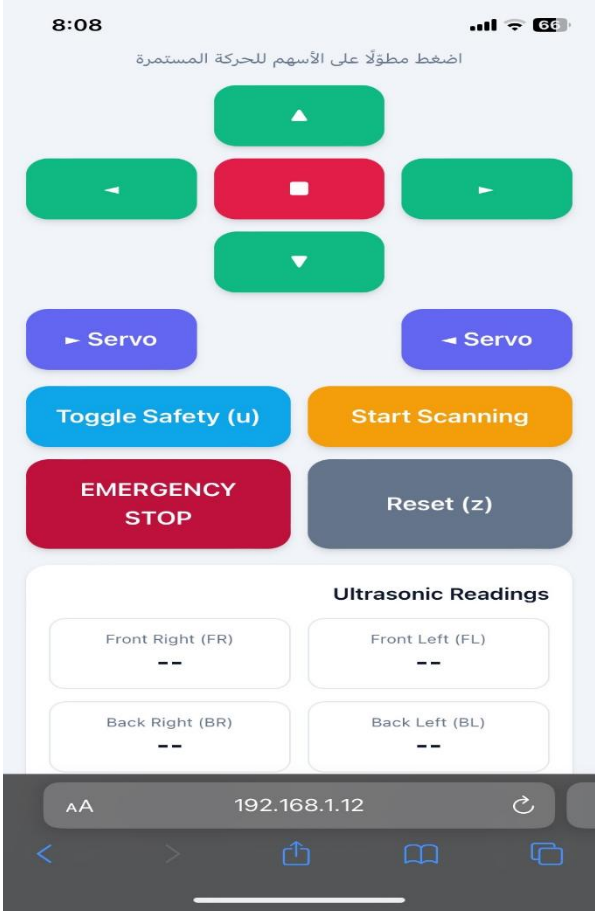
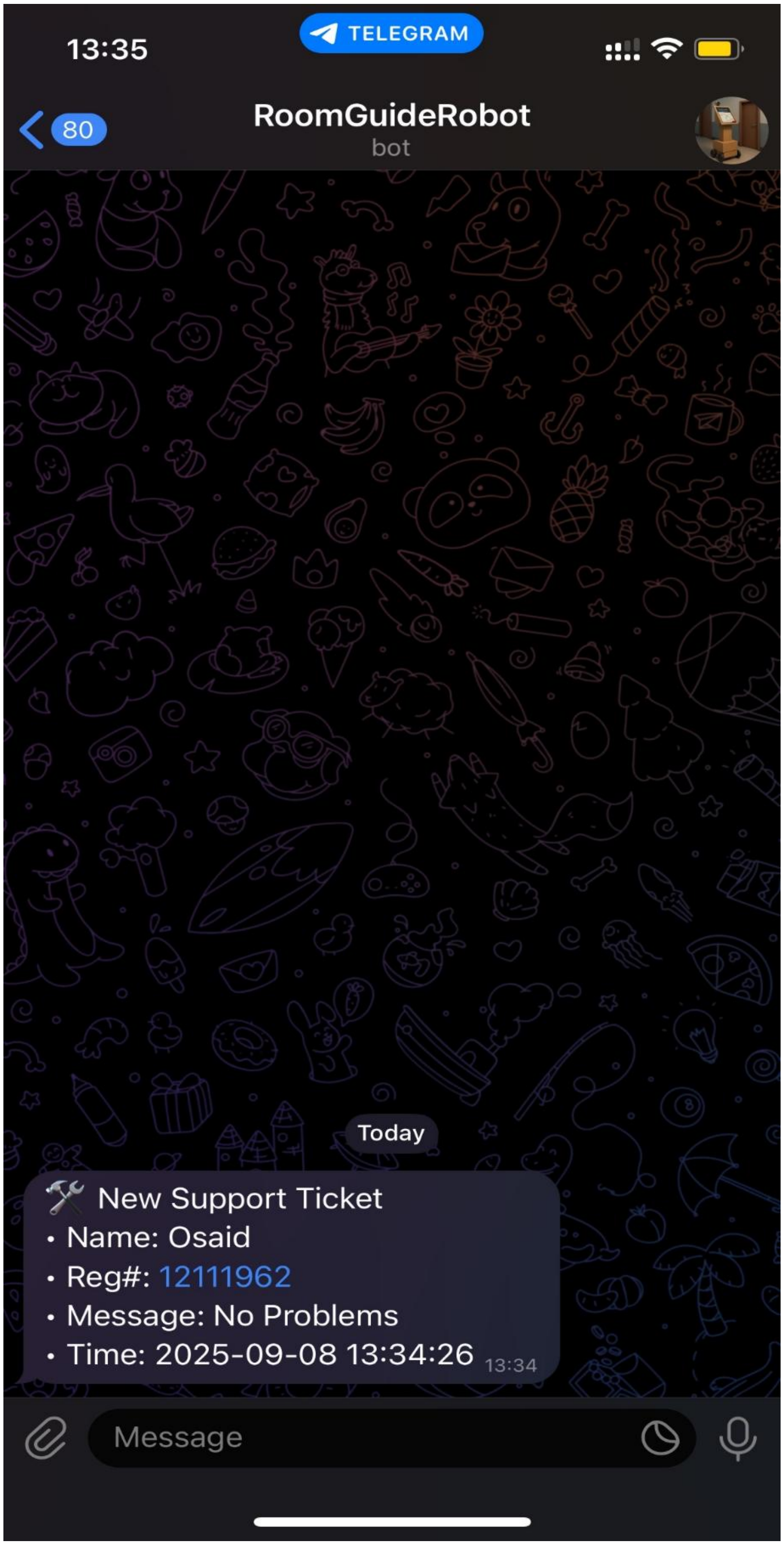


Figure 3-11 Remote Control Interface



🛠️ New Support Ticket
- Name: Osaid
- Reg#: 12111962
- Message: No Problems
- Time: 2025-09-08 13:34:26

13:34

Figure 3-12 Telegram Support Msg

3.9 Hazards and Precautions

- **Lead-acid battery:** proper fusing, polarity checks, supervised charging, ventilation.
- **Spinning LiDAR & moving parts:** keep clear of optics and drive train; no access to the base while energized.
- **Edges/stairs:** ultrasonics gate motion near drop-offs; software E-stop halts immediately; geofencing limits area.
- **Electrical isolation:** separate motor and logic rails; common ground; appropriate wire gauges.

3.10 Constraints (realistic design constraints)

- **Economy:** Hardware-intensive student project with a costly BOM. High-torque **NEMA-23** steppers and **TB6600** drivers, the **RPLIDAR A1**, **OAK-D** camera, 7-inch touchscreen, and the lead-acid battery/power-bank setup all add significant cost. The **OAK-D unit was provided by Dr. Anas** and the **RPLIDAR A1 was provided by Dr. Ahmed**; without their support the total cost would have been substantially higher. Costs were contained where possible by using COTS parts, an MDF chassis, open-source ROS 2, and re-using tooling/cabling.

- **Environment:** Moderate acoustic noise from micro-stepping; energy use tracked per hour during tests; lead-acid charging performed in a ventilated area.

- **Society / Ethics / Privacy:** No persistent video recording; OCR runs on-device and only the recognized room ID is used transiently; minimal telemetry; Telegram messages avoid personal data; operation aligned with campus policies.

- **Health & Safety:** Capped speeds; geofenced operating area; tilt-halt; clearly labeled main **ON/OFF** switch; E-stop behavior; proper fusing and wiring protection.

- **Manufacturability:** 5 mm MDF chassis cut with simple tools; modular wiring using lever terminals; off-the-shelf components.

- **Sustainability:** Parameterized configs; easy battery swap; maintainable codebase; components chosen for availability and future replacement.

3.11 Documented Design Changes

- **Motors:** replaced **NEMA-17** with **NEMA-23** after load tests showed insufficient torque; final layout in **Fig. 3-2** (early layout is documented in Chapter 5/Appendix F).

- **Chassis mass:** removed the 25 cm base box and switched to **0.5 cm MDF** to reduce weight.
- **Perception load:** OAK-D kept for OCR only; bottle-detection pipeline removed due to CPU overhead on the Pi; kiosk kept bilingual (Arabic/English).

Figure 3-13 Pre Final Chassis Look

Figure 3-14 Final Chassis look

Chapter 4 — Results and Analysis

4.1 Experimental Setup

Environment: Single-floor university corridor with straight segments, T-junctions, and typical pedestrian traffic. A prior floor map is used as an initial reference; online SLAM updates accommodate minor layout changes (temporary obstacles, moved furniture).

Robot: Raspberry Pi (ROS2 Humble), Arduino Mega motor bridge, RPLIDAR A1 mounted at approximately 100 cm, ultrasonic array for near-field, IMU for tilt/yaw, 4× NEMA-23 stepper motors with TB6600 drivers, 12 V 8.4 Ah lead-acid battery, buck 12→5 V for logic, 7-inch touchscreen and web remote.

Software: slam_toolbox (online mode), Nav2 (global planner A*/Smac, local controller DWB or Regulated Pure Pursuit), layered costmaps, Behavior-Tree–based navigation. Touchscreen local app and Flask-based web remote with telemetry and control.

Trials: Each scenario repeated **N = 5** runs unless otherwise stated. Log crowd level, lighting, and Wi-Fi conditions. Keep key parameters fixed within a batch (max speed, acceleration limits, inflation radius, obstacle persistence).

4.2 Metrics and Definitions

- **Localization RMSE (cm):** measured against corridor reference markers over a 10–20 m route.

- **Path tracking error:** mean lateral deviation (cm) relative to planned path; report maximum deviation.

- **Obstacle-avoidance success:** percentage of runs completed with no contact or manual intervention.

- **Web remote latency:** press-to-command, command-to-motion, and total round-trip time (ms).

- **Battery runtime:** minutes of mixed-motion operation from full charge to safe cutoff voltage.

- **Safety response:** E-stop stopping distance at given speed; tilt-alarm detection and stop reaction time.

4.3 Test Scenarios

- **S1:** Straight Hallway (no obstacles)
- **S2:** T-Junction Turn (left/right)
- **S3:** Dynamic Pedestrian Crossing (1–3 crossers)
- **S4:** Narrow Passage (~1.2–1.5 m)
- **S5:** Destination Recognition (door label via OCR/fiducial, optional)
- **S6:** Remote Supervision (web panel teleop and latency)
- **S7:** Battery Endurance (loops to cutoff)

4.4 Raw Measurements (placeholders)

Tables are prepared for direct data entry with TBD fields for all metrics (localization, tracking, success rates, latency, endurance, safety). Add rows per run and annotate notes (e.g., crowd spikes, slippery tiles).

4.5 Representative Figures (placeholders)

- SLAM Map Snapshot from RViz showing occupancy grid and trajectory.
- Trajectory vs. Planned Path with lateral error plot.
- Remote UI screenshot and latency time-series.
- Battery Discharge Curve (voltage vs. time under mixed motion).

4.6 Analysis (to be finalized after data entry)

Provide narrative once data are entered, focusing on:

- Expected low RMSE on straight segments; larger deviations near junctions due to occlusions and slip.
- Success rates in S2–S4; typical slowdowns when pedestrians cross.
- Latency ranges over campus Wi-Fi; effect of CPU load on jitter.
- Endurance minutes for the 12 V, 8.4 Ah battery under mixed duty cycle.
- Safety responses scaling with commanded speed and tilt thresholds.

4.7 Threats to Validity

- Environment changes between days (furniture, crowd density).
- Ground-truth sparsity and marker placement.
- Lead-acid battery health variation with temperature and history.
- Wireless variability (AP load, interference).

4.8 Chapter Summary

Defined a consistent evaluation framework with scenarios, metrics, data tables, and figures. After populating the tables and figures, consolidate findings and cross-reference in Chapter 5.

Chapter 5 — Discussion

This chapter **interprets** the results that will be presented in **Chapter 4 (Results & Analysis)**. It avoids repeating raw numbers; instead, it explains implications, limitations, and links design choices to observed behavior.

5.1 Summary of Findings (for Discussion)

- **Objective achieved:** A mobile Hall Guide Robot that guides students to target rooms on a single floor, enables user interaction via a touchscreen kiosk, and includes a recycling/bottle feature.
- **Navigation:** Reliable autonomous navigation on a **static map**; when deployed on a different floor, the system can **build a new map** (dynamic mapping option) without re-engineering.
- **Safety:** Real-time obstacle avoidance using **LiDAR + ultrasonic** and **cliff/edge stop** worked consistently.
- **User interfaces:**
    - **Touchscreen kiosk** boots directly to the HTML interface via a custom **systemd** service (autostart on power-up).
    - **Web remote** control service runs continuously; after fixes, it does not interfere with the kiosk.
- **Notifications: Telegram** alerts operational.
- **Actuation add-ons: Servo "door"** mechanism and **bottle bin** function correctly.
- **Connectivity: Netplan** adjustments provided stable access (including phone hotspot use cases).

- **Hardware integration:** LiDAR/USB stability achieved after cable/port corrections and rigid mounting.

5.2 Discussion

Interpretation of outcomes. The robot fulfills its core purpose with fully working navigation, safety, and interaction features. Combining a kiosk (local, walk-up use) with a web remote (operator/maintenance use) proved practical for campus deployment. The recycling/bottle interaction adds a small but meaningful community feature without complicating the navigation stack.

Positioning vs. common practice. The design follows widely-adopted indoor robotics patterns (ROS 2 + Nav2 + 2D LiDAR with ultrasonic supplementation), which are known to be robust for corridor environments. Our contribution is an **integrated, low-cost implementation** that merges (i) single-floor guidance with **dynamic mapping** for new floors, (ii) **dual-UI** operation (kiosk + web), and (iii) **lightweight add-ons** (servo door, bottle bin, Telegram alerts) that enhance usability and readiness for demos/pilots.

Design decisions that affected results (brief):

- **Chassis weight reduction** → noticeably better acceleration/turning and lower wheel slip (see Methodology: chassis design).
- **Motor upgrade (NEMA-17 → NEMA-23)** → adequate torque and smoother trajectory tracking under full load (see Methodology: motors & drivers).
- **Wheel mounting with Poxipol** → improved mechanical reliability of the wheel–coupler joint (see ***Figure 5-1***).

Figure 5-1 Wheel hub and coupler secured using Poxipol to prevent loosening under load

(Even though these points are detailed in Methodology, they are summarized here because they directly influence performance and explain the final reliability.)

5.3 Limitations and Observed Issues

- **Wheel slip on smooth tiles:** noticeable during sharp turns; mitigations include surface roughening of wheel tread and/or alternative tire material.
- **Excess chassis weight (early):** robot could not move under load; resolved by redesigning to a lighter chassis.

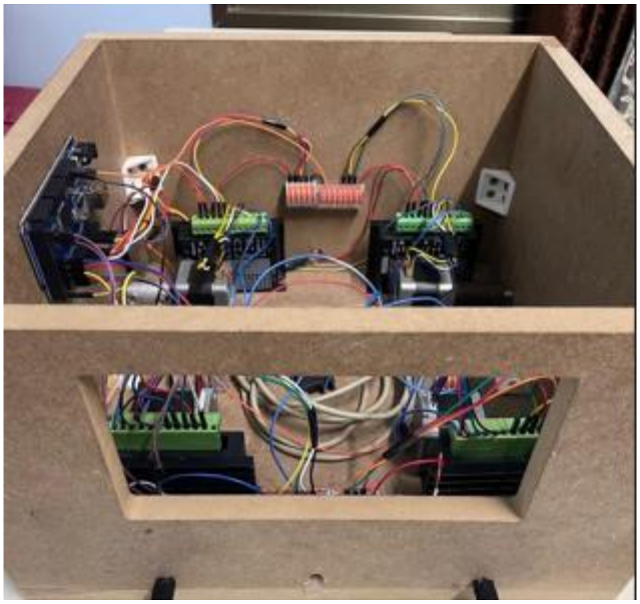


Figure 5-2 initial heavy chassis

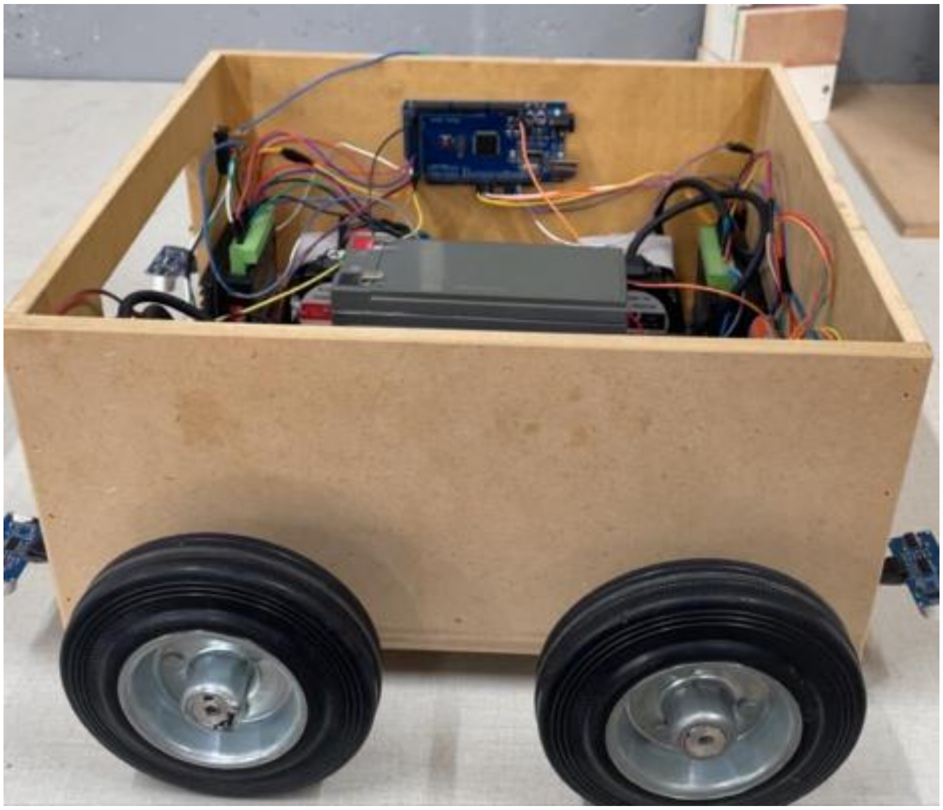


Figure 5-3 ***lighter chassis enabling movement under load."***

- **LiDAR/USB disconnects (early):** caused by cable length/quality; resolved with shorter, spec-compliant cables and secure routing/mounting.
- **Wheel hub loosening (early build):** screws were insufficient to lock the wheel–coupler; fixed using Poxipol.
- **Access & UX friction (early):** network configuration and kiosk startup caused delays; **netplan** tuning and a **systemd** service achieved consistent, hands-free boot into the UI.
- **Nav2/SLAM tuning:** initial issues were resolved during integration; remaining refinements (e.g., parameter tuning for new floors) are handled during mapping sessions.

5.4 Implications

The implemented architecture is **deployment-ready** for a pilot on real corridors within a single floor. The **dynamic mapping** option enables rapid relocation to other floors, while the **dual-UI** model supports both users and operators. With the mechanical fixes (chassis weight reduction, stronger motors, secured hubs), the platform provides a stable baseline for future extensions (e.g., richer perception or multi-floor workflows).

Chapter 6 — Conclusions and Recommendations

6.1 Conclusions

- **Objective met.** The Hall Guide Robot successfully guides users to target rooms on a single floor, with a usable touchscreen kiosk and a recycling/bottle interaction.

- **Deployment-readiness.** The current architecture is suitable for a pilot in real corridors; safety layers (LiDAR + ultrasonic + cliff stop) behaved reliably.

- **Adaptability.** Dynamic mapping allows relocation to new floors without re-engineering; mapping sessions are straightforward once parameters are set.

- **User experience.** Dual UI (kiosk + web remote) serves both walk-up users and operators; netplan tweaks and a systemd service reduced friction at boot.

- **Mechanical reliability matters.** Chassis weight reduction enabled movement under load; NEMA-23 motors provided adequate torque; securing wheel hubs improved robustness.

- **What we learned.** (i) Mechanical and wiring quality dominate early reliability; (ii) cable length/spec affects LiDAR stability; (iii) smooth-tile traction needs attention; (iv) “boot-to-UI” UX is crucial for demos and daily use.

6.2 Recommendations (cost-effective improvements)

1. **Traction on smooth tiles:** add tread tape/texture or lightly roughen wheel surface; test low-cost rubber sleeves.
2. **Cable & port stability:** keep LiDAR/USB cables short, spec-compliant; add strain relief and fixed routing.
3. **IMU fusion & calibration:** integrate yaw stabilization and periodic calibration to reduce drift in long corridors.
4. **Landmark aids:** place **QR/AprilTags** at key junctions/doors for quick relocalization—cheap and effective.
5. **Nav2 parameter presets per floor:** maintain small YAML profiles (costmaps/limits) to speed redeployments.
6. **System health & logging:** watchdog for kiosk/remote services; concise logs for field diagnosis.
7. **Mechanical fastening:** prefer keyed hubs or dual set-screws; keep **Poxipol** as a fallback for couplers.
8. **Power management:** verify battery health under load; tidy power distribution and fusing for serviceability.

6.3 Future Work and Open Problems

- **Multi-floor operation with elevator handling.** Detect elevator lobbies, ride elevators safely, and relocalize per floor.
- **Autonomous docking/charging.** A docking station and charge management for extended operation.
- **Improved traction.** Select wheel materials/geometry optimized for smooth tiles and turning.
- **Stronger perception & interaction.** Robust room-number recognition; optional **person recognition** (privacy-aware) and **natural spoken interaction**.
- **Crowd-aware local planning.** Safer navigation in high-traffic corridors while preserving comfort.
- **Semantic mapping.** Fuse vision with SLAM to tag doors/landmarks for faster relocalization and richer guidance.

- **Operational KPIs & user study.** Formal trials to quantify success rate, time-to-target, and user satisfaction.

6.4 Constraints (Brief)

- **Time & access:** work sessions were constrained by **workshop closing hours**, which limited iteration speed and influenced scheduling of mechanical fixes and mapping sessions.
- **Scope:** single-floor pilot; broader generalization intentionally deferred to future work.

References

- Atkinson, R. C., & Shiffrin, R. M. (1971). The control of short-term memory. *Scientific American, 225*, 82–90.
- Colledanchise, M., & Ögren, P. (2018). *Behavior trees in robotics and AI: An introduction*. CRC Press.
- Fox, D., Burgard, W., Dellaert, F., & Thrun, S. (1999). Monte Carlo localization for mobile robots. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1322–1328.
- Grisetti, G., Kümmerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine, 2*(4), 31–43.
- Leadshine. (2016). *TB6600 stepper drive—User manual* [Manual]. https://www.leadshine.com/
- Luxonis. (2024). *DepthAI & OAK-D documentation* [Documentation]. https://docs.luxonis.com/
- Navigation2 Project. (2025). *Navigation2 (Nav2) documentation* [Computer software]. https://navigation.ros.org/
- Open Robotics. (2025). *ROS 2 documentation (Humble/Jazzy)* [Computer software]. https://docs.ros.org/
- Slamtec. (2018). *RPLIDAR A1: 360° laser scanner—Datasheet* [Data sheet]. https://www.slamtec.com/
- StepperOnline. (2022). *NEMA-23 stepper motor—Datasheet* [Data sheet]. https://www.omc-stepperonline.com/
- Telegram. (2025). *Telegram Bot API* [Web API]. https://core.telegram.org/bots/api
- Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. MIT Press.
- Wang, J., & Olson, E. (2016). AprilTag 2: Efficient and robust fiducial detection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4193–4198).

- Macenski, S., & The SLAM Toolbox Contributors. (2020). *SLAM Toolbox* [Computer software]. https://github.com/SteveMacenski/slam_toolbox
- Macenski, S., & The Navigation2 Project. (2021). Regulated Pure Pursuit controller (RPP) [Documentation]. https://navigation.ros.org/
- Macenski, S., The Navigation2 Project, & Community. (2021). DWB local planner [Documentation]. https://navigation.ros.org/
- Canonical Ltd. (2024). *Netplan documentation* [Documentation]. https://netplan.io/
- Canonical Ltd. (2024). *Ubuntu Frame documentation* [Documentation]. https://snapcraft.io/ubuntu-frame