Department of Electronic & Telecommunication Engineering

University of Moratuwa, Sri Lanka.

EN2160 - Electronic Design Realization

AMR Controller 1

1. **Kuruppuarachchi K.A.R.R – 220350T**
2. **Bandara G.A.M.I.K – 220059J**
3. **Jayaweera M.V.L.M – 220285X**
4. **Madusanka S.P.S – 220374U**
5. **Pitigala P.K.N.W. – 220481U**
6. **Priyanjana T.P.I.M. – 220495P**
7. **Nethmina M.A.P. – 220424B**
8. **Malshan K.K.R. – 220379N**
9. **Peiris T.P.N.S – 220454P**
10. **Anuradha D.M.B.P – 220036L**

May 21, 2025

Contents

1 Introduction/project scope and objective

This report presents the design and development of an Autonomous Mobile Robot (AMR) Controller built to address key challenges in mobile robot navigation and mapping. The primary goals of the project are:

1. Construction of a four-wheel platform with two driving wheels, designed to carry a payload of 5 kg.
2. Deployment of a LiDAR sensor to enable environment sensing and obstacle detection.
3. Development of a SLAM (Simultaneous Localization and Mapping) algorithm in C++, based on the theoretical framework outlined in "Probabilistic Robotics" by Sebastian Thrun et al.

The report covers the theoretical principles that guided the design, initial development efforts during the conceptual design phase, mechanical design and design considerations, calculations for sizing and power, detailed system requirements, careful selection of electronic and mechanical components, schematic design and the overall project budget. Together, these sections provide a comprehensive understanding of the AMR Controller's development process and its practical application in real-world robotics.

2 Initial Developments Stage

2.1 Reference Design

For the development of our Autonomous Mobile Robot (AMR), we selected the **Omron LD-60** as the primary reference design. The LD-60 is a widely adopted commercial AMR known for its reliability in automating material transport tasks within indoor environments such as factories and logistics facilities.

Key features of the LD-60 include:

- **Natural Feature Navigation:** It utilizes onboard laser scanners to localize itself and plan routes without requiring environmental modifications. This aligns well with our goal to develop a SLAM-based navigation system.
- **Fleet Management Capabilities:** The LD-60 supports coordination with up to 100 other AMRs, which is particularly beneficial in multi-robot deployment scenarios.
- **Ease of Deployment:** Minimal installation time and no need for infrastructure changes make it a practical model for emulation.

- **Load Capacity and Mobility:** With a 60 kg maximum payload and a top speed of 1800 mm/s, it offers a balanced trade-off between load handling and agility.
- **Sensor Suite:** Equipped with safety laser scanners and optional side lasers, the LD-60's sensor configuration provides a 360-degree environmental perception — an ideal layout for benchmarking against our RPLiDAR S2L implementation.
- **Power and Runtime:** The robot runs approximately 15 hours without payload and 12 hours under full load, supporting a broad range of test scenarios.

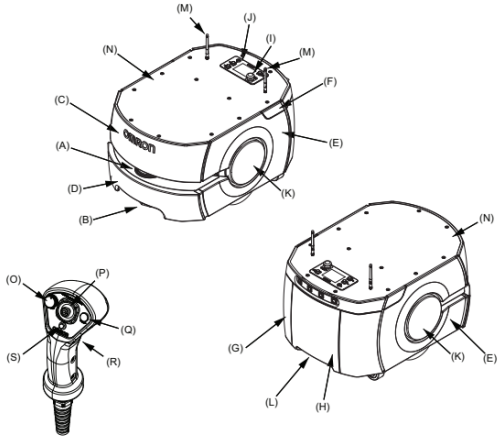


Figure 1: Omron LD-60 Autonomous Mobile Robot

LD-60, LD-90, LD-90x, LD-60 ESD, LD-90 ESD, and LD-90x ESD

Item		LD-60	LD-90	LD-90x
Weight (with Battery)		62 kg		
Environment	**Ambient temperature**	5 to 40°C		
	Ambient humidity	5 to 95% (non-condensing)		
	Operating Environment	Indoor usage only, no excessive dust, no corrosive gas or liquid. Floor must be free of water, oil, dirt, and debris. Direct sunlight may cause safety laser false positives.		
	Dust / Smoke	Airborne particle size: > 37 µm Floor accumulation: < 10 ml / m^2 Avoid smoky areas		
	Ingress Protection Class	IP20		
	Altitude	1000 m above mean sea level maximum		
Floor Conditions	**Minimum floor flatness**	Fr25 (ACI 117 standard)		
	Traversable step	15 mm max. ✻1	10 mm max. ✻1	
	Traversable gap	15 mm max ✻2		
	Maximum Slope	Up to 60 kg: 4.8° / 8.3% incline Over 60 kg: Level floor only		
	Minimum floor compressive strength	2.6 Mpa.	3.27 Mpa	
Navigation	**Routing**	Autonomous routing by localizing with safety scanning laser based on environment mapping		
	Environmental map making method	Scan by walking the AMR through the environment, and upload the scan data to the MobilePlanner software		
	Low Front Laser	One Class 1 laser at front of AMR with a 126° field of view		
	Side Laser (optional)	Two Class 1 lasers with a 270° field of view on the sides of payload structure, user-mounted		
Visual Indicators		Light discs are located on the sides of the AMR. Additional indicators can be added.		
Payload	**Maximum Weight**	60 kg	90 kg	
Mobility	**Run Time (no payload)**	15 h approx.		20 h approx.
	Run Time (full payload)	12 h approx.		15 h approx.
	Maximum Speed	1800 mm/s	1350 mm/s	900 mm/s
	Maximum Rotation Speed	180 °/s		
	Stop Position Repeatability (single AMR) ✻3	· To a position: ±65 mm · To standard target: ±25 mm, ±2° · With CAPS: ±8 mm, ±0.5° · With HAPS: ±8 mm, ±0.4°		
	Stop Position Repeatability (Fleet) ✻3	· To a position: ±85 mm · To standard target: ±35 mm, ±2° · With CAPS: ±12 mm, ±0.5° · With HAPS: ±10 mm, ±0.5°		
Drive wheels	**Materials**	Solid aluminum with non-marking, non-conductive, foam-filled rubber tread		
Passive casters	**Materials**	Conductive thermoplastic rubber on polyolefin		

Figure 2: Omron LD-60 Datasheet showing key specifications

Our design seeks to emulate and improve upon the LD-60 platform's navigation efficiency and modularity using custom components including the NVIDIA Jetson Nano processor and RPLiDAR S2L sensor.

2.2 Stakeholder Analysis

In the context of our Autonomous Mobile Robot (AMR) project, understanding and mapping stakeholders is essential for ensuring the solution aligns with technical, operational, and regulatory expectations. The stakeholder map shown in Figure 3 categorizes key stakeholders based on their **power** and **interest** in the project:

- **Manage Closely:** Includes manufacturers, network engineers, managers, and technicians. These groups are critical to the deployment and maintenance of the AMR and must be engaged regularly for requirements validation, testing, and integration.
- **Keep Satisfied:** End users and government regulators have significant influence but less frequent engagement. Ensuring compliance and satisfaction with the product is essential for acceptance and scaling.

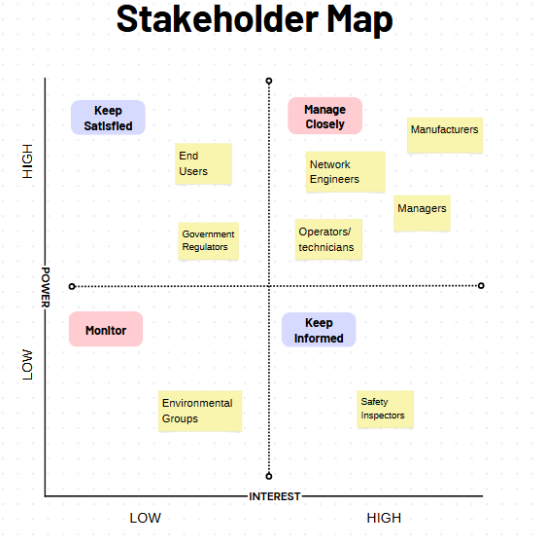


Figure 3: Stakeholder Map for AMR Project

- **Keep Informed:** Safety inspectors and other external observers who require updates to ensure safe operation and standard adherence.
- **Monitor:** Environmental groups and other passive stakeholders with limited interaction but potential to influence public perception or compliance in niche environments.

Understanding this stakeholder distribution allows for effective communication planning, resource allocation, and prioritization of features.

2.3 User Needs Analysis

A comprehensive user needs analysis was conducted to identify the most critical expectations from an AMR in an industrial setting.

- **Reliability and Uptime:** Users expect minimal downtime to ensure continuous workflow automation.
- **Dynamic Navigation:** The AMR must autonomously reroute in response to dynamic obstacles without human intervention.
- **Scalability:** The system should support easy integration of additional units for future expansion.
- **Workforce Adaptability:** Operators with minimal technical training should be able to use and maintain the AMR.

- **Predictive Maintenance:** Early fault detection and usage tracking are key for reducing unplanned failures.
- **Cost Efficiency:** Lower total cost of ownership, including maintenance and deployment costs, is a key driver for adoption.
- **Regulatory Compliance:** The robot must conform to safety and operational standards to be used in commercial and industrial zones.

These user-centric considerations are actively mapped to technical requirements in the system design phase to ensure the robot delivers value throughout its lifecycle.

2.4 Conceptual Design

The conceptual design phase of our AMR robot project was structured to encourage creativity and ensure a broad exploration of possible design directions. Each team member contributed by independently developing three key documents:

- **Enclosure Design:** Focused on the robot's physical structure, sensor positioning, cooling, and accessibility.
- **Functional Block Diagram:** Illustrated the major system modules and their interconnections, including sensors, computing units, and motor control systems.
- **System Flow Diagram:** Mapped the flow of data and control logic across various software and hardware components, particularly emphasizing the integration of the SLAM algorithm.

Following this, we conducted internal evaluations of each submission. Based on these assessments, two refined conceptual designs were created for each of the three categories — totaling six collaborative outputs. These were developed by synthesizing the best elements of the initial designs, guided by a consistent evaluation framework agreed upon by the team.

As the project progressed and our component selection became more defined, we continually revisited these conceptual designs. Updates were made to reflect integration requirements, hardware limitations, and feedback from testing phases. This iterative approach helped maintain a strong alignment between our conceptual vision and practical implementation, ensuring a reliable and adaptable foundation for the AMR system.

2.4.1 Enclosure Design

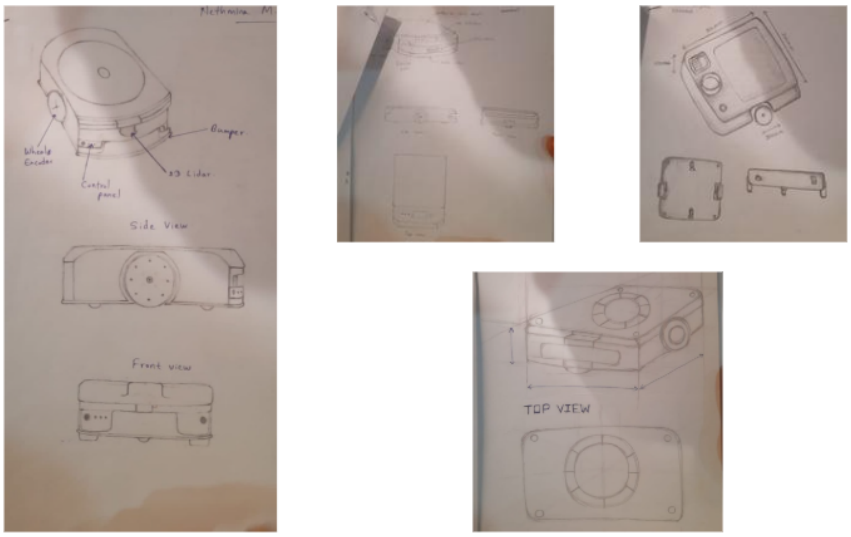


Figure 4: Few Team member submissions for Enclosure Design

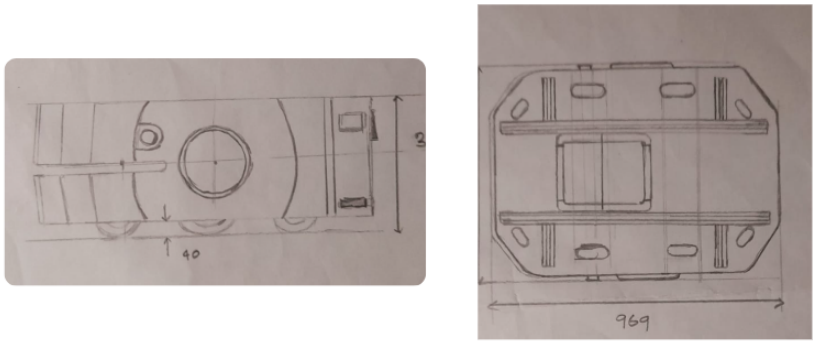


Figure 5: Enclosure Design after team evaluation

2.4.2 System Flow Diagram

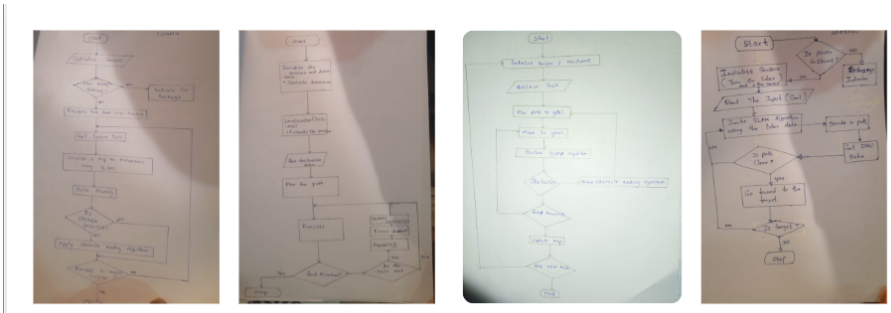


Figure 6: Few Team member submissions for system Flow Diagram

2.4.3 Functional Block Diagram

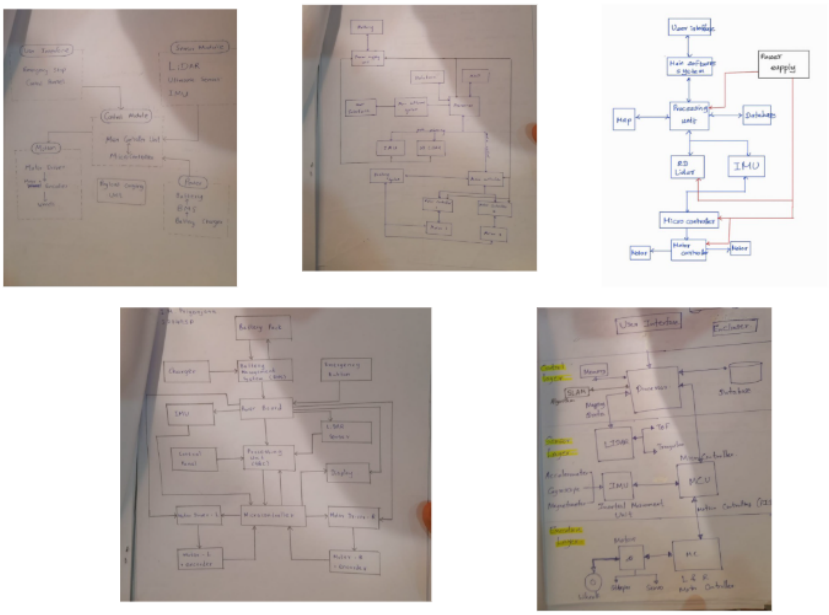


Figure 7: Few Team member submissions for Fuctional block diagram

2.4.4 Evaluation Criteria

2.5 Design Evaluation Criteria

To ensure objectivity in selecting the most suitable conceptual designs for our AMR robot, we developed a scg system based on a set of well-defined evaluation criteria. Each team member submitted three design components — Enclosure Design, Functional Block Diagram, and Flow Diagram — which were then evaluated by the group using these criteria.

Marks were assigned independently to each design under several sub-criteria such as:

- **For Enclosure Design:** Aesthetic appeal, user experience, safety, component placement, manufacturability, platform layout, and thermal management.
- **For Functional Block Diagram:** Completeness, accuracy, feasibility, interconnections, power management, and cost effectiveness.
- **For Flow Diagram:** Logical flow, completeness of operations, obstacle handling, and input management.

The scores for each criterion were added to obtain a total score per section per member. These totals were then compared to identify the best-performing

designs in each category. The highest-rated designs served as the foundation for the final conceptual models selected for implementation and refinement.

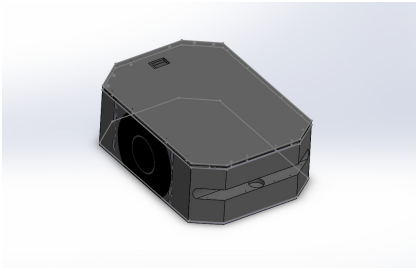
	Evaluation Criteria	Student Names									
		Bandara G.A.M.J.K	Kurupuarachchi K.A.R.R	Madusanka S.P.S	Malshan K.K.R	Pitigala P.K.N.W	Jayaweera M.V.L.M	Nethmina M.A.P	Peiris T.P.N.S	Priyanjana T.P.I.M	Anuradha D.M.B.P
Enclosure Design	Aesthetic look	7	8	8	9	7	9	9	8	7	9
	User experience	8	8	8	8	8	8	9	8	8	9
	Platform design	7	9	9	9	8	8	8	8	8	9
	Manufacture Feasibility	9	8	8	8	8	8	8	7	9	7
	Safety	8	9	7	7	9	7	9	9	8	8
	Sensor and component placement	8	7	8	7	8	9	9	8	8	8
	Thermal Management	8	9	8	8	8	8	8	9	8	8
	Total	55	58	56	56	56	57	60	57	56	58
Functional Block Diagram	Completeness of Components	7	8	8	7	8	7	8	7	8	8
	Fulfillment of main Functionality	8	8	8	8	8	8	8	7	8	8
	Accuracy of interconnections	8	8	8	7	8	7	8	7	9	8
	Power Management	6	8	6	9	8	8	6	7	8	6
	Cost effectiveness	8	8	8	8	8	8	7	8	8	8
	Manufacture Feasibility	8	8	8	8	8	8	8	8	8	8
	Total	45	48	46	47	48	46	45	44	49	46
Flow Diagram	Logical Flow and Operation	8	7	8	7	8	8	8	9	7	7
	Completeness of the process	8	9	7	8	8	7	7	9	8	8
	Obstacle avoidance	8	8	8	9	9	6	9	8	8	9
	Sensor inputs	8	8	7	8	8	8	9	8	8	8
	Total	32	32	30	32	33	29	34	34	32	32

Figure 8: Evaluation table for all design submissions by team members

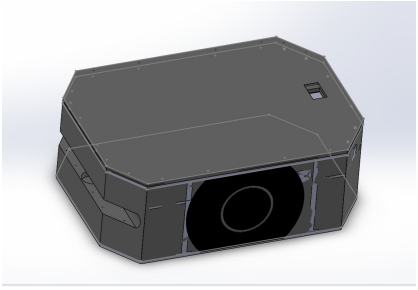
This quantitative approach enabled transparent decision-making and ensured that the final selections were based on both design quality and functional feasibility, as assessed by the entire team.

3 Mechanical design

3.1 AMR Enclosure Design



(a) Front view



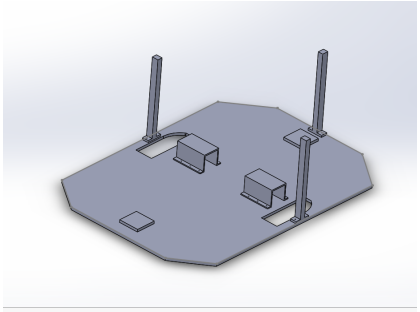
(b) Side view

3.1.1 Steel Chassis

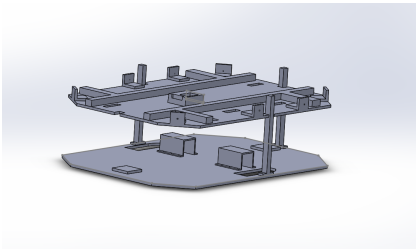
The AMR chassis forms the structural backbone of the entire robot system. Constructed of steel, this framework provides the necessary rigidity and strength to support all operational components while coping with the mechanical stresses encountered during navigation. The chassis design incorporates strategic reinforcement points at high-stress junctions, with additional gussets in the motor mounting areas to prevent flexing during rapid acceleration and deceleration.

The steel structure features predrilled mounting holes with pressed-in threaded inserts to simplify component attachment and future modifications.

- Material: Steel, 10mm thickness
- Dimensions: 963mm × 690mm × 333 mm

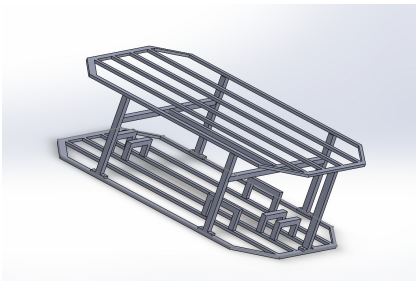


(a) Chassis

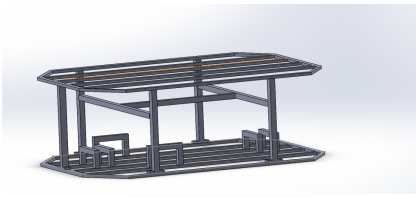


(b) Chassis + Top plate

3.1.2 Framework



(a) Steel frame



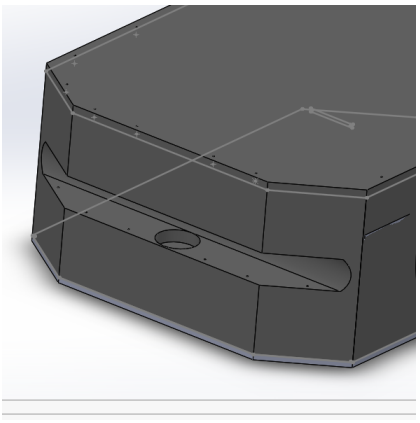
(b) Steel frame side view

3.2 Component Placement

3.2.1 LiDAR Placement

The LiDAR sensor, a critical component for environmental perception and navigation, is strategically positioned at the highest point of the AMR to maximize its effective scanning range and minimize blind spots. This elevated position provides more than 180° field of view, enabling comprehensive spatial awareness during operation. The mounting system utilizes four vibration-isolating rubber mounts that effectively dampen chassis vibrations, ensuring scan quality remains consistent even during movement over uneven surfaces.

The LiDAR assembly is recessed slightly within a protective housing that shields it from potential impacts while maintaining full sensor functionality. Power and data connections route directly through the mounting base, eliminating exposed cabling that could be damaged during operation. The mounting bracket is designed with quick-release functionality, allowing for rapid removal and replacement during maintenance or upgrades without requiring recalibration of the entire system.



(a) LiDAR mounting position

(b) The mounted LiDAR

3.2.2 Motor Placement

The propulsion system employs two DC motors positioned in the middle of the chassis in a differential drive configuration. This arrangement provides excellent maneuverability with the ability to execute zero-radius turns while maintaining a stable platform during operation. Each motor is housed in a custom-designed mounting bracket that precisely aligns the drive shaft with its corresponding wheel assembly while isolating motor vibration from the main chassis.

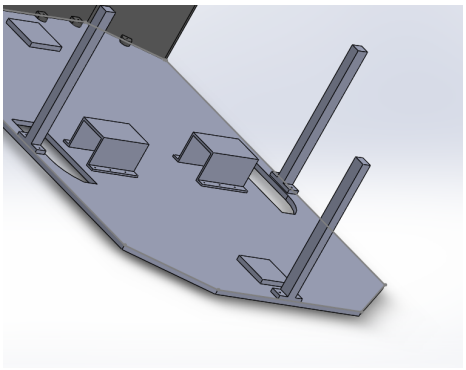


Figure 13: Motor mounting configuration

3.2.3 IMU / Compass Sensor Placement

The IMU sensor is mounted near the center of the AMR robot, close to the main controller, and isolated from mechanical vibrations to ensure accurate measurements. Unlike directional sensors, the IMU does not have a field of view; instead, it continuously monitors the robot's orientation, tilt, and acceleration. This data is essential for maintaining navigation stability, tracking heading, and

assisting in localization, especially during turns or when operating on uneven surfaces.

3.2.4 Power and emergency control

The emergency stop button and power switch are prominently and accessibly positioned on the enclosure for quick and safe operation. The emergency button is designed to immediately halt the robot's motion in case of any hazard, while the power switch allows users to turn the system on or off with ease. Their clear placement ensures intuitive use during both normal operation and emergency situations.

(a) The emergency button

(b) The power switch

3.3 Finite Element Analysis

Finite Element Analysis (FEA) was done using SolidWorks Simulation. Alloy steel is selected as the material for the chassis, while ABS was used for the outer casing and the top plate. We create place at top plane for 5kg payload placement. Then simulation was performed to evaluate the structural performance under 5kg load condition.

3.3.1 Displacement Analysis

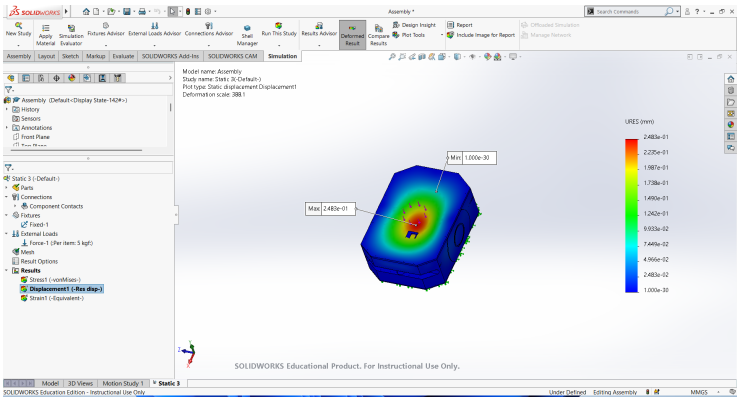


Figure 15: Displacement Analysis

According to simulation result Maximum displacement of top plate is 0.24mm.

3.3.2 Strain Analysis

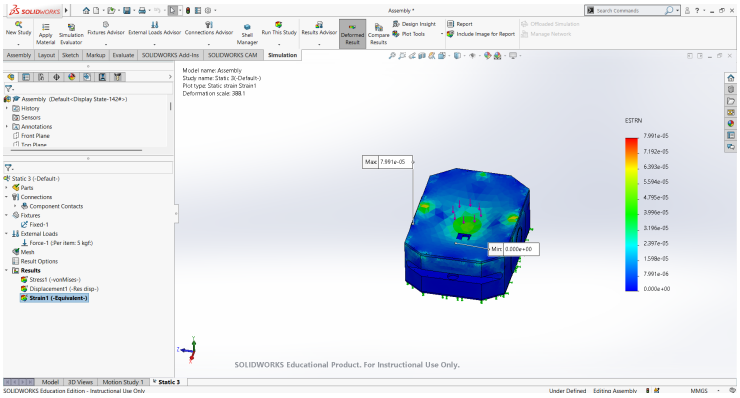


Figure 16: Strain Analysis

According to simulation result Maximum strain of top plate is 0.00008 mm.

3.3.3 Stress Analysis

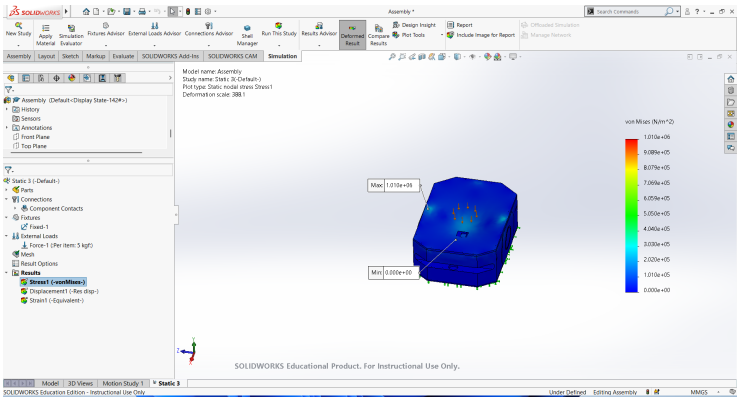


Figure 17: Stress Analysis

3.4 Manufacturing Method & Assembly Process

Chassis Design and Fabrication

The chassis of the AMR robot was initially designed using SolidWorks. The material selected for the chassis was ..... .All necessary mounting holes for motors, sensors, and electronic components were included in the design and fabricated during this stage.

Wheel and Motor Mounting

Here use stepper motors to drive the wheels. Motors will be securely mounted onto the chassis using brackets and standard M3 screws and nuts. The wheels were then fixed to the motor shafts with proper alignment to ensure balanced movement and minimize mechanical wear. Two stepper motors were mounted on opposite sides of the chassis, while a caster wheel was placed at the rear and back to support the robot's balance during turns and uneven terrain navigation.

Sensor Mounting

The LiDAR sensor was centrally mounted at the middle of the chassis, for 180° view for optimal environmental scanning. The base was reinforced to minimize vibrations and maintain sensor stability during motion. The IMU sensor was placed at the center of the robot to avoid vibration and electromagnetic interference, ensuring accurate orientation and motion detection data essential for navigation and path planning.

Controller and Power System Integration

The main controller, such as the Jetson Nano or a microcontroller, was installed on insulated spacers to prevent vibration damage and ensure electrical insulation. The battery pack was carefully positioned to maintain the robot's center of gravity and was secured using custom injection moulded holders. A power

distribution module was used to regulate and distribute power to all subsystems, including motors, sensors, and the controller, with appropriate safety margins.

Wiring and Cable Management

All wiring between components was routed cleanly along predefined paths and secured with cable ties to avoid tangling or contact with moving parts. Modular connectors were used to make maintenance and troubleshooting easier. Care was taken to separate power lines from signal lines to reduce electromagnetic interference. Basic protections like fuses or voltage regulators were included to safeguard the electronics against voltage spikes or short circuits.

Final Assembly and Testing

Once all mechanical and electrical parts were installed, the full assembly of the AMR robot was completed. The system was tested for mechanical stability, wheel alignment, and tightness of connections. Sensor functionality, such as LiDAR scanning and IMU orientation tracking, was verified through initial calibration tests. Adjustments were made where needed to improve balance, alignment, and sensor accuracy before the robot was considered fully operational.

4 Calculations

4.1 Motor type comparison

There are servo motors, stepper motors and etc. Among them we choose stepper motors based on following criteria.

Motor Type Comparison

Specification	Marks Allocation	Servo Motor	Closed-loop Stepper	Open-loop stepper
Precision	20	18	15	10
Toque at low speed	15	12	15	15
Cost	20	1	16	18
Efficiency	15	15	10	5
Reliability	15	15	12	10
Torque at High speed	5	5	4	2
Heat Generation	10	8	6	2
Total	**100%**	**74%**	**78%**	**62%**

Figure 18: Motor comparison Table

Although, servo motors provide excellent precision and control, We choose closed loop stepper motor because its cost effectiveness and it is sufficient for our AMR.

4.2 Choosed Motor , Motor Driver and GearBox

We choose 24HS40-5004D-E1000: 4.0Nm Closed loop stepper motor and CL57T-V41 Driver based on our torque requirement.Torque caculations are following.

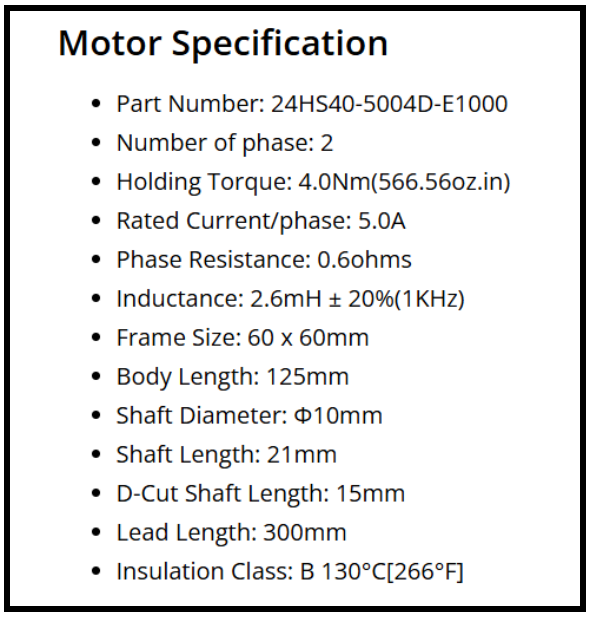


Figure 19: 24HS40-5004D-E1000 Stepper Motor

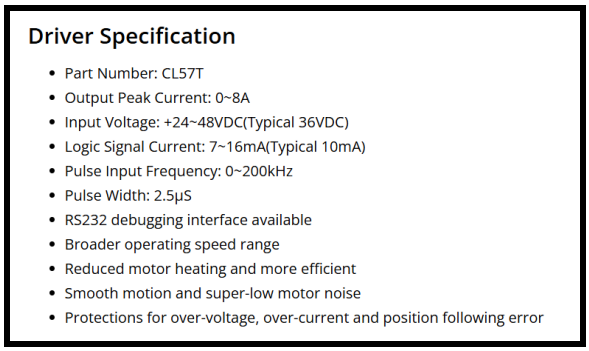


Figure 20: CL57T-V41 Driver

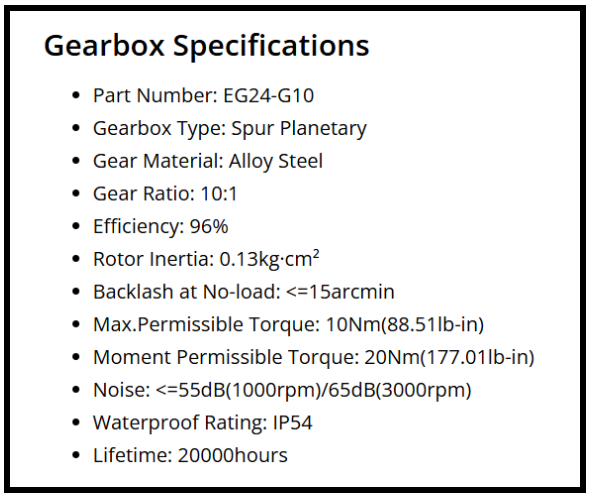


Figure 21: EG24-G10 Gearbox

4.3 Torque and Power Calculation

Our AMR is designed to carry a payload of 5 kg, and total mass is 25 kg. Our motion profile follows a trapezoidal velocity curve, with a maximum velocity of $0.6\,\mathrm{ms}^{-1}$ and an acceleration of $0.6\,\mathrm{ms}^{-2}$. We use rubber wheels with a diameter of 15 cm.

4.3.1 Velocity Profile

$V_{\max} = 0.6\,\mathrm{m/s}$
$a = 0.6\,\mathrm{m/s^2}$
$t_{\mathrm{acc}} = 1\,\mathrm{s}$

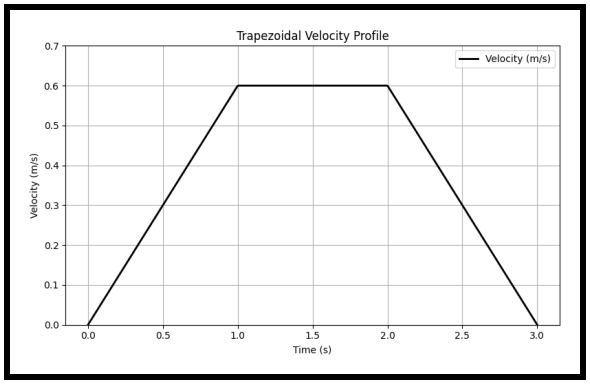


Figure 22: Velocity Profile

4.3.2 Calculation of Parameters

Rolling Friction factor between thermoplastics Rubber and cement - 0.02
Payload - 5 kg
Total Mass - 25 kg
Radius of wheel - 7.5 cm
Gravitational acceleration(g) - 9.81 m/s^2
acceleration(a) - 0.6 m/s^2

$F_{\text{friction}} = \mu \times M \times g = 0.02$ * (20+5) * 9.81 = 4.9 N
$F_{\text{acceleration}} = M \times a = (20 + 5) \times 0.6 = 15$ N

4.3.3 Motor Torque

Torque = force * radius

Motor Torque constant = Holding Torque/Rated Current = 4/5 = 0.8 Nm/A

4.3.4 Constant Velocity

Required Torque per motor = $\frac{4.9 \times 0.075}{2} = 0.188$ Nm
current = 0.188/0.8 = 0.235 A

4.3.5 Accelaration

Required Torque per motor = $\frac{(15+4.9) \times 0.075}{2} = 0.75$ Nm
current = 0.75/0.8 = 0.94 A

4.3.6 Electrical Power for motors

supply voltage - 24 V
current(with safety factor 2) = 4 A
power - 24 * 4 = 92 W

4.3.7 Total Power When Mapping

Components	Current (A)	Voltage (A)	Power (W)
Motors	4	24	92
LiDAR	0.5	5	2.5
Jetson Nano	2	5	10

Table 1: Total Power Calculations

Assume when mapping, robot has to accelaration or deaccelarate full time.
Total current = 4 + 2 + 0.5 = 6.5 A
Total power = 92 + 2.5 + 10 = 105 W

4.3.8 Total Power Under Normal Operation

When normal operation robot move 20 percent of time accelaration and deaccelaration and rest of the time working at constant velocity.
Total current = 0.84 + 2 + 0.5 = 3.5A
Total power = 36.5 + 2.5 + 10 = 49W

4.4 Power sharing

4.4.1 Power Distribution Unit

We use "Power Distribution Board (PDB) 60A Xt30 Pre-Soldered" to share power across stepper motors,jetson nano and lidar.

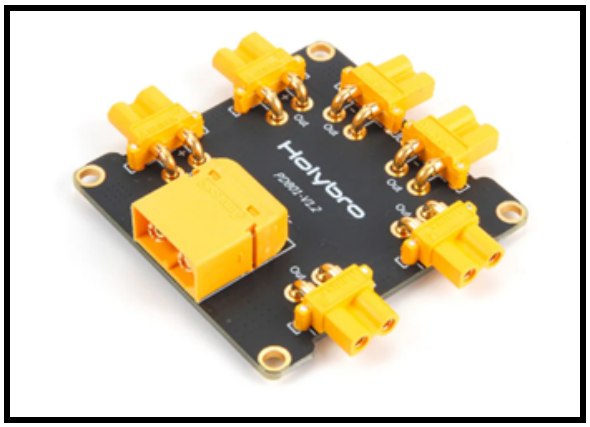


Figure 23: Power Distribution Unit

4.4.2 Power for stepper motors

We use boost converter module to boost 22.2V to 36V.

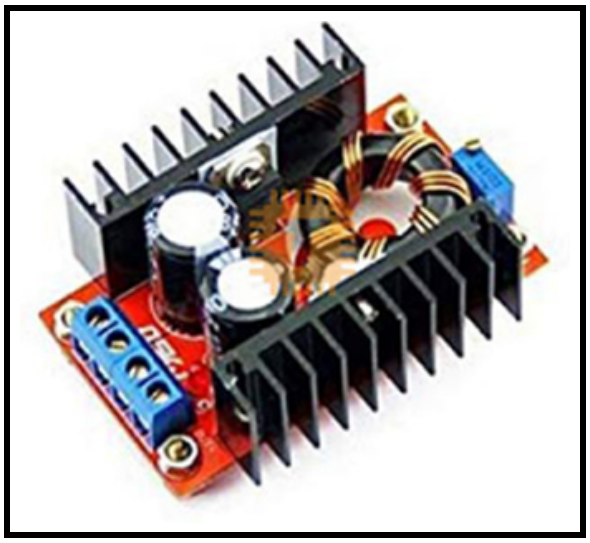


Figure 24: Boost Convertor Module

4.4.3 Power for Jetson Nano

We use buck conveter module for supply power for jetson nano

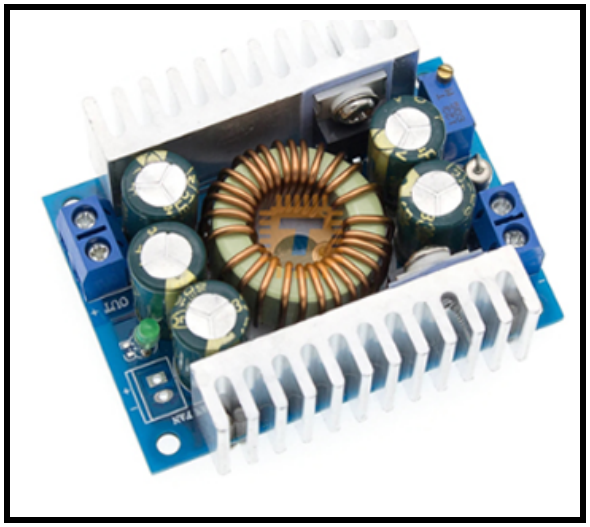
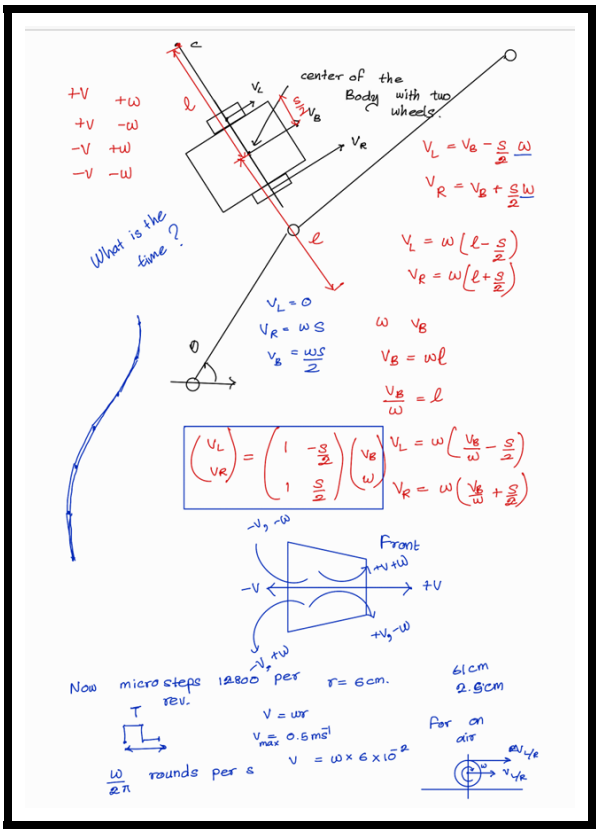


Figure 25: Buck Convertor Module

4.5 Differential Drive Calculations

The complete calculation done on differential drive is shown through following three diagrams.



1 round $\longrightarrow$ 12800

$\frac{\omega}{2\pi}$ rev. s^{-1} $\longrightarrow$ $12800 \times \frac{\omega}{2\pi}$ steps per sec.

pulse frequency.

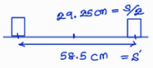
When max v is Not decided yet.

$v = \omega \times 6\times10^{-2}$

$\frac{v}{6\times10^{-2}} = \omega$ ← wheel velocity.

pulse frequency $\left(12800 \times \frac{v_{L/R}}{6\times10^{-2}\times 2\pi}\right)$

$$f = \frac{12800 \times v_{L/R}}{6\times10^{-2}\times 2\pi}$$



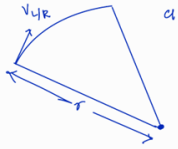
When pass v_B, ω

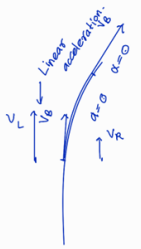
$\begin{pmatrix} v_B \\ \omega \end{pmatrix} \longrightarrow \begin{pmatrix} v_L \\ v_R \end{pmatrix} \longrightarrow \begin{pmatrix} f_L \\ f_R \end{pmatrix}$

$$\begin{pmatrix} f_L \\ f_R \end{pmatrix} = \frac{12800}{6\times10^{-2}\times 2\pi} \begin{pmatrix} 1 & -29.25\times10^{-2} \\ 1 & 29.25\times10^{-2} \end{pmatrix} \begin{pmatrix} v_B \\ \omega \end{pmatrix}$$

$\dot{v} = \dot{\omega} r$

$a = \alpha r$





Transitions when commands

$v_L = \omega\left(\dot{r} - \frac{s}{2}\right)$

$v_R = \omega\left(\dot{r} + \frac{s}{2}\right)$

$\dot{v}_L = \dot{\omega}\left(\dot{r} - \frac{s}{2}\right)$

$\dot{v}_R = \dot{\omega}\left(\dot{r} + \frac{s}{2}\right)$

$\dot{v}_L = -\frac{\dot{\omega} s}{2}$

$\dot{v}_R = \frac{\dot{\omega} s}{2}$

5 System requirements

5.1 Processor Justification -AMR Robot

5.1.1 Key Processing Requirements

To effectively support the RPLiDAR S2L (Time-of-Flight LiDAR with 360° scanning, 18m range, and UART interface) in an Autonomous Mobile Robot (AMR), the selected processor must meet these critical requirements:

- **Real-Time LiDAR Data Processing Capability**
  - Must handle continuous high-frequency (8-15 Hz) 2D point cloud data streams
  - Should process UART serial data without latency or bottlenecks
- **SLAM and Navigation Support**
  - Capable of running Simultaneous Localization and Mapping algorithms efficiently
  - Able to perform real-time obstacle detection and path planning
- **Power Efficiency**
  - It is good to maintain power consumption below 10W for battery-powered operation
  - Needs to support power optimization features
- **Memory**
  - Requires minimum 2-4GB RAM for smooth LiDAR data processing
- **Interface and Connectivity**
  - Must provide UART for direct LiDAR communication
  - Requires USB ports for additional sensors
  - Needs GPIO for motor control and peripheral integration

5.1.2 NVIDIA Jetson Nano (4GB) Justification

5.1.3 Computational Performance

- Features quad-core ARM Cortex-A57 CPU running at 1.43 GHz, providing sufficient processing power for real-time LiDAR data handling
- Includes 128-core NVIDIA Maxwell GPU that accelerates point cloud processing and mapping algorithms

5.1.4 Memory and Storage

- 4GB LPDDR4 RAM capacity meets the requirements for LiDAR data processing and SLAM operations
- MicroSD slot provides necessary storage expansion for system and mapping data

5.1.5 Power Efficiency

- Operates within 5-10W power range, making it suitable for battery-powered AMR applications
- Supports power optimization modes for energy-efficient operation

5.1.6 Connectivity and Interfaces

- UART interface enables direct connection to RPLiDAR S2L
- USB 3.0 ports allow integration of additional sensors like IMU or cameras
- GPIO pins for peripheral control
- Includes Gigabit Ethernet and Wi-Fi for network connectivity and remote monitoring
- The Jetson Nano, although not equipped with a built-in cooling fan by default, features a substantial passive heatsink. This design choice provides several benefits, particularly in environments like warehouses or factories where dust and particulate matter are common.

5.1.7 Comparative Analysis

Table 2: Processor Comparison

Processor	**CPU/GPU**	**RAM**	**Suitability**
NVIDIA Jetson Nano (4GB)	4x A57 + 128-core GPU	4GB	Optimal balance
Jetson Xavier NX	6x Carmel + 384-core GPU	8GB	Capable but potentially excessive
Intel NUC (i3)	x86 CPU (No GPU)	8GB+	Higher power, less optimized

5.1.8 Conclusion

The NVIDIA Jetson Nano (4GB) represents the most appropriate processing solution for RPLiDAR S2L integration in an AMR due to its:

- Adequate computational capabilities for real-time LiDAR data processing
- Balanced power consumption suitable for battery operation

- Comprehensive interface options for sensor and motor integration
- Cost-effectiveness compared to higher-performance alternatives

While more powerful processors are available, the Jetson Nano provides the optimal combination of performance, power efficiency, and connectivity for standard LiDAR-based AMR applications.

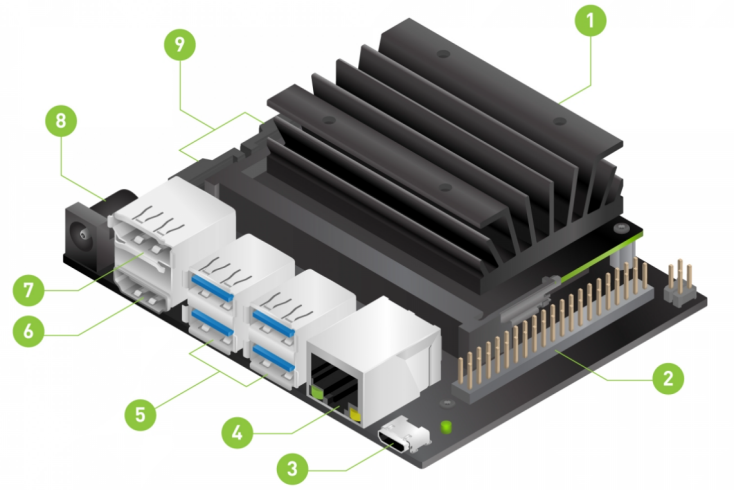


Figure 26: Jetson Nano developer kit

1. microSD card slot for main storage
2. 40-pin expansion header
3. Micro-USB port for 5V power input, or for Device Mode
4. Gigabit Ethernet port
5. USB 3.0 ports (x4)
6. HDMI output port
7. DisplayPort connector
8. DC Barrel jack for 5V power input
9. MIPI CSI-2 camera connectors

5.2 LiDAR Selection

Accurate perception of the environment is critical for implementing SLAM (Simultaneous Localization and Mapping) in an Autonomous Mobile Robot (AMR). As such, selecting a LiDAR sensor with optimal range, frequency, resolution, and reliability was a crucial design decision.

We considered six different LiDAR models from the Slamtec family: TG30, G4, TG15, S2M1, S2L, and S2E. These models differ in their sensing technology (Time of Flight vs. Triangulation), range, scanning resolution, output type, and cost.

Specification	TG30	G4	TG15	S2M1	S2L	S2E
Ranging Technology	ToF	Triangulation	ToF	ToF	ToF	ToF
Ranging Frequency	Up to 20,000 Hz	Up to 9,000 Hz	Up to 20,000 Hz	32K	32K	32K
Scan Frequency	5–12 Hz	5–12 Hz	5–12 Hz	10 Hz	10 Hz	10 Hz
Ranging Distance (White/Black)	0.05–30 m / N/A	0.12–16 m / N/A	0.05–15 m / N/A	0.05–30 m / 0.05–10 m	0.05–18 m / 0.05–8 m	0.05–30 m / 0.05–10 m
Scan Angle	360°	360°	360°	360°	360°	360°
Angle Resolution	0.09–0.22°	0.2–0.48°	0.09–0.22°	0.12°	0.12°	0.12°
Output	TTL UART Serial	TTL UART Serial	TTL UART Serial	TTL UART Serial (3.3V)	TTL UART Serial (3.3V)	Ethernet UDP
Retail Price	$482	$337	$360	$399	$299	$469

Table 3: Specification Comparison of LiDAR Models

Evaluation Criteria: To objectively assess the suitability of each LiDAR model, we defined a scoring framework with weights assigned to each key parameter:

Model	**Range** (20)	**Freq** (15)	**Res** (15)	**Out** (10)	**Size** (10)	**Scan** (10)	**Price** (10)	**Tech** (10)	**Total** (100)
TG30	20	15	15	7	8	10	5	10	**90**
G4	15	10	8	7	7	10	6	7	**80**
TG15	17	15	15	7	8	10	6	10	**88**
S2M1	20	15	12	9	10	10	7	10	**93**
S2L	18	15	12	9	10	10	10	10	**94**
S2E	20	15	12	10	10	10	4	10	**91**

Table 4: Final Scoring of LIDAR Models Based on Evaluation Criteria

The **RPLiDAR S2L** emerged as the best candidate with a total score of **94/100**. It offers a good balance between performance and price, supports a full 360° scan, provides a resolution of 0.12°, and has a compact form factor. Moreover, it operates on TTL UART (3.3V), which is directly compatible with our Jetson Nano via USB bridge. For that the lidar includes an USB to serial adopter.

To evaluate the impact of angular resolution on LiDAR precision, we calculated the maximum positional error of the selected RPLiDAR S2L at a range of 10 meters.

- **Angular Resolution:** $0.12^\circ = 0.002094\,\text{rad}$
- **Distance:** $r = 10\,\text{m}$

The lateral spacing (error) between two adjacent scan points is calculated as:

$$\Delta x = r \cdot \theta = 10 \cdot 0.002094 = \boxed{2.1\,\text{cm}}$$

Thus, the RPLiDAR S2L has a maximum angular spacing error of approximately **2.1 cm at 10 meters**, which is acceptable for our indoor SLAM application where obstacle sizes are typically larger.

Although the S2M1 and S2E also scored highly, the S2L was ultimately preferred for its better cost-efficiency and adequate range (up to 18 m), making it ideal for indoor SLAM applications.

Figure 27: S2L LiDAR

Communication Protocol and Commands: RPLIDAR S2L follows a binary protocol based on request-response architecture. The host system initiates all sessions via commands. Key features include:

- **Response Packets:**
  - Measurement data includes:
    - Distance (in mm)
    - Angle (Q6 format in degrees)
    - Quality value (based on reflectivity)
    - Checksum for data validation
- **Express Modes:**
  - `Legacy Express` – 32 points per packet (cabin structure)
  - `Ultra Capsuled` – 96 points per packet with compression
  - `Dense Capsuled` – 40 points per packet optimized for 32kHz operation

Data Flow Example: A typical scanning workflow involves:

1. Powering up the device.
2. Sending `GET_HEALTH` to verify operational state.

3. Sending `EXPRESS_SCAN` request.

4. Receiving capsulated scan data packets continuously until a `STOP` command is sent.

SDK and Platform Compatibility: Slamtec provides an open-source SDK. It supports:

- Windows, Linux, macOS
- Embedded ARM Linux (e.g., Jetson Orin Nano)
- RoboStudio Framegrabber plugin for debugging

Conclusion: These additional protocol and interface details make the RPLIDAR S2L an ideal candidate for SLAM-based AMRs. Its robust data handling, high resolution, and straightforward integration with UART/USB bridges enable seamless usage with Jetson-based control systems.

5.3 Microcontroller: Atmega32u4

After a comparison with several other microcontrollers we came up with atmega32 u4 as the mcu in amr robot. Mainly the task of the mcu is to send the control signals to the motor driver. Simply, the processor takes the imu readings and encoder readings, then process the task and send the signals to the mcu. Mcu executes the motor driving tasks according to the moving signals given by the processor.

ATmega32U4 is an 8-bit AVR microcontroller developed by Atmel. It features a high-performance RISC architecture.

5.3.1 Peripherals we can see in Atmega32u4

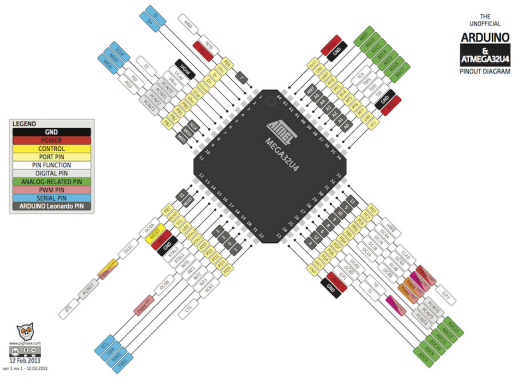


Figure 28: Atmega32u4 pinout

- Core: 8-bit AVR CPU
- Clock Speed: Up to 16 MHz
- Flash Memory: 32 KB
- SRAM: 2.5 KB
- EEPROM: 1 KB
- USB 2.0 Full-Speed Controller (native USB support — useful for direct PC communication)
- I/O Pins: 26 programmable I/O lines
- Timers/Counters: Three timers (one 16-bit and two 8-bit)
- PWM Channels: Useful for motor control
- ADC: 10-bit ADC with multiple channels (for sensor input)
- USART, SPI, and I2C interfaces: For communication with peripherals
- Operation voltage Max. 5.5V
- Low power consumption

5.3.2 Sufficient for the project?

	Weight	ATmega32U4	ATmega328P	ATmega2560	AT5AMD21	AT5AMD51	LPC5556x	MSP430G2553	TM4C123GH6PM
IO Ports	1	7	8	2	4	2	5	8	4
Clock Speed	1.5	2	2	2	3	6	8	2	4
USB 2.0	0.25	10	0	0	10	10	10	0	10
Availability	1.5	5	7	6	0	0	0	7	0
Cost	1.5	9	9	9	4	1	0	4	2
Flash	1	2	2	6	8	10	8	2	6
RAM	1	2	1	7	4	8	8	2	4
Hardware Timers	1	4	3	6	3	5	5	2	6
Total Score		41.5	41	41.5	32	38	40.5	33.5	31.5

Figure 29: Evaluation of Microcontrollers

- This microcontroller has facilitate to communicate with jetson nano processor via buiod in USB. It is easy to use the mcu without an additional peripherals.
- Our stepper motor driver (CL57T-V41) identify 5v input as logic 1. To keep the threshold value of the logic state at that voltage value is essential for the communication with the motor driver.
- We use only 4 pwm pins for controlling the motor driver from available 7 pwm pins. We have sufficient number of i/o ports to drive the essentials.
- compared to ARM-based MCUs when only moderate performance is needed.

5.3.3 Coding

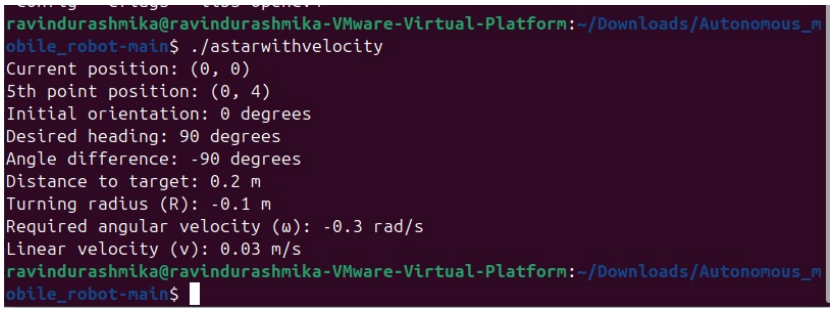
```
#include <avr/io.h>
#include <string.h>
#include <stdlib.h>
#include "m_usb.h"
#define F_CPU 16000000UL
#define MAX_PWM_FREQ 200000UL
#define MIN_PWM_FREQ 1UL
#include <util/delay.h>

#define BUFFER_SIZE 32
char inputBuffer[BUFFER_SIZE];
uint8_t bufferIndex = 0;
uint8_t pwm1Started = 0;
uint8_t pwm3Started = 0;

void InitPWM_OC1A() {
    // PWM Output (D9, PB5 = OC1A)
    DDRB |= (1 << PB5);

    // Motor Control Pins
    DDRD |= (1 << PD5);  // TX LED, Motor Enable
    PORTD &= ~(1 << PD5);
    DDRD |= (1 << PD4);  // D4, Motor Direction
    PORTD |= (1 << PD4);

    // Timer1 Configuration for Fast PWM, Mode 14 (ICR1 as TOP)
    TCCR1A = (1 << COM1A1) | (1 << WGM11);
    TCCR1B = (1 << WGM13) | (1 << WGM12); // Set mode first
    ICR1 = (F_CPU / 50000UL) - 1;  // Default 50 kHz
    OCR1A = ICR1 / 2; // 50% duty cycle
    TCNT1 = 0; // Reset counter
    TCCR1B |= (1 << CS10); // Start timer with no prescaler
}
```

```
void UpdatePWM_OC1A(uint32_t freq) {
    if (freq < MIN_PWM_FREQ || freq > MAX_PWM_FREQ) return;

    // Disable timer during update to prevent glitches
    uint8_t oldTCCR1B = TCCR1B;
    TCCR1B = (1 << WGM13) | (1 << WGM12); // Stop timer

    uint32_t top = (F_CPU / freq) - 1;
    if (top > 65535) top = 65535;
    if (top < 1) top = 1;

    ICR1 = (uint16_t)top;
    OCR1A = ICR1 / 2; // Maintain 50% duty cycle
    TCNT1 = 0; // Reset counter

    // Restore timer operation
    TCCR1B = oldTCCR1B;
}

void InitPWM_OC3A() {
    // PWM Output (D5, PC6 = OC3A)
    DDRC |= (1 << PC6);

    // Motor Control Pins
    DDRD |= (1 << PD7);  // D6, Motor Enable
    PORTD &= ~(1 << PD7);
    DDRD |= (1 << PD6);  // D12, Motor Direction
    PORTD &= ~(1 << PD6);

    // Timer3 Configuration for Fast PWM, Mode 14 (ICR3 as TOP)
    TCCR3A = (1 << COM3A1) | (1 << WGM31);
    TCCR3B = (1 << WGM33) | (1 << WGM32); // Set mode first
    ICR3 = (F_CPU / 50000UL) - 1; // Default 50 kHz
    OCR3A = ICR3 / 2; // 50% duty cycle
    TCNT3 = 0; // Reset counter
    TCCR3B |= (1 << CS30); // Start timer with no prescaler
}
```

```
void UpdatePWM_OC3A(uint32_t freq) {
    if (freq < MIN_PWM_FREQ || freq > MAX_PWM_FREQ) return;

    // Disable timer during update to prevent glitches
    uint8_t oldTCCR3B = TCCR3B;
    TCCR3B = (1 << WGM33) | (1 << WGM32); // Stop timer

    uint32_t top = (F_CPU / freq) - 1;
    if (top > 65535) top = 65535;
    if (top < 1) top = 1;

    ICR3 = (uint16_t)top;
    OCR3A = ICR3 / 2; // Maintain 50% duty cycle
    TCNT3 = 0; // Reset counter

    // Restore timer operation
    TCCR3B = oldTCCR3B;
}

void StopPWM_OC1A() {
    TCCR1A = 0;
    TCCR1B = 0;
    PORTB &= ~(1 << PB5);
    PORTD |= (1 << PD5); // Disable motor
    pwm1Started = 0;
}

void StopPWM_OC3A() {
    TCCR3A = 0;
    TCCR3B = 0;
    PORTC &= ~(1 << PC6);
    PORTD |= (1 << PD7); // Disable motor
    pwm3Started = 0;
}
```

```
// Function to parse frequency string (like "30.10") to Hz (30100)
uint32_t parseFrequency(const char* str) {
    char* dot = strchr(str, '.');
    if (dot == NULL) {
        // No decimal point, treat as integer Hz
        return atol(str);
    }

    // Split into integer and fractional parts
    char intPart[16] = {0};
    char fracPart[16] = {0};

    strncpy(intPart, str, dot - str);
    strncpy(fracPart, dot + 1, sizeof(fracPart) - 1);

    uint32_t integer = atol(intPart);
    uint32_t fraction = atol(fracPart);

    // Calculate how many digits we have in the fractional part
    uint8_t digits = strlen(fracPart);
    uint32_t multiplier = 1;

    // Calculate multiplier (10^digits)
    for (uint8_t i = 0; i < digits; i++) {
        multiplier *= 10;
    }

    // If we have something like "30.1", we need to make it "30.10" (2 digits)
    if (digits == 1) {
        fraction *= 10;
        multiplier = 100;
    }

    // Calculate final frequency in Hz
    return (integer * multiplier) + fraction;
}
```

The code below is responsible for providing velocity commands to the motors using the Jetson Nano. It takes the generated cost map and uses the A algorithm to plan an optimal path. Based on this path, it calculates the necessary translational and rotational velocities required for navigation and sends these commands to the microcontroller for motor control.



5.4 IMU (Inertial Measurement Unit)

5.4.1 Introduction

An Inertial Measurement Unit (IMU) is an electronic device that measures and reports an object's acceleration, angular velocity, and sometimes magnetic field orientation. It typically combines accelerometers, gyroscopes, and sometimes magnetometers into a single module. For our project we are using only the gyroscope.

Typically, from the gyroscope we read angular velocities along X, Y and Z axes.

Rotation speed around the X-axis – °/s (degrees per second)
Rotation speed around the Y-axis – °/s
Rotation speed around the Z-axis – °/s

X-axis – tilting forward/backward (pitch)
Y-axis → tilting left/right (roll)
Z-axis → rotating left/right on the spot (yaw)

- Since we are trying to control an amr robot using gyroscope what we basically need is the angular measurements around the Z axis.
- Rotation Tracking – You can track how much the robot has rotated by integrating angular velocity over time.

$$\theta(t) = \theta_0 + \int \omega(t)\, dt$$

- We are using the gyroscope in our robot to achieve precise turning control, for slip detection and dead reckoning.

5.4.2 Options

For our project we considered many IMUs and here are the best selections.

1. Analog Devices ADIS16470
2. Bosch BMI088
3. Bosch BNO055
4. TDK InvenSense ICM-42688-P
5. Murata SCC2230-D10

5.4.3 Comparison

Table 5: Evaluation of Gyroscopes for AMR

Sensor	**Gyro Drift (40%)**	**Gyro Range (30%)**	**Gyro Bandwidth (30%)**
Analog Devices ADIS16470	40 (3.5°/hr)	30 (2000°/s)	30 (800 Hz)
Bosch BMI088	30 (15 °/h (typical))	30 (2000°/s)	20 (300 Hz)
Bosch BNO055	20 (40–60 °/h)	30 (2000°/s)	15 (100 Hz)
TDK InvenSense ICM-42688-P	25 (20 °/h)	30 (2000°/s)	30 (1 kHz)
Murata SCC2230-D10	35 (5 °/h)	15 (300°/s)	30 (1 kHz)

We decided to go with the BNO055 because it just makes sense for what we need. It's affordable, easy to work with, and still gives us solid performance for tasks like orientation tracking and motion sensing. One of the biggest perks is that it handles sensor fusion on its own, so we don't have to deal with writing or tuning complex algorithms—it just gives us usable data right out of the box. That saves us time and effort, especially when working with limited processing power. Sure, it's not as precise as some of the high-end sensors, but for our use case, it does the job well without blowing the budget. It's a practical, plug-and-play solution that lets us focus more on building and less on debugging.

6 Wheel Selection

The wheel selection for our AMR was guided by mechanical requirements, expected load, and operational context. Specifically, the robot requires:

- **Two drive wheels:** 4-inch diameter, 14 mm shaft bore (motor coupling).
- **Two caster wheels:** 3-inch diameter for balancing and maneuverability.
- **Load capacity:** Each wheel should safely support a minimum of 10 kg, considering both static and dynamic loading.

We considered the following wheel types for evaluation:

1. **Pneumatic Wheels** – Air-filled rubber wheels with high shock absorption.
2. **Semi-Pneumatic Rubber Wheels** – Solid rubber wheels with internal air pockets for some cushioning.
3. **Thermoplastic Wheels** – Solid plastic wheels known for their durability, efficiency, and low maintenance.

Evaluation Criteria: Each wheel type was evaluated on a 10-point scale for the following criteria:

Criteria	**Pneumatic**	**Semi-Pneumatic**	**Thermoplastic**
Load Capacity	10	10	10
Shock Absorption	10	8	5
Rolling Resistance	6	7	9
Durability	6	8	9
Maintenance	4	8	10
Cost and Availability	4	7	9
Total Score (out of 60)	40	48	**52**

Conclusion: Thermoplastic wheels scored the highest overall (52/60), making them the most suitable choice for our AMR. While pneumatic wheels offer excellent shock absorption, they require high maintenance and are less cost-effective. Semi-pneumatic wheels performed moderately well but lacked the low rolling resistance and long-term durability of thermoplastics.

Given our indoor use-case, emphasis on durability, and minimal maintenance preference, **thermoplastic wheels were selected** as the optimal solution for both drive and caster wheel applications.

Figure 30: Selected 4-inch thermoplastic motor wheel with 14 mm bore

Specifications of Astro Heavy Duty Wheel:

- Wheel Diameter: 125mm (5 inch)
- Wheel Width: 34mm
- Rolling Bearing: Single ball bearing
- Load Capacity: 70kg
- Material: High Strength Nylon + High Quality TPR Rubber
- Wheel Type: Grey rubber + Ball bearings
- Applicable Temp.: -30~70 centi-degree
- Bearing Type: 6201

Specifications of Coupling:

- Material: High Grade Aluminum-Alloy (same high strength alloy as used in meccanum wheels)
- Bore size ID : 14mm hole for the shaft of the Motor
- Flange OD: 56.8mm
- Flange mounting holes : 6 holes x PCD 47.6 (compatible to meccanum wheels)
- Total thickness: 26.4mm
- Net weight : 50g

Figure 31: Data of the selected wheel

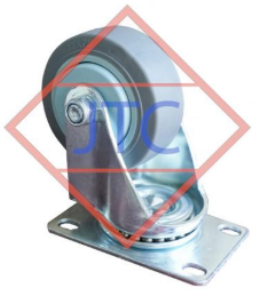
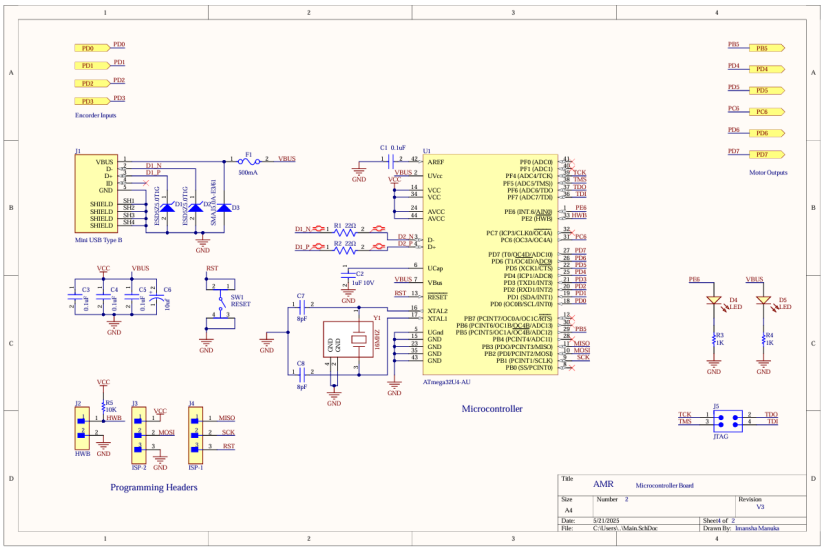


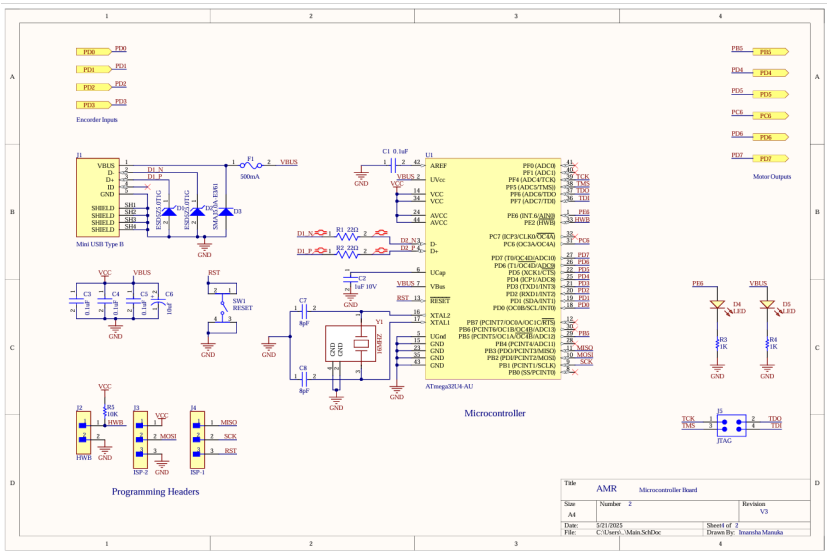
Figure 32: Selected 3-inch thermoplastic caster wheel

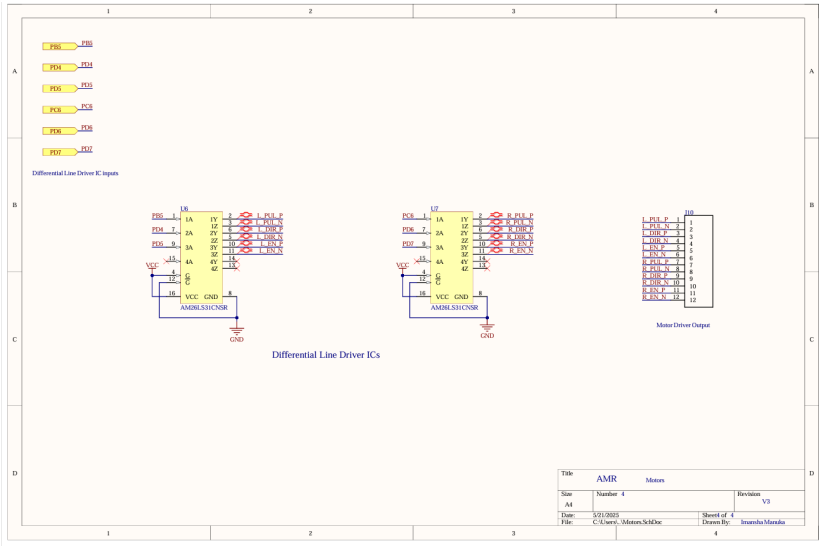
7 Schematic design/PCB

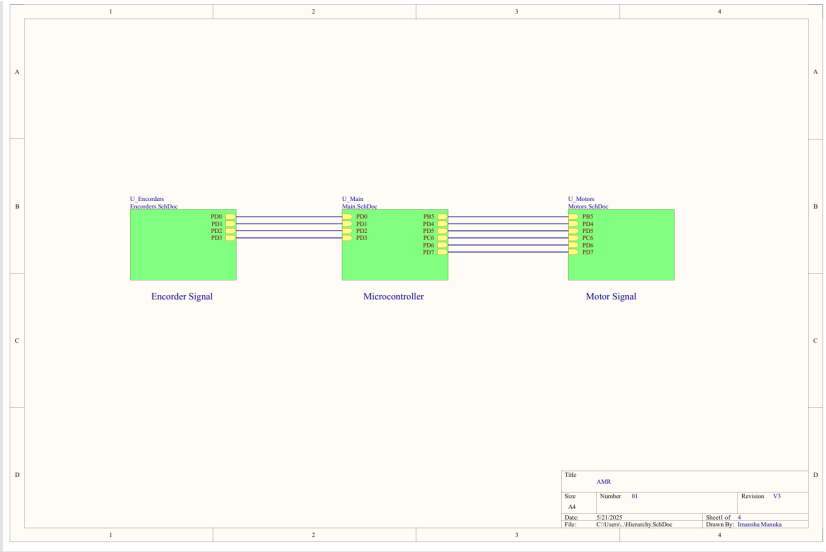
7.1 Microcontroller schematic

In this schematic, the microcontroller serves as the main control unit for stepper motor drivers. Communication with a host device is established through a Mini USB interface, which also supplies power to the board. For programming and firmware updates, dedicated male header connectors are provided, supporting in-system programming protocols.The microcontroller generates pulse, direction, and enable signals for two stepper motor drivers. These control signals are converted into differential pairs to improve noise immunity and ensure reliable transmission over longer distances to the motor drivers. Each signal line is routed to a terminal block for easy external connection.To handle feedback from the motors, the design supports two quadrature encoders, one for each motor. The encoder signals, which are differential in nature, are first conditioned and then buffered. This allows the same encoder signals to be safely split and shared between the microcontroller and motor driver. All encoder input and output connections are exposed via clearly labeled terminal blocks for straightforward integration.Additionally, the design includes proper power filtering capacitors, indicator LEDs, a crystal oscillator for accurate timing, and ESD protection for the USB data lines.




Encoder Inputs
Mini USB Type-B
Microcontroller
ATmega32U4-AU
Motor Outputs
Programming Headers
JTAG
Title
AMR
Microcontroller Board
Size
A4
Number 2
Revision
V3
Date: 5/21/2025
File: C:\Users\...\Main.SchDoc
Sheet of 2
Drawn By: Imansha Maruka

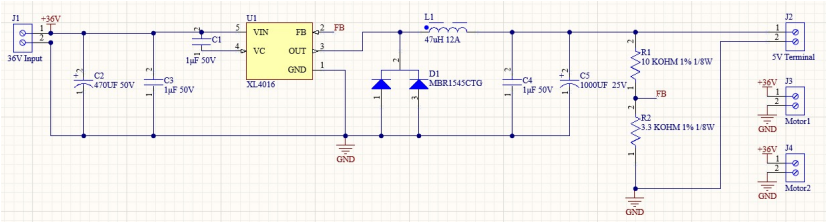

Differential Line Driver IC inputs
Differential Line Driver ICs
Motor Driver Output
Title
AMR
Motors
Size
A4
Number 4
Revision
V3
Date: 5/21/2025
File: C:\Users\...\Motors.SchDoc
Sheet of 4
Drawn By: Imansha Maruka



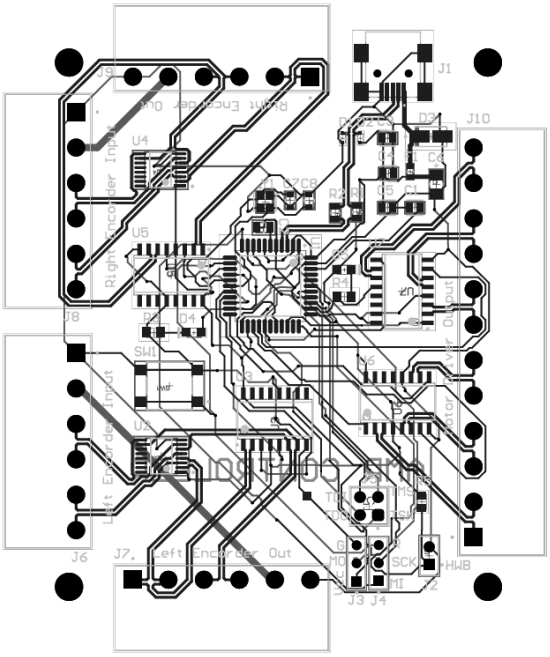
7.2 Power schematic

The presented schematic illustrates a DC-DC buck converter power supply designed to efficiently step down a 36V input to a regulated 5V output, suitable for powering control electronics and low-voltage peripherals. The design is centered around the XL4016 switching regulator IC, which supports high current outputs and offers good conversion efficiency for demanding applications. Voltage regulation is achieved using a feedback resistor network composed of a 10k and a 3.3k resistor, precisely setting the output to approximately 5V according to the IC's internal reference.

To ensure stable operation and reduce input voltage ripple, the circuit incorporates three 1F ceramic capacitors and one 470F electrolytic capacitor at the input. On the output side, a 1000F electrolytic capacitor provides effective bulk decoupling, helping to manage dynamic load conditions. A 47H, 12A inductor is used to maintain continuous current flow in the buck topology, and a Schottky diode (MBR1545CTG) is placed in the output path to provide fast switching, low forward voltage drop, and protection against reverse current flow.



7.3 Final PCB



8 Budget

8.1 Complete Budget(in LKR)

Component	Cost (LKR)
Processor – Nvidia Jetson Nano	67,600
RPLiDAR S2L – ToF LiDAR	95,000
Battery	25,000
Closed-loop Motors	70,000
IMU	15,300
Chip card	4000
Chassis	24,000
Battery	25,000
Couplers	8600
Push Button	800
5V adapter	1000
Step down/up module	1800
PCB and components	16335
Wheels	5020
Power Distribution Unit	3840
xt30 connectors	2000
Paint	5100
Power Box	5370
Amano sheet	3420
Holders for PCB	1130
Microcontroller	1830
Taxes	61,500
Total	**416,815**

9 Coding and Algorithms

9.1 SLAM Algorithm Development

This section outlines the algorithmic development of a Simultaneous Localization and Mapping (SLAM) system implemented from scratch in `C++`. The system is designed for a LiDAR-based Autonomous Mobile Robot (AMR), incorporating raw sensor data processing, mapping, localization, and path planning. Emphasis has been placed on modularity and simplicity to enable low-level control and understanding of SLAM fundamentals.

Overview of the Approach

The SLAM pipeline is divided into several functional components:

- **Sensor Data Acquisition:** LiDAR range measurements are obtained from the RPLIDAR S2L sensor and recorded as angle-distance pairs in CSV format (for simulation and current testing purposes only). Odometry data is captured from wheel encoders to estimate motion.
- **Range Data Preprocessing:** Each scan is transformed from polar to Cartesian coordinates, relative to the robot frame. This allows for easier integration into the occupancy grid map.
- **Occupancy Grid Mapping:** A 2D occupancy grid map is maintained, where each cell holds a probability value representing its occupancy status. Bresenham's Line Algorithm is used to mark free space between the sensor and detected obstacles. Future work will incorporate a Bayesian update mechanism to probabilistically refine occupancy states.

Sensor Fusion and State Estimation

To improve localization accuracy, a **sensor fusion** approach is being implemented using a **Kalman Filter** framework:

- **Motion Prediction:** The robot's motion is modeled using a differential drive kinematic model, and encoder-based odometry provides the initial estimate of the robot's state (pose and orientation).
- **Measurement Update:** LiDAR scan data is used to correct the predicted state using scan-to-map matching techniques (to be improved with ICP). The Kalman filter updates the state estimate by combining noisy motion and observation data.
- **Pose Tracking:** The robot's estimated position is continuously updated using the predicted pose and corrected using LiDAR-based observations. This reduces drift from odometry-only localization.

Keyframe Selection and Map Update

Keyframes are selected based on a distance or orientation threshold to limit map updates to significantly different robot poses. This approach reduces computational load and enhances the robustness of mapping. Each keyframe is used to update the map and the robot's trajectory.

Path Planning and Navigation

A grid-based **A* search algorithm** is used for path planning on the occupancy grid map. The robot can compute a path from its current location to a target goal while avoiding occupied cells. Future work includes dynamic re-planning and real-time trajectory smoothing.

Basic occupancy grid map generation

This picture shows the resulting occupancy grid generated by merging 80 LiDAR scans using ICP.

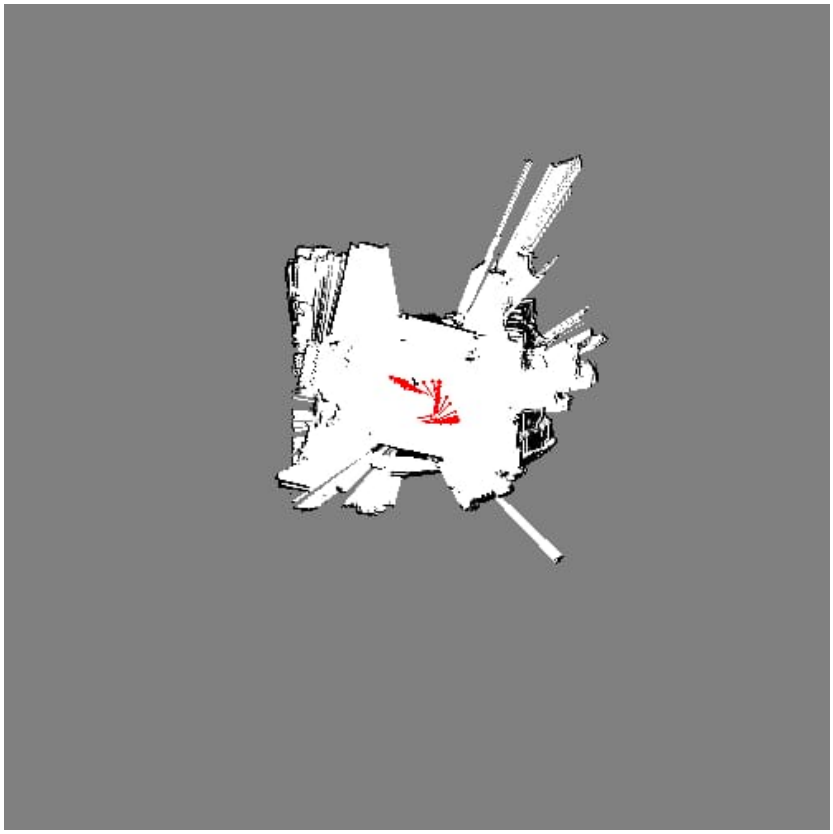


Figure 33: The resulting occupancy grid

This development marks the foundation for a full-featured SLAM system with accurate localization and robust navigation in dynamic environments.

10 Conclusion

The development of the Autonomous Mobile Robot (AMR) Controller has successfully addressed the core objectives outlined in the project scope. Through systematic design, analysis, and implementation, the following key achievements were realized:

- A robust mechanical structure capable of supporting a 5 kg payload, validated through Finite Element Analysis (FEA) to ensure structural integrity under operational loads.
- Integration of the RPLiDAR S2L sensor and NVIDIA Jetson Nano processor, enabling real-time SLAM with an angular resolution error of just 2.1 cm at 10 meters.
- Optimized motor and wheel selection, with thermoplastic wheels providing the ideal balance of durability, efficiency, and cost-effectiveness for indoor navigation.
- Development of a modular C++ SLAM pipeline incorporating sensor fusion, occupancy grid mapping, and A* path planning, laying the foundation for future enhancements like probabilistic mapping and loop closure.
- A stakeholder-aligned design process, ensuring the AMR meets industrial requirements for reliability, scalability, and user accessibility.