Motion Planning and Safety for Autonomous Driving

by

Ryan De Iaco

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2019

© Ryan De Iaco 2019

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis discusses two different problems in motion planning for autonomous driving. The first is the problem of optimizing a lattice planner control set for any particular autonomous driving task, with the goal of reducing planning time for that task. The driving task is encoded in the form of a dataset of trajectories executed while performing said task. In addition to improving planning time, the optimized control set should capture the driving style of the dataset. In this sense, the control set is learned from the data and is tailored to a particular task. To determine the value of control actions to add to the control set, a modified version of the Fréchet distance is used to score how useful control actions are for generating paths similar to those in the dataset. This method is then compared to the state of the art lattice planner control set optimization technique in terms of planning runtime for the learned task.

The second problem is the task of extending the Responsibility-Sensitive Safety (RSS) framework by introducing swerve manoeuvres in addition to the nominal braking manoeuvres present in the framework. This includes comparing the clearance distances required by a swerve to the braking distances in the original framework. This comparison shows that swerve manoeuvres require less distance gap in order to reach safety from a braking agent in front of the autonomous vehicle at higher speeds. For more realistic swerve manoeuvres, the kinematic bicycle model is used rather than the 2-D double integrator model considered in RSS. An upper bound is then computed on the required clearance distance for a swerve manoeuvre that satisfies bicycle kinematics. A longitudinal safe following distance is then derived that is provably safe, and is shown to be lower than the following distance required by RSS at higher speeds. The use of the kinematic bicycle model is then validated by computing swerve manoeuvres with a dynamic single-track car model and Pacejka tire model, and comparing the single-track swerves to the bicycle swerves.

Acknowledgements

I would like to thank Professors Stephen Smith and Krzysztof Czarnecki for their time and valuable advice throughout my degree.

I'd also like to thank my lab mates in both the Autonomy Lab and the WISE Lab for their valuable discussions and help along the way.

Dedication

To my parents, Roger and Rosa, for their love and support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the fundamental tasks of autonomous driving is safe motion planning, the task of deciding where the car needs to go, while avoiding obstacles, obeying traffic rules, and respecting the fundamental limits of what the vehicle can do. One factor that makes safe motion planning difficult is the speed with which the situation around a car can evolve with time. To compensate, autonomous driving motion planners often need to make decisions quickly, so as to respond to ever-changing stimuli in the environment. In addition, autonomous vehicles need to be proactive when it comes to safety, in order to be robust to a variety of different potential actions from other agents in the environment.

This thesis analyzes and discusses two problems in the domain of safe motion planning for autonomous driving. The first aims to improve path planning efficiency by leveraging human data to specialize a particular planner to a particular driving task. These driving tasks come from a wide variety of problems encountered in motion planning for autonomous driving, ranging from parking lot manoeuvres to unprotected left turns. By making the planner specialized to a particular task, the planner is able to solve motion planning problems associated with that task more quickly. The next problem involves determining when it is safe for the autonomous vehicle to perform swerve manoeuvres in specific situations. This is done by computing the initial distance required between the swerving autonomous vehicle and the vehicle in front of it. In addition, this initial distance is compared to the distance required by a brake manoeuvre, and it is shown that swerving allows for vehicles to use the roadway more efficiently by reducing the gap between vehicles. To demonstrate safety, this thesis shows that these manoeuvres are robust to a range of possible agent actions, while not being so conservative as to over-constrain the autonomous vehicle.

1.1 Learning a Lattice Planner Control Set for Autonomous Driving

A crucial portion of autonomous driving is motion planning. It is important for autonomous vehicles to be able to quickly generate a collision-free, kinematically feasible path towards their goal that minimizes the total cost of the path. An algorithm commonly used in path planning is the lattice planner[81]. The goal of this work is to leverage data to improve lattice planning for a particular autonomous driving task, while also capturing aspects of the driving style present in the dataset. In this work, the driving task is assumed to be implicitly encoded in the given dataset.

As will be discussed in further detail in section 3.1, the lattice planner is a graph-based approach to the path planning problem that reduces the search space into a uniform discretization of vertices corresponding to positions and headings. Each vertex in the discretization is connected to other points by kinematically feasible motion primitives, known as control actions[84]. The lattice planner thus reduces the path planning problem into a graph-search problem, which can be solved with A* or any other appropriate graph search algorithm[83, 68, 41, 121]. An example of a lattice graph is given in 1.1.

The aim of this portion of the thesis is to develop a method for learning a lattice planner control set from a dataset of trajectories representative of a particular driving task. In doing so, the learned control set should allow for fast planning for that particular task, and in addition, the learned control set should capture the driving style present in the dataset. To do this, a densely populated lattice control set is refined into a set of control actions that is optimized for the task implicitly represented by the dataset. Crafting control sets in this manner will allow for specific control sets for specific driving tasks. For example, one could generate a control set for roundabouts, parking lots, and u-turns. This would allow for a speedup in planning time as compared to having one large general lattice for each task, as the branching factor of each vertex in the lattice graph would be smaller.

1.2 Safe Swerve Manoeuvres for Autonomous Driving

The main bottleneck for the public acceptance and ubiquity of autonomous driving is the current lack of safety guarantees. There are three main ways to establish the safety of an autonomous vehicle. The first involves measuring crash statistics over a large number of

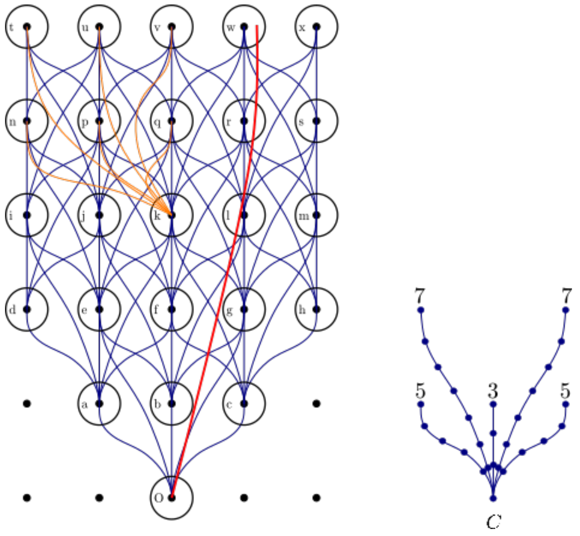


Figure 1.1: An example of a lattice graph which will be used in the first portion of this thesis, with labelled vertices. The control set is given by C, and each control action is labelled by the number of path points (excluding the origin point). An example control set from a different initial heading is given in orange at vertex k. The dataset path to search the lattice for, P_d, is given in red.

autonomously driven kilometres and comparing them to the equivalent human rates for each category of collision severity. However, particularly with severe collisions, the number of kilometres required to establish a statistically significant collision rate renders this method impractical for establishing safety [55].

An alternative method for determining the safety of a system is through scenario-based verification [104]. This method uses a set of scenarios that validate the vehicle's behaviour across a representative set of situations. The goal is for the set of scenarios to capture most of the required driving behaviour necessary for safe driving. However, it is difficult to construct such a set of scenarios that captures all of the challenging conditions faced by an autonomous vehicle [1].

A third approach for verifying the safety of a system is formally proving the behaviour of a vehicle is safe [3, 4, 96, 62]. In order to compute useful safety bounds, these works often include simplifying assumptions. The difficulty with this method lies in selecting reasonable assumptions to make. The stronger the assumptions made, the easier to prove the system is safe. However, if the assumptions are too strong, they may not hold in general driving scenarios. An additional challenge with this method is that to prove safety, the driving behaviour may need to be conservative, or highly restrictive.

This portion of the thesis addresses the latter issue, especially as it pertains to the Responsibility-Sensitive Safety (RSS) framework [96]. Fundamental to the RSS framework is its assumption of responsibility, and that agents have a duty of care to one another. The assumption of responsible behaviour allows for the autonomous vehicle to make meaningful progress in the driving task. Under other frameworks that assume adversarial agents, the autonomous vehicle often exhibits over-conservative behaviour that impedes progress. By assuming responsible behaviour, one can compute safe distance gaps between vehicles based on their speeds, reaction times, and maximum accelerations, assuming both agents exhibit particle model kinematics. As long as these distance gaps are maintained, no collisions can occur. The work in this thesis extends this analysis to prove safety when using swerves feasible for the kinematic bicycle model. The kinematic bicycle model is discussed in detail in section 3.3.

1.3 Thesis Contribution

This thesis analyzes two problems in the domain of motion planning for autonomous driving. The first part of the thesis focuses on the problem of optimizing a lattice planner control set according to a particular task, which is encoded as a set of trajectories executed as part of that task. We use a modified version of the Fréchet distance in combination with a graph search to determine which control actions are most useful for executing the trajectories present in the task dataset. The control actions are selected from a densely populated control set, and the optimized control set is built bottom-up [50]. We then validate this method by comparing it to the state of the art lattice planner control set computation technique [81] with respect to computation time, as well as matching the driving style of the dataset. Driving style is measured by comparing the curvature of the planned paths to those in the dataset. The datasets used in the experiments are both real human-driven trajectories in a roundabout, as well as synthetic turn manoeuvres.

The second part of the thesis explores extending the Responsibility-Sensitive Safety (RSS) by using swerve manoeuvres feasible for the bicycle model as a valid response in addition to braking. There are two main contributions in this chapter. The first is a novel extension of the RSS framework to include kinematically feasible swerve manoeuvres, according to the kinematic bicycle model. To do so, an initial distance between the swerving agent and the agent in front of it is computed that is sufficient to guarantee safety. This initial distance is then used to compute the gap required to a lead vehicle when performing lane change or lane shift manoeuvres. The second contribution is a validation of the use of the kinematic bicycle model by comparing the bicycle swerve manoeuvres to manoeuvres generated under a dynamic single-track model. This dynamic single-track model includes a Pacejka tire model to account for road surface traction. This thesis shows that the kinematic model, when lateral acceleration is constrained, can accurately estimate the longitudinal distance required to perform swerve manoeuvres using the dynamic single-track model.

1.4 Organization

This thesis is organized as follows. A review of the literature on motion planning for autonomous driving, RSS, and swerve manoeuvres is presented in chapter 2. Next, chapter 3 discusses the necessary mathematical preliminaries for lattice planning, spiral path planning, vehicle kinodynamic models, and RSS. The contents of chapter 4 discuss how to optimize a lattice planner control set. Next, chapter 5 analyzes swerve manoeuvres and proves their safety. Finally, the conclusions of this research and potential future work is discussed in chapter 6.

Chapter 2

Literature Review

2.1 Motion Planning for Autonomous Driving

The task of motion planning for autonomous driving has been studied extensively [76]. In the literature, there are three families of planners that are commonly used.

2.1.1 Variational Optimization Planners

The first group is those that use calculus of variations to optimize a trajectory function of time according to a cost functional. Alternatively, they may optimize a path function of arc length instead. If π is the trajectory function to optimize, J is the cost functional, $\mathbf{x}_{\text{init}}$ is the initial state, X_{goal} is the set of goal states, and f and g capture the kinodynamic constraints of the system, then the general form of the problem these planners try to solve is

$$\begin{aligned} \underset{\pi}{\text{argmin}} \quad & J(\pi) \\ \text{s.t.} \quad & \pi(0) = \mathbf{x}_{\text{init}} \\ & \pi(T) \in X_{\text{goal}} \\ & f(\pi(t), \pi'(t), ...) = 0, \forall t \in [0, T] \\ & g(\pi(t), \pi'(t), ...) \leq 0, \forall t \in [0, T]. \end{aligned}$$

These problems are usually projected into a lower dimensional vector space and solved as non-linear programs [76, 22, 21]. These types of planners have been successfully applied to autonomous driving [105, 118, 106], and more generally to mobile robotics as well. In particular, the CHOMP [90, 122] and TrajOpt [95] are two variational methods that are commonly used.

2.1.2 Sampling-based Planners

The second group of planners are the sampling-based planners, which sample the configuration space of the vehicle to generate kinodynamically feasible paths to a goal set. In general, these methods incrementally build up an increasingly fine discretization of the configuration space until the goal is reached [76]. RRT* and PRM* are examples of such algorithms [56]. These algorithms have been used with great success in both urban and off-road environments, and are robust to unstructured or previously untested scenarios [59].

2.1.3 Lattice Planners

The final group of planners, and the focus of the first problem discussed in this thesis, are the lattice planners. At a high level, a lattice planner discretizes the configuration space with the goal of decomposing the motion planning problem into a tractable graph search problem.

Broadly speaking, there are two varieties of lattice planner used in autonomous driving. The first is the standard lattice planner [83, 81], in which a database of manoeuvres is computed offline. The vertices of the graph are uniform samples of the configuration space, and the edges are formed by the manoeuvres in the database. The advantage of these types of lattice planners is that most of the planning and collision checking process is pre-computed offline [82], which reduces online planning to graph search. The disadvantage is that these planners do not exploit the on-road structure inherent to motion planning in autonomous driving. These planners then require a large database of manoeuvres, in which only a small fraction of manoeuvres are relevant at a given time [65]. This type of planner excels in semi-structured environments where complex manoeuvres are required, and as a result, a large variety of manoeuvres within the database will be used. Examples of such environments include culdesacs and parking lots [105].

The second variety of lattice planners are the conformal lattice planners. The fundamental difference between these lattice planners and the standard lattice planners is that the configuration space is sampled relative to the road the car is travelling on. In general, this means that the configuration states are sampled along the Frenet frame of the road, instead of sampled uniformly in configuration space [68, 42, 112, 40]. The advantage of these lattice planners is that since the configurations are relative to the road, the planner needs to consider fewer manoeuvres to plan a feasible path. However, since the Frenet frame of the road varies between each planning cycle, a conformal lattice planner cannot pre-compute the manoeuvres that make up the edges of its search graph. Instead, they must be generated through optimization methods (such as those in Section 2.1.1). This can be computationally expensive, and limits the breadth and depth of the search graph that can be considered.

2.2 Data-driven Motion Planning

The first portion of this thesis focuses on using datasets of paths to inform the construction of lattice planner control sets. However, previous research into data-driven motion planning has often focused on learning search heuristics or policies for the motion planner rather than learning the underlying structure of the planner itself. Ichter et al. developed a method for learning a sampling distribution for RRT* motion planning[51]. Imitation learning can also be used to learn a search heuristic based on previously planned optimal paths[18, 8]. Paden et al. have developed a method for optimizing search heuristics for a given kinodynamic planning problem[77]. Xu et al. used reinforcement learning to learn a control policy for quadcopters by training on MPC outputs[116].

For work involving lattice planner control set optimization, Pivtoraiko et al. have developed a D*-like (DL) algorithm for finding a subset of a lattice control set that spans the same reachability of the original control set, but does so within a multiplicative factor of each original control action's arc length[82]. This algorithm does not rely on data, but instead relies on the structure of the original control set to find redundancy. In contrast, the method outlined in this thesis attempts to leverage data for a particular application to optimize the control set. This thesis uses the DL algorithm as the state-of-the-art comparison for the quality of the presented learning algorithm.

Optimizing a lattice planner control set requires a measure of similarity between two paths. This has been discussed in the field of path clustering [115], where measures such as the pointwise Euclidean distance, Hausdorff distance[14], the Longest Common Sub-

Sequence, and the Fréchet distance[26] are commonly used.

The work most closely related to the process of matching a specific path in a graph is the map-matching problem[107, 13]. The problem entails finding a path in a planar graph embedded in Euclidean space that best matches a given polygonal curve according to the Fréchet distance. However, unlike the work in this thesis, the map-matching algorithm requires the full graph to be defined beforehand, and cannot be used if the graph is implicitly defined in terms of the lattice control set. Another similar problem is that of following a path in the workspace for a redundant manipulator[74, 47].

When using data to inform the construction of a lattice control set, it would be beneficial for the learned control set to capture the driving style present in the data. In terms of driving style, Macadam gives a broad overview of the driving task[67]. This thesis focuses on the properties of paths and not trajectories. For the driving style of a given path, one of the most intuitive indicators is the vehicle's steering function, which under the commonly used bicycle model[86], is directly related to path curvature. As such, curvature serves as a natural measure for comparing the driving styles of different paths. A path with points of high curvature corresponds to a a more aggressive steering function, and vice versa.

2.3 Responsibility-Sensitive Safety (RSS)

While there are many approaches to formally defining and proving safety for autonomous driving, this thesis focuses on extending the RSS framework [96]. At the core of this framework is the required distance that must be maintained between the autonomous vehicle and the agents adjacent to it at all times. Under the assumptions of the framework, so long as the appropriate distance gap is maintained, no collisions can occur. This required distance gap is computed such that the autonomous vehicle can safely and comfortably brake in response to another agent in front of it performing a hard brake. In RSS, this distance gap takes the limits of comfortable and responsible deceleration as well as the reaction delay of the autonomous vehicle into account.

Fundamental to the RSS framework is its assumption of responsibility, and that agents have a duty of care to one another. This assumption allows for the autonomous vehicle to make meaningful progress in the driving task. Under other frameworks that assume adversarial agents, the autonomous vehicle often exhibits over-conservative behaviour. This is especially true with reachability methods, which can cause the autonomous vehicle to be overly cautious or evasive when interacting with other agents [4, 62].

2.4 Swerve Manoeuvres

Previous work on emergency manoeuvres for autonomous driving have often focused on feasible swerve manoeuvres according to various kinodynamic models [94]. In particular, many of these papers have assumed some variant of the bicycle model [100, 98, 97, 24] and performed optimization to generate optimal swerve manoeuvres. However, under these models the optimal solution is not generated through a closed form solution, which makes formally proving safety challenging.

Other work has instead simplified the vehicle model to a point mass or particle model [99, 52, 78] in order to yield closed form, optimal solutions. However, this comes at the cost of the non-holonomic constraint present in the bicycle model, which can result in manoeuvres that would be unrealistic for a car to execute. An important part of the second problem discussed in this thesis is to generate closed form, feasible solutions to swerve manoeuvre boundary conditions, while still preserving the kinematic constraints that allow the manoeuvre to be executable by a real vehicle.

To validate the use of the kinematic bicycle model when proving safety in this thesis, the bicycle model is compared to a dynamic vehicle model that represents the limits of what a real vehicle can truly execute. To do this, a dynamic single track model [37] with a Pacejka tire model [75] is used.

Chapter 3

Preliminaries

This chapter enumerates some of the mathematical definitions and concepts used in this thesis. Section 3.1 defines the terms used in lattice planning [83], which will be of use in chapter 4. Section 3.2 gives an overview of using polynomial spirals for path planning, which will be used for computing control actions for lattice planner control sets in chapter 4. Section 3.3 introduces some commonly used kinodynamic vehicle models, and Section 3.4 discusses some of the background on RSS, and the definition of safety used in this thesis. Both of these two sections will be used in chapter 5.

3.1 Lattice Planning

A graph G is composed of two sets, a set of vertices V and a set of edges E. Each $v \in V$ represents a vertex in the graph, and each $e \in E$ represents an edge. Each edge $e = (u, v)$ connects two vertices, $u, v \in V$. In a directed graph, each edge $e = (u, v)$ has a unique direction, from u to v.

Definition 3.1.1. SE(2). *SE(2)* is a topological space that is equivalent to $\mathbb{R}^2 \times \mathbb{S}^1$. It corresponds to the set of all possible transformations to a 2-D rigid body that can translate and rotate in the plane [60].

Definition 3.1.2. Configuration Space. The *configuration space* of the vehicle, $W \subset$ SE(2), is the set of feasible transformations that can be applied to the vehicle [60].

Definition 3.1.3. Lattice Point. A *lattice point* $(x, y, \theta) \in W$ is a single configuration of the vehicle considered by the lattice planner. x and y correspond to displacement from a global origin, and θ corresponds to yaw rotation (heading) relative to a global frame. Lattice points are discretized according to some fixed x spacing Δx, fixed y spacing Δy, and a set of headings Θ.

Definition 3.1.4. Control Action. A *control action* c corresponds to a kinematically feasible spatial path from one lattice point to another. This action results in a transition from a lattice point (x, y, θ) to a lattice point (x', y', θ'), where the relative position $(x' - x, y' - y, \theta' - \theta)$ is fixed for that action. Thus, the action connects all identically arranged pairs of lattice points[85].

Definition 3.1.5. Lattice Graph. A *lattice graph* G is a directed graph of vertices corresponding to lattice points $v \in V$ and edges $e = (u, v) \in E$ where each edge corresponds to the traversal from vertex u to v by a control action c.

Definition 3.1.6. Control Set. A *control set* C is the collection of all control actions considered by the lattice planner. For each heading $\bar{\theta} \in \Theta$ there is an associated control subset $C_{\bar{\theta}} \subseteq C$, which corresponds to the control actions that can be applied at any lattice point $(x, y, \bar{\theta})$.

Definition 3.1.7. Lattice Path. A *lattice path* P_l is a sequence of contiguous edges in the lattice graph. This lattice path corresponds to a spatial path formed by the concatenation of the control actions underlying the sequence of contiguous edges in the lattice path.

3.2 Spiral Path Planning

When constructing lattices, one must select an underlying path representation that allows vertices to connect to one another using the control set. In autonomous driving, vehicles have curvature constraints that prevent the instantaneous turning radius from being too small at any point. To account for this, a commonly used motion primitive is the polynomial spiral [57]. The spiral is defined as a curvature function that is polynomial in arc length, s. In this thesis, when constructing the lattice, a cubic function of arc length is used

$$\kappa(s) = a + bs + cs^2 + ds^3. \tag{3.1}$$

The curvature being a closed form polynomial function of arc length allows one to constrain the curvature along the path, and ensures that the curvature is smooth and well-behaved along the entire path. This is not necessarily true when using spline control actions [79, 9]. In addition, if the initial heading of the vehicle is denoted by θ_0, the heading at any point of the path is easily computed as the integral of the curvature with respect to arc length

$$\theta(s) = \theta_0 + as + \frac{b}{2}s^2 + \frac{c}{3}s^3 + \frac{d}{4}s^4. \tag{3.2}$$

However, the drawback with using polynomial spirals is that they do not have a closed form solution for the x and y position at any intermediate or final point along the path. If the (x, y) position of the start of the spiral is given by (x_0, y_0), then the x and y positions as functions of arc length are given by

$$x(s) = x_0 + \int_0^s \cos(\theta(s'))\ ds', \tag{3.3}$$

$$y(s) = y_0 + \int_0^s \sin(\theta(s'))\ ds'. \tag{3.4}$$

If the final arc length of the spiral is defined as s_f, to satisfy boundary conditions as required by edges in the lattice graph

$$x(s_f) = x_f, \tag{3.5}$$

$$y(s_f) = y_f, \tag{3.6}$$

$$\theta(s_f) = \theta_f, \tag{3.7}$$

$$\kappa(s_f) = \kappa_f, \tag{3.8}$$

one needs to solve for the parameters of the spiral numerically. To encourage an even distribution of curvature across the spiral, it is common to minimize the bending energy of the spiral. The bending energy is the integral of square curvature with respect to arc length along the entire spiral [48].

To limit path curvature, one method is to add constraints for the curvature at evenly spaced points along the spiral [113]. To compute a spiral that satisfies the boundary conditions required by the lattice control set, one can use the following optimization formulation

$$\min_{b,c,d,s_f} \int_0^{s_f} \kappa^2(s)\ ds \quad s.t. \tag{3.9}$$

$$x(s_f) = x_f, \tag{3.10}$$

$$y(s_f) = y_f, \tag{3.11}$$

$$\theta(s_f) = \theta_f, \tag{3.12}$$

$$\kappa(s_f) = \kappa_f, \tag{3.13}$$

$$\left|\kappa\left(\frac{s_f}{3}\right)\right| \leq \kappa_{max}, \tag{3.14}$$

$$\left|\kappa\left(\frac{2s_f}{3}\right)\right| \leq \kappa_{max}, \tag{3.15}$$

where spiral parameter a is given by the initial curvature.

To improve optimization convergence, a common tool is to remap the spiral parameters to the curvature at points uniformly spaced along the spiral [113]. For the above formulation, this results in the following mapping

$$p_1 = \kappa\left(\frac{s_f}{3}\right) = a + b\left(\frac{s_f}{3}\right) + c\left(\frac{s_f}{3}\right)^2 + d\left(\frac{s_f}{3}\right)^3, \tag{3.16}$$

$$p_2 = \kappa\left(\frac{2s_f}{3}\right) = a + b\left(\frac{s_f}{3}\right) + c\left(\frac{2s_f}{3}\right)^2 + d\left(\frac{s_f}{3}\right)^3, \tag{3.17}$$

$$p_3 = \kappa\left(s_f\right) = a + bs_f + cs_f^2 + ds_f^3, \tag{3.18}$$

$$p_4 = s_f. \tag{3.19}$$

p_3 is known immediately from the k_f constraint.

In addition, softening the constraints on the final position improves optimizer performance, using weights α_1, α_2, and α_3. By doing so, the final optimization formulation for generating spirals is then

$$\min_{p_1,p_2,p_4} \int_0^{s_f} \kappa^2(s)\ ds + \alpha_1(x(p_4) - x_f)^2 + \alpha_2(y(p_4) - y_f)^2 + \alpha_3(\theta(p_4) - \theta_f)^2 \quad s.t. \tag{3.20}$$

$$|p_1| \leq \kappa_{max}, \tag{3.21}$$

$$|p_2| \leq \kappa_{max}. \tag{3.22}$$

This formulation also has the added benefit of optimizing over 3 variables instead of 4, which significantly improves performance.

3.3 Kinodynamic Vehicle Models

The analysis in this thesis relies upon three different kinodynamic models. Each model has a varying degree of complexity. The more complex the model, the more faithfully it captures the true dynamics of a car, at the cost of higher computational complexity.

3.3.1 Particle Model

The first model considered is the particle kinematic model, which is used in the RSS framework. As with all kinodynamic models in this section, x is longitudinal displacement and y is lateral displacement. The control input is the acceleration in each dimension, a_x and a_y

$$\ddot{x} = a_x, \tag{3.23}$$

$$\ddot{y} = a_y. \tag{3.24}$$

This model is a two-dimensional double integrator. It is useful because computing optimal swerve manoeuvres with respect to longitudinal distance travelled has a closed form solution [98]. However, it does not capture the non-holonomic constraints present in a vehicle [101], and as a result feasible manoeuvres for the particle model may not be kinematically feasible for a real car.

3.3.2 Kinematic Bicycle Model

In order to generate realistic swerve manoeuvres, one option is to capture the non-holonomic constraints present in a car. To do so, one can use the kinematic bicycle model, a model commonly used in autonomous driving [101, 64, 58]. This model is illustrated in Figure 3.1a.

In this model, v is the velocity of the vehicle, ψ is the heading of velocity at the centre of mass, θ is the yaw of the chassis, β is the slip angle of the centre of mass relative to the chassis, a is the input acceleration, δ is the input steering angle, R_c is the turning radius of the centre of mass, and l_r and l_f are the distances from the rear and front axle to the centre of mass, respectively

$$\begin{aligned}
\dot{x} &= v\cos(\psi+\beta), & \beta &= \tan^{-1}\left(\frac{l_r}{l_r+l_f}\tan(\delta)\right), \\
\dot{y} &= v\sin(\psi+\beta), & \theta &= \psi-\beta, \\
\dot{\theta} &= \frac{v\tan(\delta)}{l_r+l_f}, & |\delta| &\le \delta_{\max} \\
\dot{v} &= a, & |a_{\text{lat}}| &= \frac{v^2}{R_c} \le a_{\min}^{\text{lat}}, \\
R_c &= \frac{l_r+l_f}{\cos(\beta)\tan(\delta)}, & -a_{\text{brake,min}} &\le a \le a_{\max}.
\end{aligned} \tag{3.25}$$

3.3.3 Dynamic Single Track Model

To further improve the representation of the true vehicle model, one can include dynamics. For a car, some important considerations omitted in the previous models include the tire friction, wheel slip, rotational inertia, and drag. These are captured in the dynamic single-track vehicle model [37] with tires modelled using the Pacejka tire model [75]. This model is shown in Figure 3.1b. In this vehicle model, v, ψ, β, δ, l_f, and l_r are the same as the bicycle model (except the direction of β is flipped). α_f and α_r denote the slip angles of the front and rear tires, respectively. F_{sf} and F_{sr} denote the lateral tire forces and F_{lf} and F_{lr} denote the longitudinal tire forces at the front and rear tires, respectively. e_{SP} is the drag mount point, and F_{Ax} and F_{Ay} are the longitudinal and lateral drag forces, respectively. w_z is the yaw rate, and w_δ is the input steering rate. m is the mass of the car, and I_{zz} is the inertia about the z-axis. The vehicle dynamics are given by

$$\dot{x} = v\cos(\psi-\beta), \tag{3.26}$$

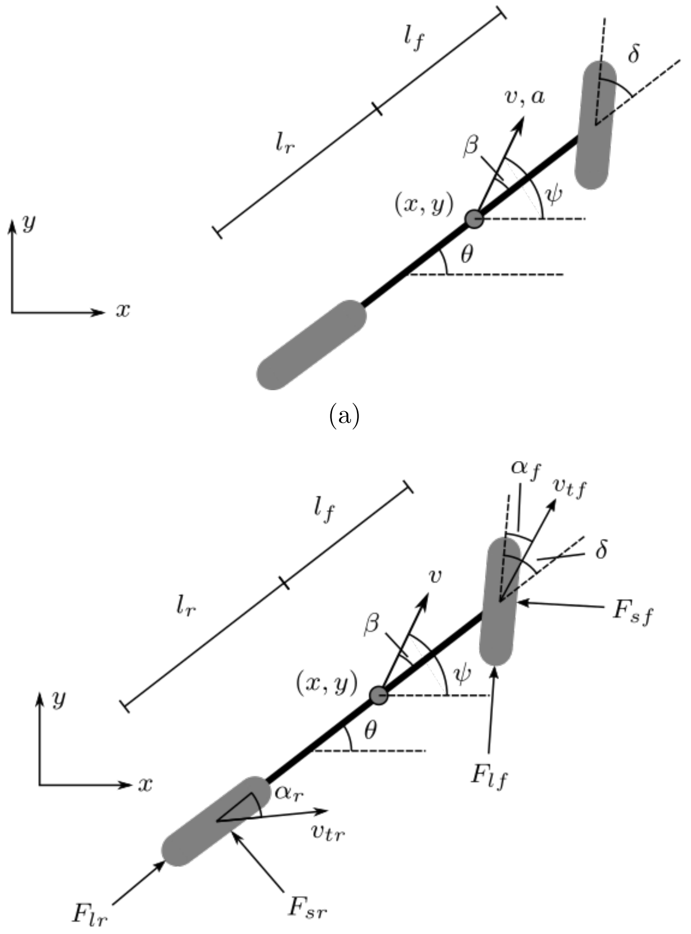
$$\dot{y} = v\sin(\psi-\beta), \tag{3.27}$$

$$\dot{v} = \frac{1}{m}[(F_{lr}-F_{Ax})\cos(\beta) + F_{lf}\cos(\delta+\beta) - (F_{sr}-F_{Ay})\sin(\beta) - F_{sf}\sin(\delta+\beta)], \tag{3.28}$$

$$\dot{\beta} = w_z - \frac{1}{mv}[(F_{lr}-F_{Ax})\sin(\beta) + F_{lf}\sin(\delta+\beta) + (F_{sr}-F_{Ay})\cos(\beta) + F_{sf}\cos(\delta+\beta)], \tag{3.29}$$

$$\dot{\psi} = w_z, \tag{3.30}$$

$$\dot{w}_z = \frac{1}{I_{zz}}[F_{sf}l_f\cos(\delta) - F_{sr}l_r - F_{Ay}e_{SP} + F_{lf}l_f\sin(\delta)], \tag{3.31}$$



(a)

(b)

Figure 3.1: (a) The kinematic bicycle model, along with its associated variables. (b) The dynamic single track model [37]. Drag forces are omitted for simplicity, but are included in computation.

$$\dot{\delta} = w_\delta. \tag{3.32}$$

The lateral tire forces are computed according to the Pacejka model, where A, B, C, D are empirically calculated for the front and rear tires (denoted by subscript f and r, respectively)

$$F_{sf}(\alpha_f) = D_f \sin(C_f \tan^{-1}(B_f \alpha_f - \tan^{-1}(B_f \alpha_f))), \tag{3.33}$$

$$F_{sr}(\alpha_r) = D_r \sin(C_r \tan^{-1}(B_r \alpha_r - \tan^{-1}(B_r \alpha_r))), \tag{3.34}$$

$$\alpha_f = \delta - \tan^{-1}\left(\frac{l_f \dot{\psi} - v\sin(\beta)}{v\cos(\beta)}\right), \tag{3.35}$$

$$\alpha_r = \tan^{-1}\left(\frac{l_r \dot{\psi} + v\sin(\beta)}{v\cos(\beta)}\right). \tag{3.36}$$

The longitudinal front and rear tire forces are based on the braking forces F_{Bf} and F_{Br}, the rolling resistance forces F_{Rf} and F_{Rr}, and the drivetrain moment M_{wheel}, which is a function of the accelerator input ϕ and the gear μ

$$F_{lf} = -F_{Bf} - F_{Rf}, \tag{3.37}$$

$$F_{lr} = \frac{M_{wheel}(\phi, \mu)}{R} - F_{Br} - F_{Rr}. \tag{3.38}$$

The equations for air resistance, rolling resistance, and drivetrain torque are omitted, but are present in the reference [37].

Implicit in the use of this dynamic single-track model is the assumption that the effect of pitching and rolling of the vehicle, which can affect normal forces of the tires, is negligible. This deviation of the normal force at each tire impacts each tire's lateral (cornering) force. In addition, our tire model does not include combined tire slip, which can impact the cornering and braking forces at the tires during aggressive manoeuvres.

3.4 Responsibility-Sensitive Safety

This thesis relies on two aspects of the RSS framework when analyzing safety, the longitudinal and lateral safe distances required between two vehicles. In particular, the second problem of this thesis examines how the equivalent longitudinal safe distance for a swerve manoeuvre compares to that of a brake manoeuvre, while maintaining an appropriate lateral safe distance when required. This thesis compares swerve manoeuvres moving to the left as in Figure 5.1, however, the same analysis is symmetric and can be applied to swerves moving to the right.

In RSS, safe distances are a function of several variables that describe the situation. The initial speed of the rear autonomous vehicle is given by v_r, and the initial speed of the front vehicle is denoted by v_f. The reaction time is given by ρ. The interpretation of the reaction time is the duration after which an agent can apply a mitigating action. During the reaction time, both agents apply the most dangerous acceleration possible, $a_{\text{max,accel}}$, $a_{\text{max,brake}}$ in the longitudinal case, and $a^{\text{lat}}_{\text{max}}$ in the lateral case. This is done to be robust to all possible actions during the reaction delay. To ensure passenger comfort, as well as to prevent tailgater safety issues, the mitigating reaction of the rear vehicle is assumed to be a comfortable deceleration, denoted $a_{\text{min,brake}}$. This term comes from RSS, and is interpreted as the threshold acceleration for a safe, responsible braking response for the self-driving car. As a result, $a_{\text{min,brake}}$ is smaller in magnitude than $a_{\text{max,brake}}$.

The positive part of an expression is denoted with $[\cdot]_+$. If the post-reaction speeds $v_{r,\rho}$ and $v_{f,\rho}$ are given by

$$v_{r,\rho} = v_r + a_{\text{max,accel}}\rho, \tag{3.39}$$

$$v^{\text{lat}}_{r,\rho} = v^{\text{lat}}_r - a^{\text{lat}}_{\text{max}}\rho, \tag{3.40}$$

$$v^{\text{lat}}_{f,\rho} = v^{\text{lat}}_f + a^{\text{lat}}_{\text{max}}\rho, \tag{3.41}$$

then the *longitudinal and lateral safe distances* in RSS are given by

$$d_{\text{long}} = \left[v_r\rho + \frac{1}{2}a_{\text{max,accel}}\rho^2 + \frac{(v_r + v_{r,\rho})^2}{2a_{\text{min,brake}}} - \frac{v_f^2}{2a_{\text{max,brake}}}\right]_+, \tag{3.42}$$

$$d_{\text{lat}} = \mu + \left[-\left(\frac{v^{\text{lat}}_r + v^{\text{lat}}_{r,\rho}}{2}\right)\rho + \frac{(v^{\text{lat}}_{r,\rho})^2}{2a^{\text{lat}}_{\text{min}}} + \frac{v^{\text{lat}}_f + v^{\text{lat}}_{f,\rho}}{2}\rho + \frac{(v^{\text{lat}}_{f,\rho})^2}{2a^{\text{lat}}_{\text{min}}}\right]_+. \tag{3.43}$$

Chapter 5 extends the RSS framework to include swerves. For the purposes of this thesis, the definitions of longitudinal and lateral safe distances are modified as follows.

Definition 3.4.1. Longitudinal Safe Distance. An agent is at a *longitudinal safe distance* from another agent in front of it if the front agent can either decelerate at $a_{\text{max,brake}}$ or perform a swerve manoeuvre and then decelerate at $a_{\text{max,brake}}$, and after reaction delay ρ the rear agent can either brake at $a_{\text{min,brake}}$ or swerve and brake at $a_{\text{min,brake}}$ to avoid collision.

Definition 3.4.2. Lateral Safe Distance. An agent is at a *lateral safe distance* from another agent if both agents can laterally accelerate towards one another at $a_{\text{max}}^{\text{lat}}$, then laterally accelerate away from one another at $a_{\text{min}}^{\text{lat}}$ until reaching zero lateral velocity, while still maintaining at least a μ distance buffer at all times.

The longitudinal safe distance is between the frontmost point of the rear vehicle and the rearmost point of the front vehicle along the longitudinal direction, and the lateral safe distance is between the rightmost point of the rear vehicle and the leftmost point of the front vehicle (or vice versa) along the lateral direction. The distances from the centre of mass to the front and sides of the chassis are left implicit in the original RSS formulation, but since swerves involve rotation of the chassis, these distances are made explicit in this thesis. When computing safety for swerve manoeuvres, the vehicle must maintain these safe distances with relevant agents. These agents are relevant according to longitudinal and lateral adjacency, as defined below. The vehicle dimensions are denoted d_f, d_r, b_l, b_r as in Figure 5.2a. Let the positions of each agent be denoted as (x_1, y_1) and (x_2, y_2).

Definition 3.4.3. Laterally Adjacent. Two agents are *laterally adjacent* if $x_2 - d_r - d_f \leq x_1 \leq x_2 + d_r + d_f$.

Definition 3.4.4. Longitudinally Adjacent. Two agents are *longitudinally adjacent* if $y_2 - b_l - b_r - d_{lat} \leq y_1 \leq y_2 + b_l + b_r + d_{lat}$.

Combining the definitions for safe distances and adjacency gives a definition of safety.

Definition 3.4.5. Laterally/Longitudinally Safe. An agent is *laterally/longitudinally safe* from another agent if it is not laterally/longitudinally adjacent to the other agent, or if it is laterally/longitudinally adjacent to the other agent and there is at least the lateral/longitudinal safe distance between them.

When computing the safe distance required for swerve manoeuvres, the distance at which the swerving agent has cleared the agent in front of it is also useful.

Definition 3.4.6. Lateral Clearance Distance. For a swerving agent and a non-swerving agent, as well as a given swerve manoeuvre, the *lateral clearance distance*, y_c, is defined as the earliest point in the swerve at which the swerving agent is no longer longitudinally adjacent to the non-swerving agent.

In Figure 5.1, y_c is reached at the green dot along the swerve. The lateral clearance distance allows one to compute the longitudinal distance covered by the swerve, which is denoted by x_c. In Chapter 5, x_c is used to compute the equivalent of d_{long} for a swerve manoeuvre, and is compared to Equation 3.42.

Chapter 4

Learning a Lattice Planner Control Set for Autonomous Driving

4.1 Problem Formulation

In this chapter, the main goal is to learn a sparse control set for a lattice planner that retains the driving style that is present in a dataset. The dataset will be a representative sample of trajectories of performing a particular driving task. The process starts with a dense control set and then incrementally generates a subset by selecting the control actions that best improve the ability of the lattice planner to execute the paths present in the dataset. In essence, the dataset paths should become approximate subpaths of lattice paths formed using the learned control set, as in Figure 4.1.

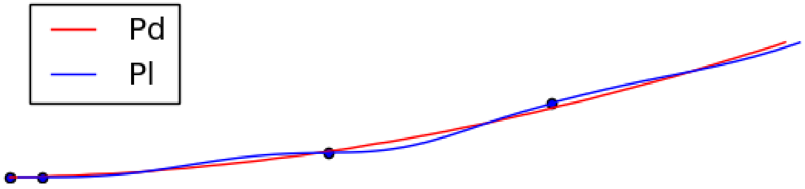


Figure 4.1: An example of the closest path found (blue) by Algorithm 1 with the red path as input.

While optimizing in this way, however, encouraging sparsity is also important, since larger control sets result in longer planning times. This then yields the following high-level problem.

High-Level Problem. Given a dense set of control actions C, and a dataset of representative paths D, compute a minimal subset $\hat{C} \subset C$ that allows a lattice planner to execute the paths present in D.

The high-level problem can be split into two sub-problems. The first is measuring how well control sets match the dataset, and the second is optimizing the control set accordingly.

Subproblem 1. Given a path P_d and a set of control actions $\hat{C}$, compute how well $\hat{C}$ executes P_d according to a scoring measure d.

Subproblem 2. Given a scoring measure d, a dataset of paths D, and a dense set of control actions C, select as small a subset of C, $\hat{C}$, as possible that best executes D in aggregate according to a scoring measure d.

4.2 Sparse Control Set Generation

4.2.1 Scoring Measure

To find the closest path generated by a lattice planner, P_l, to a path in the dataset, P_d, one first needs a scoring measure d to evaluate the similarity of two paths. For two paths parameterized by $t \in [0, 1]$, and two monotonic increasing onto functions $\alpha, \beta : [0, 1] \to [0, 1]$, the Fréchet distance is given by

$$d_f(P_d, P_l) = \inf_{\alpha,\beta} \max_{t\in[0,1]} ||P_d(\alpha(t)) - P_l(\beta(t))||.$$

The scoring measure should rewards P_l for matching P_d closely at each point along the path, where points of comparison are at equal arc lengths along each path. This means that rather than allowing any monotonic increasing traversal of the paths during distance computation as in the Fréchet distance, the paths should be traversed at the same rate. In other words, if both paths were traversed at a constant velocity, then the scoring measure should compare points that are reached at the same time. When traversing both paths at the same rate, path pairs with a low score are likely to have similar driving styles along the entire path.

To get such a scoring measure, the Fréchet distance is then modified as follows. For a given path to match P_d with arc length T, a matching path P_l that is at least as long as P_d, and where t is an arc length parameterization of both paths, then the scoring measure, denoted as d, is

$$d(P_d, P_l) = \max_{t\in[0,T]} ||P_d(t) - P_l(t)||. \tag{4.1}$$

An advantage of using this measure instead of the Fréchet distance is that its simplicity allows for faster computation than the discrete Fréchet distance in a graph[108]. Note that this scoring measure is no longer a distance metric, as it is asymmetrical. The fact that this measure performs a comparison only along the arc length of P_d (and no further) is motivated as follows: rather than forcing the lattice path P_l to be the same length as P_d, P_l can be planned to be arbitrarily longer and then truncated to the arc length of P_d. This opens up a greater number of terminal lattice vertices when computing P_l, which results in closer matching paths and faster runtime. The generation of P_l is discussed in further detail in Section 4.2.2.

Now, assume that d is calculated for two discrete paths, sampled with respect to arc length with segments of equal length δ. Appendix A, contains implementation details, including how to handle paths with length not integer-divisible by δ. Let P_d contain K sampled path points, $\{0, ..., K-1\}$, where the 0^{th} point is the origin. Let $P_d(k), P_l(k)$ denote the k^{th} path point of each respective path. Then Equation (4.1) simplifies to

$$d(P_d, P_l) = \max_{k \in \{0,...,K-1\}} ||P_d(k) - P_l(k)||. \tag{4.2}$$

Equation (4.2) can be evaluated in $O(K)$ time.

Finally, the algorithm discussed in the section below requires the calculation of d between a control action $c \in C$ and a sub-path of an input path, where the sub-path starts at path point k_1 and ends at path point k_2 of P_d. In this case, both c and the sub-path have $k_2 - k_1$ segments between path points. This is denoted by

$$d(P_d, c, k_1, k_2) = \max_{k \in \{k_1,...,k_2\}} ||P_d(k) - c(k - k_1)||. \tag{4.3}$$

4.2.2 Closest Path Algorithm

In lattice planning, one typically searches for the shortest path in the lattice graph to some goal point or region, where the lattice graph is constructed according to a particular control set. However, to address Subproblem 1 of Section 4.1, the path P_l in the lattice graph with minimum distance d to a given dataset path P_d must be found instead. It is assumed both paths start at the origin O.

Algorithm 1 solves this problem. In the following subsections, the first describes the input of a given problem instance. The next sections then discuss how to generate a search graph for the algorithm, followed by the algorithm's searching process. The final section analyzes the proposed algorithm.

Algorithm Input

Figure 1.1 illustrates example input to the algorithm. Here there is a dataset path P_d overlaid on top of a lattice graph constructed from an input control set C. The labelled vertices correspond to particular positions and headings in space. A single heading across all vertices is displayed for visual clarity, except at vertex k, which contains a control set for an alternative initial heading in orange. The edges correspond to the underlying paths of the control actions that join points in space according to C. The set C is illustrated adjacent to the lattice graph. The underlying paths of each control action are uniformly sampled with arc length δ, and the corresponding number of path points along each control action's path (excluding the origin point) are given as labels.

Each path is represented by a sequence of discrete path points, and as a result, the scoring measure d requires that the k^{th} point along P_l be compared with the k^{th} point along P_d during computation. To handle this, when generating the search graph the lattice vertex is augmented with the number of discrete path points k along the path used to reach said lattice vertex. The number of discrete path points is given by the sum of path points along the control actions that compose the path. This is illustrated in Figure 4.2. The integer value alongside each edge denotes the number of path points along each control action. The augmenting integer value shown in the vertex is the total number of path points used to reach the lattice vertex. As can be seen, lattice vertex g can be reached in two different ways, each requiring a different number of path points, and as a result, there are two different search graph vertices overlapping in space.

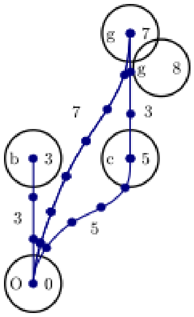


Figure 4.2: An illustration of how lattice vertices are augmented in the search graph. The control actions are labelled with their length, in terms of discrete path points.

Search Graph

This section describes the construction of the search graph. As discussed in the previous section, there are multiple ways to reach vertices within the lattice graph (for example, vertex l in Figure 1.1, some of which have different numbers of path points used along the way. If P_d contains K path points, the search graph contains up to K copies of each vertex in the lattice graph to compute d. Each copy is differentiated by the number of path points required to reach it.

These copies are illustrated in Figure 4.3. Revisiting vertex l, there are now three copies of l in the search graph, each of which have a different value for the number of path points required to reach it. The copies all correspond to the same point in configuration space, but with a different number of path points used to reach them. This is similar to vertex g in Figure 4.2.

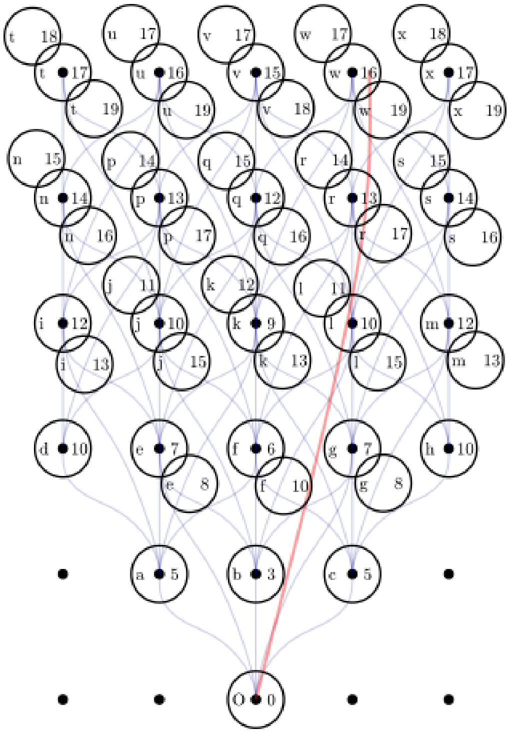


Figure 4.3: The search graph derived from Figure 1.1. Overlapping vertices correspond to the same point in configuration space, but reached with a different number of path points. Some vertices are omitted for visual clarity.

To illustrate why the search graph is useful, suppose one wishes to compute the d scoring measure of the control action from $(g, 7)$ to $(l, 10)$, as in Equation (4.3). This is shown in Figure 4.4a. The path points along this edge must be compared to the path points 7 to 10 of P_d. This is shown by the dark green line segments between both paths. The scoring measure of the control action from $(g, 7)$ to $(l, 10)$ is then the length of the longest dark green line. However, if instead one wishes to compute the d scoring measure of the control action from $(h, 10)$ to $(l, 15)$, the path points 10 to 15 of P_d must be compared. This comparison is given by the light green lines between the paths. In this way, the search graph allows the algorithm to keep track of how much of P_d has already been used in the computation of the d scoring measure, and as a result, match the underlying paths of each edge to the proper portion of P_d.

Algorithm 1 CLOSESTPATH(P_d, C, O, B)

```
bestEnd ← $O$
costs, predecessors ← HashTable()
$K$ = length($P_d$)
$V$ = Array(HashTable(), $K$)
$V[0][O] = O$
costs$[O, 0] = 0$
for all $i \in 0, ..., K - 1$ do
    for all $u \in V[i]$ do
        for all $c \in C_{u.\theta}$ do
            $(v, j)$ ← applyControlAction($u, c, i$)
            $d_{u,v} \leftarrow d(P_d, c, i, j)$
            if $d_{u,v} > B$ then
                continue
            $V[j][v] = v$
            if max(costs$[u, i], d_{u,v}$) < costs$[v, j]$ then
                predecessors$[v]$ ← $u$
                costs$[v, j]$ ← max(costs$[u, i], d_{u,v}$)
            if costs$[v, j] < B$ and $j \geq K$ then
                bestEnd ← $v$
                $B$ ← costs$[v, j]$
return (bestEnd, predecessors)
```

Search Process

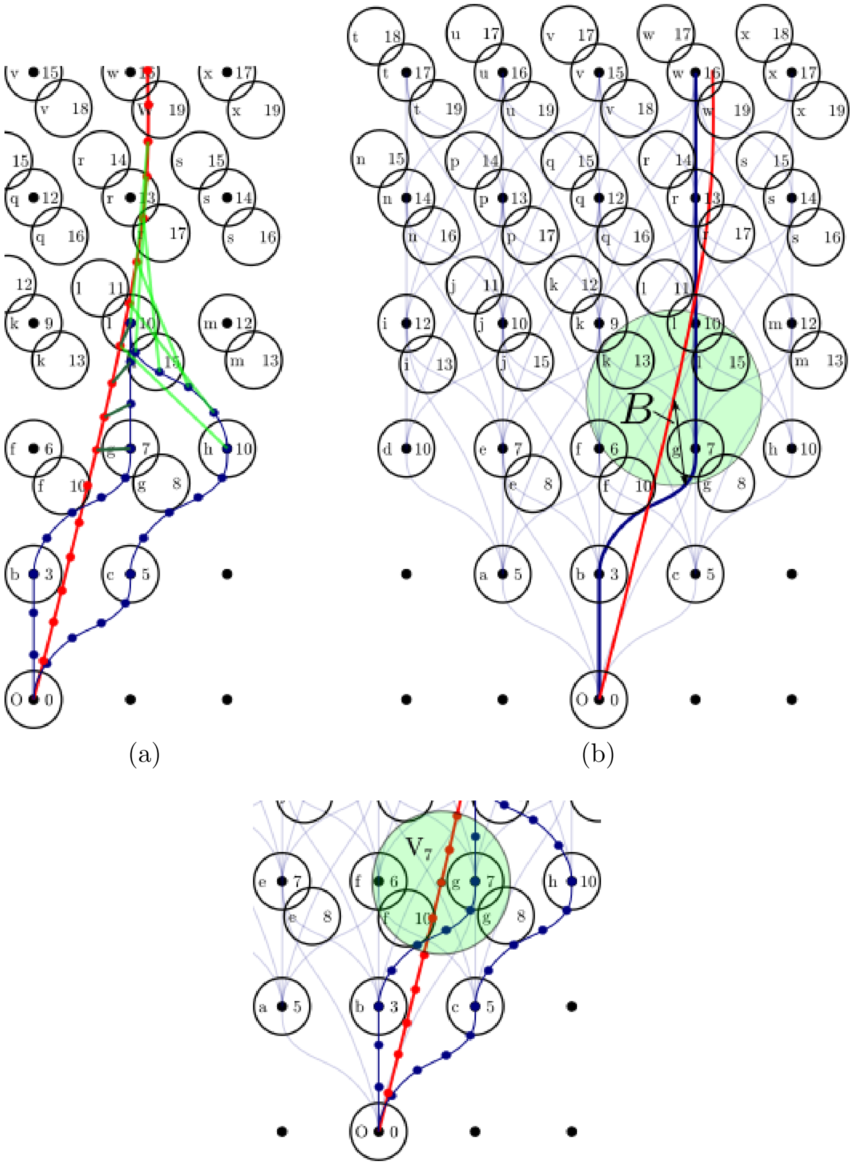
Recall Equation (4.2) solves for the maximum pointwise distance between P_d and P_l. The algorithm then seeks to minimize this distance, i.e., find the closest path to P_d in the search graph. As the algorithm explores the search graph, it keeps track of the maximum pointwise distance computed along the closest path that reaches each search graph vertex.

To solve this search problem, first denote the set of search graph vertices that require k path points to reach them as V_k, and the collection of all V_k as V as shown in Line 4 of Algorithm 1. All edges entering a vertex in V_k come from some vertex in $V_{k'}$ such that $k' < k$. This then gives the vertices in the search graph a topological ordering that can be exploited, which is iterated through in Lines 7-20. Through each iteration, successor vertices are found through applyControlAction(), which takes in a lattice vertex, a control action, and the path point i of that vertex, and outputs the successor lattice vertex as well as the resulting path point j after applying the control action. The next step is to apply a dynamic programming update for each search graph vertex in every V_k in increasing order of k that computes the closest scoring measure across all paths to each search graph vertex. If costs[] stores the best d measure found so far for each vertex, U is the set of all predecessors of vertex (v, j), and $d_{u,v}$ is computed for the control action linking (u, i) to (v, j) according to Equation (4.3), then the update is given by

$$\text{costs}[v, j] = \min_{u \in U} \max(\text{costs}[u, i], d_{u,v}).$$

This update is shown in Lines 15 to 17.

To reduce the number of vertices searched, an upper bound is computed on the optimal d scoring measure by greedily selecting control actions that minimize the d of the appropriate section of P_l. An example of the greedy selection process is given in Figure 4.4b. The maximum distance from P_d along the greedily selected path, according to the scoring measure d, is then an upper bound on the optimal scoring measure in the lattice. This bound, denoted as B, then defines a radius around each k^{th} point of the path P_d. B then restricts the size of each V_k. This is illustrated in Figure 4.4c. Only points within the shaded green circle can meet the scoring measure threshold B given by the greedy path. This means that $(g, 7)$ belongs to V_7, but $(e, 7)$ does not, as it is too distant. As a result, outgoing control actions that reach $(e, 7)$ can be safely ignored, as any path that passes through them is not as “close” to P_d as the greedily selected path. This is shown on Lines 12-13. Recall the lattice resolution is given by Δx and Δy. If $A = \Delta x \Delta y$, the cardinality



(a) (b) (c)

Figure 4.4: (a) An example scoring measure computation to vertex l. The light green line segments correspond to comparisons for the control action coming out of $(h, 10)$, and the dark green lines represent comparisons for the control action coming out of $(g, 7)$. (b) An example of a greedily selected path, and the resulting upper bound B at the point of maximum deviation along the path. (c) An illustration of a particular V_k based on the greedy bound on the scoring measure.

of each set V_k is bounded by $\lceil \frac{B}{\Delta x} \rceil \lceil \frac{B}{\Delta y} \rceil |\Theta| \in O(\frac{B^2}{A}|\Theta|)$.

Figure 4.1 gives an example solution using this method. The algorithm takes in a path to follow, P_d, a control set, C, the origin of the lattice, O, and the greedy bound, B, as input. The algorithm starts at the origin, iterating through each V_k and applying the dynamic programming update described above. The V_k are populated during the graph search by successively applying control actions.

For example, if the search starts at the origin in Figure 4.4b, the vertices $(b, 3)$ and $(g, 7)$ can be reached using control actions that remain within the greedy bound B, so these vertices are added to V_3 and V_7, respectively, with predecessor $(O, 0)$. The best scoring measure for each search graph vertex (as well as the associated predecessor vertex) is stored as the search progresses. On the next iteration, the algorithm goes through all the vertices in V_3, since it is the set of vertices that is closest to the origin. There is only one, $(b, 3)$, and there are two outgoing vertices that satisfy B, $(f, 6)$, and $(g, 8)$. These are then added to V_6, and V_8, respectively, with $(b, 3)$ as their predecessor vertex. If the algorithm encounters a destination vertex more than once, its saved score is updated with the minimum scoring measure value. This continues until all viable vertices have been searched. The vertex in the vertex sets V_k, for $k \geq K$, which has the minimum score then denotes the terminal vertex of the closest path P_l.

Algorithm Analysis

This section discusses the correctness and runtime of Algorithm 1. In the algorithm, an empty entry in the costs hash table corresponds to infinite cost. To prove the algorithm is correct, the proof shows that when each vertex is processed in topological order, the cost for said vertex is the minimum across all incoming paths. In terms of runtime, recall that B is the greedy bound, $A = \Delta x \Delta y$, K is the number of points in P_d. In addition, the maximum number of path points across all control actions is denoted as N.

Theorem 1. *Algorithm 1 is correct, and has runtime* $O(N\frac{B^2}{A}K|C|)$.

Proof. The first part of this proof demonstrates correctness. This can be shown through induction on the vertices processed from V, as well as the fact that the vertices are processed in topological order.

Induction Assumption. For each vertex $u \in V_k$ processed from each $V_k \in V$, we have that the cost assigned to u is the minimal d possible on any path from the origin to u, when comparing said path to u to the subpath $P_d(0:k)$.

Base Case. The origin is the first processed vertex, and since P_d starts at the origin, d is zero, which is the correct distance.

Induction. Now, assume every processed vertex satisfies the induction assumption. Suppose vertex v is the current vertex to be processed. Since the algorithm processes vertices in topological order, all potential predecessors of v have already been processed, and therefore satisfy the induction assumption. By the dynamic programming update, taking U to be the set of predecessors of v, we then have that

$$\text{costs}[v] = \min_{u \in U} \max(\text{costs}[u], d_{u,v}).$$

Now, let u' in $V_{0:k-1}$ denote the optimal predecessor of v. By the update, we have that

$$\text{costs}[v] \leq \max(\text{costs}[u'], d_{u',v}),$$

thus the induction assumption holds for v.

For runtime, Algorithm 1 iterates through a topological ordering of the search graph, which can be thought of as K groups of at most $\frac{B^2}{A}$ vertices. For each vertex in the topological ordering, we perform a dynamic programming update for each control action available to it. Across all headings, the total number of control actions available to any particular vertex is $|C_{\bar{\theta}}|$, which in aggregate yields $\sum_{\bar{\theta} \in \Theta} |C_{\bar{\theta}}| = |C|$. Each dynamic programming update calculates d for an edge, which takes $O(N)$ time. Combining, this gives the algorithm a computational complexity of $O(N\frac{B^2}{A}K|C|)$. □

The runtime is heavily dependent on the quality of the bound B provided, as a tight bound results in far fewer vertices to search. The N factor is generally small relative to $K|C|$, so for a tight greedy bound B the runtime of the algorithm approaches $O(K|C|)$. This would be ideal, as it corresponds to searching the control set at each point along the path.

4.2.3 Control Set Optimization

This section presents a method for optimizing the control set structure such that it is best able to reproduce a given dataset. This is required to address Subproblem 2 in Section 4.1.

Recall that the objective is to select as small of a subset as possible, $\hat{C}$, of an original dense control set C, while still maintaining the ability to execute the paths in a given dataset. To accomplish this, the objective function should trade off between the sparsity of $\hat{C}$ and the ability of $\hat{C}$ to match the dataset. Recall that the scoring measure in Equation (4.2) is denoted as d, the dataset of paths as D, the initial dense control set as C, and the optimized control set as $\hat{C}$. Define the set of all potential paths in the lattice as $\mathcal{P}(\hat{C})$, and the parameter that trades off between sparsity and dataset matching as λ. Then, the resulting objective formulation is

$$\min_{\hat{C} \subset C} \frac{1}{|D|} \sum_{P_d \in D} \min_{P_l \in \mathcal{P}(\hat{C})} d(P_d, P_l) + \lambda \frac{|\hat{C}|}{|C|}. \tag{4.4}$$

For each P_d, d is computed between P_d and the closest path in the lattice graph constructed from $\hat{C}$, and summing over the entire dataset. This value is normalized by the size of the dataset, to ensure consistency between different dataset sizes. The second term penalizes the size of the learned control set to encourage sparsity, and is normalized by the size of the initial dense control set. The λ term is what trades off between sparsity and dataset matching; a larger λ results in a sparser control set, whereas a smaller λ allows the control set to fit the data more closely. In this sense, the λ term acts as a regularizer in the objective function. Occam's Razor objective functions that encourage simplicity are commonly used for tasks such as model selection or learning, one of which is the Bayesian Information Criterion (BIC)[70].

To perform the optimization, one starts with a small control set $\hat{C}$. The optimization then greedily add the control action that results in the largest decrease in Equation (4.4), and repeat until no control action can be added to further decrease the objective. Algorithm 1 is used when computing the closest path according to d as required by Equation (4.4).

4.2.4 Clustering

The optimization method above requires evaluation of the objective function for each available control action not yet within $\hat{C}$ across all dataset paths to determine which control action is best to add. However, this is computationally expensive. In addition, real world data often contains many similar paths. This is because there are often a limited number of ways to navigate a given scenario, and certain ways are more common than others. To alleviate these issues, the dataset is first clustered using the K-means algorithm[70].

To measure the distance between paths, the pointwise Euclidean norm[14] is used. An example of a clustering result is shown in Figure 4.5.

After clustering, the search process is biased based on how well the learned control set is currently matching each path cluster. Initially, each cluster has a large, equal weight. The optimization algorithm proceeds as follows:

Control Set Optimization

1. Select a path cluster according to the selection weights, and randomly sample a subset of the path cluster and a subset of control actions.

2. Compute the optimization objective for these subsets, adding each control action individually to $\hat{C}$ and calling Algorithm 1 for each path in the cluster subset.

3. Add the control action that decreases the objective the most to $\hat{C}$ permanently. Terminate if no control action improved the objective.

4. Update the cluster selection weights with the resulting value of the optimization objective. Return to Step 1.

This method focuses the optimization on clusters that are poorly matched. Through this process, the optimization runs faster, and is more likely to match all types of paths present in the dataset, rather than the most common ones.

4.3 Experiment and Results

To evaluate the preceding method, this thesis analyzes three experiments. The first two used data from human-driven trajectories around a roundabout, and the third used synthetic paths created through randomly generated scenarios. In all three experiments, there is an 85%-15% split of the dataset between the training and test sets. The algorithms were written in Julia. The source code for the experiments can be seen at `https://github.com/rdeiaco/learning_lattice_planner`. For all experiments, the dense initial control set was a set of cubic spirals[57] arranged in a cone, generated for all $\theta \in \Theta$. The endpoints of the control actions in the cone had a range of x values between 0.4m and 4.0m, a range of y values between -2.0m to 2.0m, and θ values within $[0, \tan^{-1}(\frac{1}{3}), \tan^{-1}(\frac{1}{2}), \frac{\pi}{4}, \tan^{-1}(2), \tan^{-1}(3)]$. These angles were chosen because they encourage straight line traversal between vertices in the lattice graph, which improves path quality[81]. The initial dense control set is shown in Figure 4.6a.

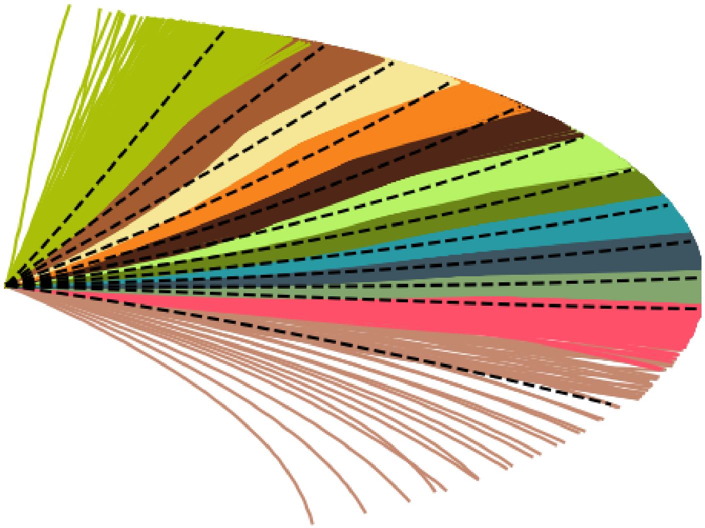


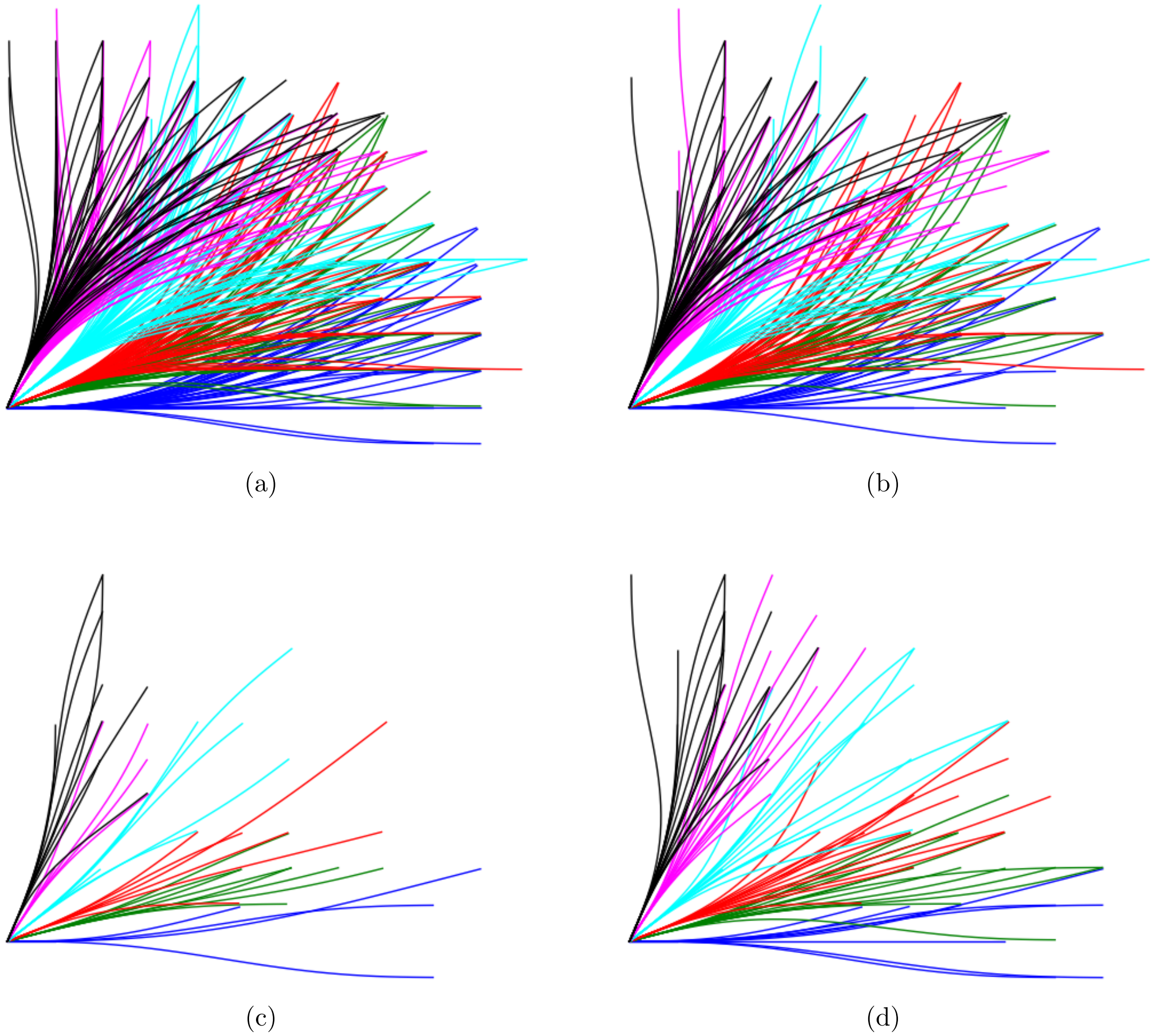
Figure 4.5: An example of the K-means clustering on a roundabout path dataset. Each cluster of paths has a different assigned colour, and the dotted line represents each cluster's mean path.

Each experiment compares the performance of the learning algorithm to the state-of-the-art lattice computation algorithm[82]. The learning algorithm was run with $\lambda_1 = 0.311$ and $\lambda_2 = 0.0311$. These values were determined by logarithmically spaced grid search. Values of λ larger than this were found to generate control sets that were too sparse with poor manoeuvrability. Swath-based collision checking was performed using a rectangular vehicle footprint of length 4.5m and width 1.7m. Since the goal was not necessarily reachable in the lattice graph, the lattice planner instead searched for goal points that minimized the distance and heading difference from this goal.

4.3.1 Experimental Setup

Experiment 1: Roundabout Scenario The first experiment involved taking 213 paths in a roundabout dataset[1]and sampling them at a constant arc length step size. The roundabout is illustrated in Figure 4.7a. The training portion of the dataset was then sliced into 10m arc length slices using a sliding window with a 1m step size. These slices were then taken as input to the clustering and optimization algorithms. This slicing method allows the extraction of as much information as possible from the dataset[2]. To evaluate

[1] Dataset obtained with permission from DataFromSky. The paths were extracted from cars driving through a European roundabout. The paths ranged in length from 27.6 to 87.4m.



(a)

(b)

(c)

(d)

Figure 4.6: Comparison of the dense (a), DL[82] (b), λ_1 (c), λ_2 (d) control sets generated in Experiment 2. Each colour corresponds to a different $C_{\bar{\theta}}$.

the learned control sets, the test portion of the dataset was used to construct scenarios from each path. This was done by taking the test set path as the lane centerline, with lateral offsets from the path forming the lane boundaries in an occupancy grid. Finally, the endpoint of the test set path was used as the goal. Using the occupancy grid, a lattice planner was run using each generated control set to compare the quality of each control set's planned paths.

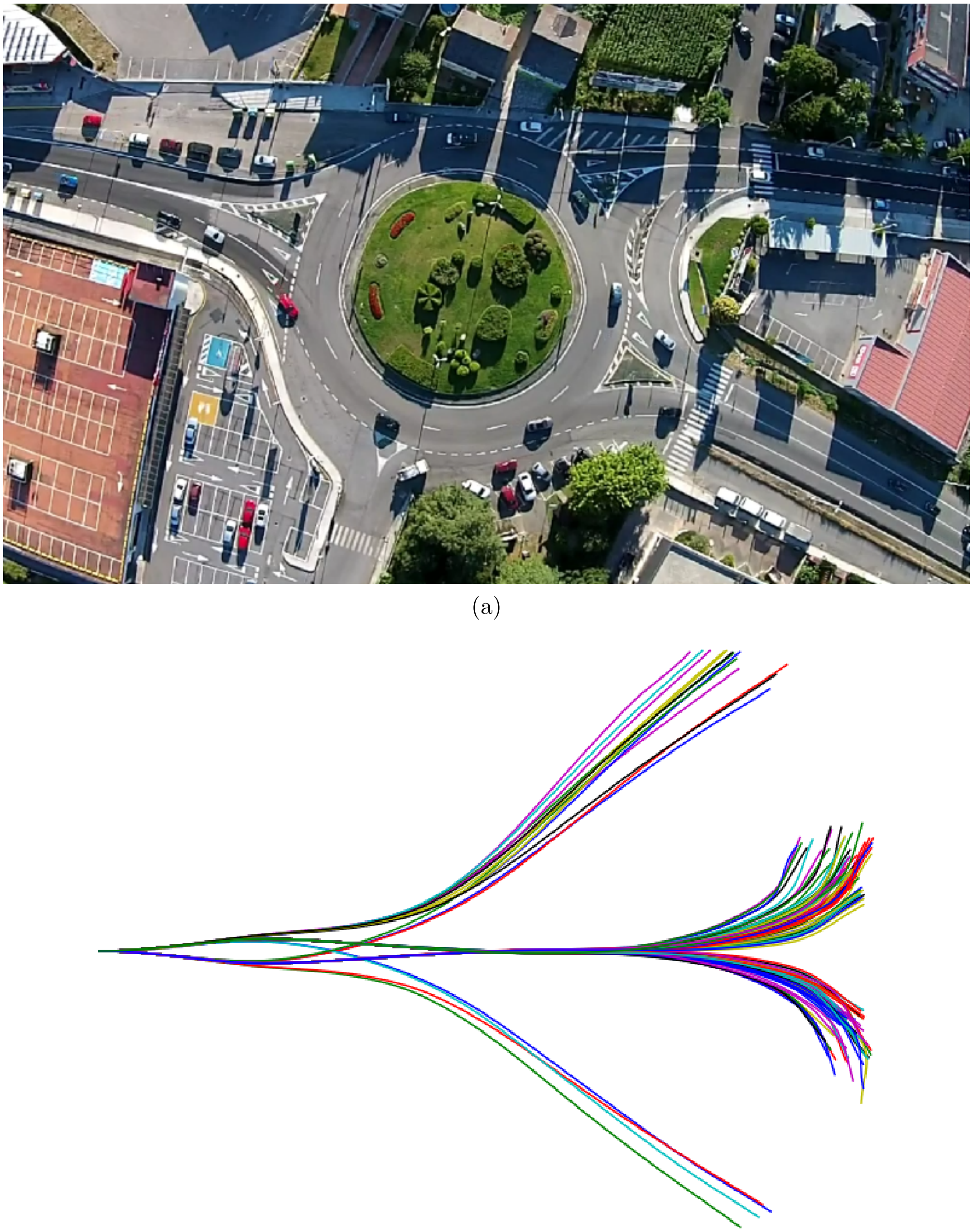
Experiment 2: Roundabout Lane Change Scenario The second experiment also involved the same training paths from the roundabout dataset, except this time a second lane was added to the test set by extending the lateral offset forming the lane boundaries. Rather than the goal being to travel to the end of the original lane, the goal was changed to be the end of the adjacent lane. This meant that the planner was required to perform a lane change, in order to demonstrate that the learned control set could generalize to a situation not explicitly present in the training set. The direction of the lane change was equally distributed between a left and right lane change. Otherwise, scenario generation was the same as in Experiment 1.

Experiment 3: Synthetic Double Swerve Scenario The third experiment involved generating 100 different lane structures by randomly sampling clothoids of varying length and curvature connected to straightaways of varying length. Next, a second lane was added, along with an obstacle in the first lane. The goal of this experiment was for the planner to perform a double swerve manoeuvre to avoid the obstacle. The motion planner currently used on the University of Waterloo Autonomoose self-driving car[118] was then used to generate the training set of synthetic paths. This dataset is shown in Figure 4.7b.

4.3.2 Experimental Results

The results of all 3 experiments are shown in Table 4.1. It shows that the learned control sets are significantly smaller than both the dense control set as well as the control set formed after performing the DL[82] lattice computation algorithm, illustrated in Figure 4.6. Notably, this results in up to an approximately 7.5x planning speedup over the dense set and up to a 4.31x planning speedup over the DL[82] set when executing the test set.

To measure how well each control set matched the dataset in terms of driving style, the curvature at each point was computed along each planned path and dataset path as a proxy for the steering function, as discussed in Section 1.1. Next, the maximum difference in curvature was computed between each path point along the planned path and the dataset



(a)

(b)

Figure 4.7: (a) The roundabout the dataset was extracted from for Experiments 1 and 2. (b) The synthetic dataset generated using the Autonomoose planner.

Table 4.1: Planning Runtime Results

Experiment 1	Dense	DL[82]	λ_1	λ_2
Control Set Size	311	194	64	109
Planning Speedup Ratio	1.00	1.82	6.40	3.49
Matching Differential (31 Scenarios)	-	-1	+9	+11
Experiment 2				
Control Set Size	311	194	65	109
Planning Speedup Ratio	1.00	1.73	7.46	3.83
Matching Differential (31 Scenarios)	-	+7	+13	+23
Experiment 3				
Control Set Size	311	194	57	83
Planning Speedup Ratio	1.00	1.90	7.73	4.70
Matching Differential (15 Scenarios)	-	+5	+11	+13

path. This is called the *curvature matching score*. Afterwards, these curvature matching scores were compared across the planned paths for each control set. The value in the table reports the number of times a planned path had a lower maximum curvature deviation than the dense set's planned path; a positive number denotes the control set was better at matching more often than the dense set, and negative the opposite. A sample comparison between the DL control set and the λ_2 control set is given in Figure 4.8.

This shows that the learned control sets match the driving style (measured by curvature) of the dataset more closely than both the dense and DL[82] control sets, while also offering faster planning times. In addition, it is clear that as λ gets smaller, the planned paths more closely match the data, at the cost of a larger control set and slower planning times.

Figure 4.9 shows a sample planning run from Experiment 3, comparing all 4 control sets. The red box denotes the obstacle for the scenario. It is clear all 4 planners were able to complete a plan to the goal state equally well, which shows that the learned planners had no loss of manoeuvrability.

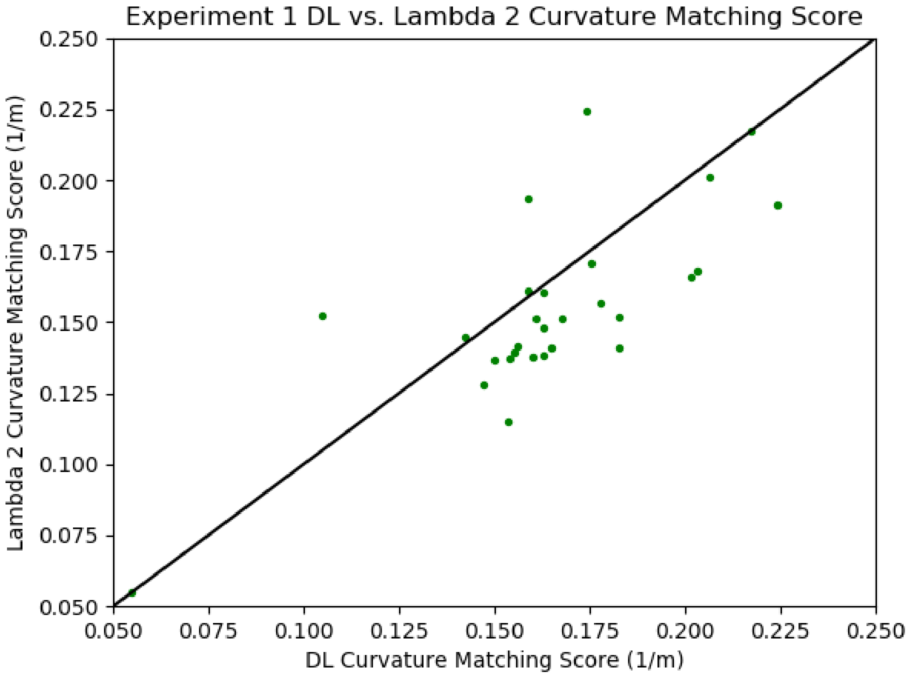


Figure 4.8: An example comparison of the curvature values between planned paths using the DL[82] and λ_2 control sets. Each datapoint corresponds to a test scenario; below the straight line means that the λ_2 control set performed better.

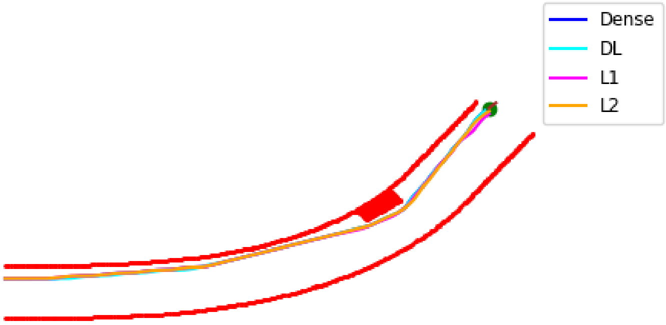


Figure 4.9: Comparison of the lattice planner paths for the dense, DL[82], λ_1, and λ_2 control sets for one of the scenarios in Experiment 3.

Chapter 5

Safe Swerve Manoeuvres for Autonomous Driving

5.1 Swerve Problem Formulation

The fundamental problem this chapter addresses is to compute the longitudinal safe distance required when there is a free lane (or shoulder) to the left or right of the vehicle, allowing for an evasive swerve manoeuvre. This requires knowing the longitudinal safe distance required for a braking lead vehicle as well as a swerving lead vehicle. These manoeuvres are illustrated in Figure 5.1. As can be seen, when computing the longitudinal safe distances for swerves, one needs to consider both longitudinal and lateral clearance, since swerves contain lateral and longitudinal displacement.

Since vehicles rotate during swerves, rotation must compensated for when computing these clearances. After compensating for rotation, the distance x_c can then be used to compute the longitudinal safe distance required for a swerve. In RSS, safety was proved for a particle model. This chapter extends those results to prove the safety for swerves feasible for the kinematic bicycle model. It is then shown how this result can be applied to more general models in Section 5.3. This task then breaks down into five subproblems.

Subproblem 1. Given the initial speed of the swerving vehicle v_r, the vehicle dimensions d_f, d_r, b_l, b_r as in Figure 5.2a, and parameters μ and ρ, compute a lateral clearance distance y_c sufficient for lateral safety when a swerving vehicle becomes laterally adjacent to a lead vehicle.

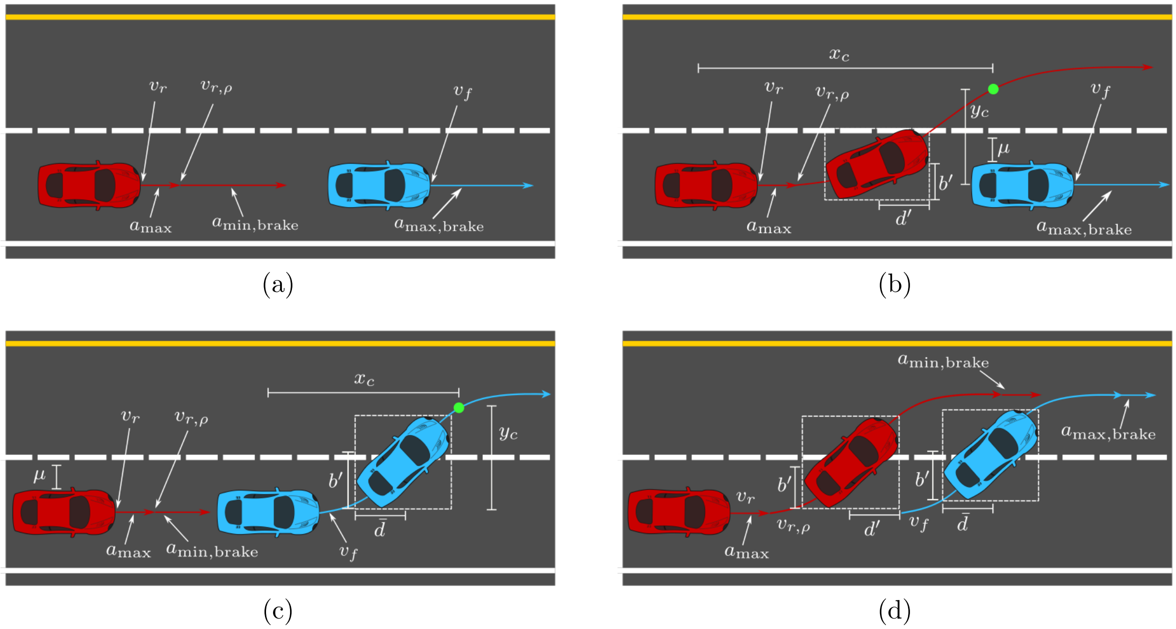


Figure 5.1: (a) The standard RSS braking manoeuvre for a braking leading vehicle. Velocities and acceleration arrows point to path segment where they occur. (b) The proposed swerve manoeuvre for a leading braking vehicle. The green dot represents the lateral clearance point (x_c, y_c), according to RSS μ-lateral distance. (c) The braking manoeuvre required for a swerving leading vehicle. (d) The swerving manoeuvre required for a swerving leading vehicle.

Subproblem 2. Given the kinematic constraints in (3.25), the initial vehicle speeds v_r and v_f, the lateral clearance distance y_c, and parameters ρ, $a_{\max}$, $a_{\min,\text{brake}}$, $a_{\max,\text{brake}}$, $a_{\max}^{\text{lat}}$, and $a_{\min}^{\text{lat}}$, compute a longitudinal safe distance sufficient for safety when swerving for a braking lead vehicle. This is illustrated in Figure 5.1b.

Subproblem 3. Given the initial vehicle speeds v_r and v_f, the clearance point y_c, and parameters ρ, $a_{\max}$, $a_{\min,\text{brake}}$, $a_{\max}^{\text{lat}}$, and $a_{\min}^{\text{lat}}$, compute a longitudinal safe distance sufficient for safety when braking for a swerving lead vehicle. This is illustrated in Figure 5.1c.

Subproblem 4. Given the kinematic constraints in (3.25), the initial vehicle speeds v_r and v_f, the parameters ρ, $a_{\max}$, $a_{\min,\text{brake}}$, $a_{\max,\text{brake}}$, $a_{\max}^{\text{lat}}$, and $a_{\min}^{\text{lat}}$, compute a longitudinal safe distance sufficient for safety when swerving behind a swerving lead vehicle. This is illustrated in Figure 5.1d.

Subproblem 5. Given longitudinal safe distance sufficient for safety when swerving for a braking vehicle, braking for a swerving lead vehicle, and swerving for a swerving lead vehicle, compute a longitudinal safe distance akin to d_{long} that is sufficient for universal safety when maintained by all vehicles on the road.

The first subproblem is addressed in Section 5.2.1, the second in Section 5.2.2, the third in Section 5.2.3, the fourth in Section 5.2.4, and the fifth in Section 5.2.5.

The work in this chapter makes the following assumptions on responsible behaviour:

1. A vehicle will only perform a swerve manoeuvre if it is not braking, and will only perform a brake manoeuvre if it is not swerving.

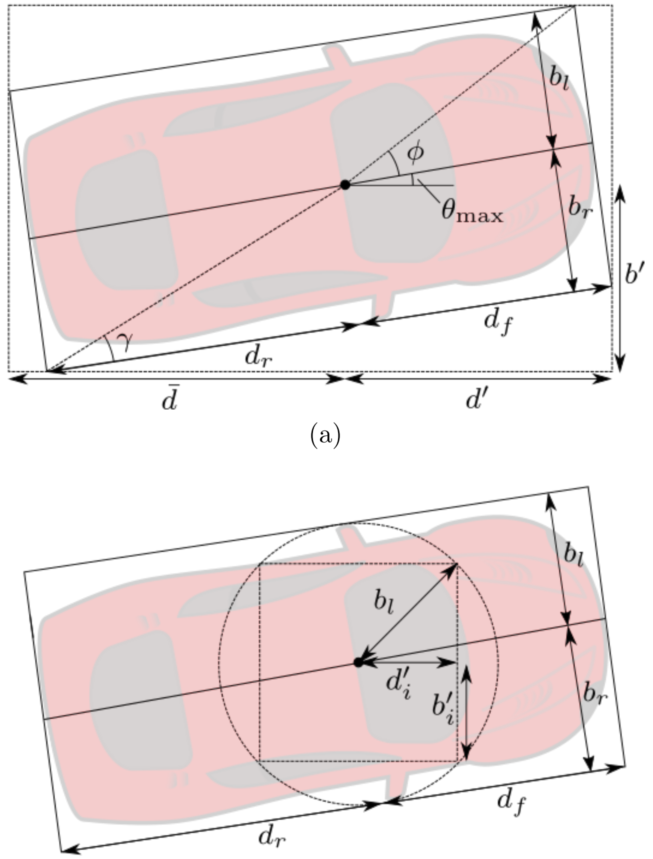
2. For every swerve manoeuvre, each vehicle reaches the lateral clearance distance only once. As a result, once an vehicle has committed to a lane change by reaching the lateral clearance distance, it will not return to its previous lane.

3. Each vehicle moves forward along the road, $v \geq 0$ and $\frac{-\pi}{2} \leq \psi \leq \frac{\pi}{2}$.

5.2 Computing the Longitudinal Safe Distance

5.2.1 Lateral Clearance Distance

The lateral clearance distance, as was defined in Section 3.4, is used to determine when a swerving vehicle is laterally safe from a braking lead vehicle. To compute the lateral clearance distance, denoted y_c, Equation (3.43) is modified to account for vehicle rotation. If the maximum chassis yaw $\theta_{\max}$ during the manoeuvre is known, an axis-aligned bounding rectangle can be computed as an outer approximation to the vehicle footprint. This is useful for safety analysis, and is illustrated in Figure 5.2a.



(a)

(b)

Figure 5.2: (a) An outer approximation to a vehicle chassis that rotates by $\theta_{\max}$. d' and $\bar{d}$ are used for longitudinal buffers during swerve manoeuvres, and b' is used as a lateral buffer. (b) An inner approximation to a rotating vehicle chassis.

The three distances needed for safety analysis are from the centre of mass to the front of the bounding rectangle, d', from the centre of mass to the side of the bounding rectangle, b', and from the centre of mass to the rear of the bounding rectangle, $\bar{d}$. d_r and d_f are the distances from the centre of mass to the rear and front of the chassis, respectively. b_l and b_r are the distances to the left and right of the chassis, respectively. As the vehicle rotates, the length and width of the bounding rectangle increases until $\theta_{\max}$ reaches the angles from the centre of mass to the corners of the rectangle. Further rotation past these points decreases the dimensions of the bounding rectangle. These angles can be written in terms of ϕ and γ, illustrated in Figure 5.2a. The equations for the bounding rectangle distances are then d', $\bar{d}$, and b' are

$$d' = \begin{cases} d_f \cos(\theta_{\max}) + b_r \sin(\theta_{\max}) & \theta_{\max} \leq \phi, \\ \sqrt{d_f^2 + b_r^2} & \theta_{\max} > \phi, \end{cases} \tag{5.1}$$

$$\bar{d} = \begin{cases} d_r \cos(\theta_{\max}) + b_l \sin(\theta_{\max}) & \theta_{\max} \leq \gamma, \\ \sqrt{d_r^2 + b_l^2} & \theta_{\max} > \gamma, \end{cases} \tag{5.2}$$

$$b' = \begin{cases} d_r \sin(\theta_{\max}) + b_r \cos(\theta_{\max}) & \theta_{\max} \leq \frac{\pi}{2} - \gamma, \\ \sqrt{d_r^2 + b_r^2} & \theta_{\max} > \frac{\pi}{2} - \gamma. \end{cases} \tag{5.3}$$

Using $\theta_{\max}$, which is computed in Section 5.2.2, the required bounding rectangle dimensions of a rotating vehicle can be computed.

Using b' and the lateral clearance distance d_{lat}, the lateral clearance distance y_c can be computed, as required for Subproblem 1.

$$y_c = b' + b_l + d_{\text{lat}}. \tag{5.4}$$

Denote the time y_c is attained as t_c.

Theorem 2. *Equation 5.4 gives a lateral clearance distance sufficient for lateral safety when a swerving vehicle becomes laterally adjacent to a lead vehicle, or any time before.*

Proof. To show lateral safety, we must show that laterally adjacent agents are at least d_{lat} from one another, as given in Equation 3.43. Since the swerving agent's lateral speed is variable but nonnegative, a conservative lower bound on its lateral velocity is zero when computing d_{lat}. From assumption 1, since the other agent is braking, it is not swerving, and therefore has lateral velocity during the swerve. The required d_{lat} can then be computed using Equation 3.43, taking v_r^{lat} and v_f^{lat} to be zero, and using the parameters $a_{\min}^{\text{lat}}$, $a_{\max}^{\text{lat}}$, and ρ. The distance d_{lat} acts as a buffer to ensure that upon reaching lateral adjacency, both agents are laterally safe from one another.

For $t < t_c$, the swerving vehicle is not laterally adjacent to the other vehicle, and is laterally safe. For $t \geq t_c$, from Assumption 2, t_c is the time at which the two vehicles are closest while laterally adjacent. From Equation 5.4, there is at least d_{lat} of distance between the agents, and thus they are laterally safe $\forall t \geq t_c$. □

5.2.2 Swerving for a Braking Vehicle

The clearance distance y_c can be used to compute the longitudinal safe distance required when swerving to avoid a braking lead vehicle, denoted $d_{s,b}$. This distance $d_{s,b}$ is computed under the constraints of the bicycle model outlined in Section 3.3. In addition, if α denotes the lane width, t_f denotes the end time of the swerve, and the origin of the coordinate frame is at the centre line of the current lane at the rear vehicle's position at $t = 0$, the swerve must satisfy the following boundary conditions:

$$\theta(t_f) = 0, \; y(t_f) = \alpha. \tag{5.5}$$

However, computing the optimal bicycle swerve manoeuvre with respect to longitudinal clearance is an optimization problem with no closed form solution [98]. Instead, one can compute a swerve manoeuvre feasible for the kinematic bicycle model, and use that to obtain an upper bound on the actual longitudinal distance required by a swerve constrained by the kinematic bicycle model.

As in Equation 3.42, the lead vehicle is travelling with velocity v_f, and then brakes at $a_{\text{max,brake}}$ during the entire manoeuvre. The swerve is preceded by the rear vehicle maximally accelerating during the reaction delay ρ, at which point it begins the swerve manoeuvre with post-acceleration velocity $v_{r,\rho}$. To ensure monotonicity in the gap between the rear and lead vehicles, a lower bound on the distance travelled until t_f by lead vehicle is used, denoted x_f.

The swerve considered is bang-bang in the steering input with zero longitudinal acceleration, and is illustrated in Figure 5.3. The longitudinal distance travelled by the swerving vehicle until the swerving vehicle reaches the lateral clearance distance is denoted as x_c. This distance x_c is computed in Equations 5.16 and 5.21.

For the swerve manoeuvre, the turning radius of the circular arcs depends on the maximum lateral acceleration, as well as the kinematic limits of the steering angle. The constraints on steering angle and lateral acceleration from (3.25) give two constraints on the turning radius

$$R_{\min,\delta} = \sqrt{\frac{(l_r + l_f)^2}{\tan(\delta_{\max})^2 + l_r^2}}, \tag{5.6}$$

$$R_{\min,a} = \frac{v_{r,\rho}^2}{a_{\min}^{\text{lat}}}. \tag{5.7}$$

To ensure both constraints are satisfied, R_c from (3.25) is set to the maximum of the two. From this turning radius, the steering angle δ_c and the slip angle β_c can be computed

$$\delta_c = \tan^{-1}\left(\sqrt{\frac{(l_r + l_f)^2}{R_c^2 - l_r^2}}\right), \tag{5.8}$$

$$\beta_c = \tan^{-1}\left(\frac{l_r \tan(\delta_c)}{l_r + l_f}\right). \tag{5.9}$$

The $\theta_{\max}$ required to satisfy the boundary conditions in Equation 5.5 can now be computed. The angle $\theta_{\max}$ denotes how far the vehicle travels along each circular arc, which gives the switching point for the bang-bang steering control. From the rear axle, the two circular arcs are symmetrical in lateral distance travelled, as in Figure 5.3. Therefore, the angle along the first circular arc required to reach a lateral distance of $\frac{\alpha}{2}$ can be computed. First, the turning radius at the rear axle, R_r, is computed

$$R_r = \frac{l_r + l_f}{\tan(\delta_c)}. \tag{5.10}$$

The lateral distance travelled during the first circular arc is then given by

$$y(t) = R_r(1 - \cos(\theta(t))). \tag{5.11}$$

For a given value of δ_c, $\theta_{\max}$ is then

$$\theta_{\max} = \cos^{-1}\left(1 - \frac{\alpha}{2R_r}\right). \tag{5.12}$$

To compute x_c, there are two cases, depending on if y_c is reached in the first or second circular arc. The angle $\psi_{\max}$ can be computed using (3.25). From Assumption 3, $\psi_{\max} \leq \frac{\pi}{2}$. Thus, the first case occurs if

$$y_c \leq R_c(\cos(\beta_c) - \cos(\psi_{\max})), \tag{5.13}$$

otherwise the second case occurs.

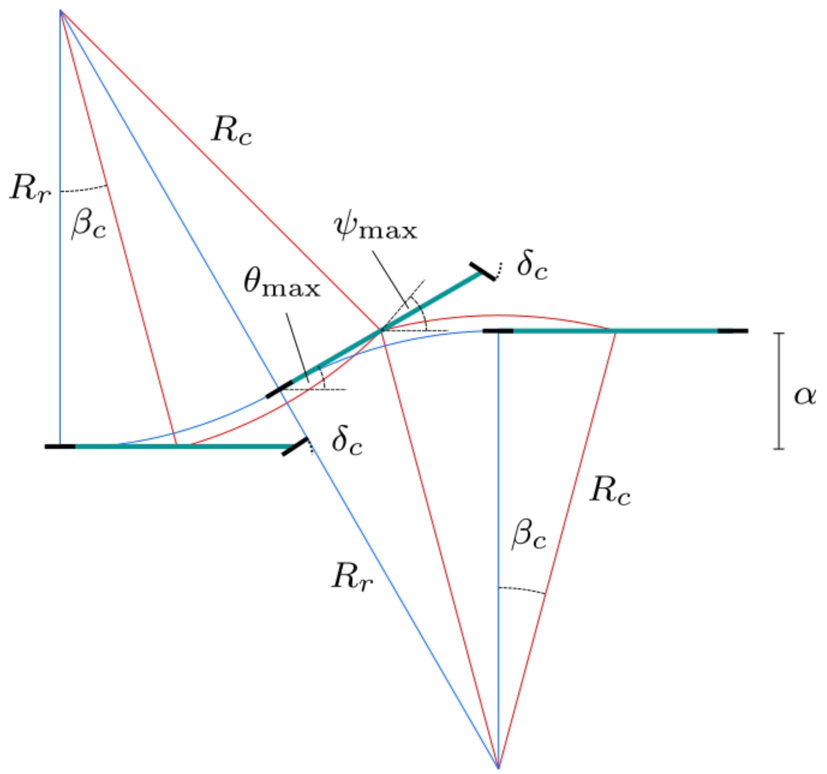


Figure 5.3: The swerve manoeuvre used for safety analysis. The red path is taken by the centre of mass, and the blue path is taken by the rear axle. α *is the distance between lanes,* δ_c *is the steering angle,* β_c *is the slip angle. The maximum angles achieved by the chassis yaw and the velocity of the centre of mass are given by* $\theta_{\max}$ *and* $\psi_{\max}$*, respectively. The turning radius of the rear axle and centre of mass's paths are given by* R_r *and* R_c*, respectively.*

First Circular Arc

Similar to Equation 5.11, the longitudinal position along the first circular arc is given by

$$x(t) = R_c(\sin(\psi(t)) - \sin(\beta_c)). \tag{5.14}$$

Using the centre of mass equivalent of Equation 5.11 and y_c, the ψ value at the clearance point, ψ_c, can be computed

$$\psi_c = \cos^{-1}\left(\cos(\beta_c) - \frac{y_c}{R_c}\right). \tag{5.15}$$

Substituting this value for ψ in Equation 5.14, and adding the outer approximation for the chassis d' gives the longitudinal swerve clearance distance

$$x_c = R_c(\sin(\psi_c) - \sin(\beta_c)) + d'. \tag{5.16}$$

The magnitude of the velocity is constant during the swerve, and so t_c can be computed using the arc length travelled up to the clearance point y_c,

$$t_c = \frac{R_c(\psi_c - \beta_c)}{v}. \tag{5.17}$$

Second Circular Arc

In the second circular arc, the initial heading of the centre of mass is denoted as $\hat{\psi} = \psi_{\max} - 2\beta_c$, the initial x position as $\hat{x} = R_c(\sin(\psi_{\max}) - \sin(\beta_c))$, and the initial y position as $\hat{y} = R_c(\cos(\beta_c) - \cos(\psi_{\max}))$. The longitudinal and lateral distances along this arc are then

$$x(t) = R_c(\sin(\hat{\psi}) - \sin(\psi(t))) + \hat{x}, \tag{5.18}$$

$$y(t) = R_c(\cos(\psi(t)) - \cos(\hat{\psi})) + \hat{y}. \tag{5.19}$$

As in Case 1, substituting y_c yields ψ_c,

$$\psi_c = \cos^{-1}\left(\frac{1}{R_c}(y_c - \hat{y}) + \cos(\hat{\psi})\right) \tag{5.20}$$

Substituting this value for ψ in Equation 5.18 gives

$$x_c = R_c(\sin(\hat{\psi}) - \sin(\psi_c)) + \hat{x}. \tag{5.21}$$

Similar to Case 1, the clearance time, t_c, can be computed

$$t_c = \frac{R_c(\psi_{\max} - \beta_c + \hat{\psi} - \psi_c)}{v}. \tag{5.22}$$

From these longitudinal swerve clearance values, the longitudinal safe distance can then be computed. To do this, the rear braking distance in Equation 3.42 can be replaced with the longitudinal swerve distance x_c. In addition, to ensure a monotonically decreasing gap between the two vehicles, the initial speed of the lead vehicle is set (as a conservative lower bound) to

$$v'_f = \min(v_f, v_r \cos(\psi_{\max})). \tag{5.23}$$

The distance travelled by the lead vehicle, x_f, depends on the the clearance time t_c. If $\rho + t_c \geq \frac{v'_f}{a_{\max,\text{brake}}}$, then the lead vehicle brakes to a stop during the swerve, and the distance travelled is the stopping distance. Otherwise, it is the distance travelled during deceleration up to time t_c. Thus, the distance travelled by the lead vehicle is given by

$$x_f = \begin{cases} v'_f(\rho + t_c) - \frac{a_{\max,\text{brake}}(\rho+t_c)^2}{2}, & \rho + t_c \leq \frac{v'_f}{a_{\max,\text{brake}}}, \\ \frac{v'^2_f}{2a_{\max,\text{brake}}}, & \rho + t_c > \frac{v'_f}{a_{\max,\text{brake}}}. \end{cases} \tag{5.24}$$

Using the parameters $a_{\max,\text{accel}}$ and ρ introduced in Section 3.4, and using Equations 5.16, 5.21, and 5.24, the longitudinal safe distance between a swerving rear vehicle and a braking lead vehicle, $d_{s,b}$, is then

$$d_{s,b} = \left[v_r\rho + \frac{1}{2}a_{\max,\text{accel}}\rho^2 + x_c - x_f \right]_+ + d' + d_r. \tag{5.25}$$

Theorem 3. *Equation 5.25 gives a longitudinal safe distance sufficient for safety when swerving for a braking lead vehicle.*

Proof. For $t > t_c$, $y(t) > y_c$, and therefore the swerving vehicle is no longer longitudinally adjacent to the lead vehicle, so is safe from the lead vehicle's braking. For $t \leq t_c$, from Equation 5.23, a conservative lower bound is used for the speed of the lead vehicle to ensure the lead vehicle's speed is less than the swerving vehicle during the entire swerve. This implies the gap between the two vehicles is monotonically decreasing, which further implies that the minimum gap between the two vehicles occurs at time t_c.

The swerving vehicle travels $x_c + v_r\rho + \frac{1}{2}a_{\text{max,accel}}\rho^2$, and a conservative lower bound on the lead vehicle's travel distance is given by x_f. There is at most d' of distance from the centre of mass to the front of the swerving vehicle, and there is a constant d of distance from the front vehicle centre of mass to the front vehicle's bumper. Thus, if a swerving vehicle maintains distance $d_{s,b}$, it will not collide with a braking vehicle in front of it, and is therefore safe from the lead vehicle at time t_c. Since the gap is monotonically decreasing for $t \leq t_c$, it is safe $\forall t \leq t_c$.

□

5.2.3 Braking for a Swerving Vehicle

The longitudinal safe distance required to swerve for a braking vehicle was computed in the preceding section, and this section considers the opposite problem, computing the longitudinal safe distance required to brake for a swerving lead vehicle without collision. Since the lead vehicle intends to occupy the other lane, it requires less longitudinal distance for the rear vehicle to brake to avoid the swerving lead vehicle than it would for it to brake for a braking lead vehicle. It is assumed the front vehicle is performing the same swerve discussed in Section 5.2.2. To account for rotation of the front vehicle, $\bar{d}$ is used to compensate as defined in Section 5.2.1.

Equations 5.16, 5.21, 5.17, and 5.22 can be used to compute the x_c and t_c for the front vehicle's swerve. As in Equation 3.42, it is assumed that the rear vehicle accelerates maximally during its reaction time, and then brakes comfortably until t_c. As before, denote the rear vehicle's post-acceleration velocity as $v_{r,\rho}$. Then its minimum velocity during the braking manoeuvre is

$$v_{r,\min} = \max(\min(v_r, v_{r,\rho} - a_{\text{min,brake}}(t_c - \rho)), 0). \tag{5.26}$$

As in Section 5.2.2, the proof of safety is simplified if the gap is monotonically decreasing until lateral safety is reached. To ensure this, the lead vehicle speed is conservatively approximated with v'_f

$$v'_f = \min(v_f \cos(\psi_{\max}), v_{r,\min}). \tag{5.27}$$

A conservative lower bound for the longitudinal distance travelled by the swerving front vehicle is then

$$x_f = v'_f t_c. \tag{5.28}$$

The distance x_f is a lower bound on the distance travelled by the front vehicle during the swerve that creates a monotonically decreasing gap.

The distance travelled by the rear braking vehicle during its reactions delay and its braking manoeuvre is denoted by x_r. This distance depends on the clearance time t_c, similar to the distance travelled by the front vehicle in the preceding section. The distance travelled during the rear vehicle's braking manoeuvre, $x_{r,\text{brake}}$, is given by

$$x_{r,\text{brake}} = \begin{cases} v_{r,\rho}(t_c - \rho) - \frac{a_{\text{min,brake}}(t_c-\rho)^2}{2}, & t_c - \rho \leq \frac{v_{r,\rho}}{a_{\text{min,brake}}}, \\ \frac{v_{r,\rho}^2}{2a_{\text{min,brake}}}, & t_c - \rho > \frac{v_{r,\rho}}{a_{\text{min,brake}}}. \end{cases} \tag{5.29}$$

Following this, the distance travelled by the braking rear vehicle is

$$x_r = \frac{(v_r + v_{r,\rho})\rho}{2} + x_{r,\text{brake}}. \tag{5.30}$$

Using Equations 5.28 and 5.30, the longitudinal safe distance when braking for a swerving vehicle, $d_{b,s}$ is then

$$d_{b,s} = [x_r - x_f]_+ + d_f + \bar{d}. \tag{5.31}$$

Theorem 4. *Equation 5.31 gives a longitudinal safe distance sufficient for safety when braking for a swerving lead vehicle.*

Proof. For $t > t_c$, the swerving vehicle is laterally clear from the rear braking vehicle, and therefore the rear vehicle is safe. The velocity used for the lead vehicle is a conservative lower bound on its true speed $\forall t \leq t_c$, as per Equation 5.27. In addition, $v'_f \leq v_r$, $\forall t \leq t_c$, and as a result the gap between the two vehicles is monotonically decreasing on that interval. The minimum distance between the two vehicles thus occurs at time t_c. Equation 5.31 thus gives enough clearance such that no collision occurs at time t_c, so the rear vehicle is safe at time t_c. Since the gap is monotonically decreasing over the interval, the rear vehicle is safe $\forall t \leq t_c$. □

5.2.4 Swerving for a Swerving Vehicle

The final relevant longitudinal safe distance is the distance required when swerving behind a swerving lead vehicle. This is illustrated in Figure 5.1d. Both vehicles are longitudinally adjacent during the entire manoeuvre. From Assumption 1, the lead vehicle will not brake during its swerve. The goal is then to compute the longitudinal distance required to swerve behind a lead swerving vehicle, such that if the lead vehicle were to immediately brake with deceleration $a_{\text{max,accel}}$ at the end of its swerve, and the rear vehicle were to brake with deceleration $a_{\text{min,accel}}$ at the end of its reaction-delayed swerve, there would be no collision. Using Equation 5.12 to compute the maximum yaw angle during each vehicle's swerve, $\theta_{\max,r}$, $\theta_{\max,f}$, and Equation 5.10 to compute each vehicle's rear axle turning radius, $R_{r,r}$, $R_{r,f}$, the swerve completion times of the rear and front vehicle are given by t_1 and t_2, respectively

$$t_1 = \frac{2R_{r,r}(\theta_{\max,r})}{v_r}, \tag{5.32}$$

$$t_2 = \frac{2R_{r,f}(\theta_{\max,f})}{v_f}. \tag{5.33}$$

Similar to the previous section, v'_f denotes a conservative lower bound on the front vehicle's speed

$$v'_f = \min(v_f \cos(\psi_{\max,f}), v_r). \tag{5.34}$$

The longitudinal safe distance required to swerve in response to a swerving vehicle, $d_{s,s}$, is then

$$d_{s,s} = \frac{v_r + v_{r,\rho}}{2}\rho + v_{r,\rho}(t_1 - \rho) + \frac{v_{r,\rho}^2}{2a_{\text{min,brake}}} - \left(v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}}\right) + d' + \bar{d}. \tag{5.35}$$

Theorem 5. *Equation 5.35 gives a longitudinal safe distance sufficient for safety when swerving for a swerving lead vehicle.*

Proof. The gap between each vehicle can be written as a piecewise function of time. The endpoints of the intervals are the reaction delay, ρ, the time it takes for the front vehicle to finish its swerve, t_2, the time it takes for the rear vehicle to finish its swerve, t_1, the brake time of the front vehicle, $t_{b,2}$, and the brake time of the rear vehicle, $t_{b,1}$. The swerve times for the kinematic bicycle model for varying speeds are proportional to $v \cos^{-1}\left(1 - \frac{1}{v^2}\right)$, which is quasi-constant across all relevant road speeds. In addition, $a_{\text{max,accel}} > a_{\text{min,accel}}$, and swerve times are longer than reasonable reaction times. From this, it is reasonable to assume that $\rho < t_2 < \rho + t_1 < t_2 + t_{b,2} < \rho + t_1 + t_{b,1}$. If the longitudinal distance travelled during the swerves by the front and rear vehicle are denoted by $x_{s,2}(t)$ and $x_{s,1}(t)$ respectively, the initial gap between the vehicles is denoted by g_0, and (for the moment) the distances from the centre of mass to the front and rear bumpers are ignored, then the gap as a function of time $g(t)$ is given by

$$g(t) = \begin{cases} g_0 + x_{s,2}(t) - (v_r t + \frac{1}{2}a_{\text{max,accel}}t^2) & t \leq \rho, \\ g_0 + x_{s,2}(t) - (\frac{v_r+v_{r,\rho}}{2}\rho + x_{s,1}(t-\rho)) & \rho < t \leq t_2, \\ g_0 + x_{s,2}(t_2) + v_f(t-t_2) - \frac{1}{2}a_{\text{max,brake}}(t-t_2)^2 - & \\ \quad (\frac{v_r+v_{r,\rho}}{2}\rho + x_{s,1}(t-\rho)) & t_2 < t \leq \rho + t_1, \\ g_0 + x_{s,2}(t_2) + v_f(t-t_2) - \frac{1}{2}a_{\text{max,brake}}(t-t_2)^2 - & \\ \quad (\frac{v_r+v_{r,\rho}}{2}\rho + x_{s,1}(t_1-\rho) + v_{r,\rho}(t-t_1) - \frac{1}{2}a_{\text{min,brake}}(t-t_1)^2) & \rho + t_1 < t \leq t_2 + t_{b,2}, \\ g_0 + x_{s,2}(t_2) + \frac{v_f^2}{2a_{\text{max,brake}}} - (\frac{v_r+v_{r,\rho}}{2}\rho + x_{s,1}(t_1-\rho) + & \\ \quad v_{r,\rho}(t-t_1) - \frac{1}{2}a_{\text{min,brake}}(t-t_1)^2) & t_2 + t_{b,2} < t \leq \rho + t_1 + t_{b,1}, \\ g_0 + x_{s,2}(t_2) + \frac{v_f^2}{2a_{\text{max,brake}}} - (\frac{v_r+v_{r,\rho}}{2}\rho + x_{s,1}(t_1-\rho) + \frac{v_{r,\rho}^2}{2a_{\text{min,brake}}}) & t > \rho + t_1 + t_{b,1}. \end{cases} \tag{5.36}$$

The maximum longitudinal velocity during the rear vehicle swerve is $v_{r,\rho}$. If the maximum ψ value during the front vehicles swerve is denoted $\psi_{\text{max},f}$, the minimum longitudinal velocity during the front vehicle's swerve is given by $v_f \cos(\psi_{\text{max},f})$. Set $v'_f = \min(v_f \cos(\psi_{\text{max},f}), v_r)$. This means that

$$x_{s,1}(t) \leq v_{r,\rho}t, \tag{5.37}$$

$$x_{s,2}(t) \geq v'_f t. \tag{5.38}$$

Substituting this in Equation 5.36 results in a monotonically decreasing function of t, $\hat{g}(t)$, with the property that $\hat{g}(t) \leq g(t), \forall t$

$$\hat{g}(t) = \begin{cases} g_0 + v'_f t - (v_r t + \frac{1}{2}a_{\text{max,accel}}t^2) & t \leq \rho, \\ g_0 + v'_f t - (\frac{v_r+v_{r,\rho}}{2}\rho + v_{r,\rho}(t-\rho)) & \rho < t \leq t_2, \\ g_0 + v'_f t - \frac{1}{2}a_{\text{max,brake}}(t-t_2)^2 - (\frac{v_r+v_{r,\rho}}{2}\rho + v_{r,\rho}(t-\rho)) & t_2 < t \leq \rho + t_1, \\ g_0 + v'_f t - \frac{1}{2}a_{\text{max,brake}}(t-t_2)^2 - & \\ \quad (\frac{v_r+v_{r,\rho}}{2}\rho + v_{r,\rho}(t-\rho) - \frac{1}{2}a_{\text{min,brake}}(t-t_1)^2) & \rho + t_1 < t \leq t_2 + t_{b,2}, \\ g_0 + v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}} - (\frac{v_r+v_{r,\rho}}{2}\rho + v_{r,\rho}(t-\rho) - & \\ \quad \frac{1}{2}a_{\text{min,brake}}(t-t_1)^2) & t_2 + t_{b,2} < t \leq \rho + t_1 + t_{b,1}, \\ g_0 + v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}} - (\frac{v_r+v_{r,\rho}}{2}\rho + v_{r,\rho}(t_1-\rho) + \frac{v_{r,\rho}^2}{2a_{\text{min,brake}}}) & t > \rho + t_1 + t_{b,1}. \end{cases} \tag{5.39}$$

This implies that the minimum of $\hat{g}(t)$ occurs for $t > t_{b,1}$, where $\hat{g}(t)$ is constant

$$\min_t \hat{g}(t) = g_0 + v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}} - \left(\frac{v_r + v_{r,\rho}}{2}\rho + v_{r,\rho}(t_1 - \rho) + \frac{v^2_{r,\rho}}{2a_{\text{min,brake}}} \right). \tag{5.40}$$

Since $\hat{g}(t) \leq g(t), \forall t$, if $\hat{g}(t) \geq 0, \forall t$, no collision occurs. This is satisfied if the initial gap satisfies

$$g_0 \geq \frac{v_r + v_{r,\rho}}{2}\rho + v_{r,\rho}(t_1 - \rho) + \frac{v^2_{r,\rho}}{2a_{\text{min,brake}}} - \left(v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}} \right). \tag{5.41}$$

By adding in the distances from the centre of mass to the ends of the chassis, compensating for the rotation of each swerving vehicle, an initial gap is sufficient for safety $\forall t$ if

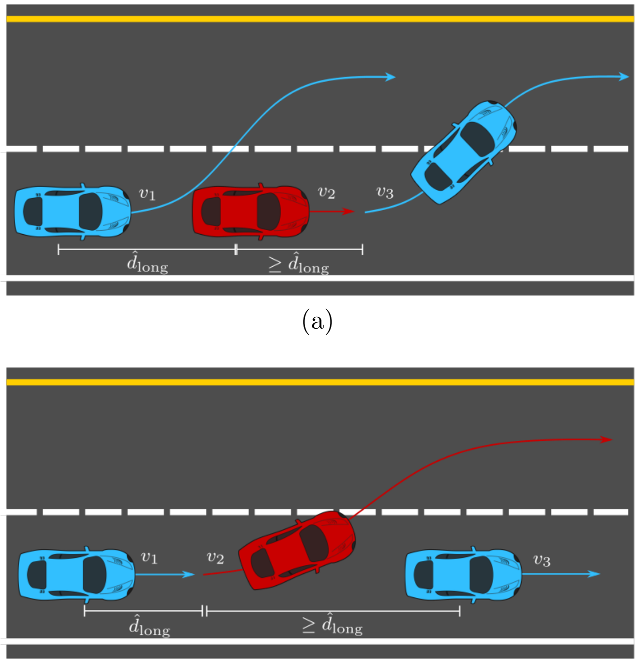
$$g_0 \geq \frac{v_r + v_{r,\rho}}{2}\rho + v_{r,\rho}(t_1 - \rho) + \frac{v^2_{r,\rho}}{2a_{\text{min,brake}}} - \left(v'_f t_2 + \frac{v'^2_f}{2a_{\text{max,brake}}} \right) + d' + \bar{d}. \tag{5.42}$$

Which yields Equation 5.35.

At $t \geq t_2$, the time at which the lead vehicle begins hard braking, there is enough longitudinal distance to brake for the leading vehicle, as $\hat{g}(t) \geq 0, \forall t \geq t_2$, so the rear vehicle is safe. Since $\hat{g}(t)$ in Equation 5.39 is monotonically decreasing with t, the safe longitudinal distance is satisfied for $t < t_2$, and thus the rear vehicle is safe $\forall t$. □

5.2.5 Universal Following Distance

The final subproblem addressed in this chapter aims to combine the results of the previous sections into a final following distance that can be maintained by all vehicles in a given straight road system to ensure universal safety, assuming the vehicles can brake or swerve as a response to the behaviour of other vehicles in front of them. In this sense, this section extends the analysis of the preceding sections into the case of more than two vehicles in a road system. The following distance will be a function of the speed of the vehicle, as well as the speed of the 2 vehicles in front of the vehicle, and the parameters outlined in 3.4. Denote the distance required to brake for a braking lead vehicle as $d_{\text{b,b}}(v_r, v_f, \rho)$, the distance required to swerve for a braking lead vehicle as $d_{\text{b,s}}(v_r, v_f, \rho)$, the distance required to swerve for a braking lead vehicle as $d_{\text{s,b}}(v_r, v_f, \rho)$, and the distance required to swerve for a swerving lead vehicle as $d_{\text{s,s}}(v_r, v_f, \rho)$.



(a)

(b)

Figure 5.4: (a) Scenario where the rear vehicle must swerve for a swerving vehicle 2 cars ahead. (b) Scenario where the rear vehicle must brake for a braking vehicle 2 cars ahead.

In such a road system, there will be blocks of vehicles where the front vehicle is much greater than both $d_{b,b}$ and $d_{s,s}$ away from the nearest vehicle in front of it. Since it is at least this far, it can safely brake or swerve for any vehicle in front of it, and therefore any vehicle in front of it can be ignored. Because of this, these blocks can be considered in isolation, and if each block of vehicles is considered safe, all vehicles in the road system is considered safe. For any vehicle in a given block, denote its speed by v_1, and the speeds of the first and second vehicles in front of it (if they exist within the block) as v_2 and v_3, respectively. The longitudinal position of each vehicle as a function of time is denoted by $x_1(t)$, $x_2(t)$, and $x_3(t)$. A sufficient safe following distance for each vehicle is then

$$\begin{aligned}\hat{d}_{\text{long}} = \max(&d_{\text{b,s}}(v_1, v_2, \rho), d_{\text{b,s}}(v_1, v_2, \rho),\\ &d_{\text{s,s}}(v_1, v_3, 2\rho) - d_{\text{s,b}}(v_2, v_3, \rho), d_{\text{b,b}}(v_1, v_3, 2\rho) - d_{\text{s,b}}(v_2, v_3, \rho)).\end{aligned} \tag{5.43}$$

Theorem 6. *Equation 5.43 gives a longitudinal safe distance sufficient for universal safety when maintained by all vehicles.*

Proof. As mentioned earlier, each block of vehicles can be analyzed individually for safety, and if every block is safe, all vehicles are safe. The safety of any given block can be proved using an inductive argument across all of the vehicles, starting from the front of the block. The following is a proof sketch.

- For the base case, the safety of the first two vehicles is proven when following with at least $\hat{d}_{\text{long}}$.
- For the inductive step, it is assumed the i^{th} agent is following with at least $\hat{d}_{\text{long}}$ and is safe, and it is shown that if the $(i+1)^{\text{th}}$ agent follows with at least $\hat{d}_{\text{long}}$, then it is safe.

Base Case

The first vehicle at the front of the block is by definition at least $d_{b,b}$ and $d_{s,s}$ from any vehicle in front of it (if such an vehicle exists). As a result, any potential vehicle in front of the first can be safely avoided if necessary with either a brake or a swerve. This means that the first vehicle in the block is safe, and any potential vehicle in front of the first can be safely ignored by all vehicles in the block.

The second vehicle follows the first vehicle at $\hat{d}_{\text{long}}$. If the front vehicle brakes, the second vehicle is at least $d_{s,b}$ away from it, and can swerve to safety. If the front vehicle swerves, the second vehicle is at least $d_{b,s}$ away from it, and can brake safely. The second vehicle is therefore safe from the first vehicle, and is therefore safe.

Induction

Now, suppose the i^{th} vehicle is following with at least $\hat{d}_{\text{long}}$ of distance, and is safe from the vehicles in front of it. Denote the $(i+1)^{\text{th}}$ as vehicle 1, the i^{th} vehicle as vehicle 2, and the $(i-1)^{\text{th}}$ vehicle as vehicle 3. The distance between vehicle 1 and vehicle 2 is $\hat{d}_{\text{long}}$. If vehicle 2 brakes or swerves, vehicle 1 is at least $d_{s,b}$ and $d_{b,s}$ away from it, and is safe from vehicle 2 if it responds with a swerve or brake, respectively.

If vehicle 1 swerves in response to vehicle 2's brake, there are 2 cases to consider. The first case is if vehicle 3 brakes. Since vehicle 2 was assumed to be safe from vehicle 3, $x_2(t) \leq x_3(t), \forall t$. Combining this with the fact that $d_{s,b}$ is sufficient for vehicle 1 to swerve safely from vehicle 2, vehicle 1 must be safe from vehicle 3 if vehicle 3 brakes.

If vehicle 3 swerves, $d_{s,s}(v_1, v_3, 2\rho)$ is a sufficient distance for vehicle 1 to follow vehicle 3 to ensure safety. This case is illustrated in Figure 5.4a. The reaction delay is doubled to account for the reaction propagating through 2 vehicles instead of the usual one. Since vehicle 2 was assumed to be safe from vehicle 3, $d_{s,b}(v_2, v_3, \rho)$ is a lower bound on vehicle 2's following distance from vehicle 3. This means that in this case, $d_{\text{s,s}}(v_1, v_3, 2\rho) - d_{\text{s,b}}(v_2, v_3, \rho)$ is a sufficient following distance between vehicle 1 and 2 to guarantee safety.

If vehicle 1 brakes in response to vehicle 2's swerve, as before there are 2 cases to consider. The first case is if vehicle 3 is swerving. As before, since vehicle 2 was assumed to be safe from vehicle 3, $x_2(t) \leq x_3(t), \forall t$. Combining this with the fact that $d_{b,s}$ is sufficient for vehicle 1 to brake safely from vehicle 2's swerve, vehicle 1 must be safe from vehicle 3's swerve.

If vehicle 3 brakes, $d_{b,b}(v_1, v_3, 2\rho)$ is a sufficient distance for vehicle 1 to follow vehicle 3 to ensure safety. This case is illustrated in Figure 5.4b. Again, the reaction delay is doubled to account for propagation between two vehicles. Since vehicle 2 was assumed to be safe from vehicle 3, $d_{s,b}(v_2, v_3, \rho)$ is again a lower bound on vehicle 2's following distance. Thus, in this case, $d_{\text{b,b}}(v_1, v_3, 2\rho) - d_{\text{s,b}}(v_2, v_3, \rho)$ is a sufficient following distance between vehicle 1 and 2 to guarantee safety.

Since $\hat{d}_{\text{long}}$ is greater or equal to each of these following distances, vehicle 1 is safe, and thus the $(i+1)^{\text{th}}$ is safe. By induction, any block of vehicles is safe that maintains this following distance, and as a result, the entire system is safe. □

At high speeds, this new following distance can be used to allow for tighter following between agents. At low speeds, the agents can revert to the braking following distance used in RSS. A comparison between the RSS following distance and this new following distance across a range of speeds is shown in Figure 5.7.

5.3 Validation and Results

To validate the bicycle model assumptions, the first subsection checks the validity of the conservative upper bound on the required swerve distance by computing a lower bound. In the next subsection, a dynamic single-track vehicle model [37] is used to see if the previously computed swerve distances are reasonable conservative approximations. The lower bound is computed and compared to the upper bound distance, as well as the braking distance, in Section 5.3.1. In Section 5.3.2, the upper bound swerve clearance distance, as computed in Section 5.2.2, is compared to swerves from the dynamic single-track model.

5.3.1 Lower Bound Validation

The lower bound on the longitudinal swerve distance in this section is computed while satisfying the constraints of the particle model in Equations 3.23 and 3.24. The minimum a_x and maximum a_y values are set to be $-a_{\text{min,brake}}$ and $a^{\text{lat}}_{\text{min}}$, respectively, from the bicycle model. This ensures that any acceleration possible for the bicycle model is also possible for the particle model.

For a particle model, maximal lateral acceleration towards y_c as well as maximal longitudinal deceleration leads to lateral clearance in the shortest longitudinal distance $\bar{x}_c$ [99]. Thus, $\bar{x}_c \leq x_c$ for any other manoeuvre feasible for the particle model.

Finally, for computing the clearance, an inner approximation of the vehicle's chassis during rotation is used. To do so, the clearance distance uses the square inscribed on the circle of radius b_l centred on the centre of mass with side length $2d_i'$. This is shown in Figure 5.2b. Through this inner approximation, $d_i' \leq d'$ for any possible chassis rotation. This implies that anything the chassis can clear during the swerve will be cleared by the inner square. If x_f is used as in Section 5.2.2, a lower bound on the longitudinal safe distance, denoted by $\bar{d}_{\text{long}}$, is given by

$$\bar{d}_{\text{long}} = [v_r \rho + \frac{1}{2} a_{\text{max,accel}} \rho^2 + \bar{x}_c - x_f]_+ + d_i' + d. \tag{5.44}$$

Theorem 7. *Equation 5.44 gives a longitudinal safe distance necessary for safety when swerving for a braking lead vehicle.*

Proof. The clearance time and associated longitudinal distance at which point the particle model reaches y_c are given by

$$t_c = \sqrt{\frac{2y_c}{a_{\text{min}}^{\text{lat}}}}, \tag{5.45}$$

$$\bar{x}_c = vt_c - \frac{a_{\text{min,brake}} t_c^2}{2} + d_i'. \tag{5.46}$$

By the acceleration constraints imposed on the particle model, any feasible acceleration in the bicycle model is feasible for the particle model. In addition, the manoeuvre is optimal with respect to longitudinal distance travelled for the particle model. Both of these points imply that the $\bar{x}_c$ in Equation 5.46 is a lower bound on any feasible x_c for the bicycle model. Next, the inner approximation implies that for any manoeuvre, if the chassis can clear, the square with side length $2d_i'$ can clear as well, allowing a buffer of d_i' to be added.

If the initial longitudinal distance between the vehicles is denoted as x_2, then the distance between the swerving vehicle and the braking vehicle during the reaction delay is given by $x_2 - d_i' - d_r + v_f t - \frac{1}{2} a_{\text{max,brake}} t^2 - v_r t - \frac{1}{2} a_{\text{max}} t^2$. If the distance between the vehicles at the end of the reaction delay is denoted as x_ρ, then after the reaction delay the distance between the vehicles is given by $x_\rho + v_f t - \frac{1}{2} a_{\text{max,brake}} t^2 - v_{r,\rho} t + \frac{1}{2} a_{\text{min,brake}} t^2$. Since $-a_{\text{max,brake}} - a_{\text{max}} < 0$ and $-a_{\text{max,brake}} + a_{\text{min,brake}} < 0$, the distance between the swerving and braking vehicle is concave on both intervals. This implies that the minimum gap occurs at the boundaries of the time intervals $\{0, \rho, t_c\}$. Since the distance between the vehicles is differentiable everywhere, the time ρ is a critical point only if the derivative is zero. In this case, since the distance is concave before and after time ρ, the derivative is positive for $t < \rho$ and negative for $t > \rho$, implying the distance at time ρ is a local maximum. Taking everything together, assuming the vehicles are not already in collision at $t = 0$, this implies that Equation 5.44 is a lower bound on the longitudinal safe distance required for a swerve feasible for the bicycle model. □

A comparison between the lower bound and upper bound on the longitudinal distance travelled during a swerve, as well as the equivalent braking distance is shown in Figure 5.5. The plot is across a range of initial speeds.

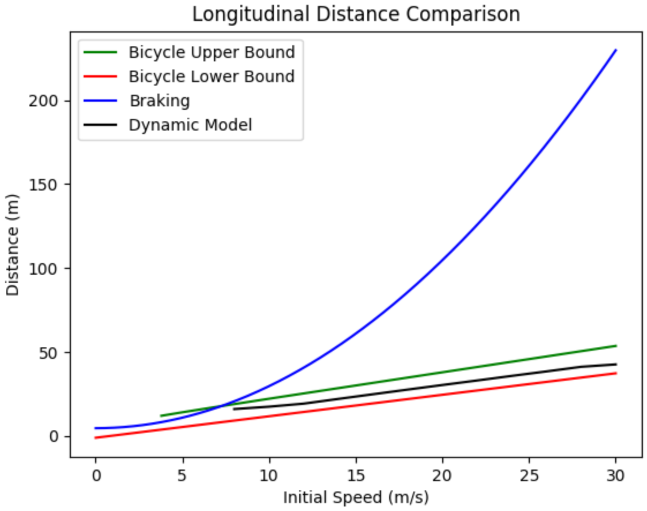


Figure 5.5: A comparison of the longitudinal distance travelled between swerve and brake manoeuvres, for varying initial velocities. The swerving distance required by the dynamic model is similar to the distance required by the bicycle model.

5.3.2 Dynamic Model Validation

Next, the kinematic approximation is validated by comparing the longitudinal swerve distance under a dynamic model to the distance computed in the preceding sections. This section focuses on the ability of the dynamic model to swerve, and not an associated controller, and as a result the manoeuvres are generated in open loop. However, doing a grid search over all possible control inputs to find the best swerves is impractical. Instead, the steering input is broken into 4 equal length intervals of time, and binary search is performed over steering rate magnitudes until the boundary conditions in Equation 5.5 are satisfied. In addition, linear search is performed over brake input and the total time of the manoeuvre, where the manoeuvre that minimizes the longitudinal swerve distance x_c is selected.

The parameters used in this validation are summarized in Table 5.1. The deceleration $a_{\text{min,brake}}$ was chosen to represent braking at the limit of comfort, and $a_{\text{max,brake}}$ was chosen to represent a hard, uncomfortable brake. The swerves generated for various initial speeds are illustrated in Figure 5.6.

Table 5.1: Parameters Table

m	1239 kg	l_f	1.19 m	l_r	1.37 m
I_{zz}	1752 $\text{kg} \cdot \text{m}^2$	e_{SP}	0.5 m	R	0.302 m
c_w	0.3	ρ_{drag}	1.25 $\frac{\text{kg}}{\text{m}^3}$	A	1.438
B_f	10.96	C_f	1.3	D_f	4560.4
E_f	-0.5	B_r	12.67	C_r	1.3
D_r	3947.81	E_r	-0.5	$a_{\text{max}}^{\text{lat}}$	4.0 $\frac{\text{m}}{\text{s}^2}$
$a_{\text{min}}^{\text{lat}}$	2.0 $\frac{\text{m}}{\text{s}^2}$	$a_{\text{min,brake}}$	2.0 $\frac{\text{m}}{\text{s}^2}$	$a_{\text{max,brake}}$	8.0 $\frac{\text{m}}{\text{s}^2}$
$a_{\text{max,accel}}$	2.0 $\frac{\text{m}}{\text{s}^2}$	μ	0.1 m	ρ	0.1 s
α	3.7 m	d_r	2.3 m	d_f	2.4 m
b_r	0.9 m	b_l	0.9 m	δ_{max}	$\frac{\pi}{6}$

Using these computed swerves, the clearance distance y_c is then computed as before and is used to find the longitudinal swerve distance travelled x_c that occurs at time t_c. Substituting this value in at Equations 5.25 and 5.31 then gives the required longitudinal safe distance for the dynamic model. For the range of initial vehicle speeds where swerving is more efficient than braking, the longitudinal safe distances required for the dynamic model are plotted and compared to those computed in Section 5.2 in Figure 5.5.

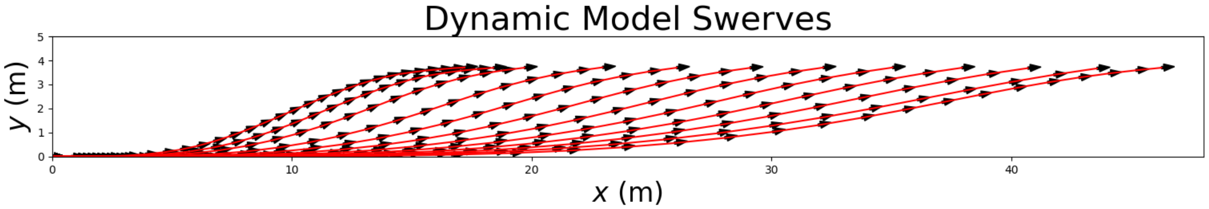


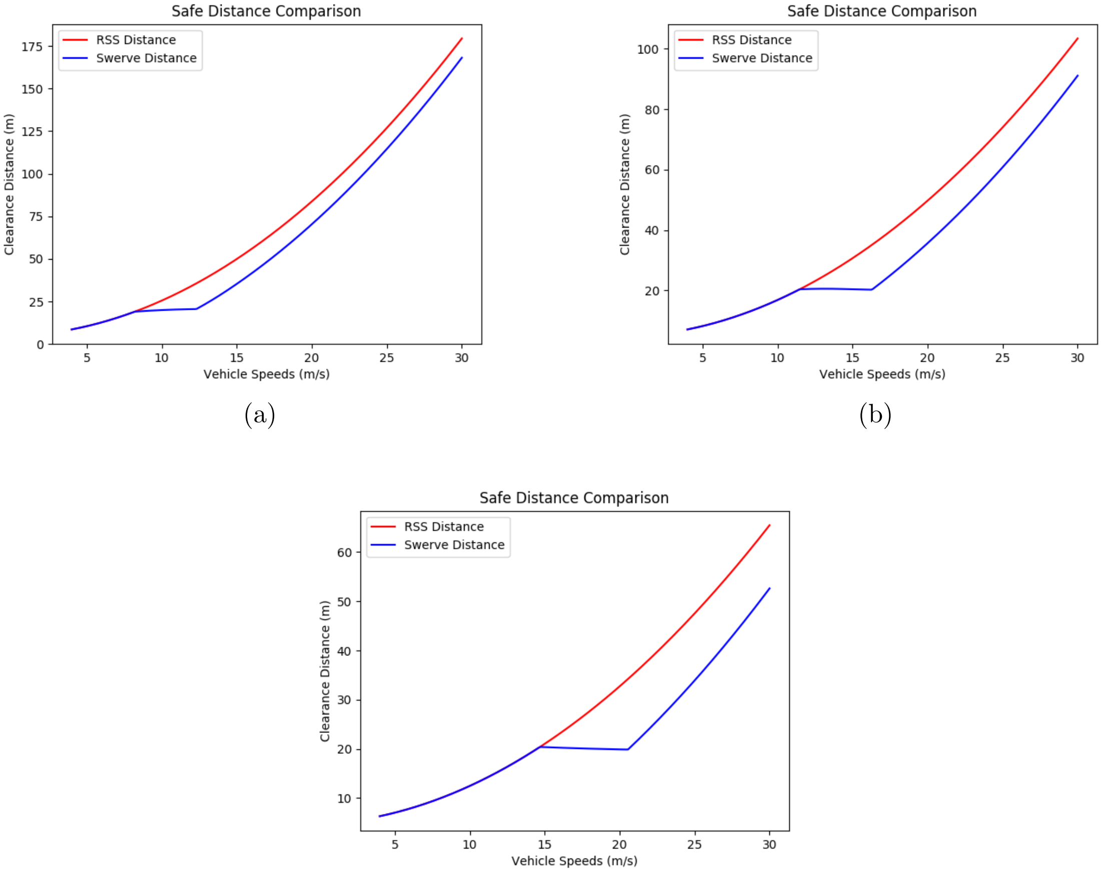
Figure 5.6: The swerve manoeuvres generated according to the dynamic model. Each swerve is for a different initial speed in the interval [10, 30] $\frac{m}{s}$. The arrows denote the heading of the vehicle.

5.3.3 Simulation Results

In Figure 5.5, the lower bound is within 30.3-61.9% error of the upper bound, which corresponds to a range of 8.8-16.3m of additional distance. This gives a range of clearance values required for a swerving bicycle to clear a lead vehicle.

This plot also illustrates the advantage of swerves; across a initial speeds ranging from 5-30 $\frac{\text{m}}{\text{s}}$, the swerves reach safety using up to 63.5m less of longitudinal distance travelled than braking does. In addition, the longitudinal swerve distance required by the dynamic single-track model is within 15.6-24.0% error of the upper bound distance, and is always above the lower bound distance. This shows that the kinematic approximation can accurately bound the swerve distance required by the dynamic single-track model, across initial speeds from 8-30 $\frac{\text{m}}{\text{s}}$. This verifies that the kinematic approximation can be used as a closed form and efficient method for computing longitudinal safe distances as described in the preceding sections.

Figure 5.7 compares the safe distance required when using swerves, $\hat{d}_{\text{long}}$, as compared to the standard d_{long} required in the RSS framework, when each vehicle is moving along at the same speed. For low speeds, the required braking distance is lower, which is in line with what is expected from Figure 5.5. However, at higher speeds it is clear that the required following distance to maintain safety is reduced. By allowing swerves, the following distance can be reduced by up to 40%, or 14.1 - 14.9m, even when allowing for large magnitudes of $a_{\text{min,brake}}$, and as a result, more aggressive brake responses. The main difference that increasing $a_{\text{min,brake}}$ has is increasing the cutoff velocity at which the swerving following distance falls below the braking following distance.



(a)

(b)

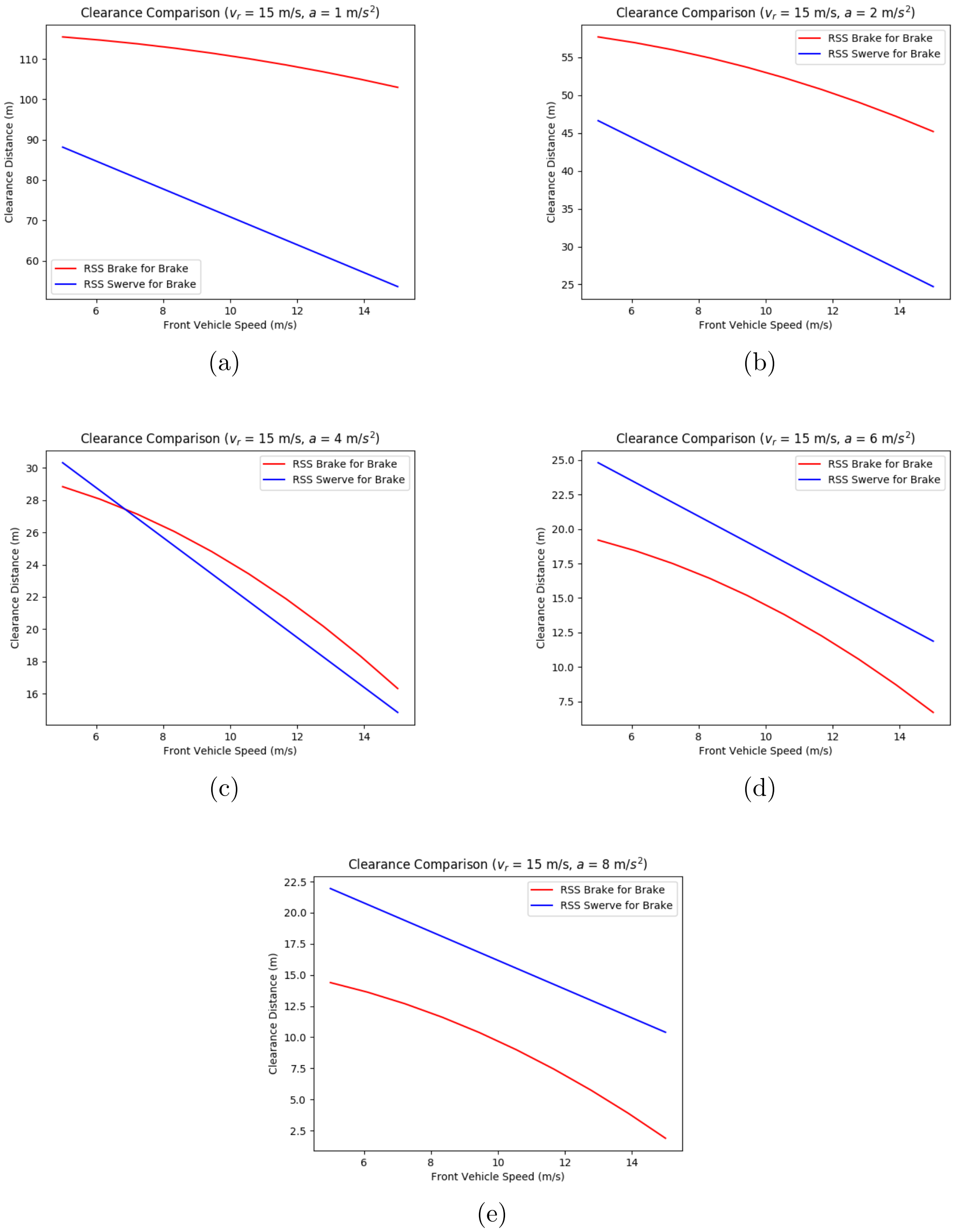
(c)

Figure 5.7: (a) Plot of the longitudinal safe distance across a range of speeds, as compared to the standard RSS braking distance, with $a_{\min,brake} = 2\frac{m}{s^2}$*. (b) Same as (a), but with* $a_{\min,brake} = 3\frac{m}{s^2}$*. (c) Same as (a), but with* $a_{\min,brake} = 4\frac{m}{s^2}$*.*

Along the same vein, it is also interesting to compare the safe distance required for braking for a braking vehicle to the safe distance required for swerving for a braking vehicle as the $a_{\text{min,brake}}$ and $a^{\text{lat}}_{\text{min}}$ parameters are varied. The maximum deceleration is set to $a_{\text{max,brake}} = 8\frac{\text{m}}{\text{s}^2}$, and the other acceleration parameters $a_{\text{min,brake}}$ and $a^{\text{lat}}_{\text{min}}$ are varied between $[1, 8]\frac{\text{m}}{\text{s}^2}$ while holding both equal to one another. Comparing the safe longitudinal distances when braking or swerving in this manner gives a visualization for when braking or swerving is advantageous, depending on the magnitude of acceleration allowed. The rear vehicle initial speed v_r is set to 15 and $20\frac{\text{m}}{\text{s}}$, and the braking front vehicle initial speed v_f is swept from 5 to $v_r\frac{\text{m}}{\text{s}}$.

In Figure 5.8, when the rear vehicle is travelling at $15\frac{\text{m}}{\text{s}}$, for low acceleration rates, the swerve manoeuvres require less longitudinal safe distance than braking. For acceleration values higher than $4\frac{\text{m}}{\text{s}^2}$, braking requires less following distance. This trend implies that braking scales better with increased acceleration limits than swerving does. However, swerving allows for more efficient manoeuvres than braking when acceleration is limited. This can be useful in certain situations, such as when there is a tailgating vehicle behind the autonomous vehicle. Different vehicles may also have different braking capabilities, which may prevent high acceleration rates from being achievable.

In Figure 5.9, a similar trend emerges; at lower accelerations swerving requires less following distance than braking. However, of note is that swerving allows the autonomous vehicle to reach safety earlier than braking does even when the maximum acceleration is set to $6\frac{\text{m}}{\text{s}^2}$. Even when the maximum acceleration is set to $8\frac{\text{m}}{\text{s}^2}$, for a range of lead vehicle initial speeds, swerving and braking require roughly the same longitudinal safe distance. This implies that as the initial speed of the autonomous vehicle increases, the effectiveness of swerves in reaching safety improves. This gives additional evidence that swerves can be useful in high speed situations, such as highway driving, particularly when other vehicles cut off the autonomous vehicle.



(a)

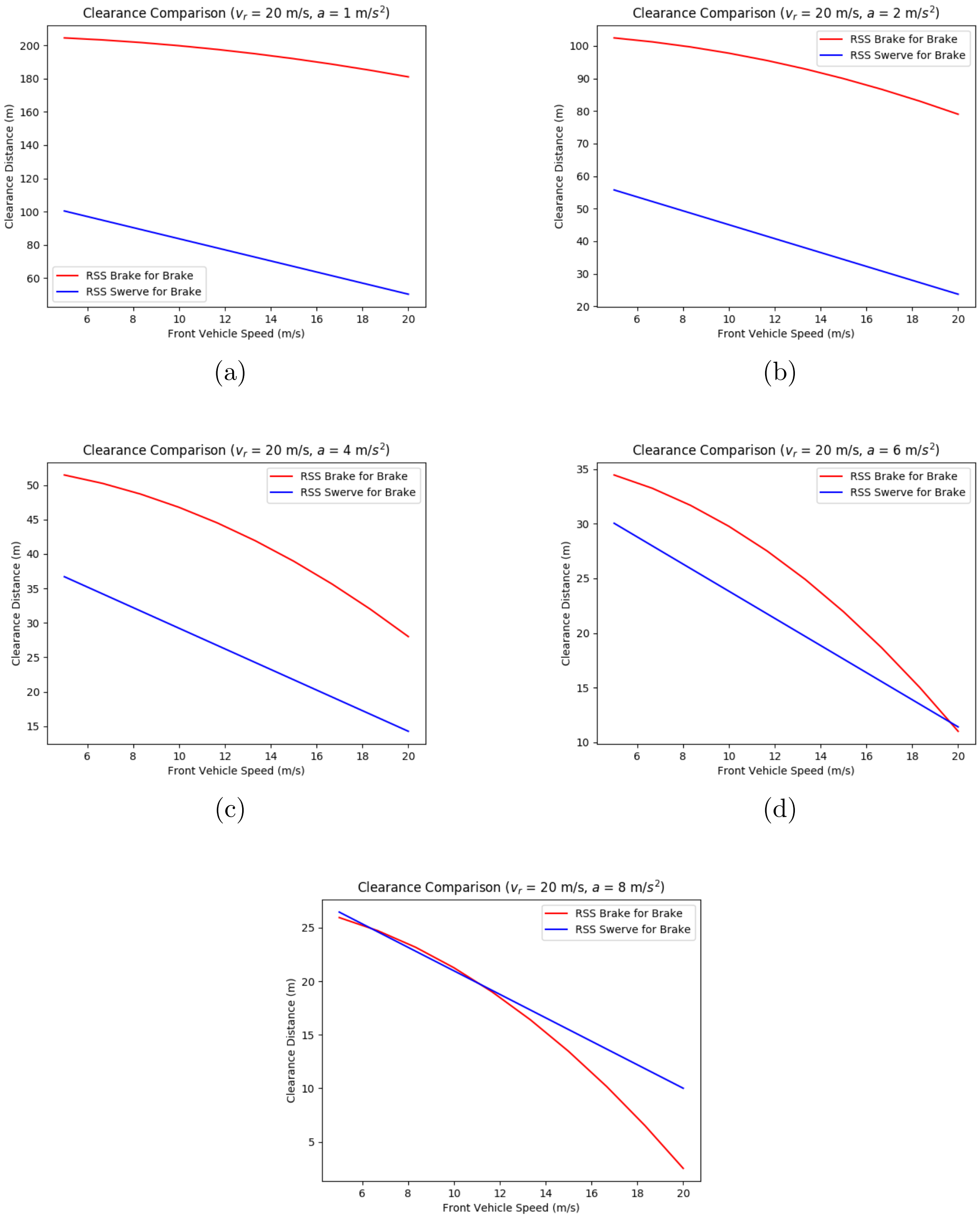
(b)

(c)

(d)

(e)

Figure 5.8: A comparison of different safe longitudinal distances for when $v_r = 15\frac{m}{s}$*. The maximum longitudinal and lateral accelerations is set to (a)* $1\frac{m}{s^2}$*, (b)* $2\frac{m}{s^2}$*, (c)* $4\frac{m}{s^2}$*, (d)* $6\frac{m}{s^2}$*, (e)* $8\frac{m}{s^2}$*.*



(a)

(b)

(c)

(d)

(e)

Figure 5.9: A comparison of different safe longitudinal distances for when $v_r = 20\frac{m}{s}$. The maximum longitudinal and lateral accelerations is set to (a) $1\frac{m}{s^2}$, (b) $2\frac{m}{s^2}$, (c) $4\frac{m}{s^2}$, (d) $6\frac{m}{s^2}$, (e) $8\frac{m}{s^2}$.

Chapter 6

Conclusions and Future Work

This thesis discussed and analyzed two different problems related to motion planning for autonomous driving. The first problem involved optimizing a lattice planner control set for a particular driving task, with the goal of improving planning time as well as capturing the driving style present in a given dataset of trajectories for said driving task. The proposed approach involved selecting control actions that allowed a lattice planner to plan paths similar to those in the dataset. This measurement of similarity was computed using the modified Fréchet distance. We evaluated this method using both real data from roundabout driving as well as synthetic datasets.

The second problem discussed in this thesis extended the RSS framework by introducing safe swerve manoeuvres. This involved developing a method for computing the longitudinal distance required for safety during a swerve manoeuvre under the RSS framework, and compared it to the standard braking distance used previously. The safety of these manoeuvres was proven under a set of reasonable assumptions about responsible behaviour, while incorporating the original assumptions in the RSS framework. This extended framework can be used to reduce longitudinal clearance, to react more quickly to dangerous situations, or to compute the clearance required for proactive swerve manoeuvres, such as lane changes.

6.1 Future Work for Learning a Lattice Planner Control Set

To improve this approach, there are two promising avenues of research. The first is to replace the search for the closest path in the lattice with a greedy search over the motion primitives in the control set instead. The analysis in this thesis shows that the closest path search algorithm tends to $O(K|C|)$ for a tight greedy bound, but in practice the algorithm tends to slow down significantly as the control set grows larger. To improve runtime, one option may be to use the path generated through greedy search for the closest path instead of the optimal closest path. Then the runtime will be, even in the worst case, $O(K|C|)$. The value of d_L for the greedy path has been empirically observed to be close to the optimal found during closest path search, so it is likely that the control sets generated by using the greedy path instead are not significantly degraded.

The next improvement could come from the use of inverse reinforcement learning (IRL) in the learning process. Currently, the dense control set is populated based on spirals to uniformly distributed points, with various initial and final headings. Instead, one may first generate the motion primitives directly from the dataset using IRL. This dense set of primitives could then be refined according to the learning process already outlined in this work. This would allow for stronger results in terms of matching the driving style present in the dataset, as IRL is a powerful tool for doing so.

In terms of other related problems, it would be interesting to combine learning the structure of a lattice planner with learning the lattice planner's search heuristic, to see if lattice planner performance can be improved even further for specific applications. Extending this algorithm to handle trajectories rather than paths is also a potential improvement. This would require a way to scale the algorithm efficiently as the dimensionality of the problem increases. In a separate vein, formulating the optimization objective as a submodular function, similar to what was done by Dey et al. [23], could also yield interesting theoretical guarantees on learning performance.

6.2 Future Work for the RSS Framework

To further improve the RSS framework, one option would be to extend the inclusion of swerve manoeuvres to more general cases. An example of this would be to generalize the swerve manoeuvre to arbitrary Frenet frames, as opposed to straight lines. One could also compute bounds on the error from using a straight line approximation to the Frenet frame. Further experimental work of the RSS framework and its extensions, through on-car testing or scenario simulation, would also be beneficial, to determine the strengths and weaknesses of the framework in real driving situations.

An interesting related problem would be to try to extend the RSS framework to consider interactive agents who react to the autonomous vehicle. This would allow for safe lead distances to be computed for more complex driving tasks, such as lane changes, lane merging, or cut-in situations. Since computing agent reactions are tightly linked to behaviour prediction, extending RSS in this manner will probably require the framework to shift to be probabilistic, rather than having fixed safe distances. This is due to the probabilistic nature of behaviour prediction.

In its current form, the RSS framework addresses the instantaneous safe distance between different agents, but ignores the transient effects present as a system of agents evolves with time. While there is a prescription to brake when the safe distance is violated, the braking response as well as the stability or convergence of agents to a given speed are not explored in detail. As a result, an interesting avenue to explore would be to consider control laws that result in stable, safe following distances, with convergence to a constant following distance, as opposed to oscillations.

The analysis in this thesis has ignored the effects of occlusions when analyzing chains of safe agents. Instead, safe distances are computed assuming reaction delays propagate uniformly between consecutive agents. A more robust definition of safety would account for the time it takes for an occluded object to be revealed to the reacting vehicle.

6.3 Conclusion

In the field of autonomous driving, two important areas of research are how to leverage data to improve motion planning, as well as how to ensure safety in a multi-agent environment.

This thesis has briefly explored these two exciting areas of research, and has laid the groundwork for future endeavours. While it is clear that this thesis has only scratched the surface of these topics, continued work on these problems will certainly help make autonomous driving a reality.

References

[1] Y. Abeysirigoonawardena, F. Shkurti, and G. Dudek. Generating adversarial driving scenarios in high-fidelity simulators. *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[2] F. Altche and A. De La Fortelle. An LSTM network for highway trajectory prediction. *IEEE ITSC*, 2017.

[3] M. Althoff, D. Althoff, D. Wollherr, and M. Buss. Safety verification of autonomous vehicles for coordinated evasive maneuvers. *2010 IEEE Intelligent Vehicles Symposium*, 2010.

[4] M. Althoff and J. M. Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, 2014.

[5] M. Althoff, M. Koschi, and S. Manzinger. Commonroad: Composable benchmarks for motion planning on roads. *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[6] J. Van Den Berg, M. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. *2008 IEEE International Conference on Robotics and Automation*, 2008.

[7] P. Berman, A. Bhattacharyya, K. Makarychev, S. Raskhodnikova, and G. Yaroslavtsev. Approximation algorithms for spanner problems and directed steiner forest. *Information and Computation*, 222:93–107, 2013.

[8] M. Bhardwaj, S. Choudhury, and S. Scherer. Learning heuristic search via imitation. *CoRR*, abs/1707.03034, 2017.

[9] C. Bianco, G. Lo, and A. Piazzi. Optimal trajectory planning with quintic g/sup 2/-splines. *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511)*, 2000.

[10] A. Blake, A. Bordallo, M. Hawasly, S. Penkov, S. Ramamoorthy, and A. Silva. Efficient computation of collision probabilities for safe motion planning. *CoRR*, abs/1804.05384, 2018.

[11] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer. Scalable decision making with sensor occlusions for autonomous driving. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[12] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes. Safe driving envelopes for path tracking in autonomous vehicles. *Control Engineering Practice*, 61:307–316, 2017.

[13] D. Chen, A. Driemel, L. Guibas J., A. Nguyen, and C. Wenk. Approximate map matching with respect to the Fréchet distance. *Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 75–83, 2011.

[14] J. Chen, R. Wang, L. Liu, and J. Song. Clustering of trajectories based on Hausdorff distance. *International Conference on Electronics, Communications and Control (ICECC)*, 2011.

[15] J. Chen, W. Zhan, and M. Tomizuka. Constrained iterative lqr for on-road autonomous driving motion planning. *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[16] O. Cheong, H. Haverkort, and M. Lee. Computing a minimum-dilation spanning tree is np-hard. *Computational Geometry*, 41(3):188–205, 2008.

[17] I. Chevyrev and A. Kormilitzin. A Primer on the Signature Method in Machine Learning. *ArXiv e-prints*, March 2016.

[18] S. Choudhury, M. Bhardwaj, S. Arora, A. Kapoor, G. Ranade, S. Scherer, and D. Dey. Data-driven Planning via Imitation Learning. *ArXiv e-prints*, November 2017.

[19] B. Cserna, W. J. Doyle, T. Gu, and W. Ruml. Safe temporal planning for urban driving. *AAAI 2019 Workshop*, 2019.

[20] A. Dragan D., K. C. T. Lee, and S. S. Srinivasa. Legibility and predictability of robot motion. *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2013.

[21] A. Dragan D., K. Muelling, A. J. Bagnell, and S. S. Srinivasa. Movement primitives via optimization. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[22] J. David, R. Valencia, R. Philippsen, P. Bosshard, and K. Iagnemma. Gradient based path optimization method for autonomous driving. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[23] D. Dey, T. Y. Liu, B. Sofman, and D. Bagnell. Efficient optimization of control libraries. *Robotics: Science and Systems VIII*, Jan 2011.

[24] P. Dingle and L. Guzzella. Optimal emergency maneuvers on highways for passenger vehicles with two- and four-wheel active steering. *Proceedings of the 2010 American Control Conference*, 2010.

[25] A. D. Dragan, G. J. Gordon, and S. S. Srinivasa. Learning from experience in manipulation planning: Setting the right goals. *Springer Tracts in Advanced Robotics Robotics Research*, pages 309–326, 2016.

[26] T. Eiter and H. Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Technische Universitat Wien, 1994.

[27] J. Fisac F., A. Akametalu K., M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 2018.

[28] P. Felzenszwalb F. and D. P. Huttenlocher. Distance transforms of sampled functions. *Theory of Computing*, 8, 2012.

[29] Y. Chen Fan, M. Everett, M. Liu, and J. P. How. Socially aware motion planning with deep reinforcement learning. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[30] A. Faust, O. Ramirez, M. Fiser, K. Oslund, A. Francis, J. Davidson, and L. Tapia. Prm-rl: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[31] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Shankar, and A. D. Dragan. Hierarchical game-theoretic planning for autonomous vehicles. *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[32] T. Fraichard and A. Scheuer. From reeds and shepps to continuous-curvature paths. *IEEE Transactions on Robotics*, 20(6):1025–1035, 2004.

[33] C. Fulgenzi, A. Spalanzani, and C. Laugier. Dynamic obstacle avoidance in uncertain environment combining pvos and occupancy grid. *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007.

[34] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 99*, 1999.

[35] E. Galceran, A. Cunningham, R. Eustice, and E. Olso. Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction. *Robotics: Science and Systems XI*, 2015.

[36] M. Van Gennip. Vehicle dynamic modelling and parameter identification for an autonomous vehicle. Master's thesis, University of Waterloo, 2018.

[37] M. Gerdts. Solving mixed-integer optimal control problems by branch & bound: a case study from automobile test-driving with gear shift. *Optimal Control Applications and Methods*, 26(1):1–18, 2005.

[38] F. Giovannini, G. Savino, M. Pierini, and N. Baldanzini. Analysis of the minimum swerving distance for the development of a motorcycle autonomous braking system. *Accident Analysis and Prevention*, 59:170–184, 2013.

[39] D. Godbolen., V. Hagenmeyer, R. Sengupta, and D. Swaroop. Design of emergency manoeuvres for automated highway system: obstacle avoidance problem. *Proceedings of the 36th IEEE Conference on Decision and Control*, 1997.

[40] D. S. Gonzalez, O. Erkent, V. Romero-Cano, J. Dibangoye, and C. Laugier. Modeling driver behavior from demonstrations in dynamic environments using spatiotemporal lattices. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[41] T. Gu. *Improved Trajectory Planning for On-Road Self-Driving Vehicles Via Combined Graph O. Search and Topology Analysis.* PhD thesis, Carnegie Mellon University, 2017.

[42] T. Gu, J. M. Dolan, and J. Lee-Woo. Automated tactical maneuver r. discovery and trajectory planning for autonomous driving. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[43] M. Gmez, R. V. Gonzlez, T. Martnez-Marn, D. Meziat, and S. Snchez. Optimal motion planning by reinforcement learning in autonomous mobile vehicles. *Robotica*, 30(02):159–170, 2011.

[44] J. Hardy and M. Campbell. Contingency planning over probabilistic hybrid obstacle predictions for autonomous road vehicles. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[45] P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[46] K. Hauser. Continuous pseudoinversion of a multivariate function : Application to global redundancy resolution. *The International Workshop on the Algorithmic Foundations of Robotics*, 2016.

[47] R. M. Holladay and S. S. Srinivasa. Distance metrics and algorithms for task space path optimization. *IEEE/RSJ IROS*, 2016.

[48] B. K. P. Horn. The curve of least energy. *ACM Transactions on Mathematical Software*, 9(4):441–460, Jan 1983.

[49] T. Howard, M. Pivtoraiko, R. A. Knepper, and A. Kelly. Model-predictive motion planning: Several key developments for autonomous mobile robots. *IEEE Robotics Automation Magazine*, 21(1):64–73, March 2014.

[50] R. De Iaco, S. L. Smith, and K. Czarnecki. Learning a lattice planner control set for autonomous vehicles. *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[51] B. Ichter, J. Harrison, and M. Pavone. Learning sampling distributions for robot motion planning. *arXiv preprint arXiv:1709.05448*, Sep 2017.

[52] H. Jula, E. B. Kosmatopoulos, and P. A. Ioannou. Collision avoidance analysis for lane changing and merging. *IEEE Transactions on Vehicular Technology*, 49(6):2295–2308, 2000.

[53] P. Agarwal K., R. Avraham Ben, H. Kaplan, and M. Sharir. Computing the discrete frchet distance in subquadratic time. *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 156–167, Jun 2013.

[54] N. Kaempchen, B. Schiele, and K. Dietmayer. Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):678–687, 2009.

[55] N. Kalra and S. M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reiliability? Technical Report RR-1478-RC, RAND Coroporation, 2016.

[56] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *CoRR*, abs/1105.1186, 2011.

[57] A. Kelly and B. Nagy. Reactive nonholonomic trajectory generation via parametric optimal control. *The International Journal of Robotics Research*, 22(7):583–601, Jan 2003.

[58] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015.

[59] Y. Kuwata, G. Fiorea., J. Teo, E. Frazzoli, and J. P. How. Motion planning for urban driving using rrt. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[60] S. M. LaValle. *Planning algorithms*. Cambridge University Press, 2014.

[61] J. Lee, J. Han, and K. Whang. Trajectory clustering: A partition-and-group framework. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data - SIGMOD 07*, 2007.

[62] K. Leung, E. Schmerling, M. Chen, J. Talbot, J. C. Gerdes, and M. Pavone. On infusing reachability-based safety assurance within probabilistic planning frameworks for human-robot vehicle interactions. *2018 International Symposium on Experimental Robotics*, 2018.

[63] L. Li, X. Peng, F. Wang, D. Cao, and L. Li. A situation-aware collision avoidance strategy for car-following. *IEEE/CAA Journal of Automatica Sinica*, 5(5):1012–1016, 2018.

[64] A. De Luca, G. Oriolo, and C. Samson. Feedback control of a nonholonomic car-like robot. *Lecture Notes in Control and Information Sciences Robot Motion Planning and Control*, pages 171–253, 1998.

[65] T. Howard M., C. Green J., A. Kelly, and D. Ferguson. State space sampling of feasible motions for high-performance mobile robot navigation in complex environments. *Journal of Field Robotics*, 25(6-7):325–345, 2008.

[66] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng. Efficient sampling-based motion planning for on-road autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):1961–1976, 2015.

[67] C. C. Macadam. Understanding and modeling the human driver. *Vehicle System Dynamics*, 40(1-3):101–134, Jan 2003.

[68] M. Mcnaughton, C. Urmson, J. M. Dolan, and J. Lee-Woo. Motion planning for autonomous driving with a conformal spatiotemporal lattice. *IEEE ICRA*, 2011.

[69] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, 2005.

[70] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT Press, 2012.

[71] M. Naumann, H. Knigshof, and C. Stiller. Provably safe and smooth lane changes in mixed traffic. *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2019.

[72] M. Naumann, M. Lauer, and C. Stiller. Generating s. comfortable and comprehensible trajectories for automated vehicles in mixed traffic. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[73] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. *Autonomous Robots*, 39(4):555–571, 2015.

[74] G. Oriolo, M. Ottavi, and M. Vendittelli. Probabilistic motion planning for redundant robots along given end-effector paths. *IEEE/RSJ IROS*, 2002.

[75] H. B. Pacejka and E. Bakker. The magic formula tyre model. *Vehicle System Dynamics*, 21:1–18, 1992.

[76] B. Paden, M. Cap, S. Yong Zheng, D. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, 2016.

[77] B. Paden, V. Varricchio, and E. Frazzoli. Verification and synthesis of admissible heuristics for kinodynamic motion planning. *IEEE Robotics and Automation Letters*, 2(2):648–655, 2017.

[78] C. Pek, P. Zahn, and M. Althoff. Verifying the safety of lane change maneuvers of self-driving vehicles based on formalized traffic rules. *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[79] A. Piazzi, C. Bianco, and G. Lo. Quintic g/sup 2/-splines for trajectory planning of autonomous vehicles. *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511)*, 2000.

[80] J. Pimentel, J. Bastiaan, and M. Zadeh. Numerical evaluation of the safety of self-driving vehicles: Functionality involving vehicle detection. *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 2018.

[81] M. Pivtoraiko and A. Kelly. Generating near minimal spanning control sets for constrained motion planning in discrete state spaces. *IEEE/RSJ IROS*, 2005.

[82] M. Pivtoraiko and A. Kelly. Kinodynamic motion planning with state lattice motion primitives. *IEEE/RSJ IROS*, 2011.

[83] M. Pivtoraiko, R. A. Knepper, and A. Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3):308–333, 2009.

[84] M. Pivtoraiko, I. A. D. Nesnas, and A. Kelly. Autonomous robot navigation using advanced motion primitives. *IEEE Aerospace Conference*, 2009.

[85] M. N. Pivtoraiko. *Differentially constrained motion planning with state lattice motion primitives.* PhD thesis, Carnegie Mellon University, 2012.

[86] P. Polack, F. Altche, B. Dandrea-Novel, and A. De La Fortelle. The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? *IEEE Intelligent Vehicles Symposium*, 2017.

[87] M. Pollack. Letter to the editor-the maximum capacity through a network. *Operations Research*, 8(5):733–736, 1960.

[88] X. Qian, A. De La Fortelle, and F. Moutarde. A hierarchical model predictive control framework for on-road formation control of autonomous vehicles. *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016.

[89] Y. Rasekhipour. *Prioritized Obstacle Avoidance in Motion Planning of Autonomous Vehicles*. PhD thesis, University of Waterloo, 2017.

[90] N. Ratliff, M. Zucker, A. J. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. *2009 IEEE International Conference on Robotics and Automation*, 2009.

[91] E. Rehder, J. Quehl, and C. Stiller. Driving like a human: Imitation learning for path planning use convolutional neural networks. *IEEE/RSJ International Conference on Robotics and Automation Workshops*, 2017.

[92] M. Rufli, D. Ferguson, and R. Siegwart. Smooth path planning in constrained environments. *2009 IEEE International Conference on Robotics and Automation*, 2009.

[93] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone. Multimodal probabilistic model-based planning for human-robot interaction. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[94] C. Schmidt, F. Oechsle, and W. Branz. Research on trajectory planning in emergency situations with multiple objects. *2006 IEEE Intelligent Transportation Systems Conference*, 2006.

[95] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel. Finding locally c. optimal-free trajectories with sequential convex optimization. *Robotics: Science and Systems IX*, 2013.

[96] S. Shalev-Shwartz, S. Shammah, and A. Shashua. On a formal model of safe and scalable self-driving cars. *CoRR*, abs/1708.06374, 2017.

[97] Z. Shiller and S. Sundar. Emergency maneuvers for ahs vehicles. *SAE Technical Paper Series*, 1995.

[98] Z. Shiller and S. Sundar. Emergency maneuvers of autonomous vehicles. *IFAC Proceedings Volumes*, 29(1):8089–8094, 1996.

[99] Z. Shiller and S. Sundar. Emergency lane-change maneuvers of autonomous vehicles. *Journal of Dynamic Systems, Measurement, and Control*, 120(1):37, 1998.

[100] Z. Shiller and S. Sundar. Optimal emergency maneuvers of automated vehicles. *eScholarship, University of California*, Oct 2005.

[101] J. M. Snider. Automatic steering methods for autonomous automobile path tracking. Master's thesis, Carnegie Mellon University, 2009.

[102] A. Stentz. Optimal and efficient path planning for partially-known environments. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 1994.

[103] S. Su, K. Muelling, J. Dolan, P. Palanisamy, and P. Mudalige. Learning vehicle surrounding-aware lane-changing behavior from observed trajectories. *IEEE IV Symposium*, 2018.

[104] E. Thorn, S. Kimmel, and M. Chaka. A framework for automated driving system testable cases and scenarios. Technical Report DOT HS 812 623, Virgina Tech Transportation Institute, Sep 2018.

[105] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y. Seo-Woo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson. *Autonomous Driving in Urban Environments: Boss and the Urban Challenge*, pages 1–59. Springer Berlin Heidelberg, H. Berlin, 2009.

[106] S. L. Waslander, S. Fischmeister, and K. Czarnecki. From zero to autonomous in six months: Creation of an autonomous driving demonstrator for ces 2017. 2016.

[107] C. Wenk. *Shape Matching in Higher Dimensions.* PhD thesis, FU Berlin, 2003.

[108] C. Wenk, R. Salas, and D. Pfoser. Addressing the need for map-matching speed: Localizing global curve-matching algorithms. *18th International Conference on Scientific and Statistical Database Management*, 2006.

[109] M. Werling, J. Ziegler, Kammel Soren, and S. Thrun. Optimal trajectory generation for dynamic street scenarios in a frenet frame. *2010 IEEE International Conference on Robotics and Automation*, 2010.

[110] J. Xie, A. R. Hilal, and D. Kulic. Driving maneuver classification: A comparison of feature extraction methods. *IEEE Sensors Journal*, 18(12):4777–4784, 2018.

[111] H. Xu, Y. Zhou, W. Lin, and H. Zha. Unsupervised trajectory clustering via adaptive multi-kernel-based shrinkage. *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[112] W. Xu, J. Pan, J. Wei, and J. M. Dolan. Motion planning under uncertainty for on-road autonomous driving. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[113] W. Xu, J. Wei, J. M. Dolan, H. Zhao, and H. Zha. A real-time motion planner with trajectory optimization for autonomous vehicles. *2012 IEEE International Conference on Robotics and Automation*, 2012.

[114] W. Yang, T. Lyons, H. Ni, C. Schmid, L. Jin, and J. Chang. Leveraging the path signature for skeleton-based human action recognition. *CoRR*, abs/1707.03993, 2017.

[115] G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang. A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review*, 47:123–144, Mar 2016.

[116] T. Zhang, G. Kahn, S. Levine, and P. Abbeel. Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search. *IEEE ICRA*, 2016.

[117] Y. Zhang, H. Chen, S. L. Waslander, J. Gong, G. Xiong, T. Yang, and K. Liu. Hybrid trajectory planning for autonomous driving in highly constrained environments. *IEEE Access*, 6:32800–32819, 2018.

[118] Y. Zhang, H. Chen, S. L. Waslander, T. Yang, S. Zhang, G. Xiong, and K. Liu. Toward a more f. complete, and safer speed planning for autonomous driving via convex optimization. *Sensors*, 2018.

[119] Y. Zhang, A. Dakibay, J. Lee, and S. Waslander. Continuous optimization based path planning for autonomous vehicles. 2016.

[120] A. Zhou and A. D. Dragan. Cost Functions for Robot Motion Style. *ArXiv e-prints*, August 2018.

[121] J. Ziegler and C. Stiller. Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. *IEEE/RSJ IROS*, 2009.

[122] M. Zucker, N. Ratliff, A. Dragan D., M. Pivtoraiko, M. Klingensmith, C. Dellin M., A. J. Bagnell, and S. S. Srinivasa. Chomp: Covariant hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, 32(9-10):1164–1193, 2013.

APPENDICES

Appendix A

Learned Lattice Planner Control Set Practical Considerations

Arc Length Relaxation. Since the lattice control actions connect vertices in the lattice graph, a realistic application of this method would require a small line segment length δ, which would in turn increase the size of K required in each path matching calculation. To remedy this, the requirement that each control action has an arc length that is integer-divisible by δ is relaxed. This potentially results in a leftover portion of each control action that would be left out of the closest path calculation. This is overcome by checking if the leftover portion of the control action is greater than or equal to half of δ. If it is, then it is treated as a full line segment for d computation. Otherwise, it is ignored. In practice, using a δ that is a $\frac{1}{4}$ of $\min(\Delta x, \Delta y)$ allows for good results.

Optimization Initialization. The learned control set is initialized with a single short, straight action for each possible initial direction, to ensure that the closest path algorithm can make forward progress when it encounters a point with any particular heading.