YILDIZ TECHNICAL UNIVERSITY
MECHANICAL ENGINEERING FACULTY

VISUAL NAVIGATION IN AUTONOMOUS VEHICLES

1906A033 Dilruba Cankurt

2006A022 Ceyda Biber

PREPARED IN THE DEPARTMENT OF MECHATRONICS ENGINEERING

UNDERGRADUATE GRADUATION PROJECT REPORT

Project Advisor: Asst. Prof. Ertuğrul Bayraktar

ISTANBUL, 2024

CONTENTS

	Page
REVISION HISTORY	3
LIST OF ABBREVIATIONS	1
FIGURE LIST	1
LIST OF TABLES	2
1. INTRODUCTION	**4**
1.1 Objective	4
1.2 Motivation	5
1.3 Scope	5
1.4 Literature Review	6
1.5. Assumptions and Presumptions	10
2. REQUIREMENT SPECIFICATIONS	**11**
2.1. Market Requirements	11
2.3 Design Requirements	12
3. THEORETICAL BACKGROUND INFORMATION	**13**
3.1 Mechanical System	13
3.2 Electrical System	13
3.2.1 Lidar Sensor	13
3.2.2 Ultrasonic Sensor	14
3.2.3 Camera Sensor	15
3.2.4 DC Motors	15
3.3 Algorithmic System	17
3.3.1 SLAM	17
3.3.2 Obstacle Avoidance	19
3.3.3 Object Detection	19
3.4 Control System	20
3.4.1 PID Control	20
3.4.2 Differential Wheel Control System	22
4. DESIGN	**25**
4.1 Overall System View	25
4.1.1 Mechanical Design	25
4.1.2 Algorithm Architecture	27
4.1.3 Electrical Architecture	28
5. DESIGN JUSTIFICATION	**32**
5.1 Test Plans	32
5.1.1 Mobile Robot	32
5.1.2 PID Control	33
5.1.3 LIDAR SLAM	35
5.1.4 Obstacle Avoidance	37
5.2 Design Justification Via Simulations	38

5.2.1 Mechanic Tests Ansys 38
5.2.2 CARLA Simulator 40
5.2.3 ROS 42
5.2.4 Gazebo/RViz Simulation 42
5.2.5 Differential Drive System 43
5.2.6 LIDAR SLAM 44
5.2.7 PID Control 46
5.2.8 Obstacle Avoidance 47
6. TIMELINE 49
7. BUDGET 50
7.1 Component List 50
7.2 Service Procurements 50
7.3 Application for External Support 51
REFERENCES 52

REVISION HISTORY

Date	Rev. No	Description	Writer(s)
05/01/2024	1.0	Initial document	Dilruba Cankurt Ceyda Biber
11/01/2024	1.1	MSD Final Report	Dilruba Cankurt Ceyda Biber
28/05/2024	1.2	Preliminary Report for Finals	Dilruba Cankurt Ceyda Biber
30/05/2024	1.3	Final Report	Dilruba Cankurt Ceyda Biber

LIST OF ABBREVIATIONS

SLAM Simultaneous Localization and Mapping

RMSE Root Mean Square Error

PSNR Peak Signal-to-Noise Ratio

SSIM Structural Similarity Index

CNN Convolutional Neural Network

ATE Absolute Trajectory Error

LiDAR Laser Imaging Detection and Ranging

RADAR Radio Detection and Ranging

IMU Inertial Measurement Unit

PID Proportional Integral Derivative

PPR Pulse Per Revolution

IOU Intersection Over Union

FIGURE LIST

Figure 1.4.1 Components of a typical SLAM system
Figure 1.4.2 Maps of Lidar and radar SLAM algorithms for smoky environments
Figure 1.4.3 The trajectory results of different SLAM algorithms on sequences 17-13-26-39 of the Oxford Radar RobotCar Dataset
Figure 1.4.4 The RMSE of Absolute Trajectory Error (ATE) for position on the MulRan Dataset is expressed in meters. 'N/A: SuMa could not complete sequences Riverside01, 02, and 03
Figure 1.4.5 Estimated trajectories and actual data in multiple sessions during normal, rainy, and snowy conditions and reference photos
Figure 1.4.6 Visual quality comparison in the Rain 200H dataset. Zooming in on the figures allows for a better view of drainage capacity
Figure 1.4.7 Comparison of quantitative results between synthetic and real datasets. Bold and underlined depict the best and second-best outcomes
Figure 2.1.1 Levels of Autonomous Driving
Figure 3.2.1.1 Lidar Sensor
Figure 3.2.2.1 Ultrasonic Sensor
Figure 3.2.3.1 Camera Sensor
Figure 3.2.4.1 Encoder DC Motor
Figure 3.3.1.1 LIDAR SLAM Architecture
Figure 3.4.1.1 PID control diagram
Figure 3.4.1.1 Differential Wheel System
Figure 4.1.1 Overall system diagram
Figure 4.1.1.1 Mechanical design
Figure 4.1.1.2 Technical drawing of lower body
Figure 4.1.1.3 Technical drawing of upper body
Figure 4.1.2.1 Visual Navigation Schematic

Figure 4.1.3.1 Electrical Design
Figure 5.1.1.1 Printed body parts
Figure 5.1.1.2 Before the electronic integration
Figure 5.1.1.3 : After the electronic and mechanic integration
Figure 5.1.2.1 Motor Testing Setup
Figure 5.1.2.2 Motor PID Control Testing
Figure 5.1.2.3 PID Control Results
Figure 5.1.3.1 Mapping of a dorm room
Figure 5.1.3.2 Mapping of a dorm 7th floor corridor
Figure 5.1.3.3 Mapping of the walkway
Figure 5.1.4.1 Obstacle Avoidance Testing Setup
Figure 5.2.1.1 Total deformation and total strain under 65N
Figure 5.2.1.2 Total deformation and total strain under 350N
Figure 5.2.2.1 Carla Enviroment
Figure 5.2.2.2 Lidar and Radar data obtained Carla Environment
Figure 5.2.5.1 Differential Drive System Simulation
Figure 5.2.5.2 Differential Drive Robot System
Figure 5.2.6.1 SLAM Algorithm Flow
Figure 5.2.6.2 LIDAR Data
Figure 5.2.6.3 Open Field Test
Figure 5.2.7.1 Main loop Decision Tree
Figure 5.2.8.1 Flow Diagram of Obstacle Avoidance
Figure 6.1 Timetable

LIST OF TABLES

Table 3.2.4.1 Motor Specifications Table
Table 7.1.1 Component prices

SUMMARY

The main objective of this project is to develop a specialized Simultaneous Localization and Mapping (SLAM) algorithm for autonomous mobile robots to provide more effective navigation in adverse weather conditions. To achieve this goal, a large-scale radar sensor and a high-resolution camera sensor will be integrated and the data obtained by these sensors will be processed through a complex algorithm. In addition, a suitable power source will be selected to meet the energy needs of the mobile robot and this source will be optimally placed in the design of the robot during the integration process. The developed algorithm will be extensively tested in both simulation and real-world tests under various scenarios and adverse weather conditions. These tests will play a critical role in evaluating the performance of the algorithms and shaping the development process. Once the project is completed, the results will represent a significant technological advancement in enabling autonomous vehicles to operate reliably not only in ideal weather conditions, but also in challenging atmospheric conditions. In this context, the successful completion of the project will contribute to a more reliable and effective deployment of autonomous mobile robot technology in a wide range of applications.

KEYWORDS: Autonomous Mobile Robot, SLAM, Visual Navigation

1. INTRODUCTION

In recent years, autonomous vehicles have emerged as a cutting-edge technology that has the potential to transform the way we commute, transport goods, and interact with our urban environments. Autonomous vehicles address a range of societal challenges, from reducing traffic accidents to enhancing mobility for people with disabilities and increasing the efficiency of transportation systems.

Self-driving cars, trucks, and even drones represent a remarkable convergence of advanced technologies, including artificial intelligence, machine learning, sensor technology, and connectivity. This harmonious integration of various fields not only underscores the immense potential of autonomous vehicles but also positions them as an exceptionally compelling choice for a mechatronics project.

1.1 Objective

This project aims to develop a navigation algorithm that prioritizes the importance of radar sensor data for autonomous/driverless vehicles in adverse weather conditions when the most commonly used sensors such as lidar and camera fail to transmit reliable data.

When existing SLAM algorithms are analyzed under adverse weather conditions, comparisons based on Absolute Trajectory Error, Root Mean Square Error (RMSE) and Relative Error metrics for Position are important. The results of the comparison of the improved RadarSLAM algorithm with other algorithms based on the Relative Error metric show that it performs the best with an average value of 1.83/0.6 (Translation error/Rotation error) [1]. At the same time, it was decided to utilize image enhancement technology in the project to ensure the reliability of the data obtained from the camera in adverse weather conditions. Among the available image enhancement technologies, the PSNR (Peak Signal-to-Noise Ratio) [2] and SSIM (Structural Similarity Index) [3] results obtained in tests on the Rain200L [4] dataset using Convolutional Neural Network (CNN) and Transformer-based methods such as DualGCN [5], SPDNet [6], Restormer [7] and DRSformer [8] were evaluated. In this context, DRSformer was found to perform at the highest level with a PSNR of 41.23 and SSIM of 0.9894 [8]. As a result of the literature review, it was observed that position errors vary between 13.784 meters and 3.917 meters in

different datasets depending on weather conditions [1]. With reference to these studies, it is aimed to process radar, lidar and image data with the algorithms to be developed and to increase the position estimation accuracy and to reduce the error range below the current best results.

1.2 Motivation

The main motivation for this project is to reduce accidents caused by human error and consequently reduce the loss of life and property in accidents. The biggest advantage that vehicles have over drivers is the ability to anticipate possible scenarios. This ability is based on advanced sensors such as Radar, Lidar and cameras. In particular radar can obtain accurate data even in foggy, dark and rainy driving environments. This means that in those conditions the autonomous vehicles can instantly detect scenarios that humans cannot foresee and controllers can take action accordingly.

1.3 Scope

The primary goal of our project is to significantly elevate the safety and efficiency of autonomous vehicles navigating through challenging environmental scenarios. Addressing conditions like low-light settings, adverse weather encompassing snow, rain, and fog, as well as dense vegetation cover, we're undertaking a multifaceted approach. Our scope spans several key areas, beginning with the refinement of visual navigation. Leveraging radar based SLAM algorithm and improving quality of image data , we're striving to enhance the vehicle's ability to comprehensively map and recognize its surroundings. Furthermore, we're delving into the realm of multi-sensor integration, employing a range of tools from LIDAR and RADAR to cameras and various other sensors. Our aim is to ensure continuous environmental perception and data collection, supporting the vehicle's awareness across diverse conditions. Additionally, we're rigorously evaluating existing SLAM algorithms (Simultaneous Localization and Mapping) with an eye towards enhancements wherever possible, maximizing their accuracy and adaptability. Lastly, our validation processes involve tests within the CARLA simulator and validating the efficiency and precision of our algorithms through simulated experiments, ensuring real-world readiness and reliability. After ensuring the reliability of our approach real life scenarios panned to be tested via mobile robot design.

1.4 Literature Review

The research began with an exploration of the fundamentals of the SLAM algorithm to gain a deeper understanding, aiming to identify problems and potential solutions. Our initial focus involved analyzing a standard SLAM component, as visually outlined in the provided schematic in Figure 1.4.1.

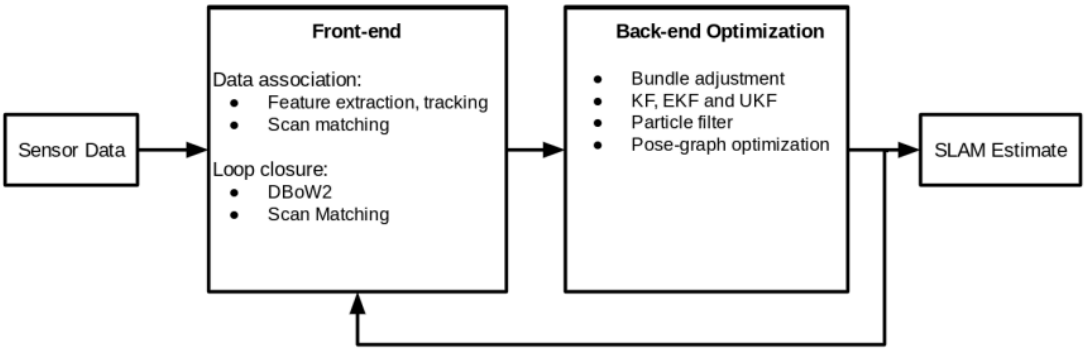
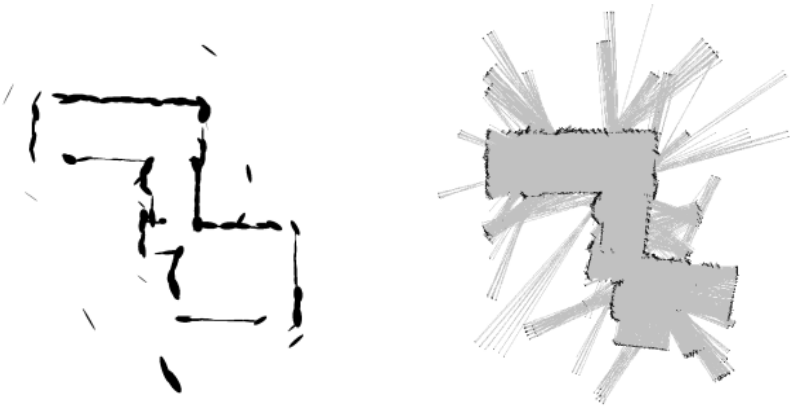


Figure 1.4.1: Components of a typical SLAM system[9]

An in-depth exploration of existing literature reveals a shortfall in relying solely on SLAM algorithms for autonomous driving, prompting the observed trend towards developing fusion technologies using diverse sensor arrays [10].

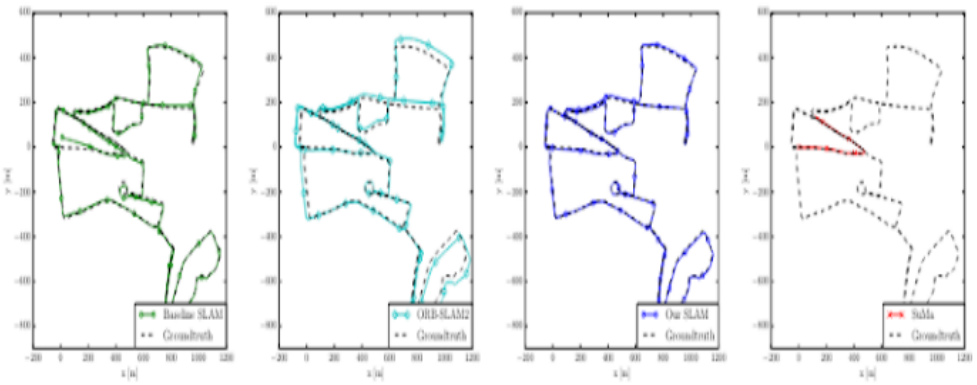
Our project's diligent literature review displays the unreliability of data sourced from camera and lidar sensors under adverse weather conditions. To support this unreliable data, we've opted to integrate radar alongside the camera and lidar [11]. The laser beam used in Lidar technology experiences deviations in adverse weather conditions, leading to uncertainties and rendering it insufficient when used alone due to this sensitivity. On the other hand, radar technology measures the distance and speed of objects using electromagnetic waves and remains unaffected by weather conditions. The lower cost of radar sensors compared to Lidar sensors makes radar usage appealing. Therefore, the decision has been made to utilize a radar-based SLAM algorithm. To reinforce this idea, studies comparing radar-based and Lidar-based SLAM algorithms have been examined and as seen in Figure 1.4.2, environment mapping obtained using a radar-based SLAM algorithm can similarly be achieved using a Lidar-based SLAM [11].



(a) Lidar based SLAM **(b) Radar based SLAM**

Figure 1.4.2 :Maps of Lidar and radar SLAM algorithms for smoky environments [11]

Among the studies comparing Lidar and radar-based SLAM algorithms, the ones done in adverse weather conditions aligns with the project's objective. The tests were conducted on the Oxford RobotCar [12,13], MulRan [14], and RADIATE [15] datasets using KITTI odometry evaluation criteria. These tests encompassed smoky, rainy, snowy, sunny, and nighttime weather conditions. In Figure 1.4.3, trajectory data obtained from sequences 17-13-26-39 of the Oxford RobotCar [12,13] dataset is compared between ORB-SLAM2 [16], RadarSLAM [17], SuMa [18], and Improved RadarSLAM [1] against actual trajectory data. While Lidar-based SuMa [18] couldn't complete sequences, it was observed that OBR-SLAM2 [16] also yielded more erroneous results compared to radar-based SLAM algorithms.



a)RadarSLAM b) ORB-SLAM2 c) Improved RadarSLAM d) SUMA

Figure 1.4.3: The trajectory results of different SLAM algorithms on sequences

17-13-26-39 of the Oxford Radar RobotCar Dataset [1].

In Figure 1.4.4, a comparison is presented based on the RMSE metric of the trajectory results obtained by RadarSLAM [17], SuMa [18], and Improved RadarSLAM [1] algorithms on different sequences of the MulRan [14] dataset. Since the dataset comprises solely Lidar and radar data, ORB-SLAM2 [16] is not included in this comparison. Radar-based SLAM algorithms have outperformed the Lidar-based SLAM algorithm, SuMa [18], by completing all sequences with better results.

	Sequence								
Method	DCC01	DCC02	DCC03	KAIST01	KAIST02	KAIST03	Riverside01	Riverside02	Riverside03
SuMa	13.509	17.834	29.574	38.693	31.864	45.970	N/A	N/A	N/A
Baseline SLAM	17.458	24.962	76.138	4.931	3.918	50.809	10.531	95.247	1091.605
Our SLAM	**12.886**	**9.878**	**3.917**	**6.873**	**6.028**	**4.109**	**9.029**	**7.049**	**10.741**

Figure 1.4.4: The RMSE of Absolute Trajectory Error (ATE) for position on the MulRan Dataset is expressed in meters. 'N/A: SuMa could not complete sequences Riverside01, 02, and 03 [1].

In Figure 1.4.5, observations from a study using the RADIATE [15] dataset showcase a comparison between results obtained from RadarSLAM [17], SuMa [18], and Improved RadarSLAM [1] algorithms against data acquired via GPS across various weather conditions. It is observed that in adverse weather conditions, radar-based algorithms complete trajectories with less error. The quality of data from the camera is detailed in item d of Figure 1.4.5, highlighting that ORB-SLAM2 [16] did not yield reliable results due to blurring effects caused by water droplets, obscured visibility in low light conditions, and snow, hence it was not included in the comparison.

a) Regular weather sequence b) Rainy weather sequence c) Snowy weather sequence d) Reference data

Figure 1.4.5: (a)-(c): Estimated trajectories and actual data in multiple sessions during normal, rainy, and snowy conditions and reference photos [1].

Beyond SLAM algorithms, our project delves into enhancing visual navigation through exploring image enhancement technologies. The examined studies have demonstrated that the success of a mobile robot's visual navigation is closely related with the quality and clarity of the data captured by the camera. However, images obtained from the camera under adverse weather conditions do not meet the requirements for visual navigation. In this context, image enhancement algorithms play a critical role. These algorithms typically excel in handling challenging situations such as low-light conditions, high levels of noise, or blurry images.

These algorithms typically stand out for their ability to handle challenging situations such as low-light conditions, high levels of noise, or blurry images. The results obtained by the MPRNet [19], SPDNet [6], Uformer [20], Restormer [7], IDT [21], and DRSformer [8] algorithms when provided with rainy input from the Rain 200H [22] dataset are depicted in Figure 1.4.6.

Rainy Input MPRNet SPDNet Uformer Restormer IDT DRSformer Reference

Figure 1.4.6: Visual quality comparison in the Rain 200H dataset. Zooming in on the figures allows for a better view of drainage capacity [8].

Effective image enhancement for visual navigation holds critical importance in terms of reliability and accuracy in real-time applications. The table in Figure 1.4.7 compares existing image enhancement methods based on PSNR [2] and SSIM [3] metrics across Rain200L [4], Rain200H [22], DID-Data [23], DDN-Data [24], and SPA-Data [25] datasets [8]. The transformer-based image enhancement method we will use in our project aims to improve the visual navigation by targeting better results on images, aiming for a PSNR of 41.23 and an SSIM of 0.9894, as achieved by DRSformer.

Datasets		*Rain200L*		*Rain200H*		*DID-Data*		*DDN-Data*		*SPA-Data*	
Metrics		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Prior-based methods	DSC [30]	27.16	0.8663	14.73	0.3815	24.24	0.8279	27.31	0.8373	34.95	0.9416
	GMM [24]	28.66	0.8652	14.50	0.4164	25.81	0.8344	27.55	0.8479	34.30	0.9428
CNN-based methods	DDN [8]	34.68	0.9671	26.05	0.8056	30.97	0.9116	30.00	0.9041	36.16	0.9457
	RESCAN [23]	36.09	0.9697	26.75	0.8353	33.38	0.9417	31.94	0.9345	38.11	0.9707
	PReNet [36]	37.80	0.9814	29.04	0.8991	33.17	0.9481	32.60	0.9459	40.16	0.9816
	MSPFN [19]	38.58	0.9827	29.36	0.9034	33.72	0.9550	32.99	0.9333	43.43	0.9843
	RCDNet [43]	39.17	0.9885	30.24	0.9048	34.08	0.9532	33.04	0.9472	43.36	0.9831
	MPRNet [59]	39.47	0.9825	30.67	0.9110	33.99	0.9590	33.10	0.9347	43.64	0.9844
	DualGCN [9]	40.73	0.9886	31.15	0.9125	34.37	0.9620	33.01	0.9489	44.18	0.9902
	SPDNet [56]	40.50	0.9875	31.28	0.9207	34.57	0.9560	33.15	0.9457	43.20	0.9871
Transformer-based methods	Uformer [48]	40.20	0.9860	30.80	0.9105	35.02	0.9621	33.95	0.9545	46.13	0.9913
	Restormer [58]	40.99	0.9890	32.00	0.9329	35.29	0.9641	34.20	0.9571	47.98	0.9921
	IDT [50]	40.74	0.9884	32.10	**0.9344**	34.89	0.9623	33.84	0.9549	47.35	**0.9930**
	DRSformer	**41.23**	**0.9894**	**32.17**	0.9326	**35.35**	**0.9646**	**34.35**	**0.9588**	**48.54**	0.9924

Figure 1.4.7: Comparison of quantitative results between synthetic and real datasets. Bold and underlined depict the best and second-best outcomes [8].

1.5. Assumptions and Presumptions

I. Aerodynamic drag force is negligible.

II. The autonomous mobile robot is assumed to encounter a maximum slope of 10 degrees.

III. The maximum speed of the autonomous mobile robot is designed to be 8.3 m/s

IV. When simulating autonomous mobile robot body analyses, it was assumed that there was no lidar sensor in the area where the force was applied.

V. It is assumed that the wireless connectivity features of the ESP-32 CAM module will effectively transmit image data.

VI. The ESP-32 C3 module is expected to ensure seamless data transmission between the YDLidar X2 and Jetson Nano.

VII. The communication between the Arduino Uno and Jetson Nano is assumed to be uninterrupted and reliable.

VIII. It is also assumed that the batteries will provide sufficient power for the expected operational duration without unexpected power interruptions.

2. REQUIREMENT SPECIFICATIONS

2.1. Market Requirements

Advancements in autonomous vehicle technology have revolutionized the automotive industry, classifying these vehicles into six distinct levels based on their driving capabilities, as illustrated below in Figure 2.1.1. Among the players in the market, Waymo made significant strides by completing the world's first level 4 vehicle in 2015 [26]. Our project's foundation aims to contribute to reaching level 5 in autonomous driving.

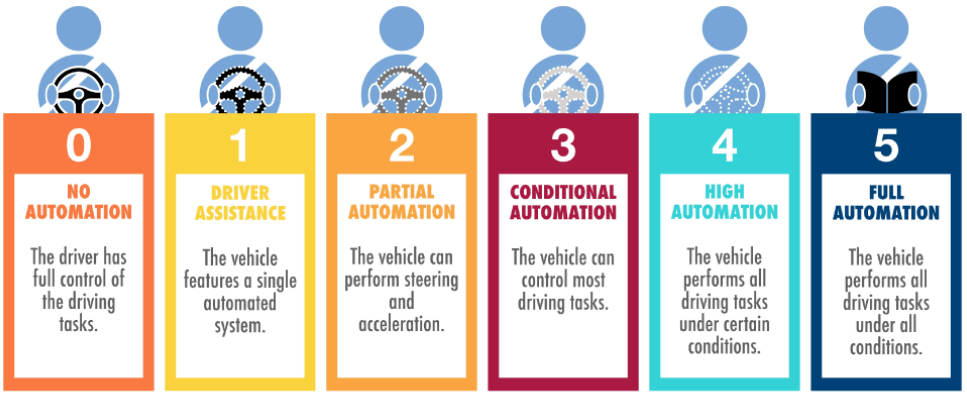


Figure 2.1.1: Levels of Autonomous Driving [27].

When examining the current autonomous vehicles, their capability to ensure autonomous driving in rainy conditions appears insufficient. For instance, Tesla models require the driver to keep their hands on the wheel during rainy weather, as the existing technology isn't sufficiently reliable for rainy driving.

2.2 Technical Requirements

For processing image data, we have chosen the ESP-32 CAM module, which meets our criteria of an 8MP HDR camera sensor with 30 fps. For evaluating ultrasonic sensor data, our requirements include a sensor with a 1-4 m range, 40kHz operating frequency, and a 20-degree angle. The HC-SR04 ultrasonic sensor was selected to meet these needs. For the Lidar sensor in our project, we specified the need for a 2D Lidar with a 360-degree range. The YDLidar X2 Lidar Distance Sensor was chosen to fulfill this requirement.

2.3 Design Requirements

To ensure balanced and effective movement on various surfaces such as asphalt, soil, grass, concrete, etc., the design will incorporate 4 large wheels. Sensor integration will enable the vehicle to effectively perceive its surroundings and successfully perform autonomous navigation tasks. The vehicle should be designed to be able to resist environmental factors such as weather conditions and capable of supporting the weight of the components, approximately 12kg. The positions of the sensors in the vehicle design will be determined to minimize their sensitivity to external factors and maximize the efficiency of the data gathering process in dynamic environmental conditions.

3. THEORETICAL BACKGROUND INFORMATION

In a Mechatronic System Design project, various mathematical expressions and equations, theoretical explanations, and software analyses are utilized for the systems and subsystems used. These methods form the foundational building blocks of the project.

3.1 Mechanical System

In the mechanical design of our autonomous mobile robot, four large wheels, each driven by an individual motor, will be used to ensure optimal performance in challenging conditions. These large wheels, paired with differential control, will enhance the robot's maneuverability and stability on various surfaces. By controlling the speed and direction of each wheel independently, the robot can make precise turns and navigate effectively in complex environments. This design supports our autonomous mobile robot in achieving maximum performance and maneuverability, both in everyday use and under challenging conditions, ensuring the successful completion of its tasks

3.2 Electrical System

Our autonomous mobile robot integrates various sensor technologies to provide high-level precision and reliability in environmental perception. Through these sensors, the robot enhances its ability to understand the surroundings, detect objects, and interact with its environment.

3.2.1 Lidar Sensor

Lidar (Light Detection and Ranging) is a sensor technology that utilizes laser beams for environmental mapping and distance measurement. Our autonomous mobile robot optimizes its environmental perception capabilities by integrating the YDLidar X2 Lidar Distance Sensor. This lidar sensor offers fast and precise data collection with a variable frequency of up to 3kHz. Additionally, it provides a variable measurement range of at least 8 meters, allowing the robot to reliably map its surroundings and detect objects in a wide area. The sensor scans its surroundings comprehensively with a 360-degree omnidirectional scanning capability, and the motor speed is adjustable (recommended speed is 6kHz). Moreover, it exhibits strong

resistance to ambient light interference, ensuring accurate measurements even in environments with varying external light conditions. The YDLidar X2 Lidar Distance Sensor shown in Figure 3.2.1.1, optimizes our robot's environmental perception capabilities, thereby enhancing its autonomous movement capability.

Figure 3.2.1.1: Lidar Sensor

3.2.2 Ultrasonic Sensor

Ultrasonic sensors use sound waves to measure distance and detect objects. Our robot employs high-precision sensors to accurately determine the positions of surrounding objects. This is a fundamental feature for the robot to detect obstacles and navigate safely.

Our autonomous mobile robot's ultrasonic sensor system is based on the HC-SR04 model. This sensor can measure distances from 2 cm to 4 meters with an accuracy of ±3 mm. Operating at a frequency of 40 kHz, the HC-SR04 works effectively under various environmental conditions. With a working voltage of 5V DC, the sensor can be easily integrated and controlled via the microcontroller using the trig and echo pins. These features enable the sensor to efficiently scan the robot's surroundings and optimize its detection capabilities.

Figure 3.2.2.1: Ultrasonic Sensor

3.2.3 Camera Sensor

Camera Sensor To enhance our robot's vision capabilities, a high-resolution camera is employed. The ESP32-CAM module, represented in Figure 3.2.3.1, is the camera used in our project. This module features an OV2640 sensor and provides a 2 MP resolution. With a wide 66.5-degree field of view, this module effectively captures various objects.

The ESP32-CAM module offers wireless communication via built-in Wi-Fi and Bluetooth, making remote control easy. It also includes a microSD card slot that supports up to 4 GB of storage for saving images and videos. The module has a built-in LED flash that can be controlled with Python scripts, allowing clear images even in dark environments.

Additionally, the camera module leverages the powerful processing capabilities of the ESP32, enabling high-performance image processing. It can stream video at resolutions up to 1600x1200 pixels (UXGA) at 15 FPS. The compact design and low power consumption of the ESP32-CAM make it a suitable choice for our project.

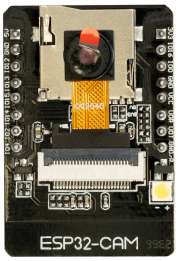


Figure 3.2.3.1: Camera Sensor

3.2.4 DC Motors

We utilize two different types of DC motors to facilitate movement and achieve high-precision control for our robot. Both motors are equipped with internal encoders, enabling precise position and speed control.

- **MPBFN 12V DC Motor:** The MP5BFN 12V DC Motor is employed for the two

wheels of our autonomous mobile robot. This motor stands out for its high-speed operation and lack of a gearbox. With its built-in encoder, it provides precise position and speed feedback. This motor is ideal for applications requiring high speed, and its internal encoder enables precise control through feedback. These motors enable our robot to move swiftly and accurately, making them suitable for agile and dynamic tasks.

- **12V 350 RPM Encoder DC Motor:** The other two wheels are equipped with 12V 350 RPM Encoder DC Motors. These motors provide higher torque and operate at lower speeds due to their geared structure. These motors allow the robot to handle applications requiring high torque at low speeds, particularly for overcoming obstacles and executing precise maneuvers. With the encoder, position and speed feedback ensure the robot's stable and controlled movement.

Both motors are selected to meet the diverse needs of our robot. While MP5BFN motors excel in scenarios requiring high speed and precise control, the 12V 350 RPM motors deliver superior performance for tasks demanding high torque and low speed. With their internal encoders, both motors provide precise control and feedback, thereby ensuring the safe and efficient operation of our mobile robot. The key specifications of these motors are shown in the table 3.2.4.1.

	MP5BFN 12V EncoderDC Motor	**12V 350 RPM Encoder DC Motor**
Voltage (V)	12 V DC	12V DC
No-Load Speed (RPM)	1500 RPM	350 RPM
No-Load Current (A)	Approximately 0.2 A	Approximately 0.3 A
Encoder Resolution (PPR)	11 PPR	11 PPR or higher
Torque (kg.cm)		2.0 kg.cm

Table 3.2.4.1: Motor Specifications Table

a) MP5BFN **b) 12V 350 RPM**

Figure 3.2.4.1: Encoder DC Motor

3.3 Algorithmic System

3.3.1 SLAM

Two critical aspects for autonomous vehicles are Location and Mapping. Knowing the vehicle's exact position and devising plans for the next steps based on this location data are essential.

Lidar SLAM

LiDAR SLAM (Simultaneous Localization and Mapping) enables autonomous systems to create maps of their environments while tracking their own location within these maps in real-time. This technology is not only used in autonomous vehicles but have its place in wide range of applications from robotics to drones and augmented reality.LiDAR, which stands for Light Detection and Ranging, uses laser beams to measure distances to objects, providing highly accurate spatial data. SLAM algorithms process this data to generate maps and determine the system's position and orientation within those maps. The general view for this process is given in Figure 3.3.1.1.

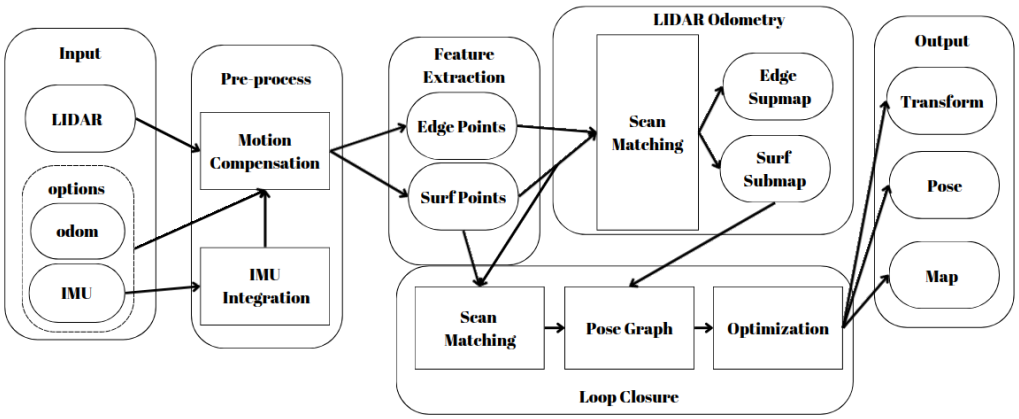


Figure 3.3.1.1: LIDAR SLAM Architecture

There are various types of LiDAR SLAM, including 2D and 3D, each suited for different applications and environments. While 2D LiDAR SLAM is commonly used for navigation on flat surfaces and in simpler environments, 3D LiDAR SLAM is essential for complex, three-dimensional spaces where understanding height and depth is critical. In this project due to budget constraints we opted for 2D SLAM options.

2D LiDAR SLAM involves using 2-dimensional LiDAR sensors to create a detailed map of an environment while simultaneously localizing the sensor within that map. The LiDAR sensor emits laser beams in multiple directions, measuring the time taken for these beams to bounce back after hitting objects in the environment. By analyzing the return times, the sensor constructs a precise 2D representation of its surroundings. SLAM algorithms then process this continuous stream of LiDAR data to determine the sensor's position and orientation in real-time. This complex process includes several steps such as feature extraction, where distinct landmarks or features in the environment are identified from the LiDAR data, and matching these features between successive scans to track changes and movement. The algorithms also estimate the robot's movement, adjusting the map and the robot's location within it to reflect new data accurately. This simultaneous mapping and localization enable the robot to navigate effectively and build an ever-evolving map, crucial for real-time decision-making and navigation in dynamic environments.

3.3.2 Obstacle Avoidance

Obstacle avoidance in autonomous vehicles is the capability to detect and navigate around obstacles or potential hazards in their environment without human intervention. To achieve true autonomous navigation, obstacle avoidance is a crucial part.

While there are many ways to avoid obstacles, using an ultrasonic sensor is a common choice due to its reliability and cost-effectiveness. Ultrasonic sensors emit sound waves at a higher frequency than the human ear can hear. These sound waves travel through the air and bounce back when they hit an object. The sensor then measures the time it takes for the echo to return and calculates the distance to the object based on the speed of sound. Using this simple but effective method, the autonomous vehicle can accurately detect objects in its path and make real-time decisions to avoid collisions. Ultrasonic sensors are particularly useful in environments with poor visibility where optical sensors may have difficulties.

Upon finalizing the navigation the number of collisions made by the robot will be evaluated to understand the success of the navigation system.

3.3.3 Object Detection

Object detection is a computer vision technology that aims to understand what is shown in a given image and where that said object is within the image. Not only that it will also return a confidence level for the prediction. When it comes to autonomous systems, such as self-driving cars or intelligent robots, accurate object detection is crucial for ensuring safe navigation through dynamic environments.

It also helps improve the obstacle avoidance methods, by accurately identifying objects in the vehicle's path, whether it be pedestrians, vehicles, traffic signs, or any other potential hazards, the system can take measures to avoid collisions and ensure the safety of both the passengers and the surrounding environment.

Image processing and machine learning techniques allow object detection systems to analyze the captured data, identify objects of interest and accurately determine their position relative to the vehicle. This information is then used to make informed decisions regarding navigation, such as adjusting speed, changing lanes or stopping completely if necessary.

As with any autonomous navigation system, the success of obstacle avoidance depends heavily on the effectiveness of the object detection module. It is important to test the algorithm in various environments and record the result to analyze properly. Usually precision/recall curve and average precision is chosen as evaluating metrics.

When it comes to understanding the procedure there are some important terms to know such as: intersection over union (IOU), True Positive, True Negative, False Negative, and False Positive.

- IOU: By assessing how much two bounding boxes overlap using the threshold percentage (typically 50%, 75%, or 95%), it determines if a detection is accurate or not.
- True Positive: Correct detection (IOU ≥ threshold)
- False Negative: A missed detection.
- False Positive: Wrong detection (IOU < threshold).
- True Negative: Not applicable in object detection.

In our project we plan to use this method to find the wanted objects in an unknown area. While our robot is exploring the environment and creating a map, the camera will identify what is in that said environment and send the visual stream to a computer that is connected to the same WIFI as our camera model.

3.4 Control System

3.4.1 PID Control

Proportional-Integral-Derivative (PID) control is a widely used feedback control technique in engineering and industrial applications. It provides a method for controlling dynamic systems and processes to achieve desired performance by minimizing errors between a desired setpoint and the actual process variable.

PID control is based on three main concepts: proportional gain, ıntegral gain and the derivative gain.

a. The proportional term (K_p) aims to reduce the position error to zero by adjusting the control signal in proportion to the current position error. Essentially, it strives to move the output towards the setpoint.

b. The derivative term (K_d) aims to reduce the rate of change of the error to zero by adjusting the control signal in proportion to the derivative of the error. Essentially, it strives to make the output move at the same rate as the setpoint.

c. The integral term (K_i) aims to reduce the total accumulated error to zero by adjusting the control signal in proportion to the sum of all past errors. Essentially, it strives to align the average of all past output values with the average of all past setpoint values.

The overall system can be seen in Figure 3.4.1.1 where;

- r(t): setpoint
- e(t): error at time t
- u(t): control output
- y(t): output
- K_p is the proportional gain.
- K_i is the integral gain.
- K_d is the derivative gain.

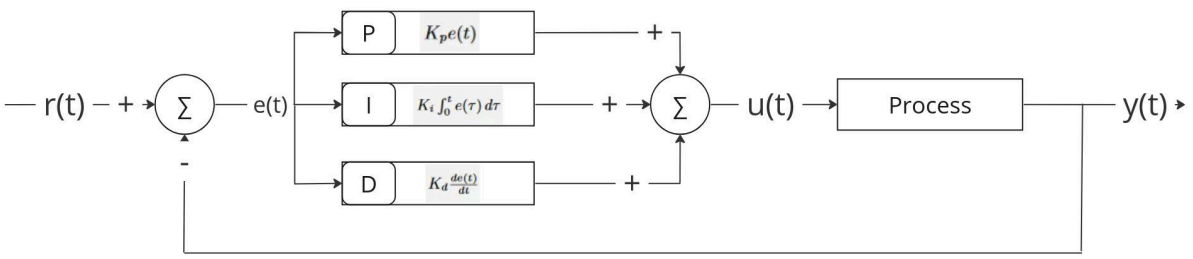


Figure 3.4.1.1: PID control diagram

Mathematical representation of the system is given by the formula (1) in below:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d\,e(t)}{dt} \quad (1)$$

PID control is simple and easy to implement and tune which makes it an advantageous choice for many system designs. However tuning the gains requires time and the system is sensitive to changes and noise.

3.4.2 Differential Wheel Control System

To provide precise and flexible movement control during operation, our autonomous mobile robot uses a differential drive system. Differential drive is a common and effective method for maneuvering wheeled robots, allowing a wide range of movements such as moving in a straight line, making smooth turns, and rotating in place. The geometrical representation of the system is given in Figure 3.4.1.1.

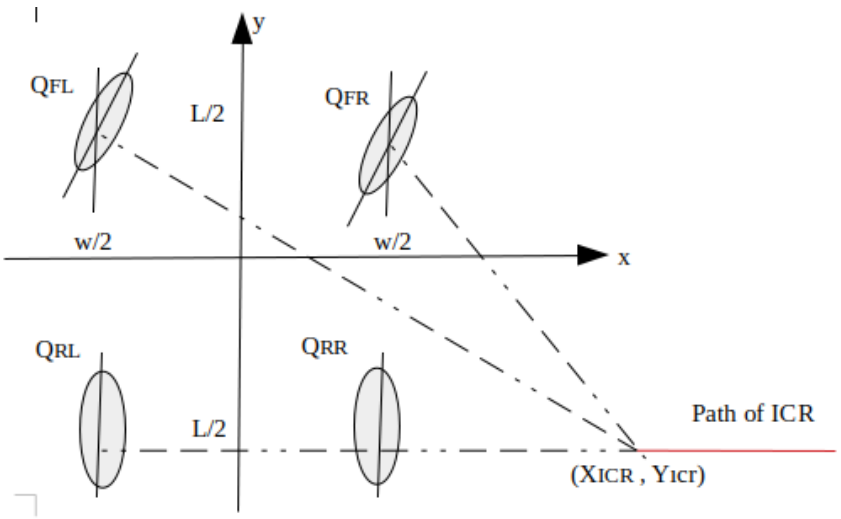


Figure 3.4.1.1: Differential Wheel System

Differential drive uses two independently controlled motors, each attached to one of the robot's drive wheels (since our vehicle has four wheels, we use four independent motors). By varying the relative speeds and directions of these motors, the mobile robot can move forward, backward, turn in place, or follow a curved path. The basic operating modes are as follows:

- Straight movement is achieved by running both motors at the same speed and in the

same direction, allowing the robot to move forward or backward.

- To turn, the speeds of the two motors are varied; for example, to turn right, the left motor runs faster than the right motor, and to turn left, the opposite applies.
- Rotation in place is achieved by running both motors at the same speed but in opposite directions, causing the robot to rotate around its center.

Motor Configuration

Our differential drive system is implemented using two types of encoder-equipped DC motors:

➔ **MP5BFN 12V Encoder DC Motors:** These motors are used for two of the main drive wheels. Operating at 12V DC with a speed of 1500 RPM, these high-speed motors provide quick and precise movements. The built-in encoders provide feedback for precise speed and position control.

➔ **350 RPM 12V Encoder DC Motors:** These motors are designed to provide higher torque at lower speeds, operating at 12V DC with a speed of 350 RPM and delivering 2.0 kg.cm of torque. These motors ensure robust performance under load and allow precise control at lower speeds.

Control System

The control of the differential drive system is handled by an Arduino Uno microcontroller, which processes sensor data and manages motor control signals. The motor drivers control the speed and direction of the DC motors. The control loop involves collecting sensor data, path planning, and generating motor control signals. The Arduino Uno serves as a central hub, gathering data from the YDLidar X2, HC-SR04 sensors, and ESP-32 CAM, and sending control signals to the L298N motor drivers. The Jetson Nano handles high-level tasks such as image recognition and advanced navigation algorithms and coordinates with the Arduino Uno. The ESP-32 C3 sends precise distance measurements from the YDLidar X2 to the Jetson Nano, enhancing the robot's obstacle detection and avoidance capabilities.

Advantages of Differential Drive System

The differential drive system offers excellent maneuverability, simplicity, and precision. Compared to other drive systems, it requires fewer mechanical components, making it simpler

and more reliable. The encoder feedback allows for precise control of the robot's movements, which is a significant advantage in tasks requiring high accuracy.

The differential drive system provides the necessary control and flexibility for our mobile robot to efficiently navigate various environments and perform tasks. Along with a robust and efficient power distribution, precise sensor integration, and reliable motor control, our robot is capable of operating independently thanks to the use of advanced components and carefully planned connection strategies.

4. DESIGN

4.1 Overall System View

Our robot can be classified as a 4 wheeled multi-purpose autonomous mobile robot. Its intended use is mainly for exploration of unknown places and monitoring the environment surrounding itself. Our mobile vehicle is equipped with sensors such as the YDLidar X2 for precise mapping and obstacle detection, HC-SR04 Ultrasonic Sensors for short-range obstacle avoidance, and the ESP-32 CAM for capturing image data. Controlled by an Arduino Uno serving as the central processing unit alongside the computational power of a Jetson Nano, the system is meticulously integrated to ensure efficient sensor data processing and motor control. Power management is facilitated by a combination of 9V batteries for sensor operations and a 15V LiPo battery for motor functionality. Together, these components enable the mobile robot to navigate autonomously, avoid obstacles, and map its surroundings accurately.

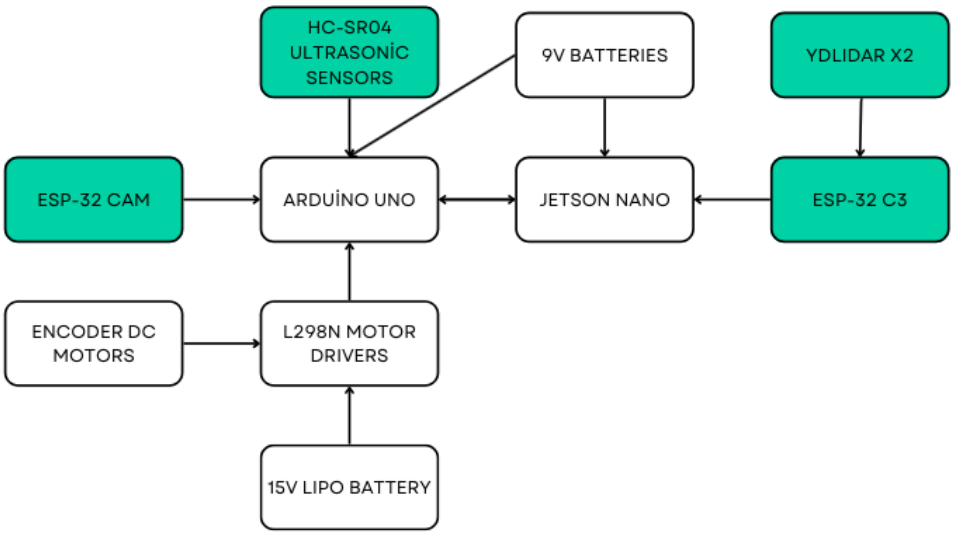
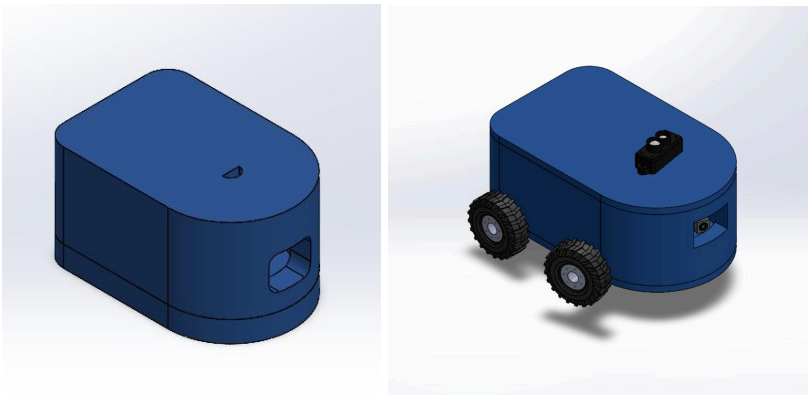


Figure 4.1.1 : Overall system diagram

4.1.1 Mechanical Design

The mechanical design of our autonomous mobile robot, created using SolidWorks software, is shown in Figure 4.1.1.1. Detailed technical drawings of the robot's complementary upper

and lower body structures are presented in Figure 4.1.1.2 and Figure 4.1.1.3.



a) The main body **b) The whole assembly**

Figure 4.1.1.1: Mechanical design

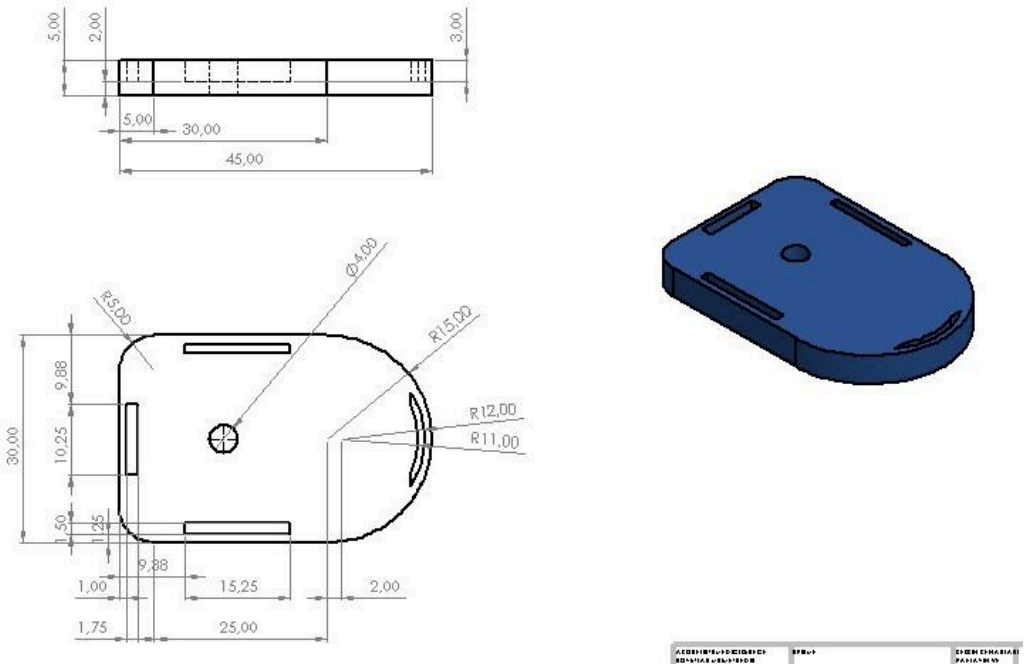


Figure 4.1.1.2: Technical drawing of lower body

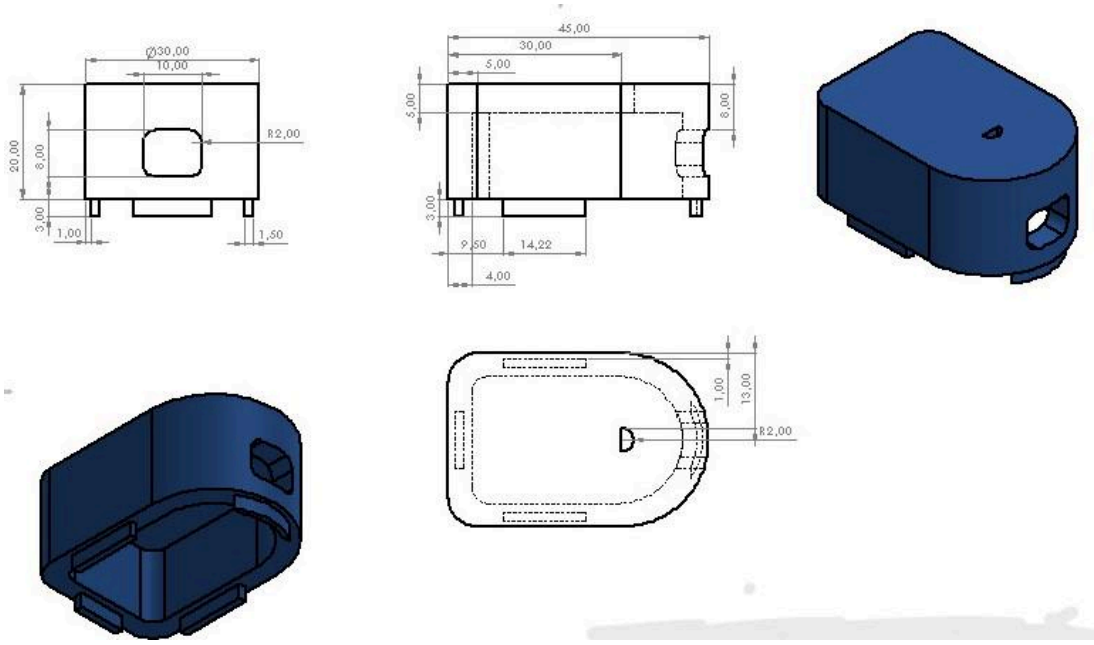


Figure 4.1.1.3: Technical drawing of upper body

4.1.2 Algorithm Architecture

Given below in Figure 4.1.2.1 is the schematic of the software architecture for the visual navigation system we intended to use.

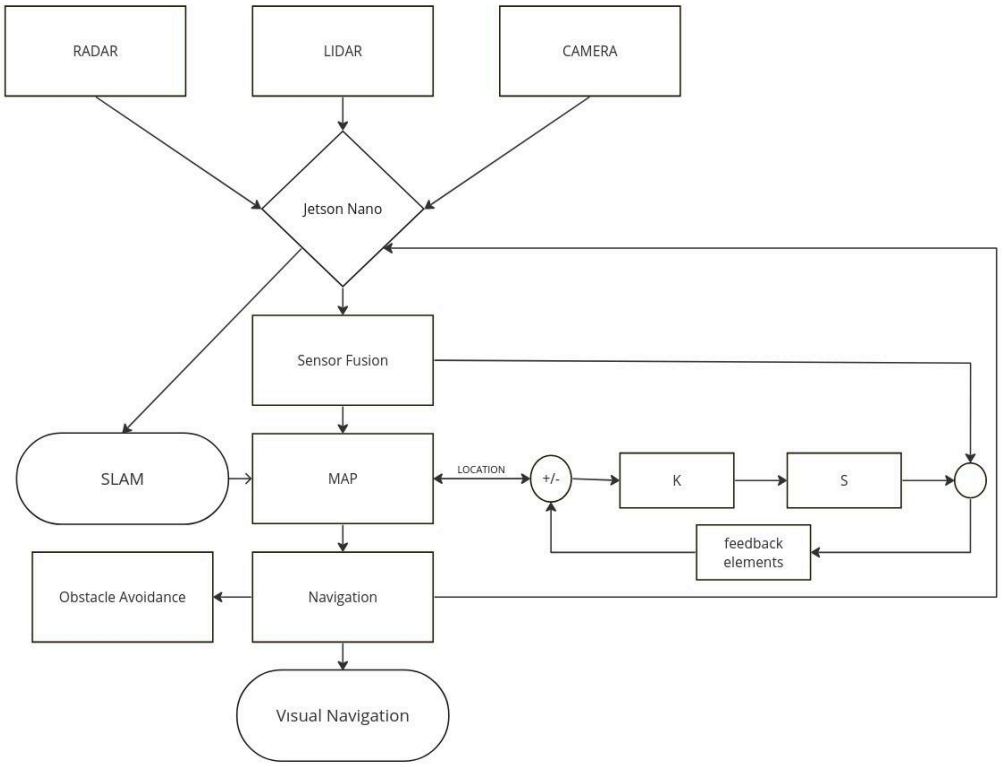


Figure 4.1.2.1: Visual Navigation Schematic

4.1.3 Electrical Architecture

The electrical architecture of our autonomous mobile robot is designed to ensure efficient power distribution, precise sensor integration, and reliable motor control. The electrical design flow is represented in Figure 4.1.3.1, and the system has undergone necessary simulation tests followed by real-world testing.

Components: The components integrated into the electrical system of our mobile robot are as follows:

- **YDLidar X2:** This lidar sensor provides high-precision distance measurement and obstacle detection. It performs 360-degree scanning to create a detailed map of the mobile robot's surroundings. The YDLidar X2 is connected to the Jetson Nano via USB through the ESP-32 C3 and provides real-time data.
- **HC-SR04 Ultrasonic Sensors:** These sensors measure distance using sound waves and are effective for obstacle detection. Each sensor can measure distances from 2 cm

to 4 meters with ±3 mm accuracy. The HC-SR04 sensors enhance the robot's ability to move safely and are connected to the Arduino Uno.

- **ESP-32 CAM:** This high-resolution camera module enhances the mobile robot's vision capabilities. It offers 2 MP resolution and a 66.5-degree wide field of view. With built-in Wi-Fi and Bluetooth features, it allows for wireless communication and is integrated with the Arduino Uno.
- **Arduino Uno:** Serving as the main microcontroller, this board processes sensor data and manages motor control signals. It is connected to the HC-SR04 sensors, ESP-32 CAM, and L298N motor drivers.
- **Jetson Nano:** This powerful microprocessor board handles image processing, machine learning, and advanced computing tasks. It processes data from the YDLidar X2 and improves the robot's navigation and obstacle avoidance capabilities. It communicates with the ESP-32 C3 via USB and directly with the Arduino Uno.
- **L298N Motor Drivers (2 Units):** These drivers are used to control the DC motors, receiving signals from the Arduino Uno to manage motor speed and direction. Each driver controls a pair of motors and provides reliable motor performance with high current capacity.
- **MP5BFN 12V Encoder DC Motors (2 Units):** Designed for high-speed operation without a gearbox, these motors run at 1500 RPM at 12V DC. They have built-in encoders that provide precise position and speed feedback, ensuring the robot moves quickly and accurately.
- **350 RPM 12V Encoder DC Motors (2 Units):** These motors feature a gearbox for high torque and operate at a lower speed. They run at 350 RPM at 12V DC, providing 2.0 kg.cm torque. The encoders offer precise control and feedback, making these motors ideal for low-speed, high-torque applications.
- **2 x 9V Batteries:** These batteries power the Arduino Uno and sensors. Each battery provides 9V independently, allowing for long-term operation.
- **1 x 15V LiPo Battery:** This battery powers the motors. Its high capacity ensures strong and continuous energy supply, supporting the mobile robot's high-performance

operation.

- **ESP-32 C3:** This module connects the YDLidar X2 to the Jetson Nano via USB, facilitating efficient data transmission for effective lidar use.
- **Breadboard:** Used for creating reliable connections between various components, the breadboard allows easy integration and modification when necessary.

Power Distribution: The power distribution for the mobile robot is meticulously planned to ensure that all components receive the required voltage and current. The 15V LiPo battery powers the L298N motor drivers, which in turn control the MP5BFN and 350 RPM motors. The 9V batteries power the Arduino Uno and sensors, including the HC-SR04 and ESP-32 CAM.

Connections and Integration: The integration of components is managed through direct wiring and the use of a breadboard. The YDLidar X2 connects to the Jetson Nano via USB through the ESP-32 C3. The HC-SR04 sensors provide additional obstacle detection data to the Arduino Uno. The ESP-32 CAM integrates with the Arduino Uno for real-time image capture and processing. The L298N motor drivers receive control signals from the Arduino Uno based on sensor inputs and predefined algorithms to control the MP5BFN and 350 RPM motors. All connections are consolidated on a breadboard for ease of modification and reliable connectivity.

Communication and Control: Communication between components is crucial for the operation of the mobile robot. The Arduino Uno serves as a central hub, collecting data from the HC-SR04 sensors and ESP-32 CAM, and sending control signals to the L298N motor drivers. The Jetson Nano handles high-level tasks such as image recognition and advanced navigation algorithms, communicating with the Arduino Uno for coordinated control actions. The ESP-32 C3 transmits precise distance measurements from the YDLidar X2 to the Jetson Nano, enhancing the robot's obstacle detection and avoidance capabilities.

The electrical architecture of our autonomous mobile robot is designed to provide robust and efficient power distribution, precise sensor integration, and reliable motor control. The use of advanced components and careful connection strategies ensures that our mobile robot can navigate and perform tasks independently in various environments.

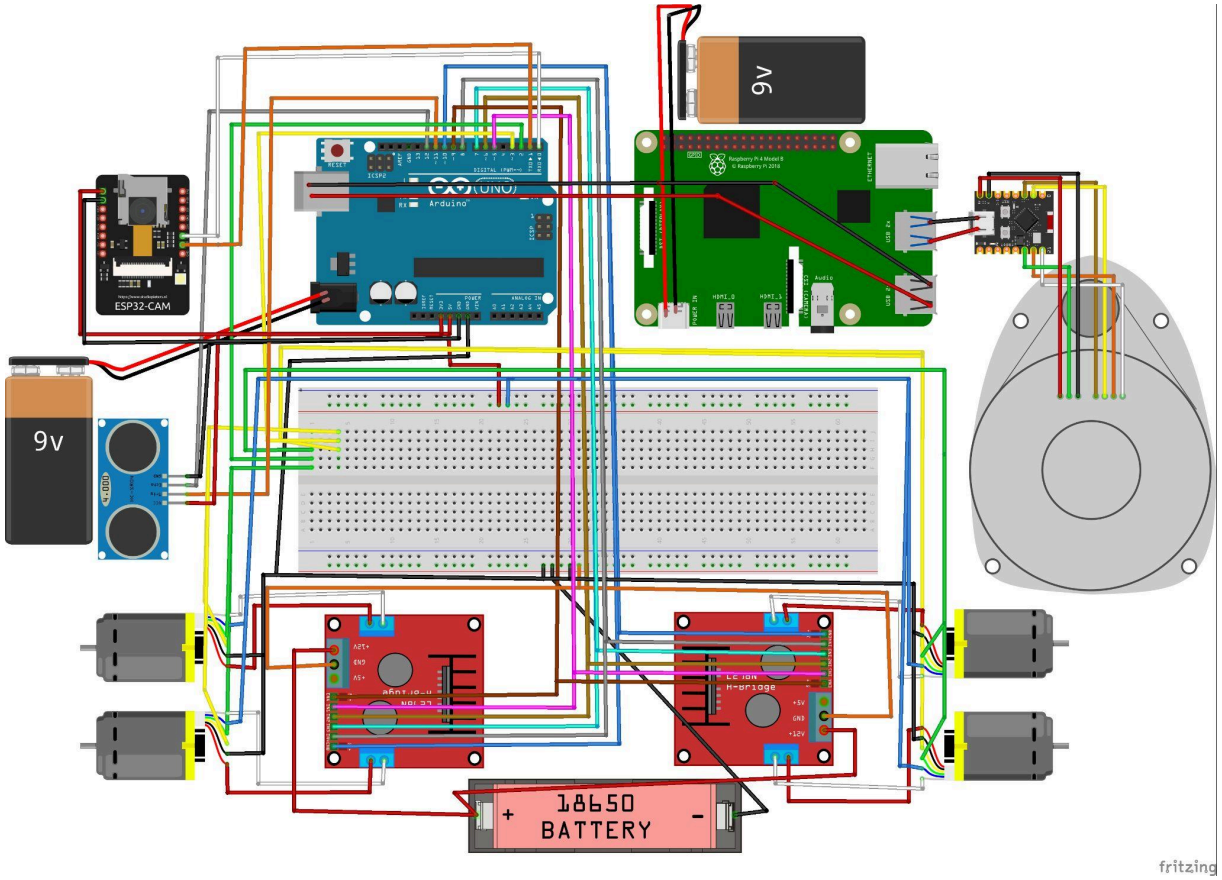


Figure 4.1.3.1: Electrical Design

5. DESIGN JUSTIFICATION

5.1 Test Plans

5.1.1 Mobile Robot

Figure 5.1.1.1 represents the state of the autonomous mobile robot's body after printing and before assembly. The upper and lower bodies are designed to close together as a single unit at the necessary connection points.

a) Mobile robot's body after printing **b) Pre assembly**

Figure 5.1.1.1 : Printed body parts

Figure 5.1.1.2 shows the trials of the electronic integration layout of the mobile vehicle. Sensor integration was completed, and the electronic circuits were installed inside the vehicle. After the mechanical assembly was finished, real-life testing began.

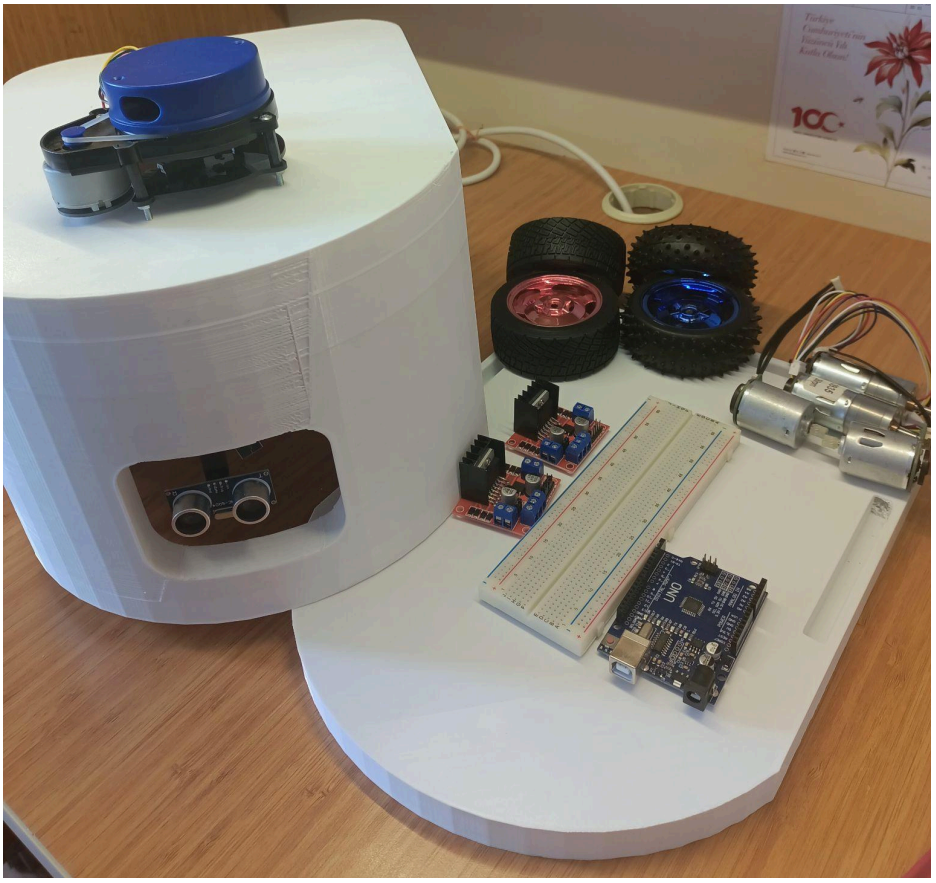


Figure 5.1.1.2 : Before the electronic integration

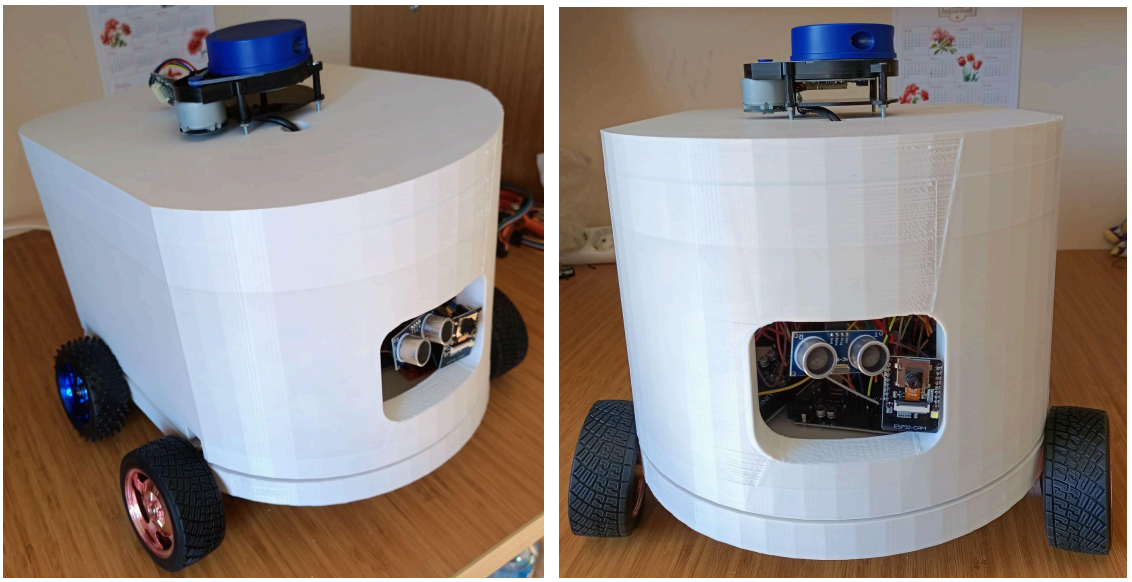


Figure 5.1.1.3 : After the electronic and mechanic integration

5.1.2 PID Control

We conducted extensive testing using one motor at a time to evaluate the performance of our PID control system. In our logic, the back and front wheels are controlled by the same output, a decision influenced by considerations of battery usage and efficiency. By synchronizing the control of both sets of wheels, we aimed to achieve more consistent and balanced movement, while also optimizing the power consumption of our system.

Figure 5.1.2.1 and 5.1.2.2 illustrates our testing setup, where the motor was mounted and connected to the control circuitry. This setup allowed us to simulate real-world conditions and gather comprehensive data on the system's behavior. The motors were equipped with encoders to provide precise feedback on their rotational speed and position, which is crucial for the accurate functioning of the PID controller.

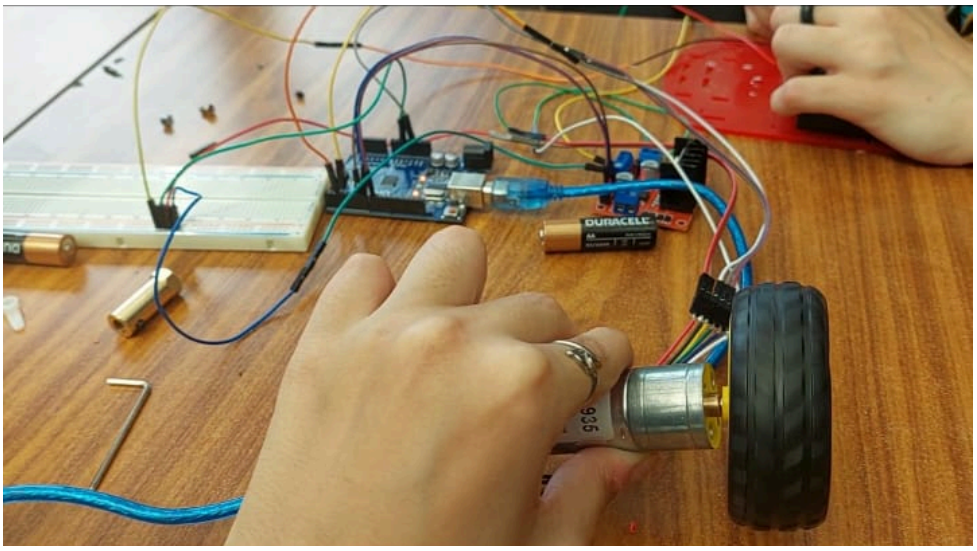


Figure 5.1.2.1 : Motor Testing Setup

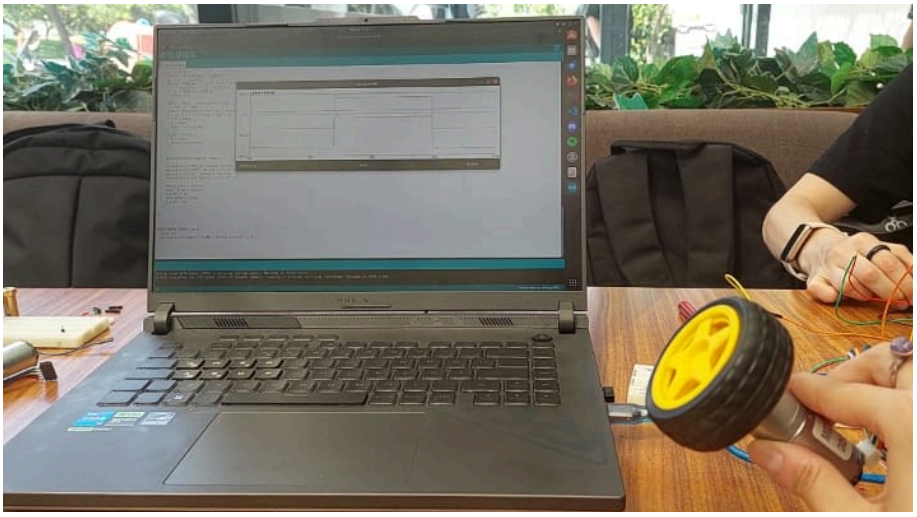


Figure 5.1.2.2 : Motor PID Control Testing

Our goal was to minimize the error between the desired and actual speeds of the motors, ensuring smooth and responsive control. The testing process involved running the motors at various speeds and load conditions, observing how well the PID controller maintained the target speed.

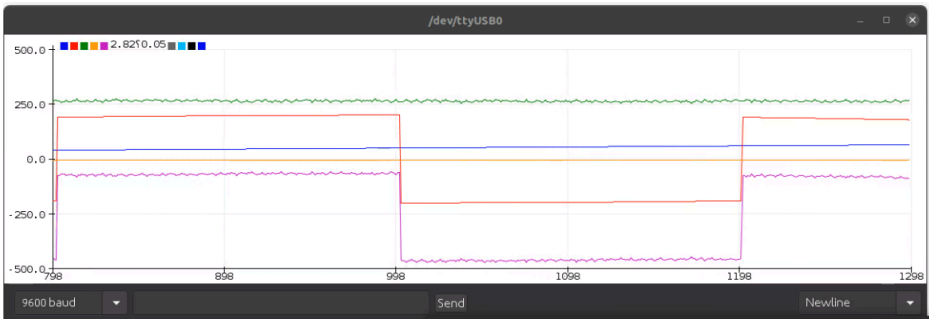


Figure 5.1.2.3 : PID Control Results

Figure 5.1.2.3 presents the results of our PID testing. The graphs depict the desired versus actual motor speeds over time, highlighting the effectiveness of the PID adjustments. Initially, we observed some oscillations and deviations from the target speed, which were gradually reduced as we refined the PID parameters. The final results demonstrated a significant improvement in speed regulation, with minimal overshoot and steady-state error.

5.1.3 LIDAR SLAM

Our SLAM application aims to simultaneously map the environment and track the position of

the mobile robot. The YDLidar X2 provides essential environmental data for the SLAM process with its high-precision distance measurement and 360-degree scanning capability.

The YDLidar X2 sensor is mounted on the mobile robot, continuously scanning the surroundings and collecting data processed by the Hector SLAM algorithm. This algorithm is effective for mapping in dynamic environments with low drift and does not require odometry information, making it a suitable choice for our project.

We conducted our mapping tests in various environments. For indoor tests, we used a dormitory room and a dormitory corridor. Figure 5.1.3.1 shows the mapping of a dormitory room, and Figure 5.1.3.2 represents the mapping of a dormitory 7th floor corridor. Additionally, we conducted outdoor tests on a garden walkway. Figure 5.1.3.3 depicts the mapping of the walkway. These environments were selected to test the SLAM system's performance in both confined indoor spaces and larger, open outdoor areas.

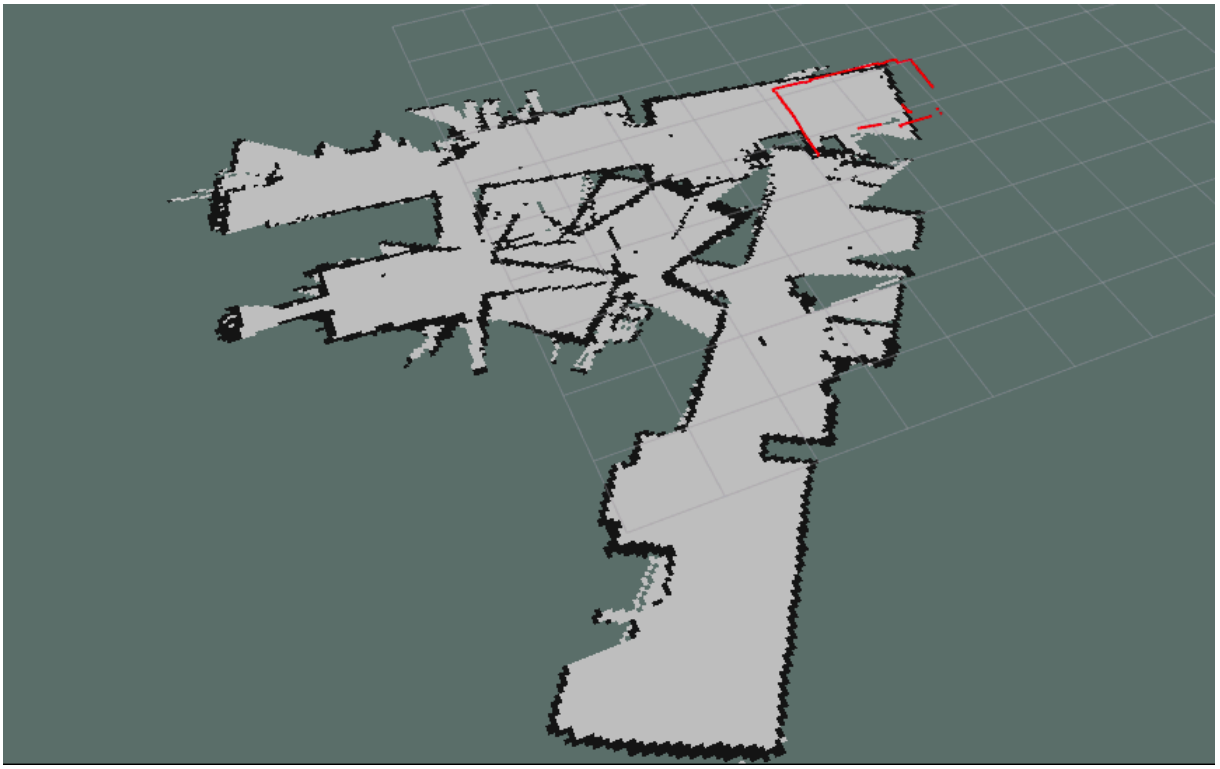


Figure 5.1.3.1 : Mapping of a dorm room

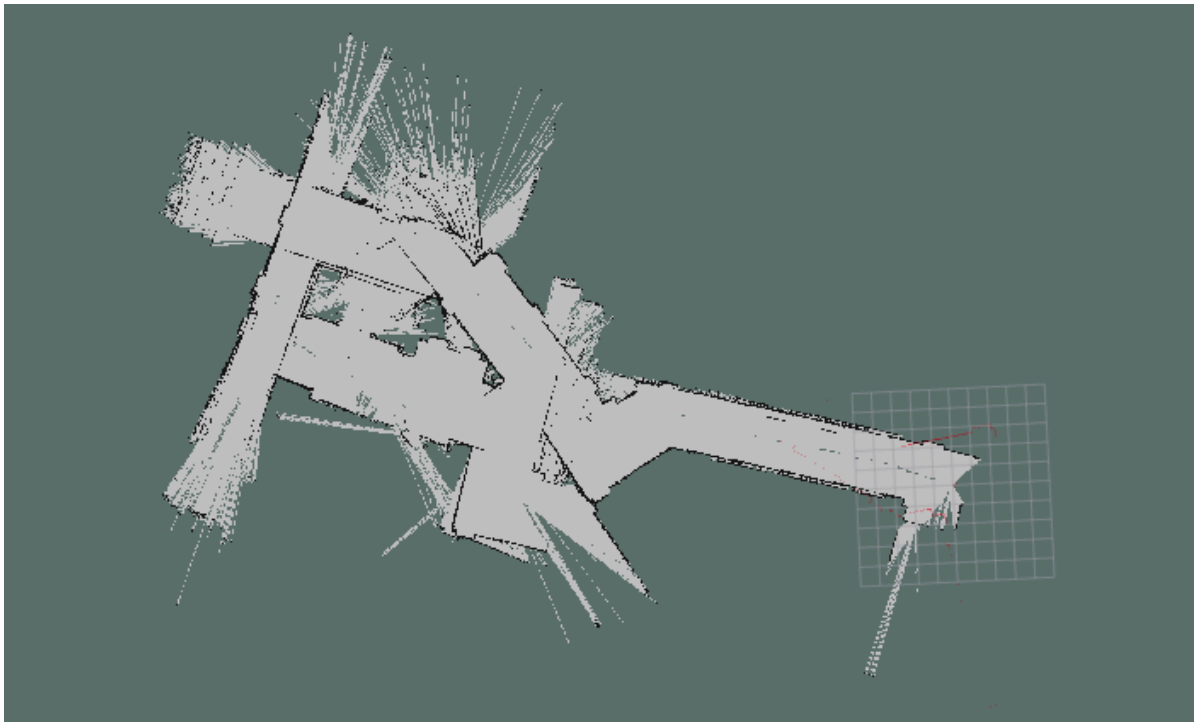


Figure 5.1.3.2 : Mapping of a dorm 7th floor corridor

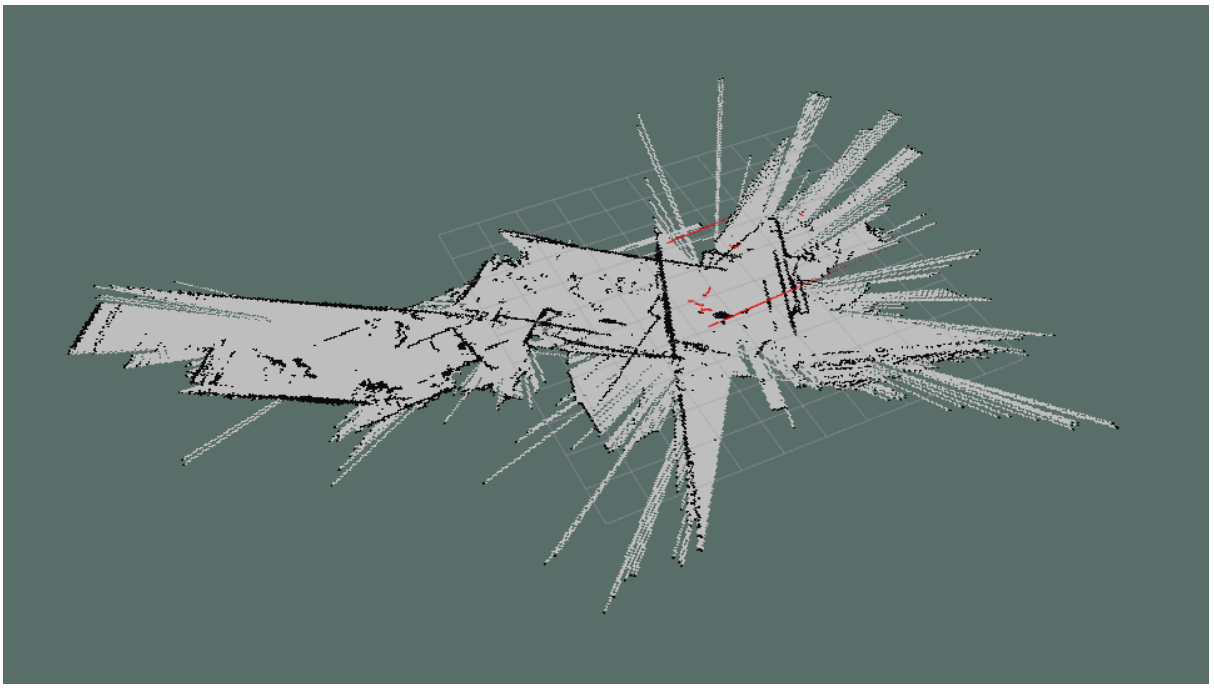


Figure 5.1.3.3 : Mapping of the walkway

5.1.4 Obstacle Avoidance

Our test setup for the obstacle system is shown in Figure 5.1.4.1. We did the tests using the right and left wheels to represent the whole system. Since we use the same outputs from Arduino for front and back wheels, the battery was preserved this way.

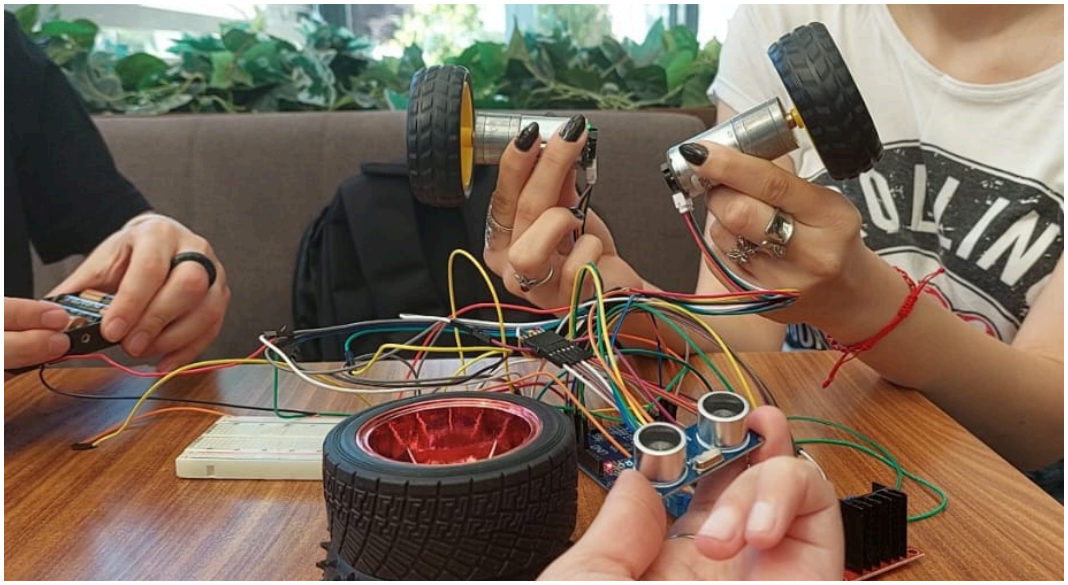


Figure 5.1.4.1: Obstacle Avoidance Testing Setup

The motors moved forward when there was no object to obstruct the ultrasonic sensor's view as we expected. When we move the ultrasonic sensor to the point of the wheels and obstruct its view, the motors stop moving and move backwards then randomly take a left or right turn. If the sensor's view was obstructed on one side, one of the wheels stopped moving and the other one continued which results in a turning movement.

5.2 Design Justification Via Simulations

5.2.1 Mechanic Tests Ansys

This section of the project report presents the results and evaluations of the static force analysis applied to the body of an autonomous mobile robot designed for search and exploration under challenging conditions. The finite element analysis (FEA) was conducted using ANSYS software. The purpose of this simulation was to evaluate the behavior of the robot's body under various forces that could occur from falling objects.

Analysis Parameters: The analysis scenarios were simulated considering the environments in which the robot might operate. These scenarios were created based on estimated weights and drop heights of objects that might fall onto the body. The force ranges used in the simulations are as follows:

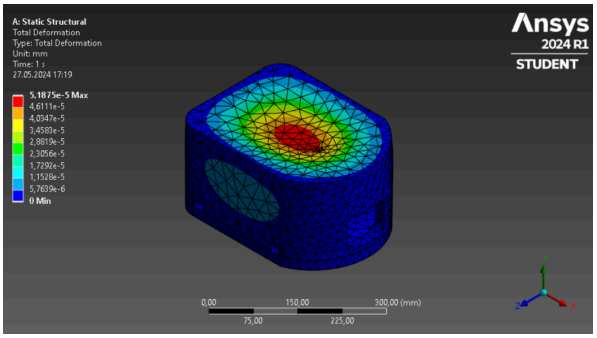
- Brick: 20-80 N
- Wood Beam: 50-200 N
- Door Beam: 130-350 N
- Stone: 40-150 N
- Tree Branch: 30-120 N

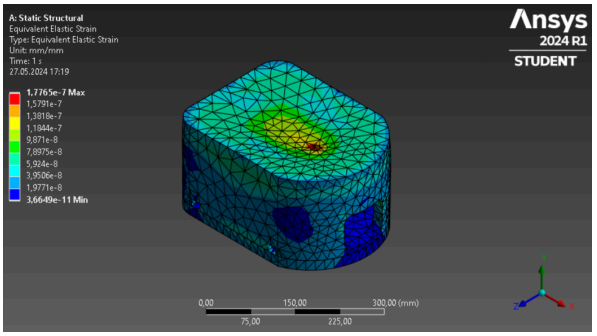
The static force analysis using ANSYS software involved several steps. First, a 3D model of the robot body was created using SolidWorks and imported into ANSYS. Then, the material properties of the body were defined. Following this, the model was meshed using the finite element method. After meshing, the surfaces of the body in contact with the ground were fixed, and the defined force ranges were applied to the model. Finally, a static analysis was performed for these force ranges, and the results were evaluated.

Results and Evaluation: The analysis results include the total deformation and von Mises stress distribution under the applied forces. The main findings are:

- **Deformation:** the total deformation values under minimum and maximum applied forces were determined, with the highest deformation typically occurring around the points of force application.
- **Von Mises Strain:** The strain analysis calculated the von Mises stress distribution, identifying critical regions. The material properties remained within acceptable limits, maintaining structural integrity.

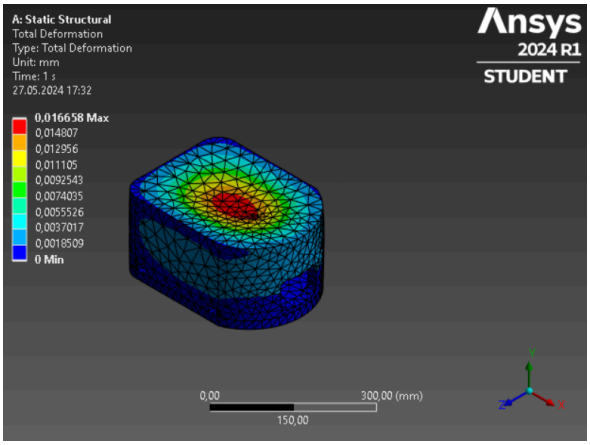
The results show that the robot body can operate safely under the specified force ranges. Even under the highest expected forces, the body exhibited acceptable levels of deformation and stress. This indicates that the body has sufficient structural integrity to perform under challenging conditions.

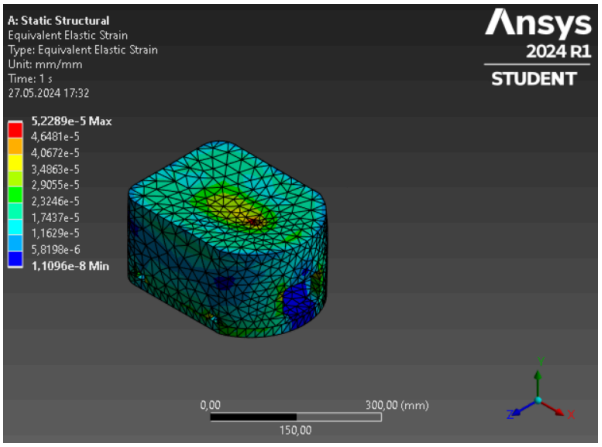




a)Total Deformation **b)Total strain**

Figure 5.2.1.1 : Total deformation and total strain under 65N





a)Total Deformation **b)Total strain**

Figure 5.2.1.2 : Total deformation and total strain under 350N

5.2.2 CARLA Simulator

The CARLA simulator is an open-source simulator used to develop, train, and test autonomous driving systems. It includes various elements such as city environments, different weather conditions, and traffic scenarios with varying densities to simulate real-world scenarios, recreating the complexities of driving in a simulation environment. The environment shown for testing is shown in Figure 5.2.2.1.

Figure 5.2.2.1 : Carla Enviroment

CARLA is a key tool for engineers and researchers in developing and refining autonomous driving systems safely. It allows diligent testing without real-world risks, ensuring the safety of both vehicles and people. This controlled space is crucial for improving algorithms and models, enabling consistent replication of scenarios to evaluate different software variations' responses. This repeatability is vital for achieving robust performance across diverse scenarios. CARLA generates extensive datasets covering various driving scenarios, crucial for training machine learning models and creating sophisticated algorithms for perception, decision-making, and control in autonomous driving. Figure 5.2.2.2 showcases this data generation capability.

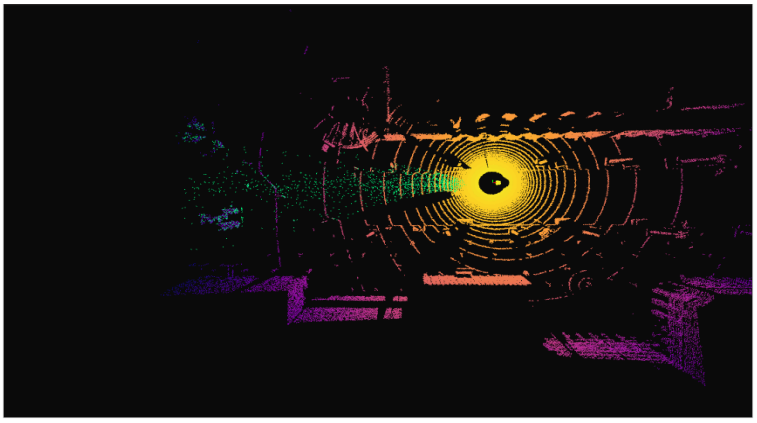


Figure 5.2.2.2 : Lidar and Radar data obtained Carla Environment

In summary, CARLA's flexibility, realism, and diverse capabilities make it a crucial tool for advancing autonomous vehicle tech while minimizing real-world testing risks.

5.2.3 ROS

ROS (Robot Operating System), functioning as software for robot control, facilitates communication between robots and humans. This open-source middleware enables processing data received from the external world through sensors, allowing the robot to send back commands based on this information. Developed under the BSD license, ROS is an open-source operating system. A robot managed by ROS can process data collected from the environment, such as sound and images, through a computer. Processed data, depending on computer algorithms, can be returned with commands suitable for the robot's position.

Being open-source allows ROS to easily communicate and share information with many individuals worldwide. Consequently, projects can be made usable by others in the future. The debugging mechanism facilitates the easy handling of errors. When an error occurs, the robot doesn't stop entirely; only the node in the error field stops.

Specifically, in the project we used ROS Melodic due to its compatibility with the Jetson Nano and its robust navigation stack. This version of ROS provides the necessary tools and libraries that are optimized for the navigation and control needs.

5.2.4 Gazebo/RViz Simulation

Gazebo, alongside the RViz application plays a crucial role in simulating robotic systems.Gazebo is a powerful open-source robot simulation tool that allows for the accurate modeling of robots in a three-dimensional environment. It enables the testing and development of robot models in a controlled, virtual setting before deploying them in the real world.

RViz, on the other hand, is a 3D visualization tool for ROS that showcases the visualization of sensor data and state information from the robot. Together, Gazebo and RViz provide a comprehensive simulation environment for robotics development.

In our project, we used Gazebo to model our robot's differential drive system. This allowed us to accurately simulate the robot's movement and interactions with its environment. Through

RViz, we were able to visualize the data from the robot's LiDAR sensors, including the results of LiDAR-based SLAM (Simultaneous Localization and Mapping). This integration enabled us to monitor the robot's perception and mapping in real-time, ensuring that the robot's navigation and localization algorithms were functioning correctly.

5.2.5 Differential Drive System

We tested the differential drive system using ROS and Gazebo simulation. Firstly we modeled a simple four wheel robot to represent our design using URDF files using necessary coefficients. The robot has a rectangular body with four wheels, defined using the Xacro XML macro language, which simplifies parameter reuse and simplifies complex URDF files. The body and wheels have specified dimensions, mass, and inertial properties, calculated using given densities and geometric parameters. The robot features a camera and a 2D LiDAR sensor mounted on the body, each with its own visual, collision, and inertial properties. In Figure the simulation state is provided where we simulate the robots' motion.

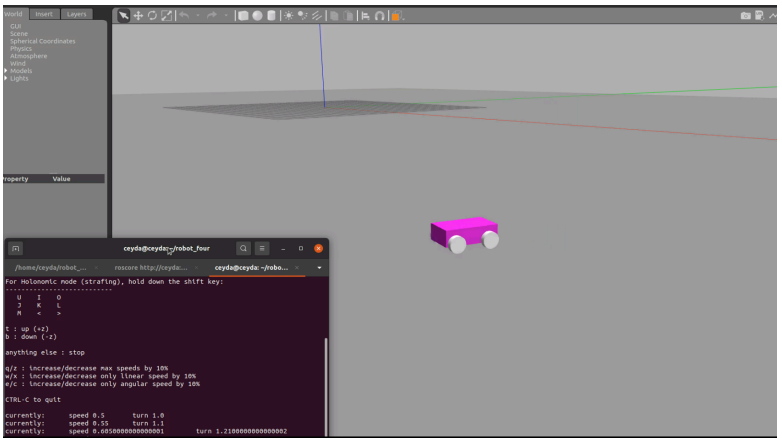


Figure 5.2.5.1.: Differential Drive System Simulation

The differential drive system is implemented through four continuously rotating wheel joints (wheel1_joint to wheel4_joint) attached to the robot's body. This drive system allows the robot to move forward, backward, and turn by varying the speeds of the left and right wheels independently. The Gazebo configuration section specifies the materials, friction coefficients, and integrates sensors with their respective plugins for simulation purposes. The skid_steer_drive_controller plugin controls the differential drive mechanism, handling wheel separation and diameter parameters, and mapping ROS topics for velocity commands

(cmd_vel) and odometry data (odom). The diagram in Figure showcases all the functionalities of the system.

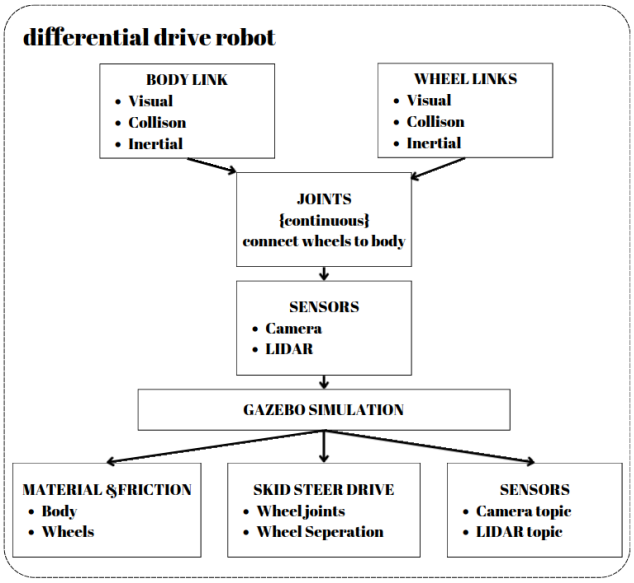


Figure 5.2.5.2. : Differential Drive Robot System

In summary, this setup enables simulation of a differential drive robot with realistic physical properties and sensor feedback in the Gazebo environment, therefore allowing us for testing and development of navigation and control algorithms.

5.2.6 LIDAR SLAM

Hector SLAM is a specific type of 2D LiDAR SLAM algorithm known for its high accuracy and efficiency in mapping environments. It is particularly effective for applications where real-time performance and robustness are critical, such as in rescue robots or autonomous vehicles aligning with the focus of our subject. Hector SLAM uses a detailed map of the environment created from 2D LiDAR sensors while simultaneously localizing the sensor within that map. The LiDAR sensor emits laser beams in multiple directions, measuring the time taken for these beams to bounce back after hitting objects in the environment. By analyzing these return times, the sensor constructs a precise 2D representation of its surroundings on a grid map. Since the algorithm does not require odometry or IMU data, it is highly suitable for our project. The general logic flow of the algorithm is represented by the Figure 5.2.6.1.

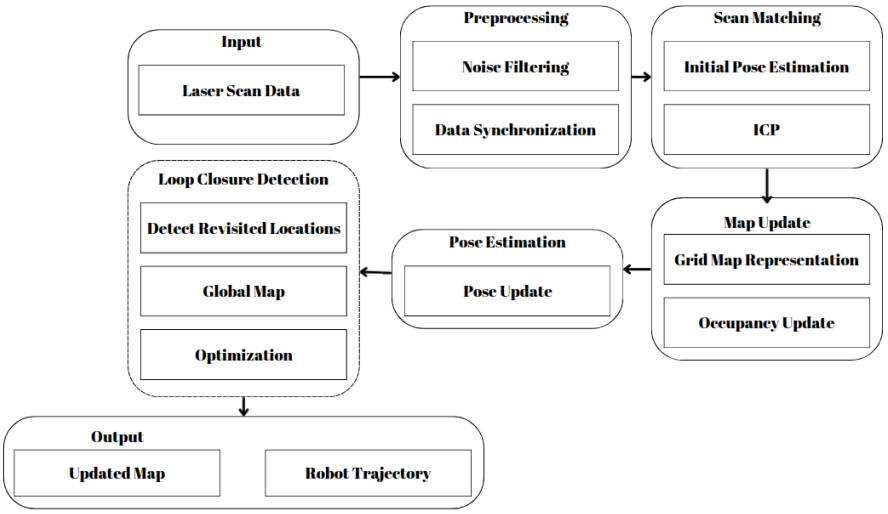


Figure 5.2.6.1 :SLAM Algorithm Flow

We first tested the algorithm using the differential robot model that we built earlier on Gazebo simulator. Later on we first made sure to correctly gather LIDAR data from our sensor and adjust the LIDAR specification accurately within the code. The LIDAR data is given in Figure 5.2.6.2 .

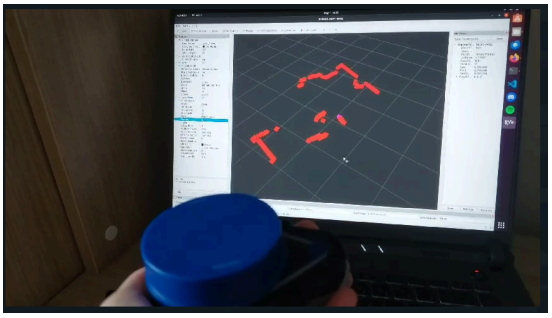


Figure 5.2.6.2: LIDAR Data

We have conducted a few tests of mapping capabilities using LIDAR and a computer to make sure our process works which result in Figure 5.2.6.3. And at last we did the necessary steps to prepare our Jetson Nano to be ready for real life experiments.

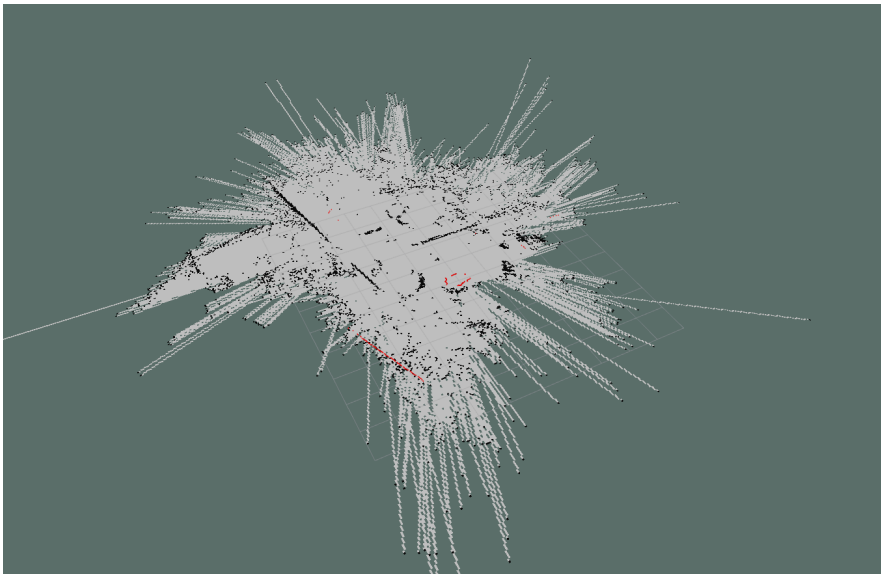


Figure 5.2.6.3. : Open Field Test

5.2.7 PID Control

For PID control we used an arduino code that implements a PID control system to manage the speed of a motor. It uses an encoder to read the motor's position, with interrupt service routines to update the encoder count. The main control loop calculates the motor's speed (RPM) and compares it to a desired speed (RPM_d) that follows a sinusoidal pattern, adjusting over time. A PID controller computes the control voltage needed to minimize the difference between the desired and actual RPM. The control voltage is then applied to the motor driver to regulate its speed. The system continuously updates and adjusts the motor's speed, printing relevant data to the serial monitor for monitoring and debugging. Additionally, a hardware timer ensures the control loop runs at consistent intervals. The main loop decision tree for the code is given in Figure 5.2.7.1.

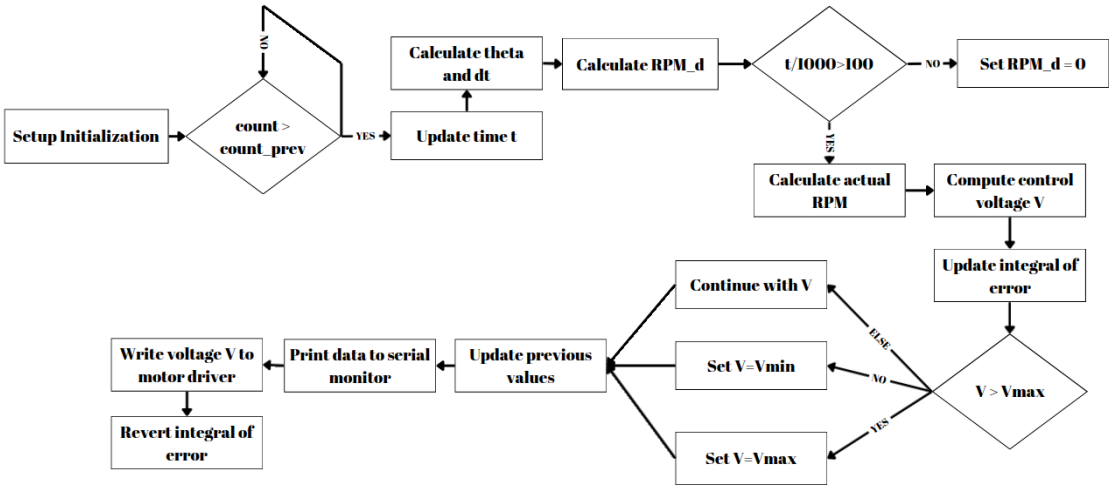


Figure 5.2.7.1: Main loop Decision Tree

5.2.8 Obstacle Avoidance

The obstacle avoidance is an important part for our robot since we want the robot to be able to navigate in unknown environments. We utilized ultrasonic sensor to achieve reliable results. Before testing the main algorithm for obstacle avoidance, we first verified our sensor is working and provides accurate data. For verification we used a simple code that tests an ultrasonic distance sensor by emitting high-frequency sound pulses and measuring the time it takes for the pulses to return, calculating the distance to an obstacle, and then printing the distance in inches to the serial monitor every 100 milliseconds.

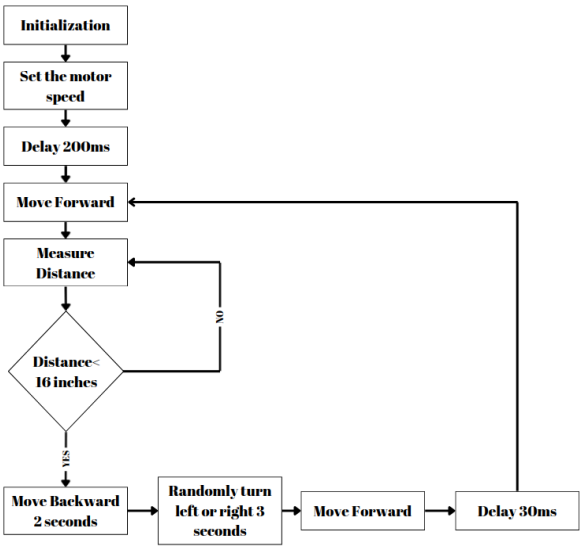


Figure 5.2.8.1: Flow Diagram of Obstacle Avoidance

Obstacle avoidance algorithm is designed for a robot to autonomously navigate its environment by avoiding obstacles using an ultrasonic sensor. In Figure 5.2.8.1 the main loop logic is given as a diagram. TO summarize shortly, the algorithm initializes the robot's motor control pins and sets an initial speed for its motors. It also seeds the random number generator to ensure variability in the robot's movement. The main loop of the algorithm continuously measures the distance to potential obstacles using an ultrasonic sensor. If an obstacle is detected within 16 inches, the robot moves backward for two seconds to create space. Then, it randomly decides to turn left or right for three seconds to avoid the obstacle before resuming forward motion. The doPing() function, which measures and averages the distance using ultrasonic pulses, ensures accurate detection of obstacles. Motor control functions (go_forward(), go_backwards(), go_right(), go_left(), stop_all()) manage the robot's movements based on the distance data. This way our robot can navigate autonomously by making real-time decisions to avoid collisions, and explore the unknown environment freely.

6. TIMELINE

The project's timeline and task distribution is provided in the Figure 6.1 below.

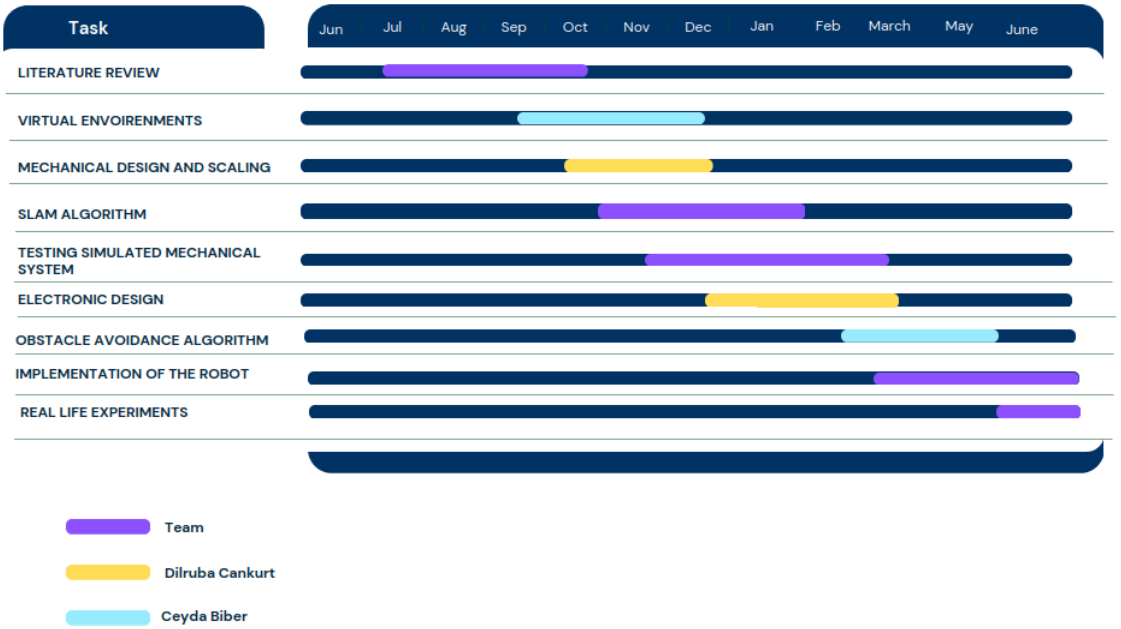


Figure 6.1 : Timetable

7. BUDGET

The budgeting process for the project includes the compilation of a list detailing materials required and the submission of applications for external funding.

7.1 Component List

The list of materials required for the project, also the services that have been utilized and their prices are shown in the table 7.1.1.

Table 7.1.1. Component prices

Component/Service	**Model**	**Cost**
Camera Sensor	ESP32-CAM	311,46 ₺
LIDAR Sensor	YDLidar X2	3007.87 ₺
Jetson Nano	Jetson Nano	10.007, 74 ₺
DC Motor	MP5BFN 12V Encoder DC Motor	561,12 ₺ (x2)
DC Motor Driver	LN298N	65,87 ₺ (x2)
3D Printing	-	4.994 ₺
Ultrasonic Sensor	HC-SR04	33,97 ₺
Wheels	85 x 32	340,68 ₺ (x2)
Shipping fees	-	200 ₺
Servo Motor	Tower Pro SG90 RC Mini	70,14 ₺
	Total Cost:	20.560,52 ₺

7.2 Service Procurements

In this project 3D printing services were used.

7.3 Application for External Support

An application has been made to the TUBITAK 2209-B program for project budget support, and it got approved for 12000 TL.

REFERENCES

[1] Hong, Z. *et al.* (2021) *Radar SLAM: A Robust SLAM System for All Weather Conditions* [Preprint]. doi: https://doi.org/10.48550/arXiv.2104.05347.

[2] Huynh-Thu, Q. and Ghanbari, M. (2008) 'Scope of validity of PSNR in Image/Video Quality Assessment', Electronics Letters, 44(13), pp. 800–801. doi:10.1049/el:20080522.

[3] Wang, Z. *et al.* (2004) 'Image quality assessment: From error visibility to structural similarity', *IEEE Transactions on Image Processing*, 13(4), pp. 600–612. doi:10.1109/tip.2003.819861.

[4] Yang, W. *et al.* (2017) 'Deep joint rain detection and removal from a single image', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1357–1366. doi:10.1109/cvpr.2017.183.

[5] Fu, X. *et al.* (2021) 'Rain streak removal via dual graph convolutional network', *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(2), pp. 1352–1360. doi:10.1609/aaai.v35i2.16224.

[6] Yi, Q. *et al.* (2021) 'Structure-preserving deraining with residue channel prior guidance', *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4238–4247. doi:10.1109/iccv48922.2021.00420.

[7] Zamir, S.W. *et al.* (2022) 'Restormer: Efficient Transformer for high-resolution image restoration', *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5728–5739. doi:10.1109/cvpr52688.2022.00564.

[8] Chen, X. *et al.* (2023) 'Learning a sparse transformer network for effective image deraining', *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* [Preprint]. doi:10.1109/cvpr52729.2023.00571.

[9] Cadena, C. *et al.* (2016) 'Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age', *IEEE Transactions on Robotics*, 32(6), pp. 1309–1332. doi:10.1109/tro.2016.2624754.

[10] Debeunne, C. and Vivet, D. (2020) 'A review of visual-lidar fusion based simultaneous localization and mapping', *Sensors*, 20(7), p. 2068. doi:10.3390/s20072068.

[11] Mielle, M., Magnusson, M. and Lilienthal, A.J. (2019) 'A comparative analysis of radar and Lidar Sensing for localization and mapping', *2019 European Conference on Mobile Robots (ECMR)* [Preprint]. doi:10.1109/ecmr.2019.8870345.

[12] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," in IEEE International Conference on Robotics and Automation (ICRA). Paris: IEEE, 2020, pp. 6433–6438.

[13] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," The International Journal of Robotics Research (IJRR), vol. 36, no. 1, pp. 3–15, 2017.

[14] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "Mulran:Multimodal range dataset for urban place recognition," in IEEE International Conference on Robotics and Automation (ICRA).IEEE, 2020, pp. 6246–6253.

[15] Sheeny, M. et al. (2021) 'Radiate: A radar dataset for automotive perception in Bad weather', 2021 IEEE International Conference on Robotics and Automation (ICRA) [Preprint]. doi:10.1109/icra48506.2021.9562089.

[16] R. Mur-Artal and J. D. Tard´os, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255–1262, 2017.

[17] Hong, Z., Petillot, Y. and Wang, S. (2020) 'RadarSLAM: Radar based large-scale slam in all Weathers', 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [Preprint]. doi:10.1109/iros45743.2020.9341287.
[18] Behley, J. and Stachniss, C. (2018) 'Efficient surfel-based slam using 3D laser range data in urban environments', Robotics: Science and Systems XIV [Preprint]. doi:10.15607/rss.2018.xiv.016.
[19] Jie Xiao, Xueyang Fu, Aiping Liu, Feng Wu, and Zheng-Jun Zha. Image de-raining transformer. IEEE TPAMI, 2022
[20] Wang, Z. *et al.* (2022) 'Uformer: A general U-shaped transformer for image restoration', *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17683–17693. doi:10.1109/cvpr52688.2022.01716.
[21] Xiao, J. *et al.* (2023) 'Image de-raining transformer', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11), pp. 12978–12995. doi:10.1109/tpami.2022.3183612.
[22] Yang, W. *et al.* (2017) 'Deep joint rain detection and removal from a single image', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1357–1366. doi:10.1109/cvpr.2017.183.
[23] Zhang, H. and Patel, V.M. (2018) 'Density-aware single image de-raining using a Multi-stream dense network', *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 695–704. doi:10.1109/cvpr.2018.00079.
[24] Fu, X. *et al.* (2017) 'Removing rain from single images via a deep detail network', *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3855–3863. doi:10.1109/cvpr.2017.186.
[25] Wang, T. *et al.* (2019) 'Spatial attentive single-image deraining with a high quality real rain dataset', *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12270–12279. doi:10.1109/cvpr.2019.01255
[26] YİĞİT, E., ONER, A.E. and YÖNTEM, O. (2020) 'Otonom Araçların Otomotiv sektörüne etkileri ve Beraberinde Getirdiği Yenilikler', *European Journal of Science and Technology* [Preprint]. doi:10.31590/ejosat.820722.
[27] *Autonomous cars: Levels of autonomous driving explained* (2022) *AckoDrive*. Available at: https://ackodrive.com/car-guide/autonomous-cars-and-levels-of-autonomous-driving/ (Accessed: 07 January 2024).