THE UNIVERSITY OF WESTERN AUSTRALIA

The University of Western Australia
School of Mechanical and Chemical Engineering

Path Planning for Autonomous Driving Vehicles

Jiajian Shao
22380597

Submitted 30th Oct 2019

Supervisor: Professor Dr. Thomas Bräunl

Word Count: 6841

Acknowledgments

I want to thank my research project supervisor, Professor Thomas Bräunl, for his support and guidance over the whole research project. Also, I want to thank him for offering me the valuable opportunity to have this meaningful research project.

I want to thank all members of the REV team for their collaboration and assistance. Special thanks to the previous year's student Mr. Chao Zhang for his insightful suggestions and patient instructions. Special thanks to Special thanks to the previous year's student Craig Brogle for assisting me in setting up the simulator and adjusting the simulator when needed.

Abstract

This project is essentially a “design and build” project focusing on the modification and improvement of the existing path planning software for the UWA Electric SAE (Society of Automotive Engineers) Race Car.

The modified local path planner aims to enable the SAE car to achieve autonomous driving in unknown environments without predefined geographical information. The modified local path planner is based on a customised probabilistic roadmap algorithm to enable the SAE car to avoid obstacles. This algorithm relies on the information provided by the odometry and LiDAR (light detection and ranging) sensors on the SAE car to detect obstacles. The software is composed in C++ programming language, operated in Linux environment and coordinated by ROS (Robot Operating System) software framework.

With only a single strategy of path planning, the existing local path planner tends to fail to find a valid path in some complicated configuration space. As a result, the most significant feature in the modification to the existing path planner is the prediction feature, which allows the vehicle to predict its following manoeuvres. With the modified local path planner, when the SAE car detects possible collision at a predicted manoeuvre, it will roll back to the prior predicted manoeuvres iteratively until it finds another valid path with a different steering angle. The modification makes use of more information provided by the sensors and certainly enhances the autonomous driving capacity and flexibility of the existing local planner.

Table of Content

List of Figures

List of Acronyms

AI	Artificial Intelligence
GPS	Global Positioning System
IMU	Inertial Measurement Unit
LiDAR	Light Detection and Ranging
REV	Renewable Energy Vehicles
ROS	Robot Operating System
SAE	Society of Automotive Engineers
SLAM	Simultaneous Localisation and Mapping
UWA	University of Western Australia

1 Introduction

This chapter provides an overview of the project, including the motivation for this work and the background of the project. Besides, a detailed statement of the current problems and a specific description of the goal of the project are shown in this chapter.

1.1 Motivation

Nowadays, developing autonomous driving is a new trend in the automotive industry. Many world-renowned companies such as Google, Tesla, Toyota and Mercedes-Benz have been developing autonomous driving systems for many years. One of the currently available autonomous systems called Autopilot [1] developed by Tesla has gained great success in the commercial market. Inspired by Tesla's Autopilot, many automotive companies started stepping into autonomous driving research in the last few years.

There are a lot of potential benefits of autonomous driving. Above all, it is predicted to reduce the number of car accidents significantly. Theoretically, all of the traffic collisions due to human errors, such as delayed reaction time, poor judgement and reckless driving, can be totally eliminated by autonomous driving. An American consulting company McKinsey & Company predicted that widespread use of autonomous vehicles could reduce the number of car accidents in the US by 90%, prevent up to US$190 in property damages and save thousands of lives [2]. Second, autonomous driving can release people from the steering wheels and, in turn, decrease the requirement for driving skills. With autonomous driving systems, the young, elderly and handicapped can even use their cars individually. Also, people can spare more time for leisure and work while travelling. Third, autonomous driving can reduce labour cost. The logistics industry is booming due to the rapid development of the e-commerce industry. However, the wages of drivers account for a large proportion of the total cost. With autonomous driving trucks, the cost in the logistics industry will decrease significantly. The same thing will happen to the transport industry as well.

All in all, it is obvious that the research in autonomous driving vehicles at UWA is corresponding to the development in the automotive industry, and the technologies developed by exploiting the Autonomous SAE Cars have huge benefits for future applications.

1.2 Background

The UWA Self Driving Formula SAE (Society of Automotive Engineers) Race Car project is part of the REV (Renewable Energy Vehicle) project in the UWA, which is led by Professor Thomas Bräunl. The SAE car was a petrol-car designed and built by UWA engineering students for formula SAE competition. In the last few years, the REV team converted it into an electric-powered vehicle equipped with two rear motors and a drive-by-wire system [3].

After the conversion, the REV team made the SAE car a self-driving vehicle, which was achieved by equipping the SAE car with an array of sensors, including LIDARs (Light Detection and Ranging), an IMU (Inertial Measurement Unit) integrated with GPS (Global Positioning System), cameras and odometry sensors. Apart from the sensors, a Jetson TX1 computer with Ubuntu 16.0.4 operating system installed was built in the car to process data. At present, all the autonomous driving programs on the car are under the coordination of a software framework called ROS (Robot Operating System). Up to now, the SAE car already has a fundamental autonomous driving capacity, which enables it to drive through a simple cone track. In order to have better computational performance and support more sophisticated software on the car, the team plan to upgrade the current processing unit to an Nvidia Jetson AGX Xavier [4] with Ubuntu 18.0.4 operating system at the end of 2019.

Figure 1: Formula SAE Racing Car

1.3 Problem Statement

This project aims to improve the performance of the existing autonomous driving software and develop new software for new autonomous driving scenarios as well. At the current stage, there are basically two autonomous driving scenarios in the scope of the project, cone driving and real road driving in UWA. No matter in which driving environment, typically, there are two autonomous driving processes involved. First, the SAE car drives in a new environment without geographical information provided in advance. A local path planner is needed to help the car drive in and explore the new environment. Ideally, the car will use SLAM (Simultaneous Localisation and Mapping) to map the driving environment at the same time. Second, after the car has collected enough geographical information of the environment, a global path planner can be deployed to assist the car in finding the optimum path in the particular environment to achieve better driving performance. Therefore, two different path planners should be developed for autonomous driving.

The challenge of path planning can be subdivided into two tasks, cones following and waypoints navigation in terms of the project goal in 2019. These two tasks are the fundamental tasks for the SAE car, which should be completed and achieve a stable stage before higher-level tasks are assigned to this topic.

- **Cones Following**

This task is the first task for path planning. It is assumed that the SAE car will mostly drive in a track consist of cones since it was built for competitions. When the SAE car starts to dive on a cone track, there will not be any geographical information provided beforehand. That means it needs to make use of the sensors on it to detect the track and find its way while it is driving, which requires instantaneous path generation in terms of the cones detected, the physical size of the car and the manoeuvre characteristics of the car. Besides, the car is required to stop whenever there are unexpected obstacles in the track for safety reasons. A fundamental local path planner program, which was developed by a previous year's student Lai [5], can basically finish all the tasks of cones following. However, it can only be applied to some cone tracks with simple layouts due to its single path planning strategy. It tends to fail when it is deployed on a track with complicated curves.

- **Waypoints Navigation**

Waypoints navigation is the second stage of path planning. After the SAE finish the first lap in a cone track with the local path planer, it should have recorded the geographical information of the track and in turn, generated a map. Based on that, the major task for the SAE car is to find the optimum path in terms of safety, speed, smoothness of driving. With the map, the SAE car can be navigated by several waypoints generated based on the map, representing the base frame of the optimum path. Similar to the local path planner, a global path planner was developed by a previous year's student Podolski [6]. However, there are a few problems with the existing global path planner. First, the current software framework used on the SAE car is ROS, but the global path planner was not designed for ROS, which means it is not compatible with ROS and thus, cannot be integrated into the present autonomous driving system directly. On the other hand, the software is not actually a path planner for controlling autonomous vehicles. It is more like a simulator that can generate plots indicating the optimum path based on the map input.

- **Real Roads Driving**

Real roads driving is the ultimate stage of the current scope of the SAE car project, which requires the SAE car drive autonomously on real roads. In fact, real roads driving is not much different from driving in a cone track, except that there is no cone on the road, so the SAE car has to follow road edges and lane markings. Besides, the SAE car is required to react with not only static objects, such as plants and buildings, but also dynamic objects such as pedestrians, moving vehicles, etc. [5] Ideally, the SAE car should also take into account the information captured from signpost and traffic lights while driving. Since the environment of real roads driving is more complicated than that of cone tracks, computer vision technology, which has not been implemented on the SAE car, is essential for the task. This problem can be solved only when the local and global path planners and new computer vision technology for the SAE car are well-developed.

1.4 Project Goal

There are two general goals for the SAE car project.

- **Driving autonomously in cone tracks**

The SAE car is required to drive autonomously through an unknown (no map provided in advance) track consisting of cones. In the first lap, the vehicle should use sensors to

detect the environment and use SLAM to map the environment at the same time. After that, the car is required to find the optimum path according to the map derived in the first lap. It is expected to have better performance and spend less time with the optimum path.

- **Driving autonomously along internal roads in UWA**

The SAE car should drive autonomously through an internal road in UWA with a map provided in advance and the predefined coordinates of the destination point. This goal is way more challenging than the first one due to dynamic obstacles on the road. The car is required to detect the environment and respond to any changes in the environment in real time so as to avoid collisions. On the other hand, new computer vision software is needed to detect road edges and lane markings.

2 Literature Review

This chapter specifies the state of the art of the existing local path planner (hereinafter referred to as “the old planner”), designed by Lai [5]. The underlying methodologies and mathematical principles of the old planner are explained and discussed.

2.1 Kinematic Model

The SAE car is a kinematic system subject to a couple of physical constraints, such as the steering limit, the size, etc. In other words, the manoeuvres of the SAE car must comply with particular rules. As a result, in order to control the SAE car in accordance with its physical constraints, the first step is to sort out its kinematic attributes. Certain measurements have been made by Lai [5] in 2018, and the physical structure of the car remains the same in 2019.

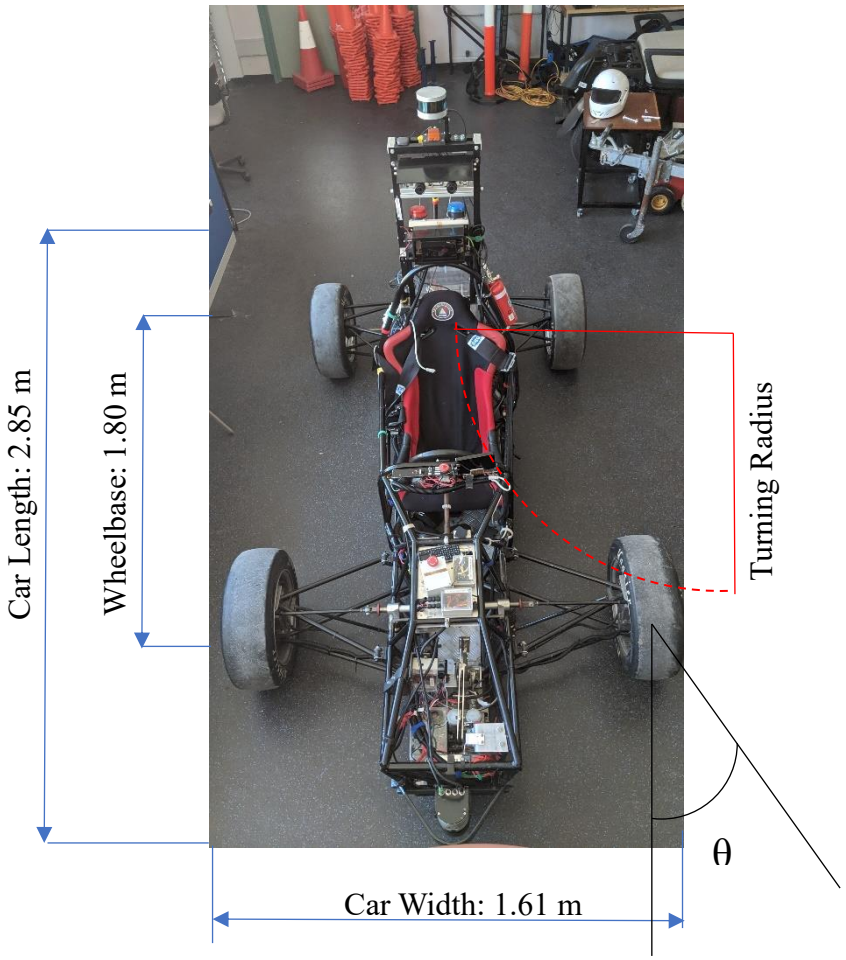

Figure 2: Dimensions of the SAE car

All the critical dimensions are shown in Figure 2, where θ suggests the steering angle of the SAE car. It was measured that the steering limit $\theta_{lim} = \pm 25°$, which means the total steering range is $50°$, from $-25°$ (right) to $+25°$ (left).

The current mechanical steering system is considered not 100% consistent with the Ackermann steering model. The simplified model derived by Lai is applied to represent the mechanical attributes of the steering system. The turning radius can be determined by the wheelbase and the steering angle:

$$Turning\ Radius = \frac{Wheelbase}{\sin(|Steering\ Angle|)} \quad (1)$$

Since the wheelbase of the car is 1.8 m and the maximum steering angle is $25°$, the minimum turning radius can be derived, which is approximately 4.26 m.

The kinematic model of the car in the planner design has to be precisely corresponding

to the actual kinematic characteristics of the car. Otherwise, the command signal sent from the planner will not be able to control the car in an expected way.

2.2 Path Planning

The path planning algorithm of the old planner is based on a customised probabilistic road map method originated by Lai. The old planner is reliant on the SICK LiDAR mounted on the front of the car to detect cones in the track and odometry sensors to determine its orientation. The old planner allows the car to search a small segment of the track for valid paths in real time, which is similar to the graph-search method [7]. After that, the car will evaluate and follow the optimum path until it cannot find any valid paths in the track.

The general logic of the cone following procedure in the old planner is shown in Algorithm 1.

Algorithm 1: Cones Following

input: cones information, processing range

output: actual steering angle

```
if (the cones locations are inside the processing range) then
    keep the cones;
else then
    exclude the cones;
end if
evaluate collision ranges against the clearance range;
exclude collision ranges from all the derived steering ranges;
if (no steering range left) then
    stop the car;
else then
    compare the sizes of all valid steering ranges;
    select the largest steering range as the desired one;
    if (there are more than one equally largest steering ranges) then
        select the one with smaller angle change to the current orientation;
    end if
end if
adopt the mean value of the desired steering range as the actual steering angle;
```

It is worth mentioning that, the path evaluation strategy of the old planner is that, after the collision evaluation, if there are several valid steering ranges left, the planner will select the largest one as the desired steering range. Otherwise, if there is more than one steering range with the same size, it will select the one with a smaller angle change to the current heading as the desired steering range. Eventually, the planner employs the

mean value of the desired steering range as the desired steering angle. The theoretical result of the old planner is shown in Figure 3.

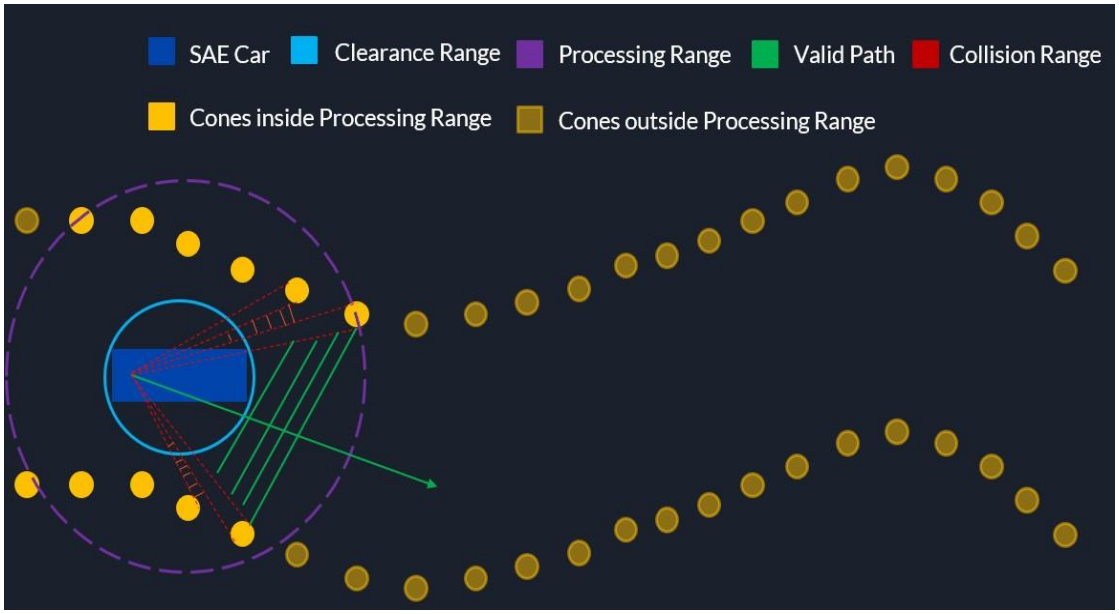

Figure 3: The theoretical result of the old planner

2.3 Collision Evaluation

In the collision evaluation process, all the cones within the processing range will be evaluated against the clearance range by iteration. Normally, for a single cone, there are two ways to avoid collision with it, driving past its left boundary or its right boundary, as is shown in Figure 4 and Figure 5. When the coordinates of a cone (x, y) are derived by sensors, the turning radius can be determined by equation (2) [5].

$$Turning\ Radius = \frac{x}{\sin(2 \times atan2(x,y))} - shift\#(2)$$

The condition of collision-free steering is $Turning\ Radius \geq 0$.

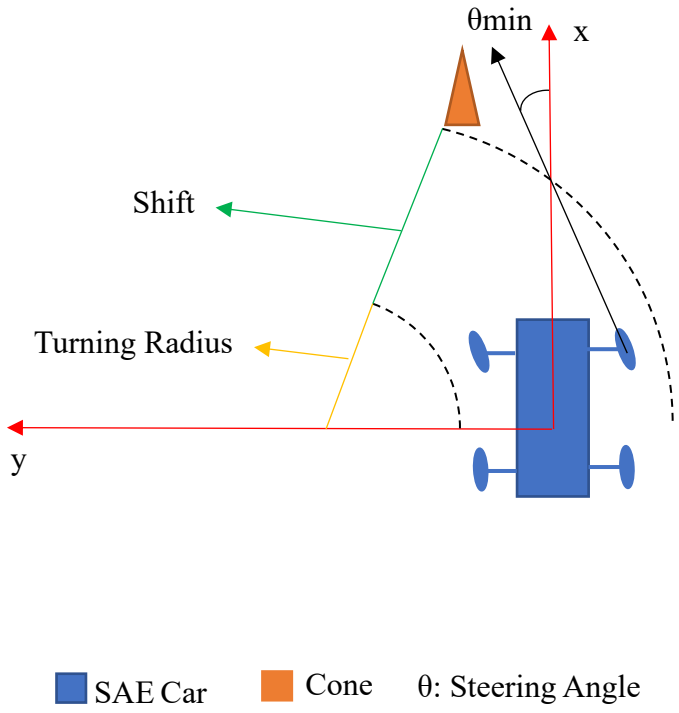

Figure 4: Passing through form the left of the cone

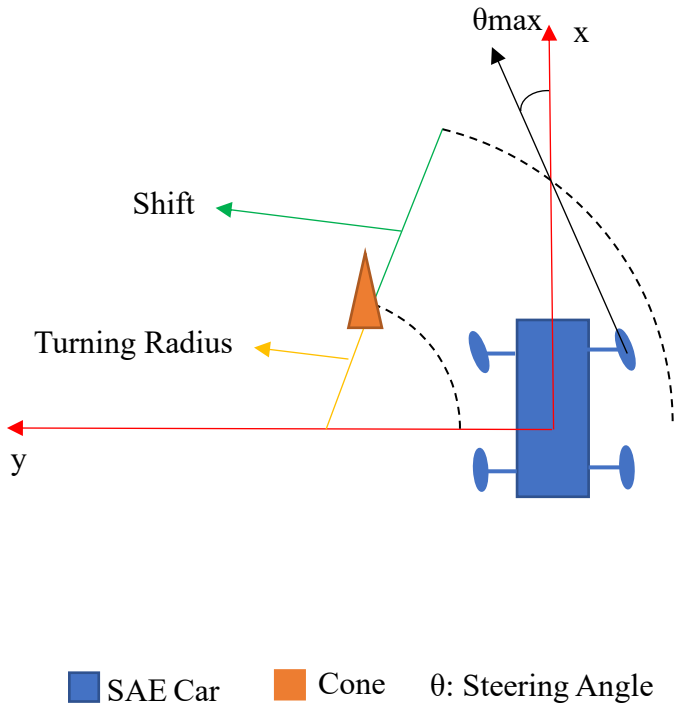

Figure 5: Passing through form the right of the cone

A maximum and a minimum steering angle to avoid collisions can be derived for each

evaluated cone. Thus, valid steering ranges can be formed by the combinations of the derived collision-free steering angles and the steering limits ($\theta_{lim} = \pm 25°$) of the SAE car. After evaluating every cone within the processing range, all valid steering ranges are determined. The theoretical result is shown in Figure 6, where θ1 is the steering angle of a cone, while θ2 is the steering angle of the other one.

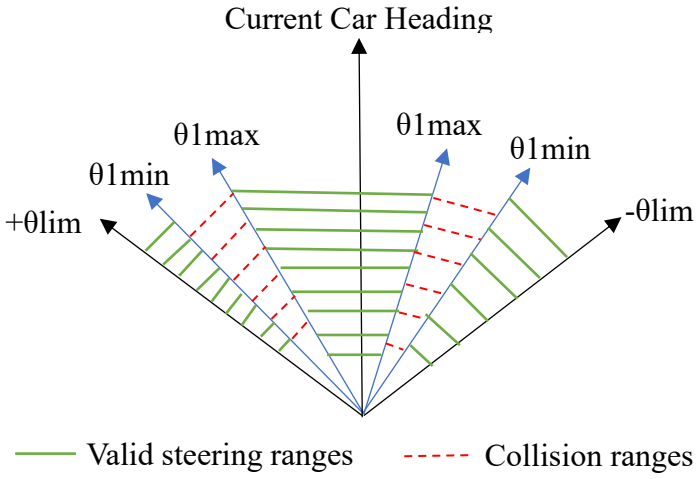

Figure 6: Theoretical result of collision evaluation

The desired steering angle is the mean value of the desired steering range.

3 Design Process

This chapter discusses the design process of modifying the old planner. As the major task of this project is to modify the existing local path planner, this chapter mainly focuses on the new features added to the old planner and explains the principles behind it. At the same time, the methodologies used to achieve each new function, as well as design tools, are specified.

3.1 Manoeuvre Projection

Manoeuvre projection is the most significant feature of the modified local path planner (hereinafter referred to as "the modified planner"). It enables the car to project a few following manoeuvres based on the obstacles detected and the velocity of the car, which is set to be a constant. The scanning range of the SICK LiDAR mounted on the front of the car is up to 50 m [8]. Namely, the car is able to pick up the cones within 50 m.

However, the old planner only makes use of the cones within the processing range, typically 3 m to 4 m, for path planning, because the collision evaluation process tends to be interfered with if too many cones are taken into account at a time. That means the old planner can only compute the optimum steering angle in respect of a very limited part of the track, which is not necessarily the optimum steering angle for the whole scanned area. Most of the collected data is wasted in the old planner, which makes the old planner “short-sighted”.

The manoeuvre projection function can make the best of the collected data and thus find the optimum steering angle of the current manoeuvre for the whole scanned area. First, the modified planner conducts the collision evaluation process, only considering the cones within the processing range and derive an “optimum steering angle” for the current manoeuvre (actual manoeuvre), as the old planner does. Then, it projects the position of the next manoeuvre according to the derived steering angle and its velocity. Next, it repeats the collision evaluation process and derives the “optimum steering angle” for the first projected manoeuvre. The projection process will iterate until it reaches the desired number of projected manoeuvres.

The outline of the manoeuvre projection function is shown in Algorithm 2.

Algorithm 2: Manoeuvre projection

```
input: cones information, processing range, total number of projections
output: desired steering angles for each manoeuvre
for (the index of the computed manoeuvre < total number of projections) do
    conduct coordinates update;
    conduct collision evaluation;
    conduct path evaluation;
    if (no valid steering found for the projected manoeuvre) then
        if (index = 0) then
            stop the car;
        else then
            conduct rollback function;
        end if
    end if
    index++;
end for
adopt the desired steering angle of manoeuvre[0] as the actual steering angle;
```

After the iteration, the subsequent manoeuvres will be projected depending on the number set by the user. At the same time, a primary and a spare steering angle will be stored for each projected manoeuvre. Although only the resultant steering angle for the current manoeuvre (manoeuvre[0]) will be deployed as the actual steering angle, the projection function, in cooperation with the rollback function, enables the car to select the desired steering angle based on the information of the whole scanned area.

3.2 Path Evaluation

The path evaluation strategy of the modified planner is different from the old planner. In the old planner, only the largest valid steering range or the largest one with smaller

angle change to the car orientation if there is more than one largest steering range is chosen as the desired steering range. However, in the modified planner, in addition to the same largest steering range as the old planner will choose, it will also consider the second largest steering range or the largest one with larger angle change to the car orientation compared to the primary one as the spare steering angle. The overview of the path evaluation function is depicted in Algorithm 3.

Algorithm 3: Path Evaluation

input: valid steering ranges, car orientation

output: primary steering angle, spare steering angle

if (no steering range left) **then**

    stop the car;

else then

    compare the sizes of all valid steering ranges;

    select the largest steering range as the primary one;

    select the second largest steering range as the spare one;

    if (there are more than one equally largest steering ranges) **then**

        select the one with a smaller change to the car orientation as the primary one;

        select the one with a larger change to the car orientation as the spare one;

    end if

    get the mean value of the primary steering range as the primary steering angle;

    get the mean value of the spare steering range as the spare steering angle;

end if

3.3 Rollback Function

The rollback function is also a very important part of the modified planner working with the manoeuvre projection function. It enables the SAE car to use the spare steering angle as its actual steering angle when a potential collision is found in a following

projected manoeuvre when it deploys the primary steering angle for the current (actual) manoeuvre.

To be more specific, in the projection process, if a collision is found in a projected manoeuvre, the rollback will be triggered. It will make the planner step back to the last projected manoeuvre and adopt the spare steering angle for that manoeuvre. Then, it will continue the projection process and project its next manoeuvre based on the spare steering angle. If there is no spare steering angle for the particular projected manoeuvre or the spare steering has already been deployed, it will step back iteratively until it reaches the manoeuvre with an unused spare steering angle. However, if it steps back to manoeuvre[0], the current (actual) manoeuvre, and it has no spare steering angle, which means there is no possibility to find a valid path, it will stop the car directly. The overview of the rollback function is shown in Algorithm 4.

Algorithm 4: Path Evaluation

```
input: index of manoeuvre, spare steering angle
output: index of manoeuvre
if (collision is found in manoeuvre[index]) then
    if (index = 0) then
        stop the car;
    else then
        For (index--; index > -2; index--) do
            if (index = -1) then
                stop the car;
            end if
            if (manoeuvre[index] has spare steering angle and it has not been used)
                adopt the spare steering angle for manoeuvre[index];
                break;
            end if
        end for
    end if
end if
```

3.4 Localisation of Projected Manoeuvres

It is essential for an autonomous driving vehicle driving in an unknown environment to know its position and orientation relative to its starting pose for keeping the direction and most importantly, the control of the vehicle [9]. As a result, the localisation process of projected manoeuvres is the foundation of the manoeuvre projection function.

Since the projected manoeuvres are imaginary manoeuvres, there is no sensor data indicating the positions and orientations of them. A localisation method called “dead reckoning” can be applied in this case. Dead reckoning is the process of deducing a robot’s

current position from its previous position, and extrapolating the next position according to the known speed, period of time and course [10].

The dead reckoning process for extrapolating projected car positions [11] is shown in Figure 7. The turning radius of the manoeuvre has been already specified in equation (1). The angular velocity of the manoeuvre can be determined by:

$$angular\ velocity = \frac{car\ speed}{turning\ radius} \tag{3}$$

The angle change θ is:

$$\theta = angular\ velocity \times time\ period \tag{4}$$

Hence, the local coordinates of the next position are:

$$x = turning\ radius \times (1 - \cos\theta) \tag{5}$$

and

$$y = turning\ radius \times \sin\theta \tag{6}$$

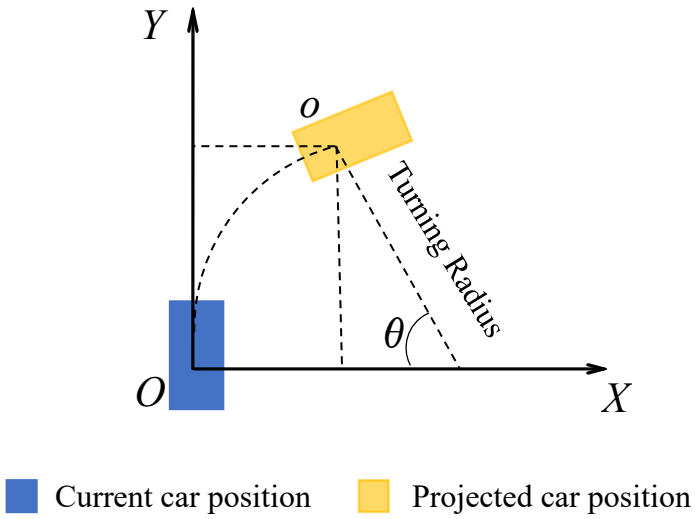

Figure 7: Dead reckoning

The algorithm used in the collision evaluation function assumes the car position is (0, 0). Thus, the goal of the localisation process of projected manoeuvres is to update the coordinates of the detected cones relative to the projected car positions. However, the

local coordinates of the cones cannot be determined directly. The localisation process comprises two parts, updating the global coordinates of the projected car position and updating the local coordinates of the cones.

3.4.1 Car Position Update

The first task of the localisation process is to update the global coordinates of projected car positions. The current (actual) car pose (including the position and orientation) is considered as the global coordinate system. The coordinates of the next projected car position relative to the current position can be derived according to the steering angle and the speed. Similarly, the coordinates of the following projected positions can be determined through iteration.

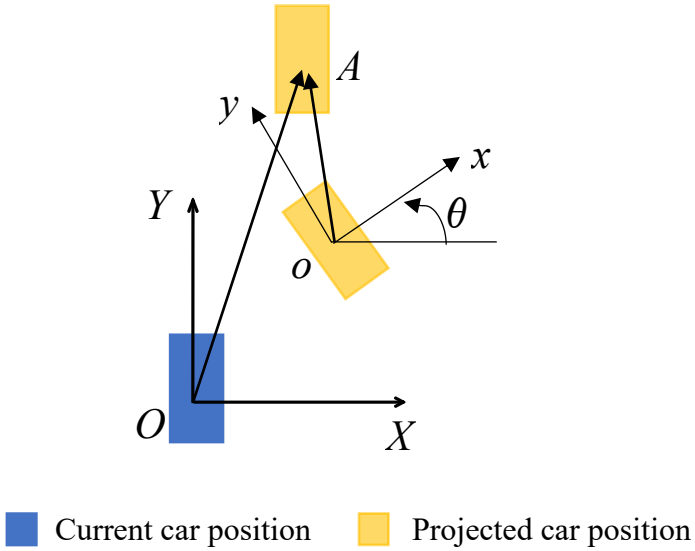

Figure 8: Coordinates transformation of projected car position

As is shown in Figure 8, O is the current car position acting as the global coordinate system, and o is the previous position of projected car position A, acting as the local coordinate system. Since the local coordinates of car position A are derived by equation (5) and (6), the global coordinates of car position A can be determined by:

$$\begin{bmatrix} X_A \\ Y_A \end{bmatrix} = \begin{bmatrix} X_o \\ Y_o \end{bmatrix} + \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x_A \\ y_A \end{bmatrix} \quad (7)$$

where (X_o, Y_o) is the global coordinates of the local origin.

3.4.2 Cone Position Update

After knowing the global coordinates of a particular projected car position, the coordinates of the detected cones relative to the car position can be figured out. The cone position update process still uses the same global coordinate system, the current (actual) car pose, as is shown in Figure 9, where A becomes the cone position and o is its corresponding car position.

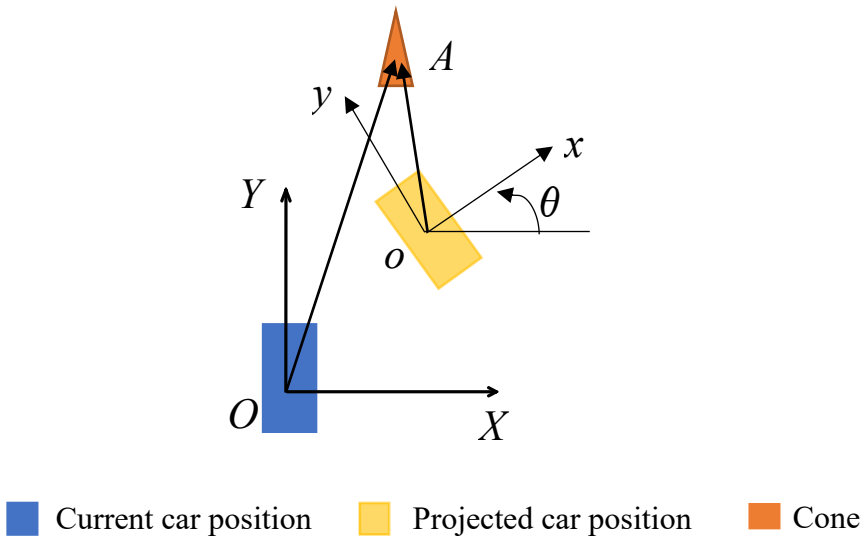

Figure 9: Coordinates transformation of cone position

The local coordinates of the detected cone can be determined by:

$$\begin{bmatrix} x_A \\ y_A \end{bmatrix} = \left(\begin{bmatrix} X_A \\ Y_A \end{bmatrix} - \begin{bmatrix} X_o \\ Y_o \end{bmatrix} \right) \times \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}^{-1} \quad (8)$$

3.5 Off-track Prevention

In order to meet the safety requirements of SAE autonomous vehicle competition [12], the SAE car is required to stop immediately when it gets out of the track. This can be achieved by either geofencing or adding a new stop rule to the path planner. To be on the safe side, both two means are applied to the SAE car. However, geofencing is not in the scope of the path planning topic, so only the off-track prevention function is discussed in this thesis.

The general structure of the off-track prevention function is demonstrated in Algorithm 5.

Algorithm 5: Off-track Prevention

input: cones information, processing range

output: stop signal

```
if (the cones locations are inside the processing range) then
    keep the cones;
else then
    exclude the cones;
end if
if (no cones left) then
    send stop signal;
end if
```

As a matter of fact, the off-track prevent can not only stop the SAE car but also help the car stay in the track, along with manoeuvre projection function. If the SAE is led out of the track by the primary steering angle in the projection process, the off-track prevent function will send a stop signal which triggers the rollback function to make the projection step back and choose another steering angle to perform until all the projected car positions are within the track.

3.6 Design Tools

The modified local path planner of the SAE car is composed in C++ language. It is just part of the autonomous driving system of the SAE car, so it will not work without other modules in the system. It needs to be run in an operating system under the coordination of a certain software framework. In addition, other tools are also applied to this project to improve the efficiency of the design process.

ROS

ROS (robot operating system) is an open-source software framework for robots. Its services include hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between different programs, and package management [13]. The current version of ROS applied to this project is Kinetic

Kame, which is compatible with Linux 16.0.4.

The modified planner is designed as a node in ROS. With the message-passing scheme, it can exchange information with other functions in the autonomous driving system of the SAE car, which are also written as nodes in ROS [14]. The system integration becomes simple since different nodes created by team members working on different topics can publish or subscribe to information on the same topics.

Rviz

Rviz is a visualisation toolkit combined with ROS. It can visualise the messages transmitted within ROS sorted by topics. Rviz can serve as a graphical user interface in the design process since so that there is no need to specially develop a user interface to visualise the information in the project. It is essential for the path planner to visualise the information because it would be extremely difficult to troubleshoot the planner if there is no graphical information provided.

Simulator

A hardware-in-the-loop autonomous driving simulation system developed by a previous year's student Brogle [15] was utilised in the design process to test the performance of the modified planner. It is constructed based on an open-source simulator, CARLA driving simulator, which supports various kinds of virtual sensors and environmental configuration [16]. In the meantime, the simulator was integrated with ROS, which allows the easy intercommunication between the simulator and the path planner.

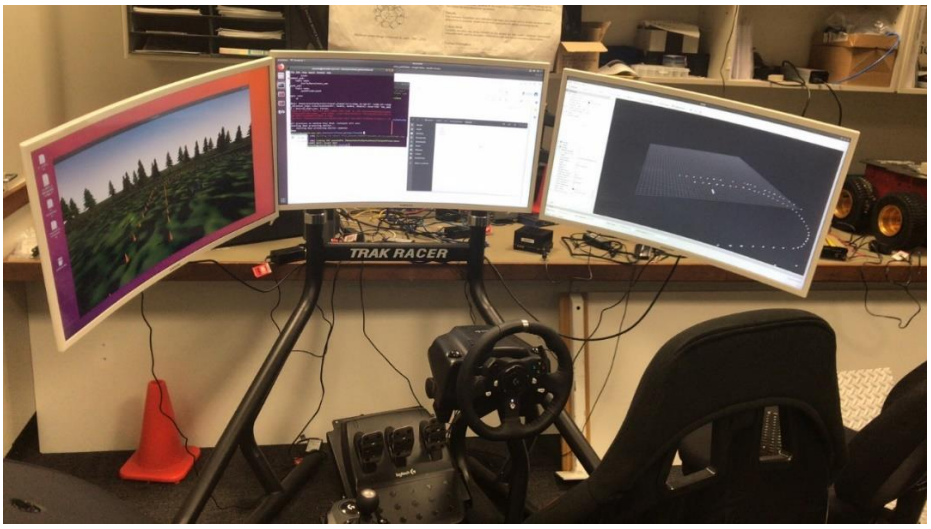

Figure 10: The simulator rig

Git

Git is an open-source distributed version control system for managing software projects. It can record the changes to all files in the project over time and roll the project back to a certain status. It is used for data backup and code tracking in this project.

4 Final Design

The modified planner is developed based on the old planner. As a result, it shares the core principle of path planning with the old planner, especially for the collision evaluation function discussed in Section 2.3. The main difference between the modified planner and the old planner is that there are a couple of new functions added to the modified planner to improve the autonomous driving performance of the SAE car.

The modified planner is run with a frequency of 10Hz. Namely, it receives the message needed from ROS, does path planning, and outputs the actual steering angle every 0.1 seconds. Thus, each steering angle output will come into effect for 0.1 seconds, which is considered to be the duration of a single manoeuvre.

When the path planning process of the modified planner starts, it will plan the path for the current (actual) manoeuvre in the first place. It will do the collision evaluation the same as the old planner. After that, it will take a different path evaluation strategy to store both the primary and spare steering angles. Then it continues to project the path for the following manoeuvres. It will update the coordinates of the car position and the cone positions. Next, it will do the collision evaluation as well as the path evaluation processes to get the primary and spare steering angles for the projected manoeuvre. It will repeat the steps above until it reaches the required number of projections. Otherwise, when a collision is predicted in a certain projected manoeuvre, the rollback function will be triggered, and the projection process will be rolled back to a manoeuvre with a valid spare steering angle. Then, it will repeat the path planning processes mentioned above again.

The general logic of the whole modified planner is shown in Algorithm 6.

Algorithm 2: Manoeuvre projection

```
input: cones information, processing range, total number of projections
output: primary steering angles for each manoeuvre
for (the index of the computed manoeuvre < total number of projections) do
    if (index != 0) then
        conduct localisation process;
    end if
    conduct collision evaluation;
    conduct path evaluation;
    conduct off-track prevention;
    if (stop signal sent) then
        if (index = 0) then
            stop the car;
        else then
            conduct rollback function;
        end if
    end if
    index++;
end for
adopt the desired steering angle of manoeuvre[0] as the actual steering angle;
```

The theoretical result of the modified planner is shown in Figure 11. With such track layout, the SAE car will project its manoeuvres through path 1 by default because the steering range of path 1 is the largest. However, a stop signal will be sent at the end of path 1, which triggers the rollback function and roll the projection back to the manoeuvre at the fork. Then the car will use the spare steering, which leads the car to path 2. Similarly, a stop signal will be sent at the end of path 3, and the rollback function will be triggered. This time, the projection will be rolled back to the second manoeuvre,

since the spare steering angle of the manoeuvre has been already used once. Finally, the projected manoeuvre will go through path 3.

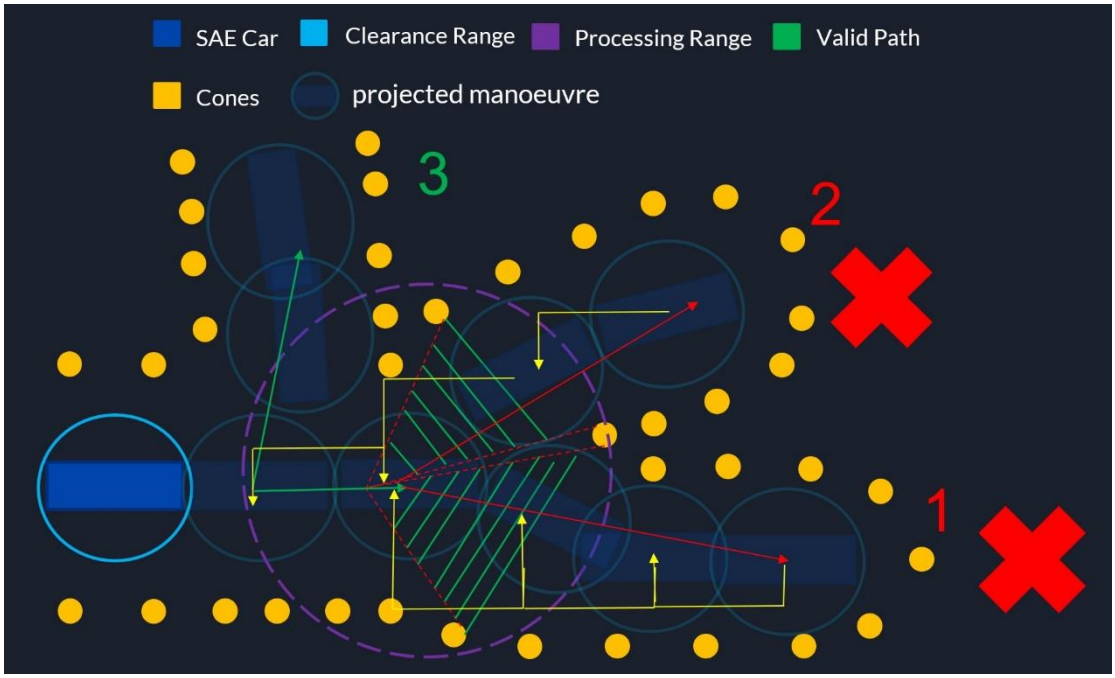

Figure 11: The theoretical result of the modified planner

5 Test and Result

The goal of the modification to the existing local planner is to improve its performance, adaptability more specifically, in cone tracks with complex layout. However, adaptability is an abstract concept that is difficult to be quantified. Therefore, the main approach of evaluating the success of the design is to conduct some qualitative tests for both the old and the modified planner and compare the results of them.

5.1 Test by Simulation

Due to the bad weather and hardware problems happening to the SAE car, only a limited number of field tests were conducted during the design process. On the other hand, simulation offers more flexible options of the layouts of tracks. As a result, the test to evaluate the performance of the modified planner was conducted in the simulator detailed in Section 3.6.

The map used in the test is a cone track on grass, which is exactly the same environment

as the usual driving tests conducted on field. There are several road junctions with dead ends to confuse the planner in the track configuration, as is illustrated in Figure 12.

Figure 12: Track configuration of the simulation test

The virtual SAE car in the simulator was deployed on the map, run with the old planner and the modified planner to display the difference.

5.2 Results from Simulation

5.2.1 Road Junction 1

Road junction 1 has two paths for the SAE car to drive through. One is a dead end and the other one is through to junction 2. The widths of the two paths are roughly the same while the angle change of the dead end to the initial car orientation is smaller than the one of the other path. Therefore, the SAE car should be led to the dead end by the original path planning logic. The test result of the old planner is shown in Figure 13, where the SAE car went into the dead end as expected.

Figure 13: Test result of the old planner at junction 1

The test result of the modified planner is shown in Figure 14. When the SAE car went to the position shown on the left-hand side, it detected the dead end and triggered the rollback function. As a result, it applied the spare steering angle to this case, as is shown in the visualised image by Rviz on the right-hand side. Although the modified did make a correct decision in this situation, the SAE car failed to complete the steering and stopped in the middle of the path due to understeer. The underlying causes are demonstrated in Discussion Section.

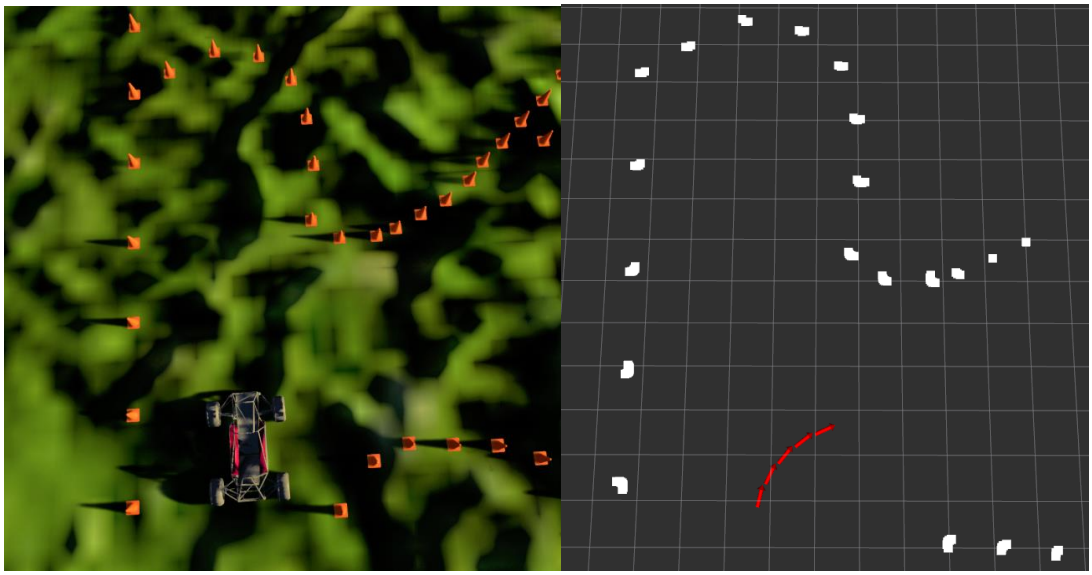

Figure 14: Test result of the modified planner at junction 1

5.2.2 Road Junction 2

Similar to road junction 1, junction 2 also has two paths to drive through. One is a dead end and the other one is through to junction 3. The width of the dead end is larger than the through path. Therefore, the SAE car should be led to the dead end by the original path planning logic. The test result of the old planner is shown in Figure 15, where the SAE car went into the dead end as expected.

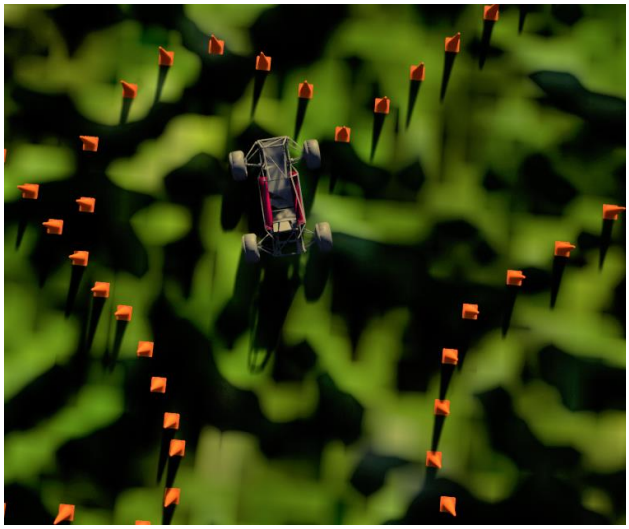

Figure 15: Test result of the old planner at junction 2

The test result of the modified planner is shown in Figure 16. When the SAE car went to the position shown on the left-hand side, it detected the dead end and triggered the rollback function which led it to choose the spare steering angle. The visualised path is shown on the right-hand side. This time, it succeeded in passing through the junction.

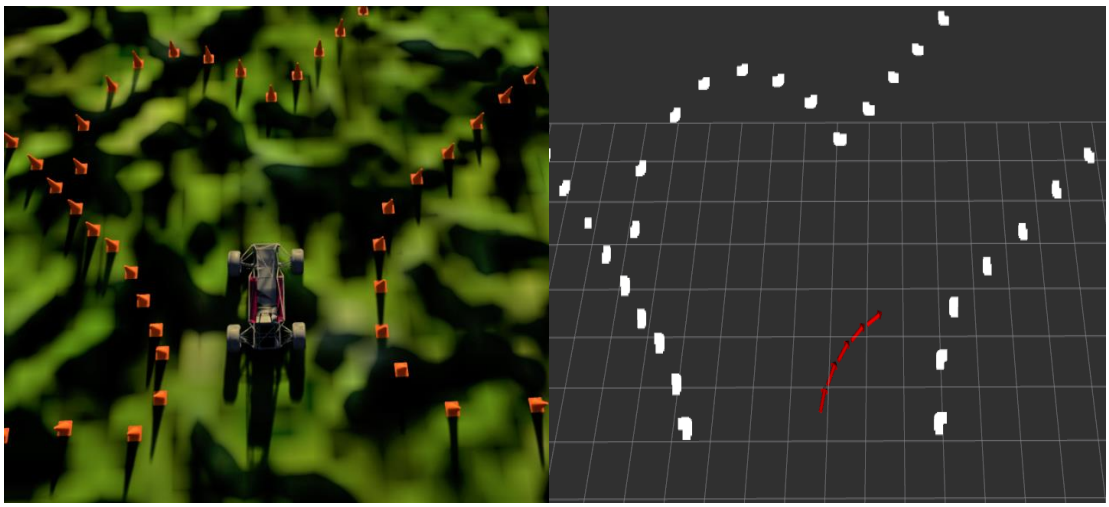

Figure 16: Test result of the modified planner at junction 2

5.2.3 Road Junction 3

Junction 3 has a straight path to drive through and a closed-end protrusion. The width of the protrusion is larger than the through path which should lead the SAE car to the dead end with the original path planning logic. The test result of the old planner is shown in Figure 17, where the SAE car ran into the protrusion as expected.

Figure 17: Test result of the old planner at junction 3

The test result of the modified planner is shown in Figure 18. When the SAE car was at the position shown on the left-hand side, it detected the dead end and triggered the rollback function which made it avoid the protrusion. The visualised path is shown on the right-hand side. Eventually, it kept going straight ahead in the track and passed through.

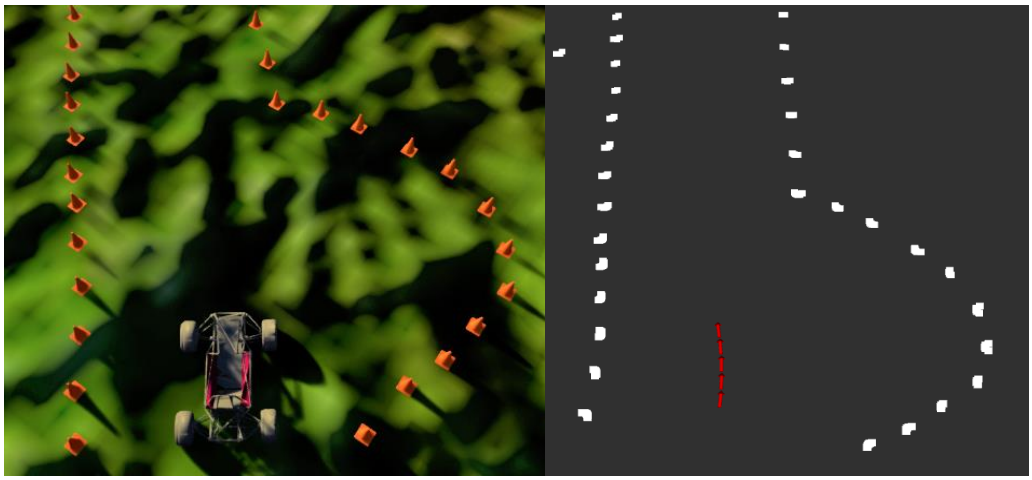

Figure 18: Test result of the modified planner at junction 3

6 Discussion

It is known from the test result that the modified planner is imperfect due to the fact that it could not completely pass the map for testing. Two major facts are limiting its performance, the kinematic model of the SAE car and the dead reckoning localisation for manoeuvre projection.

6.1 Limitation of the Kinematic Model

As is mentioned in Section 5.2.1, the virtual SAE car failed to pass through the road junction 1 though the planner made the correct decision. The root cause of this error is the disparity between the kinematic models in the planner and the simulator. In other words, they are using different equations to calculate the motion of the SAE car. Therefore, the result of path planning published by the planner is not consistent with the motional attributes of the virtual SAE car in the simulator. This is a major problem of the simulation process since all the path planning processes in the planner are based on the kinematic model.

The same problem happens to the physical SAE car. The kinematic model in the planner is not 100% the same as the actual motional attributes of the SAE car. In order to have more precise path planning and simulation, the kinematic models of the planner and the simulator should be in accordance with the actual SAE car. Certain motion studies on the SAE car and measurements are needed in the future to form an accurate kinematic model.

6.2 Limitation of the Localisation Method

As introduced in Section 3.4, the localisation method utilised for the manoeuvre projection function is based on dead reckoning. The error of the projected car position, caused by the inaccuracy of the kinematic model, will accumulate over the projection steps, which is the nature of dead reckoning. This makes the projected positions different from the actual car positions when they take the same steering angle, and it will be aggravated by increasing projection steps. However, dead reckoning is the only way to localise the projected car position because they do not exist in reality and no information from sensors will be provided in real time. Therefore, the error of the

localisation of projected car position can only be mitigated by improving the kinematic model of the SAE car.

7 Future Work

The autonomous driving capacity of the SAE car is still at a beginning level. Therefore, there are a variety of new features can be added to the SAE car to refine its autonomous driving ability.

7.1 Improvement of the Current Kinematic Model

As is stressed in the former sections, the current kinematic model of the SAE car in the modified planner is not coincident with the one in the simulator and the physical SAE car, which compromises the accuracy of the path planning process. The skid effect [17] on the wheels, the inertia [18] of the SAE car can also be taken into account to construct a new kinematic model in the future. A precise kinematic can guarantee the quality of path planning.

7.2 Sensor Fusion

At present, the modified planner works only depending on the laser scan information provided by the SICK LiDAR mounted at the front of the SAE and the pose information from odometry sensors. However, there are diverse sensors on the car, which have not been used for path planning, including cameras, IMU, GPS, IBEO Lux 4-layer Automotive LiDAR, and Velodyne 360-degree LiDAR [19]. The signals of these sensors can be fused together and utilised to provide more accurate input for path planning. Sensor fusion will be an essential process for making use of these available sensors.

7.3 AI Path Planner

Applying AI (Artificial Intelligence) model to self-driving vehicles might be the future of the autonomous driving industry since AI and deep learning enable vehicles to see, think, learn and navigate an unlimited range of driving scenarios [20]. AI can be introduced to the path planning topic of this project as well. It would be a powerful path planning tool especially for dynamics driving scenarios that require the SAE car to have

interaction with other moving objects and changing environments. Additionally, the ongonging training process of the AI planner while driving can constantly enhance the autonomous driving capacity of the car as the training sample is expanding. AI methods would be a feasible solution to the path planner of the SAE car.

8 Conclusion

The modified local path planner is proved to have better performance than the old planner with regard to the adaptability of driving on a cone track. The autonomous driving capacity of the SAE car was strengthened by adding new features, such as the manoeuvre projection function and the rollback function. Although there are some imperfections in the modified planner, due to the fact that the modified planner shares the same core principle of path planning, the collision evaluation algorithm, and the same kinematic model with the old planner, the design can be considered successful, since the modified, since all the new functions added in work well in an expected way. Once the modified planner has a more accurate kinematic model in the future, its performance will be significantly improved and more reliable.

References

[1] Tesla, “Autopilot,” [Online]. Available: https://www.tesla.com/en_AU/autopilot. [Accessed 10 Oct 2019].

[2] The Drive Media, “The Way We Talk About Autonomy Is a Lie, and That's Dangerous,” [Online]. Available: https://www.thedrive.com/opinion/7324/the-way-we-talk-about-autonomy-is-a-lie-and-thats-dangerous. [Accessed 8 Oct 2019].

[3] The REV Project, “Self Driving Formula SAE Race Car,” The University of Western Australia, 2019. [Online]. Available: http://therevproject.com/vehicles/sae-autonomous.php. [Accessed 3 Oct 2019].

[4] “Jetson AGX Xavier,” NVIDIA Corporation, 2019. [Online]. Available: https://developer.nvidia.com/embedded/jetson-agx-xavier. [Accessed 10 Oct 2019].

[5] W. L. W. Lai, “Autonomous Driving with Dynamic Path Planning,” 2018.

[6] R. C. Podolski, “Implementation of an Iterative,” 2017.

[7] B. Paden, M. Cap, Y. Sze Zheng, D. Yershov and E. Frazzoli, “A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles,” *IEEE Transactions on Intelligent Vehicles,* vol. 1, no. 1, pp. 33-55, 2016.

[8] SICK AG, “LMS111-10100 | Detection and ranging solutions | SICK.,” [Online]. Available: https://www.sick.com/au/en/detection-and-ranging-solutions/2d-lidar-sensors/lms1xx/lms111-10100/p/p109842. [Accessed 15 Oct 2019].

[9] T. Bräunl, “Evolution of Walking Gaits,” in *Embedded Robotics*, Berlin, Springer, 2006, p. 345–346.

[10] C.-L. Chen and Y.-D. Guo, “Mobile robot localization by dead reckoning,” in *IEEE*, 2013.

[11] L. Marín, M. Vallés Miquel, Á. Soriano Vigueras, Á. Valera Fernández and P. Albertos Pérez, “Event based localization in Ackermann steering limited resource mobile robots,” *IEEE/ASME Transactions on Mechatronics,* vol. 19, no. 4, pp. 1171 - 1182, Aug 2014.

[12] SAE Australia, “FORMULA SAE-A,” [Online]. Available: http://www.saea.com.au/formula_saea. [Accessed 20 Oct 2019].

[13] Open Source Robotics Foundation,, “ROS Introdcution,” [Online]. Available: http://wiki.ros.org/ROS/Introduction. [Accessed 15 Oct 2019].

[14] K. L. Lim, T. Drage, C. Zhang, C. Brogle, W. L. W. Lai, T. Kelliher, A.-Z. Manuchekhr and T. Braunl, “Evolution of a Reliable and Extensible High-Level Control System for an Autonomous Car,” 2018.

[15] C. Brogle, “Software Architecture and,” 2018.

[16] C. Team, “Carla - Open-source simulator for autonomous driving research,” 2019. [Online]. Available: http://carla.org/. [Accessed 25 Oct 2019].

[17] P. Tsiotras and E. Velenis, “Minimum Time vs Maximum Exit Velocity Path Optimization During Cornering,” *IEEE International Symposium on Industrial,* pp. 355-360, 2005.

[18] A. Rucco, G. Notarstefano and J. Hauser, “Computing minimum lap-time trajectories for a single-track car with load transfer,” in *Proceedings of the IEEE Conference on Decision and Control*, 2012.

[19] Velodyne Lidar, “Puck,” [Online]. Available: https://velodynelidar.com/vlp-16.html. [Accessed 15 Oct 2019].

[20] NVIDIA Corporation, “Driving Innovation,” [Online]. Available: https://www.nvidia.com/en-au/self-driving-cars/. [Accessed 25 Oct 2019].