

# Table of Content

ABSTRACT .....	6
<b>1. INTRODUCTION.....</b>	<b>7</b>
1.1 CONNECTED AND AUTONOMOUS VEHICLES (CAVs) .....	7
1.2 THE CURRENT STATE AND FUTURE TRENDS OF CAVS.....	8
1.3 PROJECT MOTIVATIONS .....	8
1.4 AIM AND OBJECTIVES .....	8
1.5 HARDWARE AND SOFTWARE REQUIREMENTS.....	9
1.6 SCOPE AND LIMITATIONS.....	9
1.7 STRUCTURE OF THE REPORT .....	9
<b>2. LITERATURE REVIEW .....</b>	<b>9</b>
2.1 VEHICLE TO INFRASTRUCTURE (V2I) .....	9
2.2 LIDAR .....	10
2.3 LOCALISATION .....	11
2.4 OBJECT DETECTION .....	11
2.5 PATH PLANNING .....	13
2.5.1 AN OVERVIEW OF PATH PLANNING .....	13
2.5.2 SAMPLE OF POTENTIAL SOLUTIONS .....	13
2.6 PATH FOLLOWING .....	13
2.6.1 AN OVERVIEW OF PATH FOLLOWING .....	13
2.6.2 SAMPLE OF POTENTIAL SOLUTIONS .....	14
2.7 VEHICLE REMOTELY CONTROL.....	14
2.7.1 ACKERMANN MASSAGE.....	14
2.7.2 GUI .....	14
2.8 VERIFICATION AND VALIDATION .....	15
<b>3. METHODOLOGY .....</b>	<b>15</b>
3.1 LOCALISATION .....	15
3.1.1 OFFLINE STEP .....	16
3.1.1.1 POINT CLOUD REGISTRATION .....	16
3.1.2 REAL-TIME STEP .....	16
3.1.2.1 POINT CLOUD PRE-PROCESSING .....	16
3.1.2 MOVING VEHICLE EXTRACTION.....	17
3.1.2.1 BACKGROUND FILTERING.....	17
3.1.2.2 OUTLIER REMOVAL.....	17
3.1.3 CLUSTERING .....	18
3.1.3.1 DOWNSAMPLING WITH VOXEL GRID.....	18
3.1.3.2 SEGMENTATION .....	18
3.1.3.3 EUCLIDEAN CLUSTER EXTRACTION.....	19
3.1.4 BOUNDING BOX GENERATION .....	20
3.1.5 EGO VEHICLE DETECTION.....	20
3.1.6 MATCHING THE MESH OF OBJECTS .....	20
3.2 OBJECT DETECTION AND CLASSIFICATION .....	22
3.3 PATH PLANNING .....	22
3.4 PATH FOLLOWING .....	24
3.5 REMOTELY CONTROL.....	25
3.5.1 GUI .....	25
3.5.2 STEERING COMMAND .....	26
3.6 VERIFICATION AND VALIDATION .....	27

3.6.1 ROADRUNNER.....	27
3.6.2 CARLA.....	28
<b>4. RESULTS AND DISCUSSION .....</b>	<b>29</b>
<b>4.1 LOCALISATION .....</b>	<b>29</b>
4.1.1 POINT CLOUD REGISTRATION .....	29
4.1.2 CLUSTERING .....	30
4.1.3 BOUNDING BOX.....	31
4.1.4 EGO VEHICLE DETECTION.....	32
4.1.5 MATCHING THE MESH OF OBJECTS .....	33
<b>4.2 OBJECT DETECTION .....</b>	<b>34</b>
<b>4.3 PATH PLANNING .....</b>	<b>35</b>
<b>4.4 PATH FOLLOWING .....</b>	<b>37</b>
<b>4.5 VERIFICATION AND VALIDATION .....</b>	<b>40</b>
<b>5. MARKET OPPORTUNITY AND BUSINESS MODEL.....</b>	<b>44</b>
<b>5.1 INDUSTRY OVERVIEW .....</b>	<b>44</b>
5.1.1 MARKET DRIVERS.....	44
5.1.2 MARKET RESTRAINTS .....	45
<b>5.2 PRODUCT DESCRIPTION.....</b>	<b>45</b>
<b>5.3 FOCUSED ASSOCIATE PARTNERS AND COLLABORATION .....</b>	<b>48</b>
<b>5.4 FINANCIAL ANALYSIS .....</b>	<b>48</b>
5.4.1 INFRASTRUCTURE DEVELOPMENT.....	49
5.4.2 COST ANALYSIS FOR OUR COMPANY.....	49
5.4.3 COST ANALYSIS FOR OUR CUSTOMERS.....	50
<b>5.6 PORTER ANALYSIS.....</b>	<b>52</b>
<b>5.7 RISK AND MITIGATION MEASURES.....</b>	<b>54</b>
<b>5.8 CHALLENGES AND FUTURE WORK.....</b>	<b>54</b>
<b>5.9 CONCLUSION .....</b>	<b>55</b>
<b>6. PROJECT MANAGEMENT .....</b>	<b>55</b>
<b>6.1 INITIATION PROCESS .....</b>	<b>55</b>
<b>6.2 PLANNING PROCESS .....</b>	<b>55</b>
6.2.1 WORK BREAKDOWN STRUCTURE (WBS).....	55
6.2.2 RASCI MATRIX .....	56
6.2.3 GANTT CHART .....	57
6.2.4 RISK MANAGEMENT PLAN .....	59
6.2.5 COMMUNICATION PLAN .....	59
<b>6.3 EXECUTION .....</b>	<b>60</b>
<b>7. CHALLENGES AND LESSONS LEARNED .....</b>	<b>60</b>
<b>7.1 CHALLENGES.....</b>	<b>60</b>
7.1.1 TECHNICAL .....	60
7.1.2 NON-TECHNICAL .....	61
<b>7.2 LESSON LEARNED.....</b>	<b>61</b>
<b>8. FUTURE WORK.....</b>	<b>62</b>
<b>REFERENCE .....</b>	<b>62</b>

# List of Figures

Figure 1 Levels of automation based on SAE J3016 (SAE International, 2021) .....	7
Figure 2 Left: High-level Architecture of V2I (Kumar & Ali, 2022), Right: RSU at MUEAVI .....	10
Figure 3 Left: Point Cloud of LiDAR, Right: LiDAR schematic architecture (Shenzhen da xue, 2018).....	10
Figure 4 Results of various algorithms for clustering(Comparing Python Clustering Algorithms — Hdbscan 0.8.1 Documentation, n.d.).....	12
Figure 5 High-Level System Design for Localisation and Obstacle Detection.....	16
Figure 6 Noisy Data Without Outlier Removal.....	17
Figure 7 Left. Before applying the voxel grid filter vehicle points are dense, Right. After Voxel grid filter points become less dense. ....	18
Figure 8 Scheme of the error between the centroid of the boundary box and the real reference point of the vehicle .....	21
Figure 9 Point cloud generated from the mesh of Land Rover Discovery 4 .....	21
Figure 10 ROS move_base navigation stack (ROS.org, n.d. -c).....	22
Figure 11 Occupancy grid map generation using point cloud data (Gurel, 2018). ....	23
Figure 12 teb_local_planner description (Marin-Plaza et al., 2018). ....	23
Figure 13 The Ackermann Model, with a simplified bicycle model (Sprinkle et al., 2008). ....	24
Figure 14 Pure Pursuit controller representation for a bicycle model (Theers & Singh, 2022)... ..	25
Figure 15 High level flow chart of system integration.....	26
Figure 16 Road Runner Map.....	27
Figure 17 Road Runner Map.....	27
Figure 18 Test map 2 for simulation in Carla .....	28
Figure 19 Pedestrian detection in Carla.....	29
Figure 20 Output of Clustering .....	31
Figure 21 Bounding Box for the Vehicle and the Pedestrian.....	31
Figure 22 Bounding boxes for the vehicle and pedestrian .....	32
Figure 23 Vehicle Detection .....	32
Figure 24 Generation of an appropriate point cloud of the Land Rover Discovery .....	33
Figure 25 ROS visualisation of the output of the NDT algorithm .....	33
Figure 26 No Obstacle Case.....	34
Figure 27 Obstacle Detected Case .....	35
Figure 28 Path generation using move_base package in Carla.....	36
Figure 29 Cost map generation for the boundary of the road.....	36
Figure 30 Steering Angle Results .....	37
Figure 31 Path Tracking Test.....	38
Figure 32 Tracking Error .....	39
Figure 33 Ackermann Command – Steering angle .....	39
Figure 34 Ackermann Command – Speed.....	40
Figure 35 Object detection using point cloud data in Carla. ....	41
Figure 36 Object detection using point cloud data in Real environment. ....	41
Figure 37 Working of Path following algorithm in Carl environment.....	42
Figure 38 Working of Path following algorithm in Real environment. ....	42
Figure 39 Simulation results of pure pursuit controller in Carla.....	43
Figure 40 Left – Global market for Software and other services for autonomous vehicles, right – Sales penetration of autonomous vehicles (Toptal, AT Kearney, Boston Consulting Group).....	44
Figure 41 Expected growth of the Automotive software market (McKinsey, 2019) .....	45
Figure 42 Hierarchy for the achievement of the end goal .....	46
Figure 43 Recommended LiDAR sensor comparison (Levelfivesupplies) .....	47
Figure 44 Highway Strategic Road Network UK Blue – Motorways, Red – All Purpose and Dotted Blue – Toll Roads (NationalHighways.co.uk).....	48
Figure 45 Self Driving cars component wise cost (Wired).....	50

Figure 46 Work Breakdown Structure .....	56
Figure 47 Gantt Chart .....	58

## List of Tables

Table 1 Fitness Scores of Point Registration Algorithms .....	30
Table 2 Location errors .....	37
Table 3 Business Model Canvas.....	47
Table 4 CAPEX.....	49
Table 5 Company Cost Analysis .....	50
Table 6 Cost comparison of I2V Integration Level .....	51
Table 7 Risk and Mitigation measures .....	54
Table 8 RASCI Matrix .....	57
Table 9 Key Dates .....	59
Table 10 Risk Management Plan .....	59
Table 11 Issue log table .....	59
Table 12 Communication Plan .....	60

# Abstract

This report details the outcomes of a Group Design Project (GDP) that investigate and develop driverless navigation on the highway using Vehicle to Infrastructure (V2I) communication and roadside sensors, especially LiDAR. This project focused on developing software packages which can share pre-recorded information about the road layout, detect the presence of vehicles and pedestrians, and guide a vehicle through a designated lane, with the ability to command it to stop safely if necessary. Several sub-systems have been developed to achieve these requirements, including localisation, object detection, path planning, path following, and verification and validation. The report analyses the methodology and results of each task involved in developing software to control a vehicle in each scenario. While the software has been successfully developed to satisfy the necessary functionalities, such as road layout representation, vehicle detection, speed planning, steering control, and graphical interface, the remaining problems must be resolved. In addition, the report contains comprehensive market research, financial analysis and a business plan. The market analysis identifies potential market segments and competitors. At the same time, the proposed Associate Partners and Collaboration method seeks to find partnerships between the company and other stakeholders to enhance the product's viability. Finally, the report discusses the project management procedure as the typical development procedure for the group project. Despite encountering several difficulties during the project, many methodologies have been implemented to ensure effective project management. In addition, the report evaluates the project's obstacles and lessons learned, which can be applied to future group projects in either the academic or commercial.



# Driverless Navigation on Highways using V2I Communication and Roadside Sensors

## GDP-CAVE2

## 1. Introduction

### 1.1 Connected and Autonomous Vehicles (CAVs)

Connected and Autonomous Vehicles (CAVs) are a new generation of vehicles that utilise several technologies to communicate with each other and their surroundings and operate without human intervention using advanced technologies such as sensors, communication networks, machine learning (ML), artificial intelligence (AI), navigation and localisation. These technologies enable CAVs to perceive and understand their surroundings, detect and respond to potential hazards, and appropriately navigate the destination. From the SAE J3016 standard, CAVs can be categorised into six levels of autonomy, ranging from no (level 0), partial, and full driving automation (level 5), which do not require any human intervention at all (SAE International, 2021).



#### SAE J3016™ LEVELS OF DRIVING AUTOMATION™

Learn more here: [sae.org/standards/content/j3016\\_202104](https://www.sae.org/standards/content/j3016_202104)

Copyright © 2021 SAE International. The summary table may be freely copied and distributed AS-IS provided that SAE International is acknowledged as the source of the content.

	SAE LEVEL 0™	SAE LEVEL 1™	SAE LEVEL 2™	SAE LEVEL 3™	SAE LEVEL 4™	SAE LEVEL 5™
What does the human in the driver's seat have to do?	You <b>are</b> driving whenever these driver support features are engaged – even if your feet are off the pedals and you are not steering			You <b>are not</b> driving when these automated driving features are engaged – even if you are seated in “the driver's seat”		
	You must constantly supervise these support features; you must steer, brake or accelerate as needed to maintain safety			When the feature requests, you must drive	These automated driving features will not require you to take over driving	

Copyright © 2021 SAE International.

	These are driver support features			These are automated driving features		
What do these features do?	These features are limited to providing warnings and momentary assistance	These features provide steering <b>OR</b> brake/acceleration support to the driver	These features provide steering <b>AND</b> brake/acceleration support to the driver	These features can drive the vehicle under limited conditions and will not operate unless all required conditions are met	This feature can drive the vehicle under all conditions	
Example Features	<ul style="list-style-type: none"> <li>• automatic emergency braking</li> <li>• blind spot warning</li> <li>• lane departure warning</li> </ul>	<ul style="list-style-type: none"> <li>• lane centering <b>OR</b></li> <li>• adaptive cruise control</li> </ul>	<ul style="list-style-type: none"> <li>• lane centering <b>AND</b></li> <li>• adaptive cruise control at the same time</li> </ul>	<ul style="list-style-type: none"> <li>• traffic jam chauffeur</li> </ul>	<ul style="list-style-type: none"> <li>• local driverless taxi</li> <li>• pedals/steering wheel may or may not be installed</li> </ul>	<ul style="list-style-type: none"> <li>• same as level 4, but feature can drive everywhere in all conditions</li> </ul>

Figure 1 Levels of automation based on SAE J3016 (SAE International, 2021)

The capabilities of CAVs have the potential to transform the existing transportation system by making it more efficient, safe, and accessible for everyone, especially those with physical disabilities and older people. With the ability to communicate with each other and the infrastructure, CAVs can optimise traffic flow and reduce congestion, leading to faster and more efficient transportation. However, the development of CAVs also presents unique challenges,

such as immature technologies, non-coverage infrastructure, ESR issues (Ethics, Standards, Regulation), cybersecurity concerns, and public acceptance.

## **1.2 The current state and future trends of CAVs**

CAVs have been rapidly evolving over the last few years, with numerous OEMs and technology companies investing in developing an autonomous driving feature, such as Tesla's Autopilot and GM's Super Cruise, which is defined as level 2 in SAE standards due to the driver must take over the system when it requested. However, high-level autonomous vehicles (levels 3-5), which can operate without human intervention, are still being developed and tested on public roads in various locations worldwide, such as Waymo in Arizona, Cruise in San Francisco, Argo AI in the United States, and Baidu in China (Singh & Saini, 2021). Several emerging trends are expected to play a significant role in their development, such as implementing blockchain and AI to enhance cyber security (Bendiab et al., 2023), V2X communication, especially 5G-V2X (Lu et al., 2020), and cloud computing and resource allocation to increase real-time computational power (Danquah & Altılar, 2020; Nayak et al., 2022).

## **1.3 Project Motivations**

The development of CAVs presents a unique opportunity to enhance the transportation system's reliability, safety, and efficiency. To achieve this goal, the connectivity capability of CAVs is crucial. The ability of CAVs to communicate with other vehicles (V2V), infrastructure (V2I), and the cloud (V2C) can provide real-time information about traffic conditions and road hazards, enabling them to operate more efficiently and safely on the roads. In certain situations, such as adverse weather conditions, the onboard sensor information of CAVs may not be available or insufficient to ensure the vehicle's safe operation. In such cases, V2X communication (off-board information) can be a fail-safe plan to prevent negative consequences. However, several challenges must be tackled, such as real-time processing capability and connectivity protocol (DSRC or 5G-V2X) (Damaj et al., 2022).

## **1.4 Aim and Objectives**

This project aimed to develop a system that can detect the road layout and the presence of vehicles and pedestrians and guide a vehicle safely through a target lane or safe stop when it faces a car or pedestrian by using V2I communication information (off-board information) and the business plan for the designed system. To achieve this aim, several specific objectives have been identified which will contribute to developing the software system.

- To identify the local representation of the road layout.
- To automatically detect other vehicles and pedestrians.
- To plan the vehicle's speed based on the road's characteristics and other road users.
- To control the steering of the vehicle to follow the path.
- To develop a visual display to show the desired speed to the driver.
- To remotely control the vehicle with an onboard control unit from the off-board computer.
- To analyse the potential market and formulate the business plan for the designed system.

## 1.5 Hardware and software requirements

To achieve the main goal and objectives, the following hardware and software tools were provided:

- Land Rover Discovery with steer-by-wire feature
- Ouster OS1-64 LiDAR Sensor
- OXTS RT 3000 Inertial Navigation System
- ROS
- Simulation program

## 1.6 Scope and Limitations

The main focus of this project is developing software for V2I communication in highway scenarios, using only LiDAR data to meet the requirements mentioned above. It is important to note that the project is limited to scenarios where only one road actor is in the vehicle's immediate vicinity.

This project is limited by the type of sensor data that is available. Specifically, it is only possible to utilise point cloud data from LiDAR sensors.

## 1.7 Structure of the Report

The report begins with an extensive review of the relevant literature, which covers the different categories identified in the project objectives. Following this, the methodology employed by the group to achieve these objectives is detailed, along with an analysis of the results obtained. A market analysis and business plan for a suggested solution are provided, as well as an overview of the project management strategies used by the group to manage and allocate project tasks effectively. The report also discusses challenges encountered throughout the project, both technical and non-technical, as well as key lessons learned by group members — finally, a chapter on recommendations for future work.

# 2. Literature Review

## 2.1 Vehicle to Infrastructure (V2I)

V2I technology is a communication protocol that allows vehicles to transfer data with infrastructure such as traffic lights, road signs, and roadside units (RSUs). This technology is based on a wireless communication system that enables vehicles to transmit and receive data from the infrastructure in real-time. The primary components of the V2I system are the onboard units (OBUs) installed in vehicles and the roadside units (RSUs) installed in the infrastructure. The OBUs are designed to capture information from the vehicle's sensors and transmit it to the RSUs. Similarly, RSUs are designed to capture data from infrastructure-mounted sensors and transmit it to OBUs, as shown below (Kumar & Ali, 2022). Two well-known technologies transmit data between the OBUs and RSUs, such as DSRC (Dedicated Short-Range Communication) and C-V2X (Cellular-V2X). C-V2X technology obtains more attention than DSRC. While both technologies are designed to enable V2X communication, C-V2X has several advantages over DSRC, such as C-V2X can operate on the existing cellular network. C-V2X has higher reliability, making it a desirable option for many automotive and technology companies (Jellid & Mazri, 2021).



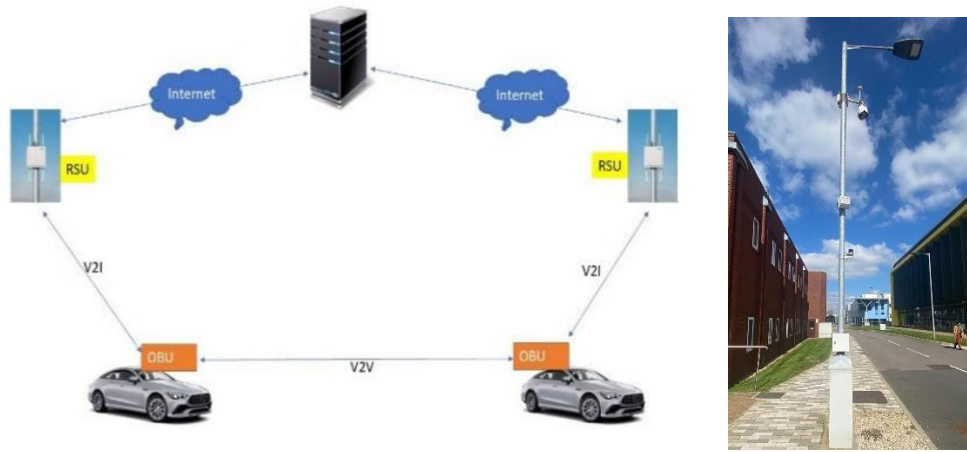


Figure 2 Left: High-level Architecture of V2I (Kumar & Ali, 2022), Right: RSU at MUEAVI

The advantages of V2I include increased safety, efficiency, mobility, and navigation enhancements. V2I can also aid in optimising traffic signals, reducing congestion, and enhancing the transportation system. In addition, it can provide real-time information about transit schedules, road conditions, and other pertinent information, enabling individuals to make more informed decisions regarding their mode of transportation (Malik et al., 2020).

## 2.2 LiDAR

Light Detection and Ranging, or LiDAR, is a remote sensing technology that uses lasers to measure distances and speed and generate highly accurate 3D maps of objects and environments called a point cloud, as shown in Figure 3-Left. The technology operates by emitting pulses of laser light reflected by the LiDAR sensor after bouncing off objects. The time it takes for the laser pulse to return to the sensor is used to determine the object's distance and speed from the LiDAR sensor, as shown in Figure 3-Right (Shenzhen da xue, 2018). By emitting and receiving these laser pulses rapidly, LiDAR sensors can generate highly detailed point clouds that can be used for various applications, such as autonomous vehicles, mapping, and environmental monitoring. Even though LiDAR can operate in low-light and low-visibility conditions, LiDAR is expensive and susceptible to interference from other light sources. Moreover, it has a limited field of view, especially on the vertical axis (Agishev et al., 2013).

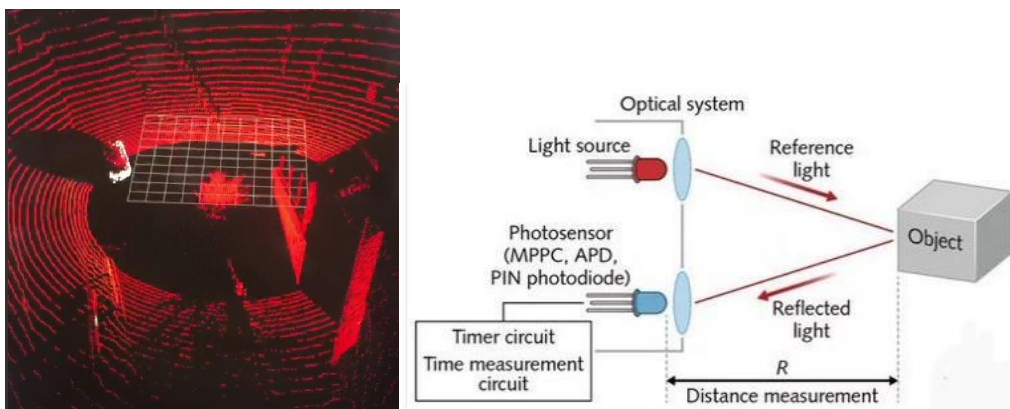


Figure 3 Left: Point Cloud of LiDAR, Right: LiDAR schematic architecture (Shenzhen da xue, 2018)

While onboard lidars are intended to offer extensive information about the vehicle's immediate surroundings, roadside lidars provide real-time data on traffic conditions and optimise traffic flow, making them an essential tool for traffic management and road safety.

## 2.3 Localisation

There have been several pieces of research on using LiDAR as a roadside sensor for real-time tracking of non-connected vehicles in mixed traffic conditions. In the first research, using a cost-effective LiDAR to detect the distance and speed of vehicles passing by is studied (Wu, n.d.) The process includes background filtering, lane identification, and vehicle tracking using Global Nearest Neighbour (GNN) algorithm. The results show that most vehicles can be detected except when occluded. The limit of this research is object classification. It does not include vehicle type identification, as the point clouds are not precise enough when the vehicle is far away. However, it suggests that regression methods can be used to estimate the vehicle type. Further research is also needed to distinguish vehicles, bicycles, and pedestrians because the LiDAR does not have enough beams to have an apparent object shape. They propose to detect the height of objects to try to classify them. However, for example, the new Land Rover Discovery has a height of 1.88 meters which can be the height of a pedestrian, so this proposal does not seem realisable. In parallel, as the current detection range of the LiDAR is limited to around 30m, new algorithms or multiple sensors on the road could expand the range, which seems interesting for further research.

The second research proposes and develops an image-based vehicle-tracking framework from roadside lidar data to track a vehicle's precise location and speed (Zhang et al., 2019). The vehicle detection process is quite like the first research. This study's added value is the breakdown of 3D point cloud clusters into 2D images relating to the plan and side views, and the use of point tracking instead of cluster tracking provides more stable results.

Above all, this shows that the three-step vehicle detection process is a widely used framework in autonomous vehicles. This process is subdivided into moving points filtering, clustering, and object classification.

Finally, 3D point cloud registration is essential for localisation in autonomous driving applications. Iterative Closest Point (ICP) and Normal Distribution Transform (NDT) algorithms are common. These are commonly used for 3D point cloud registration in robotics and autonomous driving applications. ICP is an iterative algorithm that seeks to minimise the distance between two-point clouds by iteratively refining the transformation between them. On the other hand, NDT is a probabilistic algorithm that models the point clouds using normal distributions and calculates the transformation that maximises the likelihood of the overlapping regions of the distributions. Both algorithms have been extensively studied and compared in the literature, with several variants proposed to address their limitations, such as sensitivity to initial conditions and computational efficiency (Magnusson et al., 2009).

## 2.4 Object Detection

Most roadside LiDAR detection methods use one LiDAR with a limited field of view and imperfect point cloud data. Collaboration between LiDAR point clouds improves perception accuracy. LiDAR object detection and classification uses traditional machine-learning and deep-learning methods. Some studies use LiDAR and camera sensors to improve object detection and classification. Pre-processing modules can improve radar-based classification performance (Arikumar et al., 2022; Sallab, 2018).

Clustering is a robust machine-learning algorithm for roadside LiDAR object detection and classification. DBSCAN and Euclidean clustering algorithms group nearby LiDAR points into individual objects to detect and classify cars, pedestrians, and cyclists. These algorithms consider point cloud data's spatial and temporal information and can handle noise and different

object densities. (Zhang et al., 2020) used a modified DBSCAN to classify vehicles and pedestrians in the detection range using a backpropagation artificial neural network (BP-ANN) classification model. Background filtering, object clustering, object classification, and object tracking have been suggested for processing roadside LiDAR data unsupervised (Sun et al., 2022).

Clustering is a popular unsupervised learning technique in computer vision and machine learning. It is used to group data points with similar characteristics into clusters. Mean Shift is a popular clustering algorithm, a non-parametric technique that does not require a priori assumptions about the number or size of clusters. Mean Shift works well for analysing various data distributions but is mathematically slow and expensive, especially for large data sets. (ML | Mean-Shift Clustering - GeeksforGeeks, n.d.).

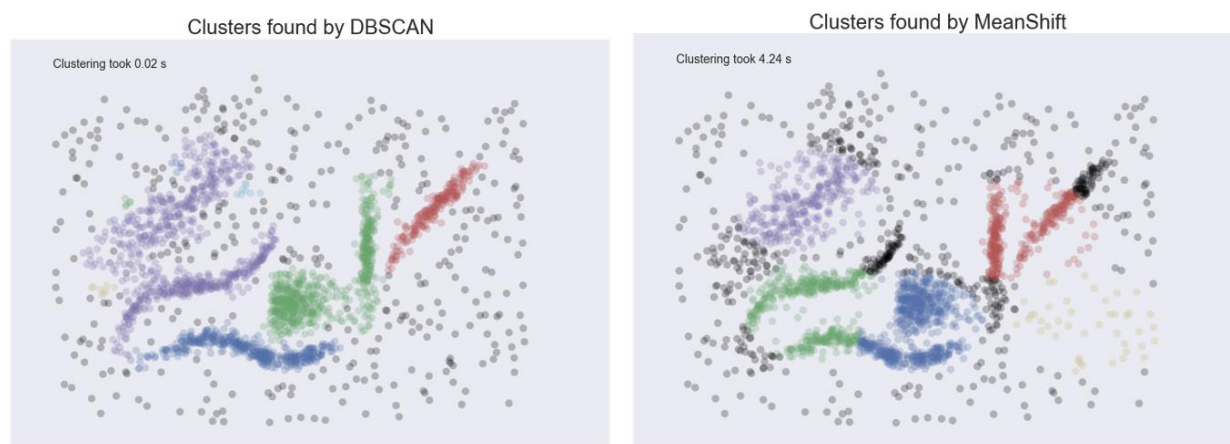


Figure 4 Results of various algorithms for clustering (Comparing Python Clustering Algorithms — Hdbscan 0.8.1 Documentation, n.d.).

DBSCAN, which stands for Density-Based Spatial Clustering of Applications with Noise, is another popular clustering algorithm. DBSCAN is a density-based algorithm that separates distant or isolated points and merges nearby points. This is especially useful for datasets with varying densities, as it allows for discovering clusters of any size. However, DBSCAN has issues with high-dimensional data, necessitating careful parameter tuning. Euclidean clustering, known as k-means clustering, is a popular point cloud data clustering algorithm. It assigns point clusters based on their proximity in Euclidean space. Because Euclidean clustering is fast and efficient, it is well-suited for real-time applications. It can also handle large amounts of data and is less sensitive to the cluster's initial position. However, it assumes that the clusters are spherical and uniform in size, which may not be suitable for all data types (Comparing Python Clustering Algorithms — Hdbscan 0.8.1 Documentation, n.d.).

Point cloud data can be processed using various techniques, including filtering, registration, and segmentation. Filtering techniques are used to remove noise and outliers from the point cloud data, while registration techniques align multiple point clouds to create a complete 3D model. Segmentation techniques divide the point cloud into meaningful segments. There are two different methods of creating a point cloud, LiDAR and photogrammetry, with LiDAR laser scanning having increased accuracy. Workflows for processing point cloud data include integrating data types, pre-processing, cleaning and restructuring a point cloud, and extracting or filtering specific features (Wang et al., 2020).

LiDAR point cloud data has several limitations and challenges associated with obstacle detection, including issues related to data quality, processing time, and accuracy. LiDAR sensors can be affected by weather conditions, such as rain and fog, which can affect the accuracy of the data. Processing LiDAR point cloud data can also be time-consuming and computationally intensive. Future research in this area could focus on developing new

algorithms and techniques to improve the accuracy and efficiency of obstacle detection using LiDAR point cloud data. This could include developing new filtering and segmentation techniques and improving the accuracy of object classification. Additionally, researchers could investigate using LiDAR data from multiple sensors placed on poles for obstacle detection and classification (Mohapatra et al., 2021).

## 2.5 Path Planning

### 2.5.1 An Overview of Path Planning

Path planning is vital for any autonomous vehicle to navigate safely within a dynamic environment. It is an essential building block between the localisation and path-tracking controller. In systems like UGV's path planning involves creating the most optimised and feasible waypoints considering the vehicle pose, orientation, speed and obstacles. Several path-planning algorithms have been developed for UGVs, each with advantages and limitations.

### 2.5.2 Sample of Potential Solutions

Some of the most used path planning algorithms for UGVs include Dijkstra's A\* algorithm and rapidly exploring random trees (RRT). Grid-based approaches are widely used for offline maps, and accurate information about the vehicle's position is essential. GPS-based approaches may not provide enough accuracy, so pre-generated point cloud maps recorded from live LiDAR data are often used instead (Ji et al., 2017).

Based on research conducted by Sariff & Buniyamin, several algorithms are used to find the optimal path in path planning. A\* is a popular heuristic algorithm that minimises the function  $f(n) = g(n) + h(n)$  to find the optimal path. Genetic algorithm (GA) is another heuristic approach that imitates the evolutionary process to acquire the best individuals (chromosomes) representing the optimal path (Sariff & Buniyamin, 2006). Another alternative method could be the Rapidly Exploring Random Tree (RRT), a probabilistic algorithm for finding paths in non-convex and high-dimensional spaces.

There are several local path planning algorithms, and the most used are Dynamic Window Approach (DWA) and Timed Elastic Band (TEB). Although most of the research is done on holonomic robots, and both perform similarly under certain cases, the TEB planner is much more suitable for Ackermann-type non-holonomic robots (Naotunna & Wongratanaphisan, 2020). Also, the generated trajectory is smoother with fewer oscillations but is susceptible to dynamic obstacles and may not be able to generate an optimised path (Marin-Plaza et al., 2018).

## 2.6 Path Following

### 2.6.1 An Overview of Path Following

Path following is a fundamental problem in the field of unmanned ground vehicle (UGV) navigation. In the context of a car-like UGV, path following involves controlling the vehicle's steering and speed to accurately follow a predefined path while avoiding obstacles and maintaining stability. Over the years, various path-tracking methods have been developed, each with its advantages and limitations.

The earliest path-tracking methods were based on proportional-integral-derivative (PID) controllers, which use feedback to minimise the error between the actual and desired trajectory,



as Pure Pursuit or Stanley controllers (Amer et al., 2017). Subsequently, researchers developed more advanced control methods, such as model predictive control (MPC) (Klauer et al., 2020) and sliding mode control (SMC) (Rokonuzzaman et al., 2021), which offered better tracking performance and robustness. In recent years, machine learning (ML) and artificial intelligence (AI) methods have also been applied to path tracking (Kebbaty et al., 2022).

### 2.6.2 Sample of Potential Solutions

A brief selection of the different path-tracking methods that could be implemented in the project has been made. They are listed below with a brief description.

- **Model Predictive Control (MPC):** MPC predicts the future position and orientation of the vehicle based on the current state and generates a steering command that minimises the error between the predicted path and the desired path. This algorithm is often used in applications where the vehicle must follow a complex or dynamic path (Kebbaty et al., 2022; Rokonuzzaman et al., 2021).
- **Stanley Control:** This algorithm is based on controlling the vehicle's lateral position relative to the desired path rather than directly controlling the vehicle's heading. In this method, the vehicle's control system calculates the cross-track error (CTE). Then, the steering input is adjusted based on the CTE and the vehicle's current heading to bring the vehicle back onto the path. The Stanley method has been shown to be effective in handling sharp turns and discontinuous paths (Kebbaty et al., 2022; Rokonuzzaman et al., 2021).
- **Pure Pursuit:** This method is a geometric control approach that determines the appropriate steering angle and speed for an autonomous vehicle to follow a desired path based on the vehicle's current position and heading. It selects a point on the path, called the lookahead point, and computes the required steering angle and speed to move towards that point. The lookahead point is continuously updated to maintain a constant distance between the vehicle and the desired path. This method is popular due to its computational efficiency and ease of implementation (Hung et al., 2023; Mohd Shamsuddin et al., 2021; Paden et al., 2016).

## 2.7 Vehicle Remotely Control

### 2.7.1 Ackermann Message

The Ackermann message is a specific type of message used in the Robot Operating System (ROS) to control mobile robots with Ackermann steering. It contains information about the desired steering angle and speed of the robot, which is used by the robot's control system to adjust the rotation of the wheels and the speed of the propulsion system, allowing the robot to move in the desired direction and turn smoothly. The Ackermann message is intended to provide a standard method for controlling robots with Ackermann steering in ROS. This enables more manageable and efficient communication and collaboration between software components, as they can all use the same message format to send steering and speed commands to the robot (Ackermann\_Msgs - ROS Wiki, n.d.).

### 2.7.2 GUI

GUI (Graphical User Interface) provides essential information, safety alerts, entertainment options, and communication tools, making the driving experience more comfortable and enjoyable. This visual representation allows developers to better understand how the autonomous vehicle will interact with its environment and to identify potential issues or areas for

improvement. It allows users to interact with the simulation by inputting commands or adjusting settings.

## 2.8 Verification and Validation

Verification and validation are two essential procedures for ensuring the quality of a product or system. Verification determines whether the software or system is constructed according to the design's specifications, standards, and requirements. The purpose of verification is to ensure that the product satisfies the desired quality standards and functions as intended. Validation is the process of determining whether the software or system fulfils the needs and specifications of the user. The purpose of validation is to ensure that the product serves its intended purpose and meets the user's expectations. In other words, validation answers the question, "Are we building the right product?" It ensures that the software or system solves the appropriate problem and meets the requirements of its intended. The process involves unit, integration, and system-level tests, which will be discussed in detail in upcoming sections.

To ensure CAVs' safety, reliability, and responsiveness, validating and verifying them in simulation before testing in the real world is crucial. Developing such a system is a complex process with the integration of several components, which can result in unpredictable behaviour. Thus, V&V provides the necessary platform to identify the system's performance and limitations. Simulation provides a risk-free environment for testing a vehicle's behaviour in various scenarios and conditions without endangering people or property. In addition, it permits testing in scenarios that may not be feasible or applicable in the actual world. Validating and verifying in simulation can aid in identifying and mitigating software and hardware safety risks and defects, saving time and resources. This project uses CARLA and Roadrunner with ROS integration as verification and validation tool.

## 3. Methodology

### 3.1 Localisation

The Localisation block is vital in enabling an autonomous driving vehicle to comprehend its location and that of other road users. This mapping lets the vehicle make informed decisions regarding its following action. For specific applications involving point cloud data, such as point cloud registration, point cloud transformation, down-sampling, segmentation, and filtering, this project employs methods provided by the PCL library(Point Cloud Library).

The tasks of the Localisation block can be classified into two categories: offline and real-time. Speed is a top priority for real-time tasks, while the algorithm's performance gains greater importance for offline tasks. We distinguish between designing a system that better aligns with our priorities.

The offline phase consists of point cloud registration. In contrast, the real-time phase involves pre-processing, moving vehicle extraction, clustering, bounding box generation, and ego vehicle detection blocks within the system design. The obstacle detection block at the end of the Localisation block utilises the ego vehicle detection block outputs and the bounding box generation block results to detect obstacles, as seen in the Figure below. The following steps provide a comprehensive description of this process.

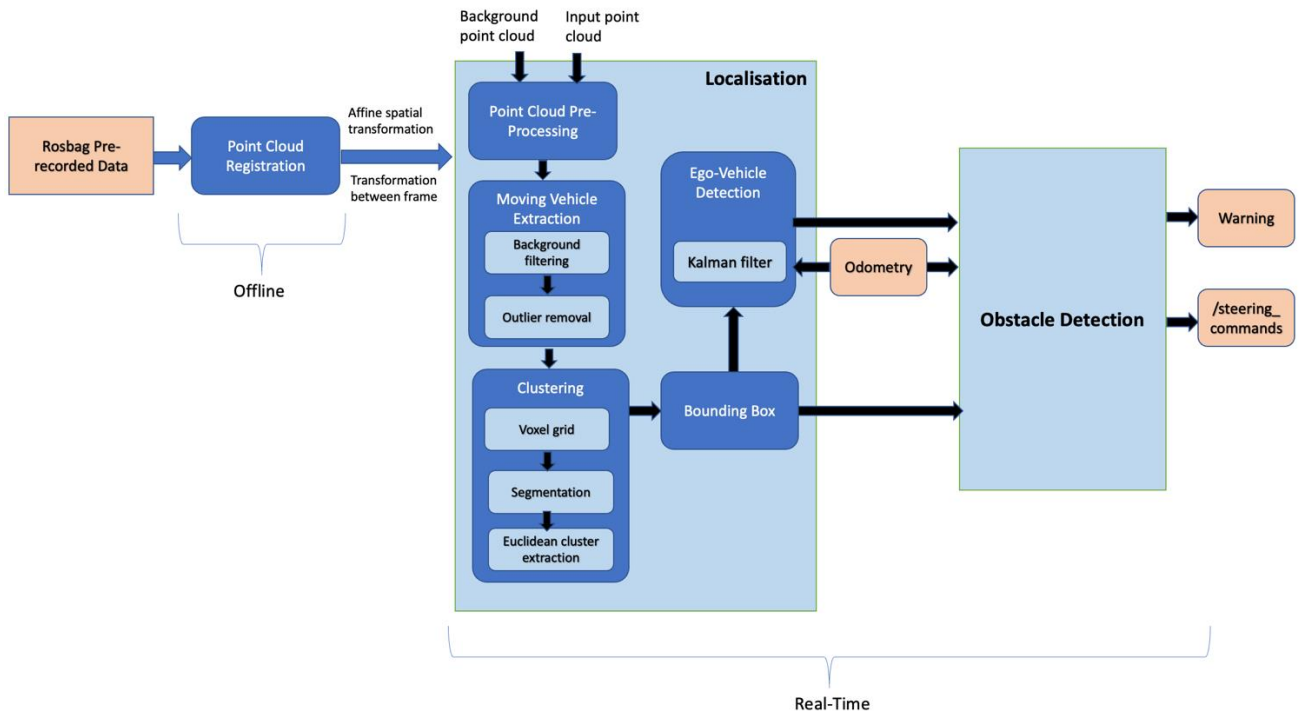


Figure 5 High-Level System Design for Localisation and Obstacle Detection

### 3.1.1 Offline Step

#### 3.1.1.1 Point Cloud Registration

The limited range of LiDAR sensors necessitates deploying multiple sensors for offroad lidar-based autonomous navigation systems, which introduces the challenge of aligning their frames. To integrate multiple sensors with different locations and rotations to the same coordinate frame, it is essential first to understand the frame of each sensor relative to the common frame. This challenge, known as Point Cloud Registration, is prevalent in various robotics and computer vision applications. Registration aims to align sets of points in different coordinate systems by determining a transformation that minimises the difference between them.

To address the issue of local minima, the ICP algorithm was executed with multiple initialisation points. Initially, CloudCompare was used to approximate the LiDAR sensor positions on the map, which were subtracted from the zero point of the map. The point cloud for each LiDAR sensor was then translated using its approximate position, and the ICP and NDT algorithm was applied to determine the affine spatial transformation matrix.

### 3.1.2 Real-time Step

#### 3.1.2.1 Point Cloud Pre-Processing

Performing KD-Tree search within a point cloud for segmentation is computationally intensive. Additionally, point clouds far from the LiDAR sensor's origin are less dense and inaccurate. Therefore, it is efficient to perform calculation filtering only the areas required for the tasks. To achieve this, we used the Passthrough filter provided by PCL, which is a function that sets the minimum and maximum regions of interest in XYZ and outputs only point clouds within these regions. Although the working principle of this filter is relatively simple, there is a disadvantage in that it is not sophisticated in setting the region of interest. Before conducting main tasks such

as vehicle pose estimation and moving object extraction, processing large amounts of LiDAR sensor data is required, which can be time-consuming and costly. Therefore, computational efficiency is crucial, particularly for real-time applications. Downsampling of point clouds can be a solution to improve computational efficiency by reducing the number of points. However, it is important to upsample the data before object detection to prevent degradation in detection and classification tasks.

### 3.1.2 Moving Vehicle Extraction

#### 3.1.2.1 Background Filtering

In each frame of a point cloud dataset, several static objects exist, such as the ground surface, infrastructure, and trees, which can be considered the background. To extract the background point cloud from the original data, it is necessary to select a certain number of frames that contain background points. For this purpose, we utilised the `pcl::SegmentDifferences` function to filter the background and obtain the moving object. Multiple background point clouds were selected and compared to evaluate performance based on the visualisation of the extracted moving object. To efficiently solve the nearest-neighbour search problem required by `pcl::SegmentDifferences`, we set the search method to the KD-Tree algorithm, which is widely used in 3D applications.

The `pcl::SegmentDifferences` algorithm includes a distance threshold parameter, where increasing its value results in fewer but larger segments. This approach can lead to the inclusion of noise into segments, the loss of essential details, and increased computational time due to the inclusion of more points. On the other hand, choosing a minimal value for the distance threshold can result in more segments, higher precision, and lower computational time. However, it may be sensitive to noise and result in difficulties capturing the entire object. Therefore, we carefully selected an appropriate value for the distance threshold based on the requirements of our application. Finally, we extracted background point clouds for each sensor from the rosbag files using the `pcl::SegmentDifferences` function.

#### 3.1.2.2 Outlier Removal

We utilised the `pcl::RadiusOutlierRemoval` filter from PCL, which removes outliers if the number of neighbours in a specific search radius is smaller than a given value. Initially, we observed many outliers, as depicted in the data below. However, by increasing the parameter of `RadiusOutlierRemoval`, we could effectively eliminate the noisy clusters.

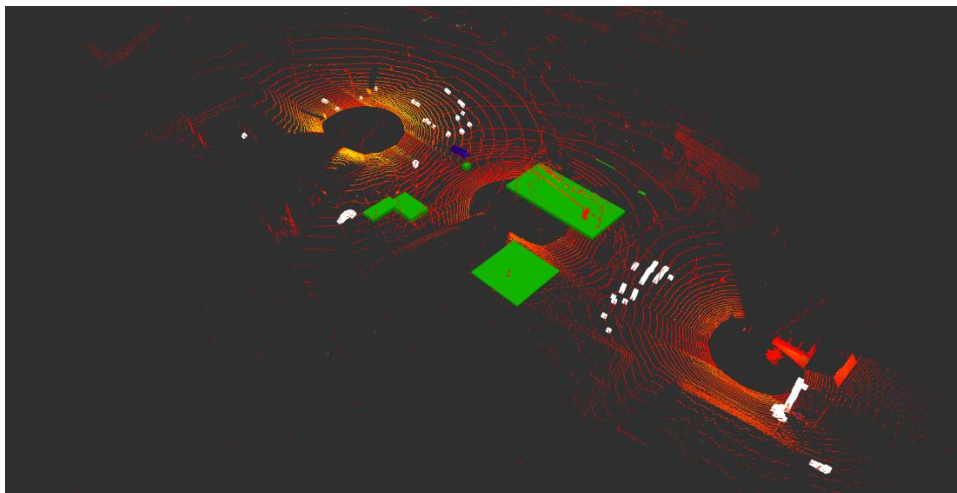


Figure 6 Noisy Data Without Outlier Removal



### 3.1.3 Clustering

#### 3.1.3.1 Downsampling with Voxel Grid

Voxel grid filters downsample and regularise point cloud data. The filter divides the point cloud data into small cubic cells called voxels and computes a centroid for each. The point cloud has fewer points because each voxel has one point. The voxel grid filter receives a point cloud as a set of (x, y, z) coordinates, each representing a point in 3D space. The filter produces a down-sampled point cloud with fewer points than the input. Voxel grid filters simplify point cloud processing algorithms and create uniformly down-sampled point clouds (Hacinecipoglu et al., 2020).

In its algorithm, Voxel Grid takes input as current point cloud data and converts it into filtered point cloud data. Here it applies the leaf size filtering parameter for down-sampling of the point cloud. The voxel grid filter has several parameters that can be adjusted to control its behaviour. The minimum number of points parameter specifies the minimum number required for centroid computation in a voxel, which helps avoid noisy centroids in voxels with few points. The down-sampling method parameter determines the voxel centroid computation method, usually the average of all points or the centre.

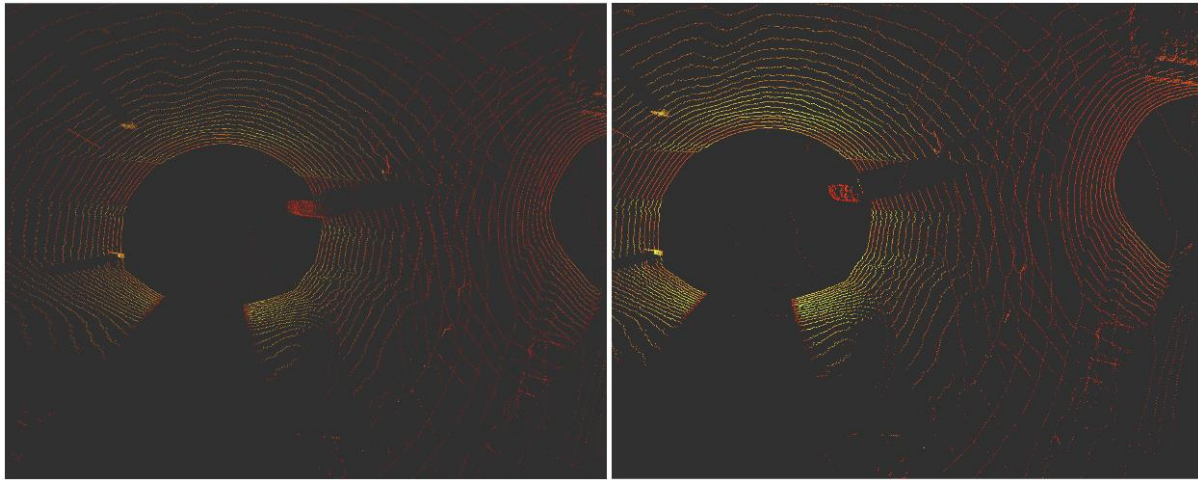


Figure 7 Left. Before applying the voxel grid filter vehicle points are dense, Right. After Voxel grid filter points become less dense.

The voxel grid filter has several advantages, including down-sampling, regularisation, and noise reduction. Down-sampling reduces the number of points in a point cloud, which can improve data compression, processing, and visualisation. Regularisation can be helpful for downstream applications that need structured input. The centroid computation in the voxel grid filter smoothens point position variations, reducing point cloud noise. However, the main drawback of the voxel grid filter is that it can reduce point cloud detail by representing multiple points in a voxel as a centroid. Choosing appropriate parameters is also important because the filter output is affected by voxel size and point count (Hacinecipoglu et al., 2020).

#### 3.1.3.2 Segmentation

Cluster segmentation is a crucial process employed in point cloud data analysis to group together points corresponding to the same object or surface. The main objective is to extract significant information from the point cloud data by separating different objects or surfaces. The input to the cluster segmentation algorithm is typically a 2D or 3D point cloud, represented as

an array of points with x, y, and z coordinates, along with additional information such as colour, intensity, or reflectance values (Cao et al., 2022).

The process of cluster segmentation involves several steps. Initially, a KdTree object is created to facilitate the search method of execution, followed by the determination of the threshold distance and tolerance of the cluster. Using specific parameters, the code then employs Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering on the point cloud. Once clustering is complete, the number of clusters is determined by counting the unique labels in the output, and each cluster is assigned a colour for visualisation purposes. Finally, a new point cloud object (pcd\_clustered) containing the clustered points with their assigned colours is created and returned as the output of the cluster segmentation code.

The applications of cluster segmentation are diverse and encompass areas such as robotics, autonomous vehicles, and augmented reality. It is widely used for object recognition, tracking, and classification, enabling the distinction of objects in a scene and simplifying data analysis. However, cluster segmentation can be computationally expensive, especially for large point clouds. Additionally, the accuracy of segmentation may be influenced by the distance threshold and minimum cluster size, which necessitates proper parameter selection for accurate and efficient cluster segmentation. Despite these limitations, the effectiveness of cluster segmentation in object recognition and classification has made it a popular tool in point cloud processing.

### **3.1.3.3 Euclidean Cluster Extraction**

The EU cluster extraction algorithm divides a point cloud into manageable clusters that can be analysed and processed separately, benefiting applications such as object recognition, scene reconstruction, and robot navigation. Based on the Euclidean distance metric, the algorithm clusters point that are close in space. Specifically, it identifies neighbouring points within a given radius from a seed point and considers all points within this radius to belong to the same cluster. This process is repeated for each point in the point cloud until all points are clustered into groups with points from the same object or surface (Piernik & Morzy, 2021).

The distance between every pair of points in the point cloud is first calculated to implement the EU cluster extraction algorithm. Then, for each point, all neighbouring points on the far boundary are identified. For each group of adjacent points, the algorithm checks if the number of points in the group exceeds the cluster size threshold and assigns them to a new group if so. This process is repeated for all unspecified points assigned to a group. The two main parameters that must be selected are the distance threshold and cluster size threshold, which determine the maximum distance between neighbouring points and the minimum number of points required for a group to be considered a cluster, respectively. The optimal values of these parameters depend on the characteristics of the point cloud and the specific application (Xu & Wunsch, 2005).

The EU cluster extraction algorithm has several advantages. First, it is easy to use and computationally efficient, making it suitable for real-time applications. Second, it is effective for separating adjacent objects in 3D space. Third, it is robust to noise and peripherals, making it suitable for use with noisy sensor data. However, the algorithm has certain limitations, including sensitivity to the choice of distance and cluster size parameters and potential difficulties in segmenting point clouds with complex shapes or many overlapping objects. Despite these limitations, the EU cluster extraction algorithm remains a widely used technique in point cloud processing, primarily due to its effectiveness in object recognition and classification (Xu & Wunsch, 2005).

### 3.1.4 Bounding Box Generation

Following the successful clustering of different objects, each object is assigned to a bounding box by calculating the minimum and maximum points of the clusters in each coordinate in 3-D space. Subsequently, the size of the bounding boxes is determined by computing the difference between the minimum and maximum points. Finally, it is assumed that the object's position is at the centre of the bounding box.

### 3.1.5 Ego Vehicle Detection

The final stage of the localisation block involves processing all bounding boxes generated in the preceding stage, which entails computing the minimum and maximum points of moving object clusters. Here, the position of each bounding box, specifically its centre, is compared with the odometry data. The closest bounding box to the odometry data is then designated as the "ego vehicle".

To mitigate the absence of bounding boxes within a certain threshold, a Kalman filter was employed. The filter is a recursive algorithm that leverages noisy measurements and a dynamic model of the system to estimate the state of the system at each time step. In a 2-D system with a constant velocity model, the system's state is represented by a four-dimensional vector containing position and velocity in the x and y directions. The measurement vector is composed of the position in the x and y directions obtained from the LiDAR sensor and the velocity in the x and y directions from the odometry. Notably, the availability of LiDAR data is not constant in some cases.

When LiDAR data is unavailable, the Kalman filter predicts the state based on the previous state estimate and odometry data. The filter algorithm has two primary steps: prediction and update. In the prediction step, the current state estimate and covariance matrix are projected based on the previous state estimate and covariance matrix and the system's dynamic model. In the update step, the projected state estimate and covariance matrix are corrected based on the measurement data.

The update step involves adjusting the Kalman gain, with the measurement matrix being a 2x4 matrix that selects the position component of the state vector. The covariance matrices for the LiDAR and odometry data are also determined, with the LiDAR data's covariance matrix having smaller values due to its greater accuracy.

### 3.1.6 Matching the Mesh of Objects

Once the point cloud is filtered through the Euclidian cluster, we are supposed to have only the points corresponding to our vehicle. Now, we want to estimate the vehicle's location in relation to the LiDAR, as we have also previously calculated the GPS coordinates of the LiDARs. We could then determine the GPS coordinates at which our vehicle is located. First, we proceeded to set a boundary box on the output cluster and calculate the position by taking the centroid of this box as the reference point for our vehicle. We wanted to go further than that by considering that the LiDAR data does not always see the vehicle's shape. In fact, at each frame, the point cloud detected matches the front, the side, or the rear of the vehicle, so the centroid of the boundary box is determined each time at a different point in our vehicle.

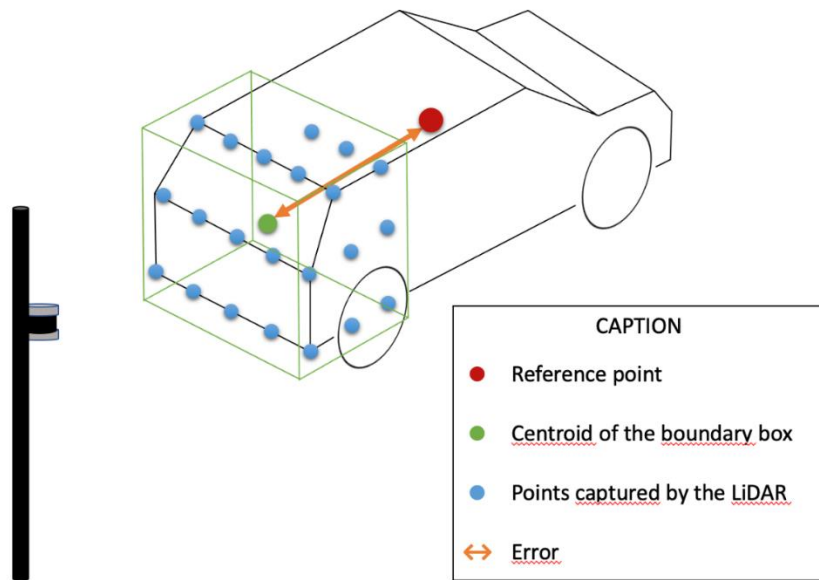


Figure 8 Scheme of the error between the centroid of the boundary box and the real reference point of the vehicle

To counterbalance the error in this approximation, a process that has already been tested is matching the mesh of the vehicle searched to the cluster. We can then approximate a more realistic boundary that considers the whole car. To do that, we can use the PCL library that integrates a function that proceeds to Normal Distribution Transform. Basically, the function `pcl::NormalDistributionTransform` takes as an input a PCD file representing the vehicle mesh and the PCD file of the cluster detected by the LiDAR. We also need to provide parameters such as epsilon that represents the maximum transformation we want to calculate and the initial guess, which is a first approximation of the mesh position we want to refine.

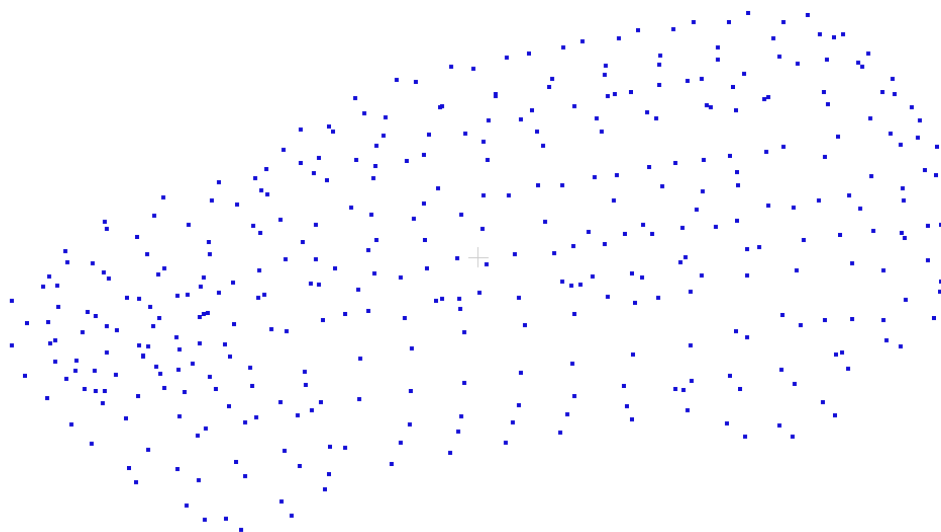


Figure 9 Point cloud generated from the mesh of Land Rover Discovery 4

## 3.2 Object Detection and Classification

The Obstacle Detection subsystem is designed to receive input from three sources: odometry data, ego vehicle detection information published by the Ego Vehicle Detection subsystem, and bounding box data published by the Bounding Box subsystem.

The subsystem incorporates an obstacle threshold distance, determining when a warning should be issued. Specifically, a warning is triggered only when an object is detected in front of the ego vehicle within the predetermined lateral range. The distance between the detected object and the ego vehicle is less than or equal to the threshold distance.

Various techniques can be employed to classify objects in a 3D points cloud, such as machine learning, deep learning, or rule-based methods. Machine learning and deep learning approaches involve training algorithms on labelled data to learn the characteristics and features of different objects. Rule-based methods, on the other hand, utilise a set of predefined rules to classify objects based on their geometric properties, such as size and shape.

Object classification can be performed at different levels of granularity, such as at the point level, where each point in the point cloud is assigned a label, or at the object level, where objects in the point cloud are segmented and labelled as a whole. Object classification accuracy is highly dependent on the quality of the point cloud data, the choice of feature extraction methods, and the selection of appropriate classification algorithms.

## 3.3 Path Planning

The path planning algorithm is expected to generate the path in a known environment. The vehicle must navigate safely by updating the path when facing any obstacles. Several path-planning algorithms are available, but due to limited time and scope of development, we chose the ROS navigation `move_base` package (ROS.org, n.d. -c). This approach provides flexibility and enables the vehicle to find the optimised path in the environment.

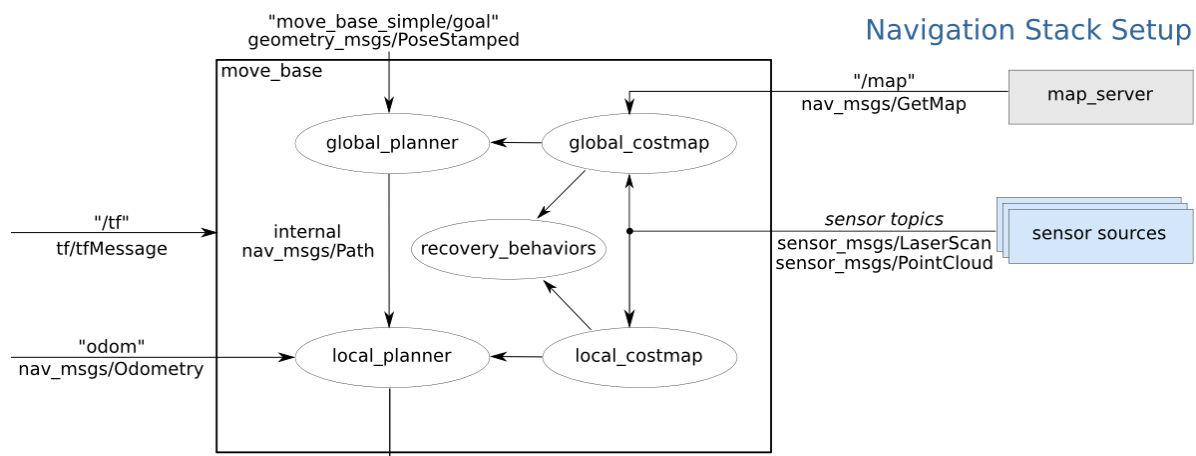


Figure 10 ROS `move_base` navigation stack (ROS.org, n.d. -c)

It is essential to localise the vehicle in the environment for optimised path generation. Although several libraries have ROS compatibility, only a few are designed specifically for non-holonomic ground vehicles and even fewer are maintained. The brief steps involved in the path of the following algorithm are given below.

1. A pre-generated map of the environment, which is created using SLAM (Simultaneous Localisation and Mapping) algorithms such as gmapping, which uses LIDAR point cloud



data to create a 2D occupancy grid map of the environment (ROS.org, n.d.-a) (refer to the above Figure).

2. Loading the map into the ROS environment using the map\_server package provides a ROS interface to the occupancy grid map (ROS.org, n.d.-b).
3. Costmap generation represents the environment that considers the obstacles and other features that may affect vehicle motion.
  - a. Global costmap (defining the parameters of the layers)
  - b. Local costmap (defining the parameters of the layers)
4. Path generation - The plan generated in the move\_base package consists of waypoints of both the global  $P_i = (x_i, y_i)^T$  and local  $P_i = (x_i, y_i, \theta_i)^T$  path planning.
  - a. Global planner: The global path planner generates a high-level plan that takes the robot from its current location to its destination. The global planner typically operates on a map of the environment and generates a plan comprising a series of waypoints. In this project, the Dijkstra algorithm is used because of the simplicity and ease of integration with the vehicle controller.
  - b. Local planner: The local planner plays a crucial role in transforming the high-level global path into a series of feasible waypoints that accounts for dynamic obstacles and vehicle configuration such as dimensions, orientation, kinematics, etc. Timed Elastic Band (teb\_local\_planner) package (ROS.org, n.d.-d) is used for local path planning.

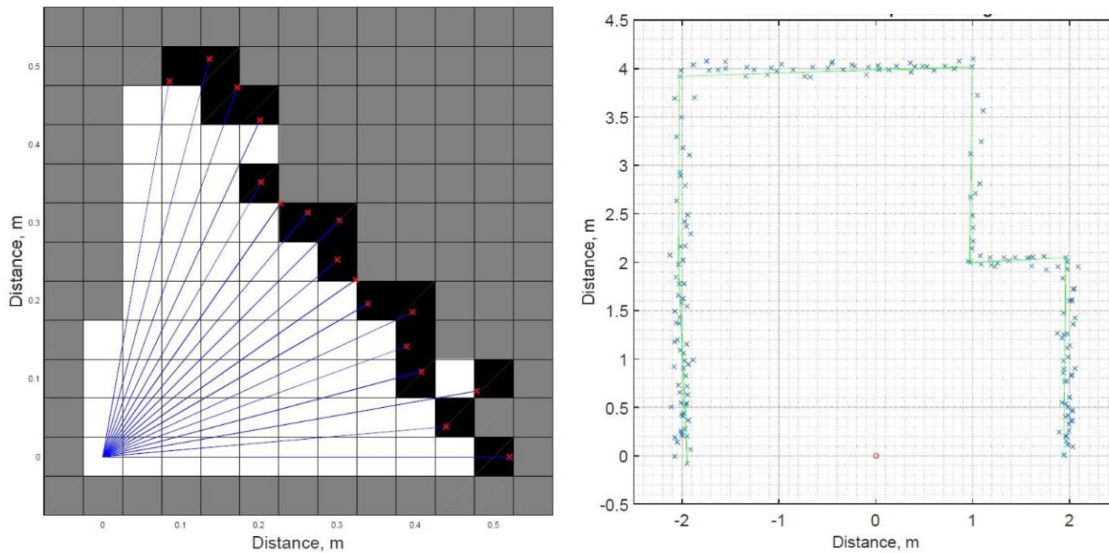


Figure 11 Occupancy grid map generation using point cloud data (Gurel, 2018).

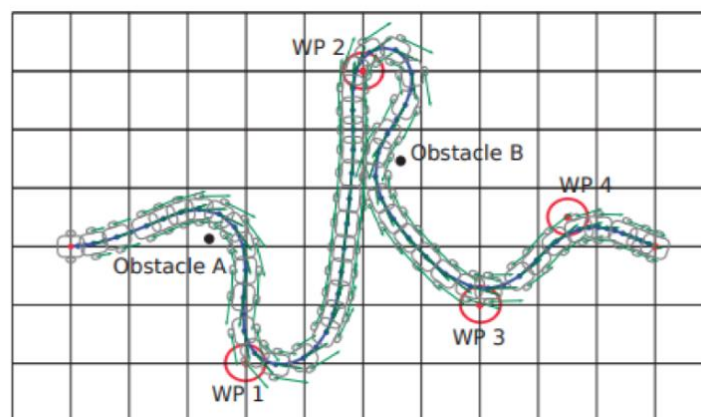


Figure 12 teb\_local\_planner description (Marin-Plaza et al., 2018).

Thus, the parameters are set up for the steps mentioned above, and finally, feasible waypoints are generated, which are then published to the path following the controller. This approach will enable the system to accommodate any environment and provides a robust path.

### 3.4 Path Following

To achieve good control, it is necessary to have an accurate kinematic model of the vehicle. For this project, the simplified Ackermann model has been used.

The Ackermann vehicle model is a basic representation of a vehicle's motion that considers steering control for the front two tires, but it doesn't account for factors like mechanical stability and wear items. Despite this, it's a good approximation for control and predictive purposes. The simpler bicycle model is often used for control, combining the left and right tires into a "virtual" bicycle with a front and rear tire at the centre of the vehicle's longitudinal axis, as below.

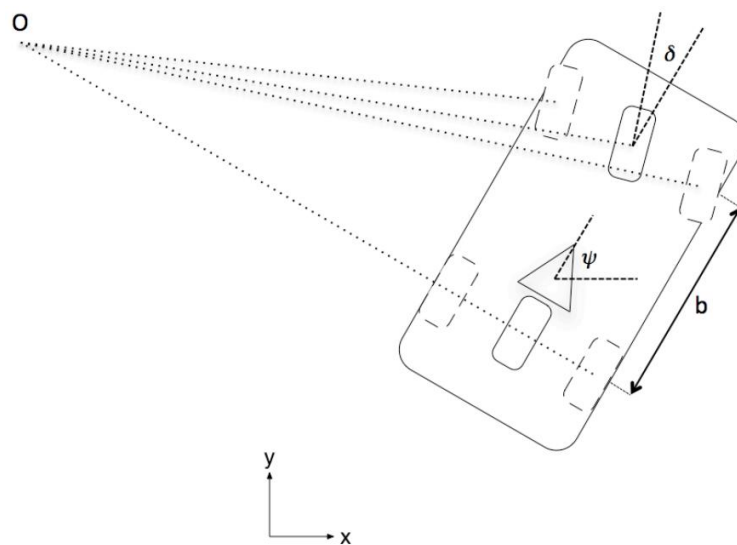


Figure 13 The Ackermann Model, with a simplified bicycle model (Sprinkle et al., 2008).

Commented the kinematics of the vehicle, we have chosen to implement the Pure Pursuit controller. This controller has been chosen because it is easy to implement and is efficient for the conditions of the project. Additionally, due to the limited time available, it is believed to be a good approach to start with.

To implement this controller, 3 python codes must run in the ROS environment. The first reads the file where the path is stored using waypoints and looks for the index of the nearest waypoint at a lookahead distance from the vehicle. This code publishes the index of that point and the index of the last waypoint on the path in 2 different topics.

The second code picks up the published indexes and searches within the trajectory file for the coordinates that correspond to both indexes. It then publishes the coordinates in a new topic.

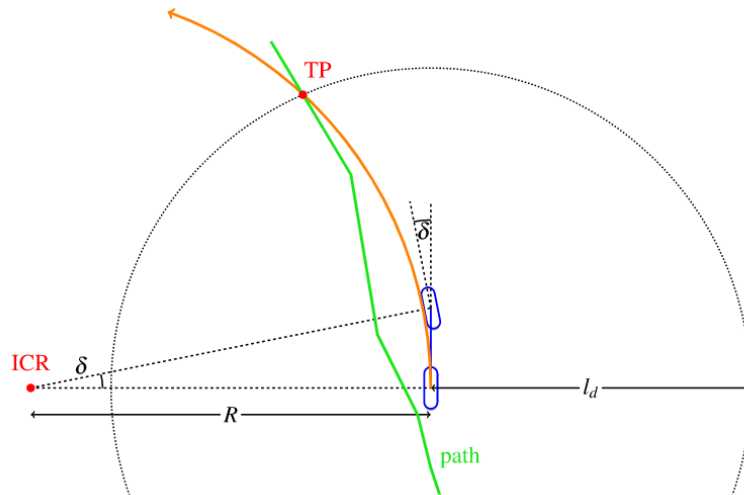


Figure 14 Pure Pursuit controller representation for a bicycle model (Theers & Singh, 2022).

Finally, the last code is the controller. This sends the speed and steering angle commands to the vehicle. It calculates the angle error considering the current position and orientation of the vehicle and the position of the next waypoint (obtained in the second code). It then divides the obtained angle by the maximum angle of rotation of the wheels. In this way, the steering angle command is obtained gradually. In addition, it calculates on which side the goal is located to know which way to turn.

On the other hand, it calculates the speed at which the vehicle should go in relation to the steering angle command. When the steering angle command is 0, the speed command is maximum and vice versa, changing the value gradually. For this purpose, a maximum and minimum speed different from 0 has been set.

Lastly, the speed command will be 0 when the vehicle is at a predetermined distance from the last waypoint, causing it to stop at the end of the path.

## 3.5 Remotely Control

### 3.5.1 GUI

Numerous methods were discovered when developing the webpage that displays the results of vehicle speed, steering angle, and obstacle recognition. The crew opted to use ROS gauges to develop the website based on the literature study that was done. An extra protocol was introduced to integrate ROS gauges. The ROS software framework RQT supports a variety of GUI widgets as plugins. In addition, RQT contains a variety of other forms of ROS data, such as gauges, charts, and 3D models. ROS nodes were developed and subscribed to for the vehicle's speed, obstacle detection, and steering angle.

To assist in visualising the gauges on a web browser, an HTML code was developed for the web page's development. In the beginning, there were several designs, but eventually, there was only one left. Using the integrated RQT web server and other tools, there were other webpage creation methods. The ability to develop interactive webpages that can take user orders and transfer them to the ROS system was one advantage noted during contact with RQT, but HTML was chosen due to familiarity. After the compilation, the website correctly displayed the data obtained from the ROS Topics.



### 3.5.2 Steering Command

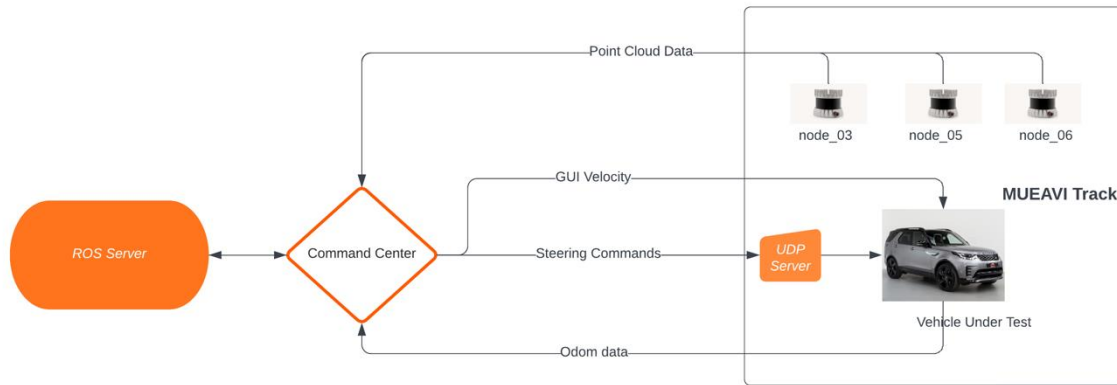


Figure 15 High level flow chart of system integration

The Ackermann drive is a steering system that improves manoeuvrability by ensuring all four wheels remain in contact with the ground during turns. It uses a modified four-wheel system and steering geometry to prevent slipping or skidding.

In ROS, the Ackermann drive is implemented using the `ackermann_msgs` message type, which includes fields for steering angle and speed, as well as for wheelbase, which is the distance between the front and rear axles of the vehicle. The steering angle is typically specified in radians, while the speed is specified in meters per second.

The formula is:

$$\tan(\delta) = \frac{L}{\left(R + \frac{W}{2}\right)}$$

where:

$\delta$  is the turning angle of the front wheels.

L is the wheelbase.

R is the turning radius.

W is the width of the vehicle.

$$\delta = \tan^{-1} \left( \frac{L}{\left(R + \frac{W}{2}\right)} \right)$$

Nodes:

`ackermann_drive_controller`: Controlling the system.

`ackermann_drive_publisher`: Publishes the current state of the Ackermann drive system.

Topic:

`/steering_commands`: This topic is used to send steering and speed commands to the Ackermann drive controller node. Where speed and `steering_angle` are actual msgs sent to the vehicle.

## 3.6 Verification and Validation

Since autonomous vehicles are safety-critical systems, testing is one of the most crucial parts of developing such a system. However, real-time testing is costly and inefficient, with several risk factors. Simulation tests are becoming increasingly popular for testing various aspects of autonomous systems in a virtual environment that mimics real-world scenarios, allowing developers to test their systems safely, controlled, and cost-effectively (Ramakrishna et al., 2022). Therefore, our team's approach was to develop test cases to test different algorithms and evaluate the performance and limitations in the virtual environment of Carla and Roadrunner.

### 3.6.1 Roadrunner

Roadrunner is a powerful 3D simulation tool that is commonly used to design and analyse road networks, transportation systems, and city planning. It can simulate large-scale transportation networks that include multiple intersections, traffic lights, and other complex features (Jeong et al., 2019). The project required us to drive the vehicle autonomously on a test track equipped with sensors at Cranfield University. A map of the test track was created by overlaying the point cloud data of the track. We extracted the necessary features and created a virtual representation of the test track. Using the scene editor in Roadrunner, we created a similar road with minimalistic error, with multiple intersections, traffic lights, and other complex features, as shown in Figure 16. Roadrunner provides a set of libraries for different types of roads, crosswalks, materials, and props. We added poles on the road to mount the sensors, as shown in Figure 16. Using the lane editing tool, we set the road width to about 7m and each lane width to 3.5m.

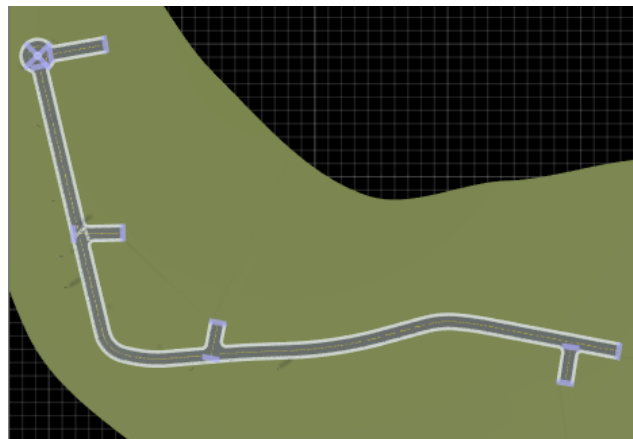


Figure 16 Road Runner Map

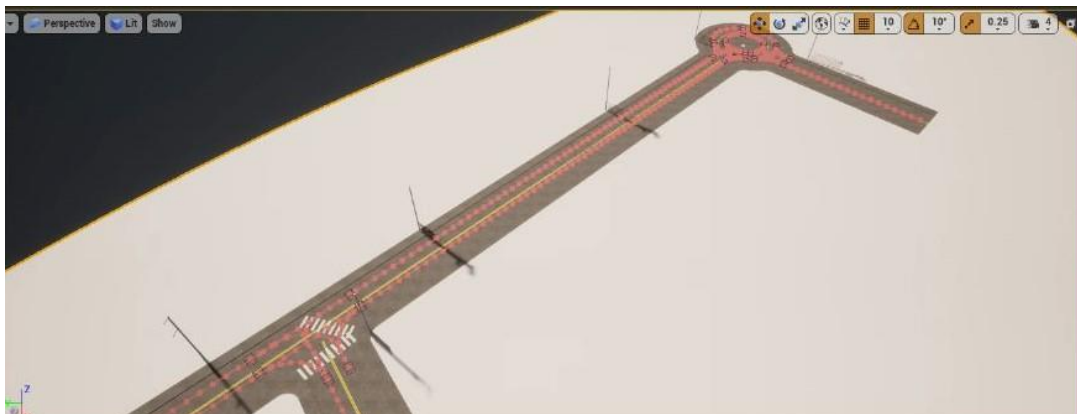


Figure 17 Road Runner Map

However, we faced a challenge in matching the coordinates of the point cloud map and the Roadrunner map. In the point cloud map, the origin (0,0) was at the 19th pole, whereas in our map, the 19th pole we had different coordinates. We overcame this issue by converting the coordinates.

Roadrunner is highly flexible and can be integrated with several other simulation tools. To use the Roadrunner map in CARLA simulation, we exported the map in fbx format, which is supported by CARLA. This provided us with a high degree of freedom to simulate in CARLA. To import the map in CARLA, we required two files: the fbx file containing the road architecture and the xodr file, which contains all the points and mesh of the map. We also noted that every surface, except the road surface, was automatically coloured green by Roadrunner, representing the surrounding land and dividers between the roads.

Overall, the Roadrunner tool proved to be instrumental in designing and analysing the road network for our project. Its features, such as the lane editing tool and the set of libraries for roads and materials, enabled us to create a realistic simulation of the track for testing our vehicle.

### 3.6.2 CARLA

CARLA is an open-source simulator created to evaluate and develop autonomous driving systems. The simulator provides a virtual environment for testing autonomous driving algorithms, perception systems, and control systems, eliminating the need for physical road testing. Furthermore, CARLA offers a variety of built-in features such as traffic simulation, pedestrian simulation, weather simulation, and dynamic objects, making it possible to simulate complex scenarios and evaluate the performance of our system in different environments and conditions (Dokur & Katkooi, 2022).

CARLA provides us with a high-fidelity simulation of the physical world, including realistic vehicle physics, sensor models, and environmental factors, allowing us to evaluate the effectiveness and reliability of our system in a safe and controlled environment (Mendhe et al., 2022). In our project, we used CARLA to test and validate our perception (LIDAR data) and navigation system (path planning and tracking algorithms). The simulation was carried out in 2 different maps, one with the imported road network that we designed in Roadrunner as shown in Figure 16-17, and the other with the pre-built CARLA town map as shown in Figure 18. Thus, we created a realistic virtual environment for performing simulation in varied environment and made sure that the algorithm works effectively independent of the surrounding.



Figure 18 Test map 2 for simulation in Carla

Different test cases are developed to test the vehicle performance which will be discussed in section 4. The vehicle model used is Tesla Model 3 and LIDAR is mounted at a remote location similar to that of the test track environment and the point cloud data is processed for localisation and object detection algorithms. The LIDAR attributes used in Carla is as shown in Figure 3.6.3. Additionally, we incorporated pedestrian control into our simulations to test the vehicle's ability to detect and respond to pedestrians in its path as shown in Figure 18. We utilized CARLA's PedestrianCrossing and WalkerSpawner classes to generate virtual pedestrians and evaluate the accuracy of the vehicle's pedestrian detection and avoidance capabilities (Actors - CARLA Simulator, n.d.). The path planning algorithm is tested using the 2D map generated using gmapping algorithm which is then visualised in rviz by publishing the goal location. An optimised path was generated using move\_base package of ROS navigation toolbox. As to the path following algorithm, two tests were carried out (a) using pre-loaded waypoints of the path in csv file, (b) using the path generated from move\_base package. The pure-pursuit controller accessed these points and published steering angle and speed commands for the vehicle to track the waypoints. Thus, the testing was carried out in the simulated environment of Carla allowing us to evaluate the performance of our autonomous system and gain insights into its strengths and weaknesses, allowing us to improve and refine our system before deploying it on the actual track.



Figure 19 Pedestrian detection in Carla.

## 4. Results and Discussion

### 4.1 Localisation

#### 4.1.1 Point Cloud Registration

This project employed the `pcl::IterativeClosestPoint` (ICP) method for point cloud registration. The algorithm's performance was measured using the `getFitnessScore` method, which provides the mean squared distance from each point in the source to its nearest point in the target. The maximum correspondence distance, which determines the radius of the distance from each point in the source point cloud, was also utilised for performance tuning. A higher radius value increased computation time as the neighbour search would be performed over a larger radius.

The point cloud for each LiDAR sensor was translated using its approximate position, and the ICP algorithm was applied to determine the affine spatial transformation matrix. The same procedure was followed using the NDT algorithm, which yielded a slightly better performance result of 2.29.

When the ICP algorithm was applied without translation, it produced an error of approximately 6.1, and the point cloud was not located correctly. However, after the translation step, the error was reduced to 2.43. Although some resources suggest that the NDT algorithm is more efficient, this project's vanilla implementation of ICP ran faster. Thus, it was used to determine the relative coordinate frames.

Table 1 Fitness Scores of Point Registration Algorithms

	ICP fitness score	NDT fitness score
Initially translated point cloud	2.43	2.29
No translation of point cloud	6.1	6.32

### 4.1.2 Clustering

Through calibration of the parameters, the clustering of objects of interest was achieved. Specifically, the voxel size and leaf size parameters in the Voxel Filter algorithm are adjusted. The leaf size parameter determines the size of the voxel grid that is employed for point cloud downsampling. Changing the leaf size directly impacts the resolution of the voxel grid. Thus, affects the level of detail in the resulting point cloud.

Increasing the leaf size reduces the resolution of the voxel grid, reducing the level of detail in the point cloud. This can result in losing important information, especially for small or intricate features such as pedestrians. However, this also leads to faster processing times due to reduced memory requirements. Conversely, decreasing the leaf size increases the resolution of the voxel grid and results in a more detailed point cloud. However, this also increases processing time and memory requirements, which can be problematic for larger point clouds.

The Euclidean Cluster Extraction algorithm employs a cluster tolerance parameter, which dictates the maximum distance between adjacent points grouped into clusters. The value of the cluster tolerance significantly impacts the accuracy and segmentation quality of the resulting point cloud. If the cluster tolerance is increased excessively, the algorithm will group points that belong to separate objects or surfaces into larger clusters, which can cause over-segmentation and result in a loss of detail and accuracy. Furthermore, a high cluster tolerance can also cause small objects or features to be missed entirely. Conversely, decreasing the cluster tolerance too much can form numerous small clusters, leading to over-segmentation and loss of detail. Furthermore, a low cluster tolerance can increase computational costs and slow down the segmentation process. Thus, the optimal value for this parameter is carefully determined.



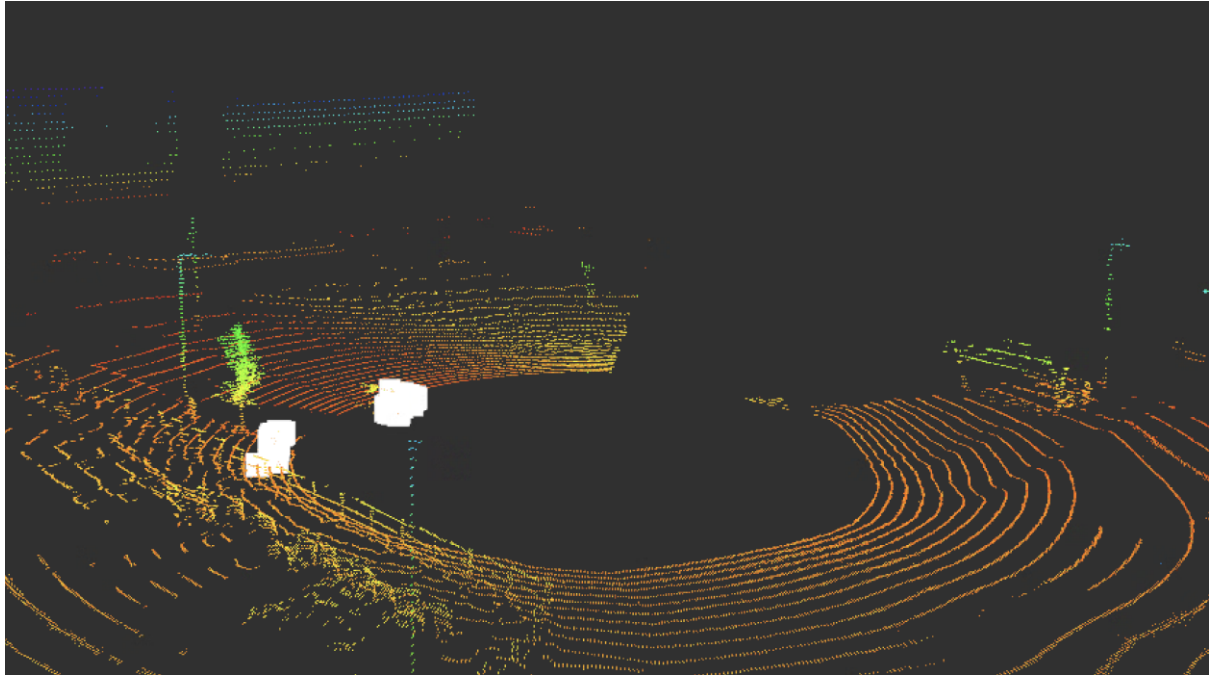


Figure 20 Output of Clustering

### 4.1.3 Bounding Box

The functionality of this block performed as anticipated since it does not involve complex operations. As illustrated in the images below, we successfully drew and placed the bounding boxes on the clusters of the objects. However, the only possible issue that may arise pertains to the clusters themselves, which could be problematic.

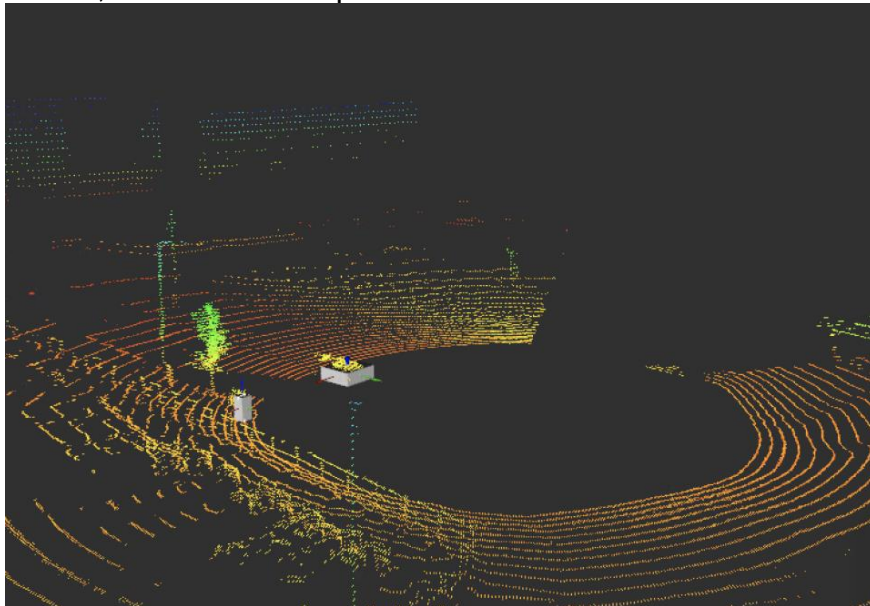


Figure 21 Bounding Box for the Vehicle and the Pedestrian

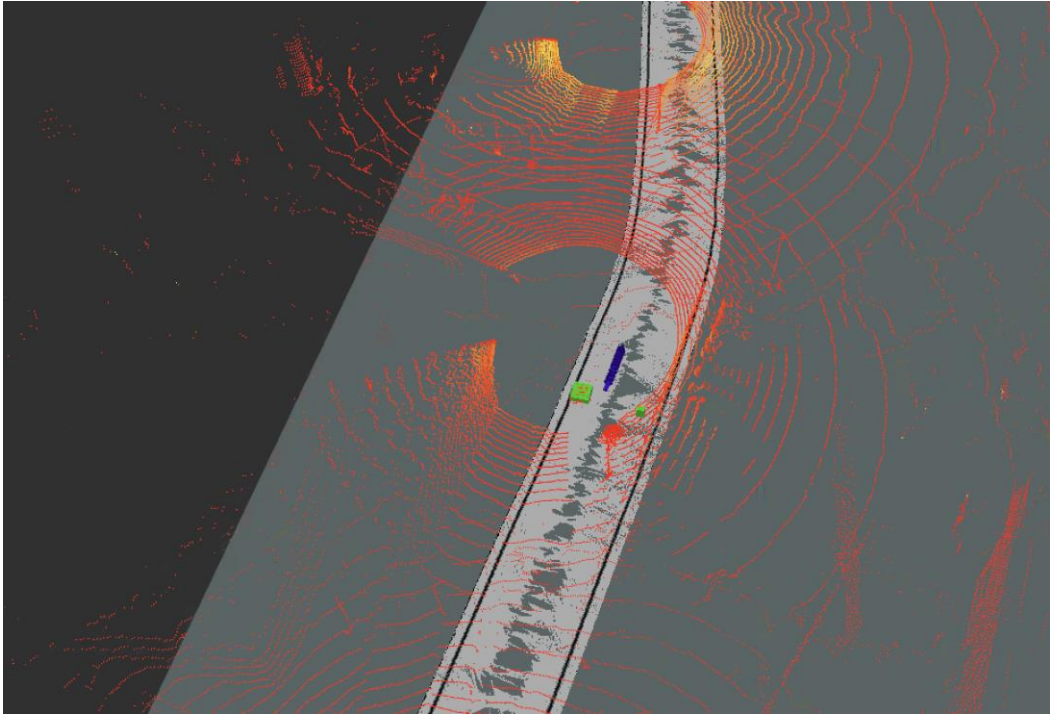


Figure 22 Bounding boxes for the vehicle and pedestrian

#### 4.1.4 Ego Vehicle Detection

As seen in the Figure below, the software is able to detect the ego vehicle.

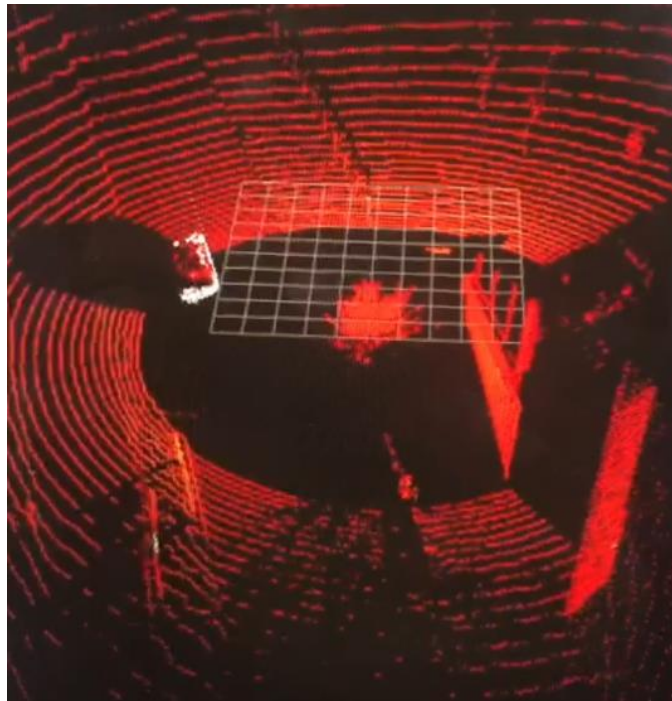


Figure 23 Vehicle Detection

### 4.1.5 Matching the Mesh of Objects

First, we needed to find a Land Rover Discovery mesh file and convert it to a PCD file. The mesh had, at first, too many points (nearly 60,000 points). Therefore, we downsampled it by applying a voxel filter with a leaf size of 0.2 which is the size advised by the PCL library. It gave us a final PCD file with 384 points.

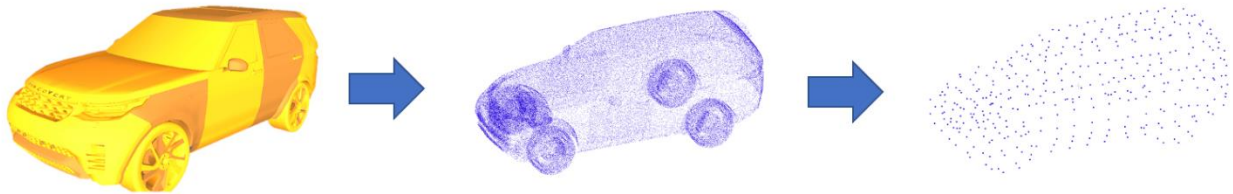


Figure 24 Generation of an appropriate point cloud of the Land Rover Discovery

Then, we could run the NDT algorithm with a ROS bag file acquired during a hands-on session. The algorithm reads a point cloud that had been filtered and clustered and the point cloud of the mesh file that is shown above. The algorithm roughly tracks the cluster but lacks accuracy.

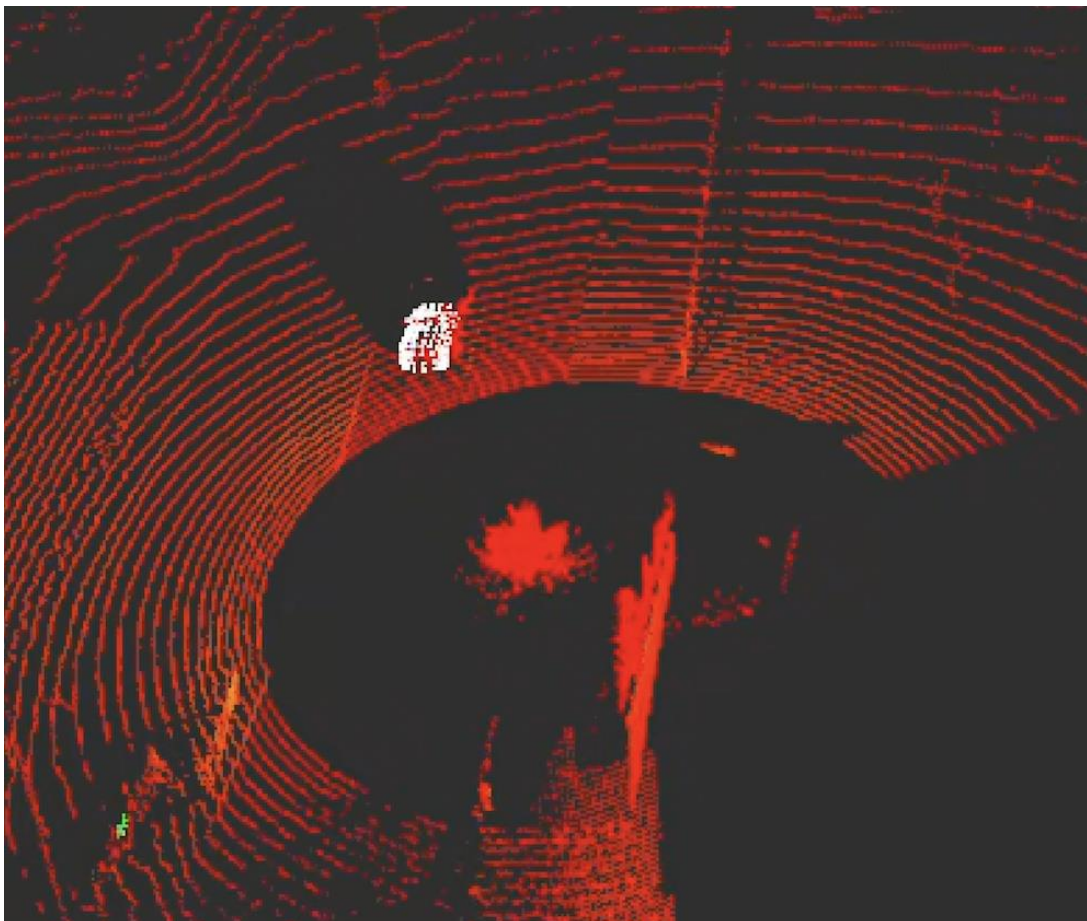


Figure 25 ROS visualisation of the output of the NDT algorithm



## 4.2 Object Detection

To validate the system's performance, we designed test cases and subjected the software package to them. The tests involved ROS bags, and the system was deemed functional as it passed both tests, generating the appropriate warning and non-warning messages as required.

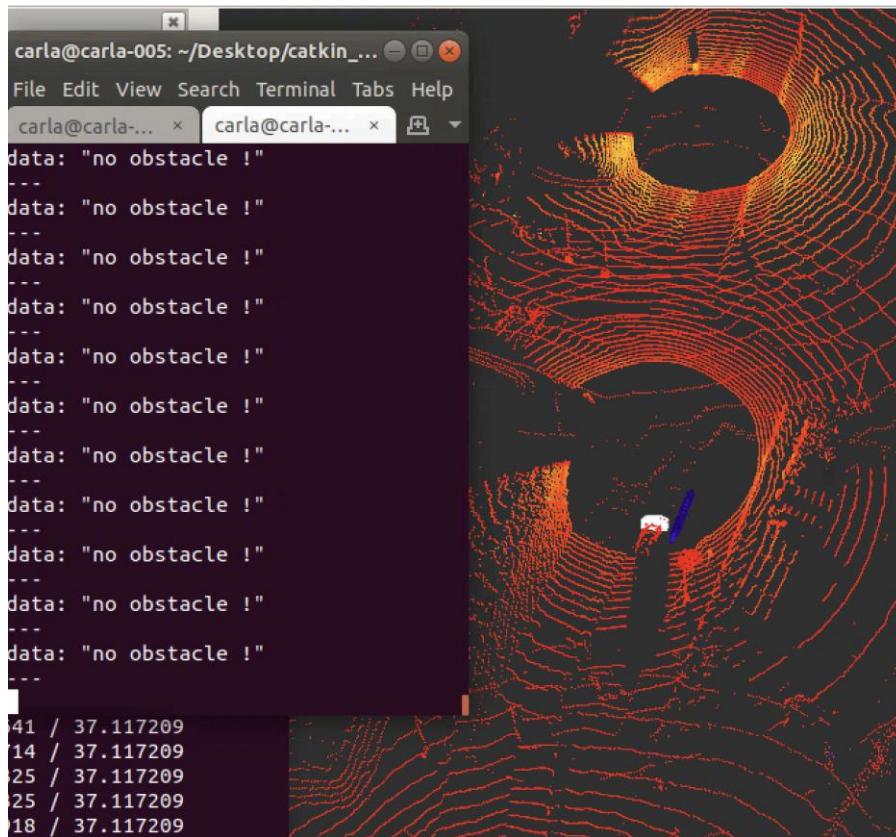


Figure 26 No Obstacle Case

In this particular test case, a pedestrian is walking towards the ego vehicle, and the expected outcome is for the system to display 'obstacle detected'. As the pedestrian walks in close proximity to the ego vehicle, the LiDAR sensor detects that the distance between the pedestrian and the ego vehicle is less than the specified threshold distance, and subsequently publishes a command indicating that an obstacle has been detected to the driver.

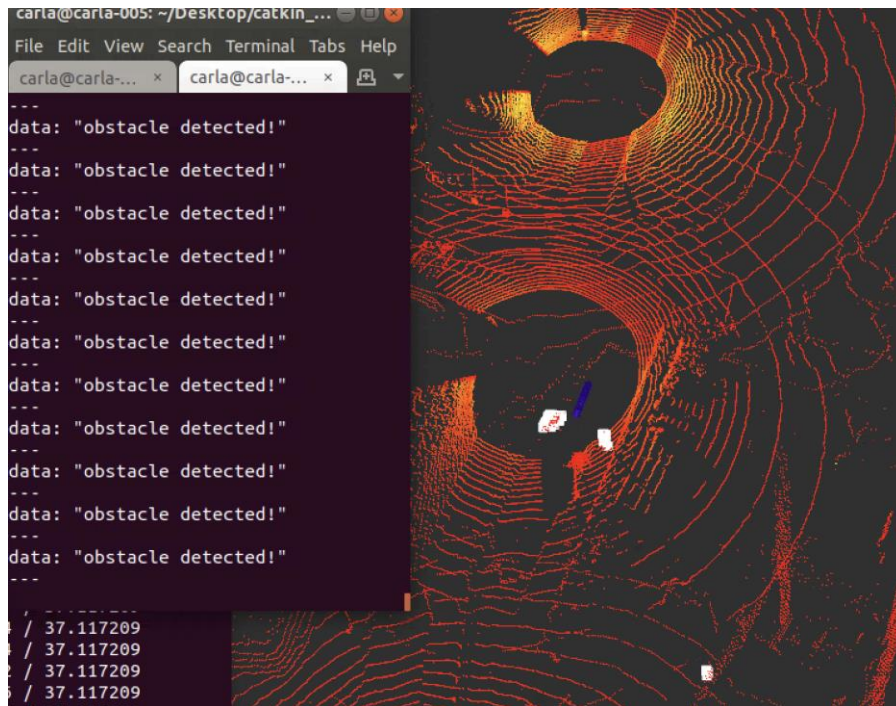


Figure 27 Obstacle Detected Case

In this test case, a pedestrian was walking towards the ego vehicle, and the system was expected to display the message "obstacle detected." As the pedestrian moved closer to the ego vehicle, the LiDAR sensor detected that the distance between the pedestrian and the vehicle was less than the threshold distance, prompting the system to publish the "obstacle detected" command to the driver. The system successfully passed this test.

For the object classification, We endeavoured to evaluate multiple algorithms and pre-trained models, including PointPillar, CenterPoint-Pillar, VoxelNet, and MPPNet, for point cloud classification tasks trained with ModelNet40. However, we encountered issues with certain algorithms due to dependencies or limitations in processing data in real time.

## 4.3 Path Planning

The path planning algorithm is validated by performing several tests on Carla simulator for different goal location and map areas. The algorithm is able to generate shortest path from current vehicle location to the goal location on the 2D map generated using gmapping. The visualisation in rviz is as shown in Figure 28.

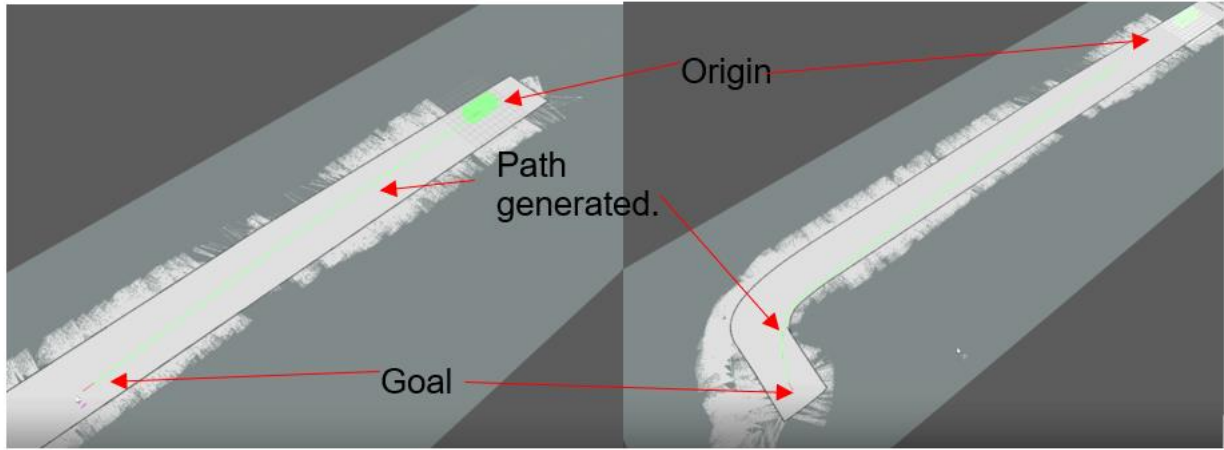


Figure 28 Path generation using move\_base package in Carla.

Similarly, the cost map generated is as shown below in Figure 29. The costmap parameters such as static layer, inflation layer and obstacle layer can be modified as required to optimise the generated path.

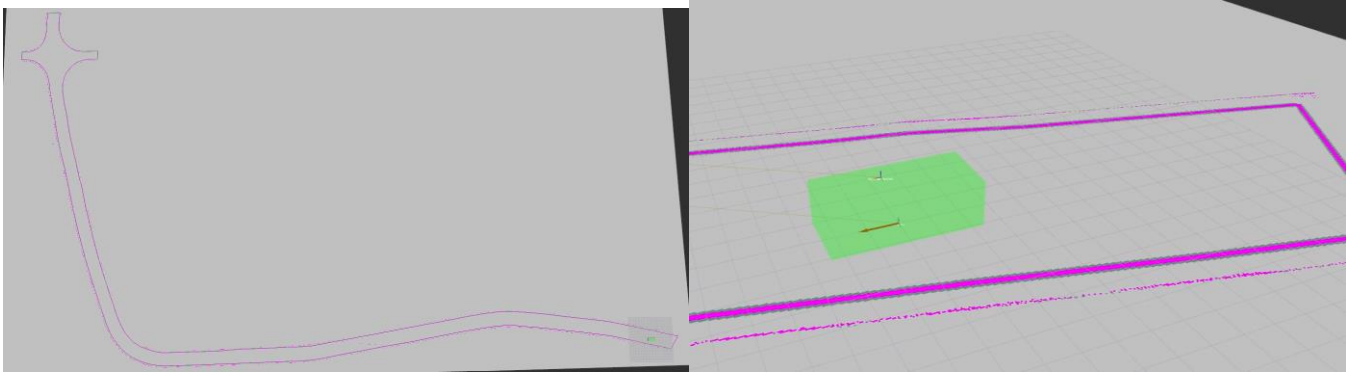


Figure 29 Cost map generation for the boundary of the road.

It is also worth noting that this algorithm is capable of taking into account of the kinematic model of the vehicle but is not capable of maintaining the lane and generates the shortest path. Hence it is worth exploring the Frenet path planning algorithm (Kusuma et al., 2023). During the test drive of the vehicle, different steering commands were published and the odometry data was recorded for further analysis. This data was further analysed in Matlab to understand the deviation as shown in Figure 30 and the approximate distance between these points were 3m.

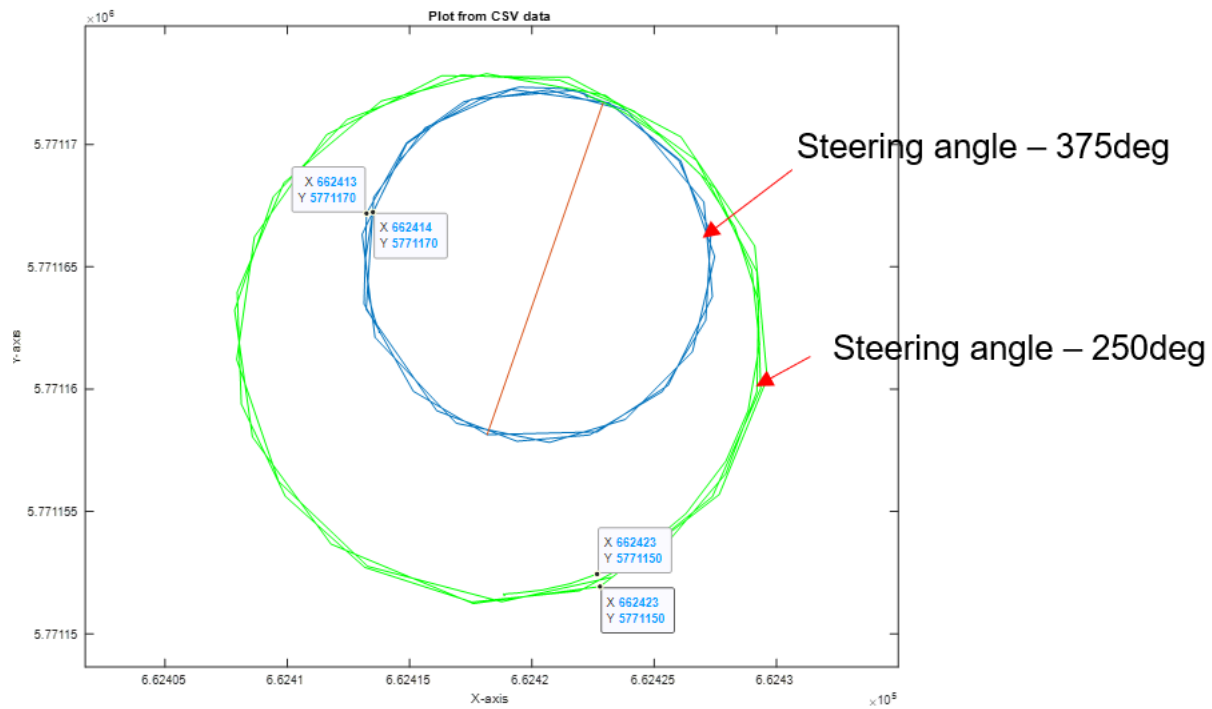


Figure 30 Steering Angle Results

Table 2 Location errors

UTM coordinates	Latitude	Long	Error
662413.5007, 5771167.235	52.0672475819147	-0.630482988302179	~3.11m
662414.8836, 5771170.031	52.0672722941561	- 0.630461500840562	

The path tracking algorithm was dependent on odometry data and if there is deviation in the odometry data then the tracking error will increase.

## 4.4 Path Following

Several tests have been done on Carla's simulator to validate and verify the path following part. A trajectory has been generated manually with the different waypoints that form it. This trajectory has been made so that it makes a right turn.

The vehicle has been spawned at a point other than the first waypoint of the path. In this way, it will be possible to check how the vehicle approaches the trajectory.

As the behaviour of the Pure Pursuit controller depends on the lookahead distance, the test has been performed with 4 different distances: 0.5 m, 1 m, 3 m, and 5 m.

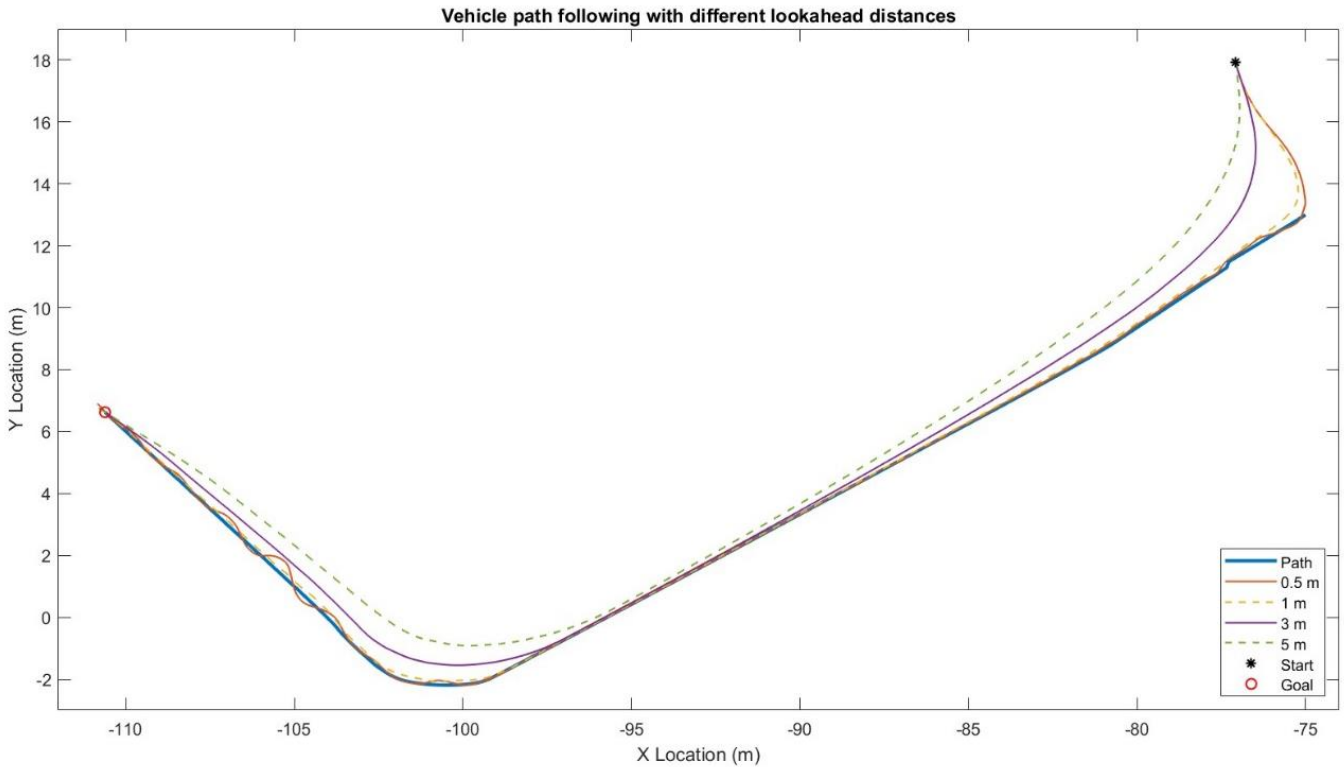


Figure 31 Path Tracking Test

The above Figure shows how the vehicle has followed the path depending on the lookahead distance. Starting from the beginning of the path, you can see how the lookahead distance affects the speed at which the vehicle reaches it. When this distance is small, the vehicle tries to reach the road as quickly as possible, so it turns sharply and heads straight for it. This would be the case at 0.5 m and 1 m. However, when this distance is larger, such as the 3 m and 5 m cases, the vehicle steers more gently towards the road, reaching it slowly.

We can see clear differences if we continue to look at the turning part. On the one hand, for small distances, the vehicle performs a trajectory closer to the real one but with more oscillations, as shown in the 0.5 m case. On the other hand, as the lookahead distance increases, the vehicle's trajectory becomes smoother but, at the same time, farther away from the real one. It can be seen how the 3 m and 5 m cases start to turn early and move away from the real path.

Finally, looking at the target point, all cases reach the goal and stop at less than 0.5 m from that position, as expected.



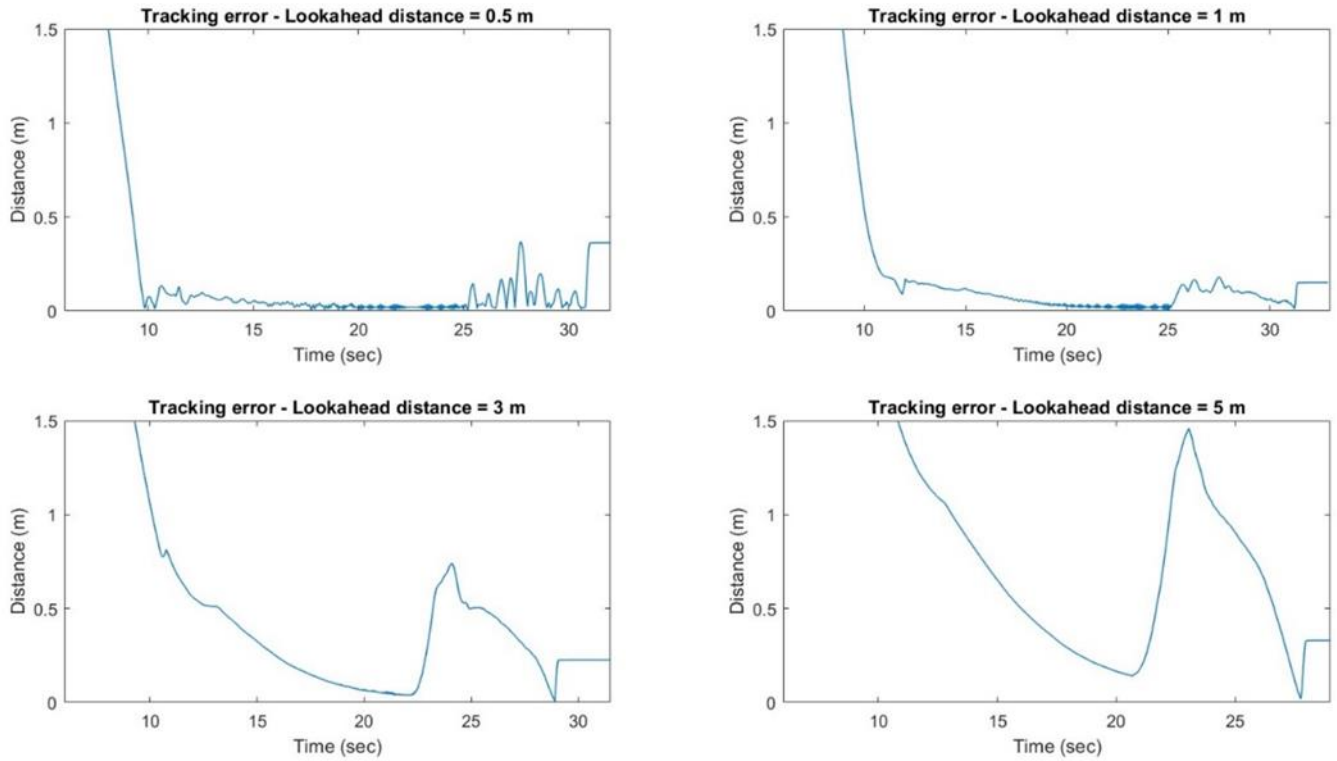


Figure 32 Tracking Error

Returning to the trajectory realised by the vehicle, the above Figure shows the error between this trajectory and the real path. In the beginning, the time it takes for the vehicle to reach the path is shown to be faster for a smaller lookahead distance. Finally, in the middle, there is a peak, which represents the turn. This peak becomes larger but smoother as the lookahead distance grows and starts to grow earlier, as expected.

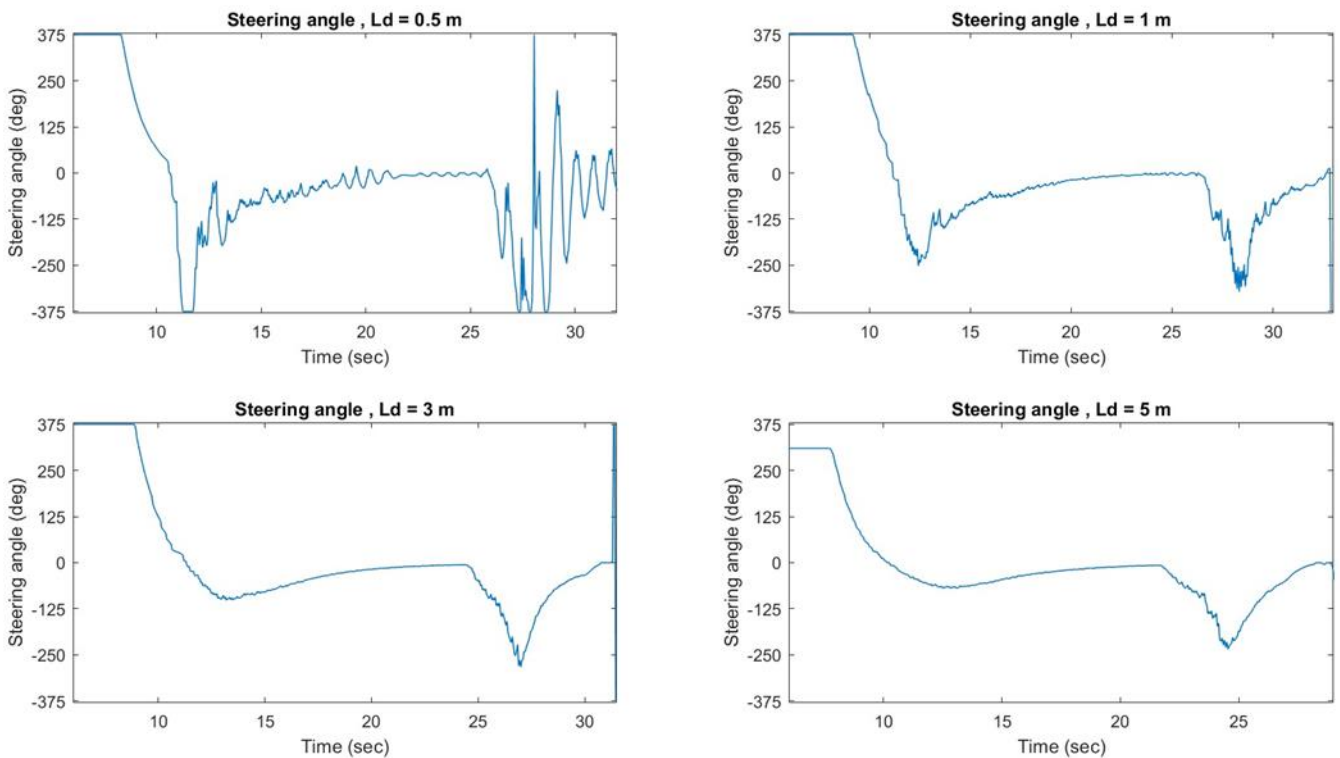


Figure 33 Ackermann Command – Steering angle

The results of the Ackermann Commands sent to the vehicle in each of the four cases will be discussed below.

As predicted, the value of this command changes abruptly when the lookahead distance is smaller and changes smoothly when it is larger. This is because when the distance is smaller, the vehicle constantly tries to correct its position to be on the real path. One aspect to consider in real life is the speed at which the vehicle can turn the steering wheel. This command can be sent to the vehicle, but future work is required.

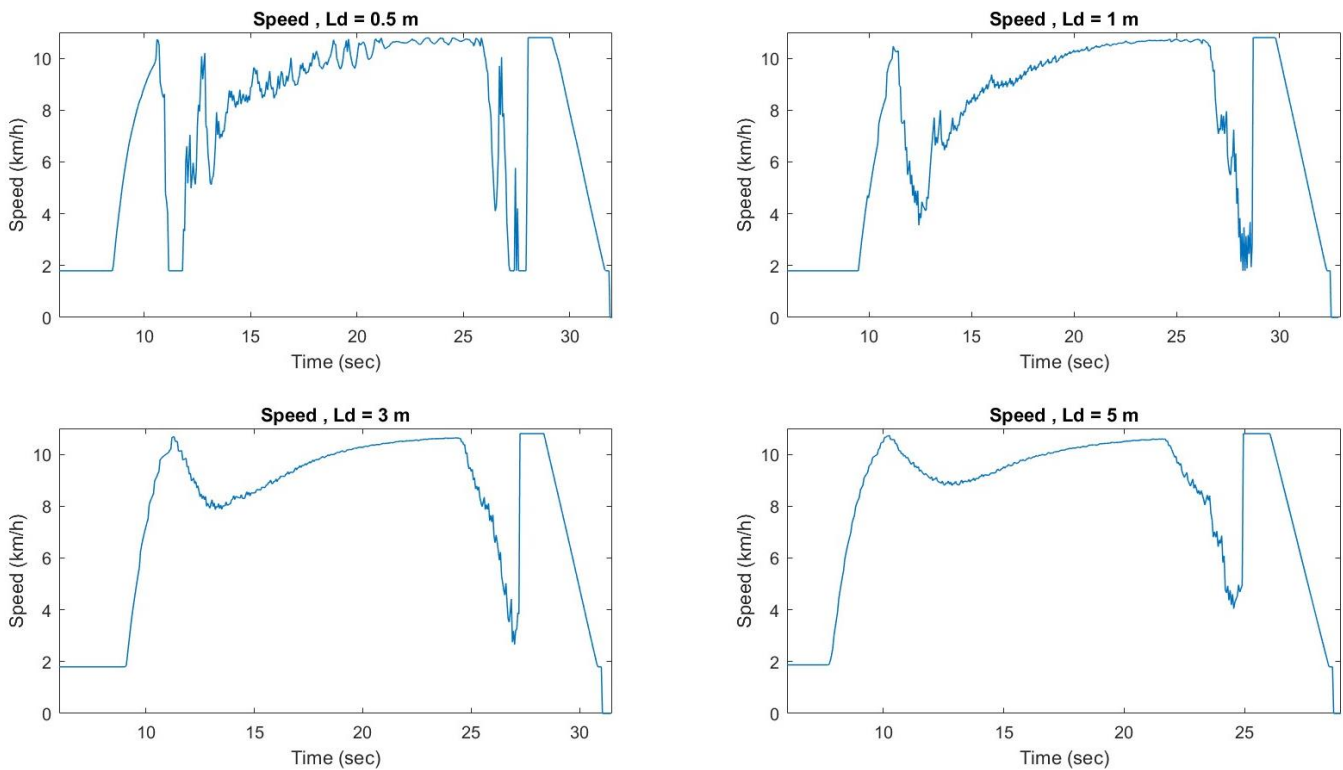


Figure 34 Ackermann Command – Speed

On the other hand, Because the controller is programmed so that the speed is inversely proportional to the steering angle, the speed will maintain more constant values when the lookahead distance is greater since the vehicle makes smoother turns. However, for shorter distances, as the vehicle is constantly correcting direction, the speed command will fluctuate continuously.

## 4.5 Verification and Validation

A comprehensive test case was developed as per the individual objective of autonomous navigation. The entire project requirement was divided into 3 main sections,

1. Localisation and obstacle detection using LIDAR.
2. Path planning for the robot to navigate in the environment.
3. Path tracking to follow the generated waypoints.

The unit-level test cases defined aim at identifying that each requirement is fulfilled with acceptable deviation. Due to the time constraint, exact acceptable performance metrics is not defined, but the simulation results show promising results.

### a) Lidar point cloud data processing for object detection

We were able to successfully identify the vehicle and pedestrian in both Carla and real environment.

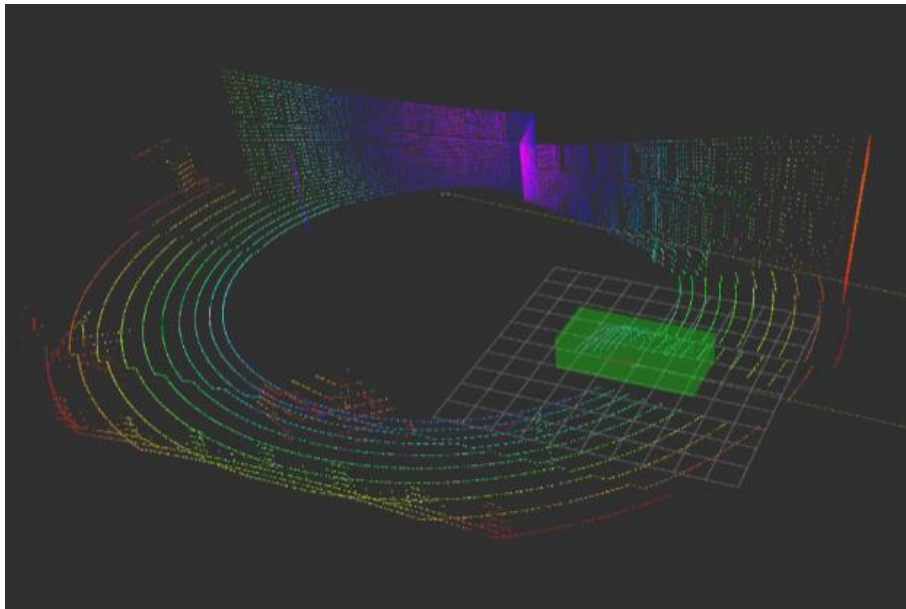


Figure 35 Object detection using point cloud data in Carla.

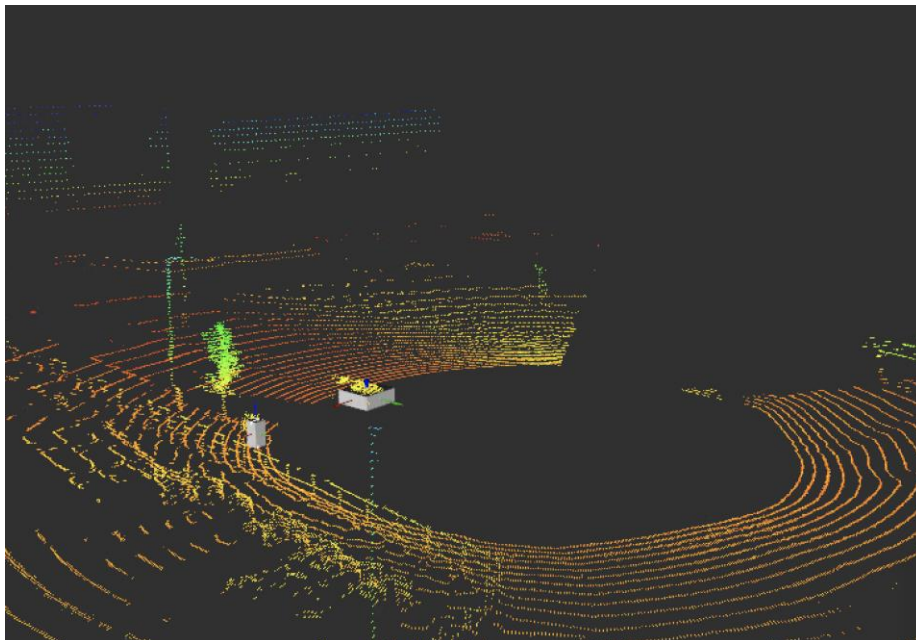


Figure 36 Object detection using point cloud data in Real environment.

#### **b) Path planning to reach from point A to point B**

We were able to generate the path without obstacle and the same was visualised in rviz as shown in below.

#### **c) Path following on different curvatures of road.**

The path following algorithm was successful in tracking the generated path and several performance metrics were studied both in simulation and real environment. Figure 4.6.3 and Figure 4.6.4 shows the working of path tracking algorithm in Carla and real environment.



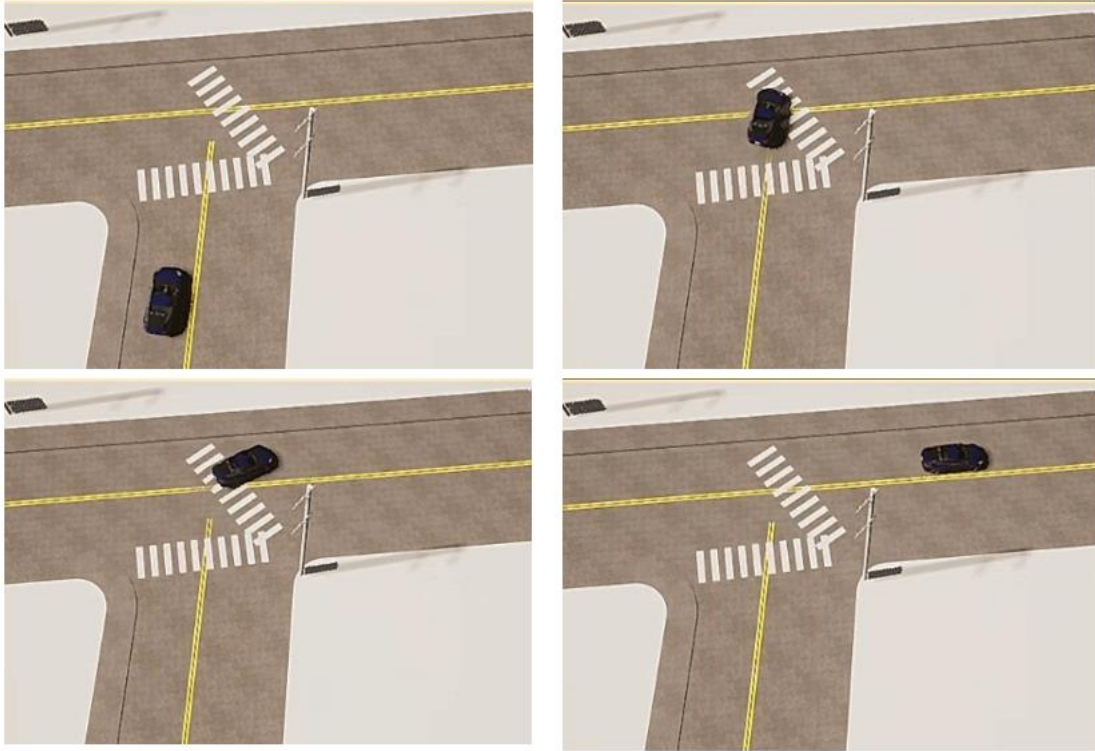


Figure 37 Working of Path following algorithm in Carl environment.

The path tracking algorithm was tested on the recorded rosbag data and the results are shown in Figure below.

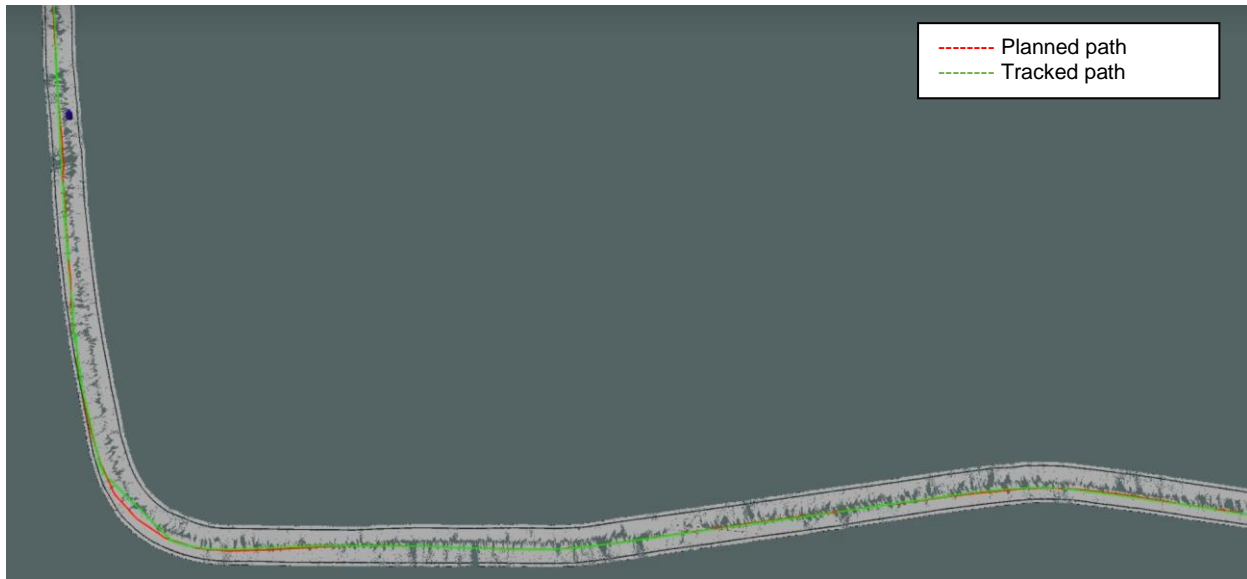


Figure 38 Working of Path following algorithm in Real environment.

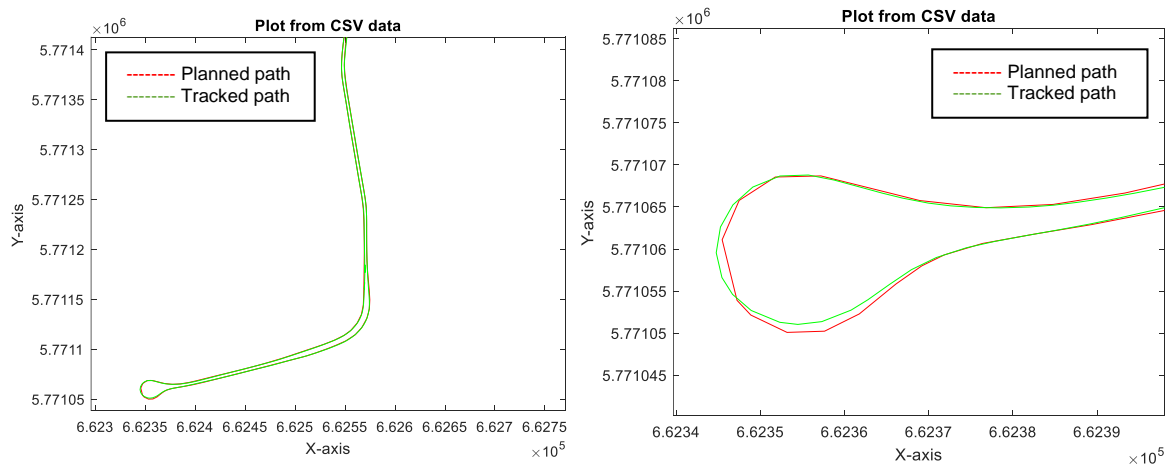


Figure 39 Simulation results of pure pursuit controller in Carla

# 5. Market Opportunity and Business Model

## 5.1 Industry Overview

The autonomous vehicle industry is reaching new heights and is increasingly relying on strong software solutions. Software companies are now focusing on creating advanced solutions for OEMs to enhance compatibility and provide consumers with a safe autonomous driving experience. In this regard, using in-vehicle sensors to operate the vehicle is constantly refined to ensure a foolproof system. It is important to note that all sensors used in self-driving car applications are run on back-end software that collects raw data from different sensors and processes it as per their intended use (Gwak et al., 2019).

### 5.1.1 Market Drivers

The market for automotive software and electrical and electronic components (E/E) is expected to grow at an annual growth rate of 7%, reaching GBP 376 billion by 2030, up from GBP 191 billion in 2020. This growth rate is significantly higher than the overall automotive market, estimated to grow at a compound rate of 3 per cent during the same time frame. As a result, software and electronics have become the focus of most automotive companies. Customer adoption will largely depend on the cost of vehicles, as Autonomous driving features are expected to remain expensive. Level 3 systems could cost up to GBP 4,000 or more, which would significantly increase the effective price for the customers, and the reason for the increment would be due to the overall cost incurred to the OEMs for the development and research of the technology (Burkacky et al., 2023; Leminen et al., 2022).

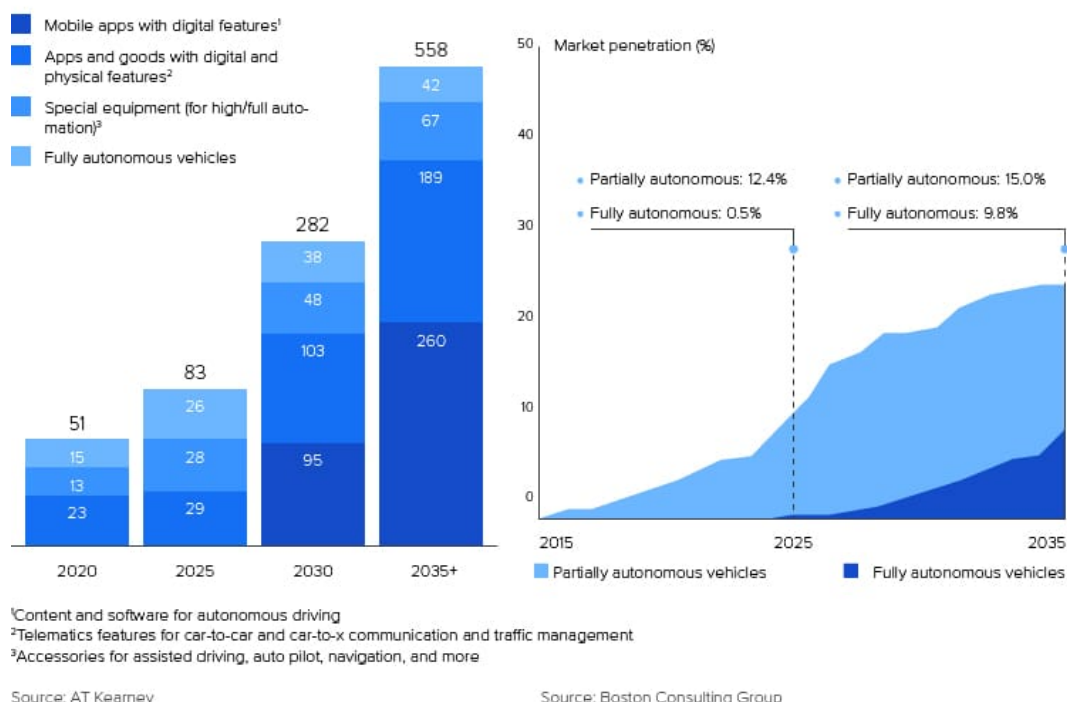


Figure 40 Left – Global market for Software and other services for autonomous vehicles, right – Sales penetration of autonomous vehicles (Toptal, AT Kearney, Boston Consulting Group)

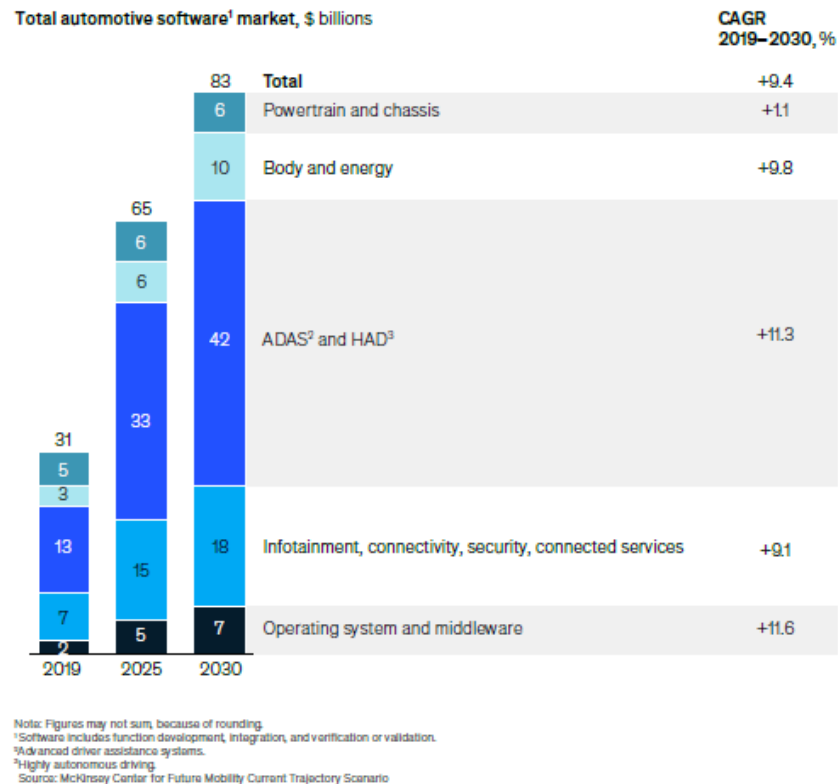


Figure 41 Expected growth of the Automotive software market (McKinsey, 2019)

The market for automotive software is expected to grow by over 9% annually, reaching \$80 billion by 2030. Advanced driver assistance and autonomous driving software will account for much of this growth and makeup almost half the software market by 2030. Software development for higher-level autonomous driving will happen before it becomes available in the market (Burkacky et al., 2023).

### 5.1.2 Market Restraints

The pattern of motivating investments in Autonomous vehicles has decelerated since 2019 due to the slower-than-expected rollout of AVs for public consumption, many industries, from ridesharing to auto and IT, have already been primed for the emergence of AV technology. New start-ups in ridesharing, technology development, AI and navigation innovations are disrupting the overall market scenario, which would asymmetrically change the arrangements between existing automakers and auto-parts suppliers. As a result, the question of who will define and control the AV market from its initial stages remains crucial (Alvarez León & Aoyama, 2022).

## 5.2 Product Description

Our I2V communication software is an innovative solution that enables real-time communication between street sensors and vehicles to navigate without internal sensors. Our cutting-edge technology will use advanced machine-learning algorithms to process data from street sensors and provide vehicles with accurate and reliable navigation information.

With I2V communication software, businesses and individuals can save money by avoiding the need to install expensive and complex sensors inside their vehicles ([Estimated calculations mentioned in Financial Analysis](#)). Our highly scalable and customisable software makes it ideal for the road transport industry. It can also integrate with other software and systems, making it

an ideal solution for smart cities and other complex infrastructures. Our user-friendly interface provides real-time updates on traffic conditions, road closures, and other relevant information to ensure vehicles reach their destination safely and efficiently. Our system is easy to install and operate, and we provide comprehensive support and training to our clients.

According to analysts, the market for software in autonomous vehicles is expected to increase by 28.2% yearly, adding more than 5 billion USD. The increased expectations of technological advancements in the vehicle show the requirements for innovation in the hardware and software integrated with the vehicle. Being an early adopter of the I2V techniques would allow exploring the potential of increasing the safety and comfort of the end user.

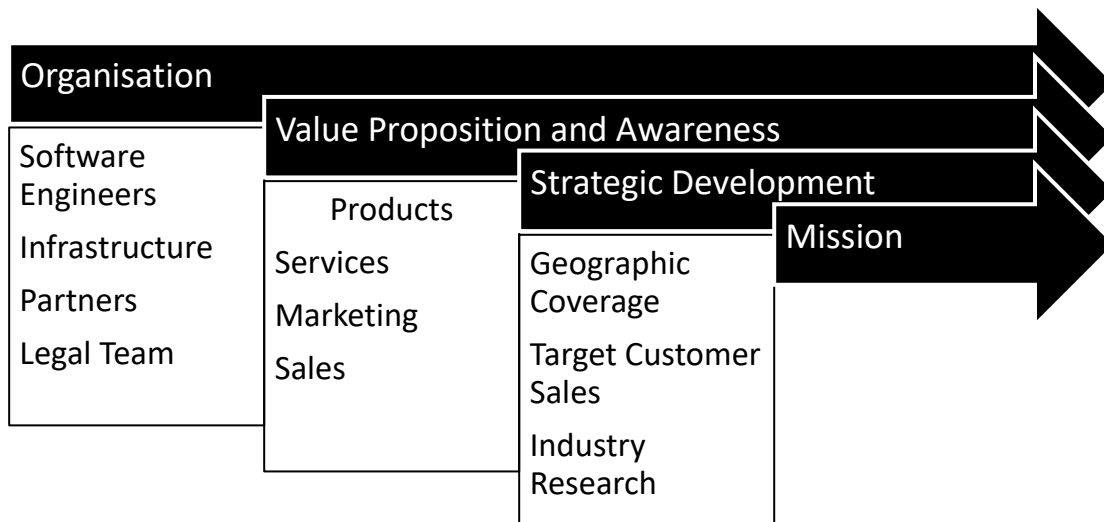


Figure 42 Hierarchy for the achievement of the end goal



Table 3 Business Model Canvas

BUSINESS MODEL CANVAS				
Key Partners	Key Activities	Value Propositions	Client Relationships	Customer Segments
B2B Autonomous Vehicles manufacturers, Tier 2 suppliers, Start-ups and Technology providers	Development of the Software solutions for Integrating I2V compatibility	Increasing safety and reducing the overall cost of ownership of vehicles and traffic congestion by communicating with the Infrastructure	Feedback, Collaboration, Regular development	Fleet Companies and Private users
	Key Resources		Channels	
	Ships, Infrastructure, Workforce, IT and Patents		Software Sales and After Sales Customer support	
Cost Structure		Revenue Streams		
Salaries to employees, Maintenance, IT and Research		Sales of software and yearly service charges for the after sales services provided		

### Prerequisites for I2V Software Implementation

1. 5G compatible or consistent high-speed internet connectivity
2. LiDAR sensors with mid-range scanning and high scan rate capabilities for real-time data processing (Having horizontal view road sensors on the curves and vertical on the long-stretched roads are preferable for having a better V&H FOV (Wang et al., 2022)). Considering our highly scalable software capabilities, we can develop specific requirements based on shared details.
3. Adequate power supply and backup systems to ensure uninterrupted communication between the infrastructure and the vehicles.
4. Availability of relevant data from street sensors, such as traffic conditions, road closures, and weather information
5. Compliance with relevant regulations and standards for autonomous and connected vehicles.




	Model	Field of View (horizontal x vertical)	Horizontal angular resolution	Vertical angular resolution	Minimum range (metres)	Maximum range (metres)	Scan rate
	Ouster OS1-64 LiDAR Sensor	360° x 45°	NA	2.8°	0.3m	120m	20Hz
	Blickfeld Cube 1 Outdoor - wide FoV 3D LiDAR	70° x 30°	1.0°	NA	NA	250m	50Hz
	Blickfeld Cube 1 - wide field-of-view 3D LiDAR	70° x 30°	1.0°	NA	NA	250m	50Hz

Figure 43 Recommended LiDAR sensor comparison ([Level5supplies](https://www.level5supplies.com/))

## 5.3 Focused Associate Partners and Collaboration

Initially, we would like to associate with businesses manufacturing autonomous vehicles for the fleet and develop software solutions for them to integrate their vehicle compatibility with I2V. The fleet vehicles usually run on the same route and mainly on the Motorways, also called Strategic Road Network (SRN) in the UK. The Strategic Road Network (SRN) is a vital component of the UK's national transport system, comprising 4,500 miles of motorways and major "A" roads that carry a significant proportion of the country's traffic and freight, enabling trade and business growth. Starting with the motorways would reduce the risk and help optimise the software for urban road readiness.

The Highways England and the government aim to make the roads connected vehicle ready. After successful trials and optimisation of our software technology with the fleet owners on the Motorways then, we would associate with private vehicle manufacturers where the vehicles would be mainly running on all-purpose and Toll Roads. The current Motorways under SRN have been installed with 4000 CCTV cameras and 8750 traffic monitoring signs. The investments for developing the connected vehicle ecosystem could be expected towards preparing the Strategic Road Networks to have installed these sensors. The same could be proposed to the government authorities.



Figure 44 Highway Strategic Road Network UK  
Blue – Motorways, Red – All Purpose and Dotted Blue – Toll Roads ([NationalHighways.co.uk](http://NationalHighways.co.uk))

**Collaboration** – Companies like [VortexIoT](#), [Cepton](#), [Intel](#), and [Vrio](#) are building the roads to be ready for the Intelligent Transport System (ITS), and strong associations could lead to mutual benefit making the system robust and ready for the users with collaborative efforts.

## 5.4 Financial Analysis

The cost of installing LiDAR sensors on roads can vary depending on various factors, such as the type and quality of the sensors, the length and width of the roads, the number of sensors required, and the complexity of the installation process. The cost of installation, labour, and maintenance must also be considered. It is not easy to estimate the cost accurately. However, generally, the cost of LiDAR sensors can range from a few hundred dollars to several thousand dollars per unit, depending on the specifications and features of the sensors.

Financial calculations are mentioned below as well as available to access [online](#),

### 5.4.1 Infrastructure Development

Table 4 CAPEX

<b>CAPEX for infrastructure development</b>	LiDAR Sensor Cost X 1 Qty <sup>1</sup>	£ 8,000
	Range of LiDAR Sensor (metres) <sup>2</sup>	120
	Total LiDAR's to be installed <sup>3</sup>	4166
	Hardware and IT Solutions <sup>4</sup>	£ 6,000,000
	Installation Cost <sup>4</sup>	£ 50,000
	<b>Total (A)</b>	<b>£ 39,378,000</b>

*\*Tentative cost mentioned against each header*

*1 - Ouster OS1 Sensor*

*2 - Working range of Ouster OS1 Sensor*

*3 - 120m range of LiDAR \* 500 kms coverage of SRN = 4166 sensors*

*4 – Estimated*

The total cost mentioned above is tentatively calculated. Various factors could lead to effective cost reduction, including tax concession, and bulk order benefits. The government authorities could contribute the cost from the overall budget allocated for the development of the highways and business conglomerates who could invest and charge an additional fee from the highway users for generating revenue.

### 5.4.2 Cost Analysis for Our Company

Table 5 Company Cost Analysis

<b>CAPEX for Business Development</b>	Technology and Infrastructure <sup>1</sup>	£ 200,000
	<b>Total (A)</b>	<b>£ 200,000</b>
<b>OPEX</b>	Employee Salary and Benefits (50) <sup>2</sup>	£ 2,500,000
	Travel and accommodation <sup>1</sup>	£ 20,000
	<b>Total (B)</b>	<b>£ 2,520,000</b>
<b>Revenue</b>	Incorporation Cost for one OEM/Business <sup>1</sup>	£ 60,000
	Associated OEMs/Businesses in first year <sup>1</sup>	20
	20 Associated OEMs/Businesses in first year <sup>1</sup>	£ 1,200,000
	Annual Maintenance Fees from 1 OEM/Business <sup>1</sup>	£ 25,000
	Annual Maintenance Fees from 20 OEM/Business	£ 500,000
	Additional Earnings <sup>1</sup>	£ 50,000
	<b>Total Earning (C)</b>	<b>£ 1,750,000</b>
<b>Cost Payoff</b>	Total Investment (A+B)	£ 2,720,000
	Total Earning (C)	£ 1,750,000
	Pay Off Time	1 Year and 5 Months

1 - Estimated

2 - 50 Employees and an average pay of GBP 50000 per employee

### 5.4.3 Cost Analysis for Our Customers

Our technology provides an innovative approach for our associate partners to stand out from the competition in their respective business niches, leading to a swift return on investment. For OEMs, our solution enables a reduction in vehicle sensor requirements, resulting in lower vehicle costs and decreased computation needs within the vehicle. This approach is expected to increase consumer acceptance and improve user safety. With high scalability, we are committed to providing excellent customer service to ensure high satisfaction and optimal results throughout their journey with us.

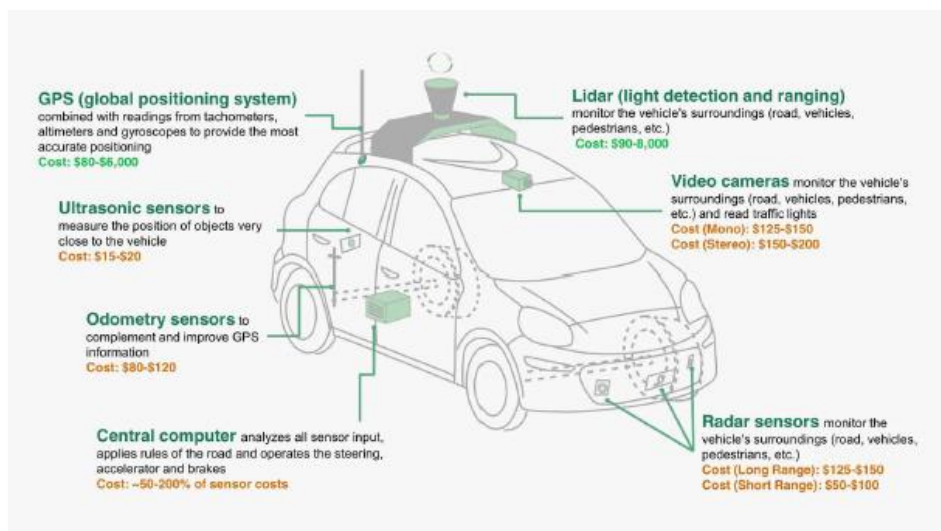


Figure 45 Self Driving cars component wise cost ([Wired](#))

Table 6 Cost comparison of I2V Integration Level

Estimated car cost (Mercedes EQS)	£ 105,610	I2V Integration		
Sensors	Current Cost	Basic Level @ 70% current cost of sensors	Moderate Level @ 40% current cost of sensors	Advanced Level @ 10% current cost of sensors
LiDAR	£ 12,000	£ 3,600	£ 4,800	£ 1,200
Camera (Stereo 16 Cameras)	£ 3,200	£ 2,240	£ 1,280	£ 320
Radar (12)	£ 2,000	£ 1,400	£ 800	£ 200
Central Computer - (200% cost of Sensors)	£ 34,400	£ 24,080	£ 13,760	£ 3,440
<b>Total</b>	<b>£ 51,600</b>	<b>£ 36,120</b>	<b>£ 20,640</b>	<b>£ 5,160</b>
<b>%ge of total car cost</b>	<b>49%</b>	<b>34%</b>	<b>20%</b>	<b>5%</b>
<b>Cost with I2V Integration</b>	<b>-</b>	<b>£ 90,130</b>	<b>£ 74,650</b>	<b>£ 59,170</b>

#### Considerations

1. Highest possible cost inclusion of the automation sensor as the vehicle has advanced level of autonomous capabilities
2. With higher level of integration less advanced sensors and computation could be used. That would reduce the overall cost based on the less requirement of hi-tech sensors.
3. 10% of sensor cost is still calculated in the Advanced Level considering the basic features operation in the vehicle (Vehicle Parking)

In this analysis, with a higher level of Autonomous vehicle integration, the overall cost of the vehicle would go as low as 50% of the total cost. This would assist the manufacturers in enhancing their sales of the vehicle and growing their market contribution.

## 5.5 PESTEL Analysis

### 1. Political

- Government regulations and policies related to adopting and using autonomous vehicles can impact the demand for I2V communication software.
- Changes in taxation and incentives related to road safety and environmental concerns can influence the market for intelligent transportation solutions.

### 2. Economic

- Economic conditions and consumer spending power can impact the willingness of businesses and individuals to invest in new technologies like I2V communication software.
- Economic downturns or recessions can also affect the demand for transportation services and related technologies.

### 3. Sociocultural



- Increasing awareness and concern about road safety and environmental sustainability can drive the demand for innovative transportation solutions like I2V communication software.
- Changes in consumer preferences and expectations related to convenience, comfort, and connectivity can also impact the adoption of new vehicle technologies.

#### 4. Technological

- Advances in artificial intelligence, machine learning, and sensor technology can enhance the performance and capabilities of I2V communication software.
- Integration with other software and systems, such as smart city infrastructure, can create new opportunities for innovation and growth.

#### 5. Environmental

- Environmental concerns and regulations related to air pollution and carbon emissions can drive the adoption of more sustainable transportation solutions like I2V communication software.
- The ability of I2V communication software to optimise traffic flow and reduce congestion can also positively impact the environment.

#### 6. Legal

- Liability and insurance issues related to the use of autonomous vehicles and related technologies can impact the adoption of I2V communication software.

## 5.6 PORTER Analysis

**The Threat of New Entrants** - The threat of new entrants into the I2V communication software market is **medium**. Developing and launching a software solution requires significant research and development investment and specialised technical knowledge. Additionally, the industry is highly regulated, making it difficult for new players to enter.

**Bargaining Power of Buyers** - The bargaining power of buyers in the I2V communication software industry is **moderate**. Although this technology has many potential customers, they are often large corporations or government entities with significant negotiating power. However, the unique capabilities and benefits of the software may give suppliers some leverage in negotiations.

**The threat of Substitutes** - The threat of substitutes in the I2V communication software industry is **low**. Although there are alternative solutions for vehicle navigation, such as GPS systems, none provide the autonomy capabilities while communicating with the infrastructure. This technology's potential cost savings and convenience make it challenging to replace.

**The intensity of Competitive Rivalry** - The intensity of competitive rivalry in the I2V communication software industry is **moderate**. Although few direct competitors exist, many companies offer related solutions and services, such as autonomous vehicle software or smart city technology. The industry is also rapidly evolving, with new players entering and established companies innovating to stay competitive.

**The threat of Government Regulation** - The threat of government regulation in the I2V communication software industry is **high**. The adoption and implementation of autonomous vehicles and related technologies are highly regulated, with governments worldwide developing

laws and policies to ensure safety, security, and privacy. Any changes in these regulations can significantly impact the industry and the demand for I2V communication software.

## 5.7 Risk and Mitigation Measures

Table 7 Risk and Mitigation measures

Types of Risk	Risk	Mitigation
<b>Business Risk</b>	Attracting Potential Investors	Preparing detailed CAPEX and OPEX model attracting more investors in the Business Plan
	Inefficiency in delivering the standards	Developing a timeline chart and adhering to the designated deadlines to ensure timely completion and implementation of the project.
		Conducting thorough market and industry research to inform investment planning.
	Foreign trade and currencies	Conducting market analysis and evaluating trading opportunities to determine the optimal investment and divestment strategies.
<b>Operational Risk</b>	Communication failure	Implement redundancy in communication channels, such as multiple communication protocols and backup communication links, to ensure reliable communication
	Cybersecurity threats	Implement robust security measures, such as data encryption, authentication, and access control, to prevent unauthorized access and data breaches
	Data privacy concerns	Implement measures to protect sensitive data, such as anonymization and data minimization, and comply with relevant data privacy regulations
	Accuracy and reliability of sensor data	Implement quality control measures, such as calibration, validation, and error detection and correction, to ensure the accuracy and reliability of sensor data
<b>Other Risk</b>	Regulatory compliance	Ensure compliance with relevant regulations, standards, and guidelines, such as those related to vehicle-to-infrastructure (V2I) communication and data privacy, through regular audits and updates to policies and procedures
	Development of Technology Standards	Having a core team of Research and Development working in the direction of increasing the efficiency with the new advancements
	Health and Safety of Manpower	Collaborating with a competent resource for regular health checks of the team
	Maintenance of the Sensors	Conducting regular maintenance of the sensors with the GANTT chart for the engineers

## 5.8 Challenges and Future Work

To ensure the success of our company, it is essential to identify and address the various challenges we may encounter along the way. One of the biggest challenges we face is the infrastructure cost. This requires a significant investment of time, money, and resources, which may be challenging to acquire, especially in the early stages of the business. Additionally, we are heavily dependent on network and connectivity, which can be unreliable at times and may cause disruptions to our operations.

Another challenge that we face is the ever-increasing threat of cyber-attacks. As we deal with sensitive information and data, we must take all the necessary measures to protect our system from potential attacks, which can have severe consequences for the company and our customers. In addition to these challenges, there may be other issues that arise in the future that we will need to address and overcome to stay competitive and successful.

Looking towards the future, we must continue to innovate and improve our technology to stay ahead of the competition. This may involve expanding our offerings or developing new products and services to meet the changing needs of our customers. Additionally, we must keep up with the latest industry trends and technologies and invest in research and development to stay at the forefront of our field. Furthermore, we must continue building and maintaining strong relationships with our customers and partners. This includes providing excellent customer service, offering competitive pricing and flexible payment options, and establishing a reputation as a trustworthy and reliable company. Finally, we need to attract and retain top talent in the industry, which can be a challenge but is essential for the growth and success of our business.

## 5.9 Conclusion

The proposed solution is an innovative approach to address the safety and security concerns of customers. It is designed to provide a robust and reliable solution tailored to customers' needs, utilizing the latest technology and industry best practices. The solution aims to increase customers' security and peace of mind, providing them with the highest level of service and support. Overall, the business idea represents an exciting opportunity for customers and the company, offering increased safety and peace of mind to customers and an opportunity to work towards achieving business goals.

## 6. Project Management

Project management is a process of planning, organising, and controlling available resources to achieve specific goals and objectives within a defined timeframe in a systematic way. Project management involves several stages: initiation, planning, execution, monitoring and control, and evaluation. However, this report focuses only on the initiation, planning and execution.

### 6.1 Initiation process

The initiation process is the first stage of the project management process, where the project is defined, and the project team is established. The objectives, scope, and available resources of the project are identified. Even though all of these were given by the supervisor in this group design project, we needed to make a mutual understanding with team members again before starting the project.

### 6.2 Planning process

The planning stage is an essential part of the project management process, where the team members develop a detailed project plan that serves as a roadmap for the project. The project plan provides a framework for executing and controlling the project work. It includes the following elements: Work Breakdown Structure (WBS), RASCI Matrix, project schedule, risk management plan, and communication plan.

#### 6.2.1 Work Breakdown Structure (WBS)

WBS is a management tool that organises the project work into smaller, more manageable components called work packages. The WBS is a visual representation of the project scope, and it helps the project team to identify and organise all the tasks required to complete the project.

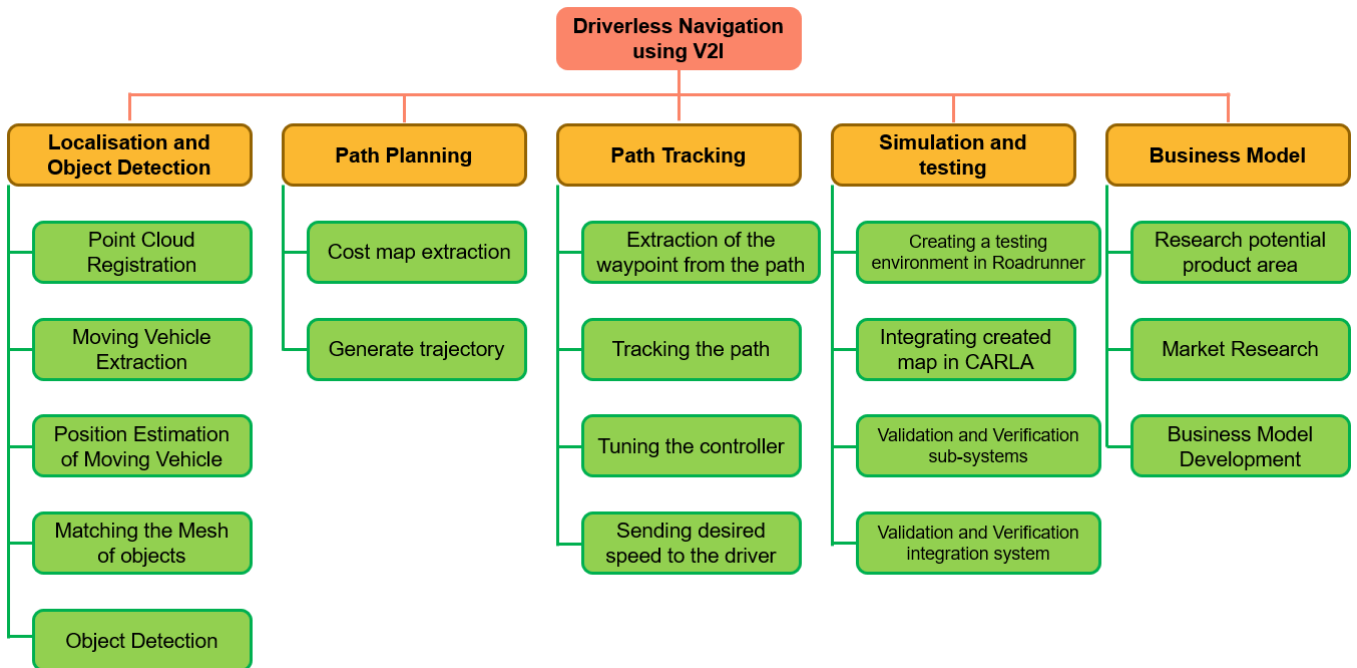


Figure 46 Work Breakdown Structure

### 6.2.2 RASCI Matrix

Once the work packages are identified in the WBS, each must be allocated to team members. Several tools, such as RACI, RASCI, and PARIS, can be used. Using these tools helps to avoid confusion, duplication of effort, and miscommunication among team members. Defining clear roles and responsibilities helps to ensure that tasks are completed on time with satisfactory results. However, this project used the RASCI matrix (Responsible, Accountable, Support, Consult, Inform) as a tool for several reasons (1) straightforward and easy to understand, (2) more comprehensive than RACI, (3) some members were not familiar with the ROS/CARLA environment so that they might need some support from the others. The assigned process was based on team members' preferences and interests. During the project, the RASCI matrix had to be updated to reflect changes in the priority of specific tasks. Some tasks proved to be more challenging than expected, and as a result, adjustments were made to the roles and responsibilities assigned to team members. While the RACI matrix is a helpful tool for clarifying roles and responsibilities at the beginning of a project, it is not always set in stone. As the project progresses, new information may emerge, and priorities may shift, requiring changes to the RASCI matrix.



Table 8 RASCI Matrix

Tasks	Adele	Arsh	Manel	Nimil	Nirmal	Puvit	Shrinidhi	Swapnil	Vinit	Zeliha
Localisation and Object Detection										
Point Cloud Registration	A	I	I	I	I	C	C	S	I	RA
Moving Vehicle Extraction	A	I	I	I	I	C	C	S	I	RA
Position Estimation of Moving Vehicle	A	I	C	C	I	A	S	A	I	RA
Matching the Mesh of objects	A	I	I	C	I	C	C	S	I	RA
Object Detection	C	I	C	C	I	A	C	RA	C	A
Path Planning										
Cost map extraction	I	I	C	I	I	C	RA	S	A	C
Generate trajectory	I	I	A	I	I	S	RA	S	A	I
Path Tracking										
Extraction of the waypoint from the path	I	I	RA	I	I	C	A	C	C	I
Tracking the path	I	I	RA	I	I	C	A	C	C	I
Tuning the controller	I	I	RA	I	I	A	R	C	A	I
Sending desired speed to the driver	C	RA	C	C	RA	C	S	S	C	C
Simulation and testing										
Creating a testing environment in Roadrunner	I	I	I	RA	I	I	S	C	I	I
Integrating created map in CARLA	I	I	I	A	I	C	R	S	I	I
Validation and Verification sub-systems	C	I	A	A	I	A	RA	A	C	A
Validation and Verification integration system	C	I	A	A	I	A	A	RA	C	A
Business Model										
Research potential product area	I	I	I	I	I	A	C	I	RA	I
Market Research	I	I	I	I	I	S	C	I	RA	I
Business Model Development	I	I	I	I	I	S	C	I	RA	I

### 6.2.3 Gantt Chart

In situations with a limited timeframe for the project, it is essential to keep it on schedule to ensure it is completed on time and the objectives are met. Failure to complete a project on time can have several negative consequences. Thus, project schedule tools like the Gantt chart can be used for several reasons (1) this chart shows the timeline of tasks, dependencies between tasks, and task durations. (2) this chart is simple and easy to understand. Due to the growing uncertainty throughout the project, this chart should be updated frequently to reflect the progress because some tasks require more time than anticipated. In addition, to efficiently manage the project, our group decided to use our schedule table in parallel, which shows the due dates and key dates throughout the project period.

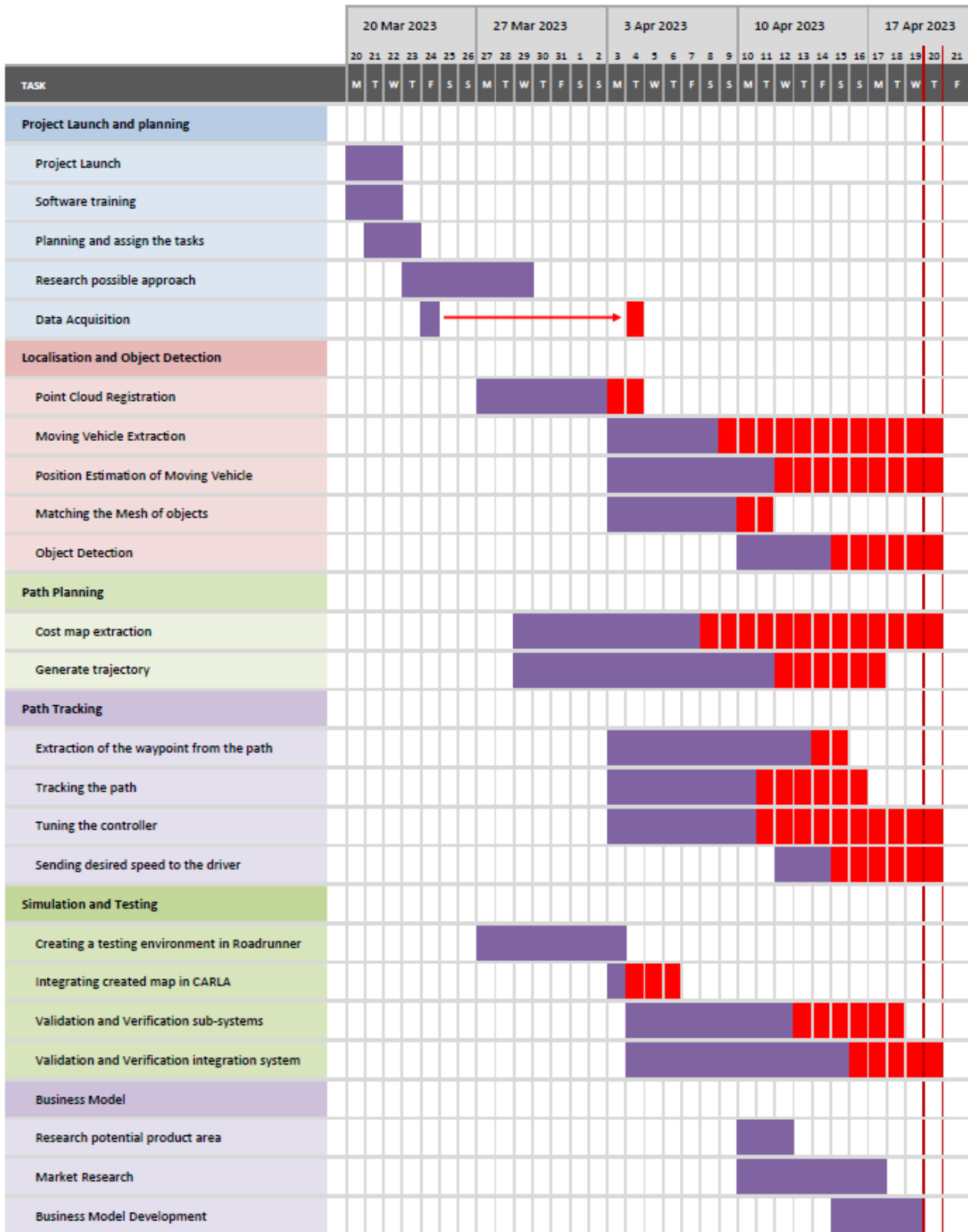


Figure 47 Gantt Chart

Table 9 Key Dates

	Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon
	27-Mar	28-Mar	29-Mar	30-Mar	31-Mar	03-Apr	04-Apr	05-Apr	06-Apr	07-Apr	10-Apr	11-Apr	12-Apr	13-Apr	14-Apr	17-Apr	18-Apr	19-Apr	20-Apr	21-Apr	24-Apr
Due date for document					MoM					MoM					1.MoM 2.Poster					1.MoM 2.Full report 3.Presentation	1.Presentation Assessment 2.Presentation ppt
Consultation day	O					O							O			O					
Hands-on				1.Data Acquisition 2.Remote control			Path following - simu	Path following with other actor - simu						Path following with other actor - car							

## 6.2.4 Risk management plan

Risk management is the process of identifying, assessing, prioritising, and managing potential threats to the project. Risk management aims to minimise the likelihood and impact of potential hazards while maximising success opportunities. Thus, a risk management plan should be generated before starting the project. In this project, our group began by brainstorming any possible risks that might occur during the project, as shown below. However, some risks are beyond our expectations, so we used the risk and issue log to record the newly identified risks and issues, including their descriptions, likelihood of occurrence, potential impact, and mitigated solutions. These logs allowed us to keep track of the various risks and issues that were encountered efficiently. An example of our group's issue log is provided below.

Table 10 Risk Management Plan

ID	Risk Description	Scenario	Former Probability	Former Impact		Status	Action	New Probability	New impact	
				Performance	Schedule				Performance	Schedule
R1	Compatibility issues	1. Hardware and Software incompatibility 2. Repository, Library, and ROS Package incompatibility	1. High 2. High	High	Medium	Addressed	1. Compatibility check before implementing into the project.	1. Medium 2. Medium	Medium	Low
R2	Data loss in CARLA	1. Developed code loss 2. Installed package/library loss CARLA virtual machine constantly reset after use. Thus it needs to be re-install every day.	1. High 2. High	Medium	Medium	Addressed	1. Create required package lists which can follow later. 2. Upload all of the developed code to the Onedrive every day.	1. Low 2. Low	Low	Low
R3	Software bugs and errors	1. Configuration errors (improperly modify the open-source code) 2. Syntax errors in computer languages.	1. High 2. High	High	High	Addressed	1. Try to understand the code before implementing it to know which part needs to be modified for the project. 2. Nominate team members with experience in those errors to figure out the problems.	1. Medium 2. Medium	Medium	Medium
R4	lack of familiarity or experience with ROS development/simulation program	1. Inadequate number of members or insufficient skill leading to the project delays	High	High	High	Addressed	1. Provide training sessions from experienced members. 2. Analyze required skills that need to use in the project and assign them to the members to focus on that skill.	Medium	Low	Low

Table 11 Issue log table

ID	Description	Type	Date Raised	Raised By	Assigned to	Priority	Status	Closure Date	Comment
1	VSCode could not find the pcl header file.	Problem	03-Apr	Zeliha	Zeliha	High	Closed	04-Apr	It was solved by following the supervisor's advice.
2	Pure Pursuit Algorithm could not stop when it reached the target point.	Problem	07-Apr	Shrinidhi	Shrinidhi and Manel	High	Closed	12-Apr	It was solved by debugging the parameter in the algorithm.
3	After creating a map in Roadrunner, it could not implement in CARLA.	Problem	11-Apr	Nimil	Shrinidhi	High	Closed	13-Apr	It was solved by following the instruction given by the supervisor.
4	unable to match coordinates of point cloud data and CARLA.	Problem	11-Apr	Nimil	Nimil and Shrinidhi	High	Open	-	Still pending as of now
5	A lot of noise in point cloud data, after implement filter it also filtered the moving vehicle.	Problem	12-Apr	Zeliha	Zeliha	High	Open	-	Still pending as of now

## 6.2.5 Communication plan

A communication plan could help ensure that all members know the project's goals, status, progress, and issues, which can help to prevent misunderstanding, conflicts, overlapping, and delays. Our group created the communication plan as shown below.

Table 12 Communication Plan

Topic	Method	Required member	Frequency
Individual Progress Update	Microsoft Teams/ In-person	Everyone	every other day
Overall progress and project monitoring	Microsoft Teams/ In-person	Everyone	Weekly after consultation session
Sub-system urgent issues	Microsoft Teams/ In-person / Whatsapp	Relevant members	Not fixed
Consultation Sessions	Microsoft Teams/ In-person	Everyone	Fixed date (27 Mar, 3 Apr, 12 Apr, 17 Apr)
Hands-on Sessions	In-person	Everyone	Fixed date (23 Mar, 30 Mar, 4 Apr, 5 Apr, 17 Apr)
Presentation Practice	Microsoft Teams/ In-person	Everyone	Fixed date (12 Apr, 21 Apr, 22 Apr, 23 Apr)
Final Report Review	Microsoft Teams/ In-person	Everyone	Fixed date (20 Apr)

## 6.3 Execution

Execution is a crucial project management phase involving implementing the project plan. It entails managing resources effectively, monitoring the progress of individual tasks, ensuring deliverables meet quality standards, and addressing any modifications to the project plan that may occur during execution. Finally, a mindset of continuous learning and improvement is required for effective execution. All members must be willing to adapt to new information, identify and address issues promptly, and collaborate to achieve project objectives. By doing so, we can ensure that our projects are effectively completed with minimal disruptions or delays.

# 7. Challenges and Lessons Learned

## 7.1 Challenges

When conducting a group project, challenges are inevitable. Group members need to be aware of and proactively work to address these potential challenges. Group members can work more effectively together and increase the likelihood of producing a high-quality project if they recognise the obstacles and develop strategies to overcome them. Thus, some technical and non-technical that we faced during the project are shown below.

### 7.1.1 Technical

- The steering actuator of the vehicle responds a little slower with respect to vehicle speed and the ROS controller node; hence, the team found it difficult during real time steering testing in hands on sessions.
- OXTS INS sensor gives offset values up to ~4.0 meters during various test runs, It affected accuracy of path tracking algorithms.
- The CARLA virtual machine caused a problem for the team because it removed all data after each use. Also, system is in-accessible for most of the times. Every day, the team had to reinstall packages, libraries, and settings.

- Despite the project's initial intention to utilise an existing repository, the team frequently found itself compelled to modify it to meet the specific requirements significantly.
- The project included numerous sub-systems, each of which required the selection of suitable algorithms. However, many algorithms made it difficult for the team to determine the most appropriate algorithms for each sub-system.
- The team sometimes encountered bugs during the project that were challenging and time-consuming to fix.
- Although the team had obtained positive results after simulating the project's testing, they encountered several issues when testing the actual vehicle.

### 7.1.2 non-Technical

- Due to most team members' unfamiliarity with ROS and the Linux operating system, task organisation proved to be a challenge at the beginning of the project. This caused considerable time to be invested in understanding these technical aspects.
- When time is limited, changing the project's approach may not always be possible. The team may need to utilise readily available resources and methods, even if they are not perfect.
- Team members may struggle to balance competing priorities, as they may have other responsibilities and tasks that require their attention and time outside the project.

## 7.2 Lesson Learned

Throughout the project, the team encountered many technical and non-technical challenges that required them to adapt and innovate to ensure its success. Some of the lesson learned has been summarised below.

- Collaboration is essential to the success of any activity. The success or failure of a project may depend on the capacity of team members to collaborate effectively, utilising their unique perspectives and skill sets.
- Each team member's participation and contribution are crucial to the success of a project. Team members must be dedicated to and collaborate to achieve the project's goals.
- Beginning a project by underestimating the difficulty of specific tasks can be a fatal error. It can result in exaggerated expectations and timelines, leading to frustration, delays, and the potential inability to meet project objectives.
- Time management is essential for assuring the overall quality of a project's outcomes. Ineffective time management can result in rushed work, missed deadlines, and a lack of attention to detail, negatively impacting the project's content.
- While several codes are available online, modifying them to meet the project's specific requirements can be challenging. Modifying code often requires a deep understanding of the code structure and its underlying concepts, which can be time-consuming and challenging. Additionally, modifying code can introduce unexpected errors and bugs, requiring further debugging and testing.
- Parallel task completion can be crucial to the success of a project, even if the output of one task is dependent on the input of another. Planning and coordination amongst team members help determine which duties can be simultaneously worked on without compromising quality or progress.
- To achieve the best results for a project, it is necessary to evaluate every potential algorithm and select the most appropriate one.
- Task prioritisation is crucial to achieving a project's overall goals, as it enables team members to focus on the most critical tasks and allocate resources effectively. In some



cases, prioritisation requires a trade-off between achieving a perfect result or an acceptable one based on time and technical feasibility.

- Effective communication is essential to keep team members on the same page. By communicating regularly and clearly, team members can stay informed about project progress, changes in scope, and potential roadblocks or issues.

Finally, the most valuable lesson learned from this project is the importance of hands-on experience. By working on the project and interacting directly with its technical components and challenges, team members obtained invaluable insights and skills that could not have been acquired through theory alone. This hands-on experience enabled team members to acquire practical skills, troubleshoot issues in real-time, and better understand the project's technical requirements and constraints. Moreover, hands-on experience can assist team members in identifying potential development and innovation areas, encouraging creativity and forward-thinking, which can improve their overall skills and capabilities, contributing to their success in both the current project and the future.

## 8. Future work

After completing the project, the group identified several areas for future development that could improve the project's performance. These recommendations are founded on the project-related experiences of our group. Some potential future developments could include:

- Regarding the mesh matching part, we could implement an algorithm that generates the mesh of the vehicle passing directly by combining the point clouds captured by the LiDARS.
- Regarding path tracking, it would be necessary to implement an adaptive lookahead distance for different driving situations. Also, the implementation of this distance in the code needs to be improved as it may not be accurate depending on the type of path. Finally, finding the closest waypoint to the vehicle's location may result in getting an unwanted waypoint, so improvements are needed in that part of the code.
- The move\_base ros navigation stack is capable of generating optimised path but there are limitation in terms of maintaining the lane of the road. Frenet path planning shows promising results as per the research and can be explored further. Also, obstacle layer through external point cloud LIDAR data needs to be further improved for optimising the paths in dynamic environment.
- Investigate alternative algorithms for each sub-system.
- V&V in the real-world scenario to understand the system response.

## Reference

- ackermann\_msgs - ROS Wiki. (n.d.). Ackermann\_Msgs - ROS Wiki.  
[http://wiki.ros.org/ackermann\\_msgs](http://wiki.ros.org/ackermann_msgs)
- Actors - CARLA Simulator. (n.d.). Retrieved April 20, 2023, from  
[https://carla.readthedocs.io/en/latest/core\\_actors/#spawning](https://carla.readthedocs.io/en/latest/core_actors/#spawning)
- Agishev, R., Comerón, A., Bach, J., Rodriguez, A., Sicard, M., Riu, J., & Royo, S. (2013). Lidar with SiPM: Some capabilities and limitations in real environment. *Optics and Laser Technology*, 49, 86–90. <https://doi.org/10.1016/j.optlastec.2012.12.024>
- Ali, W., Abdelkarim, S., Zidan, M., Zahran, M., & Sallab, A. El. (2018). YOLO3D: End-to-end real-time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud.

- Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11131 LNCS, 716–728.  
[https://doi.org/10.1007/978-3-030-11015-4\\_54](https://doi.org/10.1007/978-3-030-11015-4_54)
- Alvarez León, L. F., & Aoyama, Y. (2022). Industry emergence and market capture: The rise of autonomous vehicles. *Technological Forecasting and Social Change*, 180.  
<https://doi.org/10.1016/j.techfore.2022.121661>
- Amer, N. H., Zamzuri, H., Hudha, K., & Kadir, Z. A. (2017). Modelling and Control Strategies in Path Tracking Control for Autonomous Ground Vehicles: A Review of State of the Art and Challenges. *Journal of Intelligent & Robotic Systems*, 86(2), 225–254. <https://doi.org/10.1007/s10846-016-0442-0>
- Bendiab, G., Hameurlaine, A., Germanos, G., Kolokotronis, N., & Shiaeles, S. (2023). Autonomous Vehicles Security: Challenges and Solutions Using Blockchain and Artificial Intelligence. *IEEE Transactions on Intelligent Transportation Systems*, 1–24.  
<https://doi.org/10.1109/tits.2023.3236274>
- Burkacky, O., Deichmann, J., Guggenheimer, M., & Kellner, M. (2023). *Outlook on the automotive software and electronics market through 2030*.
- Damaj, I. W., Yousafzai, J. K., & Mouftah, H. T. (2022). Future Trends in Connected and Autonomous Vehicles: Enabling Communications and Processing Technologies. *IEEE Access*, 10, 42334–42345. <https://doi.org/10.1109/ACCESS.2022.3168320>
- Danquah, W. M., & Altılar, D. T. (2020). Vehicular Cloud Resource Management, Issues and Challenges: A Survey. In *IEEE Access* (Vol. 8, pp. 180587–180607). Institute of Electrical and Electronics Engineers Inc.  
<https://doi.org/10.1109/ACCESS.2020.3027637>
- Dokur, O., & Katkoori, S. (2022). CARLA Connect: A Connected Autonomous Vehicle (CAV) Driving Simulator. *Proceedings - 2022 IEEE International Symposium on Smart Electronic Systems, ISES 2022*, 656–659.  
<https://doi.org/10.1109/ISES54909.2022.00146>
- Dosovitskiy, A., Ros, G., Codevilla, F., López, A., & Koltun, V. (2018). *CARLA: An Open Urban Driving Simulator*.
- Gurel, C. S. (2018). *REAL-TIME 2D AND 3D SLAM USING RTAB-MAP, GMAPPING, AND CARTOGRAPHER PACKAGES*. <https://www.researchgate.net/publication/326986124>
- Gwak, J., Jung, J., Oh, R. D., Park, M., Rakhimov, M. A. K., & Ahn, J. (2019). A review of intelligent self-driving vehicle software research. *KSII Transactions on Internet and Information Systems*, 13(11), 5299–5320. <https://doi.org/10.3837/tiis.2019.11.002>
- Hung, N., Rego, F., Quintas, J., Cruz, J., Jacinto, M., Souto, D., Potes, A., Sebastiao, L., & Pascoal, A. (2023). A review of path following control strategies for autonomous robotic vehicles: Theory, simulations, and experiments. *Journal of Field Robotics*, 40(3), 747–779. <https://doi.org/10.1002/ROB.22142>
- Jellid, K., & Mazri, T. (2021). DSRC vs LTE V2X for Autonomous Vehicle Connectivity. *Lecture Notes in Networks and Systems*, 183, 381–394. [https://doi.org/10.1007/978-3-030-66840-2\\_29](https://doi.org/10.1007/978-3-030-66840-2_29)
- Jeong, J., Shen, D., Kim, N., Karbowski, D., & Rousseau, A. (2019). Online implementation of optimal control with receding horizon for eco-driving of an electric vehicle. *2019 IEEE Vehicle Power and Propulsion Conference, VPPC 2019 - Proceedings*.  
<https://doi.org/10.1109/VPPC46532.2019.8952220>
- Ji, J., Khajepour, A., Melek, W. W., & Huang, Y. (2017). Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, 66(2), 952–964.  
<https://doi.org/10.1109/TVT.2016.2555853>
- Kebbaty, Y., Ait-Oufroukh, N., Ichalal, D., & Vigneron, V. (2022). Lateral control for autonomous wheeled vehicles: A technical review. *Asian Journal of Control*.  
<https://doi.org/10.1002/ASJC.2980>

- Klauer, C., Schwabe, M., & Mobalegh, H. (2020). Path Tracking Control for Urban Autonomous Driving. *IFAC-PapersOnLine*, 53(2), 15705–15712.  
<https://doi.org/10.1016/j.ifacol.2020.12.2569>
- Kumar, P., & Ali, K. B. (2022). Intelligent Traffic System using Vehicle to Vehicle (V2V) & Vehicle to Infrastructure (V2I) communication based on Wireless Access in Vehicular Environments (WAVE) Std. *2022 10th International Conference on Reliability, Infocom Technologies and Optimisation (Trends and Future Directions), ICRITO 2022*.  
<https://doi.org/10.1109/ICRITO56286.2022.9964590>
- Kusuma, Z. D. S., Indriawati, K., Widjiantoro, B. L., Hija, A. I., & Nurhadi, H. (2023). Optimal Trajectory Planning Generation for Autonomous Vehicle Using Frenet Reference Path. 480–484. <https://doi.org/10.1109/ISRITI56927.2022.10052833>
- Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (n.d.). *PointPillars: Fast Encoders for Object Detection from Point Clouds*. Retrieved 20 April 2023, from <https://github.com/nutonomy/second.pytorch>
- Leminen, S., Rajahonka, M., Wendelin, R., Westerlund, M., & Nyström, A. G. (2022). Autonomous vehicle solutions and their digital servitisation business models. *Technological Forecasting and Social Change*, 185.  
<https://doi.org/10.1016/j.techfore.2022.122070>
- Lu, R., Zhang, L., Ni, J., & Fang, Y. (2020). 5G Vehicle-to-Everything Services: Gearing up for Security and Privacy. *Proceedings of the IEEE*, 108(2), 373–389.  
<https://doi.org/10.1109/JPROC.2019.2948302>
- Magnusson, M., Nüchter, A., Lörken, C., Lilienthal, A. J., & Hertzberg, J. (2009). Evaluation of 3D registration reliability and speed-A comparison of ICP and NDT. *Proceedings - IEEE International Conference on Robotics and Automation*, 3907–3912.  
<https://doi.org/10.1109/ROBOT.2009.5152538>
- Magnusson, M., Nüchter, A., Lörken, C., Lilienthal, A. J., & Hertzberg, J. (2009). Evaluation of 3D registration reliability and speed-A comparison of ICP and NDT. *Proceedings - IEEE International Conference on Robotics and Automation*, 3907–3912.  
<https://doi.org/10.1109/ROBOT.2009.5152538>
- Malik, R. Q., Ramli, K. N., Kareem, Z. H., Habelalmatee, M. I., & Abbas, H. (2020). A review on vehicle-to-infrastructure communication system: Requirement and applications. *2020 3rd International Conference on Engineering Technology and Its Applications, IICETA 2020*, 159–163.  
<https://doi.org/10.1109/IICETA50496.2020.9318825>
- Marin-Plaza, P., Hussein, A., Martin, D., & De La Escalera, A. (2018). Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous Vehicles. *Journal of Advanced Transportation*, 2018. <https://doi.org/10.1155/2018/6392697>
- Mendhe, A., Chaudhari, H. B., Diwan, A., Rathod, S. M., & Sharma, A. (2022). Object Detection and Tracking for Autonomous Vehicle using AI in CARLA. *2022 International Conference on Industry 4.0 Technology, I4Tech 2022*.  
<https://doi.org/10.1109/I4TECH55392.2022.9952468>
- Meyer, G. P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., & Wellington, C. K. (2019). LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June*, 12669–12678.  
<https://doi.org/10.1109/CVPR.2019.01296>
- Mohd Shamsuddin, P. N. F., Ramli, R. M., & Mansor, M. A. Bin. (2021). Navigation and motion control techniques for surface unmanned vehicle and autonomous ground vehicle: A review. *Bulletin of Electrical Engineering and Informatics*, 10(4), 1893–1904.  
<https://doi.org/10.11591/EEI.V10I4.3086>
- Naotunna, I., & Wongratanaphisan, T. (2020). Comparison of ROS Local Planners with Differential Drive Heavy Robotic System. *International Conference on Advanced Mechatronic Systems, ICAMEchS, 2020-December*, 1–6.  
<https://doi.org/10.1109/ICAMECHS49982.2020.9310123>

- Nayak, B. P., Hota, L., Kumar, A., Turuk, A. K., & Chong, P. H. J. (2022). Autonomous Vehicles: Resource Allocation, Security, and Data Privacy. *IEEE Transactions on Green Communications and Networking*, 6(1), 117–131. <https://doi.org/10.1109/TGCN.2021.3110822>
- Paden, B., Čáp, M., Yong, S. Z., Yershov, D., & Frazzoli, E. (2016). A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1), 33–55. <https://doi.org/10.1109/TIV.2016.2578706>
- Ramakrishna, S., Luo, B., Kuhn, C. B., Karsai, G., & Dubey, A. (2022). ANTI-CARLA: An Adversarial Testing Framework for Autonomous Vehicles in CARLA. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2022-October*, 2620–2627. <https://doi.org/10.1109/ITSC55140.2022.9921776>
- Rokonuzzaman, M., Mohajer, N., Nahavandi, S., & Mohamed, S. (2021). Review and performance evaluation of path tracking controllers of autonomous vehicles. *IET Intelligent Transport Systems*, 15(5), 646–670. <https://doi.org/10.1049/ITR2.12051>
- ROS.org. (n.d.-a). *gmapping - ROS Wiki*. Retrieved April 21, 2023, from <http://wiki.ros.org/gmapping>
- ROS.org. (n.d.-b). *map\_server - ROS Wiki*. Retrieved April 21, 2023, from [http://wiki.ros.org/map\\_server](http://wiki.ros.org/map_server)
- ROS.org. (n.d.-c). *move\_base - ROS Wiki*. Retrieved April 21, 2023, from [http://wiki.ros.org/move\\_base](http://wiki.ros.org/move_base)
- ROS.org. (n.d.-d). *teb\_local\_planner - ROS Wiki*. Retrieved April 21, 2023, from [http://wiki.ros.org/teb\\_local\\_planner](http://wiki.ros.org/teb_local_planner)
- SAE International. (2021). *SURFACE VEHICLE RECOMMENDED PRACTICE Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*.
- Shenzhen da xue. (2018). *2018 the 3rd Optoelectronics Global Conference (OGC 2018) : 4 September -7 September, 2018, Shenzhen, China*.
- Sariff, N., & Buniyamin, N. (2006). An overview of autonomous mobile robot path planning algorithms. SCORED 2006 - Proceedings of 2006 4th Student Conference on Research and Development "Towards Enhancing Research Excellence in the Region," 183–188. <https://doi.org/10.1109/SCORED.2006.4339335>
- Singh, S., & Saini, B. S. (2021). Autonomous cars: Recent developments, challenges, and possible solutions. *IOP Conference Series: Materials Science and Engineering*, 1022(1), 012028. <https://doi.org/10.1088/1757-899X/1022/1/012028>
- Sprinkle, J., Eklund, J., Gonzalez, H., Grøtli, E., Sanketi, P., & Moser, M. (2008). *Recovering Models of a Four-Wheel Vehicle Using Vehicular System Data*.
- Sun, P., Sun, C., Wang, R., & Zhao, X. (2022). Object Detection Based on Roadside LiDAR for Cooperative Driving Automation: A Review. *Sensors* 2022, Vol. 22, Page 9316, 22(23), 9316. <https://doi.org/10.3390/S22239316>
- Theers, M., & Singh, M. (2022). Pure Pursuit controller representation for a bicycle model [Image]. Recovered from <https://thomasfermi.github.io/Algorithms-for-Automated-Driving/Control/PurePursuit.html>
- Wang, S., Ma, Y., Liu, J., Yu, B., & Zhu, F. (2022). Readiness of as-built horizontal curved roads for LiDAR-based automated vehicles: A virtual simulation analysis. *Accident Analysis and Prevention*, 174. <https://doi.org/10.1016/j.aap.2022.106762>
- Wu, J. (n.d.). *An Automatic Procedure for Vehicle Tracking with a Roadside LiDAR Sensor Background Filtering and Lane Identification Background Filtering*. Retrieved April 19, 2023, from [www.ite.org](http://www.ite.org)
- Wu, J. (n.d.). *An Automatic Procedure for Vehicle Tracking with a Roadside LiDAR Sensor Background Filtering and Lane Identification Background Filtering*. Retrieved 19 April 2023, from [www.ite.org](http://www.ite.org)
- Zhang, J., Xiao, W., Coifman, B., & Mills, J. P. (2019). *IMAGE-BASED VEHICLE TRACKING FROM ROADSIDE LIDAR DATA*. <https://doi.org/10.5194/isprs-archives-XLII-2-W13-1177-2019>

- Zhang, J., Xiao, W., Coifman, B., & Mills, J. P. (2019). *IMAGE-BASED VEHICLE TRACKING FROM ROADSIDE LIDAR DATA*. <https://doi.org/10.5194/isprs-archives-XLII-2-W13-1177-2019>
- Zhao, J., Xu, H., Liu, H., Wu, J., Zheng, Y., & Wu, D. (2019). Detection and tracking of pedestrians and vehicles using roadside LiDAR sensors. *Transportation Research Part C: Emerging Technologies*, 100, 68–87. <https://doi.org/10.1016/J.TRC.2019.01.007>
- Zhou, Y., & Tuzel, O. (2017). VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 4490–4499. <https://doi.org/10.1109/CVPR.2018.00472>