YILDIZ TECHNICAL UNIVERSITY
FACULTY OF MECHANICAL ENGINEERING

GPS-Guided Path Following Autonomous Vehicle with Collision Avoidance

1606A033 Mahsun BİNGÖL

1606A040 Özkan GÖKSU

1606A019 Kadir Ertuğ ACAR

PREPARED BY MECHATRONICS ENGINEERING DEPARTMENT

MECHATRONIC SYSTEM DESIGN REPORT

Project Advisor: Assoc. Prof. Aydın YEŞİLDİREK

ISTANBUL, 2021

YILDIZ TECHNICAL UNIVERSITY
FACULTY OF MECHANICAL ENGINEERING

GPS-Guided Path Following Autonomous Vehicle with Collision Avoidance

1606A033 Mahsun BİNGÖL

1606A040 Özkan GÖKSU

1606A019 Kadir Ertuğ ACAR

PREPARED BY MECHATRONICS ENGINEERING DEPARTMENT

MECHATRONIC SYSTEM DESIGN REPORT

Project Advisor: Assoc. Prof. Aydın YEŞİLDİREK

ISTANBUL, 2021

MECHATRONIC SYSTEM DESIGN REPORT

CONTENTS

Page

REVISION HISTORY iii

LIST OF SYMBOLS 1

LIST OF ABBREVIATION 2

LIST OF FIGURES 3

LIST OF TABLES 5

ACKNOWLEDGEMENT 6

ABSTRACT 7

INTRODUCTION 8

1.1 Definition 8
1.2 Purpose 9
1.3 Motivation 10
1.4 Scope 10
1.5 Literature Review 11
1.5.1 Autonomous Driving Levels 11
1.5.2 Related Studies 13
1.6 Assumptions 15

2. REQUIREMENTS SPECIFICATION 16

2.1 Stakeholder Analysis 16
2.2 Market Requirements 16
2.3 Technical Requirements 17
2.4 Design Specifications 18

3. THEORITICAL BACKGROUND 19

3.1 Software Tools 19
3.2 Mapping and Path Planning 21
3.2.1 What is SLAM? 22
3.3 Designing Vehicle Control 23
3.3.1 Designing Lateral Control for Path Tracking 24
3.3.2 Pure Pursuit Control 25
3.4 Lane Detection 27
3.5 Object Detection 28

4. DESIGN 30

4.1 Overall System 30
4.2 System Decomposition 31
4.3 Planning of Project 31
4.4 Sensors 32
4.4.1 Camera 32

4.4.2 Lidar 34
4.4.3 GPS (Global Positioning System) 36
4.4.4 IMU (Inertial Measurement Unit) 37
4.5 Lane Detection 38
4.5.1 Image Thresholding 38
4.5.2 Perspective Transform 40
4.5.3 Detect Lane Pixels 40
4.5.4 Determine the Curvature 41
4.5.5 Back onto the Original Image 42
4.6 Object Detection 42
4.7 Simulation Environment and Vehicle Selection for Mapping 44
4.7.1 Designing Simulation Environment 46
4.7.2 ROS Packages 46
4.7.3 Collision Avoidance 51
4.7.4 ROS Rqt Graph 54
4.8 Simulink Model and Results of Pure Pursuit Control 54
4.9 Electrical Design 64
4.9.1 Vehicle Motor 64
4.9.2 Simplification of System 65
4.9.3 Dynamic model 65
4.9.4 Defining Motor Requirements 68
4.9.5 Simulation 70
4.10 Mechanical Design 72
4.10.1 Safety Factor 74
4.10.2 Stress 75
4.10.3 Displacement 76

5. WORKING PLAN 77

6. BUDGET 78

6.1 List of Materials 78
6.2 Procurement of Services 78
6.3 External Financial Support Applications 79

7. RESULTS, DISCUSSION AND FUTURE WORKS 80

REFERENCE 82

RESUME 85

REVISION HISTORY

Date	Rev. No	Definition	Author(s)
17/01/2020	1.0	Initial document	Mahsun BİNGÖL Kadir Ertuğ ACAR Özkan GÖKSU
21/01/2020	1.1	Missing parts are completed	Mahsun BİNGÖL Kadir Ertuğ ACAR Özkan GÖKSU

LIST OF SYMBOLS

R	Radius [m]
δ	Steering angle of the front wheel
L	The distance between the front axle and rear axle (wheelbase)
(g_x, g_y)	Goal point
ℓ_d	Look-ahead distance
k	The curvature of the circular arc

LIST OF ABBREVIATION

LIDAR	Light Detection and Ranging
HD	High Definition
FHD	Full High Definition
ROS	Robot Operating System
RADAR	Radio Detection and Ranging
IMU	Inertial Measurement Unit
GPS	Global Positioning System
2D	Two-Dimension
3D	Three-Dimension
FPS	Frame per Second
FOV	Field of View
GNSS	Global Navigation Satellite Systems
RGB	Red-Green-Blue
SAE	Society of Automotive Engineers
SLAM	Simultaneous Localization and Mapping
NHTSA	US National Highway Traffic Safety Administration
DDT	Dynamic Driving Task
ADS	Automated Driving System
ODD	Operational Design Domain
OEDR	Object and Event Detection and Response
YOLO	You Look Only Once

LIST OF FIGURES

Figure 1.1.1 The future traffic with autonomous cars 8
Figure 1.1.2 Estimated SAE Level 3 and 4 sales between in 2020 – 2030 [1] 9
Figure 1.3.1: Traffic accident statistics [2] 10
Figure 1.5.1.1 Summary of levels of driving automation [3] 12
Figure 1.5.2.1: MIT RACECAR vehicle platform 14
Figure 1.5.2.2: F1/10 race competition 14
Figure 2.1.1 Stakeholder Graph 16
Figure 2.2.1 Market Size forecast [7] 17
Figure 3.1.1 Working of ROS [8] 20
Figure 3.1.2 The ROS Master [8] 20
Figure 3.2.1.1 Ros-Based Simultaneous Localization and Mapping (Slam) Systems [26] 22
Figure 3.2.1.2 Maps created using the various algorithms and the Hokuyo laser scanner [26] 23
Figure 3.3.1.1 Comparison of Lateral Controllers 24
Figure 3.3.2.1 Geometric Bicycle Model [33] 25
Figure 3.3.2.2 Pure Pursuit Geometry [33] 26
Figure 3.5.1 YOLOv3 vs Other Algorithms 29
Figure 4.1.1 Overall System 30
Figure 4.2.1 System Decomposition 31
Figure 4.4.1.1 Intel® RealSense™ Depth Camera D435 [1] 34
Figure 4.4.2.1 Lidar Point Cloud Visualization 35
Figure 4.4.2.2 RPLIDAR A3M1 [15] 36
Figure 4.4.3.1 UBLOX NEO-6M 37
Figure 4.4.4.1 Sparkfun 9DoF Razor IMU M0 38
Figure 4.5.1.1: Original Image 38
Figure 4.5.1.2 Abs Sobel Threshold 38
Figure 4.5.1.3 Magnitude Threshold 39
Figure 4.5.1.4 Direction Threshold 39
Figure 4.5.1.5 HLS S-Channel Threshold 39
Figure 4.5.1.6 HLS L-Channel Threshold 39
Figure 4.5.1.7 LUV Threshold 40
Figure 4.5.1.8 Combined Gradient 40
Figure 4.5.2.1 Region of interest perspective warped to generate a Bird's-eye view 40
Figure 4.5.3.1 Histogram 41
Figure 4.5.3.2 Sliding window fit results 41
Figure 4.5.4.1 Identified lane lines 42
Figure 4.5.5.1 Processed Image 42
Figure 4.6.1 YOLO Version Performance[25] 43
Figure 4.6.2 YOLOv3 Test 44
Figure 4.6.3 Darknet Source Code Manipulated 44
Figure 4.7.1 Jackal Ground Vehicle 45
Figure 4.7.2 Jackal Ground Vehicle in Gazebo 45
Figure 4.7.1.1 "TEKNOFEST Robotaksi Binek Otonom Araç" Competition Track 46
Figure 4.7.1.2 Desired track in Gazebo 46
Figure 4.7.2.1 Final map of the track 50

Figure 4.7.2.2 Global X, Y and Z coordinate values shown in the terminal 51
Figure 4.7.3.1 Collison Avoidance with Jackal in Gazebo 53
Figure 4.7.4.1 ROS rqt graph 54
Figure 4.7.4.1 Simulink Model of Pure Pursuit Control 55
Figure 4.7.4.2 Code of Pure Pursuit Control 55
Figure 4.7.4.3 Desired road and waypoints 56
Figure 4.7.4.4 Simulation Result with V = 1 $\boldsymbol{\ell d}$ = 1 57
Figure 4.7.4.5 Steering Angle with V = 1 $\boldsymbol{\ell d}$ = 1 57
Figure 4.7.4.6 Simulation Result with V = 1 $\boldsymbol{\ell d}$ = 1 58
Figure 4.7.4.7 Steering Angle with V = 1 $\boldsymbol{\ell d}$ = 1 59
Figure 4.7.4.8 Simulation Result with V = 1 $\boldsymbol{\ell d}$ = 3 60
Figure 4.7.4.9 Steering Angle with V = 1 $\boldsymbol{\ell d}$ = 3 60
Figure 4.7.4.10 Simulation Result with V = 10 $\boldsymbol{\ell d}$ = 1 61
Figure 4.7.4.11 Simulation Result with V = 15 $\boldsymbol{\ell d}$ = 1 61
Figure 4.7.4.12 Simulation Result with V = 10 $\boldsymbol{\ell d}$ = 2 62
Figure 4.7.4.13 Simulation Result with V = 15 $\boldsymbol{\ell d}$ = 2 62
Figure 4.7.4.14 Simulation Result with V = 10 $\boldsymbol{\ell d}$ = 3 63
Figure 4.7.4.15 Simulation Result with V = 15 $\boldsymbol{\ell d}$ = 3 63
Figure 4.9.1 Electrical System Block Diagram 64
Figure 4.9.2.1 System schematic 65
Figure 4.9.3.1 Rotational part 65
Figure 4.9.3.2 Translational part 66
Figure 4.9.3.3 Force on the vehicle 66
Figure 4.9.3.4 Equivalent system 67
Figure 4.9.4.1 Traxxas 2923X 3000mAh NiMH 7-C Flat 8.4V Battery 69
Figure 4.9.5.1 Electrical drive system MATLAB model 70
Figure 4.9.5.2 Four – quadrant operation 71
Figure 4.9.5.3 Power graph 71
Figure 4.9.5.4 Battery state of charge 72
Figure 4.10.1 TRAXXAS Bigfoot No 1 [34] 72
Figure 4.10.2 Model Car 73
Figure 4.10.1.1 Safety Factor 75
Figure 4.10.2.1 Von Mises 75
Figure 4.10.2.2 1st Principal 75
Figure 4.10.2.3 3rd Principal 76
Figure 4.10.3.1 Displacement 76

LIST OF TABLES

Table 2.3.1 Technical Requirements

Table 4.3.1 Planning of Project

Table 4.4.1.1 Camera Technical Requirements

Table 4.4.2.1 LIDAR Technical Requirements

Table 4.10.1 Result Summary of Stress Test

Table 5.1 Working Plan

Table 6.1.1 Total Cost of Materials

Table 6.2.1 Procurement of Services

Table 4.9.3.1 Transmission elements properties

Table 4.9.3.2 Equivalent inertia calculation

ACKNOWLEDGEMENT

In this project, not only the project team but also many people have contributed. Firstly, we would like to express my very great appreciation to Assoc. Prof. Dr. Aydın YEŞİLDİREK for his valuable and constructive suggestions during the planning and development of this research work. His willingness to give his time so generously has been very much appreciated. Also, we would like to thank our families for always supporting us. We would also like to extend our thanks to Rass Technology for their collaboration in TUBITAK 2209B. Lastly, we want to thank the Alternative Energized Systems Society Autonomous Team and our team leader's coworkers there for helping us discover our interest in robotics and autonomous systems.

ABSTRACT

According to TURKSTAT, % 88 traffic accidents occurred due to drivers in the first 9 months of 2020. With the development of autonomous vehicles, it is expected that the traffic flow will become more stable and safer. In this way, life and property losses will be prevented. People who do not have the ability to drive have difficulty traveling alone. Autonomous vehicles provide humans independent and reliable driving. Normally, drivers should give their full attention to driving while driving for a safe journey. In Turkey, the average commute time is 48 minutes. Personal time can be increased with autonomy. It is aimed to increase the quality of life and productivity of the person.

In the autonomous vehicle project that we started to work in line with these expectations, the necessary literature reviews were made, and the scope was determined.

One of the most basic conditions of going on the road with following the lane(Sliding Window), perceiving and interpreting the traffic signs(YOLOv3-Tiny), detecting its position according to its environment (mapping and localization)(SLAM), detecting and avoiding obstacles on the road and path planning(MoveBase ROS). To control lateral movement, Pure Pursuit algorithm is selected. Aimed tasks were tested in simulation in MATLAB, ROS, Visual Studio, and Pycharm.

Keywords: Autonomous Vehicle, Lane Detection, Sliding Window, Object Detection, YOLOv3-Tiny, Mapping, SLAM, ROS, Path Planning, Collision Avoidance, RC Car, Lateral Control, Pure Pursuit

INTRODUCTION

1.1 Definition

The autonomous vehicle industry, which started in the last century and is now a major revolution, is growing day by day. Companies are making huge investments and R&D working on autonomous cars and drive systems. The reasons for all the studies and investments are saving human's life and provide a safe traffic flow.

Figure 1.1.1 The future traffic with autonomous cars

Self-driving cars provide completely new ways to deal with traffic problems of autonomous cars could in the future handle entire cities' need for personal mobility with much fewer vehicles than are used today. SAE International (Society of Automotive Engineers) has developed a six-level standard (0–5), where semi-autonomous functionality starts in the second level. The third level is the first to provide some actual autonomy in the sense that the driver can divert attention from the road although he or she must be able to regain control of the vehicle with some seconds of prior warning. The fourth level provides full autonomy in specific use cases of various complexity.

In 2015, the first semi-autonomous car model was introduced in the automotive industry after that the number of autonomous cars is increasing every year. By 2030 an estimated about 200 million autonomous cars on different levels will be sold by the market.

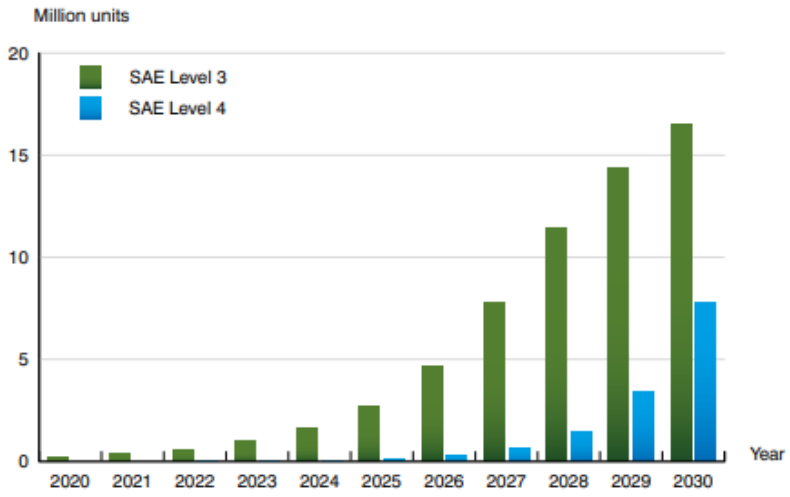


Figure 1.1.2 Estimated SAE Level 3 and 4 sales between in 2020 – 2030 [1]

In this project, a fully autonomous 1:10 model vehicle was studied. It has been ensured that the vehicle obeys the traffic signs and traffic lights on a designated road by the rules, avoids the obstacles or stops in scenarios where it should stop, and moves smoothly on the specified trajectory.

Also, a telemetry system has been built to remotely control and monitor the instantaneous speed, state, battery life, and parameters of the vehicle by the user.

YOLO algorithm will be used to detect objects around the vehicle, SLAM algorithm for mapping and localization along the road, OpenCV Library for detection of road lines, and Pure Pursuit control for vehicle control.

1.2 Purpose

This project aims to build a 1:10 model of a real and applicable autonomous vehicle. While the vehicle performs this task, the camera for recognizing road lines and objects, Lidar for detecting and avoiding objects, and the fusion of these two sensors for mapping and localization.

Also, GPS and IMU will be used to calculate the instant position of the vehicle on the Earth and the change of position, and the effect of the position change on the trajectory. Thanks to the fusion of these two sensors, it is desired to create a reasonable trajectory for the vehicle.

It is thought that the data to be detected by the sensors is large and a system with a powerful GPU will be needed for artificial intelligence, object detection, and vehicle control to be made on the data and it is planned to use an NVIDIA Jetson TX2 processor card.

1.3 Motivation

The main motivation for the project is to prevent traffic accidents. These accidents cause loss of life and property. According to the Turkey Statistical Institute data in 2019 a total of 1,168,144 accidents occurred in Turkey and a total of 5473 people lost their lives in this accident. In addition, according to data causing mortal injury traffic accidents 88.0% of the total of 204 thousand 538 defects referring to drive defect, 8.2% of pedestrians, vehicles of 2.0%, 1.3% it has been observed that the reputation is due to the passenger and 0.5% to the road in Turkey in 2019. [2]

Trafik kaza istatistikleri, 2009-2019

Yıl	Toplam kaza sayısı	Ölümlü yaralanmalı kaza sayısı	Maddi hasarlı kaza sayısı	Ölü sayısı			Yaralı sayısı
				Toplam	Kaza yerinde	Kaza sonrası(1)	
2009	**1 053 346**	111 121	942 225	**4 324**	4 324	-	201 380
2010	**1 106 201**	116 804	989 397	**4 045**	4 045	-	211 496
2011	**1 228 928**	131 845	1 097 083	**3 835**	3 835	-	238 074
2012	**1 296 634**	153 552	1 143 082	**3 750**	3 750	-	268 079
2013	**1 207 354**	161 306	1 046 048	**3 685**	3 685	-	274 829
2014	**1 199 010**	168 512	1 030 498	**3 524**	3 524	-	285 059
2015	**1 313 359**	183 011	1 130 348	**7 530**	3 831	3 699	304 421
2016	**1 182 491**	185 128	997 363	**7 300**	3 493	3 807	303 812
2017	**1 202 716**	182 669	1 020 047	**7 427**	3 534	3 893	300 383
2018	**1 229 364**	186 532	1 042 832	**6 675**	3 368	3 307	307 071
2019	**1 168 144**	174 896	993 248	**5 473**	2 524	2 949	283 234

(1) Trafik kazasında yaralanıp sağlık kuruluşuna sevk edilenlerden kazanın sebep ve tesiriyle 30 gün içinde ölenleri kapsamaktadır.

- Bilgi yoktur.

Figure 1.3.1: Traffic accident statistics [2]

Apart from traffic accidents, with the development of the autonomous vehicle sector, the investments of companies and states in this area have also been a source of motivation for the project. With the advancing artificial intelligence and powerful computers, better algorithms are created by using better sensors.

1.4 Scope

Autonomous vehicles provide safe and stable driving by using driving algorithms with the data they receive from the sensors they use. Autonomous driving algorithms are basically 4: perception of the environment, object detection, mapping/localization and path planning. The

scope of the project was determined based on these 4 basic tasks. Tasks and scope planned to be done in the project are as follows:

• Finding and following the road lines of the vehicle

• Detection of traffic signs and lights in front of the vehicle along the road

• The vehicle avoids obstacles along the way

• The vehicle moves to a specific location

• Mapping the environment during the movement of the vehicle and instantly following the path is determined.

• Control of the vehicle

• Tracking and observing the instant information of the vehicle by the user

1.5 Literature Review

This section includes literature research on autonomous vehicles. Detailed information will be given about the autonomous driving levels previously described. In addition, projects and competitions similar to the vehicle to be designed within the scope of the project will be mentioned.

1.5.1 Autonomous Driving Levels

The sector's autonomous vehicle levels are broken down into 2 main classifications. The US National Highway Traffic Safety Administration (NHTSA) is the first of these, and the Society of Automotive Engineers is the other (SAE). The main difference between these two is that SAE uses a classification of 6 levels, while NHTSA uses a classification of 5 levels. NHTSA has accepted the 6-level classification published by SAE. For this reason, SAE's 6-level autonomous vehicle classification was adopted in our study.

Level	Name	Narrative definition	DDT		DDT fallback	ODD
			Sustained lateral and longitudinal vehicle motion control	OEDR		
Driver* performs part or all of the *DDT						
0	**No Driving Automation**	The performance by the *driver* of the entire *DDT*, even when enhanced by *active safety systems*.	*Driver*	*Driver*	*Driver*	n/a
1	**Driver Assistance**	The *sustained* and *ODD*-specific execution by a *driving automation system* of either the *lateral* or the *longitudinal vehicle motion control* subtask of the DDT (but not both simultaneously) with the expectation that the *driver* performs the remainder of the *DDT*.	*Driver* and *System*	*Driver*	*Driver*	Limited
2	**Partial Driving Automation**	The *sustained* and *ODD*-specific execution by a *driving automation system* of both the *lateral* and *longitudinal vehicle motion control* subtasks of the *DDT* with the expectation that the *driver* completes the *OEDR* subtask and *supervises* the *driving automation system*.	***System***	*Driver*	*Driver*	Limited
***ADS* ("*System*") performs the entire *DDT* (while engaged)**			*System*	***System***	*Fallback-ready user (becomes the driver during fallback)*	Limited
3	**Conditional Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire DDT with the expectation that the *DDT fallback-ready user* is *receptive* to *ADS*-issued *requests to intervene*, as well as to *DDT performance-relevant system failures* in other *vehicle* systems, and will respond appropriately.				
4	**High Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	***System***	Limited
5	**Full Driving Automation**	The *sustained* and unconditional (i.e., not *ODD*-specific) performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	*System*	**Unlimited**

Figure 1.5.1.1 Summary of levels of driving automation [3]

DDT (Dynamic Driving Task): The organizational and tactical real-time functions needed to operate a vehicle safely in on-road traffic.

ADS (Automated Driving System): Hardware and software that are collectively capable of performing all DDT on a sustained basis, irrespective of whether it is limited to a particular area of operational design (ODD); this concept is explicitly used to describe a driving automation system at level 3, 4, or 5.

ODD (Operational Design Domain): Operating conditions in which a particular driving

automation system or feature thereof is explicitly designed to operate, including, but not limited to, constraints on the climate, geography and time of day, and/or the presence or absence of certain characteristics of traffic or route, as needed.[4]

OEDR (Object and Event Detection and Response): The driver or system's understanding of any circumstances that are important to an immediate driving mission, as well as the appropriate response of the driver or system to such circumstances.

1.5.2 Related Studies

MIT Racecar

MIT RACECAR is an open-source hardware platform created by the Massachusetts Institute of Technology for robotics research and education. In the relevant undergraduate course, the platform is used to teach students about car perception and planning and control algorithms that can quickly navigate complex environments. There is also a competition track for students to test the success of the algorithm that they designed a platform in the school. Let's talk about the race car, sensors, and development environment used on the platform: [5]

• The RACECAR platform is based on the 1/10 scale “Traxxas Rally” vehicle.

• NVIDIA Jetson TX1 is used as the central processing unit.

• Sparkfun IMU and visual odometer are used for position estimation.

• The vehicle is equipped with a laser scanner Hokuyo UST-10LX and a stereo camera (Stereolabs ZED).

• In order to test the platform in a virtual environment, ROS software is used where different algorithms can work together and sensor fusion can be performed.[5]

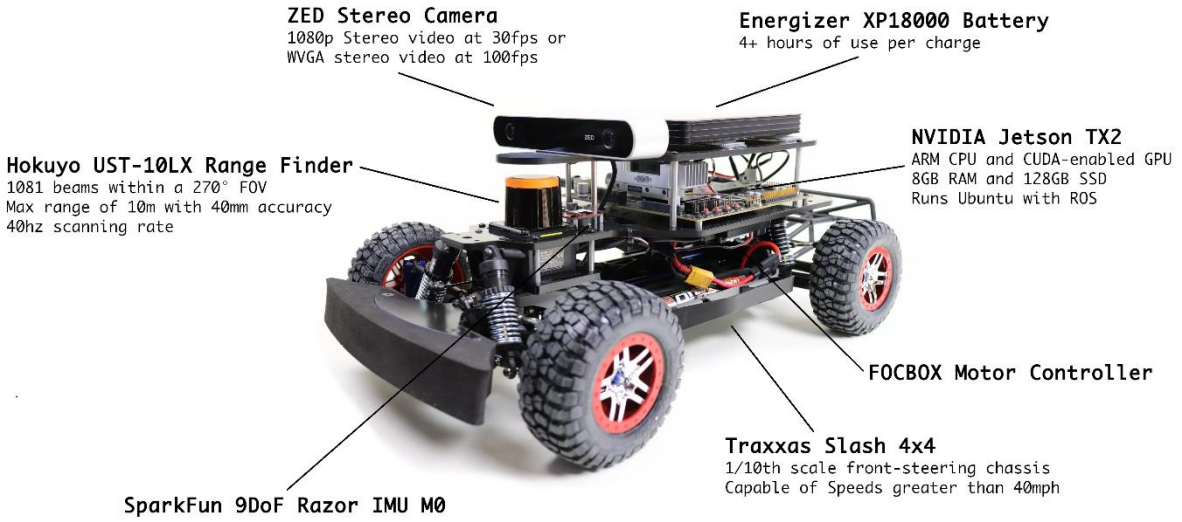


Figure 1.5.2.1: MIT RACECAR vehicle platform

F1/10

F1TENTH is an international group of enthusiasts for researchers, engineers, and autonomous systems. It was initially founded in 2016 at the University of Pennsylvania but has since spread to several other institutions around the world. Their purpose is to stimulate curiosity, passion, and critical thinking about the autonomous systems.[6]

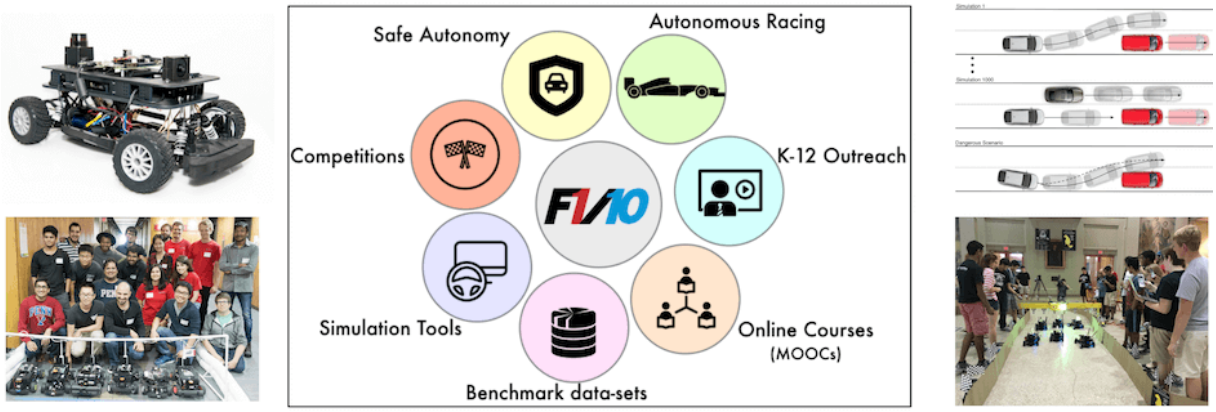


Figure 1.5.2.2: F1/10 race competition

1.6 Assumptions

In this section, the assumptions and assumptions about the project will be mentioned. We mentioned that we are based on 4 basic goals within the scope of the project. These tasks have certain limitations.

In the mapping task, it will be limited to a small-scale track to be designed. The vehicle will travel on a straight road as much as possible. You will not face a dynamic obstacle.

In the object detection task, Stop, No entry, and Yield traffic signs will be used as traffic signs.

In the route-finding task, the vehicle will create a route on the road in the track.

Studies on the physical properties of the track will be carried out in the second term within the scope of the Final Study.

2. REQUIREMENTS SPECIFICATION

2.1 Stakeholder Analysis

In every project, there are stakeholders involved in the implementation of the project which are supporters of the project, those who have expectations from the project, project consultant, etc. Those who contributed to this project are given below, taking into account their contributions.

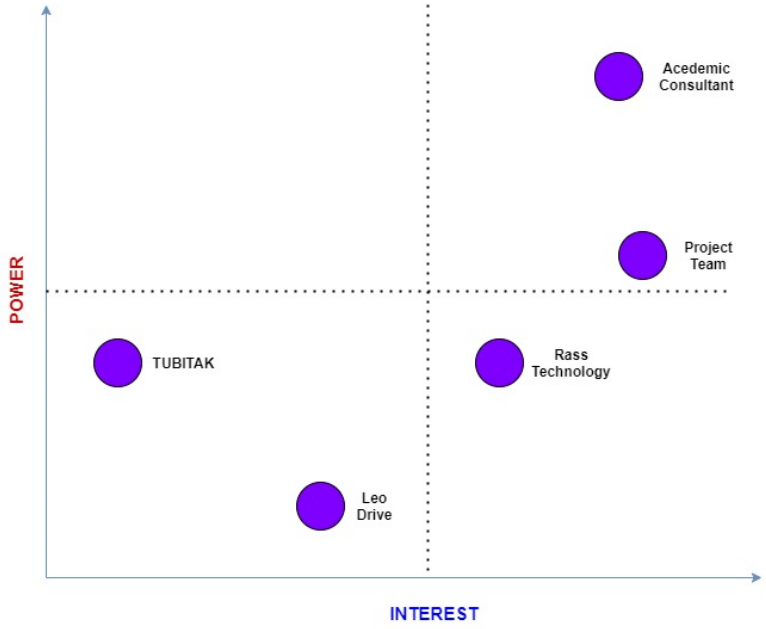


Figure 2.1.1 Stakeholder Graph

2.2 Market Requirements

The global autonomous vehicle market size is $60 billion in 2020 and is projected to garner $556.67 billion by 2026. The main problem is still battery life and recharge points. Investments are ongoing for this problem, but still, it is not sufficient.

Figure 2.2.1 Market Size forecast [7]

2.3 Technical Requirements

In addition to determining the scope during the design of the project, technical requirements should also be determined. In this way, the area to be worked on becomes clear. Equipment selection and control parameters will be made considering these limitations. Technical requirements were determined as a result of literature studies, and it was aimed to save energy by choosing relatively few items that are not the main test elements such as the speed of the vehicle.

Table 2.3.1 Technical Requirements

Vehicle Weight	**3.5 kg**
Maximum Velocity	1 m/s
Maximum Avoidance Distance	0.5 m
Traffic Signs to Recognize	STOP - No Entry - Give Way
Confidence of Object Detection	90%
Battery Working Life	20 minutes

Equipment specifications are mentioned in Section 4.4 and limitations of interest are given there.

2.4 Design Specifications

During the design, the features expected to be made from the vehicle were determined and the parameters were selected accordingly. The working time and energy distribution of the vehicle, the working environment, the movement limit in the working environment, the object detection accuracy, the detection of the main road, the deviation of the targeted point are the main design elements.

This project is made to test autonomous vehicle behavior. Nearly 20 min is enough to finish parkour more than one time. Necessary calculations are in Section 4.9.

The model vehicle will be tested in 1/10 scaled test parkour in real and simulation. Weather conditions must be good (Sunny, windless, rainless) to obtain good data from sensors.

Traffic signs are a very important element of traffic. They must be recognized with high accuracy. Confidence is selected as %90 according to processor capacity.

In traffic, its aimed that vehicle can approach to other vehicle up to max 0.5 meter for safety. It is chosen as constant for now but can be change with adaptive speed controller design.

The model vehicle must stay between lane lines. It is the one of the most important rule in traffic for avoid Collison easily. Lanes must be recognized with %100 accuracy.

3. THEORITICAL BACKGROUND

In this section, information about the programs used in the project will be given. Programs allow us to design, test and simulate the project and analyze the results.

3.1 Software Tools

Robot Operating System (ROS)

A literature study has been made for each task title for the project and the necessary algorithms for autonomous driving have been learned. These algorithms will be written in C ++ and Python languages throughout the project on the grounds of the suitability of the language tools, ease of writing, and the necessary libraries. The algorithms written will be used with tools for robot applications, libraries, ease of sending and receiving messages between codes written with different languages, necessary packages and ROS software, which is an open source software, and the Ubuntu operating system. ROS (Robot Operating System) mobile robot, drone, industrial robot arms etc. It is an open source operating system developed for robotic systems such as. The description of ROS software is as follows:

- Packages: Includes ROS operations, libraries and configurations
- Nodes: It is the unit where task operations are performed.
- Master: It is the unit that provides the control that communicates all running packages and nodes.
- Publisher / Subscriber: The publisher sends the tasks, the subscriber performs these tasks on the node.
- Messages: Used for standard communication between nodes. Provides data communication.
- Topics: It is the unit of communication between different subscribers and publishers.
- Rosbags: It is a tool used to record messages on topics.
- Rviz: It is the program used to visualize the data used in ROS operations.
- Rqt: Tool used to view data flow diagrams.

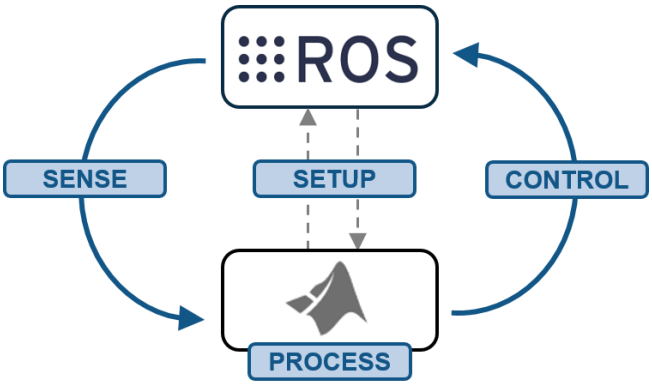


Figure 3.1.1 Working of ROS [8]

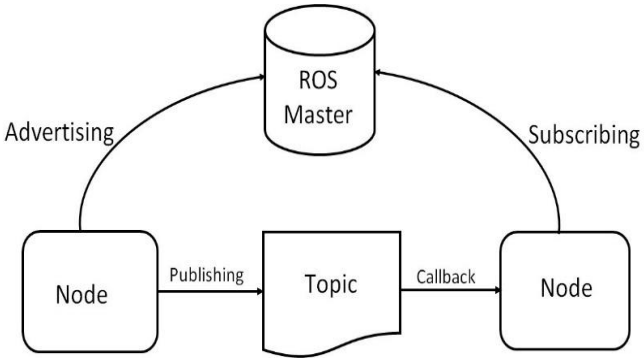


Figure 3.1.2 The ROS Master [8]

Within the scope of the project, ROS Melodic will be used due to both resource abundance and some packages and software not working in Kinetic version. [9]

Within the scope of the project, we realized the communication between the autonomous driving algorithm through ROS. We created mapping and routing algorithms using ROS packages. We controlled the vehicle on the simulation using driving algorithms.

Ubuntu Operating System

Ubuntu is a complete Linux operating system available for free with both community and professional support. Since ROS runs on the Ubuntu operating system, it becomes mandatory for us to use Ubuntu. Since we will be using the ROS Melodic version, 18.04.5 LTS was chosen as the Ubuntu version that supports it. [10]

MATLAB

Matlab helps to design, test, simulate and analyze the results of engineering projects with the various toolboxes it contains. As part of the project, we performed vehicle control, simulation and analysis using Matlab.

Gazebo

Gazebo is a 3D dynamic simulator capable of accurately and efficiently simulating robot populations in complex indoor and outdoor environments. While similar to game engines, Gazebo offers a much higher level of fidelity physics simulation, a sensor package and interfaces for both users and programs. [11]

Within the scope of the project, Jackal mobile vehicle was used in Gazebo environment. A track was built for the vehicle and the mapping and route-finding algorithms of the vehicle within the track were tested.

SolidWorks

Solidworks is a 3D computer-aided design program. Within the scope of the project, we completed the physical construction of the vehicle and the construction of the sensors on the vehicle.

Fusion 360

Fusion 360 is a cloud-based CAD / CAM tool for collaborative product development. Fusion 360 provides discovery and iteration on product ideas and collaboration within the distributed product development team. Fusion 360 combines organic shape modeling, mechanical design and manufacturing in one comprehensive package. [12]

Within the scope of the project, we positioned the sensors on a platform built on the vehicle. Stress and force analyze of this designed platform were performed.

Visual Studio Code

Visual Studio Code is a code editor where autonomous driving algorithms are compiled, debugged and optimized.

3.2 Mapping and Path Planning

In this section, mapping and path planning algorithms and their use together with ROS in

Gazebo environment will be discussed.

Mapping is very important in autonomous vehicles. In order for the vehicle to avoid obstacles and create a safe route, the vehicle must instantly perceive the location and the environment where the vehicle is located. Mapping algorithms show the environment of the vehicle according to its location. There are many algorithms for mapping. SLAM (Simultaneous Localization and Mapping) is the most used mapping algorithm.

3.2.1 What is SLAM?

Basically, it predicts the map of the area where the vehicle is located and its location according to this map with the help of sensors in an unknown environment. The mentioned location estimation is called localization in the literature. It has two main steps:

1. Prediction step. This is done when the vehicle localization and map status are updated using previous information about the system status and access control commands.
2. Measurement update. Matching existing sensor data with predicted system state to predict new system state.

After the specified steps, the algorithm takes the location information to create a map. Then, localization is completed with map information.

SLAM can be used with many sensors. Mostly mobile tools are used with IMU (Initial Measurement Unit), encoder Lidar. There are different SLAM algorithms according to the sensors used. Some of these are listed below.

Year	System	Sensor
2007	GMapping	2D lidar
2007	Parallel Tracking and Mapping (PTAM)	mono
2011	Hector SLAM	2D lidar
2014	Semi-direct Visual Odometry (SVO)	mono
2014	Large Scale Direct monocular SLAM (LSD SLAM)	mono
2014	Real-Time Appearance-Based Mapping (RTAB map)	stereo
2015	ORB SLAM	mono, stereo
2015	Dense Piecewise Parallel Tracking and Mapping (DPPTAM)	mono
2016	Direct Sparse Odometry (DSO)	mono
2016	Cartographer	2D lidar
2017	Stereo Parallel Tracking and Mapping (S-PTAM)	stereo

Figure 3.2.1.1 Ros-Based Simultaneous Localization and Mapping (Slam) Systems [13]

In this project, 2D Lidar, encoder and IMU in the vehicle will be used together. As a result of the literature research, it was decided to use the Gmapping SLAM algorithm. The main reasons for choosing the Gmapping SLAM algorithm are that it is stronger than other algorithms, it processes noisy data better and has less deviation rate. [14]

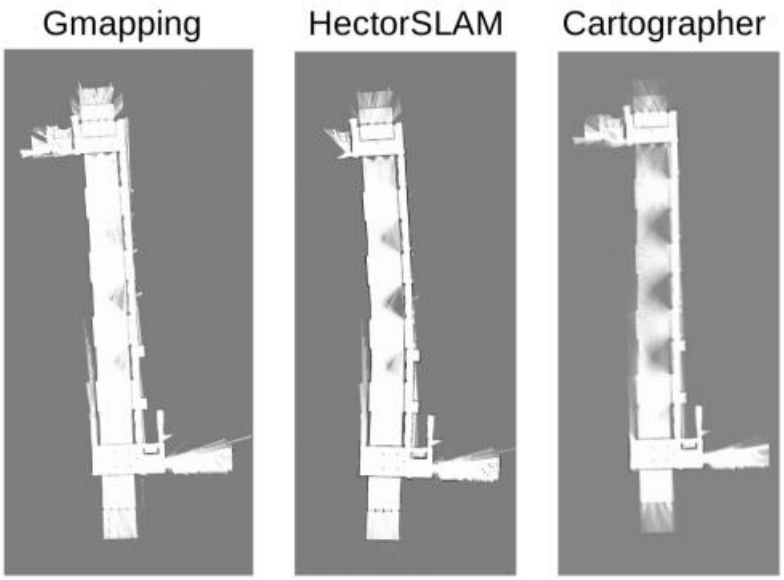


Figure 3.2.1.2 Maps created using the various algorithms and the Hokuyo laser scanner [14]

3.3 Designing Vehicle Control

In this section, the control of the mobile vehicle on the road will be explained. Control is very important for the safe movement of the vehicle on the designated road. Lateral and longitudinal control should be designed for the vehicle. Within the scope of the MST, only the lateral controller is designed, and longitudinal control will not be explained. The criteria taken into consideration when designing a controller are as follows:

- Vehicle speed is taken as constant 1 m / s.

- Axes sets to work with: (x, y) and (x ′, y ′) global and tool coordinate axes. z = 0, z ′ = 0 that is, the road will be considered uneven.
- Angle range = 45 °

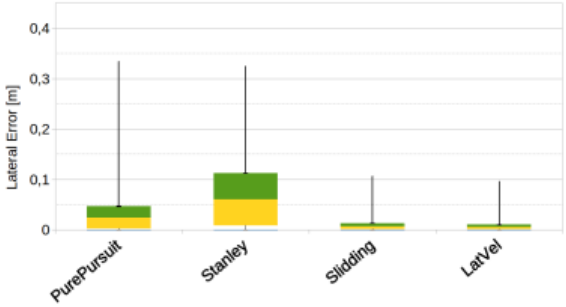
Literature research has been done for lateral control in line with the accepted acceptances.

3.3.1 Designing Lateral Control for Path Tracking

Mapping and routing algorithms determine a global position in the (x, y) coordinate plane using data from GPS, IMU and Lidar. Within the scope of the project, the vehicle will move to a determined point. It makes the movement to the specified point according to the waypoints coming from the path planning algorithm. Path tracking controllers are used to realize the path following waypoints. Basically, their goal is to continuously navigate the road autonomously and drive the mobile vehicle, generating speed and steering commands that compensate for tracking errors mainly due to distance and road deviations. The main controller examples for autonomous mobile vehicles are:

- H_∞ control ⟶using kinematic or dynamic vehicle model
- Sliding mode control and adaptive controllers ⟶ using rough vehicle model
- Pure pursuit and Stanley ⟶ geometric method

Within the scope of this project, it is aimed to use control methods with geometric model approach in terms of not using vehicle dynamics and ease of application. In addition, since general waypoint data among autonomous driving algorithms will be frequent, the geometric model approach will be sufficient. As a result of the literature studies, it has been concluded that the Pure Pursuit control algorithm has an advantage over the Stanley control algorithm in low speed and sharp turns. [15]



	Precision	Stability	Smoothness
P. Pursuit	Acceptable	Very good	High
Stanley	Good at low speed	Bad at higher speeds	Low
Sliding	Very Good	Very good	Acceptable
LatVel	Very Good	Very good	Good

Figure 3.3.1.1 Comparison of Lateral Controllers

3.3.2 Pure Pursuit Control

The main purpose of the Pure Pursuit algorithm is to create the steering angle of the vehicle according to a target point determined on the trajectory. This destination point is a point on the road viewed from the current vehicle position. An arc is created connecting the current point with the target point. The arc length of this arc is its distance from the front and serves as the third constraint in determining a unique arc connecting the two points. Imagine that the look-ahead distance is similar to the distance from a point in a car that a human driver can look to follow the road. Pure pursuit constantly looks in front of the vehicle. As an assumption, it takes the wheel slips of the vehicle as zero.

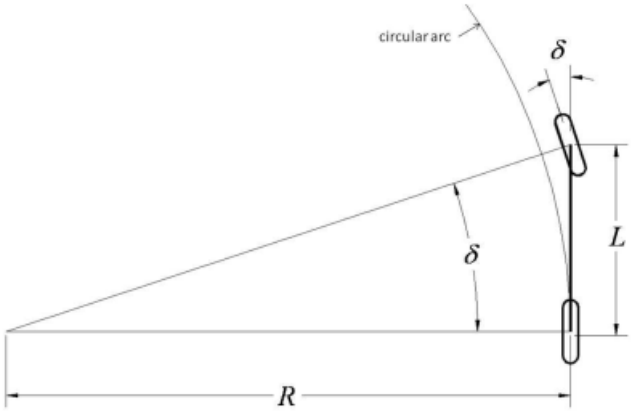


Figure 3.3.2.1 Geometric Bicycle Model [16]

For the geometric model, the bicycle model, which is a simplified version of the Ackerman steered steering model, is used. The bike model combines the front two wheels and combines one wheel and the rear two wheels as a single wheel. Thus, a 4-wheeled vehicle turns into a 2-wheel bike model. As a result, it results in a simple geometric relationship between the front wheel steering angle and the arc the rear axle will follow.

$$\tan(\delta) = \frac{L}{R} \tag{3.3.2.1}$$

δ: steering angle of the front wheel

L: the distance between the front axle and rear axle (wheelbase)

R: the radius of the circle that the rear axle will travel along at the given steering angle

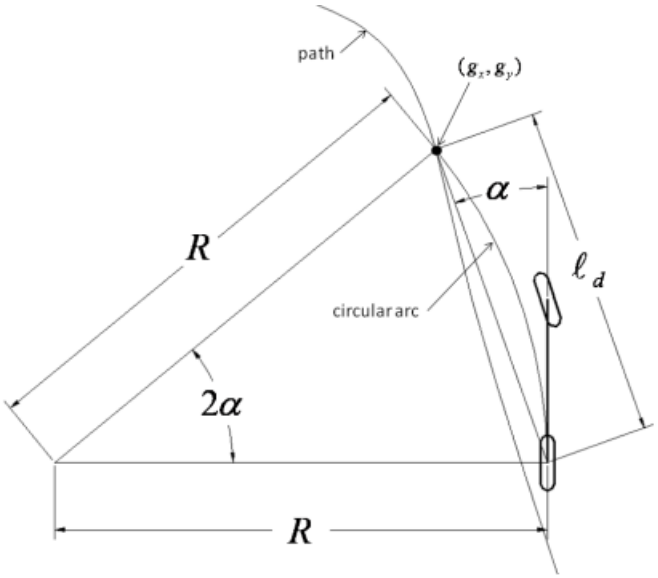


Figure 3.3.2.2 Pure Pursuit Geometry [16]

The pure pursuit method consists of geometrically calculating the curvature of a circular arc that connects the rear axle location to a goal point on the path ahead of the vehicle. The goal point is determined from a look-ahead distance ℓ_d from the current rear axle position to the desired path.

$$\frac{\ell_d}{\sin(2a)} = \frac{R}{\sin(\frac{\pi}{2}-a)} \tag{3.3.2.2}$$

$$\frac{\ell_d}{2\sin(a)\cos(a)} = \frac{R}{\cos(a)} \tag{3.3.2.3}$$

$$\frac{\ell_d}{\sin(a)} = 2R \tag{3.3.2.4}$$

$$k = \frac{2\sin(2a)}{\ell_d} \tag{3.3.2.5}$$

$$\delta = \tan(kL)^{-1} \tag{3.3.2.6}$$

$$\delta(t) = \tan\left(\frac{2Lsin(a(t))}{\ell_d}\right)^{-1} \tag{3.3.2.7}$$

δ: steering angle of the front wheel

L: the distance between the front axle and rear axle (wheelbase)

R: the radius of the circle that the rear axle will travel along at the given steering angle

(g_x, g_y): Goal point

ℓ_d: Look-ahead distance

k: the curvature of the circular arc

The implementation of the pure pursuit algorithm itself is fairly straightforward. The pure pursuit
algorithm can be outlined as follows:

- Determine the current location of the vehicle
- Find the path point closest to the vehicle
- Find the goal point
- Transform the goal point to vehicle coordinates
- Calculate the curvature and request the vehicle to set the steering to that curvature
- Update the vehicle's position.

Basically, Pure Pursuit formula has 1 input and 1 output. Vehicle front wheel angle is calculated according to the determined look-ahead distance. Besides this, vehicle speed is also an important factor. Geometry's behavior changes as the speed increases.

Decreasing ℓ_d may make the system more accurate, but also more oscillatory. Conversely, increasing ℓ_d makes the tracking smoother but less accurate. It is usual to make ℓ_d proportional to velocity.

3.4 Lane Detection

Finding lane lines in camera images can be harder than you think. The probabilistic Hough transform method has a low lane detection rate as it uses edges and restrictive angles. On the other hand, the method using a sliding window can detect a curved lane when the lane is detected by dividing the image into Windows. This section will go through a slightly improved methodology that uses the Sobel operator and color threshold to clear images and uses sliding window search to find and track lane lines.

The process we will fallow can be summarized as:

- Use color transforms and gradients to create a thresholded binary image.
- Apply a perspective transform to rectify binary image ("birds-eye view").

- Detect lane pixels and fit to find the lane boundary.
- Determine the curvature of the lane and vehicle position with respect to center.
- Warp the detected lane boundaries back onto the original image.

Sobel is a technique for gradient estimation. The kernel will be translated in the direction of the image matrix and the element-wise kernel product and corresponding image pixels will be measured at each point. To get the derivative (gradient) at that point, the resulting matrix that is at the same kernel size will be summed up. The derivative in the x direction will be positive when S_x is applied to a segment of an image where pixel values increase from left to right.[17]

$$S_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \; S_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{3.1}$$

To calculate the radius of curvature that can be calculated with a circle near points in the local section of the curve. The radius of curvature of the curve at a given point can be described as the approximate radius of the circle. This radius can be calculated using the formula. [20]

$$R_{curve} = \frac{\left[1+\left(\frac{dy}{dx}\right)^2\right]^{\frac{3}{2}}}{\left|\frac{d^2x}{dy^2}\right|} \tag{3.2}$$

3.5 Object Detection

Object Detection is a computer technology which is related with computer vision and image processing. It is widely used in image processing, face detection, face recognition etc. In this study it is used to detect traffic light. There are three main object detection algorithm which depend on deep learning:

1- R-CNN and its derivatives
2- Single Shot Detector(SSD)
3- YOLO

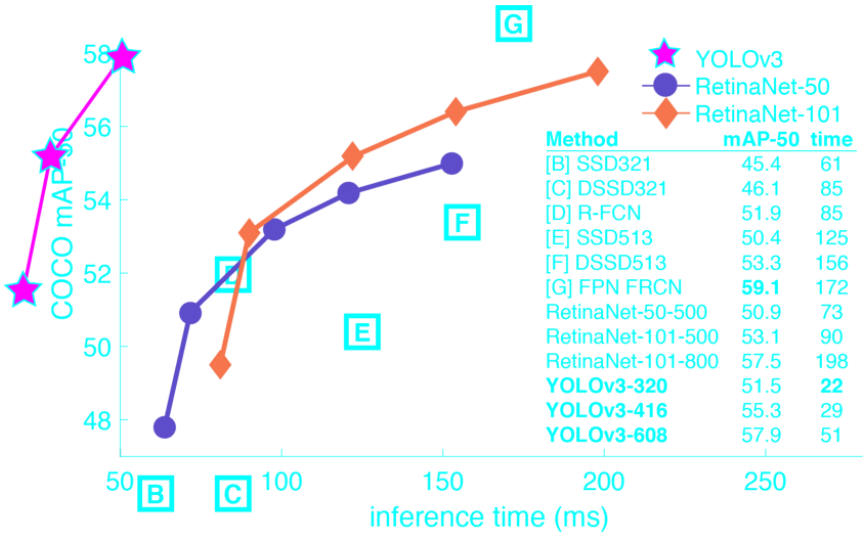


Figure 3.5.1 YOLOv3 vs Other Algorithms

As you can see, YOLO algorithm is the best choice for real time applications.

YOLO is abbreviation of “You Look Only Once” and it uses convolutional neural networks (CNN) for object detection. A single network just once applied to whole image. It can predict classes’ labels and detect location of objects at the same time during working. That is the reason of why YOLO can detect multiple objects at the same time. It divides image into regions and predicts bounding boxes and probabilities for each region. Also, it can predict confidence of bounding box.

4. DESIGN

4.1 Overall System

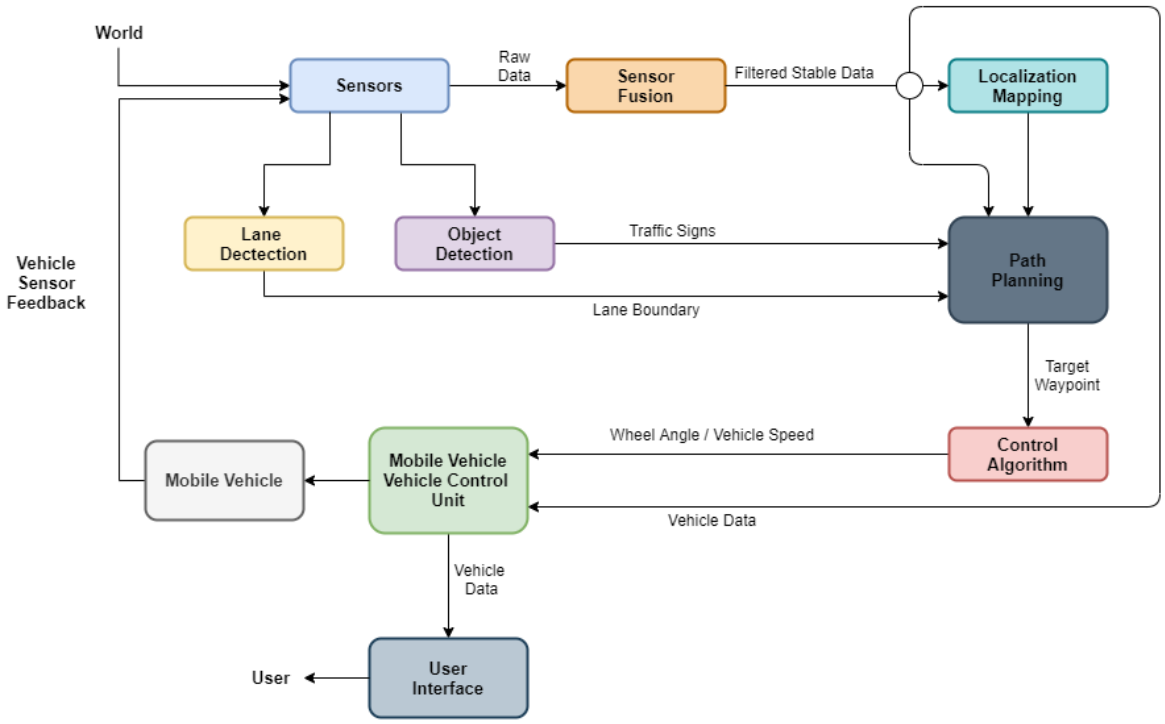


Figure 4.1.1 Overall System

4.2 System Decomposition

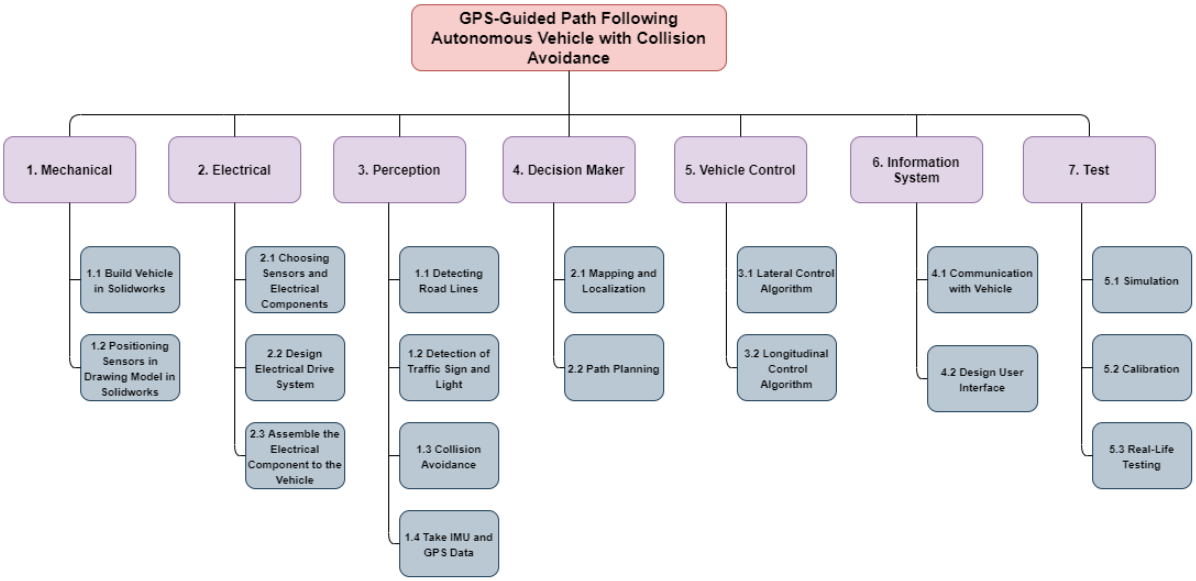


Figure 4.2.1 System Decomposition

4.3 Planning of Project

Table 4.3.1 Planning of Project

Task Packages	By whom	Date	Success Criterion
Developing lane detection algorithm	Özkan GÖKSU	12/10/2020 – 16/11/2020	The vehicle can go on lane roads with the algorithm to detect lane lines
Developing object detection algorithm	Kadir Ertuğ ACAR	12/10/2020 – 16/11/2020	Acting in accordance with traffic rules with an algorithm that will identify pedestrians, traffic signs and vehicles in traffic with 90% accuracy.
Developing Simultaneous Localization and Mapping (SLAM) algorithm	Mahsun BİNGÖL	12/10/2020 – 07/12/2020	Detecting the location of the vehicle in its environment by simultaneously mapping the vehicle environment by taking Lidar and IMU data
Modeling and analysis of the vehicle	Özkan GÖKSU	02/11/2020 – 30/11/2020	Modeling the vehicle in Solidworks program for simulation environment and making the vehicle testable in simulation environment
Developing vehicle control algorithm	Kadir Ertuğ ACAR	16/11/2020 –	Reliable vehicle movement between lanes using the Pure-Pursuit control

		07/12/2020	algorithm
Designing the simulation environment	Mahsun BİNGÖL	16/11/2020 – 07/12/2020	Preparation of the appropriate test environment for the vehicle using the Carla, Gazebo or Unity simulator
Testing all algorithms in simulation environment	Özkan GÖKSU Kadir Ertuğ ACAR Mahsun BİNGÖL	07/12/2020 – 23/01/2021	Eliminating the problems encountered in the model vehicle by ensuring that algorithms work on the simulation systematically
Building the vehicle	Özkan GÖKSU Kadir Ertuğ ACAR Mahsun BİNGÖL	23/01/2020 – 07/02/2021	All electrical and mechanical construction of the vehicle
Testing the path planning algorithm	Mahsun BİNGÖL	25/01/2021 – 15/03/2021	Saving energy and time by finding the shortest route to the specified location by the vehicle
Developing Expanded Kalman Filter (EKF)	Kadir Ertuğ ACAR	25/01/2021 – 22/02/2021	More stable data acquisition with GPS-IMU and Lidar-IMU fusion
Setting up the user interface and communication system	Özkan GÖKSU	25/01/2021 – 22/02/2021	Control of instant information of the vehicle by observing the vehicle information by the user
Sensor Calibration	Özkan GÖKSU Kadir Ertuğ ACAR Mahsun BİNGÖL	08/03/2021 – 29/03/2021	Obtaining accurate data by ensuring that the sensors are adapted to the situation conditions
Testing all algorithms on the model vehicle	Özkan GÖKSU Kadir Ertuğ ACAR Mahsun BİNGÖL	29/03/2021 – 17/05/2021	Solving the problems encountered by the model vehicle while performing the specified tasks

4.4 Sensors

4.4.1 Camera

Line detection and traffic sign detection algorithms were trained at HD (1280x720) resolution. HD (1280x720) resolution was preferred instead of FHD (1920x1080) resolution in order not to exceed the processing capability of the “NVIDIA Jetson TX2” that artificial intelligence computer and to achieve high fps (frame per second) rates. In order to have a smooth image, a

camera was needed to minimum record 60 fps video in HD resolution.

Traffic signs have many different colors. It is important for the vehicle to recognize these signs with high accuracy in order to avoid danger in traffic. Monochrome cameras, even if they have grayscale, cannot provide high enough accuracy. However, this may be possible with RGB cameras. Therefore, the model vehicle needs a camera that can take color images.

In addition, high accuracy distance measurement is desired by making camera-lidar fusion. Thus, the mapping and localization accuracy rate will be increased. In order to achieve fusion, the camera must be able to measure its distance to the object it detects. This is only possible with cameras with depth feature. In addition, the selected lidar provides 10 m maximum measurement. For this reason, it is sufficient that the camera can measure 10 meters.

It is aimed to use the data taken from the camera in ROS over the ubuntu operating system. Therefore, its camera is expected to be compatible with ubuntu directly.

Table 4.4.1.1 Camera Technical Requirements

Resolution	**1280x720**
Frame per Second	60 fps
Color	RGB
Type	Stereo
Range	10 m
Supported Software	Ubuntu

So, Intel D435 depth camera was selected for the vehicle. The RGB camera, which can record 90 fps video at 1280x720 resolution, can be synchronized with depth data. It has approximately 10 meters maximum range. Also, intel D435 can easily transfer video over Ubuntu 16.04 or higher, thanks to its USB connection. [18]

Figure 4.4.1.1 Intel® RealSense™ Depth Camera D435 [18]

4.4.2 Lidar

A LIDAR sensor is an active optical ranging sensor. Some of the applications include the detection of obstacles, the detection of pedestrians and vehicles, the recognition of lanes and the determination of the vehicle's exact position. Using a laser, it emits light pulses at a high frequency.The reflected light is measured and used to calculate the distance between the sensor and the object if the emitted light hits the object. Since the laser can only measure a single small point at a time, in order to build a high-resolution depth image at the desired update rate, the LIDAR has to measure at a high frequency. Meanwhile, using rotating mirrors or by rotating the entire sensor unit, the laser beam is targeted. Typically, automotive LIDARs scan in the horizontal direction, which is carried out in layers. A larger number of layers allows the LIDAR to compensate for the pitch angle of the vehicle more easily and reduces the influence of occlusions. A LIDAR can have a range of over 250 meters, and the FOV can be as high as 360 degrees. LIDAR sensors can have a large FOV and range simultaneously, unlike RADAR.[19]

Drawbacks of lidars are that they are more costly and adversely affected by weather conditions than other sensors (rain, fog, hail, snow). Owing to high data flow, it requires very powerful computers.

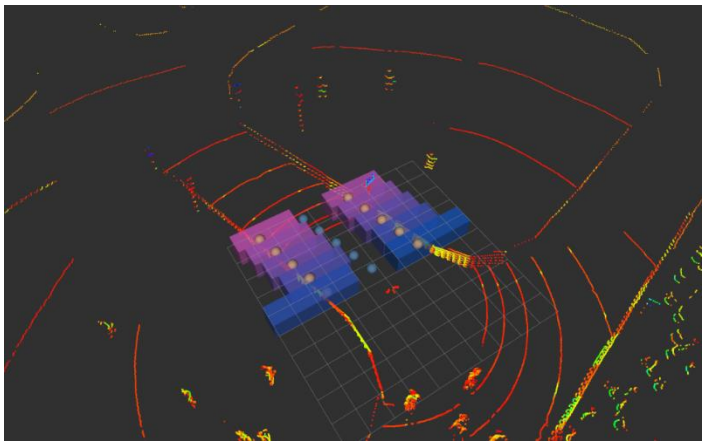


Figure 4.4.2.1 Lidar Point Cloud Visualization

The LIDAR generates a point cloud, where a single distance measurement is represented by every point. To extract object information, this point cloud has to be processed. Classification algorithms can be used to classify the detected objects. To detect an object reliably, it is necessary to have at least two lidar returns from that object.[20]

The vehicle needs to scan its entire environment in order to map and localize it. For this, one lidar positioned at the top of the vehicle should have 360 degrees rotation feature.

Although 3D lidars give a better quality and detailed image, there are very high price differences with 2 dimensional lidars.2D Lidar was chosen for this project as it has budget limits.

Considering that the lidar will be used on the model vehicle, it must be compatible with outdoor. In this way, the possibility of encountering various errors is reduced.

Since the vehicle software is based on ROS, lidar must be compatible with ROS.

Table 4.4.2.1 LIDAR Technical Requirements

Angular Range	**360 deg**
Dimension	2D
Outdoor	Yes
ROS compatible	Yes

When the market research was conducted, it was observed that RPLIDAR A3M1 was the only option suitable for the requirements. Although it is more expensive than similar lidars in terms

of price, it is the only lidar that provides safe reliable in outdoor. 2D RPLIDAR A3M1, which can rotate 360 degrees and has a distance of 10 meters, is mostly used in scale autonomous vehicles in the literature.

Figure 4.4.2.2 RPLIDAR A3M1 [20]

4.4.3 GPS (Global Positioning System)

GPS is the sensor used to determine the location of the vehicle on the earth. The GPS system determines its own global coordinates by receiving signals from satellites orbiting the earth. These coordinates are matched with the roadmap coordinates and the position of the vehicle on the road is determined.[21]

Within the scope of the project, the model vehicle must go to a specific location determined by the user so that the error rate is reduced below 30 cm. In this context, the location information of the vehicle on the world should be obtained globally with the help of GPS, which is planned to be placed on the model vehicle. In order to obtain global position information, a certain number of satellites must be connected.

The reason we chose the Ublox Neo-6M GPS module is that, despite its low cost, other GPS modules can connect to a maximum of 5 satellites, while itself can connect to 10 satellites.

Figure 4.4.3.1 UBLOX NEO-6M

A good high sensitivity GNSS receiver can acquire signals down to −155 dBm and tracking can be continued down to levels approaching −165 dBm. UBLOX NEO-6M boasts -147 dBm acquisition sensitivity from a cold start and -161 dBm tracking sensitivity.

In addition, it provides more precise results from the extended Kalman filter thanks to its 2.5 m positional accuracy and 0.1 m / s lessness accuracy.

4.4.4 IMU (Inertial Measurement Unit)

In this project, it is planned to use IMU together with GPS due to insufficient sensitivity of GPS systems (approximately 2.5 meters of error), negative effects of weather conditions, and the low number of satellites to which the GPS system can be purchased financially. The term IMU stands for "Inertial Measurement Unit" and describes a set of measurement tools.[22]

Data about the motion of the model vehicle can be recorded by the IMU. IMUs contain sensors such as accelerometers, gyroscopes, and magnetometers. IMUs can measure a variety of factors including velocity, direction, acceleration, special force, angular ratio, and magnetic fields surrounding the device in the presence of a magnetometer.

Each tool in an IMU is used to capture different types of data:

- Accelerometer: Measures speed and acceleration
- Gyroscope: Measures rotation and rotation speed
- Magnetometer: Determines the main direction (directional direction).

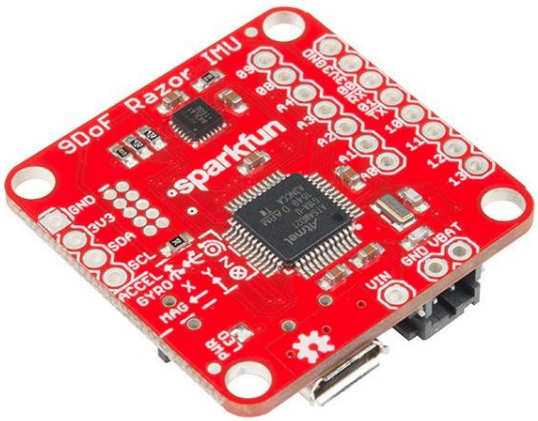


Figure 4.4.4.1 Sparkfun 9DoF Razor IMU M0

4.5 Lane Detection

4.5.1 Image Thresholding

The concept behind this step is to create a pipeline for image processing where the algorithm can clearly identify the lane lines. By playing around with distinct gradients, thresholds and color spaces, there are a number of different ways to get to the solution. With a number of these techniques are experimented on several different images and a combination of thresholds, color spaces, and gradients are used. The thresholds for the absolute value of the x and y gradients, the magnitude of the gradients at each point, the angle formed by the x and y gradients, the LUV threshold, the HLS color space saturation channel, and the HSV color space value channel were combined to provide the example below with the threshold image.

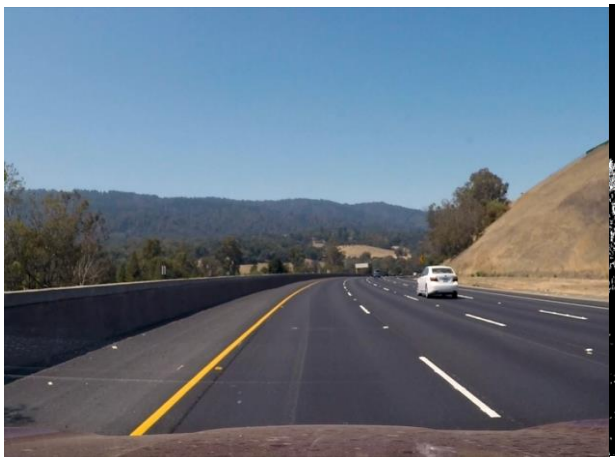


Figure 4.5.1.1: Original Image

Figure 4.5.1.2 Abs Sobel Threshold

Magnitude threshold apples Sobel in both x & y with a given kernel size and threshold range for magnitude masking. Direction threshold applies Sobel in both x & y with a given kernel size and threshold range for directional masking.

Figure 4.5.1.3 Magnitude Threshold

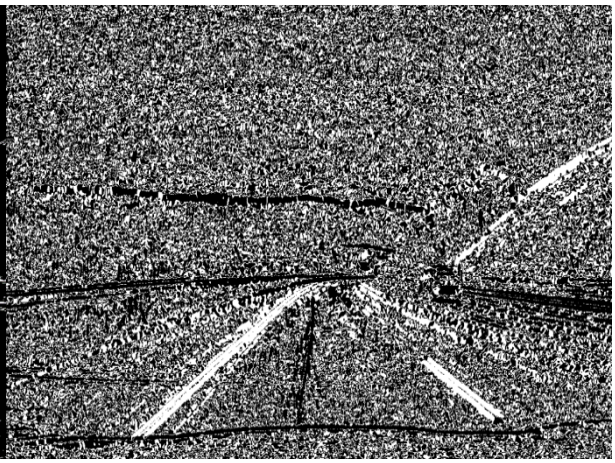


Figure 4.5.1.4 Direction Threshold

HLS and LUV Threshold converts the image to requested scale and applies threshold to desired channel with the range provided.

Figure 4.5.1.5 HLS S-Channel Threshold

Figure 4.5.1.6 HLS L-Channel Threshold

Figure 4.5.1.7 LUV Threshold **Figure 4.5.1.8 Combined Gradient**

4.5.2 Perspective Transform

Images have a perspective that makes lanes appear to be converging at a distance in an image, even though they are parallel to each other. When this perspective is removed, the curvature of the lane lines is easier to detect. By transforming the image into a 2D Bird's eye view, where the lane lines are always parallel to each other, this can be achieved. Since we are only interested in the lane lines, four points are selected on the original un-distorted image using the thresholded image (combined), as well as source (src) and destination (dst) points, and the perspective is transformed into a Bird's eye view. [23]

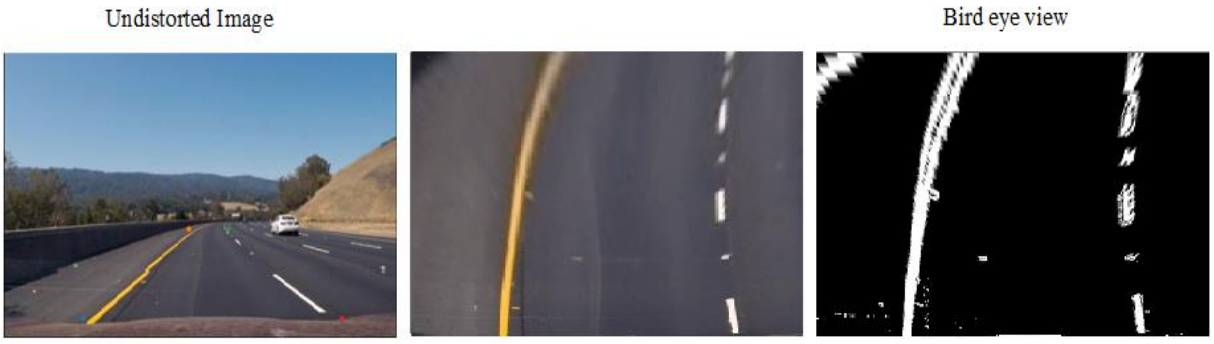


Figure 4.5.2.1 Region of interest perspective warped to generate a Bird’s-eye view

4.5.3 Detect Lane Pixels

Once the input image has been pre-processed, the next step is to find and map the lanes in the

image area. The approach would be to draw a nonzero Pixel histogram in the bottom half of the binary image (threshold image) to observe the following pattern:

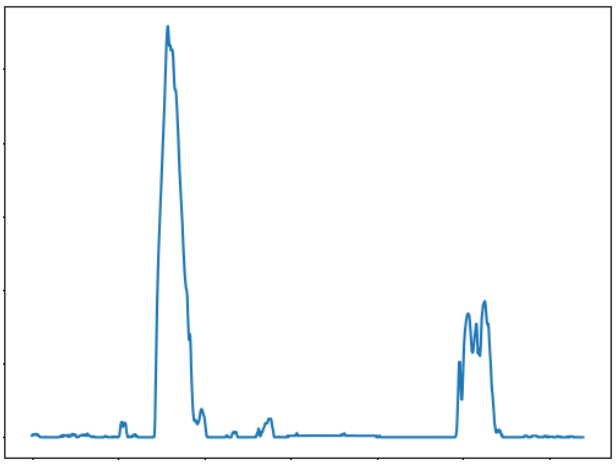


Figure 4.5.3.1 Histogram

Since pixel values are now binary, peaks can be a good indicator of lane lines and represent where most non-zero pixels are located. The x coordinates in the histogram thus serve as a starting point for related lanes to be searched. The concept of sliding windows approach, which is a window with a margin placed around the center of the line, will actually apply here. [24]

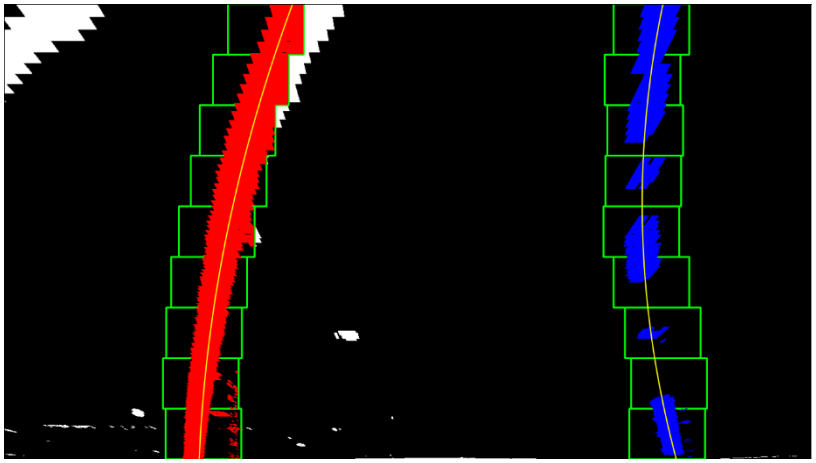


Figure 4.5.3.2 Sliding window fit results

4.5.4 Determine the Curvature

The sliding windows help to estimate the center of each lane area, so a second-order polynomial curve will suit using these x and y pixel positions. Since the lanes are vertical in the picture, the function suits with f(y) instead of f(x). Figure 4.5.4.1 is a good illustration of this step.

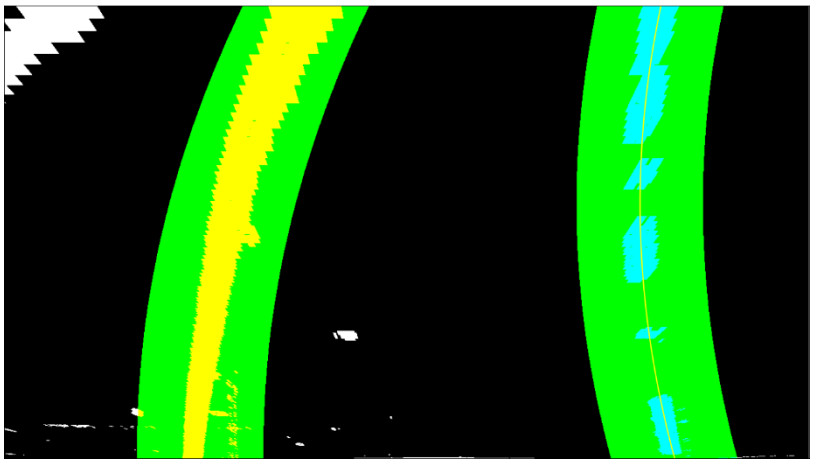


Figure 4.5.4.1 Identified lane lines

4.5.5 Back onto the Original Image

By inserting a polygon between these points and projecting it back to the original picture, the final step would be to highlight the lane region .In the image area, the lane area and curvature radius have been measured on the basis of pixel values that are not the same as the real world area, so they need to be translated to real world values, which means measuring the length and width of the lane segment where our distorted image is projected. [24]

Figure 4.5.5.1 Processed Image

4.6 Object Detection

There are more than one YOLO versions. YOLOv1 has been first released

in 2016 and it was epochal because of its detection capability and accuracy in real time. 1 year later YOLOv2 is released. It can process images at 40-90 FPS. It is faster than v1. In 2018, Joseph Redmon release his last version of YOLO. YOLOv3 allow us to tradeoff between speed and accuracy easily. Redmon's YOLO is based on Darknet that is open source neural network framework written in C. Darknet provide us GPU support because GPU is 5 times faster than CPU in image processing. Unfortunately, Redmon stopped his support to improve YOLO in early 2020. After a short time, Alexey Bochkovskiy announced YOLOv4 which is still based on Redmon's research. YOLOv4 is more accurate and faster than YOLOv3 by 10 percent and 12 percent respectively. There are two versions of YOLO that are YOLOv5 and PP-YOLO. However, they introduced newer and there is not too much documentation about them. Also, YOLOv5 use Pytorch instead of Darknet. [25] [26] [27]

Model	Train	Test	mAP	FPS
YOLO	VOC 2007+2012	2007	63.4	45
Fast YOLO	VOC 2007+2012	2007	52.7	155
YOLOv2	VOC 2007+2012	2007	76.8	67
YOLOv2 544x544	VOC 2007+2012	2007	78.6	40
Tiny YOLO	VOC 2007+2012	2007	57.1	207
YOLOv2 608x608	COCO trainval	test-dev	48.1	40
Tiny YOLO	COCO trainval	-	-	200
YOLOv3-320	COCO trainval	test-dev	51.5	45
YOLOv3-416	COCO trainval	test-dev	55.3	35
YOLOv3-608	COCO trainval	test-dev	57.9	20
YOLOv3-tiny	COCO trainval	test-dev	33.1	220
YOLOv3-spp	COCO trainval	test-dev	60.6	20

Figure 4.6.1 YOLO Version Performance[28]

In test computer has a "NVDIA GTX 1660 Ti" graphic card. It has nearly same compute capability with vehicle computer Jetson TX2. YOLOv3, YOLOv4 and YOLOv3-Tiny are tested with same dataset and the most sufficient results coming from YOLOv3-Tiny. It is working with 35 FPS which is enough to work with real time application. Also, its accuracy still good.

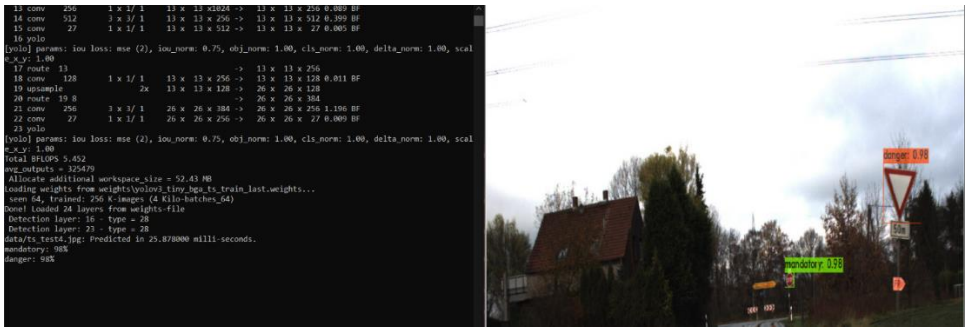


Figure 4.6.2 YOLOv3 Test

ROS is unifying element in vehicle software architecture. All subsystems related with software are connected to ROS and they have unique tasks. Object detection must inform system about traffic signs. STOP, YIELD and NO-ENTRY signs are selected. A 1x3 matrix created to carry condition of traffic signs. For example [1 1 1] indicates all traffic signs appear in camera. To do that, Darknet source code manipulated and bga.cpp added to inside relevant part of it.

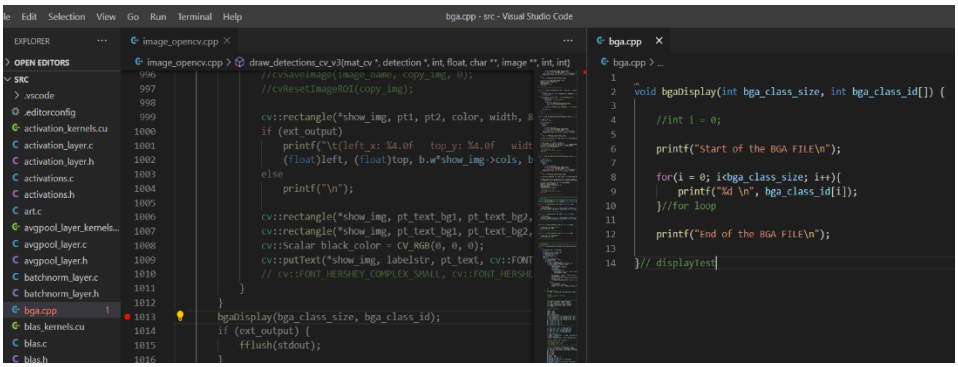


Figure 4.6.3 Darknet Source Code Manipulated

4.7 Simulation Environment and Vehicle Selection for Mapping

To do mapping in a simulation environment, it must be a mobile vehicle and track. The Jackal Unmanned Ground Vehicle of the Clearpath Company was selected for the mobile vehicle. “Jackal is a small, fast, entry-level field robotics research platform. It has an onboard computer, GPS and IMU fully integrated with ROS for out-of-the-box autonomous capability. As with all Clearpath robots, Jackal is plug-and-play compatible with a huge list of robot accessories to quickly expand your research and development.” [29]

The reasons for choosing this tool are; the vehicle has similar hardware to the vehicle intended to be designed in the project, the algorithms and ROS packages related to the vehicle are open source and can easily transfer the vehicle to the simulation environment. Velodyne VLP-16 attached to the vehicle was used in simulation environment with Lidar.

Figure 4.7.1 Jackal Ground Vehicle

After the vehicle selection, Gazebo was chosen as the simulation environment for its easy interface and ease of use integrated with ROS.

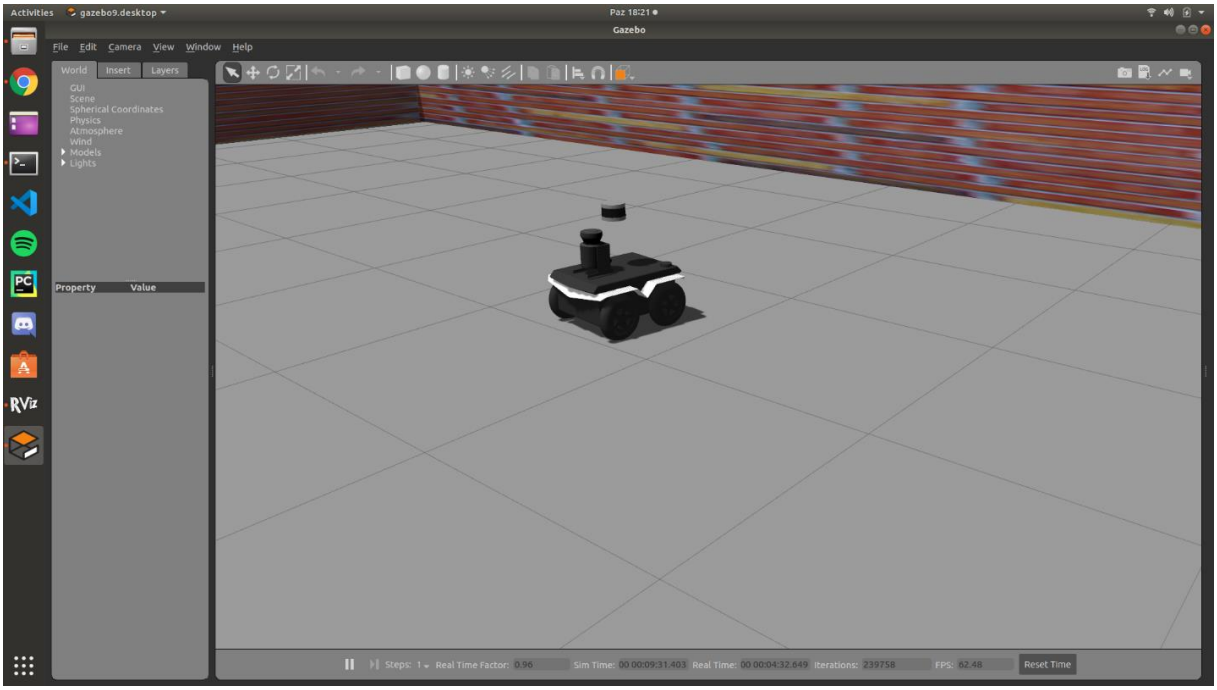


Figure 4.7.2 Jackal Ground Vehicle in Gazebo

4.7.1 Designing Simulation Environment

While designing the simulation environment, the physical dimensions of the vehicle and the track design were taken into consideration. Road and cornering dimensions were determined in accordance with the physical dimensions of the vehicle. While preparing the track, the "TEKNOFEST Robotaksi Binek Otonom Araç ", which is the only autonomous vehicle competition organized in Turkey, was taken as a basis. A track similar to the competition track has been created.

Figure 4.7.1.1 "TEKNOFEST Robotaksi Binek Otonom Araç" Competition Track

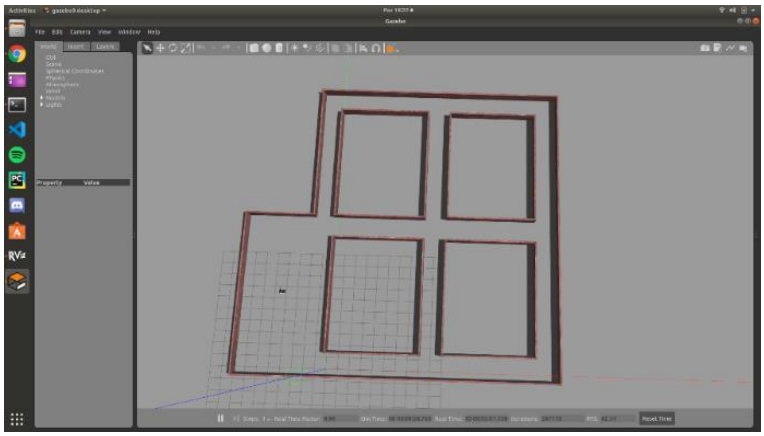


Figure 4.7.1.2 Designed track in Gazebo

4.7.2 ROS Packages

ROS package called jackal_velodyne was used for mapping and routing [30]. This package contains the .urdf file containing the sensor hardware of the Jackal vehicle, as well as the physical features, and the ROS launch files required for vehicle control. The ROS package

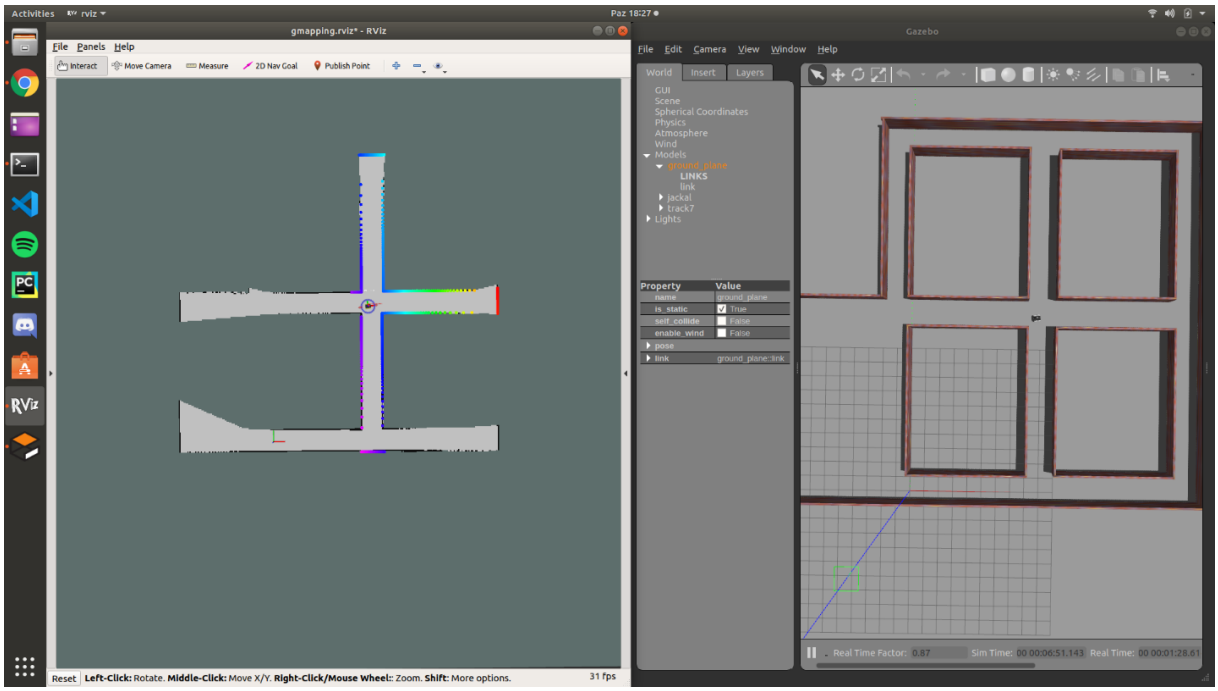
containing the features of the Jackal tool is open source and taken from the Jackal official Github page [31].

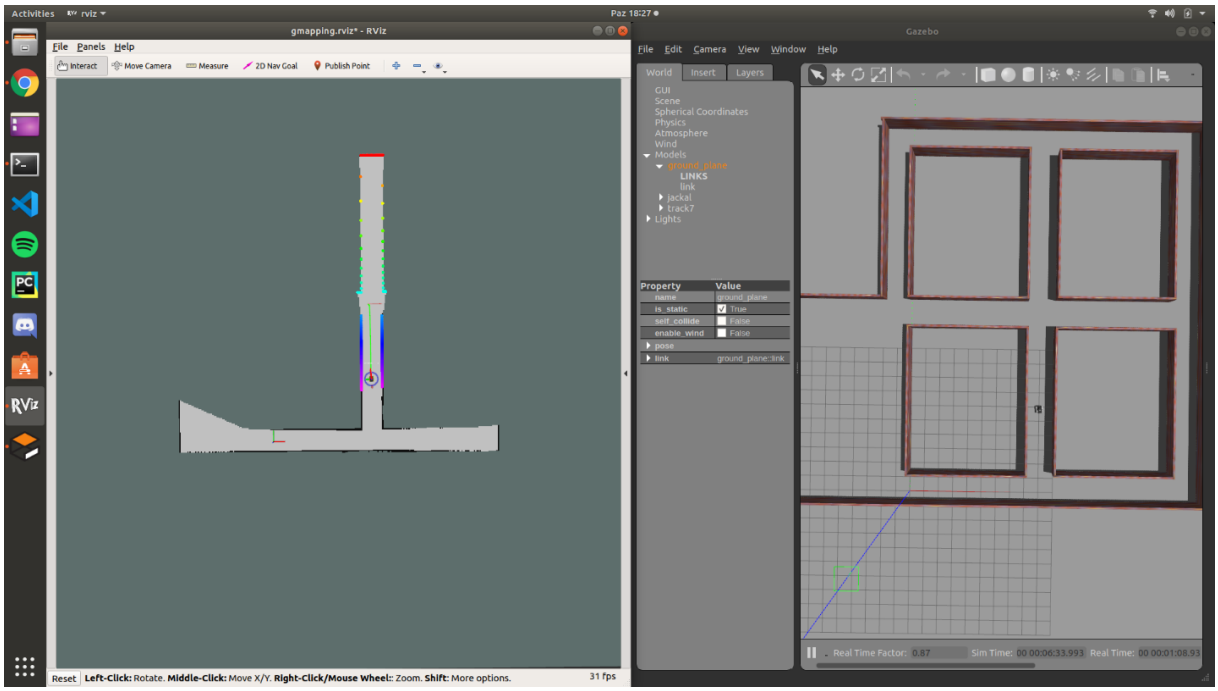
The gmapping_demo.launch file included in the ROS package has been arranged according to the track designed in the Gazebo environment. The obstacle avoidance distance of the vehicle is set to a minimum of 30 cm.

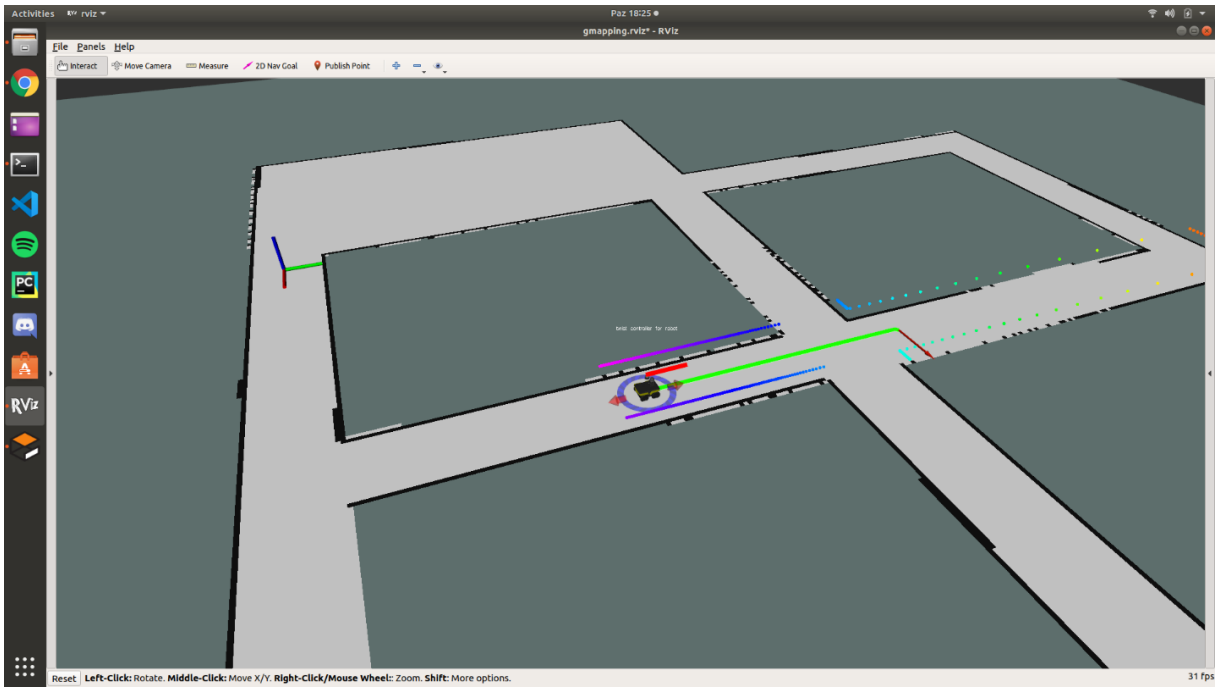
The package includes OpenSlam's Gmapping as SLAM algorithm. Detailed explanation of the algorithm on the OpenSlam website “Recently Rao-Blackwellized particle filters have been introduced as effective means to solve the simultaneous localization and mapping (SLAM) problem. This approach uses a particle filter in which each particle carries an individual map of the environment. Accordingly, a key question is how to reduce the number of particles. We present adaptive techniques to reduce the number of particles in a Rao- Blackwellized particle filter for learning grid maps. We propose an approach to compute an accurate proposal distribution taking into account not only the movement of the robot but also the most recent observation. This drastically decrease the uncertainty about the robot's pose in the prediction step of the filter. Furthermore, we apply an approach to selectively carry out re-sampling operations which seriously reduces the problem of particle depletion”. [32] [33]

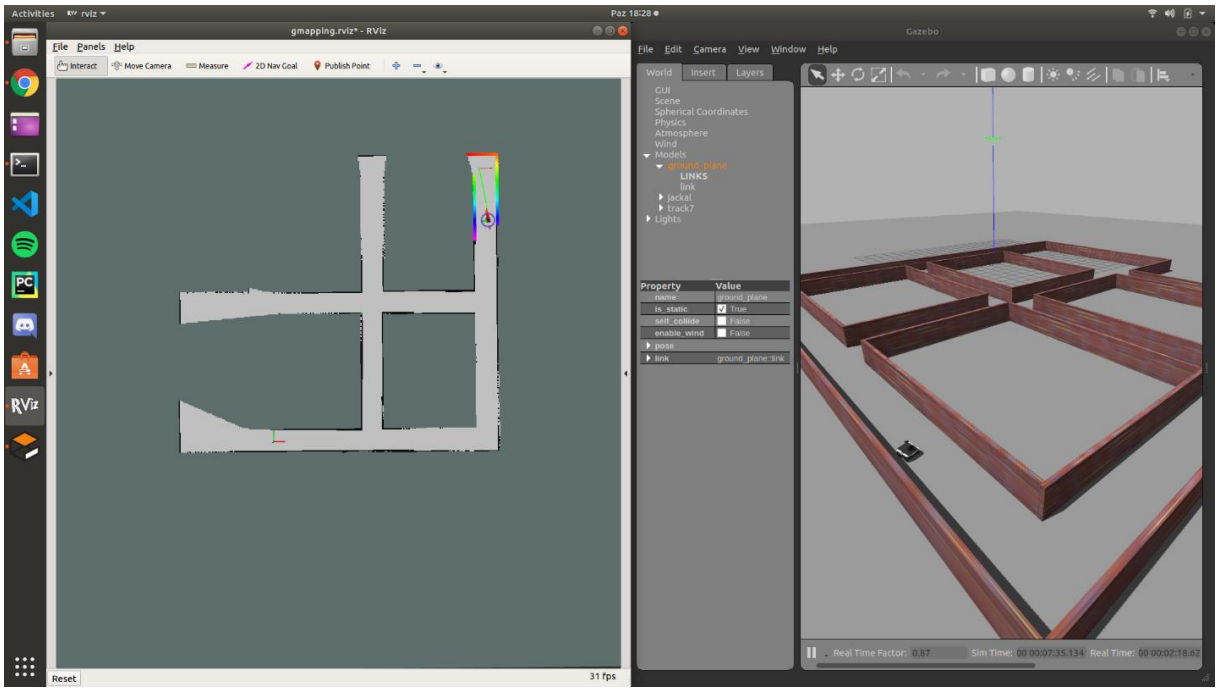
With the Gmapping SLAM algorithm used, the data received from Lidar is converted into / laser_scan message type. From here, the position of the vehicle to the objects around it reaches. Then, if there is a gap between the tool and the object, that area is painted gray. The object's closest indent to the tool is painted in black.

The following pictures show examples of mapping made with the Jackal tool in the track designed in the Gazebo environment using the Gmapping SLAM algorithm. Rviz interface is used to display ROS topic information.


Activities
rviz
Paz 18:27
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Gazebo
File Edit Camera View Window Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
LINKS
link
jackal
track7
Lights
Property
Value
name
ground_plane
is_static
True
self_collide
False
enable_wind
False
pose
link
ground_plane::link
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps
Real Time Factor: 0.87
Sim Time: 00 00:06:51.143
Real Time: 00 00:01:28.61


Activities
rviz
Paz 18:27
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Gazebo
File Edit Camera View Window Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
LINKS
link
jackal
track7
Lights
Property
Value
name
ground_plane
is_static
True
self_collide
False
enable_wind
False
pose
link
ground_plane::link
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps
Real Time Factor: 0.87
Sim Time: 00 00:06:33.993
Real Time: 00 00:01:08.93


Activities
rviz
Paz 18:25
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps


Activities
rviz
Paz 18:28
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Reset
31 fps
Gazebo
File Edit Camera View Window Help
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
LINKS
link
jackal
track7
Lights
Property
Value
name
ground_plane
is_static
True
self_collide
False
enable_wind
False
pose
link
ground_plane::link
Real Time Factor: 0.87
Sim Time: 00 00:07:35.134
Real Time: 00 00:02:18.62

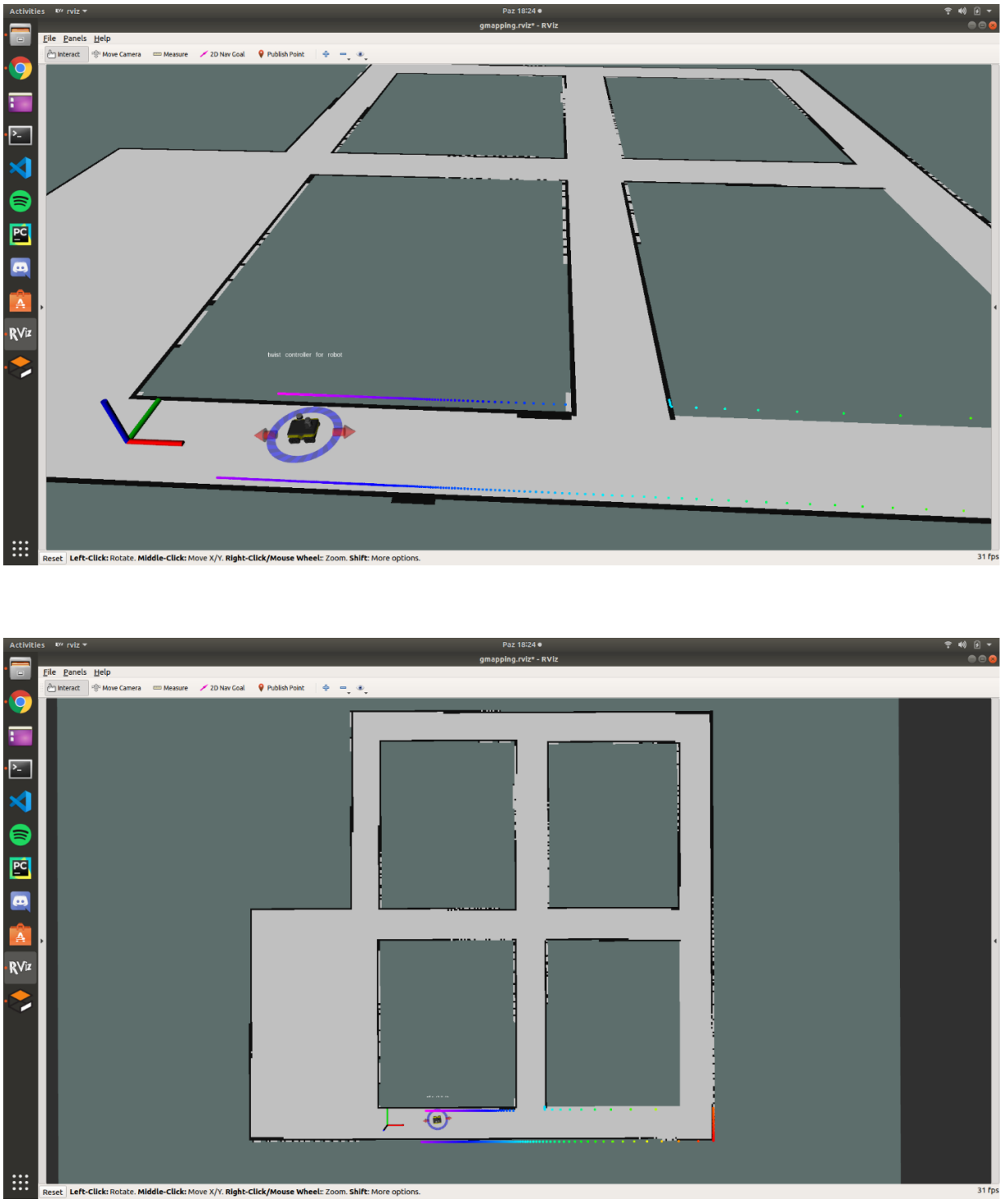


Figure 4.7.2.1 Final map of the track

The move_base ROS package is used for the path planning algorithm. This package aims to combine the local position of the vehicle with its global position. Jackal uses the GPS data contained in the vehicle and the vehicle's odometry data. The IMU provides the odometry data. The package provides the shortest path to a specified location. The green line in front of the

vehicle in the pictures above shows the safe path drawn. In the picture below, the "position x, y and z" values in the terminal are shown at the bottom left and the instantaneous global coordinate data taken during the movement of the vehicle. Data is taken from "/ move_base / TrajectoryPlannerROS / global_plan" in ROS.

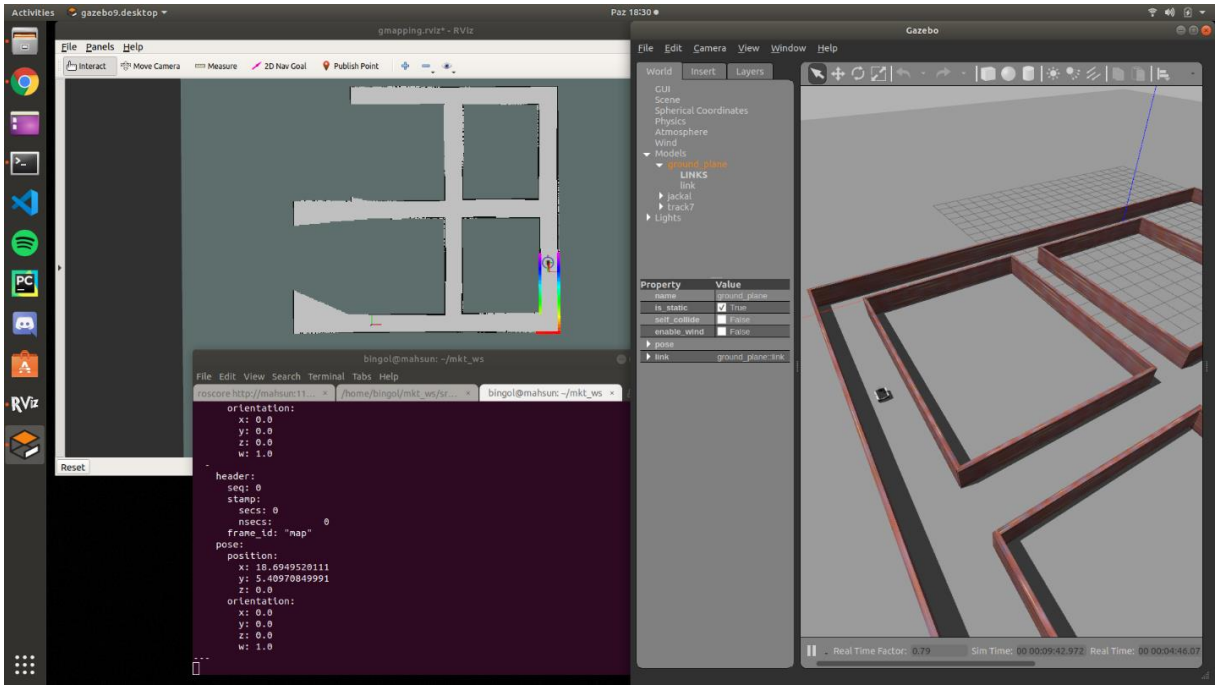
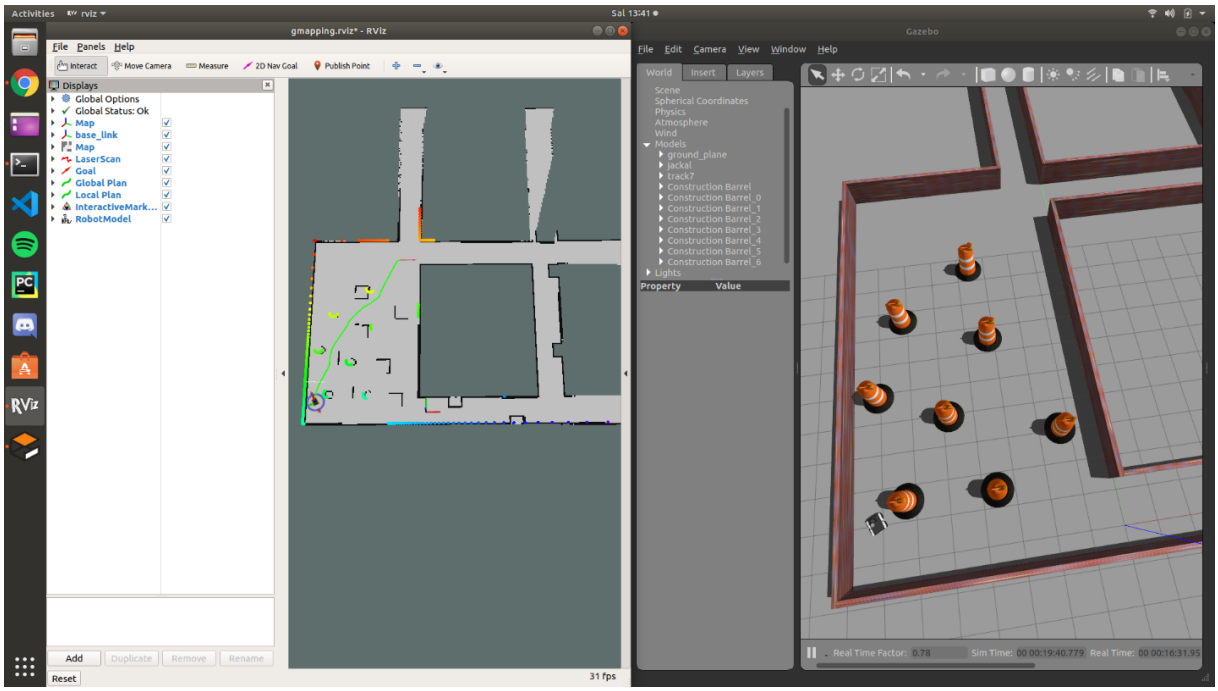
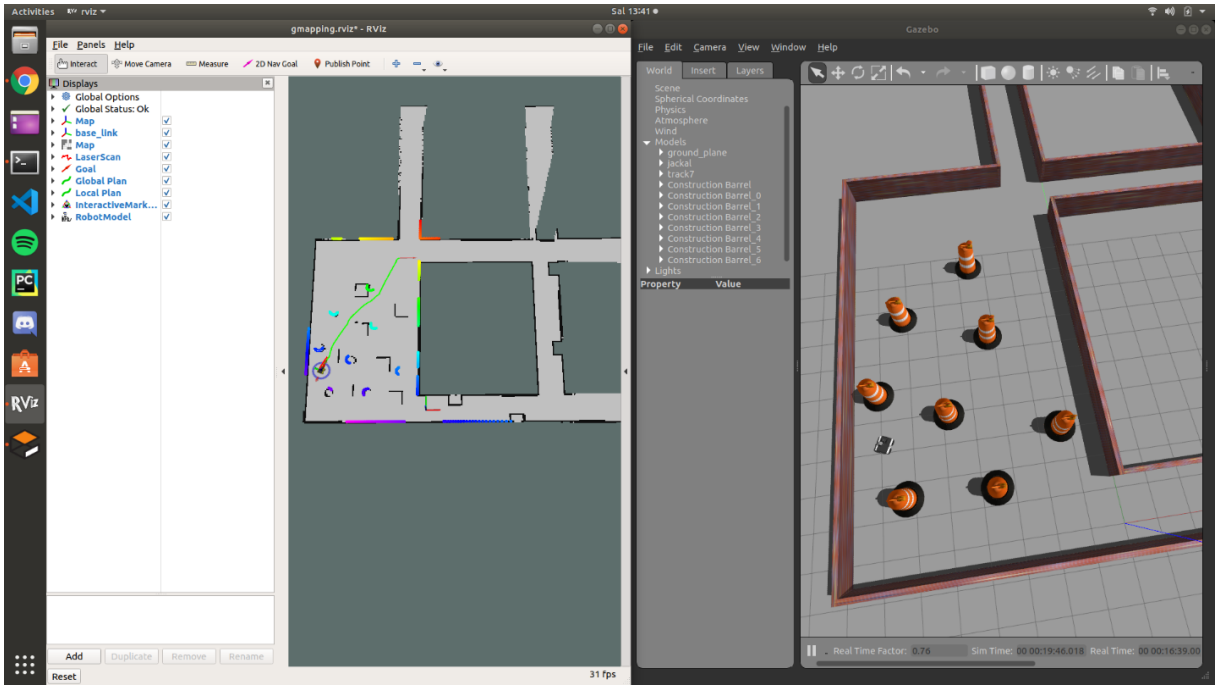


Figure 4.7.2.2 Global X, Y and Z coordinate values shown in the terminal

4.7.3 Collision Avoidance

It is mentioned that the package used was arranged to avoid obstacles and leave a minimum distance of 30 cm. This section will show the results of the obstacle avoidance test. Traffic barriers are placed in front of the vehicle. Then a location to go was determined. The vehicle has to reach the position without escaping the barriers. The steps of this test are shown in the pictures below.


Activities
rviz
Sal 13:41
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Displays
Global Options
Global Status: Ok
Map
base_link
Map
LaserScan
Goal
Global Plan
Local Plan
InteractiveMark...
RobotModel
Add
Duplicate
Remove
Rename
Reset
31 fps
Gazebo
File Edit Camera View Window Help
World
Insert
Layers
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
jackal
track7
Construction Barrel
Construction Barrel_0
Construction Barrel_1
Construction Barrel_2
Construction Barrel_3
Construction Barrel_4
Construction Barrel_5
Construction Barrel_6
Lights
Property
Value
Real Time Factor: 0.78
Sim Time: 00 00:19:40.779
Real Time: 00 00:16:31.95


Activities
rviz
Sal 13:41
gmapping.rviz* - RViz
File Panels Help
Interact
Move Camera
Measure
2D Nav Goal
Publish Point
Displays
Global Options
Global Status: Ok
Map
base_link
Map
LaserScan
Goal
Global Plan
Local Plan
InteractiveMark...
RobotModel
Add
Duplicate
Remove
Rename
Reset
31 fps
Gazebo
File Edit Camera View Window Help
World
Insert
Layers
Scene
Spherical Coordinates
Physics
Atmosphere
Wind
Models
ground_plane
jackal
track7
Construction Barrel
Construction Barrel_0
Construction Barrel_1
Construction Barrel_2
Construction Barrel_3
Construction Barrel_4
Construction Barrel_5
Construction Barrel_6
Lights
Property
Value
Real Time Factor: 0.76
Sim Time: 00 00:19:46.018
Real Time: 00 00:16:39.00

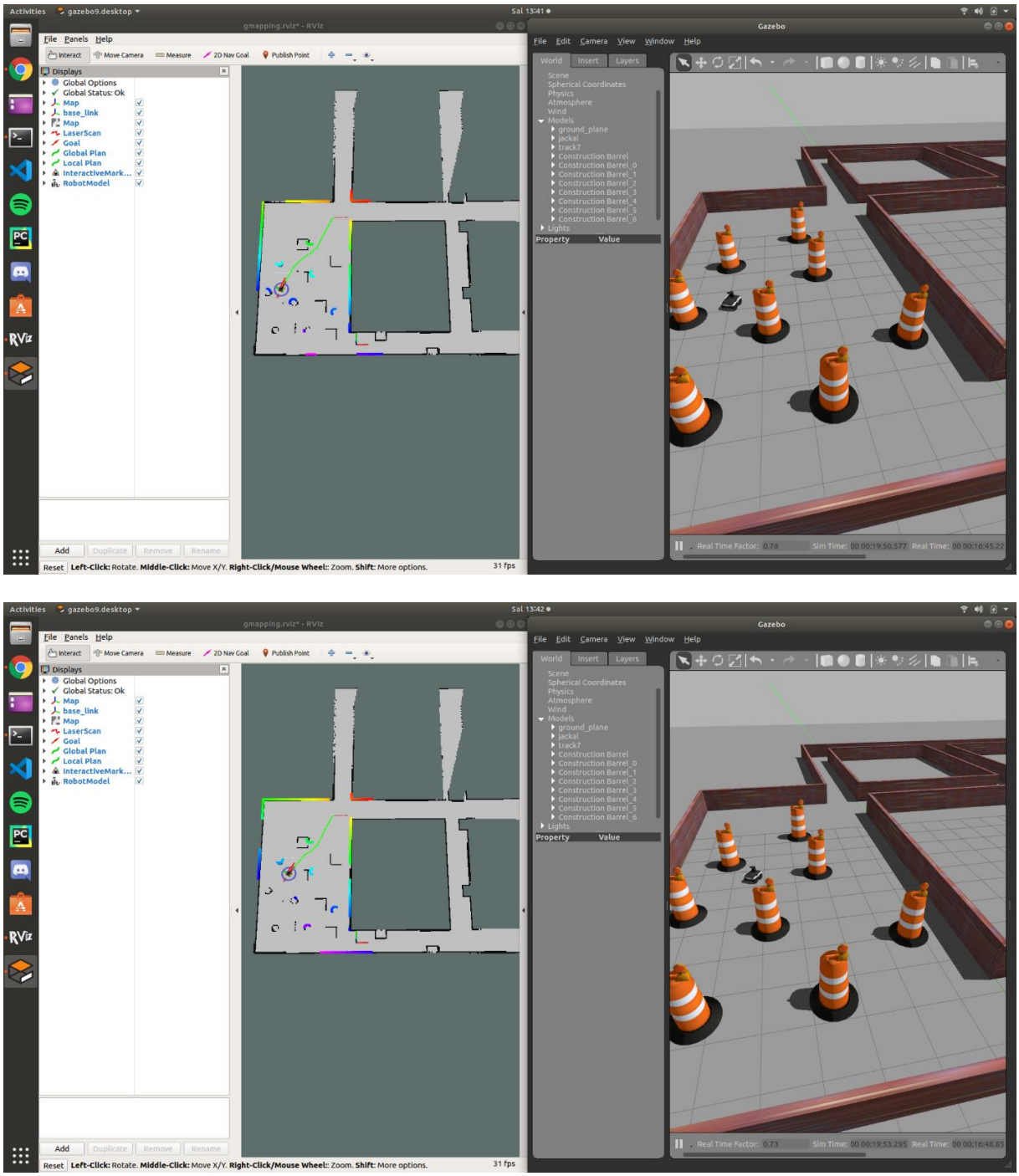


Figure 4.7.3.1 Collison Avoidance with Jackal in Gazebo

4.7.4 ROS Rqt Graph

Rqt_graph provides a GUI plugin for visualizing the ROS calculation chart. The mapping and routing tasks performed below are showing the ROS system while working with Jackal.

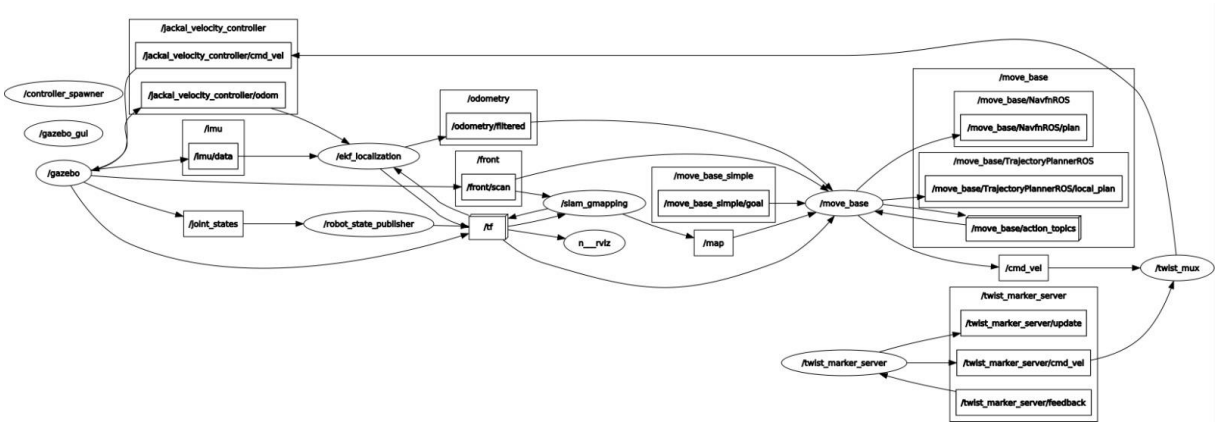


Figure 4.7.4.1 ROS rqt graph

The rectangles in the picture above are ROS topics. Those written in the ellipse are ROS Nodes. The figure shows the communication between the vehicle's node and topics.

4.8 Simulink Model and Results of Pure Pursuit Control

In this section, information will be given about the model we designed for mobile vehicle in MATLAB / Simulink environment and the outputs of the model. The vehicle information in the model has been designed according to the physical characteristics of the Traxxas Bigfoot vehicle. The Vehicle Body 3DOF block of the Vehicle Dynamics Blockset / Vehicle Body library is used in the model. Vehicle parameters are entered in this block. Robotics System Toolbox / Mobile Robot Algorithms / Navigation Toolbox / Control Algorithms library's Pure Pursuit block was used, and a control algorithm was created. In addition, 2D Visualization block was used to observe the vehicle's route and vehicle movement.

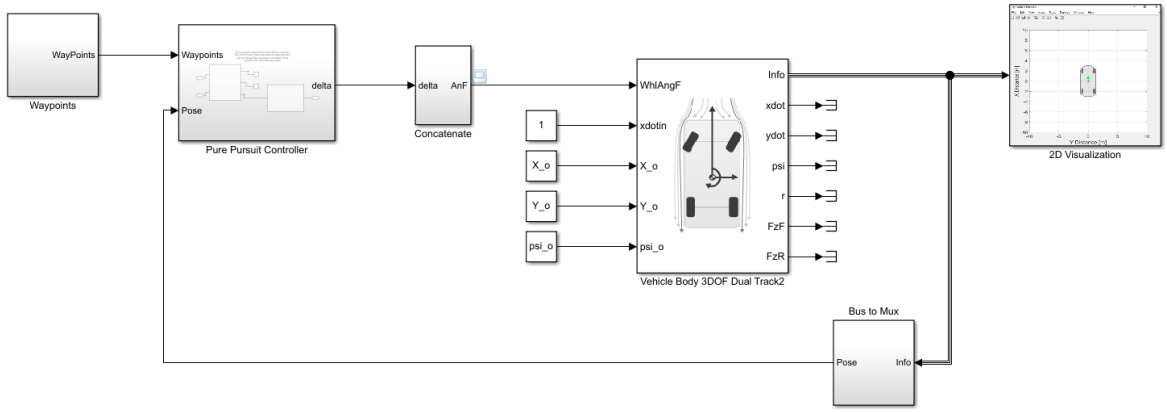


Figure 4.7.4.1 Simulink Model of Pure Pursuit Control

```
%% add Image to the path
addpath(genpath('Images'));

%% load the scene data file generated from Driving Scenario Designer
load('test7.mat');

%% define reference points
refPose = data.ActorSpecifications.Waypoints;
xRef = refPose(:,1);
yRef = -refPose(:,2);

%% define reference time for plotting
Ts = 150; % simulation time
s = size(xRef);
tRef = (linspace(0, Ts , s(1)))';% this time variable is used in the "2D Visualization"
                                 % block for plotting the reference points.

%% define parameters used in the models
L = 0.27; % bicycle length
ld = 1; % lookahead distance
X_o = refPose(1,1); % initial vehicle position
Y_o = -refPose(1,2); % initial vehicle position
psi_o = 0; % initial yaw angle
```

Figure 4.7.4.2 Code of Pure Pursuit Control

In the code above, vehicle length and look-ahead distance are determined. The road has been drawn and waypoints have been assigned in the Driving Scenario Designer Toolbox. In addition, instant location information of the vehicle was received.

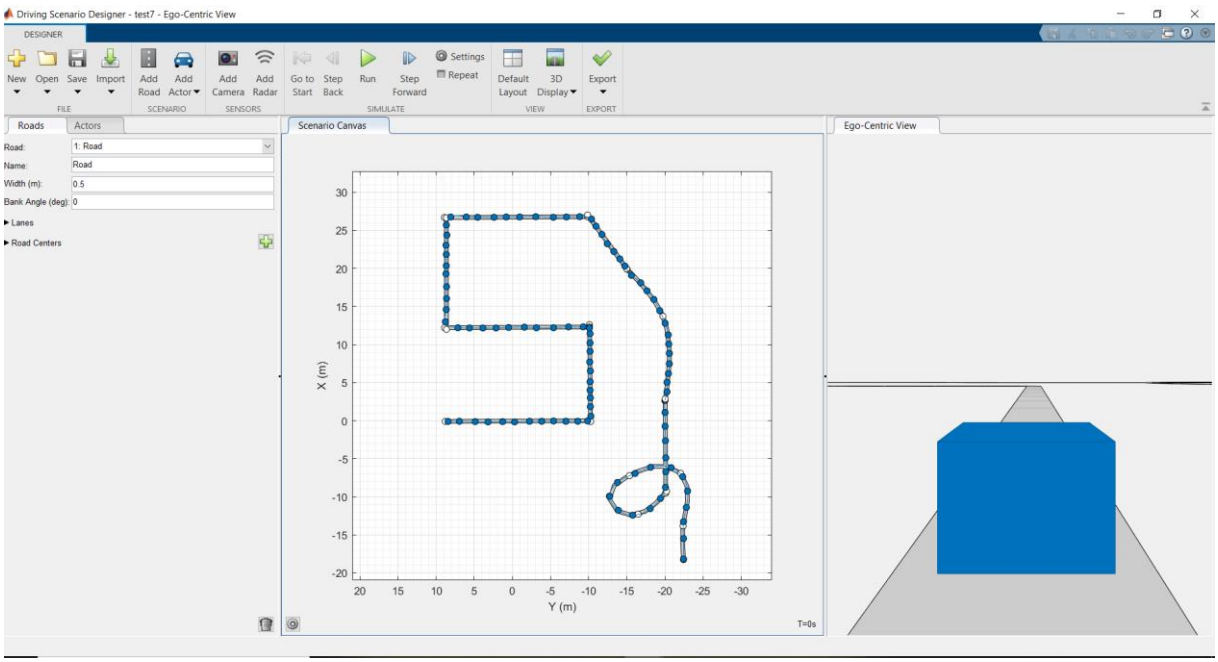
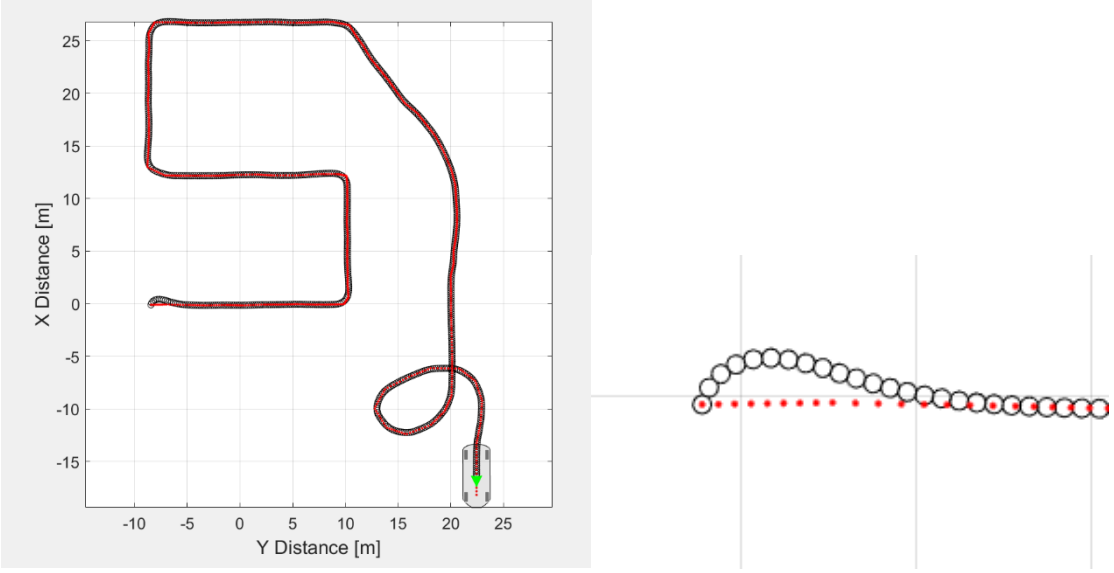


Figure 4.7.4.3 Desired road and waypoints

The road turns both right and left. It is also drawn to include a full turn and a long turn. The purpose of this is to evaluate all the movements of the vehicle on the possible track. Simulation is made according to different speed and look-ahead distance values.



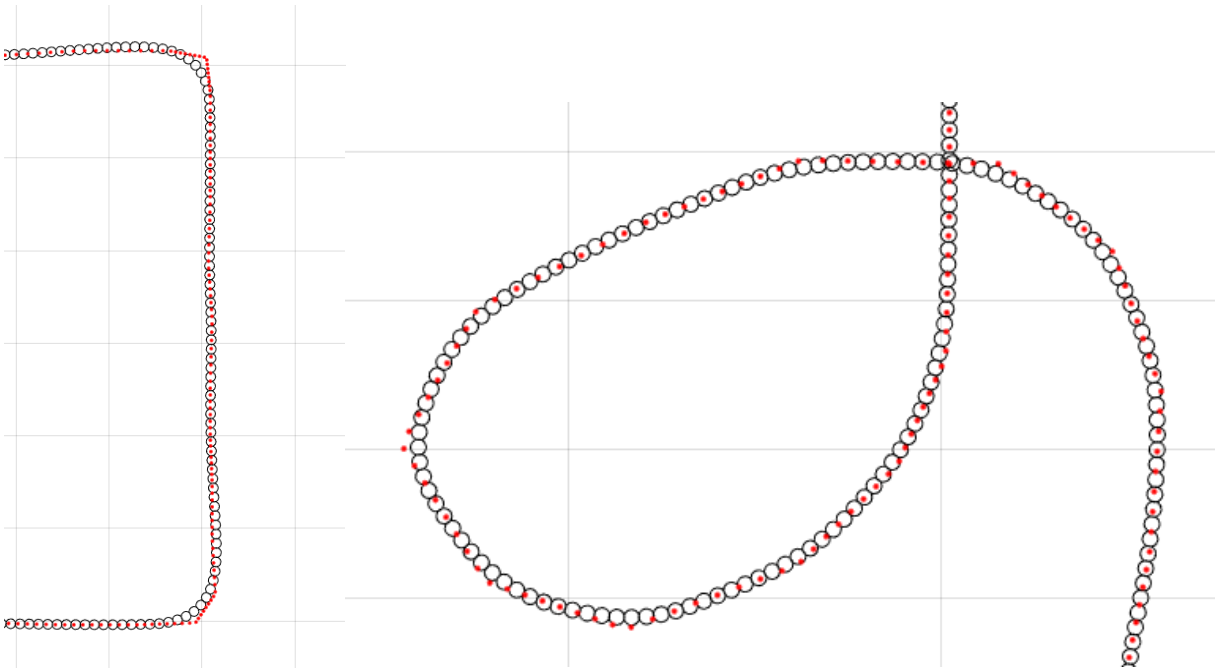


Figure 4.7.4.4 Simulation Result with V = 1 ℓ_d = 1

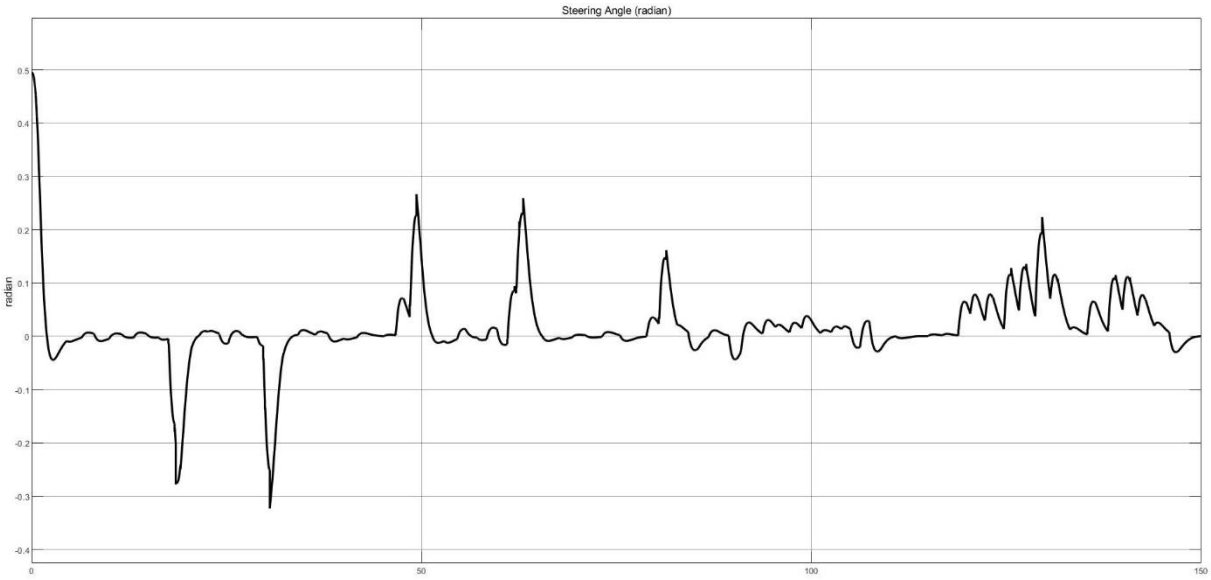


Figure 4.7.4.5 Steering Angle with V = 1 ℓ_d = 1

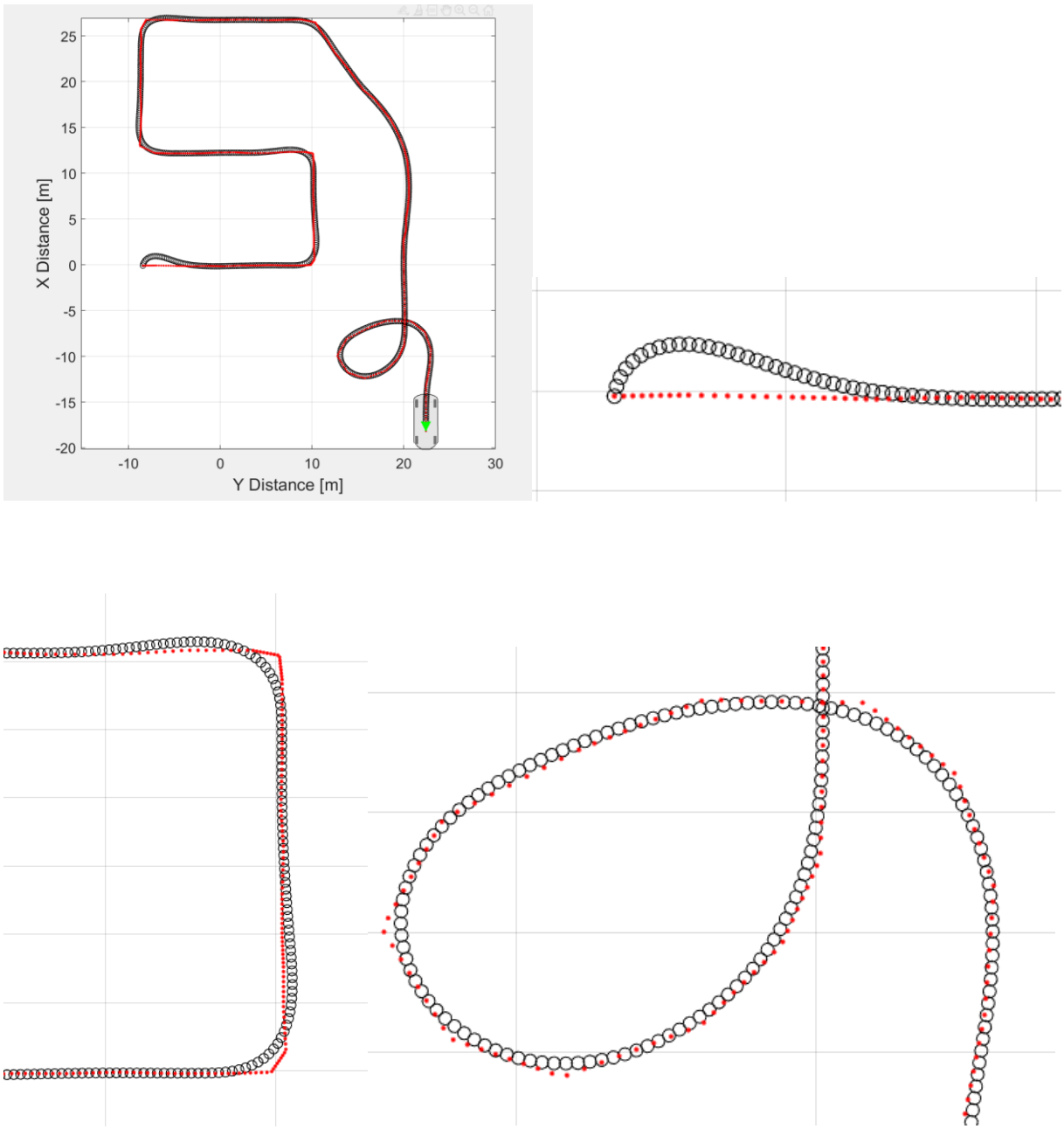


Figure 4.7.4.6 Simulation Result with V = 1 ℓ_d = 1

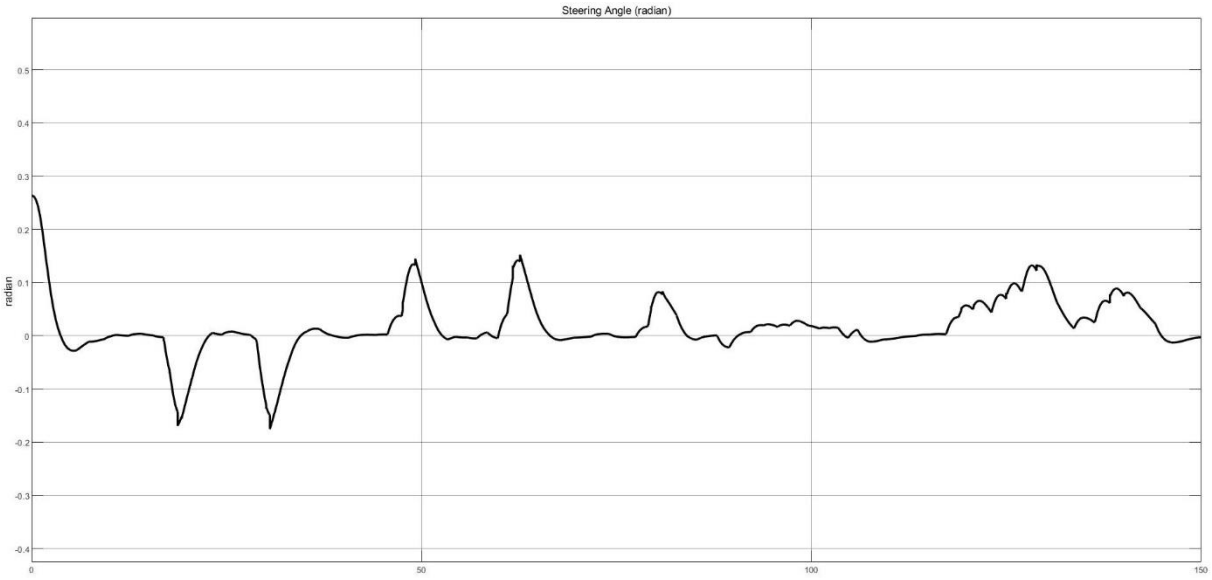
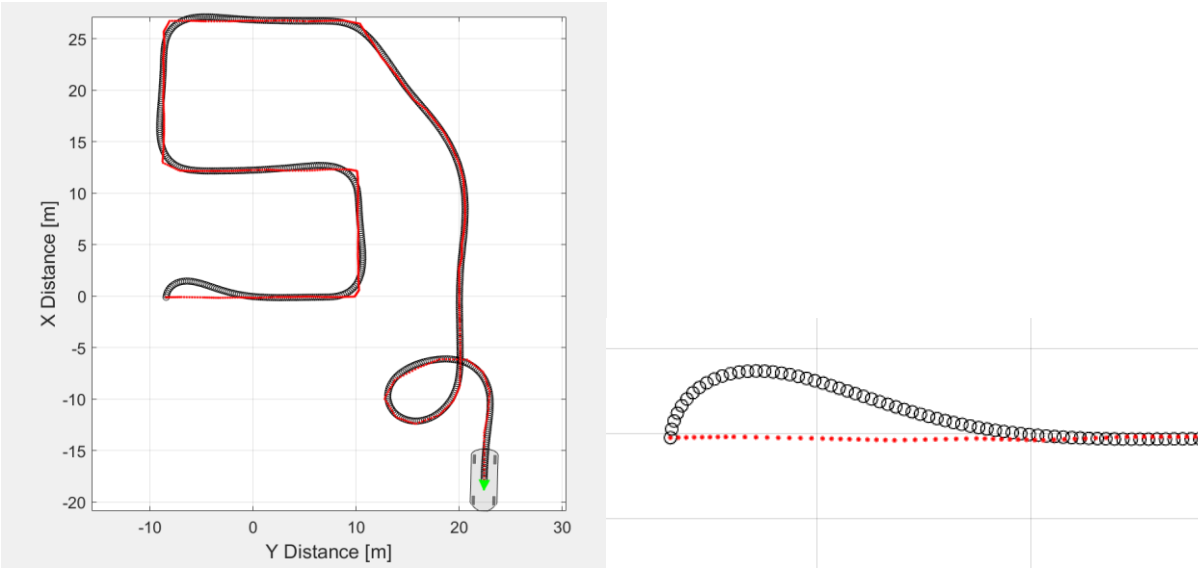


Figure 4.7.4.7 Steering Angle with V = 1 ℓ_d = 1



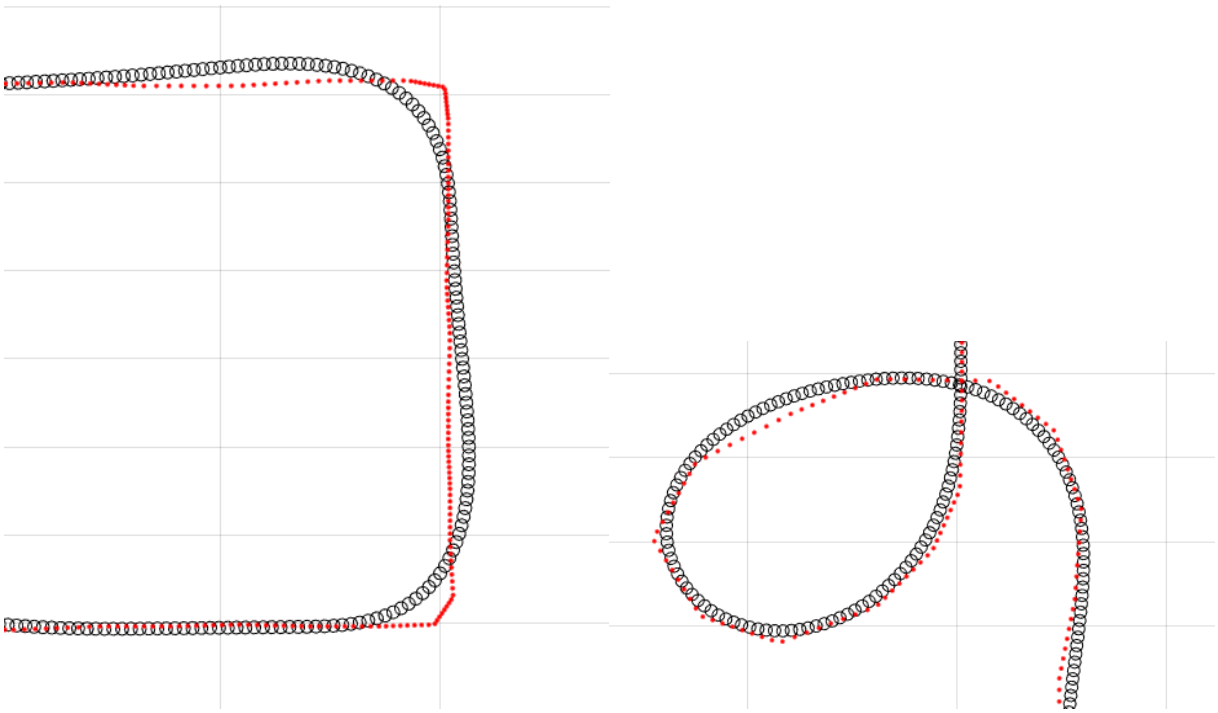


Figure 4.7.4.8 Simulation Result with V = 1 ℓ_d = 3

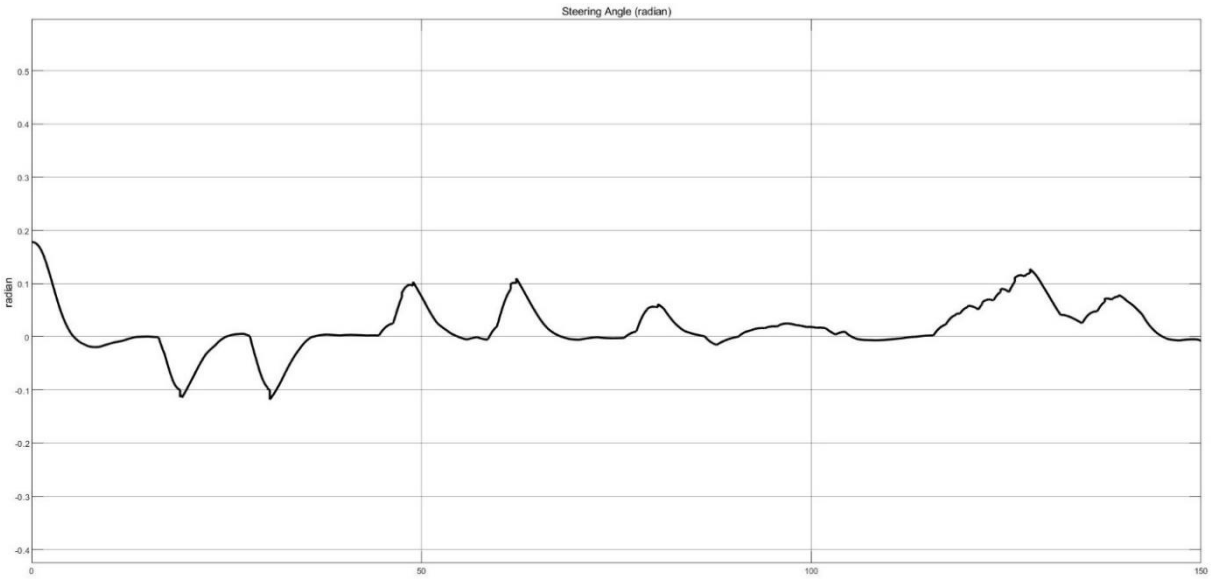


Figure 4.7.4.9 Steering Angle with V = 1 ℓ_d = 3

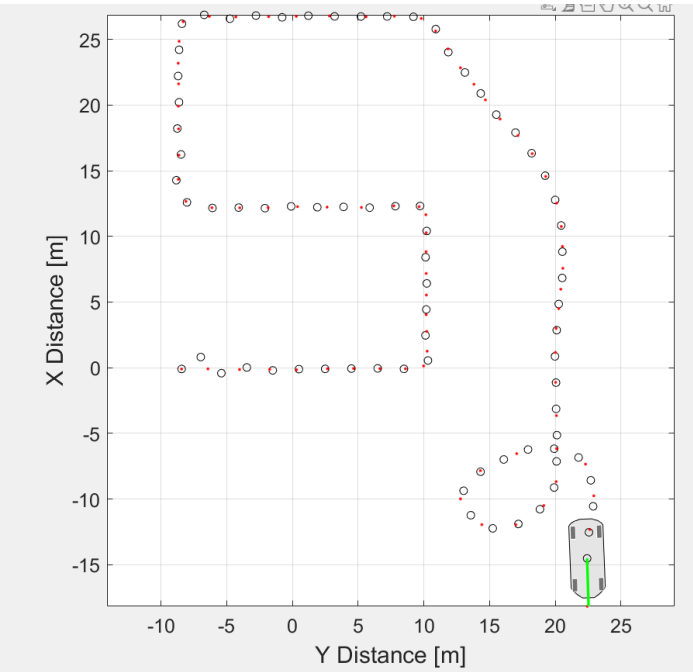


Figure 4.7.4.10 Simulation Result with V = 10 ℓ_d = 1

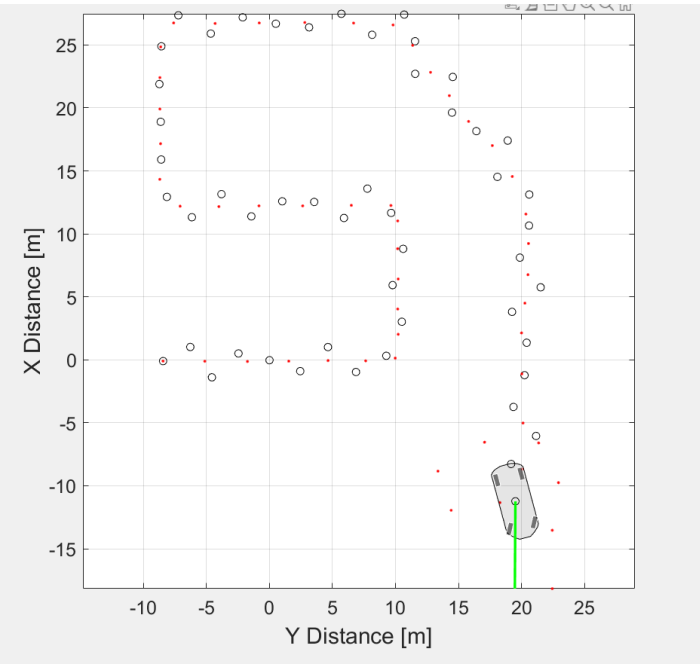


Figure 4.7.4.11 Simulation Result with V = 15 ℓ_d = 1

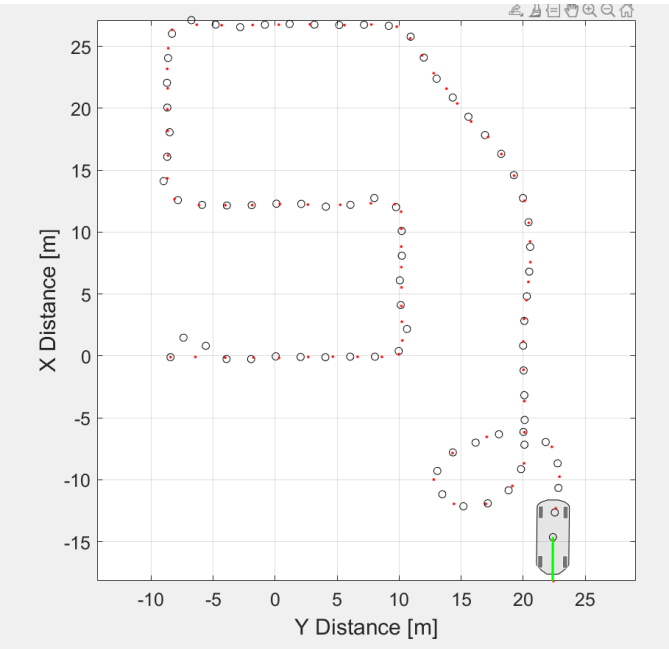


Figure 4.7.4.12 Simulation Result with V = 10 ℓ_d = 2

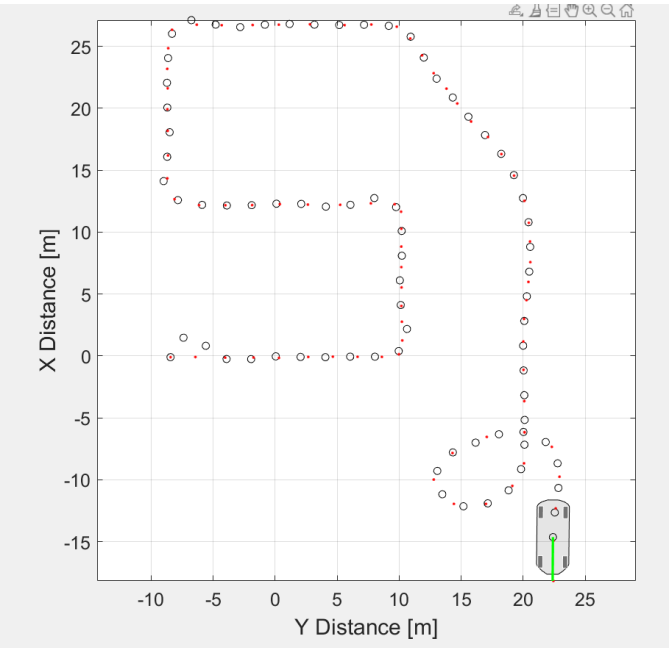


Figure 4.7.4.13 Simulation Result with V = 15 ℓ_d = 2

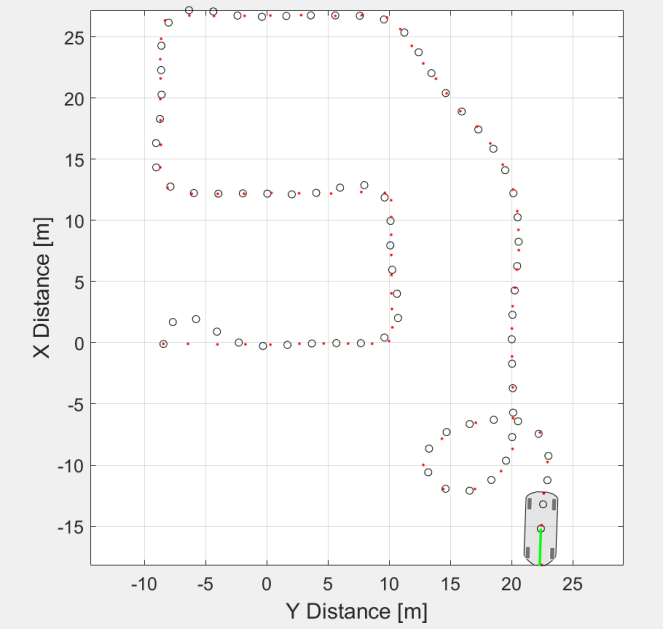


Figure 4.7.4.14 Simulation Result with V = 10 ℓ_d = 3

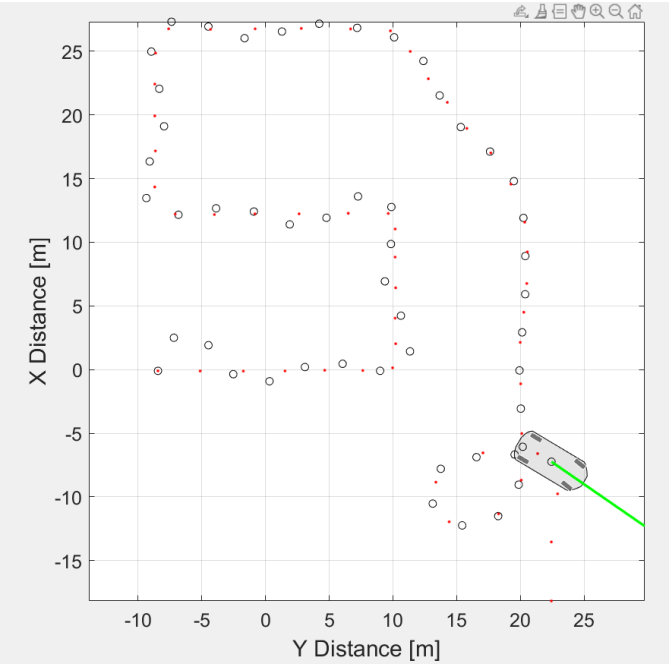


Figure 4.7.4.15 Simulation Result with V = 15 ℓ_d = 3

The results were just as expected. The vehicle sat on the desired route faster at low ℓ_d values but oscillated. At high ℓ_d values, it has set the desired route late, but oscillation has decreased. According to the results, as the vehicle speed increased, deviations from the course were observed and severe turns were observed. As a result, the mobile vehicle will travel at a constant speed of 1 m / s. When we evaluated the results for a speed of 1 m / s, it was

determined that the optimum $\ell_d$value was 1m. The vehicle steering angle is in the range of 0.3 to -0.3 radians. It was set at approximately 17 ° to -17 °.

4.9 Electrical Design

In this section, the electrical drive system design of the vehicle will be explained. The electrical drive system includes sensors, NVIDIA Jetson Tx2 vehicle computer, microcontroller, DC-DC converter, electronic speed controller (ESC), vehicle motor, servo motor, WIFI module and battery. The electric drive system block diagram is shown in the below.

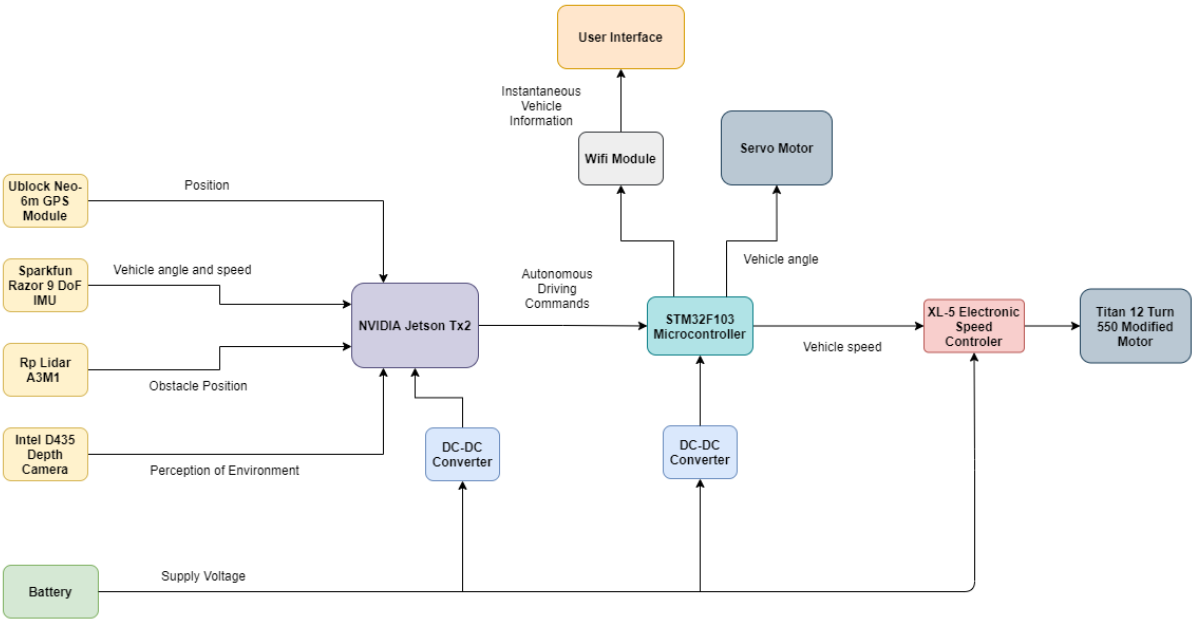


Figure 4.9.1 Electrical System Block Diagram

4.9.1 Vehicle Motor

The Titan 12T 550 modified motor is used in Traxxas Bigfoot. In this section, it has been tested whether the engine meets the expected competence. Firstly, dynamic model of system is analyzed and obtained minimum motor torque from there. Then equivalent motor parameters are entered in MATLAB simulation. PID is tuned manually. Finally, power outputs are investigated. Battery usage is visualized.

4.9.2 Simplification of System

Traxxas car has complex interior design. To analysis that, design was simplificated. Two backward wheels powered by one motor. Firstly, motor is connected Enhancer Spur Gear to increase torque. Then, Coupling Bevel is change rotation side. Finally, differential Bevel transmit torque to wheels.

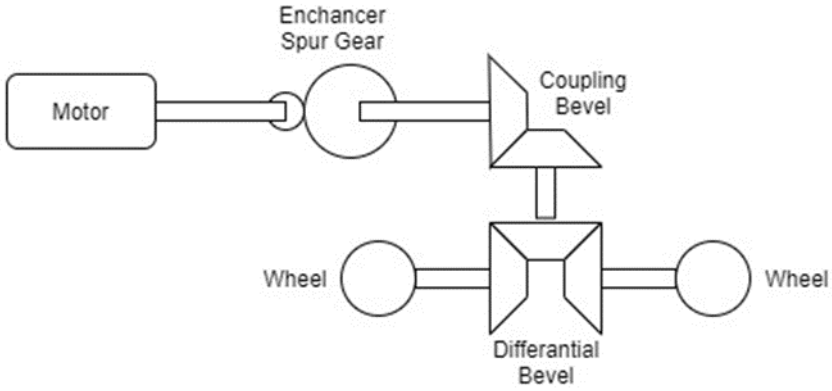


Figure 4.9.2.1 System schematic

4.9.3 Dynamic model

That system can be modelled with two part; translational and rotational. Motor torque pass through 3 gears and 4 rods then reach wheel. In wheel, force is generated via friction force. This force starts to move vehicle. Also, there is resistance force which is include friction and gravitational force. [34]

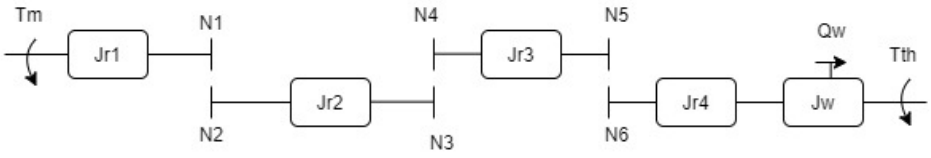


Figure 4.9.3.1 Rotational part

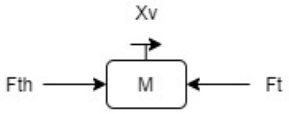


Figure 4.9.3.2 Translational part

Translational and rotational parts connected each other via wheel. The transform equations are;

$$F_{th}.r_w = \tau_{th} \tag{4.1}$$

$$Q_w.r_w = X_v \tag{4.2}$$

Resistive load represented as F_t. It is combination of friction and gravitational force.

$$F_t = F_f + F_g + F_D \tag{4.3}$$

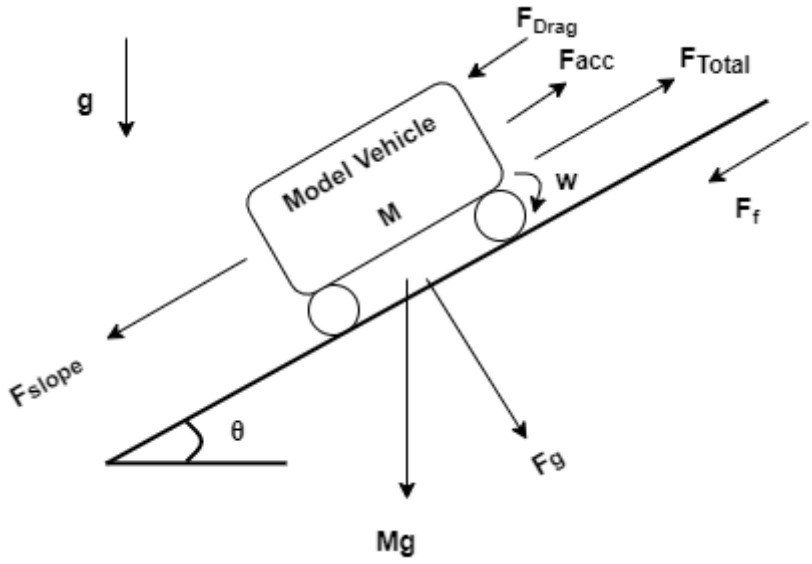


Figure 4.9.3.3 Force on the vehicle

$$F_g = M.g.\sin Q_s \tag{4.4}$$

$$F_f = M.g.\mu.\cos Q_s \tag{4.5}$$

$$F_D = \frac{1}{2}\,\rho\, C_d\, A\, v^2 \tag{4.6}$$

Max slope is as 20° and friction coefficient is 0.8.

Note: The vehicle surface area calculated by rectangle (0.10 x 0.30 m). Air drag coefficient assumed 1.2. Air density equal to 1.225 kg/m^3. Vehicle speed equal to 1 m/s. As a result, air drag force is neglected.

$$F_D = 0.02205 \text{ Nm} \tag{4.7}$$

For steady state condition; $\tau_{motor} = \tau_{load}$

$$\tau_m = F_t . r_w . N_{eq} = \mathbf{0.312\ N.m} \tag{4.8}$$

To reduce complexity of system, all elements are reflected to wheel side. General formula of reflection;

$$J_1 . (\frac{N_2}{N_1})^2 = J_2 \tag{4.9}$$

$$\tau_1 . \frac{N_2}{N_1} = \tau_2 \tag{4.10}$$

$$Q_1 . \frac{N_1}{N_2} = Q_2 \tag{4.11}$$

There is one torque source. So that we can directly reflect it to wheel side from motor. However, there is more than one inertia. Start from left, reflect first inertia through first gear. Then sum reflected inertia and inertia that already is in there. These temporary values are named like J_{1eq} and J_2 in excel table. With repeating this process, can be reach to motor side.[35]

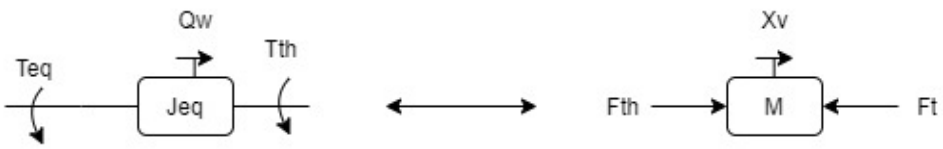


Figure 4.9.3.4 Equivalent system

Table 4.9.3.1 Transmission elements properties

Material	Element	Radius	Length	Density	Mass
Al6013	Rod1	0.005	0.01	2700	0.002120575
Al6013	Rod2	0.0075	0.016	2700	0.00763407
Al6013	Rod3	0.0075	0.01	2700	0.004771294
Al6013	Rod4	0.006	0.064	2700	0.01954322

Table 4.9.3.2 Equivalent inertia calculation

Property	Value		Gear	Value		Derived Property	Value
Jr1	2.65072E-08		N1	1		J1eq	2.65E-08
Jr2	2.14708E-07		N2	3		J2	2.39E-07
Jr3	1.34193E-07		N3	1		J2eq	4.53E-07
Jr4	3.51778E-07		N4	2		J3	1.81E-06
Jm	0		N5	1		J3eq	1.95E-06
Jw	0.000125		N6	1		J4	1.95E-06
						Jeq	0.000503

$$\boldsymbol{n_{rot}} = Q_w.\left[J_{eq}.s^2\right] + \tau_{th} = \tau_{eq} \tag{4.12}$$

$$\boldsymbol{n_{tran}} = X_w.\left[M.s^2\right] + F_t = F_{th} \tag{4.13}$$

Final form is;

$$\frac{\dot{X_v}}{\tau_{net}} = \frac{1/M_{eq}}{s} \tag{4.14}$$

Where $\tau_{net} = \tau_m.N_{eq} - F_t.r_w$ and $M_{eq} = {}^{J_{eq}}/_{r_w} + M.r_w$

4.9.4 Defining Motor Requirements

$$P = T\omega \tag{4.15}$$

When calculating the technical requirement, the $V = 1\ [m/s]$ was determined. As a

result of the mechanical calculation, the value of the maximum torque requirement was obtained as $T = 0.312\ [N.m]$.

$$\omega = \frac{V}{r} \tag{4.16}$$

Since the gear ratio between the motor and the wheel is determined as 6 and the wheel radius is $0.05\ [m]$, our value of ω is:

$$\omega = 6 \cdot \frac{V}{r} = 6 \cdot \frac{1}{0.05} = 120\ \left[\frac{rad}{s}\right] \tag{4.17}$$

Thus, substituting torque and angular velocity values:

$$P_{steadystate} = 0.312 \cdot 120 = 37.44\ [W] \tag{4.18}$$

Motor and Jetson TX2(AI Computer) are the most power consumer elements in vehicle. Motor draw 4.6 A and Jetson draw 0.8 A. Total current consuming is nearly 5.5 A.

$$5.5A \times 0.33Hour = 1.8\ AH = 1800\ mAH \tag{4.19}$$

In model vehicle, there is Traxxas 2923X 3000mAh NiMH 7-C Flat 8.4V Battery**.** Its capacity is enough and satisfies technical requirements.

Figure 4.9.4.1 Traxxas 2923X 3000mAh NiMH 7-C Flat 8.4V Battery

4.9.5 Simulation

The MATLAB model is shown in the below.

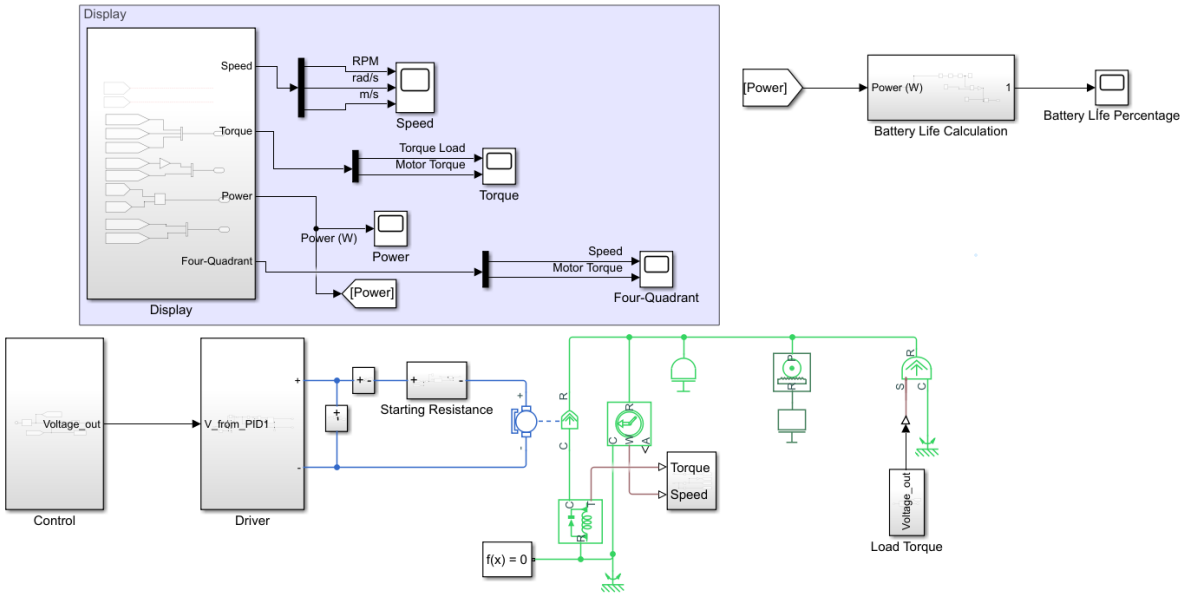


Figure 4.9.5.1 Electrical drive system MATLAB model

In simulation, vehicle desired speed selected 0.8 m/s, 0 m/s, -0.8 m/s and 0 respectively. Also, motor load selected as -0.228N.m and 0.228 N.m which are represent straight road resistance. With these values 4-Quadrant operation can be obtain in limited time interval. All situations were observed that way.

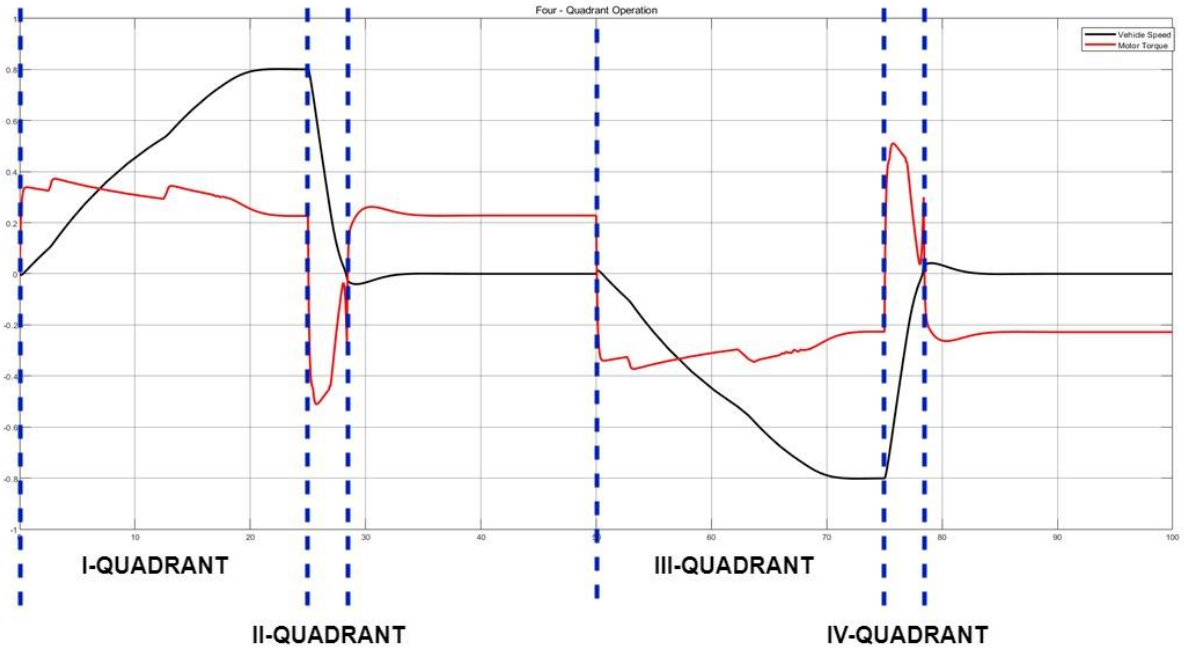


Figure 4.9.5.2 Four – quadrant operation

Power graph shows that in steady state conditions motor power is 40 W which is very closer to analytic calculation. The difference is occurred due to driver and motor resistance, they are not taken into account in analytical calculations. Attention should be paid at transient response, motor power jumps to 116 Watt. Titan 12 T can work up to 137 Watt, so it is still under control.

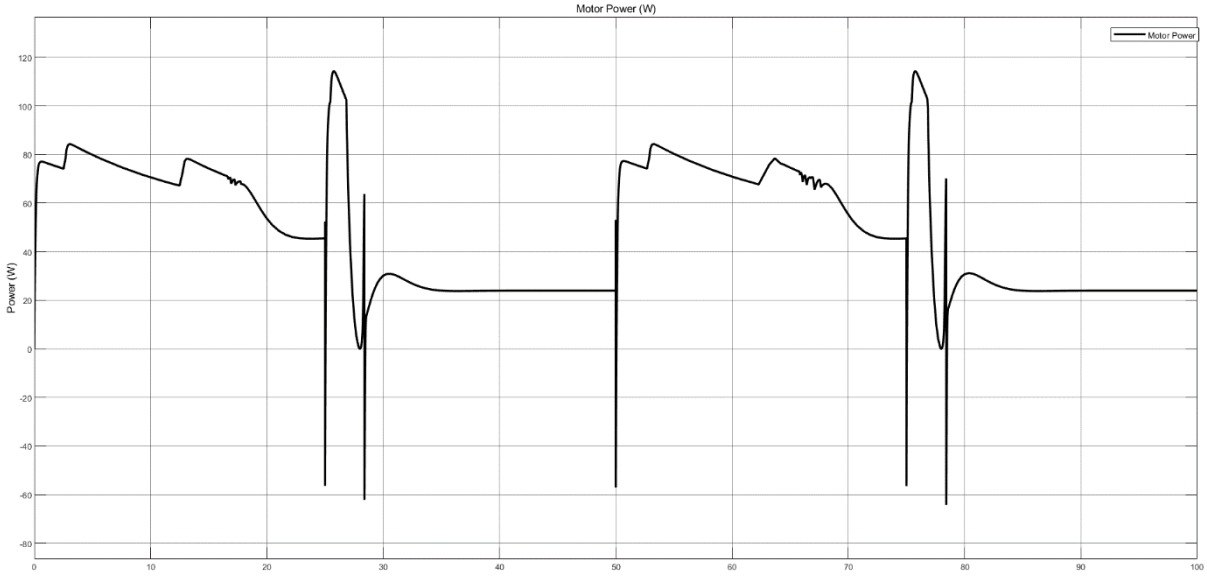


Figure 4.9.5.3 Power graph

For 100 second simulation, battery usage is calculated from energy principle. %2.75 is used in that time interval. That's mean battery recharge %1.65 in every minute. According to simulation

vehicle can work in steady state condition in 59 minute.

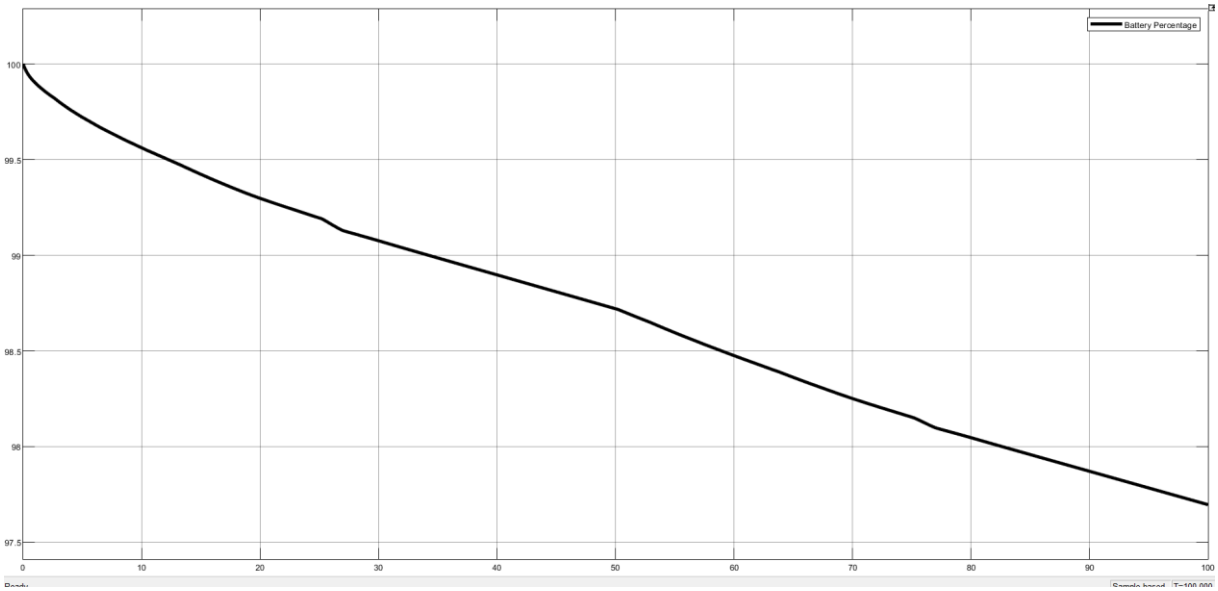


Figure 4.9.5.4 Battery state of charge

4.10 Mechanical Design

TRAXXAS Bigfoot was chosen as the model vehicle. Because the size is suitable for our sensors design on the vehicle.

Figure 4.10.1 TRAXXAS Bigfoot No 1 [36]

A platform has been designed for the sensors and components to be added on the vehicle's cover. Lidar is positioned at the top of the vehicle. One more piece is installed for this. Thus, lidar will be able to scan the whole environment without any obstacles. However, due to the lightness of lidar, stress analysis was not performed.

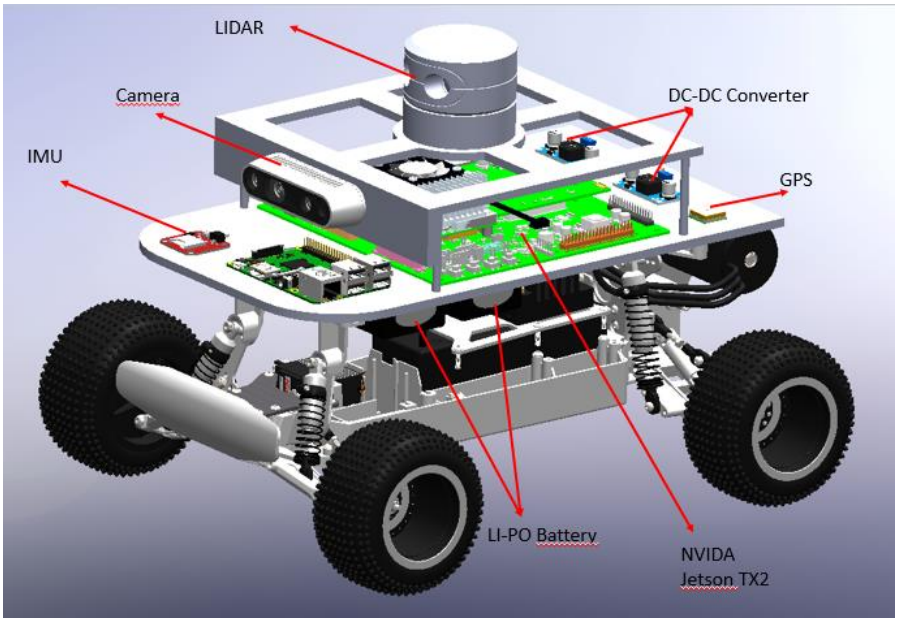


Figure 4.10.2 Model Car

The platforms to be produced will be 3D printed and will be made of ABS plastic. Static load analysis of the platform was performed on Autodesk Fusion 360. Factor of safety 8 has been selected. Stress test results are as given below.

Table 4.10.1 Result Summary of Stress Test

Name	**Minimum**	**Maximum**
Safety Factor		
Safety Factor (Per Body)	5.34	15
Stress		
Von Mises	0.001128 MPa	3.745 MPa
1st Principal	-2.873 MPa	4.841 MPa
3rd Principal	-6.914 MPa	2.426 MPa

Normal XX	-5.225 MPa	4.142 MPa
Normal YY	-4.447 MPa	3.436 MPa
Normal ZZ	-3.676 MPa	2.746 MPa
Shear XY	-1.035 MPa	1.527 MPa
Shear YZ	-0.625 MPa	1.068 MPa
Shear ZX	-0.9174 MPa	0.8248 MPa
Displacement		
Total	0 mm	0.3949 mm
X	-0.01255 mm	0.0126 mm
Y	-0.008429 mm	0.008414 mm
Z	-0.3949 mm	0.06401 mm
Reaction Force		
Total	0 N	34.21 N
X	-15.72 N	13.54 N
Y	-14.57 N	12 N
Z	-22.88 N	31.87 N
Strain		
Equivalent	8.274E-07	0.003001
1st Principal	5.837E-07	0.002305
3rd Principal	-0.003366	-6.261E-07
Normal XX	-0.001105	0.00103
Normal YY	-6.248E-04	5.605E-04
Normal ZZ	-5.956E-04	8.898E-04
Shear XY	-0.001276	0.001882
Shear YZ	-7.701E-04	0.001316
Shear ZX	-0.00113	0.001016

4.10.1 Safety Factor

0  8

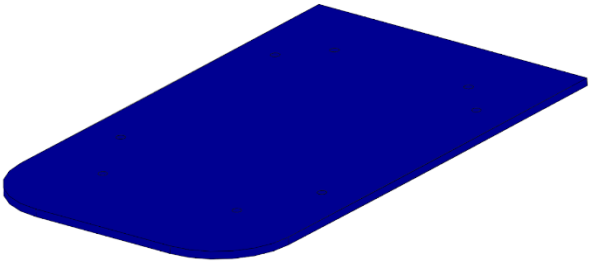


Figure 4.10.1.1 Safety Factor

4.10.2 Stress

Von Mises

[MPa] 0.001  3.745

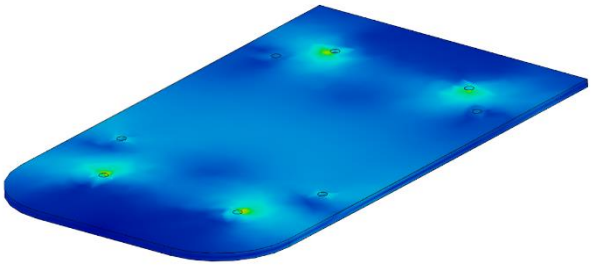


Figure 4.10.2.1 Von Mises

1st Principal

[MPa] -2.873  4.841

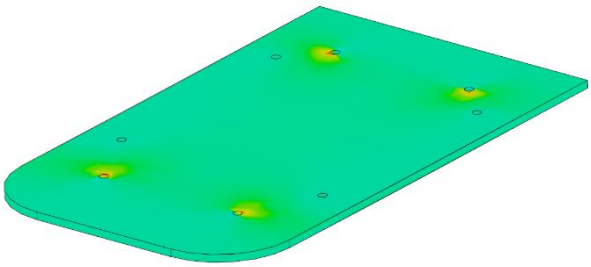


Figure 4.10.2.2 1st Principal

3rd Principal

[MPa] -6.914  2.426

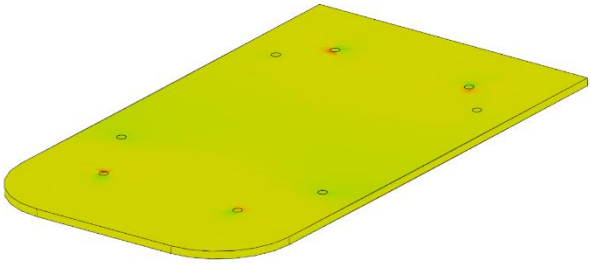


Figure 4.10.2.3 3rd Principal

4.10.3 Displacement

[mm] 0 0.3949

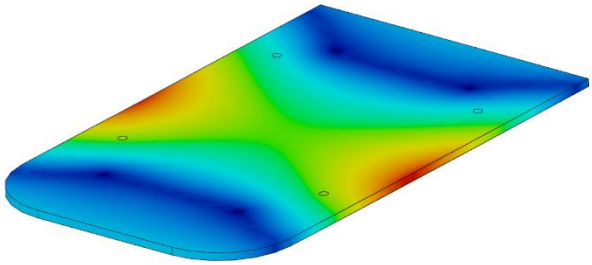
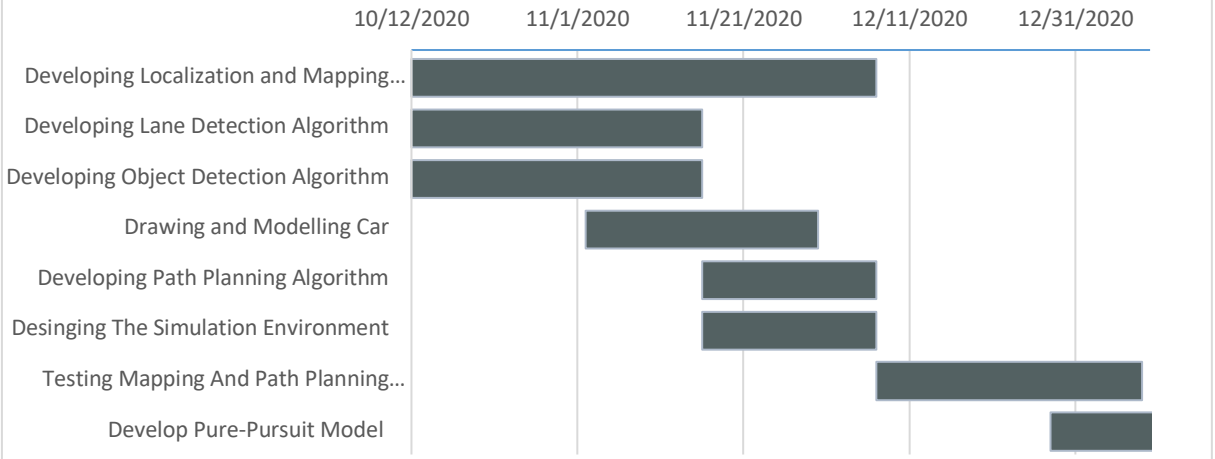


Figure 4.10.3.1 Displacement

5. WORKING PLAN

Table 5.1 Working Plan



	Develop Pure-Pursuit Model	Testing Mapping And Path Planning Algorithms in Simulation	Desinging The Simulation Environment	Developing Path Planning Algorithm	Drawing and Modelling Car	Developing Object Detection Algorithm	Developing Lane Detection Algorithm	Developing Localization and Mapping Algorithms
Start date	12/28/2020	12/7/2020	11/16/2020	11/16/2020	11/2/2020	10/12/2020	10/12/2020	10/12/2020
Duration (in days)	18	32	21	21	28	35	35	56

6. BUDGET

6.1 List of Materials

Table 6.1.1 Total Cost of Materials

Component	Price	Piece	Total
TRAXXAS BigFoot RC Car	₺1.840,00	1	₺1.840,00
NVIDIA Jetson TX2 Developer Kit	₺5.000,00	1	₺5.000,00
RPLIDAR A3M1	₺6.113,00	1	₺6.113,00
INTEL Realsense D435	₺2.000,00	1	₺2.000,00
Ublox NEO-6M Gps Module	₺150,80	1	₺150,80
SparkFun 9DoF Razor IMU M0	₺340,00	1	₺340,00
STM32F103	₺45	1	₺45
LM2596 DC-DC Converter	₺ 20	4	₺80,00
Profuse 14.6 4S LiPo Battery 3000 m AH 35C	₺370,00	2	₺740,00
Other	₺200,00	1	₺200,00
TOTAL			**₺16.508,80**

6.2 Procurement of Services

Table 6.2.1 Procurement of Services

Institution	**Services**	**Purpose of Usage**
Yıldız Technical University Faculty of Mechanical Engineering	Workshop	Working Area
Yıldız Technical University Technopark Prototype Workshop	3D Printer	Production of carrier parts in vehicle construction
Yıldız Technical University Alternative Resources System Society	Support for simulation environment	Testing algorithms in simulation
Yıldız Technical University Mechatronics Laboratory	Power supply, Oscilloscope, Signal generator, Multimeter	Control and testing of electronic elements

6.3 External Financial Support Applications

Within the scope of the project, an application was made to the TÜBİTAK 2209-B Sanayiye Yönelik Lisans Araştırma Projeleri Desteği Programı". In the application, cooperation was made with RASS Technology, which is Yıldız Technical University Technopark company, as an industrial consultant. Within the scope of the project, sensors (Lidar, Camera), on-board computer NVIDIA Jetson TX2 and Traxxas Bigfoot vehicle will be provided from our academic advisor.

7. RESULTS, DISCUSSION AND FUTURE WORKS

The general design of the project has been made. Team members were assigned for each subsystem. Later, the team members completed their literature study on their task.

Conventional lane detection algorithms have problems in that the detection rate is lowered in road environments having a large change in curvature and illumination. The probabilistic Hough transform method has low lane detection rate since it exploits edges and restrictive angles. On the other hand, the method using a sliding window can detect a curved lane as the lane is detected by dividing the image into windows. However, the detection rate of this method is affected by road slopes because it uses affine transformation. In order to detect lanes robustly, we propose driving assist system using semantic segmentation based on deep learning. Unfortunately, semantic segmentation could not be performed due to some technical deficiency. That's why sliding window was used as a backup plan.

In the object detection part, object detection algorithms were compared. The YOLO algorithm, which is the most suitable algorithm for real time, was chosen. As a result of the tests, it was decided to use the YOLOv3 Tiny algorithm for the project. As a result of the test, STOP-NO ENTRY- YIELD traffic signs with 35 fps were detected. The code was implemented to use the algorithm on the ROS.

In the mapping part, the SLAM algorithm was investigated. It was tested on the track designed in Gazebo environment. The course map was created with the ROS package used. Path planning algorithm was used on the map and successfully reached the specified point on the track.

Semantic segmentation will be used by eliminating technical deficiencies. The artificial intelligence of the model vehicle will be trained, and lane tracking will be provided

Literature search was done for lateral control. As a result of the researches, it was decided to use the Pure Pursuit control algorithm. Pure Pursuit algorithm was tested with the model designed in MATLAB / Simulink environment. Look-ahead distance was determined according to the simulation results.

Sensors required for autonomous driving were selected.

The vehicle was built in 3D in the SolidWorks environment. Selected sensors were placed on the platform built on the vehicle.

The electrical drive system of the vehicle was designed. Required components have been selected.

The instant location of the vehicle will be determined using the Kalman Filter Extended with GPS and IMU data.

Localization algorithms will be researched and tested.

A longitudinal control algorithm will be created, and the behavior of the vehicle will be tested according to different speeds.

All algorithms will communicate in ROS environment.

Sensor calibration will be done.

Traffic signs and road lines will be added to the simulation environment.

REFERENCE

[1] (2021). The Future of Autonomous Cars. Berginsight.

[2] *Karayolu Trafik Kaza İstatistikleri, 2019*. (2020, 1 16). data.tuik.gov.tr: https://data.tuik.gov.tr/Bulten/Index?p=Karayolu-Trafik-Kaza-Istatistikleri-2019-33628

[3] INTERNATIONAL, S. (2018). Taxonomy and Definitions for Terms Related to Driving Automation. SAE INTERNATIONAL.

[4] Czarnecki, K. (2018). *Operational Design Domain for Automated Driving Systems Taxonomy of Basic Terms.* Waterloo,Canada: University of Waterloo.

[5] *racecar*. (2020, 1 16). racecar.mit.edu: https://racecar.mit.edu/

[6] *About.* (2020, 1 16). f1tenth.org: https://f1tenth.org/about.html

[7] autonomous-vehicle-market. (2021, 1 16). www.alliedmarketresearch.com: https://www.alliedmarketresearch.com/autonomous-vehicle-market

[8] Hasan, K. S. (2020, 1 15). *What, Why and How of ROS*. towardsdatascience.com: https://towardsdatascience.com/what-why-and-how-of-ros-b2f5ea8be0f3

[9] *Distributions*. (2020, 1 15). wiki.ros.org: http://wiki.ros.org/Distributions

[10] *8.04*. (2020, 1 15). releases.ubuntu.com: https://releases.ubuntu.com/18.04/

[11] *tutorials*. (2020, 1 15). gazebosim.org: http://gazebosim.org/tutorials?tut=guided_b1&cat=

[12] *course_introduction_to_autodesk_fusio*. (2020, 1 15). platform.europeanmoocs.eu: https://platform.europeanmoocs.eu/course_introduction_to_autodesk_fusio#:~:text=Fusion%20360%20is%20a%20cloud,manufacturing%20in%20one%20comprehensive%20package.

[13] Aerts, P., & Demeester, E. (2017). *Benchmarking of 2D-Slam Algorithms.* DIEPENBEEK, BELGIUM: ACRO RESEARCH GROUP

[14] Filipenko, M., & Afanasyev, I. (2018). Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment. *9th IEEE International Conference on Intelligent Systems.*

Madeira: Portugal.

[15] Dominguez-Quijada, S., Ali, A., Garcia, G., & Martinet, P. (2016). Comparison of lateral controllers for autonomous vehicle : experimental results. *International IEEE Conference on Intelligent Transportation Systems* . Rio de Janeiro, Brazil: IEEE.

[16] Snider, J. M. (2009). Automatic Steering Methods for Autonomous Automobile Path Tracking. Pittsburgh, Pennsylvania: Carnegie Mellon University .

[17] C. (2017, March 14). *Finding Lane Lines with Image Processing*. CMLPR Corp. https://cmlpr.github.io/blog/2017/03/14/advanced-lane-lines

[18] RealSense™ D400 Series - Intel. (n.d.). Retrieved from https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf

[19] Waslander, P. (n.d.). 4 – Measurement Modeling. Retrieved from http://wavelab.uwaterloo.ca/sharedata/ME597/ME597_Lecture_Slides/ME597-4-Measurement.pdf

[20] Mazzari, V., & Mazzari, V. (2020, May 08). How to select the right LiDAR? Retrieved January 07, 2021, from https://blog.generationrobots.com/en/how-to-select-the-right-lidar/

[21] What is gps ? (2020, Ekim 30). gps.gov: https://www.gps.gov/systems/gps/

[22] What is IMU? Inertial Measurement Unit Working & Applications. (2020, Ekim 30). arrow.com: https://www.arrow.com/en/research-and-events/articles/imu-principles-and-applications

[23] Uppala, R. (2018, June 5). *Advanced Lane Detection for Autonomous Vehicles using Computer Vision techniques.* Medium. https://towardsdatascience.com/advanced-lane-detection-for-autonomous-vehicles-using-computer-vision-techniques-f229e4245e41

[24] Kusram, K. B. (2020, May 5). *Advanced Lane Detection - The Startup*. Medium. https://medium.com/swlh/advanced-lane-detection-fd39572cfe91

[25] training-yolo-v3-for-objects-detection-with-custom-data. (2021, 1 16). udemy.com: https://www.udemy.com/course/training-yolo-v3-for-objects-detection-with-custom-data/learn/lecture/15565560#notes

[26] Redmon, J., & Farhadi, A. (2020, 1 16). YOLOv3: An Incremental Improvement. Washington, USA: University of Washington.

[27] evolution-of-yolo-yolo-version-1. (2020, 1 16). towardsdatascience.com: https://towardsdatascience.com/evolution-of-yolo-yolo-version-1-afb8af302bd2

[28] darknet. (2021, 1 16). pjreddie.com: https://pjreddie.com/darknet/yolo/

[29] *JACKAL*. (2021, 1 12). clearpathrobotics.com: https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/

[30] *jackal_velodyne*. (2021, 1 12). Github.com: https://github.com/TixiaoShan/jackal_velodyne

[31] *jackal*. (2021, 1 12). Github.com: https://github.com/jackal/jackal

[32] *Gmapping*. (2020, 1 12). Openlsam: https://openslam-org.github.io/

[33] *wiki.ros.org*. (2021, 12 1). gmapping: http://wiki.ros.org/gmapping

[34] *YTU-MKT SystemDynamics*. (2020, 12 30). Youtube: https://www.youtube.com/channel/UCaQtr0oBp_HZe27ezfF6BrQ

[35] Palm, W. (2014). *System Dynamics.* New York, NY: McGraw-Hill.

[36] products. (2020, 1 16). traxxas.com: https://traxxas.com/products/models/electric/bigfoot-classic?t=overview

RESUME

MAHSUN BİNGÖL

Educations

2011 –- 2015: Diyarbakır Anatolian High School

2016 - …: Yıldız Technical University – Mechatronics Engineering

Internship

2019: Sinbo Deima Elektromakanik A.Ş

mahsunbngl0@gmail.com

KADİR ERTUĞ ACAR

Educations

2012 – 2016: 60. YIL Anatolian High School

2016 - …: Yıldız Technical University – Mechatronics Engineering

Internship

2019: YTU Technopark Protatip Atolyesi

k.acarertug@gmail.com

ÖZKAN GÖKSU

Educations

2012 – 2016: Celal Bayar Anatolian High School

2016 - …: Yıldız Technical University – Mechatronics Engineering

Internship

2019: Erdoğanlar Otomotiv A.Ş.

o.goksu98@gmail.com