Motion Planning for Non-holonomic Autonomous Vehicles in Parking Spaces

An Optimal Control Problem Approach

R. Cirera Rocosa

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

Motion Planning for Non-holonomic Autonomous Vehicles in Parking Spaces

An Optimal Control Problem Approach

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

R. Cirera Rocosa

August 9, 2018

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Delft University of Technology
Department of
Delft Center for Systems and Control (dcsc)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis entitled

Motion Planning for Non-holonomic Autonomous Vehicles in Parking Spaces

by

R. Cirera Rocosa

in partial fulfillment of the requirements for the degree of

Master of Science Systems and Control

Dated: August 9, 2018

Supervisor(s):

dr. M. Alirezaei

dr. P. Mohajerin Esfahani

Reader(s):

dr. M. Alirezaei

dr. P. Mohajerin Esfahani

prof.dr.ir. J. Hellendoorn

Abstract

This document is the final report of an MSc. thesis project done at the Delft Center for Systems and Control department of the faculty of Mechanical, Maritime and Materials Engineering at the Delft University of Technology. The thesis is on the topic of motion planning for non-holonomic autonomous vehicles in parking spaces. First, the thesis thesis proposal statement and the thesis goals are introduced. Then, the approach taken to fulfil the thesis goals is described. Following that, the steps taken to follow said approach are explained. Next, the results of the thesis work are presented. Finally, the conclusion of the thesis work is given, and recommendations are made for improvements and future work.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisors, dr. M. Alirezaei and dr. P. Mohajerin Esfahani, for their assistance throughout the different parts of the MSc. thesis work. Their insight and feedback was invaluable, as were their patience and support.

I would also like to thank my family, especially my parents and my brother, for supporting me morally and economically during all my time in Delft.

I would like to give special thanks to Ruud Kassing, with whom I have done almost all of my group projects during my master's degree studies.

Lastly, I would like to thank the members of SRC Thor, who have provided wonderful distractions in times of stress.

Delft, The Netherlands R. Cirera Rocosa
August 9, 2018

DONEC PERFICIAM

Chapter 1

Introduction

Recent years have seen an increase in the number of road-legal vehicles capable of some sort of autonomous driving. There are estimates that say that by the year 2030 autonomous vehicles could make up to 60% of all car sales in the United States [2]. Manufacturers say that autonomous cars are safer than human-controlled vehicles [3] and will lead to an increase in road safety that would result in a reduction of both car accident casualties and economic costs for drivers [4]. In addition, autonomous cars, with the implementation of technologies like Cooperative Adaptive Cruise Control and Predictive Cruise Control, can improve the traffic flow on roads [5] as well as reduce the fuel consumption of the vehicles [6]. Another advantage of autonomous vehicles is the increase in mobility for collectives of people that would not be able to drive a regular car like the elderly or visually impaired.

Most of the capabilities of current autopilots are focused on cruise driving on roads and highways [3]. However, for a car to be fully autonomous it needs to operate in city streets and parking spaces. Systems that help with parking can be found in many car models, but these systems are usually not completely autonomous and only act when a parking spot has been found [7][8][9], i.e. they do not allow for autonomous navigation in parking spaces.

This document is the final report of a MSc. thesis project that concerns itself precisely with this last mentioned capability of autonomous vehicles, navigation and parking manoeuvring in parking spaces. Previous to this document and the work described in it, a literature survey on the topic was conducted [10]. Parts of this final report document reference said literature survey report but it is not necessary for the reader to read it.

In the reminder of this chapter, Section 1-1 presents the thesis proposal statement, Section 1-2 translates the statement into concrete goals and Section 1-3 introduces the outline of the thesis report.

1-1 Thesis Proposal Statement

The thesis proposal statement is to design a motion planner for non-holonomic autonomous vehicles in parking spaces.

The motion planner must guarantee stability of the motion, avoid collisions with static obstacles and take into consideration the non-holonomic constraints of the vehicle. In addition, it must meet certain optimal performance criteria while respecting limits on velocity, acceleration and jerk for passenger safety and comfort. Finally, it must be possible to implement the motion planner in real-time applications.

In the framework of the thesis, the occupancy map of the road and the presence of static obstacles are known a priori but must be modelled. In addition, the dynamics of the actuators of the vehicle must also be considered.

The designed motion planner must be evaluated on simulations.

1-2 Thesis Goals

This section presents distinct goals for the thesis work. These goals have been extracted from the thesis proposal statement in Section 1-1.

Motion Planner

The thesis work shall result in a script that is capable of performing motion planning for a non-holonomic vehicle. The script must find a control history $u(t)$ that results in the vehicle state trajectory $q(t)$ from an initial state q_0 to a reference state q_{ref}.

Motion Stability

The motion planner shall guarantee the stability of the motion. All states of the vehicle model shall remain bounded during the planned motion.

Collision Avoidance

The motion planned shall not result in the collision of the vehicle with any obstacles. Only static obstacles shall be considered.

Non-Holonomic Constraints

The planned motion shall respect the non-holonomic vehicle constraints.

Performance Criteria

The motion planner shall ensure that the error between the reference state q_{ref} and the final state of the state trajectory is smaller than a certain threshold. The thresholds are set to 0.1 meters on the x and y position of the vehicle on the plane, 0.1 radians on the vehicle heading, 0.1 meters per second on the vehicle velocity and 0.1 meters per second squared on the vehicle acceleration.

Safety and Comfort

The resulting motion shall limit the vehicle velocity such that the vehicle is capable of short distance emergency braking. In addition, the longitudinal and lateral acceleration and jerk values of the vehicle shall be limitted such that any passengers inside the vehicle do not experience discomfort. The velocity limit is set to 5 meters per second. The limit values for acceleration are 1.0 and 0.8 meters per second squared for longitudinal and lateral acceleration respectively [11][12]. The jerk limits are 0.7 and 0.3 meters per second cubed for longitudinal and lateral jerk respectively [11][12].

Real Time Application

The run time of the script shall be such that it can be used in real time. A run time of 1 second is chosen to allow a 1Hz computation frequency for replanning.

Environment Modelling

The motion planner shall use a model of the environment that includes static obstacles. The environment information will be known a priori, and will not change over time.

Actuator Dynamics

The vehicle model used for the motion planning shall include the dynamics of the actuators.

1-3 Thesis Report Outline

The structure of this report is the following. Chapter 2 presents the approach to reach the thesis goals: the formulation of the planning problem as an Optimal Control Problem (OCP). Then, Chapter 3 explains how the solution to the OCP is found. Following that, the results of the thesis are addressed in Chapter 4. Finally, Chapter 5 contains the conclusions and recommendations.

Chapter 2

Thesis Approach

The literature study performed in advance to the thesis work looked at different types of path planning methods [10]. Upon consideration, it was decided that cost function approaches to motion planning would be good candidates to obtain a planner able to fulfil the thesis goals from Section 1-2. Cost function methods solve an optimization problem in order to plan a motion. The approach of this thesis work was to perform motion planning by solving the so called Optimal Control Problem (OCP). Section 2-1 of this chapter presents the definition of the OCP and Section 2-2 introduces the particular version of the OCP used in this thesis for motion planning.

2-1 The Optimal Control Problem

This subsection introduces the definition of OCP as presented in [13].

Let the dynamics of a system be described by Equation (2-1)

$$\dot{q}(t) = f(t, q(t), u(t)), \qquad u \in P, \qquad q(t_0) = q_0 \tag{2-1}$$

where t is the time, $t \in [0, T]$, and $q \in \mathbb{R}^n$ is the phase vector of the system. Let the values of the control parameters u belong to a given compact set $P \subset \mathbb{R}^m$. Let the initial conditions of the system be $q(t_0) = q_0 \in \mathbb{R}^n$, $t_0 \in [0, T]$. Assume that the terminal time moment T for the considered control process is fixed and the cost functional $J_{t_0,q_0}(x(\cdot), u(\cdot))$ be of the Bolza type:

$$J_{t_0,q_0}(q(\cdot), u(\cdot)) = h(q(T;\ t_0, q_0, u(\cdot))) + \int_{t_0}^{T} g(t, q(t), u(t))dt, \tag{2-2}$$

where $q(\cdot) = q(\cdot;\ t_0, q_0, u(\cdot)) : [t_0, T] \to \mathbb{R}^n$ is a trajectory of the dynamical system in Equation (2-1) starting at the initial point (t_0, q_0) under a measurable control $u(\cdot) : [t_o, T] \to P$.

The OCP attempts to guide the motion of the system in Equation (2-1) to provide the optimal cost $V(t_0, q_0)$, defined in Equation (2-3)

$$V(t_0, q_0) = \inf_{u(\cdot)\in U_{t_0}} J_{t_0,q_0}(q(\cdot;\ t_0, q_0, u(\cdot)), u(\cdot)), \tag{2-3}$$

where $(t_0, q_0) \in [0, T] \times \mathbb{R}^n$ and U_{t_0} is the set of all admissible open-loop controls $u(\cdot) : [t_o, T] \to P$ with $t_0 \in [0, T]$.

2-2 Optimal Control Problem for Motion Planning

This section presents the specific case of the OCP used for motion planning in this thesis. Subsection 2-2-1 presents the system dynamics of the problem, Subsection 2-2-2 introduces the cost function selected and Subsection 2-2-3 contains the constraints imposed on the problem solution.

2-2-1 Vehicle Model

In the context of motion planning, the system that the definition of the OCP refers to is the vehicle for which the motion is being planned. This subsection presents the vehicle model that defines the vehicle dynamics, a simplified kinematic bicycle model of a car[14] based on the sketch in Figure 2-1.

The state of the vehicle in the model is denoted q, a 6 dimensional vector with components shown in Equation (2-4). The position of the vehicle on the plane is given by the x and y coordinates, in meters, of the point at the intersection of the rear axis and the longitudinal center line of the vehicle. The orientation of the vehicle ψ is measured in radians, counter-clockwise with respect to the $x-$axis, and always mapped to the interval $[-\pi, \pi]$. The longitudinal velocity is denoted v, measured in meters per second, and is always parallel to the orientation of the vehicle. The steering angle δ is measured in radians, counter-clockwise with respect to the vehicle longitudinal centreline. The longitudinal acceleration a is measured in meters per second squared and, like the longitudinal velocity, is considered to always be parallel to the orientation of the vehicle. Two more important parameters characteristic to the vehicle are the distance between the front and rear wheel axis, denoted L and measured in meters, and the turn radius ρ, also in meters. Both are shown in Figure 2-1. The expression for ρ is given by Equation (2-5).

$$q = \begin{bmatrix} x \\ y \\ \psi \\ v \\ \delta \\ a \end{bmatrix} \tag{2-4}$$

$$\rho = \frac{L}{tan(\delta)} \tag{2-5}$$

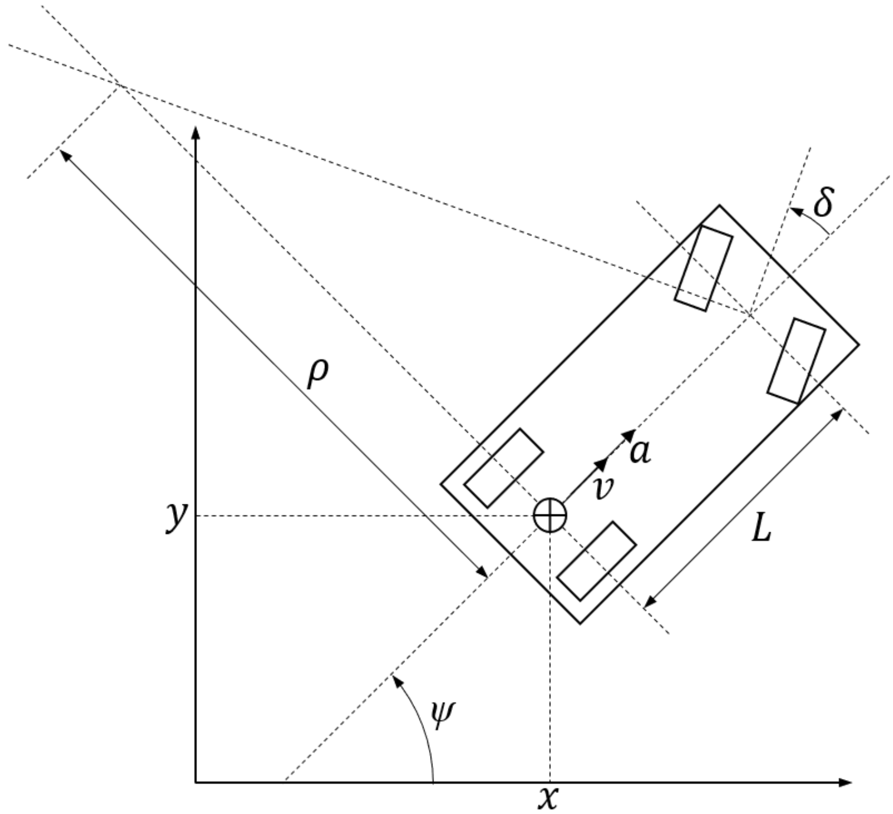

Figure 2-1: Schematic representation of the vehicle, a front-steering car, considered in the motion planner.

The input to the vehicle model is denoted u, a 2 dimensional vector shown in Equation (2-6) that contains the reference steering angle δ_{ref} and the reference longitudinal acceleration a_{ref}. The units for the inputs are radians and meters per second squared respectively. The inputs belong to a compact set with upper limits $\delta_{ref,max}$ and $a_{ref,max}$ and lower limits $\delta_{ref,min}$ and $a_{ref,min}$.

$$u = \begin{bmatrix} \delta_{ref} \\ a_{ref} \end{bmatrix}, \qquad \begin{bmatrix} \delta_{ref,min} \\ a_{ref,min} \end{bmatrix} \leq u \leq \begin{bmatrix} \delta_{ref,max} \\ a_{ref,max} \end{bmatrix} \tag{2-6}$$

The vehicle model used assumes that there is no slippage between the wheels and the ground. The dynamics of the actuators are present in the model as first order transfer functions with time constants t_d and t_a, both in seconds. The kinematic model is given in Equation (2-7), in which all the states and inputs are functions of time, but the notation has been dropped for simplicity.

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\delta} \\ \dot{a} \end{bmatrix} = f(q,u) = \begin{bmatrix} v\cos(\psi) \\ v\sin(\psi) \\ \frac{v}{L}\tan(\delta) \\ a \\ \frac{\delta_{ref}-\delta}{t_d} \\ \frac{a_{ref}-a}{t_a} \end{bmatrix} \tag{2-7}$$

2-2-2 Cost Function

This subsection presents the specific version of Equation (2-2) used in the motion planner. The cost function was selected in order to steer the motion towards a given reference state q_{ref}. Only the final state of the motion is of concern. Thus the function $g(t, q(t), u(t))$ in the integral in Equation (2-2) is set to zero,

$$g(t, q(t), u(t)) = 0, \tag{2-8}$$

and only the final state term remains, given by Equation (2-9).

$$h(q(T;\ t_0, q_0, u(\cdot))) = \|q_{ref} - q(T;\ t_0, q_0, u(\cdot))\|_W \tag{2-9}$$

The weight matrix W is a 6x6 identity matrix in which the fifth diagonal entry has been changed to a zero. This cancels the contribution of the steering angle error to the cost, because the actual steering angle in the final state is of no concern to the performance criteria, presented in Subsection 1-2. As is clear from Equation (2-9) the cost function is convex with respect to $q(T)$. However, because of the nonlinear model that relates q to u, shown in Equation (2-2-1), the cost function is nonlinear and non-convex with respect to the minimization variable, the control history $u(t)$.

2-2-3 Constraints

Many of the thesis goals can be translated into constraints for the solution of the OCP. This subsection explains how the collision avoidance, safety and comfort and performance goals were translated into constraints.

Collision Avoidance

In order to constrain the collisions to zero, a function that computes a collision cost $J_{col}(q(t))$ is used. A collision is considered to occur if the footprint of the vehicle along the state trajectory overlaps the footprint of any obstacle. The collision cost is zero if no collisions occur and positive otherwise. The cost increases with the the total vehicle and obstacle footprint overlap, added across the entire state trajectory. A safety margin is added to the computation as a virtual increase of the size of the obstacles so that a certain distance will be left at all times between the vehicle and the obstacles. The constraint is given by Equation

(2-10). Since the collision cost depends on the state trajectory $q(t)$ which in turn depends on the control history $u(t)$, this method allows a minimization algorithm to calculate the collision cost as a function of the minimization variable $u(t)$.

$$J_{col}(q(t)) = 0 \tag{2-10}$$

Safety and Comfort

The limits for velocity, acceleration and jerk, both longitudinal and lateral, have been presented in Chapter 1-2. The velocity v and longitudinal acceleration a_{long} are two of the states of the vehicle model. The longitudinal jerk j_{long} is the derivative of the longitudinal acceleration, which is also given by the vehicle kinematic model. The lateral acceleration a_{lat} is a function of the longitudinal velocity v and steering angle δ. The lateral jerk j_{lat} is simply the derivative of the lateral acceleration. The constraints can thus be written as in Equations (2-11) to (2-15).

$$-5.0 \leq v(t) \leq 5.0 \quad \forall\ t \in [0, T] \tag{2-11}$$

$$-1.0 \leq a_{long}(t) = a(t) \leq 1.0 \quad \forall\ t \in [0, T] \tag{2-12}$$

$$-0.8 \leq a_{lat}(t) = \frac{v(t)^2}{\frac{L}{tan(\delta(t))}} \leq 0.8 \quad \forall\ t \in [0, T] \tag{2-13}$$

$$-0.7 \leq j_{long}(t) = \frac{a_{ref}(t) - a(t)}{t_a} \leq 0.7 \quad \forall\ t \in [0, T] \tag{2-14}$$

$$-0.3 \leq j_{lat}(t) = \dot{a_{lat}}(t) \leq 0.3 \quad \forall\ t \in [0, T] \tag{2-15}$$

Performance

In order to satisfy the performance criteria defined in Chapter 1-2, the constraints in Equations (2-16) to (2-20) must be satisfied. The parameters x_{ref}, y_{ref}, ψ_{ref}, v_{ref} and a_{ref} are the states of the reference state vector q_{ref}.

$$\|x_{ref} - x(T)\| \leq 0.1 \tag{2-16}$$

$$\|y_{ref} - y(T)\| \leq 0.1 \tag{2-17}$$

$$\|\psi_{ref} - \psi(T)\| \leq 0.1 \tag{2-18}$$

$$\|v_{ref} - v(T)\| \leq 0.1 \tag{2-19}$$

$$\|a_{ref} - a(T)\| \leq 0.1 \tag{2-20}$$

Chapter 3

Solving the Optimal Control Problem

This chapter presents the approach followed in order to solve the Optimal Control Problem (OCP) introduced in Chapter 2. Section 3-1 addresses the discretization of the OCP that transforms it into a multivariable nonlinear minimization problem with nonlinear constraints. The MATLAB function and algorithm used in the motion planner script to solve the discretized OCP are presented in Section 3-2. Section 3-3 elaborates on the issues encountered when using said function and algorithm to solve the multivariable nonlinear minimization problem. Finally, Section 3-4 describes the actions taken to overcome those issues.

3-1 Time Discretization of the Problem

The OCP presented in Chapter 2 can be transformed into a multivariable nonlinear minimization problem with nonlinear constraints by discretizing the vehicle model, cost function and constraints with respect to time. This section explains how such discretizations were performed. Subsection 3-1-1 addresses the model discretization. Subsection 3-1-2 presents the discretization of the cost function. Finally, Subsection 3-1-3 focuses on the discretization of the optimization constraints.

3-1-1 Vehicle Model

The kinematic model was discretized by means of a first order forward discretization scheme as shown in Equation (3-1)

$$q(k+1) = \begin{bmatrix} x(k+1) \\ y(k+1) \\ \psi(k+1) \\ v(k+1) \\ \delta(k+1) \\ a(k+1) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \psi(k) \\ v(k) \\ \delta(k) \\ a(k) \end{bmatrix} + dt \cdot \begin{bmatrix} v(k)\cos(\psi(k)) \\ v(k)\sin(\psi(k)) \\ \frac{v(k)}{L}\tan(\delta(k)) \\ a(k) \\ \frac{\delta_{ref}(k)-\delta(k)}{t_d} \\ \frac{a_{ref}(k)-a(k)}{t_a} \end{bmatrix} \quad \text{for } k = 0, 1, 2, ..., N-1 \tag{3-1}$$

in which k is the index of a time step, dt is the size of the time steps and N is the total number of time steps. The state trajectory of the vehicle is then expressed by the N state vectors $q(0), q(1), q(2), ..., q(N)$, with $q(0) = q_0$. The control history is given by the $N-1$ input vectors $u(0), u(1), u(2), ..., u(N-1)$ with elements as shown in Equation (3-2).

$$u(k) = \begin{bmatrix} \delta_{ref}(k) \\ a_{ref}(k) \end{bmatrix} \qquad \text{for } k = 0, 1, 2, ..., N-1 \tag{3-2}$$

3-1-2 Cost Function

Using the discretized state and input vectors defined in Section 3-1-1, the cost function in Equation (2-9) becomes Equation (3-3), in which the dependence of $q(N)$ on q_0 and the control history has not been explicitly shown in order to simplify the expression. Similarly to its continuous counterpart, the cost function in Equation (3-3) is nonlinear and non-convex with respect to the optimization variables, the control history shown in Equation (3-2). However, for a given control history and initial state, the kinematic model in Equation (3-1) can be used to obtain the state trajectory, with which the cost can be computed.

$$h(q(N)) = \|q_{ref} - q(N)\|_W \tag{3-3}$$

3-1-3 Optimization Constraints

This subsection shows how the collision avoidance, passenger comfort and performance constraints in Subsection 2-2-3 can be rewritten for the time discrete system.

Collision Avoidance

The collision avoidance constraint from Equation (2-10) becomes the expression shown in Equation (3-4), in which $J_{col}(q(\cdot))$ is the discrete time collision cost. To calculate the discrete collision cost, a series of points are placed on the perimeter of the vehicle footprint at every time step k under consideration. Every point that is inside the footprint of an obstacle incurs in a cost. The closer to the centre of the object, the higher the cost. The collision cost is the sum of the cost of every point. A safety margin is used in the computation by considering the obstacles to be larger than they really are. This way, a minimum distance between the vehicle and the obstacles can be maintained for all time steps if no collision occurs.

$$J_{col}(q(\cdot)) = 0 \tag{3-4}$$

Safety and Comfort

The constraints for the velocity and the longitudinal and lateral acceleration and jerk in Equations (2-11) to (2-15) can be discretized as in Equations (3-5) to (3-9).

$$-5.0 \leq v(k) \leq 5.0 \quad \text{for } k = 0, 1, 2, ..., N \tag{3-5}$$

$$-1.0 \leq a_{long}(k) = a(k) \leq 1.0 \quad \text{for } k = 0, 1, 2, ..., N \tag{3-6}$$

$$-0.8 \leq a_{lat}(k) = \frac{v(k)^2}{\frac{L}{tan(\delta(k))}} \leq 0.8 \quad \text{for } k = 0, 1, 2, ..., N \tag{3-7}$$

$$-0.7 \leq j_{long}(k) = \frac{a_{ref}(k) - a(k)}{t_a} \leq 0.7 \quad \text{for } k = 0, 1, 2, ..., N \tag{3-8}$$

$$-0.3 \leq j_{lat}(k) = \frac{a_{lat}(k+1) - a_{lat}(k)}{dt} \leq 0.3 \quad \text{for } k = 0, 1, 2, ..., N \tag{3-9}$$

Performance

The performance criteria in Equations (2-16) to (2-20) become the expressions in Equations (3-10) to (3-14) for the time discrete case.

$$\|x_{ref} - x(N)\| \leq 0.1 \tag{3-10}$$

$$\|y_{ref} - y(N)\| \leq 0.1 \tag{3-11}$$

$$\|\psi_{ref} - \psi(N)\| \leq 0.1 \tag{3-12}$$

$$\|v_{ref} - v(N)\| \leq 0.1 \tag{3-13}$$

$$\|a_{ref} - a(N)\| \leq 0.1 \tag{3-14}$$

3-2 Optimization Algorithm

Solving the discretized OCP is equivalent to solving a multivariable nonlinear optimization problem with nonlinear constraints, in which the optimization variables are the elements of the control history $u(k)$ for $k = 0, 1, 2, ..., N$. The MATLAB Optimization Toolbox has many functions capable of optimizing different types of problems. The function `fmincon` finds the minimum of constrained nonlinear multivariable functions and was therefore used in the implementation of the planner [15].

There are multiple algorithms that `fmincon` can use for the minimization. Three of these algorithms can be used in this particular implementation. These algorithms are `interior-point`, `sqp` and `active-set`. Preliminary tests showed that the `sqp` algorithm was faster than `interior-point` or `active-set` and was thus used during the implementation and testing. A comparison between the performance of the different algorithms can be found in Section 4-2.

3-3 Implementation Issues

Unfortunately, the results of writing the cost and constraint functions from Section 3-1 and using them with `fmincon` are not satisfactory. The resulting motions fail to meet the performance criteria from Section 1-2 by a very large margin. Moreover, the script run time is too long to satisfy the real time application goal. This section will address the main causes for this. Subsection 3-3-1 focuses on the final state error while Subsection 3-3-2 concerns itself with the run time.

3-3-1 Final State Error

The cost function in Equation (3-3) is non-convex with respect to the optimization variables. Simply using the cost and constraints presented in Subsections 3-1-2 and 3-1-3 results in a motion planner that gets stuck in local minima, resulting in final state errors that do not meet the performance criteria.

An example can be seen in Figure 3-1, where obstacles are present between the initial state and the final state. The planner reaches a feasible local minima just before colliding with an obstacle. If the planner were to follow the gradient of the cost function in a further iteration, the motion would result in a collision with the obstacles. Furthermore, changing the final state in any other direction would increase the value of the cost function. Thus, the planner is stuck in a local minima.

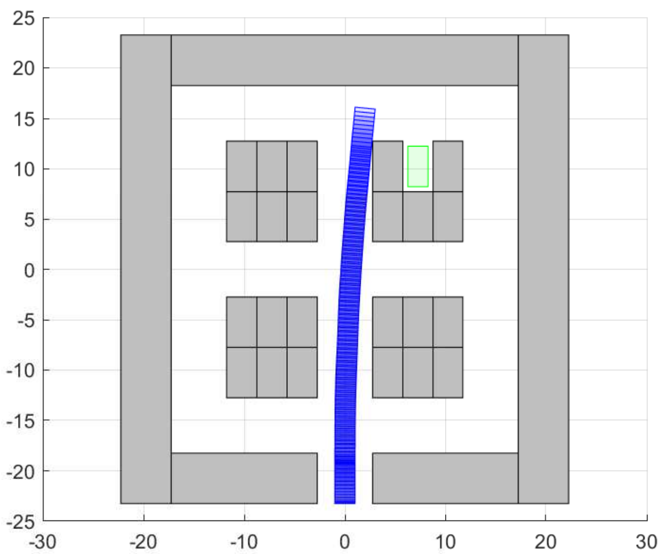

Figure 3-1: Path of a motion that ends stuck in a local minima due to the presence of obstacles. The motion begins at the bottom centre of the image. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively. The goal state is represented by the green outline in the empty parking spot. The motion duration is 45 seconds, divided into 90 steps of 0.5 seconds.

Another situation in which the planner fails is when attempting to plan the manoeuvre of the vehicle into a parking spot. The density and proximity of obstacles around the goal state make it very easy for the planner to get stuck in local minima. When the planning horizon is short, as shown in Figure 3-2, the planner seems to start in the right direction, but it is not capable of completing the motion. If the planning horizon is increased, the planner is able to plan the manoeuvring motion as shown in Figure 3-3. However, the initial state is crucial in the success of the planner. If noise is added to the initial state the planner no longer manages to find a motion that enters the parking spot, as shown in Figure 3-4. The planner also fails if the planning horizon is too long. In these cases, the initial step of the minimization results in a feasible current guess that moves past the objective parking spot and the planner gets stuck on the other side of an obstacles with respect to the goal state. This can be seen in Figure 3-5.

3-3-2 Run Time

The attempts at motion planning of the initial implementation of `fmincon` with the cost function and constraints described in Section 3-1 have run times ranging from a few seconds to well over a minute, depending on the number of time steps used in the planning horizon. These values do not comply with the real time application goal of Section 1-2. Even the lowest run times are too long, and they are obtained with planning horizons that are so short it might be impossible to obtain motions that respect all the constraints.

Figure 3-6 shows boxplots for the run time of the optimizing algorithm for different values of the number of time steps N. The boxplots were obtained with 5 identical runs each, but with a different feasible goal for every value of N. The goal was always straight in front of the initial state of the vehicle, but the distance changed according to the number of time steps so that the average speed on the motion planned would be similar.

As it can be seen by the fitted curve, the run time grows exponentially with increasing N. Therefore the planning of long motions becomes impractical. The increase of the run time with increasing N is explained in part by the increase of the number of optimization variables present in the planning of longer paths. Another aspect adding to the run time is the extra constraints that are added for every time step considered in the planning. The computation complexity of the constraints under consideration also affect the run time, the more complex the computation, the larger the run time.

3-4 Solving the Implementation Issues

This section presents all the actions taken in in the implementation of the planner in order to solve the problems presented in Section 3-3.

3-4-1 Split Motion Planning

The planning of the motion is split into two parts. The first part is the parking navigation, from the initial state of the vehicle until the vicinity of the goal parking spot. The second part is the parking manoeuvring, from the end of the parking navigation part until the goal state in the parking spot. The reason for this change in the planning implementation is that

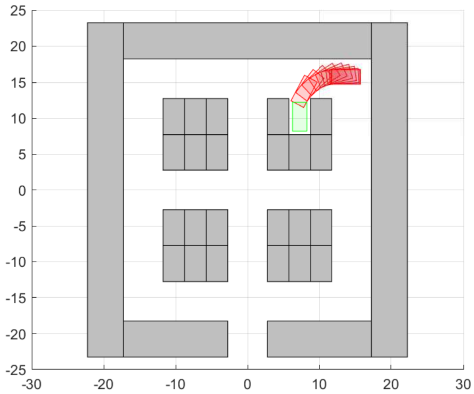

Figure 3-2: Path of a manoeuvring motion that ends stuck in a local minima due to the presence of obstacles. The motion begins at the top right of the image. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively. The goal state is represented by the green outline in the empty parking spot. The motion duration is 5 seconds, divided into 10 steps of 0.5 seconds.

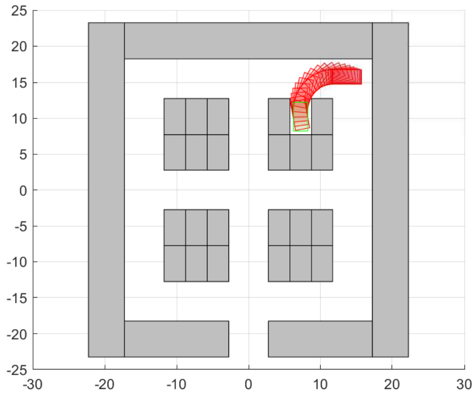

Figure 3-3: Path of a manoeuvring motion that ends stuck in a local minima due to the presence of obstacles. The motion begins at the top right of the image. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively. The goal state is represented by the green outline in the empty parking spot. The motion duration is 10 seconds, divided into 20 steps of 0.5 seconds.

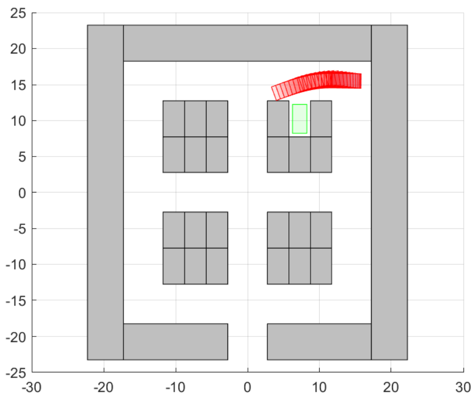

Figure 3-4: Path of a manoeuvring motion that ends stuck in a local minima due to the presence of obstacles. The motion begins at the top right of the image. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively. The goal state is represented by the green outline in the empty parking spot. The motion duration is 10 seconds, divided into 20 steps of 0.5 seconds.

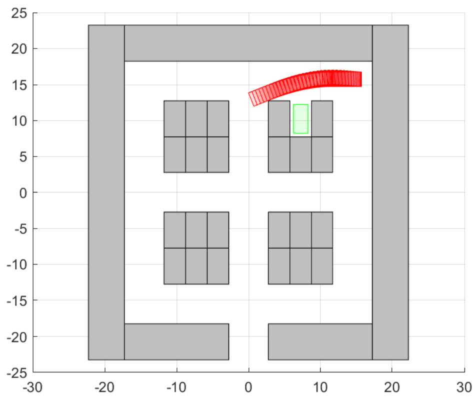

Figure 3-5: Path of a manoeuvring motion that ends stuck in a local minima due to the presence of obstacles. The motion begins at the top right of the image. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively. The goal state is represented by the green outline in the empty parking spot. The motion duration is 15 seconds, divided into 30 steps of 0.5 seconds.

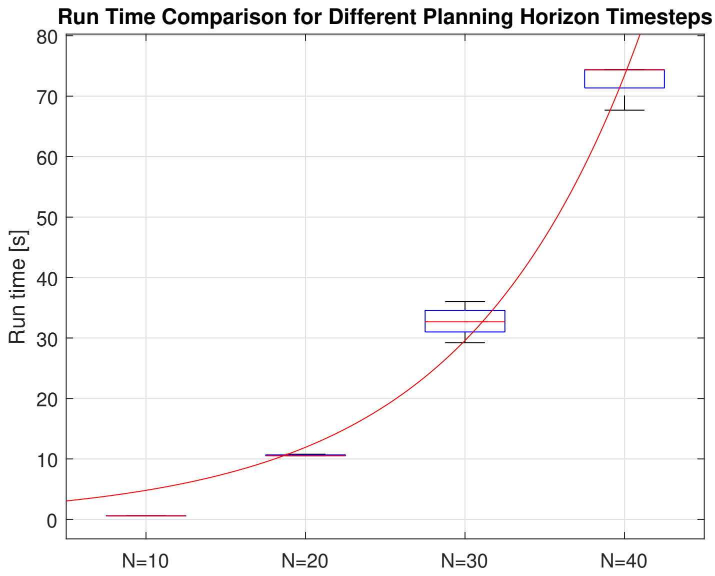

Figure 3-6: Boxplots and exponential trend of the run time of `fmincon` with `sqp` algorithm for different number of planning horizon time steps N.

the two parts of the motion have distinct characteristics, so planning them separately allows the tuning of the planner to each of them, improving the final result.

3-4-2 Local Motion Planning and Waypoint Tracking

The two parts of the motion are not planned at once. Instead, the planning horizons of the navigation and manoeuvring parts are split into equal segments, each containing a smaller number of time steps. The motion is planned by repeatedly considering a number of these segments as the local planning horizon. The planning will start at the beginning of the motion, and use the final states of earlier segments as initial states for the planning of later ones. Due to the exponential growth of the run time with respect to the number of time steps in the planning horizon, shown in Figure 3-6, splitting the planning into multiple shorter steps will result in an overall decrease in run time when compared to planning the motion considering the entire planning horizon at once.

However, in order to perform local planning, a local goal state is required. A series of waypoints are used for this purpose. In addition to allowing local motion planning, the use of waypoints also allows the planner to steer the motion around obstacles during the navigation part of the motion. For the manoeuvre part, following waypoints that are known to be in a feasible path towards the goal state prevent the planner from getting stuck in local minima.

The waypoints for the navigation and manoeuvring parts of the motion are obtained as follows. First, two candidate manoeuvre paths are created. These have one of the ends in the goal state and consist of a straight line and a minimum radius arc segment turning either to the right or to the left. Figures 3-7 and 3-8 show two candidate manoeuvre paths with the respective vehicle outlines. The length of the straight segment is determined geometrically with the dimensions of the vehicle considered so that the path observes a minimum safety distance with respect to the neighbouring parking spots. Then, the points in the path that result in a collision of the vehicle with other obstacles are eliminated. It is relevant to note that a single direction manoeuvre has been implemented in the script. The manoeuvre is such that the vehicle enters the parking spots backwards. However, it is possible to modify the script to have the manoeuvre be forwards or to have more complex manoeuvres.

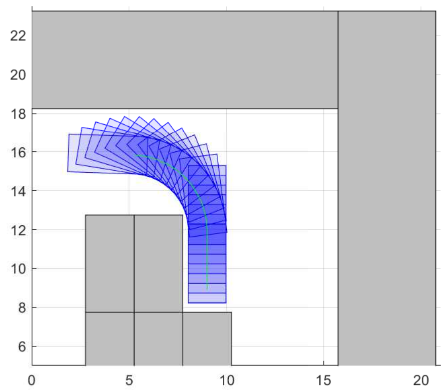

Figure 3-7: Candidate manoeuvre path with vehicle outline for every path point.

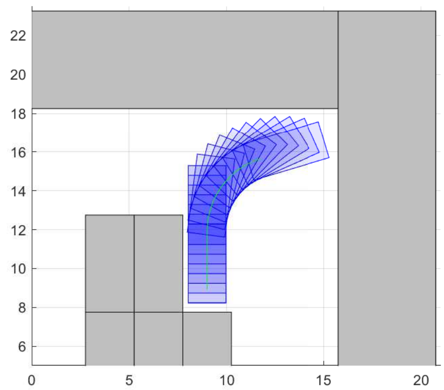

Figure 3-8: Candidate manoeuvre path with vehicle outline for every path point.

Next, A* is run to obtain two navigation paths connecting the initial state location to the first point of either candidate manoeuvre path. In order to ensure that the navigation paths approach the initial manoeuvre point with the correct orientation, a set of additional imaginary objects are considered during each A* run. These objects can be seen in Figure 3-9 and 3-10. The additional objects are two circles with radius equal to the minimum turn radius of the vehicle, and a rectangular object connecting them on the front of the initial manoeuvre state. The presence of these obstacles forces the waypoints found by A* to approach the goal state from the correct direction and with a feasible minimum turn radius.

In the following step, the shortest navigation path is selected together with its corresponding manoeuvre path. The navigation and manoeuvre waypoints are then obtained by re-sampling the paths so that consecutive waypoints are separated by a desired distance. An example of the final waypoints obtained can be seen in Figure 3-11.

3-4-3 Changed Local Planning Cost Function

The cost function presented in Subsection3-1-2 was thought for the end state of the planned motion. It is still be used as the cost function for the final states of the navigation and manoeuvre parts of the motion, but intermediate local planning steps use different cost functions.

In the intermediate navigation local planning steps, the weight matrix W is altered so that only the final x and y states of the vehicle are considered. This is because these two states are the only information known from the output of A*. An orientation could be calculated from a relative position between successive waypoints, but because of the holonomic nature of the A* generated paths, attempting to reach the waypoints with a specific orientation might

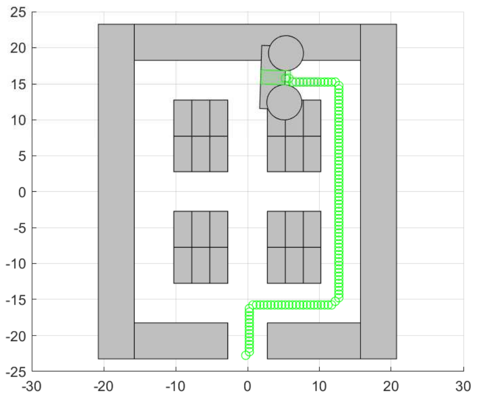

Figure 3-9: Resulting path of A*. The addition of imaginary objects force the approach of the path from the appropriate direction.

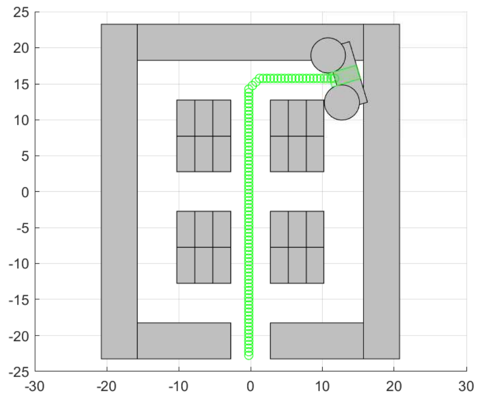

Figure 3-10: Resulting path of A*. The addition of imaginary objects force the approach of the path from the appropriate direction.

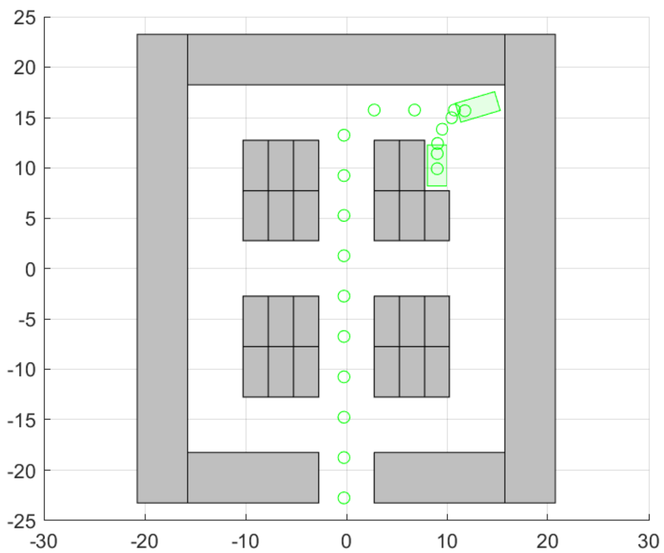

Figure 3-11: Final waypoints, obtained after re-sampling the navigation and manoeuvre paths. The green circles mark intermediate waypoints of either part of the motion. The two green rectangular vehicle outlines are the final navigation and manoeuvre waypoints.

result impossible. The orientation is thus not considered in the cost function. In addition, the velocity and acceleration at intermediate points of the motion are irrelevant and thus also not considered.

The same argument about velocity and acceleration stands for the intermediate local planning of the manoeuvre part of the motion. However, the manoeuvre waypoints are obtained in such a way that both position and orientation are available. Thus, the weight matrix used for these waypoints is such that only the x, y and ψ states are considered in the cost function.

3-4-4 Optimization Cut-off Value

A cost function lower limit is set on the function `fmincon`. If the value of the cost function goes below this threshold and the current iteration is feasible, the minimization stops. The threshold is set to 0.01. All performance constraints from Equations (3-10) to (3-14) are necessarily met when the cost function reaches this value. Actually, the constraints are met when the cost is lower or equal to 0.1. This more exigent threshold is set to ensure that the final navigation state, and thus the initial manoeuvre state, is closer to the reference in order to simplify the manoeuvre planning. Setting this lower limit on the cost function results in skipping further minimization iterations that were previously performed, reducing the overall run time of the script. The effect can be seen when comparing the run times in Figure 3-12, in which the cut-off threshold is applied, to the run times in Figure 3-6.

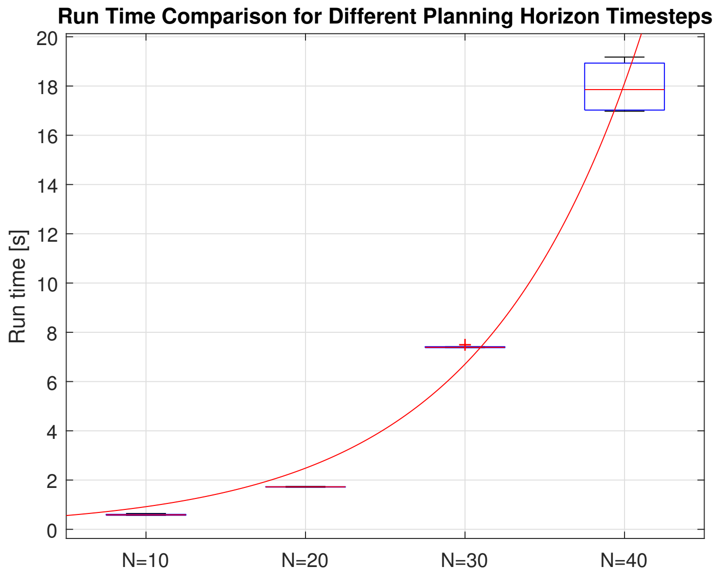

Figure 3-12: Boxplots and exponential trend of the run time of `fmincon` with `sqp` algorithm and the cut-off threshold for different number of planning horizon time steps N.

3-4-5 Time Step Simulation Subdivision

The number of variables of the optimization are reduced by increasing the duration of the time steps. In order to maintain the simulation precision, each time step is simulated by a succession of shorter steps during which the value of the inputs are constant.

3-4-6 Skipping the Performance Constraints

Because of the close relation between the cost function and the performance constraints, and in order to save time, the performance constraints are not considered in the minimization with `fmincon`. Instead, a check is performed on the value of the cost function. The performance constraints are satisfied if the cost function value is equal or lower than 0.1. This is a stricter performance measure than the original constraints, but it is used for its increased simplicity.

3-4-7 Structure of the Local Planning Step

The objective of the local motion planner is to find a motion that satisfies the safety and comfort constraints, the final state error constraint and the obstacle collision constrains. However, the calculation of the obstacle collision constraint is computationally expensive and results in undesirably long run times. Moreover, it was observed during the implementation of the planner that for a large number of local planner steps the resulting motions of the local planner do not collide with any obstacles even if the collision constraint is not considered in the optimization. In order to take advantage of this, the local motion planning was split into two steps designed to minimize the run time.

Local Planning Without Obstacles

The local planner ignores the collision constraint on its first local planning step, in hopes of finding a local motion that satisfies all the constraints while achieving a short run time. Thus, the actual optimization only considers the safety and comfort constraints. After the optimization steps, the resulting motion is evaluated on its compliance of the final state error constraint. If the constraint is not met, the planning horizon is increased backwards, considering an additional segment of the global planning horizon and the corresponding initial state. This is repeated until the performance constraint is met or the horizon can not be increased backward any further, either because the maximum planing horizon length allowed or the beginning of the motion have been reached. If the horizon can not be further increased, the planner moves on to the next global planning step, with the next waypoint as goal state and a reset size of the local planning horizon.

If the performance constraint is met, the planned motion is evaluated on its compliance of the obstacle collision constraint. If the collision constraint is satisfied, the planner has succeeded on planning a local motion that satisfies all constraints and the planner moves on to the next global planning step. There is no check made on the safety and comfort constraints because they are always satisfied by the first control history guess, a zero input resulting in no movement of the vehicle, and the chosen algorithm, `sqp`, stays in the region constrained by bounds in every iterative step [16]. If the collision constraint is not satisfied, the local planner moves on to the next local motion planning step.

Local Planning With Obstacles

This step of the local motion planning is essentially the same as the previous one, but this time the planner considers the obstacle collision constraint in the optimization. The local planning horizon that was considered in the last optimization of the previous step is used as the initial local planning horizon. In addition, the last solution of the previous step is used as the first guess for the optimization. Similar to the previous step, the resulting motions are checked for their final state error constraint compliance and their obstacle collision compliance. If they do not satisfy either of the constraints, the local planning horizon is increased in the same fashion as in the first local planning step, if possible, and the optimization is repeated. As before, if the local planning horizon can not be extended further, the planner moves on to the next global planning step. If the constraints are satisfied, the planner has succeeded in finding an acceptable local motion, and the planner moves on to the next global planning step.

While the run time in cases in which both local planning steps are required is longer than it would be if only the second step was performed, the large proportion of local planning steps that only require the first step results in an overall decrease of the run time for the motion planner. In addition, the ability of the script to increase the local planning horizon backwards when necessary allows the initial local planning horizon to be kept short, reducing overall run time. Still, in situations in which complex manoeuvres that span a larger planning horizon must be performed, the planner is capable of increasing the horizon allowing final state error requirements to be fulfilled.

3-4-8 Turn Collision Planning

In cases when one of the waypoints was situated in the middle of a turn, the result of the first local planning step resulted in a local motion that minimized the final state error but resulted in a collision. The second step of the local motion planning required the planning horizon to be extended backwards many times before the motion planner was able to find a satisfactory local motion. Naturally this resulted in long run times of the motion planner. In order to improve the performance of the motion planner, a short series of actions were added to be performed between the two steps of the local motion planning.

If the first local planning step has not resulted in a motion that satisfies the collision and final error constraints, before performing the second step of the local motion planning, the planner increases the current planning horizon forwards, considering the next waypoint as the goal state. The optimization is performed again with the new horizon but still without considering the obstacle collision constraint. The resulting motion is then checked for its compliance of the performance and collision constraints, as was done in the first step of the local planning. In many cases, considering the next waypoint as the goal state will result in a collision-less motion that satisfies the final state error constraint, as shown in Figure 3-13 and Figure 3-14, and the planner will move on to the next global planning step, avoiding the time consuming second step of the local planning that considers the obstacle collision constraint in the optimization.

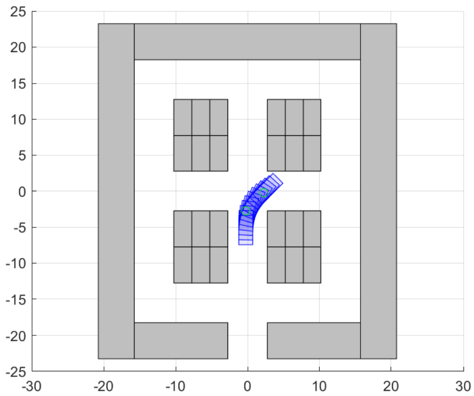

Figure 3-13: Local planning with waypoint k as goal. A collision occurs. Waypoints k-1 to k are shown as green circles. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively.

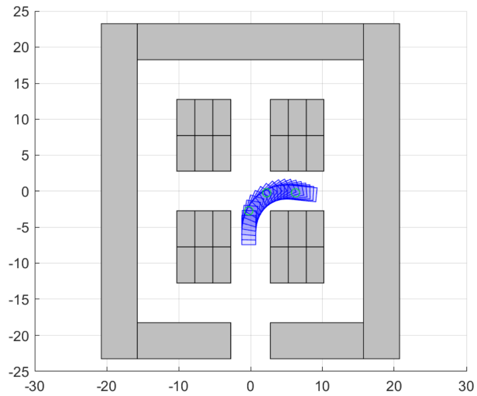

Figure 3-14: Extended local planning with waypoint k+1 as goal. Note the extra waypoint, now waypoints k-1 to k+1 are shown as green circles. By extending the horizon forward and considering waypoint k+1 as the current goal the collision is avoided. The outline of the vehicle is displayed in at every time step k. Blue and red outlines represent forward and backwards movement of the vehicle respectively.

Chapter 4

Results

This chapter presents the results of the thesis. Section 4-1 presents the results of the motion planner in simulations. The sensitivity of the motion planner is analysed in Section 4-2.

4-1 Simulation Results

This section presents results obtained with the motion planner that are representative of the general performance of the script. The vehicle model used to obtain the results is that of a Seat Ibiza. The vehicle model constants in Equation (3-1) and their values are presented in Table 4-1. The SEAT Ibiza has a vehicle length of 4.0590m and a vehicle width of 1.9420m [1]. The parking environment, shown in Figure 4-1, has 24 parking spots. The length and width of the parking spots are 5.0 and 2.5 meters respectively, and the lanes are 5.5 meters wide. These values were taken from [17]. The motion planning from the entrance to the parking spots was performed 20 times for each of the 24 parking spots. Every run had the initial condition altered by a very small perturbation, on the order of $1 \cdot 10^{-6}$ meters in position and orientation. This was done to prevent every run to perform the same exact calculations.

The planner was successful in reaching the goal state for all the parking spots and all the runs, managing to stay within the cost value threshold of 0.01 and satisfying the obstacle collision constraint as well as the safety and comfort constraints. Examples of the path of one of the motions planned can be seen in Figure 4-2. Figures 4-3 and 4-4 show the plots of the safety and comfort constraints for the same planned motion.

Model constant	Symbol	Value [Units]
Inter-axis length	L	2.5640 [m]
Steering angle transfer function time constant	t_d	2 [s]
Acceleration transfer function time constant	t_a	2 [s]

Table 4-1: Vehicle model constants used to obtain the results presented in Section 4-1[1].

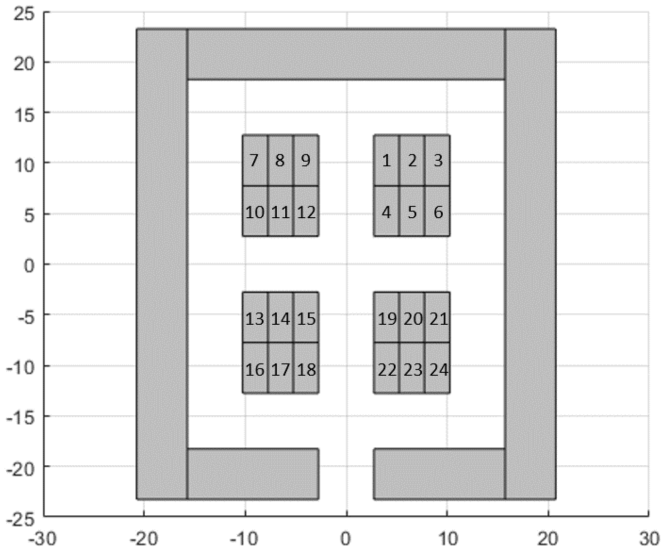

Figure 4-1: Overview of the parking environment, including the numbering system for the parking spots. The parking spot dimensions are 2.5x5.0m and the lanes are 5.5 meters wide.

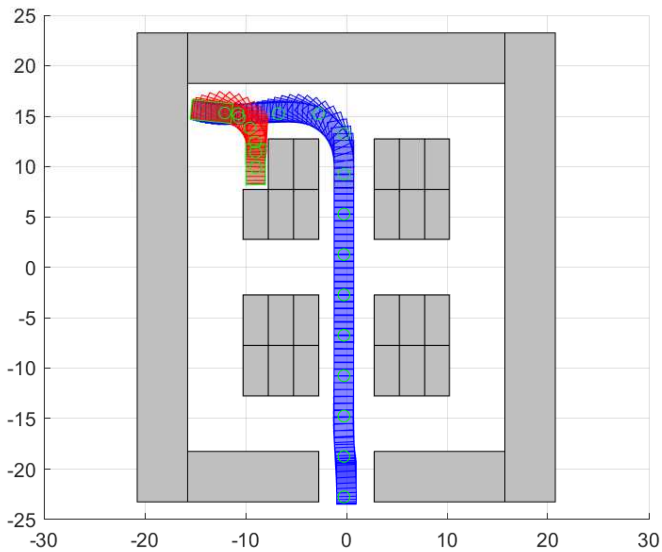

Figure 4-2: Path followed by a motion planned with parking spot 7 as objective. Blue vehicle outlines indicate states in which the velocity is positive. Red outlines indicate negative velocity. The green circles are the waypoints followed by the planner. The green vehicle outline represents the goal state of the planner.

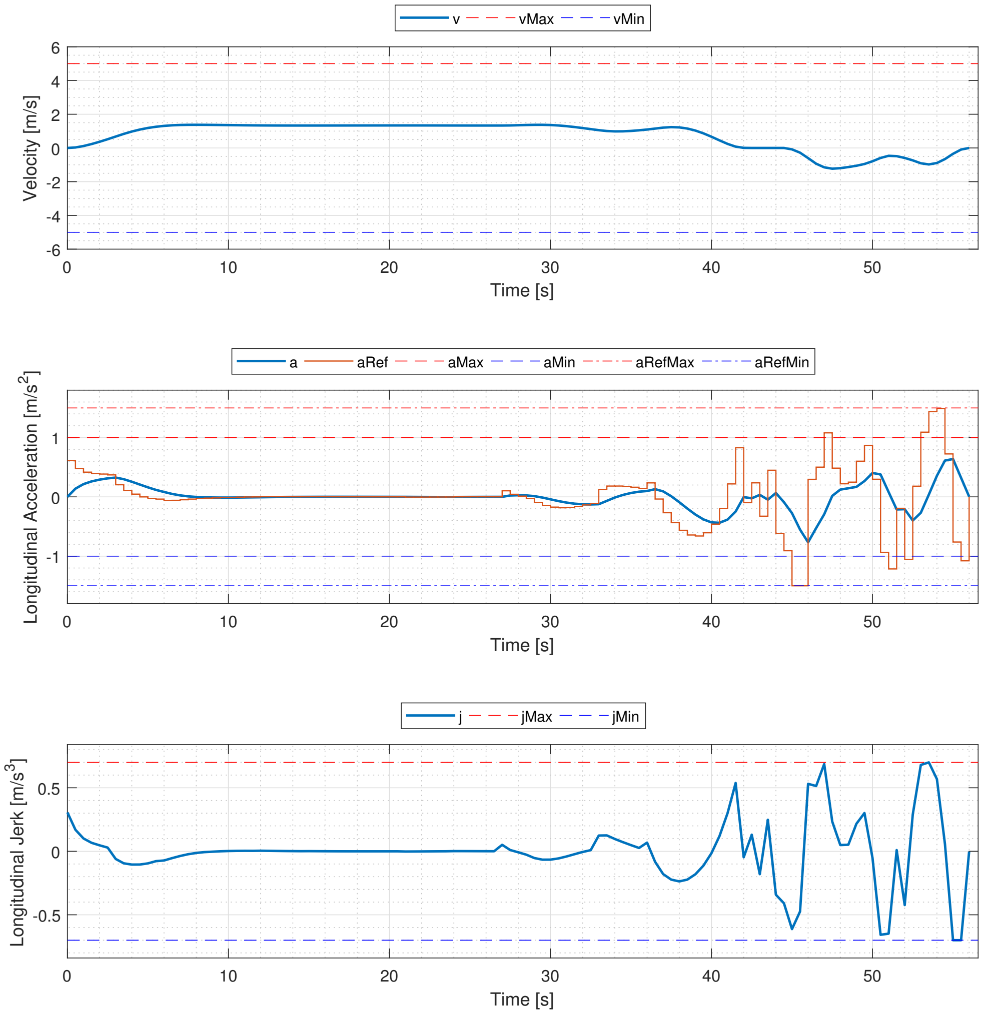

Figure 4-3: Plots of the velocity, longitudinal acceleration, reference longitudinal acceleration and longitudinal jerk of a motion planned with parking spot 7 as objective.

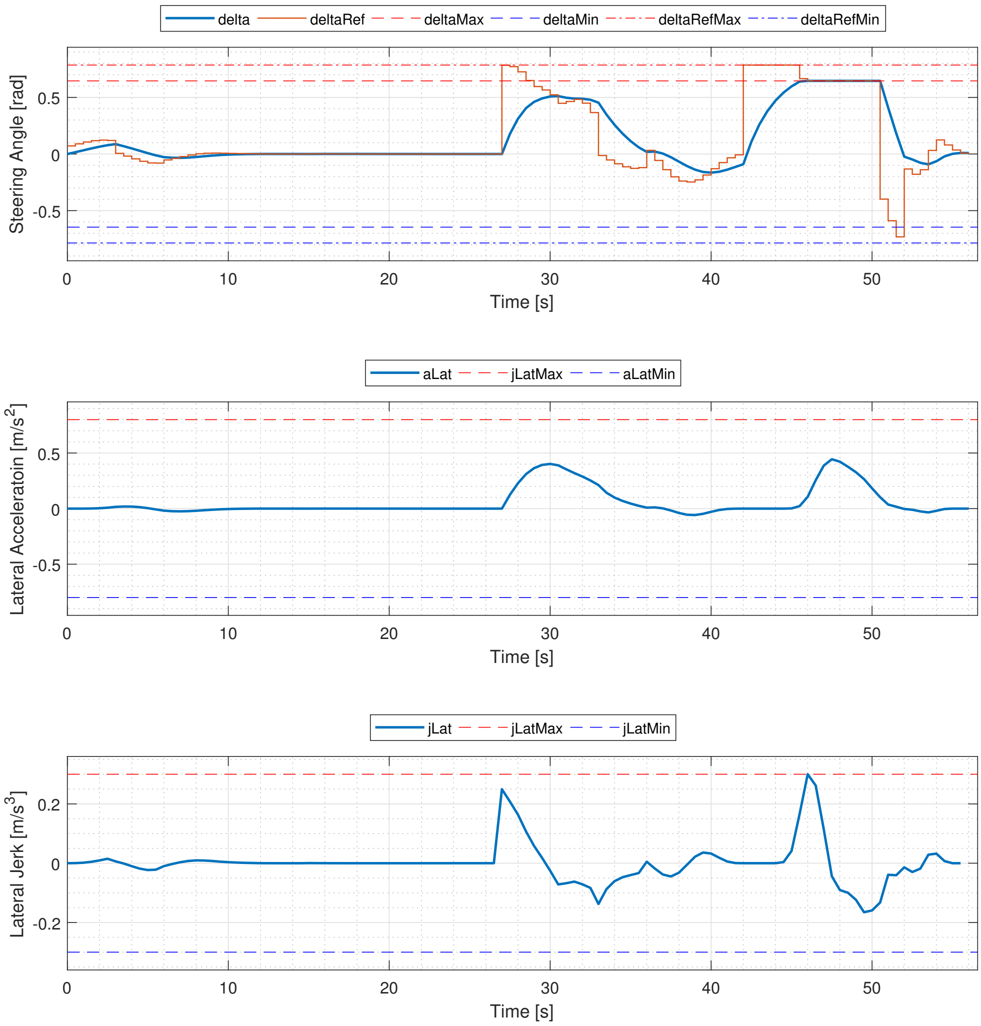

Figure 4-4: Plots of the steering angle, reference steering angle, lateral acceleration and lateral jerk of a motion planned with parking spot 7 as objective.

The run times ranged from 3.1882s to 5.4917s. The standard deviation of the run time for a given parking spot was below 0.25 seconds, except for an outlier at 0.4625s for spot 15. The boxplots of the run times for all parking spots can be seen in Figure 4-5. The run time can be divided into the run time of the initialization, the waypoint determination, the navigation planning and the manoeuvre planning phases. An overview of the breakdown of the run time for all the runs can be seen in Figure 4-6.

The initialization time had negligible values under $5 \cdot 10^{-3}$s for almost all runs. There were a small number outliers in the first runs of the first parking spot, but even these were under 0.025s. The part of the run time dedicated to obtaining the waypoints ranged from 2.6488s to 3.6504s. The waypoint determination time was quite consistent for any given parking spot, except for spots 1, 9 and 15, as shown in Figure 4-7.

The navigation planning time was consistent for individual spots, with standard deviations under 0.2s, but it varied from a minimum of 0.3638s to a maximum of 2.4039s in different parking spots. This difference can be explained by the difference in length of the navigation part of the motion planned, and the increased number of steps that the planner takes for local planning when turning around corners in certain cases. The boxplots of the navigation planning times for all parking spots can be seen in Figure 4-8.

The manoeuvre planning time was consistent across all parking spots, ranging from 0.1676s to 0.6340s with a standard deviation for each spot below 0.15s. The consistency is expected due to the similarity between all the manoeuvres. The boxplots of the manoeuvre planning times for all parking spots can be seen in Figure 4-9.

4-2 Sensitivity

This section addresses the sensitivity of the motion planner. Subsection 4-2-1 addresses the sensitivity to changes on the initial state. Subsection 4-2-2 presents the effect of using vehicle models of different car models. Finally, Subsection 4-2-3 compares the three different optimization algorithms available for `fmincon`.

4-2-1 Initial State

To assess the sensitivity of the planner to changes on the initial state, the planner was tested adding uniformly distributed noise to the initial state's position and orientation coordinates. The position noise ranged from -0.25 to 0.25 meters and the orientation noise ranged from -0.1 to 0.1 radians. The parking spots 4, 5, 6, 16, 17 and 18 were used for the testing. Twenty motions were planned for every goal parking spot, adding to a total of 120 runs.

All the 120 runs resulted in motions that satisfied all the constraints. In addition, the paths of the motions for a given parking spot did not differ much. The navigation waypoints obtained in some of the instances were not exactly the same due to the difference in initial state. Nonetheless, the difference between the waypoints was at maximum 0.5 meters, resulting only in similar differences in the motion paths. The waypoints and paths of the 120 runs can be seen in Figures Figure 4-10 to Figure 4-15.

The changes in initial state did have an effect on the script run time. While the mean run times for the different goals remained similar to those presented in Section 4-1, the number of

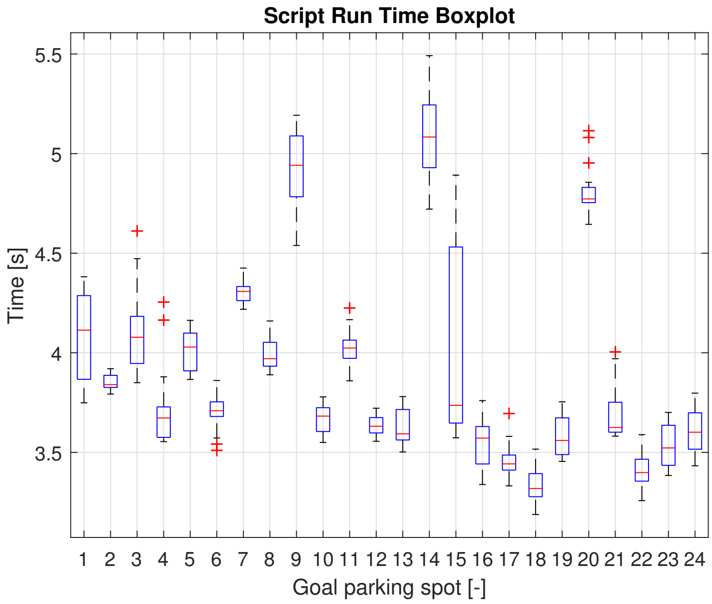

Figure 4-5: Boxplots of the script run time for all goal parking spots.

Figure 4-6: Script run time breakdown into initialisation, waypoint planning, navigation planning and manoeuvre planning. The initialisation time is so small it is not visible in the Figure.

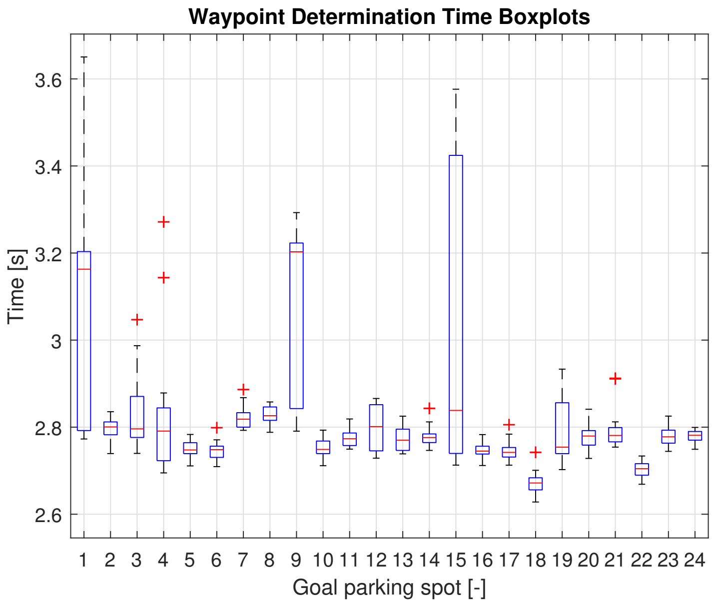

Figure 4-7: Boxplots of the waypoint determination time for all goal parking spots.

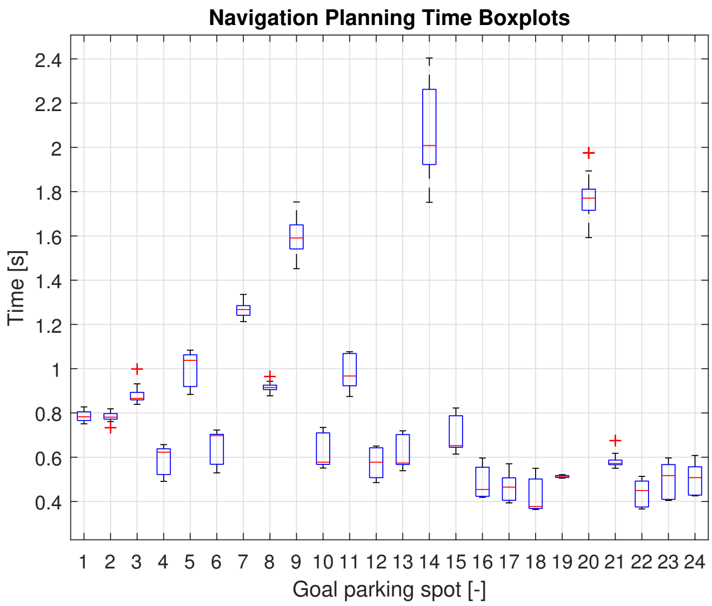

Figure 4-8: Boxplots of the navigation planning time for all goal parking spots.

outliers with much longer run times increased. The boxplots of the run times for the different goal parking spots are shown in Figure 4-16.

The differences in the script run time were not caused by the waypoint determination time, shown in Figure 4-17. When compared to the results in Section 4-1, the waypoint determination times are very similar. Neither were they caused by a difference in the manoeuvre planning times, shown in Figure 4-18, which were also very similar to the ones presented in

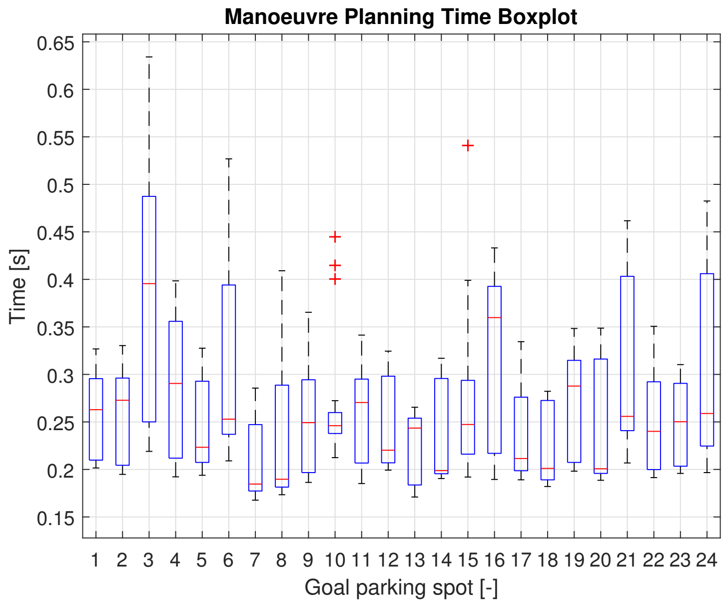

Figure 4-9: Boxplots of the manoeuvre planning time for all goal parking spots.

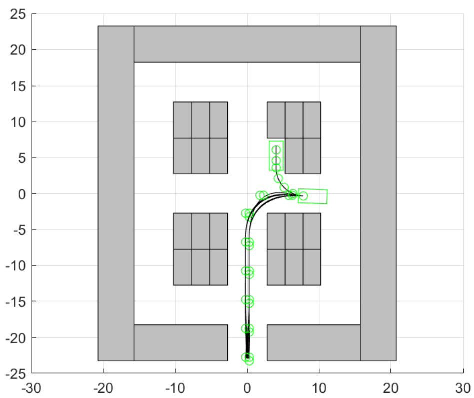

Figure 4-10: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 4 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

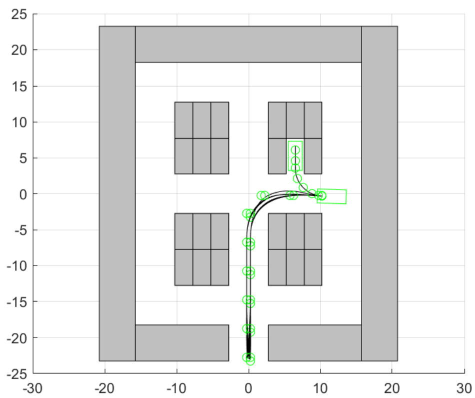

Figure 4-11: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 5 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

Section 4-1. This was expected, since the planning of the manoeuvre is only affected by the initial state of the manoeuvre, and the cost of the final local planning step of the navigation part of the motion was always under the desired 0.01 threshold.

The script run time outliers were caused by outliers in the navigation planning time. The

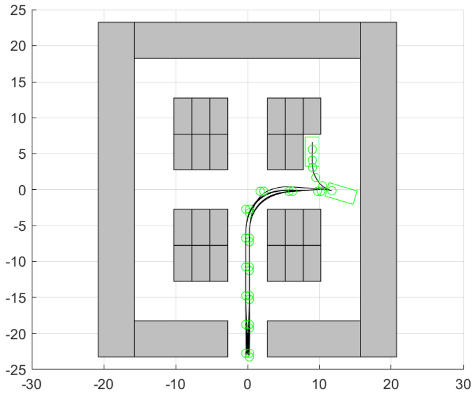

Figure 4-12: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 6 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

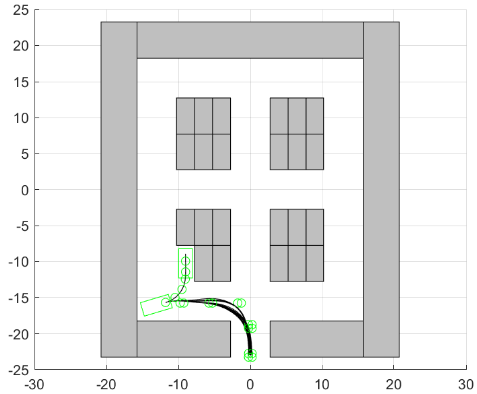

Figure 4-13: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 16 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

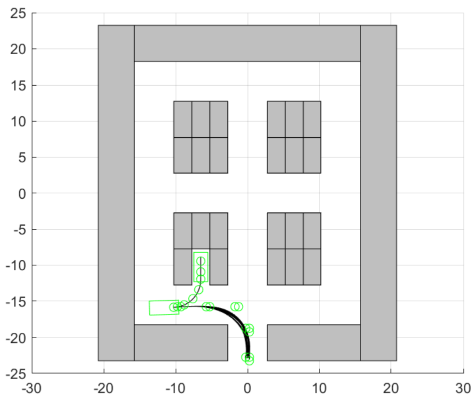

Figure 4-14: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 17 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

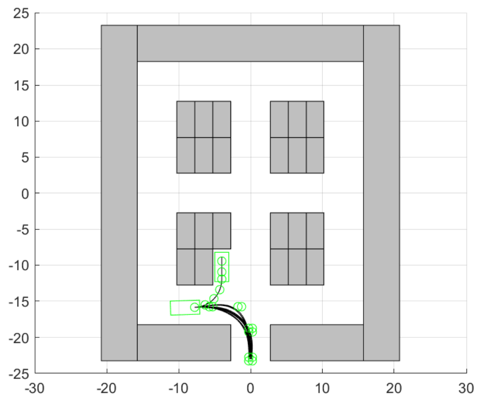

Figure 4-15: In black, the position of the vehicle c.g. along 20 motions planned with different initial conditions with parking spot 18 as goal state. The green circles are the waypoints used as goal states in the intermediate local planning steps. The rectangular green vehicle outlines are the final state references for the navigation and manoeuvre parts of the motion.

boxplots of the navigation planning times are presented in Figure 4-19. The differences in the initial states resulted in very long local planning steps due to the occurrence of obstacle

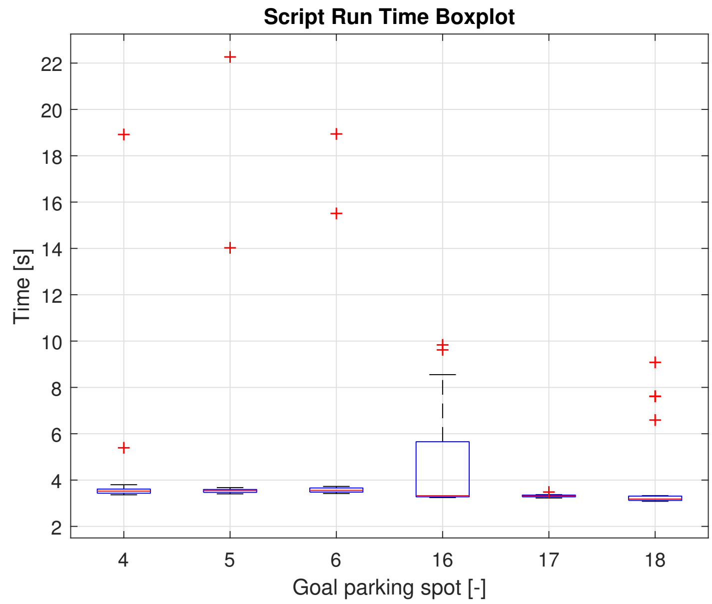

Figure 4-16: Boxplots of the script run times in the initial state sensitivity test runs.

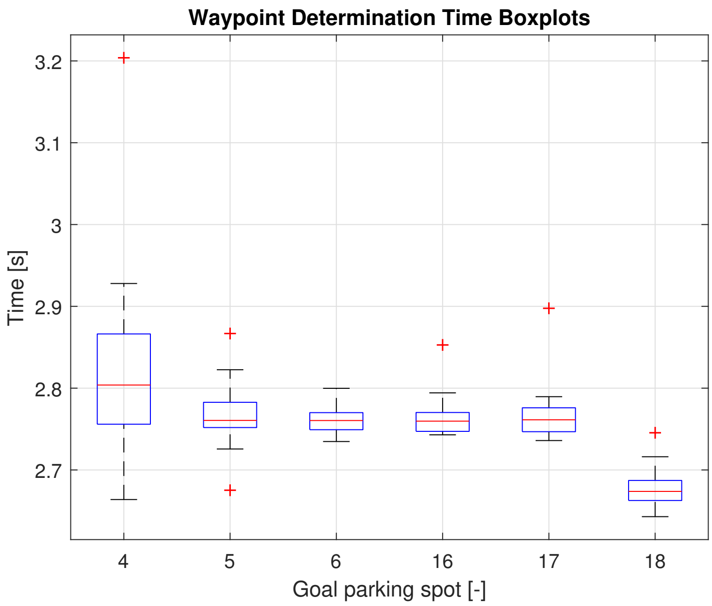

Figure 4-17: Boxplots of the waypoint determination times in the initial state sensitivity test runs.

collisions when planning local motions around corners. For these particular cases, the steps described in Subsections 3-4-7 and 3-4-8 did not result in a local motion satisfying the constraints, even when the horizon planning was increased to its maximum allowed size and the obstacle constraint was considered. These planning issues were solved in the following local planning steps of the same script run. In these future local planning steps the waypoints further down the path were used as reference state and the planning horizon spanned the

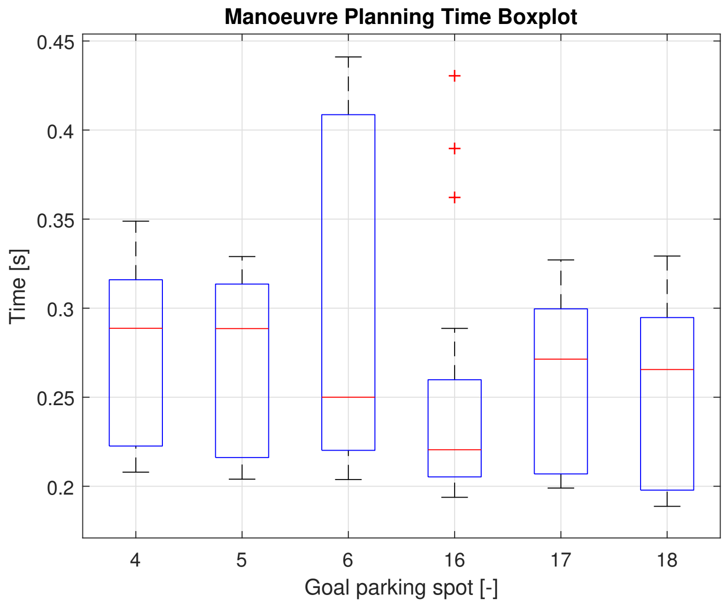

Figure 4-18: Boxplots of the manoeuvre planning times in the initial state sensitivity test runs.

entire turn, allowing the planner to solve the local planning problem, and resulting in overall motions satisfying al the constraints.

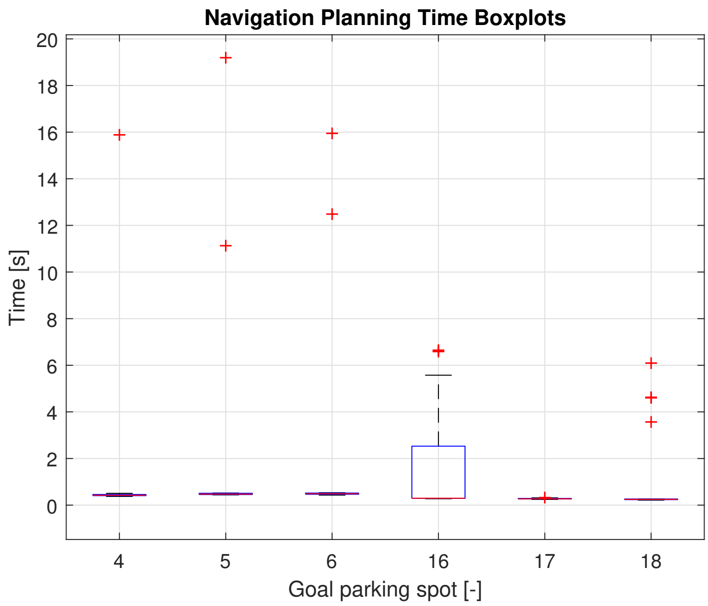

Figure 4-19: Boxplots for the navigation planning time in initial state sensitivity test runs.

4-2-2 Vehicle Model

In order to ensure that the planner works for different car models, the test performed in Subsection 4-2-1 was repeated for two different vehicle models: a Smart ForTwo and a Land Rover Defender. Figure 4-20 shows the size differences between the two vehicle models and the SEAT Ibiza used in Subsection 4-2-1.

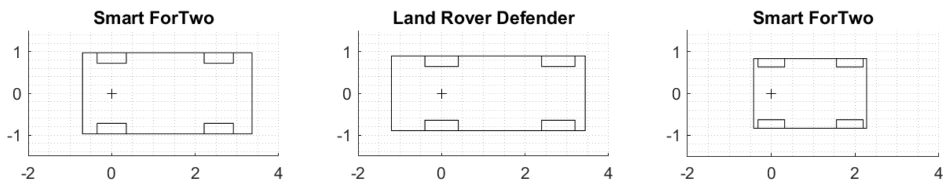

Figure 4-20: Top view of the vehicle models. The center of the rear axis, the point at which the position of the vehicle is measured, is marked with a cross.

As expected, the planner had no issues with Smart ForTwo. Its small dimensions and minimum turn radius make it easy for the planner to avoid collisions and follow the waypoints. All the 120 runs of the planner resulted in a motion that satisfied all constraints. Moreover, the script run times were more consistent than the run times obtained in Subection 4-2-1 with the SEAT Ibiza model. The boxplots of the run times per goal parking spot are presented in Figure 4-21.

Even planning for larger vehicles like the Land Rover Defender, which is slightly thinner but longer and with a larger minimum turn radius than the SEAT Ibiza, posed no issues to the script. As with the Smart ForTwo and the SEAT Ibiza, all the 120 runs of the planner resulted in a motion that satisfied all constraints. Moreover, the script run times, shown in Figure 4-22, had less variability than the results with the SEAT Ibiza. This seemingly counter-intuitive result can be understood by looking at Figure 4-20. While the Land Rover Defender is longer than the SEAT Ibiza, the distance from the rear axis to the front of the vehicle is almost the same. It is this distance, rather than the complete vehicle length, that has a predominant effect when planning forward motions like the navigation part. This fact, together with the larger width of the SEAT Ibiza explains why the planner had less run time variability.

Only when the vehicles are too big to fit in the parking spots, or to turn around corners without colliding with the obstacles does the planner encounter serious issues. In these cases it is not capable of obtaining motions that satisfy the constraints. However, the situation with smaller parking spots than the vehicle is a trivial infeasible solution and the lane width

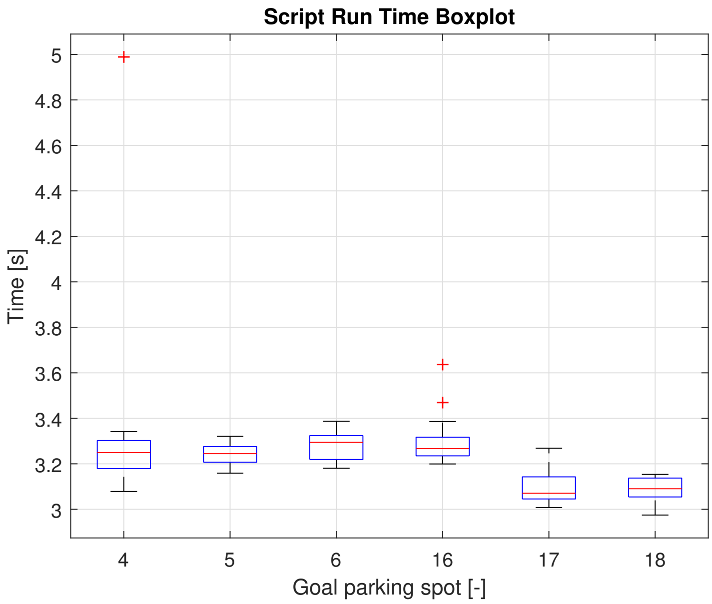

Figure 4-21: Boxplots of the script run times with a Smart ForTwo vehicle model and initial state noise.

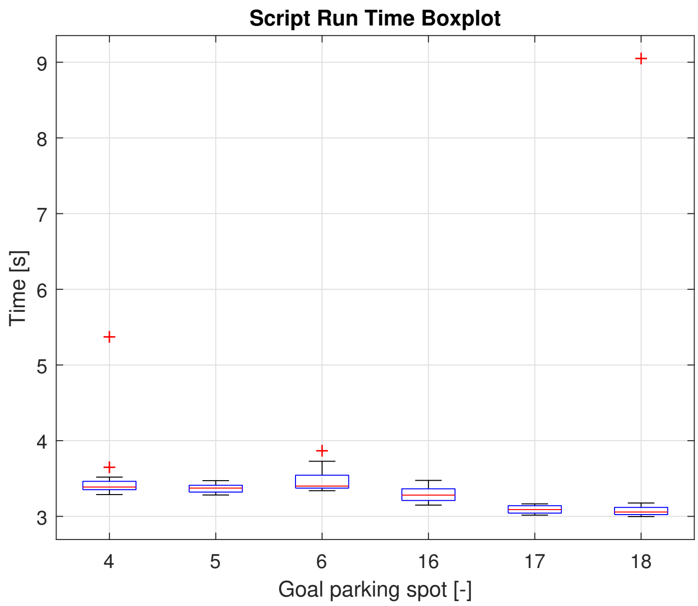

Figure 4-22: Boxplots of the script run times with a Land Rover Defender vehicle model and initial state noise.

must be well below the recommended design values [17] to affect the planner.

4-2-3 Optimization Algorithm

Three different algorithms are available for use with the MATLAB function `fmincon`. These are `interior-point`, `sqp` and `active-set`. The three were tested in the final implementation of the script to determine which one is best suited for the application. The test for each algorithm was the same: 10 runs with initial state noise as in Subsection 4-2-1 for every one of the 24 parking spots.

The results showed `sqp` to be the best suited algorithm. It managed to reach the goal state every single time while satisfying all the constraints, as opposite to Interior-Point and Active-Set, which obtained paths with collisions or that did not satisfy the final error bounds. Moreover, the script run time of `sqp` was considerably shorter and more consistent than the run times of the other two algorithms, as can be seen in Figure 4-23.

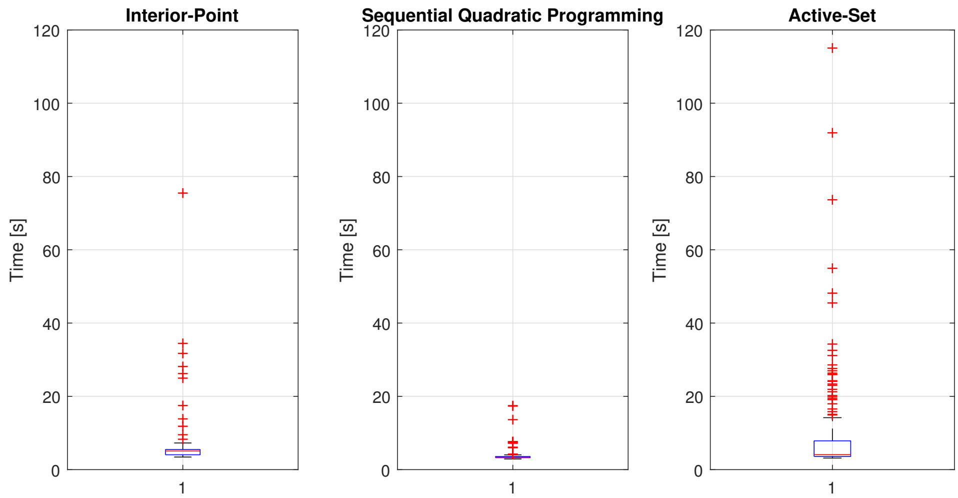

Figure 4-23: Boxplots of the script run times for the `interior-point`, `sqp` and `active-set` algorithms.

Chapter 5

Conclusions and Recommendations

The thesis conclusions and further work recommendations are presented in this chapter. 5-1 evaluates the results of the thesis work in comparison to the goals that were set. Section 5-2 addresses the shortcomings of the motion planning script and points in the directions the author believes more work should be done.

5-1 Conclusions

This section contains the conclusion of the thesis report. It presents the thesis goals stated in Section 1-2 and evaluates whether they have been fulfilled or not.

Motion Planner

The thesis work shall result in a script that is capable of performing motion planning for a non-holonomic vehicle. The script must find a control history $u(t)$ that results in the vehicle state trajectory $q(t)$ from an initial state q_0 to a reference state q_{ref}.

The thesis work did indeed result in a script capable of performing all the aforementioned tasks. However, the control history and state trajectory obtained by the script are not time continuous, but time discrete. The name of the script is `parkingMotionPlanner` and can be found with the deliverables of this thesis.

Motion Stability

The motion planner shall guarantee the stability of the motion. All states of the vehicle model shall remain bounded during the planned motion.

The vehicle states remained bounded during the planned motions addressed in Chapter 4. Because of the finite length of the planning horizon, and the limits on the control history, it is technically impossible for the states to become unbounded. However, it is not impossible that the states become such that they result in an infeasible motion. Nonetheless, this behaviour has only been observed when the motion planning script is not tuned correctly.

Collision Avoidance

The motion planned shall not result in the collision of the vehicle with any obstacles. Only static obstacles shall be considered.

Collisions with static obstacles have been accounted for in the script, added as constraints to the motion. Provided that the script is properly tuned, the planner successfully obtains motions that avoid such collisions.

Non-Holonomic Constraints

The planned motion shall respect the non-holonomic vehicle constraints.

Through the use of the kinematic vehicle model from Equation 3-1, the non-holonomic constraints of the vehicle, present in the model, are respected.

Performance Criteria

The motion planner shall ensure that the error between the reference state q_{ref} and the final state of the state trajectory is smaller than a certain threshold. The thresholds are set to 0.1 meters on the x and y position of the vehicle on the plane, 0.1 radians on the vehicle heading, 0.1 meters per second on the vehicle velocity and 0.1 meters per second squared on the vehicle acceleration.

The final error of the motion planner satisfies the required values. The cost function value, which is the norm of the state error vector is required to be below the 0.01 threshold, so all individual errors are below this value as well.

Safety and Comfort

The resulting motion shall limit the vehicle velocity such that the vehicle is capable of short distance emergency braking. In addition, the longitudinal and lateral acceleration and jerk values of the vehicle shall be limitted such that any passengers inside the vehicle do not experience discomfort. The velocity limit is set to 5 meters per second. The limit values for acceleration are 1.0 and 0.8 meters per second squared for longitudinal and lateral acceleration respectively [11][12]. The jerk limits are 0.7 and 0.3 meters per second cubed for longitudinal and lateral jerk respectively [11][12].

The required constraints are considered by the planner and are satisfied by the resulting motions.

Real Time Application

The run time of the script shall be such that it can be used in real time. A run time of 1 second is chosen to allow a $1Hz$ *computation frequency for replanning.*

While the total script run time is not short enough to run in real time, the run times are smaller than the duration of the motions planned. This means that if necessary, the planner could run in real time provided it is given a head start with respect to the motion tracker of the vehicle. However, this might be hard to do with the current version of the planner, since the variability in script run times would make it difficult to decide on a value for the head start.

Environment Modelling

The motion planner shall use a model of the environment that includes static obstacles. The environment information will be known a priori, and will not change over time.

The presence of static obstacles has been successfully implemented and modelled in the motion planning script.

Actuator Dynamics

The vehicle model used for the motion planning shall include the dynamics of the actuators.

The actuator dynamics are accounted for by including them in the kinematic vehicle model used, shown in Equation 3-1.

To summarize, the thesis work has resulted in a motion planning script that is fast, but not fast enough for the motion planning to be performed in real time. Nonetheless, the motion planner is capable of obtaining stable, collision free motions that fulfil the specified performance, safety and comfort constraints. In addition, the non-holonomic constraints of the vehicle are also satisfied, and the dynamics of the actuators are considered.

5-2 Recommendations

There are two main areas that would add to the script performance and usefulness if they were worked on. These are the inclusion of dynamic obstacles and the use of a variable planning horizon. These section addresses both of these topics in Subsections 5-2-1 and 5-2-2 respectively. Two other topics related to the thesis work that might result in smaller script improvement are presented in Subsections 5-2-3 and 5-2-4.

5-2-1 Dynamic Obstacles

The majority of vehicles in a parking space are stationed in the parking spots and can thus be considered static obstacles. These are accounted for in the planning script. However, it is not a rare occurrence for more than one vehicle to be navigating or manoeuvring in the parking at the same time. A logical next step in the development of a motion planner for parking spaces is to be able to deal with other moving vehicles in the parking, which requires the consideration of dynamic obstacles.

If the positions in time of the dynamic obstacles are known and considered to be constant, it would be possible to extend the obstacle information structure currently used in the planner so that it would include the static and dynamic obstacle information for every simulation time step of the planning horizon. The collision cost function used in the collision avoidance constraint would then look for overlaps of the vehicle and the obstacle footprints, using the obstacle information pertinent to each individual time step. However, the use of a fixed planning horizon for every planning step might result in the planner not being able to find feasible motions in certain cases. For example, it could happen that at a certain point of the parking navigation segment of the motion, the vehicle must wait for another vehicle that is currently obstructing the lane to manoeuvre into a parking spot. If the planning horizon is not long enough to allow for the vehicle to wait and then still reach the objective waypoint, the planner might fail to find a feasible motion.

5-2-2 Variable Planning Horizon

One of the characteristics of the motions that the script plans is the relatively low and constant velocity at which the vehicle moves during the straight segments. These low speeds are easy to explain. The distance between adjacent waypoints is almost constant and the planning horizon time is fixed. Moreover, not only is the planning horizon time fixed, but it is fixed at a value that allows the planner to perform manoeuvres, like turning around a corner, that are more complex than driving in a straight line. This results in longer planning horizons that in turn result in slower velocities. A variable planning horizon would allow the planner to plan faster velocities in the straight segments, while still being able to plan for more complex segments of the motion. However, the implementation of such a variable horizon is not simple. In fact, it would probably require more computations than a fixed horizon planner, resulting in an increase of the script run time. On the other hand, the resulting motions would be more efficient in terms of motion execution time.

5-2-3 Increased Cost Function Complexity

The cost function used in the motion planning script, presented in Subsubsections 2-2-2 and 3-1-2 is very simple. It works fine in the script but it would be interesting to investigate the effect of more complex cost functions. Especially interesting would be cost functions for which the integral part of Equation (2-2) is non-zero.

5-2-4 Changes in Waypoints

The use of waypoints as goal states for the local planning steps is one of the central points of the motion planner implementation. The contribution to the run time of the process through which the waypoints are determined is significant and might be reducible by using different methods. The process could also be changed in order to take environment factors like lane directions into account. In addition, the effect of the waypoint location on the navigation planning time is also significant, resulting in large variability in script run times. Investigation on these two aspects, the obtention of waypoints and the effect of their position on the planner, would be interesting.

Bibliography

[1] SEAT S.A., "SEAT - New SEAT Ibiza - Specs." http://www.seat.com/carworlds/new-ibiza/specifications/reference.html, 2018.

[2] P. Archambault, D. Tamberrino, *et al.*, "Cars 2025: Vol. 3, monetizing the rise of autonomous vehicles."

[3] Tesla, "Autopilot." https://www.tesla.com/autopilot, 2018.

[4] M. Bertoncello and D. Wee, "Ten ways autonomous driving could redefine the automotive world." http://www.mckinsey.com/industries/automotive-and-assembly/our-insights/ten-ways-autonomous-driving-could-redefine-the-automotive-world, 2018.

[5] B. Van Arem, C. J. Van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 429–436, 2006.

[6] B. Asadi and A. Vahidi, "Predictive cruise control: Utilizing upcoming traffic signal information for improving fuel economy and reducing trip time," *IEEE transactions on control systems technology*, vol. 19, no. 3, pp. 707–714, 2011.

[7] The Ford Motor Company, "Active park assist." http://www.ford.nl/OverFord/Technologie/Bestuurders-Ondersteuning/Active-Park-Assist, 2018.

[8] Bayerische Motoren Werke AG, "Intelligent parking." https://www.bmw.co.uk/bmw-ownership/connecteddrive/driver-assistance/intelligent-parking, 2018.

[9] Jeep, "Parallel and perpendicular park assist." https://www.jeep.com/grand-cherokee/safety-security.html, 2018.

[10] R. Cirera Rocosa, "Motion planning for non-holonomic autonomous vehicles in parking scenarios.".

[11] L. L. Hoberock, "A survey of longitudinal acceleration comfort studies in ground transportation vehicles," *Journal of Dynamic Systems, Measurement, and Control*, vol. 99, no. 2, pp. 76–84, 1977.

[12] H. Suzuki, “Research trends on riding comfort evaluation in japan,” *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 212, no. 1, pp. 61–72, 1998.

[13] N. Subbotina, “The method of characteristics for HamiltonâĂŤJacobi equations and applications to dynamical optimization,” *Journal of Mathematical Sciences*, vol. 135, no. 3, pp. 2955–3091, 2006.

[14] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, “Kinematic and dynamic vehicle models for autonomous driving control design,” in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pp. 1094–1099, IEEE, 2015.

[15] The MathWorks Inc., “MATLAB Optimization Toolbox - Functions - fmincon.” https://nl.mathworks.com/help/optim/ug/fmincon.html, 2018.

[16] The MathWorks Inc., “Constrained Nonlinear Optimization Algorithms - fmincon SQP Algorithm.” http://nl.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html#bsgppl4, 2018.

[17] R. Eger, “Critical design parameters for garages,” *Građevinar*, vol. 65, no. 06., pp. 547–552, 2013.

Glossary

List of Acronyms

3mE	Mechanical, Maritime and Materials Engineering
CACC	Cooperative Adaptive Cruise Control
DCSC	Delft Center for Systems and Control
OCP	Optimal Control Problem
PCC	Predictive Cruise Control
TU Delft	Delft University of Technology