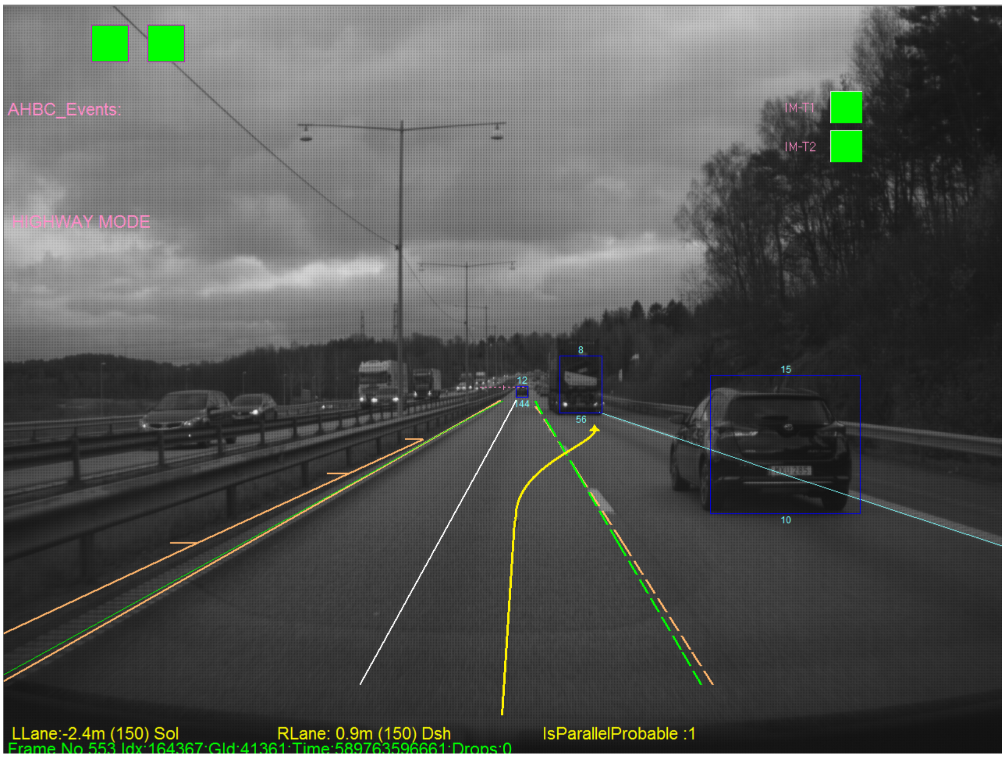
CHALMERS
UNIVERSITY OF TECHNOLOGY



Longitudinal Trajectory Estimation for Lane Change Assist

A comparison of Model predictive control and curve interpolation using smoothing splines in convex optimization

Master's thesis in Systems, Control and Mechatronics

Dandan Ge
Anders Hjelmström Sarvik

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Master's thesis EX054/2018

Longitudinal Trajectory Estimation for Lane Change Assist

A comparison of Model predictive control and curve interpolation using smoothing splines in convex optimization

DANDAN GE
ANDERS HJELMSTRÖM SARVIK

Department of Electrical Engineering
Division of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2018

Longitudinal Trajectory Estimation for Lane Change Assist
A comparison of Model predictive control and curve interpolation using smoothing splines in convex optimization
DANDAN GE
ANDERS HJELMSTRÖM SARVIK

Supervisors: Hanna Nyqvist, Kenny Karlsson, Aptiv
Examiner: Jonas Fredriksson, Department of Electrical Engineering

Master's Thesis EX054/2018
Department of Electrical Engineering
Division of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Typical scenario for a lane change to a slower moving lane on the highway.

Typeset in LaTeX
Gothenburg, Sweden 2018

Longitudinal Trajectory Estimation for Lane Change Assist
A comparison of Model predictive control and curve interpolation using smoothing splines in convex optimization
Dandan Ge
Anders Hjelmström Sarvik
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In this thesis, two algorithms to generate a longitudinal trajectory suitable for a lane change during highway driving are developed and evaluated.

One of the algorithms uses the Model predictive control framework to minimize the acceleration and deviation from the desired position while fulfilling physical and design constraints. The other algorithm uses optimal interpolating B-splines to generate a velocity profile. Both methods work in a receding horizon context which solves an optimization problem in each time instant.

It is shown in simulations that both algorithms successfully generate trajectories suitable for a lane change.

Keywords: Advanced driver-assistance systems, lane change, trajectory planning, Model predictive control, Optimal interpolating splines, B-spline functions.

Acknowledgements

We would like to thank the company Aptiv, who gave us the opportunity to do this thesis and get an insight into the active safety area.

We would also like to thank our two supervisors from Aptiv, Hanna Nyqvist and Kenny Karlsson who have given us support, guidance and their time for valuable discussions.

And thank you, Jonas Fredriksson, for being our supervisor and examiner from Chalmers University of Technology.

Last but not least, we would like to thank our loved ones, for their support during this thesis.

Dandan Ge, Anders Hjelmström Sarvik, Gothenburg, May 2018

Contents

List of Figures

List of Tables

1
Introduction

Since the very first automobiles appeared, work has been done both to prevent accidents and to reduce the injuries caused by a crash. In later years, safety systems have gone from being purely mechanical to becoming more and more electrical and computer controlled. The Advanced Driver Assistance Systems (ADAS) that are offered in cars today include a number of features to make the ride safer and more enjoyable for the driver and passengers. ADAS is a collection of systems and subsystems that is constantly being extended as more systems are being developed. Although working as standalone systems, they are building blocks towards fully autonomous vehicles (called Autonomous Drive, AD) [1]. Some of the systems which are available today are Adaptive Cruise Control (ACC), introduced in 1999, Lane Keep Assist (LKA) in 2001 and Collision Mitigation by Braking (CMbB) in 2003 [2]. Today, the development is progressing quickly, both introducing more advanced systems and making already existing systems more affordable. When these systems where first introduced, they were only available in the most expensive car models, but today you can get the benefits of ADAS in all price ranges. The technology has trickled down to cheaper models when the cost of sensors and microcontrollers has decreased.

The driver assistance systems are not only for preventing or minimizing the risk of accidents. They are also sold as comfort features, giving the driver a chance to relax and making the drive more enjoyable. However, these comfort features can also lead to a safer traffic environment. By taking care of some of the driver's tasks, like keeping the right speed and the vehicle in the correct lane, the driver's stress level and drowsiness can be reduced. With a lower stress level, the driver is less likely to make errors and thus less likely to get involved in an accident [3].

A traditional cruise control is designed to keep a constant speed, regardless of incline changes and other outside disturbances. The ACC systems available today are an evolution of the traditional cruise control, with the addition to adjust the speed depending on the traffic ahead. If the host vehicle, travelling at its user-defined speed, catches up with a slower vehicle in the same lane, it is designed to slow down in a comfortable way and keep a predefined distance to the vehicle in front. If the preceding vehicle later exits the lane, the host vehicle accelerates back up to the desired speed again.

As it is designed, the ACC only keeps track of one target vehicle at a time, the vehicle in front of and in the same lane as the host vehicle. When either the target or the host vehicle makes a lane change, the ACC drops the old target and searches for a new one. Depending on when this target shift occurs, different behaviours can be expected. A late target shift and a new slower target, can create a high difference in speed with a short distance between the vehicles, leading to the ACC either braking hard or aborting functionality, leading to an emergency brake instead. For a driver to change to a slower moving lane safely, most ACC systems require the driver to turn off the ACC, manually slow down, make the lane change and then turn on the ACC again. Otherwise the driver would enter the slower moving lane without having slowed down at all.

This thesis compares two different methods to generate a longitudinal trajectory for lane changes when moving into a lane with slower moving traffic. The trajectory needs to fulfill requirements regarding comfort and safety distances to surrounding vehicles. The first method uses a Model Predictive Control (MPC) framework which aims to minimize the acceleration while still fulfilling certain constraints. MPC has been used successfully in similar but not identical scenarios [4]. The second method uses optimal vector smoothing splines with constraints to generate a curve that acts as a velocity profile. This approach is usually used for complex path planning [5], but is now reworked and applied to the problem at hand. The developed systems will be referred to as a Lane change assistance system, LCA.

1.1 Aim

This project aims to develop and evaluate algorithms using two different methods for generating a longitudinal control suitable for a lane change. In the specified scenario, the host vehicle is travelling at higher speed than the traffic in the adjacent lane. The aim is to adjust the speed in a safe and comfortable manner to match the speed in the new lane.

The input to the algorithms will be the type of data that is available from a vehicle's on board sensors such as position, velocity and acceleration of both the host vehicle and the surrounding vehicles. The output will be a longitudinal trajectory that contains an acceleration or velocity profile which, if followed, will guide the host vehicle to the correct speed. For an overview of the system, see Figure 1.1 were this thesis handles the orange box, Trajectory planning algorithm.

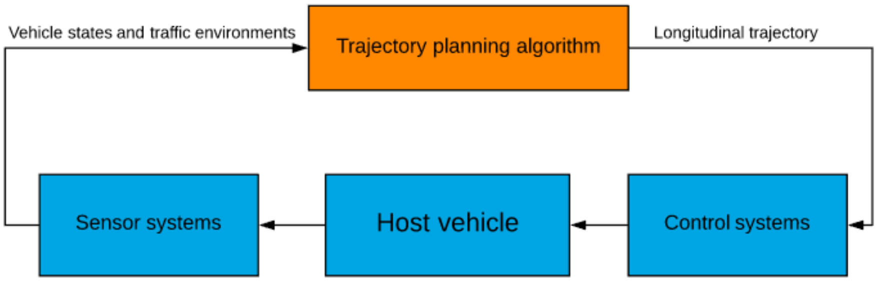

Vehicle states and traffic environments
Trajectory planning algorithm
Longitudinal trajectory
Sensor systems
Host vehicle
Control systems

Figure 1.1: System architecture

1.2 Problem formulation

The problem of this project is, as mentioned earlier, to develop and implement two algorithms to estimate a longitudinal trajectory for highway lane changing. When a lane change is initialized, the LCA system needs to calculate a longitudinal trajectory based on the current speed and the speed of the vehicles in the new lane.

The following interim targets need to be fulfilled to reach the overall aim:

- Two trajectory planning algorithms will be developed
- The developed algorithms will be evaluated using simulations of 5 variations of the same scenario
- A set of performance indicators will be used to compare the performance of the two methods

1.3 Limitations and assumptions

A number of limitations and assumptions have been set to make this project feasible within reasonable time. The following statements are true for all cases in this project.

- All sensor data are already processed by on-board computers of the host vehicle, to get information about the host and surrounding vehicles. Available signals include positions, velocities and accelerations
- A control system exists to perform the lane change with the generated trajectory as reference
- The considered scenarios are on a highway with two lanes of traffic travelling in the same direction
- The surrounding vehicles are assumed to travel at constant speed and in a predictable manner
- The driver is responsible for any lateral motion, the thesis only deals with longitudinal planning
- Only front on-board sensing system is available.

1.4 Methods

During this project, two different methods will be used to solve the trajectory generation problem. As mentioned before, one of the methods will be based on MPC, and the other will be based on smoothing splines. Both methods formulate a convex optimization problem but the cost functions are derived differently. The algorithms will be tested on predefined scenarios in a simulation environment. They will be evaluated based on measurements associated to comfort and how much the host vehicle affects other road users. All development and evaluation will be done using MATLAB.

1.5 Thesis outline

Chapter 1 - *Introduction* describes background, problem formulation, limitations and assumptions, aim and method of this project.
Chapter 2 - *Theory and method* describes theories and methods shared between the two methods.
Chapter 4 - *Model predictive control* is the main chapter for the method using MPC for trajectory planning. All details regarding this model are explained here.
Chapter 3 - *Curve interpolation using smoothing B-splines* explains the theory and implementation for using B-splines in the LCA trajectory generation.
Chapter 5 - *Results* explains how the results are evaluated, and the resulting evaluation. Data are presented in plots and tables.
Chapter 6 - *Discussion* discusses and elaborates on drawbacks and advantages as well as future improvements for the two methods.
Chapter 7 - *Conclusion* contains a short summary of the outcome of the thesis.

2
Theory and method

In this chapter, theory and methods shared between the MPC and B-spline method are presented.

The concept of a trajectory as used in this thesis and the coordinate system used are explained. For longitudinal simulations, a point-mass vehicle model is used in both methods, and a single track model is used in MPC for lateral simulation of the driver action, both of which are described in Section 2.3. The theory behind convex optimization together with quadratic programming presented, as well as how soft constraints can be implemented in optimization problems.

The objective for the generated trajectory is to estimate a safe and comfortable way to get the host vehicle from the current position and velocity to a desired position with a desired velocity. In other words, to go from following a car in the current lane to following a slower car in the adjacent lane. The output trajectory, a sequence of acceleration values, will be the input to the car's ACC. This chapter also describes the general scenario that the LCA system is designed for, how the position constraints are calculated and the three different phases of the lane change.

2.1 Definition of trajectory

When planning the motion of an autonomous car, or for a driver assistance system, the concepts of paths and trajectories are used. A path is defined as the track which the controlled object is intended to follow. For example the center of the lane on a highway. In this regard, a path is a pure geometric description of motion [6]. Path planning is done both on a global and more local scale. For an autonomous car, a global path can be the route from the user's home to the office, while the local path planning takes care of positioning the vehicle in the upcoming roundabout or similar scenarios with a much shorter planning horizon.

A trajectory is meant to work as a reference input to the control system which controls the object. The difference from the path is that the trajectory also contains a timing law [6]. This specifies at what time the object should be in a certain position, or at what velocity and acceleration it will be travelling with. The transitions described by the trajectory need to fulfill the motion laws required by the object. For a car there will be physical limits on, for example the acceleration, and it will not be possible to travel sideways. A motion model can handle this by linking the

lateral and longitudinal motions, this is explained in Section 2.3.
The terms path and trajectory are often used analogously, but in this thesis the distinction described above are adopted.

2.2 Vehicle coordinates and size

To model the motion of a vehicle, a coordinate system needs to be defined. Local coordinates will be used, with origin fixed at the center of gravity of the host vehicle, which is assumed to be in the geometric center. This is a Vehicle Fixed Coordinate system, which implies that the surrounding vehicles' positions and velocities are relative to the host vehicle. The axis are defined according to the ISO8855 standard as is shown in Figure 2.1. The x-axis points in the vehicle's forward direction and the y-axis points to the right, perpendicular to the x-axis.

For an on-board sensor system such as radar and camera, it is hard to calculate the size of surrounding vehicles without high uncertainties. For this thesis, all cars, including the host vehicle, are assumed to be 4.75 m long and 2 m wide, this corresponds well with an average car. The lateral and longitudinal distance to the surrounding vehicles are measured to the center of their rear bumper.

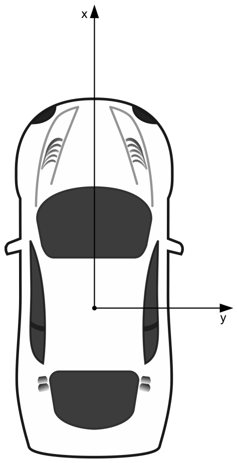


Figure 2.1: The vehicle coordinate system is defined with the x-axis pointing along the vehicle, and the y-axis perpendicular to the right.

2.3 Vehicle motion model

In order to plan a vehicle to follow a desired trajectory, an accurate mathematical motion model for the vehicle dynamics of the host vehicle is very important. There are several different vehicle modeling methods, such as single-track model, double-track model and point mass model. A comparison and evaluation of different motion models is available in [7] In this project, both point mass model and single-track model are used for longitudinal and lateral dynamic simulation of the driver.

2.3.1 Point-mass vehicle model

In the point mass model, the vehicle is considered to be a particle with finite mass and zero dimension. The kinematic equations for the point mass model are as follows:

$$a(k+1) = a(k) + j(k)h \tag{2.1a}$$

$$v(k+1) = v(k) + a(k)h + \frac{1}{2}j(k)h^2 \tag{2.1b}$$

$$x(k+1) = x(k) + v(k)h + \frac{1}{2}a(k)h^2 + \frac{1}{6}j(k)h^3 \tag{2.1c}$$

where j is the rate of acceleration, that is the derivative of acceleration with respect to time, here j is the constant jerk. a denotes the acceleration, v denotes the velocity and x denotes the position.

A state space representation for longitudinal and lateral motion can be generated using the (2.1a)-(2.1c) equations:

$$\begin{bmatrix} x(k+1) \\ v_x(k+1) \\ a_x(k+1) \end{bmatrix} = \begin{bmatrix} 1 & h & \frac{h^2}{2} \\ 0 & 1 & h \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ v_x(k) \\ a_x(k) \end{bmatrix} + \begin{bmatrix} \frac{h^3}{6} \\ \frac{h^2}{2} \\ h \end{bmatrix} j_x(k) \tag{2.2a}$$

$$\begin{bmatrix} y(k+1) \\ v_y(k+1) \\ a_y(k+1) \end{bmatrix} = \begin{bmatrix} 1 & h & \frac{h^2}{2} \\ 0 & 1 & h \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y(k) \\ v_y(k) \\ a_y(k) \end{bmatrix} + \begin{bmatrix} \frac{h^3}{6} \\ \frac{h^2}{2} \\ h \end{bmatrix} j_y(k) \tag{2.2b}$$

where h is the sampling time. x and y represent longitudinal and lateral position, v_x and v_y denote the longitudinal respective lateral velocity, a_x and a_y denotes the longitudinal and lateral acceleration and j_x and j_y are longitudinal and lateral jerks.

2.3.2 Single-track vehicle model

The singel track model is a two degrees of freedom model for lateral dynamics which contains the vehicle lateral position and the vehicle yaw angle ϕ. The vehicle lateral position is measured along the lateral axis of the vehicle to the point O that is the center of rotation of the vehicle. The vehicle yaw angle ϕ is measured with respect to the x-axis. see Figure 2.2. The standard form of the model is introduced in [8]

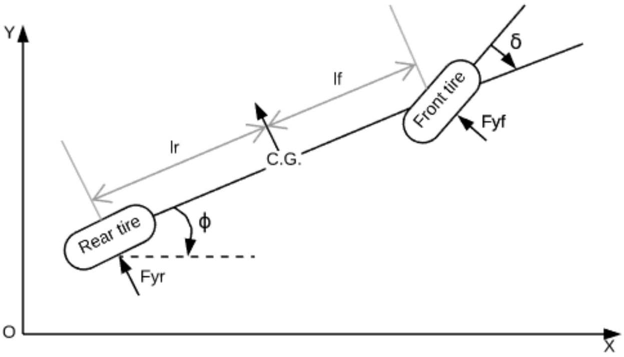


Figure 2.2: Lateral vehicle dynamics

The model is described as follows:

$$m(\ddot{y} + v_x\dot{\phi}) = F_{y_r}cos\delta + F_{x_r}sin\delta + F_{y_f}cos\delta + F_{y_r}sin\delta \tag{2.3a}$$

$$I_z\ddot{\phi} = l_f F_{x_f}cos\delta + lfF_{x_f}sin\delta - l_rF_{y_r}sin\delta - lrF_{y_r}cos\delta \tag{2.3b}$$

where m is the vehicle mass. y is the lateral position. F_{y_f} and F_{y_r} are the front and rear lateral tire forces. I_z is the moment about the z-axis. v_x denotes the longitudinal velocity of the vehicle at the center gravitation. δ is the front wheel steering angle. l_f and l_r are the distances of the front tire and the rear tire respectively from the c.g of the vehicle.

Assume that the steering angle δ is small and neglecting the longitudinal force F_{x_f}, the motion model can be simplified to:

$$m(\ddot{y} + v_x\dot{\phi}) = F_{y_r} + F_{y_f} \tag{2.4a}$$

$$I_z\ddot{\phi} = lfF_{y_f} - l_rF_{y_r} \tag{2.4b}$$

For small tire slip angles, the lateral tire forces can be approximated as a linear function of tire slip angle. The front and rear tire forces and tire slip angles are defined as follows:

$$F_{y_f} = C_f(\delta - \frac{\dot{y} + l_f\dot{\phi}}{v_x}) \tag{2.5a}$$

$$F_{y_r} = C_r(\frac{-\dot{y} + l_r\dot{\phi}}{v_x}) \tag{2.5b}$$

The state space form of the single-track motion model then becomes:

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{C_f+C_r}{mv_x} & 0 & -v_x - \frac{C_fl_f-C_rl_r}{mv_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{C_fl_f-C_rl_r}{I_zv_x} & 0 & -\frac{C_fl_f^2+C_rl_r^2}{I_zv_x} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \varphi \\ \dot{\varphi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{m} \\ 0 \\ \frac{C_fl_f}{I_z} \end{bmatrix} \delta \tag{2.6}$$

C_f and C_r are the cornering stiffness of each front tire and rear tire respectively.

2.4 Convex optimization and Quadratic Program Formulation

The two methods used in this thesis both formulate the trajectory generation problem as a convex optimization problem. The standard form convex optimization problem is formulated as

$$\min_{x \in \mathbf{R}^n} \quad f(x) \tag{2.7a}$$

$$Ax = b, \tag{2.7b}$$

$$\text{s.t.} \quad g_i(x) \leq 0, \quad i = 1, ..., m \tag{2.7c}$$

where $A \in \mathbf{R}^{p \times n}$, $b \in \mathbf{R}^p$, and $f : \mathbf{R}^n \to \mathbf{R}$ and $g_i : \mathbf{R}^n \to \mathbf{R}$ are convex functions.

Given an objective function $f(x)$ which is quadratic in terms of its variables, x, and a set of constraints, it can be formulated as the standard quadratic programming problem (QP):

$$\min_{x \in \mathbf{R}^n} \quad f(x) = \frac{1}{2} x^T G x + x^T c \tag{2.8a}$$

$$\text{s.t.} \quad Ax = b \tag{2.8b}$$

$$Cx \leq d \tag{2.8c}$$

where $C \in \mathbf{R}^{m \times n}$ and $d \in \mathbf{R}^m$. A QP problem can always be solved or shown to be infeasible, but the number of computations depends strongly on the number of decision variables and the characteristics of the objective function. If G is positive semidefinite, (2.8a) is a convex QP, and strictly convex if G is positive definite. For the convex problems, local optima are also global optima, which means the computational process can be terminated when the first optima is found, knowing it is a global optima. This means that the convex optimization problems can be solved more efficiently than nonconvex problems.

There are several both commercial and free solvers applying different methods to solve QP problem. Some examples are CPLEX, written in C but with API for several languages [9], and MATLAB's `quadprog` [10], which will be used in this thesis.

2.5 Soft constraints and slack variables

If there exists a point $x \in \mathbf{R}^n$ such that,

$$g_i(x) \leq 0, \quad i = 1, \ldots, m \tag{2.9a}$$

$$Ax = b, \tag{2.9b}$$

the optimization problem (2.7a) is said to be *feasible*, and *infeasible* otherwise. Feasibility is thus a necessary (but not sufficient [11]) condition for the existence of a solution to the optimization problem.

For complicated optimization problems, it can be difficult to decide a priori if a feasible point exists. This can be a severe problem in cases where the optimization procedure has to deliver a result (as in e.g. LCA). In such cases, it would be useful to tell the optimization procedure that it can violate some constraints (of less importance) if the problem would be infeasible otherwise. This is typically implemented using so-called *soft constraints*.

By introducing the slack variables $s \in \mathbf{R}^m$, we rewrite the constraints as,

$$g_i(x) + s_i \leq 0, \quad i = 1, \dots, m \tag{2.10a}$$

$$Ax = b \tag{2.10b}$$

Note that since the slack variables are unrestricted, the modified constraints (2.10) are always feasible provided that A has full row rank (which usually is easy to verify a priori). To enforce that the slack variables are as small as possible, and desirably identical to zero when possible, they are heavily penalized in the objective function. The resulting soft constrained problem becomes:

$$\min_{x \in \mathbf{R}^n} \quad f(x) + \gamma \|s\|_p^2 \tag{2.11a}$$

$$\text{s.t.} \quad g_i(x) + s_i \leq 0, \quad i = 1, ..., m \tag{2.11b}$$

$$Ax = b, \tag{2.11c}$$

where $\gamma > 0$ is large and typically $p = 1, 2$.

2.6 The general scenario

All evaluation and testing will be done on variations to a general scenario, the variations will include different initial conditions on the host vehicle. The general scenario takes place on a highway. The properties of a highway leads to the curvature being small enough to be neglected, i.e. the road is straight, and the speed varies from 80 *km/h* and upwards. The host vehicle is travelling in the left of two unidirectional lanes. In front of, and behind the host are two vehicles, travelling at the same speed as the host's initial speed and in the same lane. In the adjacent lane, to the right of the host vehicle, are two other vehicles, both travelling at the same speed, which is slower than the speed of the host vehicle. In Figure 2.3, the initial configuration of the lane change scenario can be seen. The travelling direction is to the right, vehicles s_1 and s_2 are travelling at the same initial velocity as the host vehicle (E) and vehicles s_3 and s_4 are travelling at a slower speed. From here on, the notation shown in Figure 2.3 will be used for the surrounding vehicles.

The trajectory should guide the host vehicle to align longitudinally between the two vehicles in the right lane, keeping the same velocity as they have. The two other vehicles in the left lane should be affected as little as possible, but are assumed to act to avoid any accidents.

2.6.1 The different phases

The LCA divides a lane change into three different phases.

- **Phase 1:** Phase 1 is initialized when the driver signals a lane change action, for example by turning on the turn signal. The host vehicle, now in the left lane is allowed to start slowing down for an upcoming gap between vehicles s_3 and s_4. When it is safe to change lanes, the driver is notified by the LCA system.
- **Phase 2:** The trajectory transits into Phase 2 when the driver starts the actual lane change by rotating the steering wheel. If the lane change becomes infeasible before the driver starts the lane change, the driver will be notified of this. The driver thus has some time between the notification that a lane change is safe, until the notification of it becoming infeasible, to start the lane change.
- **Phase 3:** When the lane change is completed, the trajectory is in Phase 3. The vehicle drives in the new lane doing small adjustments to the speed and position to match the two other vehicles, s_3 and s_4.

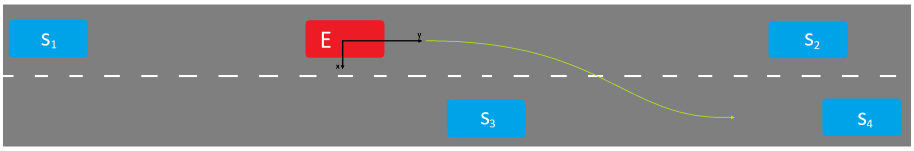


Figure 2.3: Example of initial positions for the vehicles in the lane change scenario

2.7 Safety distance

For a safe behavior, the host vehicle must keep a safe distance to the surrounding vehicles. This way a margin is created for unplanned behavior of the other vehicles such as panic braking or swerves. With a safety distance to each vehicle, a less stressful behavior can be achieved.

The safety distance that the host vehicle needs to keep in front of vehicle s_j is defined as

$$l_f(s_j) = \frac{L_{s_j} + L_E}{2} + \delta_f + t_f max(v_{s_j} - v_E, 0) \tag{2.12}$$

and measured in meters between the vehicles' centers. L_{s_j} and L_E are the length of the two vehicles, δ_f is the minimum fixed offset between the vehicles, which will be kept if they are travelling at the same speed. t_f is the minimal time gap which the host vehicle must maintain to s_j with respect to s_j's front, assuming constant velocities, v_E and v_{s_j}. The safety distance behind vehicle s_j is defined in an analogous way but the sign of the difference in velocity has been changed

$$l_r(s_j) = \frac{L_{s_j} + L_E}{2} + \delta_r + t_r max(v_E - v_{s_j}, 0) \tag{2.13}$$

2.8 Receding horizon optimization

With the receding horizon, the algorithms take the current sensor measurements into account along with what can be predicted to happen within the prediction horizon to optimize the trajectory. The trajectory is then fed to the vehicle's control system as a reference for one sample period. After the vehicle travelled the first segment of the trajectory, the procedure is repeated but with updated sensor measurements and the horizon shifted forward.

For trajectory generation, receding horizon has many advantages, the planning horizon can be relatively short, reducing the computation time. Only following each generated trajectory for a short period of time lets the algorithms compensate for unpredictable behaviour. To generate a 60 second trajectory and then follow it without updating it with regard to what has happened in the traffic would be very dangerous.

One of the major ideas behind MPC is the concept of a receding prediction horizon. However, the concept of a receding horizon can be used outside of control theory and in this thesis both developed methods use a receding prediction horizon.

3
Curve interpolation using smoothing B-splines

This chapter explains how splines can be used for smooth curve interpolations, and how this will be used to generate the desired trajectory. First the theory behind splines is explained, followed by the approach taken to use splines as a method of trajectory generation. For the LCA system, a specific type of spline, B-spline, will be used, and for the sake of simplicity this method will be called the B-spline method in this thesis.

3.1 Spline

A common method to design trajectories is to use one of several ways of curve interpolation. This can be as simple as connecting straight lines to circle segments and ellipsoids to get the desired trajectory, or more advanced methods like using interpolating techniques between waypoints that have been placed along the desired path. The goal with the interpolation is to create a feasible trajectory between the waypoints with respect to continuity in both velocity and acceleration [16].

A spline is a special case of polynomial interpolation were a function is described piecewise by different polynomials. The points where two adjacent polynomials are connected are called knots, and the curve shifts from one polynomial to the next without loss of continuity in the knot. Where a single polynomial interpolation would need a higher degree to describe a complex curve, a spline could give similar result but with multiple lower degree polynomials, lowering the computational complexity.

Depending on the functions or polynomials defining a spline, they have different properties and are given different names such as T-splines, M-splines and B-splines. B-splines will be used in this thesis.

3.1.1 B-spline

A B-spline, or basis spline, is an affine linear combination of basis functions [14], $N_{i,k}$, and some control points, τ_i.

$$S(t) = \sum_i \tau_i N_{i,k}(t) \tag{3.1}$$

The basis functions of degree k decides the continuity properties of the spline and are defined recursively. Let t_i be a bi-infinite and strictly increasing sequence of knots and the recursion formula for the basis functions $N_{i,k}$ becomes

$$N_{i,0}(t) = \begin{cases} 1 & t_i \leq t < t_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

for all i, and

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k} - t_i} N_{i,k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} N_{i+1,k-1}(t) \tag{3.3}$$

The basis functions can be seen in Figure 3.1 for different values of k. Regardless of the degree k, they all have the property of unity, $\int_{-\infty}^{\infty} N_{i,k}(t)dt = 1$. From this definition we see that the basis function has the properties of being piecewise polynomial of degree k and is positive in (t_i, t_{i+k+1}) and zero otherwise.

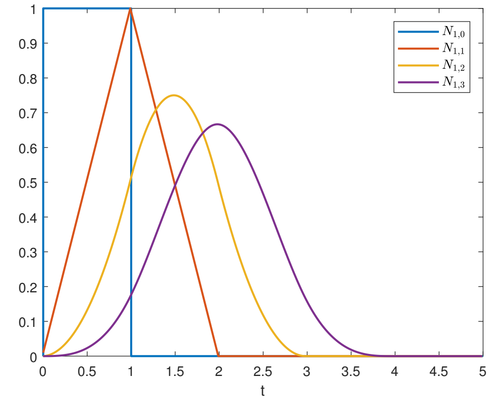


Figure 3.1: Uniform basis functions of different degree, $k = 0, ..., 3$

When constructing the B-spline in (3.1), each polynomial has support from $k+1$ control points at $k+1$ knot locations.

Let the knots be distributed evenly and let α be a normalizing coefficient so that $t_{i+1} - t_i = \frac{1}{\alpha}$ for all i. We then have what is called a normalized uniform B-spline. This simplifies the definition from (3.2) to be over a uniform interval of 1 for each basis function and for this special case we can rewrite (3.1) to

$$S(t) = \sum_i \tau_i B_k(\alpha(t - t_i)) \tag{3.4}$$

with B_k consisting of the normalized uniform basis spline functions [15]

$$B_k(t) = \begin{cases} N_{k-j,k}(t-j) & j \leq t < j+1 \\ & j = 0, 1, ..., k \\ 0, & t < 0 \text{ or } t \geq k+1 \end{cases} \tag{3.5}$$

and $N_{i,k}$ is defined recursively as before (3.2).

For a finite knot sequence, with $k+m$ knots, and the same number of control points, τ_i, the uniform B-spline function is written as

$$S(t) = \sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \tag{3.6}$$

Figure 3.2 shows the building blocks of a cubic B-spline. As can be seen in the figure, at each instance t, $k+1$ basis functions are non-zero, the sum of these are 1. This leads to every control point in (3.6) having positive support from $k+1$ basis functions. As t goes from one knot to the next, the basis functions acts as a parametric curve, shifting the weight between the control points. Note that the B-spline is not defined at the end knots. The curve can be forced to start and end at a control point and with the same tangent as its control polygon by setting the k first and last knots to the same value, $t_1 = ... = t_k$ and $t_{m+1} = ... = t_{m+k}$.

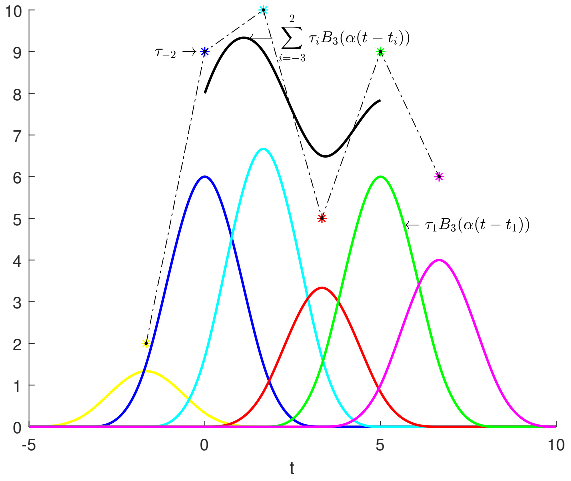


Figure 3.2: Example of a uniform b-spline, decomposed in its different parts, control points, corresponding basis functions and the spline itself

For later reference, the function $B_3(t)$ and its derivatives are shown in Table 3.1

	$B_3(t)$	$B_3'(t)$	$B_3''(t)$	$B_3'''(t)$
$0 \leq t < 1$	$\frac{1}{6}t^3$	$\frac{1}{2}t^2$	t	1
$1 \leq t < 2$	$\frac{1}{6}(-t^3 + 12t^2 - 12t + 4)$	$\frac{1}{2}(-3t^2 + 8t - 4)$	-3t+4	-3
$2 \leq t < 3$	$\frac{1}{6}(t^3 - 24t^2 + 60t + 20)$	$\frac{1}{2}(3t^2 - 16t + 20)$	3t-8	3
$3 \leq t < 4$	$\frac{1}{6}(4 - t)^3$	$-\frac{1}{2}(4 - t)^2$	-t+4	-1
$t < 0, t \geq 4$	0	0	0	0

Table 3.1: Function $B_3(t)$ and its derivatives

3.2 Trajectory planning using B-splines

The idea behind using B-splines for the trajectory planning is to formulate an optimization problem, and place the control points to the splines in an optimal way. Constraints will be introduced on the control points to control the position, the amplitude of the velocity, acceleration and jerk. The trajectory will be designed by placing the control points in the longitudinal direction and time, and the considered derivatives of the trajectory will be velocity, acceleration and jerk. Cubic B-splines are used for the smoothing between the control points which yields a C^2 continuity, the acceleration will hence be piecewise linear and the jerk piecewise constant.

3.2.1 Constructing a trajectory

Given a set of control points, τ_i over a time interval $[t_0, t_m]$ we construct a trajectory describing the position

$$x(t) = \sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \tag{3.7}$$

The trajectory is made up of normalized uniform B-splines, with normalization coefficient α and $m+k$ control points as described in Section 3.1.1. The knots t_i are placed with a uniform distance of α from $[t_{-k}, t_{m-1}]$. Cubic splines are used, which sets $k = 3$.

3.2.2 Defining a cost function

The trajectory constructed in (3.7) depends on the control points τ_i. To be able to place the control points, we need an objective for the trajectory. We introduce a reference, $f(t)$, which will be the desired final position. The reference for the end position is simply centered (longitudinally) between the two vehicles, s_3 and s_4 including the safety distance to each. There will later also be final conditions added for the velocity and acceleration. But the final position is not the only objective we have for the trajectory, for comfort's sake, we also want to penalize unnecessary high accelerations. With $a(t) = p''(t)$ being the trajectory acceleration, we can now define a cost function were the squared acceleration and deviation from the reference position are penalized

$$J(\tau) = \lambda \int_{t_0}^{t_m} ||a(t)||^2 dt + \int_{t_0}^{t_m} ||x(t) - f(t)||^2 dt \tag{3.8}$$

Here $\lambda > 0$ is a weighting parameter and $\tau = [\tau_{-k},\ \tau_{-k+1},\ \ldots,\ \tau_{m-1}]^T$ is the control point vector. Evaluating the first integral term in (3.8) we can rewrite it as a matrix multiplication

$$\begin{aligned}\int_{t_0}^{t_m} ||a(t)||^2 dt &= \int_{t_0}^{t_m} \left(\sum_{i=-k}^{m-1} \tau_i \frac{d^2}{dt^2} B_k(\alpha(t-t_i)) \right)^2 dt \\ &= \sum_{i=-k}^{m-1} \sum_{j=-k}^{m-1} \tau_i \tau_j \int_{t_0}^{t_m} \left(\frac{d^2}{dt^2} B_k(\alpha(t-t_i)) \right) \left(\frac{d^2}{dt^2} B_k(\alpha(t-t_j)) \right) dt \\ &= \tau^T Q \tau \end{aligned} \tag{3.9}$$

Q is a $[(k+m)\times(k+m)]$ matrix with elements $q_{i,j}$ for $i, j = -k, ..., m-1$. Taking the inner derivative into account and introducing a new integration variable, $\hat{t} = \alpha(t-t_0)$ the following expression for $q_{i,j}$ is derived

$$\begin{aligned}\hat{t} =& \alpha(t-t_0) \\ d\hat{t} =& \alpha dt \\ q_{i,j} =& \int_{t_0}^{t_m} \left(\frac{d^2}{dt^2} B_k(\alpha(t-t_i)) \right) \left(\frac{d^2}{dt^2} B_k(\alpha(t-t_j)) \right) dt \\ =& \alpha^4 \int_{t_0}^{t_m} \left(B_k''(\alpha(t-t_i))\right) \left(B_k''(\alpha(t-t_j))\right) dt \\ =& \alpha^3 \int_0^m \left(B_k''(\hat{t}-i) \right) \left(B_k''(\hat{t}-j) \right) d\hat{t} \end{aligned} \tag{3.10}$$

we see from (3.10) that Q will be a symmetric matrix since $q_{i,j} = q_{j,i}$. With help from Table 3.1 we can now establish the Q matrix for cubic B-splines with m control points to be

$$Q = \frac{\alpha^3}{6} \begin{pmatrix} 2 & -3 & 0 & 1 & & & & & & \\ -3 & 8 & -6 & 0 & 1 & & & & & \\ 0 & -6 & 14 & -9 & 0 & 1 & & & & \\ 1 & 0 & -9 & 16 & -9 & 0 & 1 & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & 1 & 0 & -9 & 16 & -9 & 0 & 1 & \\ & & & 1 & 0 & -9 & 14 & -6 & 0 & \\ & & & & 1 & 0 & -6 & 8 & -3 & \\ & & & & & 1 & 0 & -3 & 2 & \end{pmatrix} \tag{3.11}$$

it consists of a $[k \times k]$ matrix in the upper left corner which is mirrored in the lower right corner. In between are a band matrix of size $[m-k \times m-k]$, the rest, portrayed as empty cells, are 0.

The second integral term in (3.8) is expanded to separate terms dependent on τ, τ^2 and those not dependent on τ at all.

$$
\begin{aligned}
&\int_{t_0}^{t_m} ||x(t) - f(t)||^2 dt = \\
&= \int_{t_0}^{t_m} \left(\sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \right)^2 - 2 \left(\sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \right) f(t) + f^2(t) dt \\
&= \int_{t_0}^{t_m} \left(\sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \right)^2 dt - 2 \int_{t_0}^{t_m} \left(\sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t - t_i)) \right) f(t) dt + \int_{t_0}^{t_m} f^2(t) dt \\
&= \tau^T Q_0 \tau - 2g^T \tau + \int_{t_0}^{t_m} f^2(t) dt \qquad (3.12)
\end{aligned}
$$

Now the elements of Q_0 can be calculated using the same method as used in (3.10) for Q but with $B_k(t)$ instead of $B_k''(t)$. For explicit values of Q_0 see Appendix A. The last term is independent from τ and thus not of interest in regard of the cost function in (3.8). In the second term, g is a vector with each element defined as

$$g_i = \int_{t_0}^{t_m} B_k(\alpha(t - t_i)) f(t) dt \qquad (3.13)$$

Since $f(t)$ varies with time, g_i can not be calculated in advance, but needs to be calculated in every iteration.

3.2.3 QP-formulation

Both Q and Q_0 are the same fixed matrices for any problem portrayed as described above. They are also, and will always be, symmetric and positive definite, so it makes sense to reformulate the cost function (3.8) to a QP-problem as described in Section 2.4. Constraints are also added, which will be described in the next section. With $G = \lambda Q + Q_0$ the new cost function and the optimization problem which gives the optimal vector of control points τ becomes

$$
\begin{aligned}
\min_{\tau \in \mathcal{R}} \quad & J(\tau) = \tau^T G \tau - 2g^T \tau && && (3.14a) \\
\text{s.t.} \quad & A_{eq} \tau = b_{eq} && \textit{Boundary conditions} && (3.14b) \\
& A_{ineq} \tau \leq b_{ineq} && \textit{Upper and lower bounds} && (3.14c)
\end{aligned}
$$

3.2.3.1 Constraints

With the cost function defined, the next step is to formulate the constraints. Equality constraints will be used to impose desired initial and final conditions, while inequality constraints will set upper and lower boundaries on τ to control position, velocity, acceleration and jerk.

Since we know the time when both our initial and final conditions need to be active, we can simply use the corresponding knot index in (3.7) to the desired values. For the initial position it would look like this

$$p(t_0) = \sum_{i=-k}^{m-1} \tau_i B_k(\alpha(t_0 - t_i)) = p_{initial} \tag{3.15}$$

and the border conditions for the corresponding derivatives are formulated analogously but with the derivatives of the function $B_k(t)$ instead, depicted in Table 3.1. Most of the terms in the sum in (3.15) will be zero due to the nature of $B_k(t)$, and each equality constraint will consist of a combination of up to 4 control points.

The inequality constraints are formulated in a similar fashion to the equality constraint. The difference being the possibility to set both an upper and a lower constraint, and that they should be valid for more than a fixed time instance. An example of an upper limit to the acceleration is shown below

$$a(t) = \sum_{i=-k}^{m-1} \tau_i B_k''(\alpha(t - t_i)) \leq a_{max}, \quad \text{for } t_m < t < t_0 \tag{3.16}$$

All the constraints are set at the knots, this is not a problem for the boundary conditions, but for the inequalities that spans over several knots, the B-spline might violate the constraints in between the knots. The inequality constraints for the position will be determined by the safety distances as described in Section 2.7, and the rest will be tuning parameters.

The constraints in (3.14b) and (3.14c) can now be completed by formulating (3.15) and (3.16) as vector multiplications and stacking the constraints in two matrices on the form $A_{eq}\tau = b_{eq}$, $A_{ineq}\tau \leq b_{ineq}$.

3.2.4 Changing the constraints

To hinder the traffic behind the host vehicle as little as possible, two levels of acceleration constraints will be used. The goal is to slow down gently when approaching a suitable gap for a lane change. When the driver gets closer and starts the lateral motion, he will transition from Phase 1 to Phase 2 and the limits on the deceleration will be lifted, making it possible to reach the desired speed. The majority of the deceleration is thus moved to the new lane and the the risk of being hit from behind in the original lane is decreased.

However, even before the driver starts turning the wheel, the trajectory planning algorithm needs to know if it's feasible to reach the desired speed within the desired distance. For this reason the latest possible time instance where the lane change is feasible without crossing any safety distances is calculated. If the driver turns the wheel i.e. starts the lateral motion, at or before this time, the lane change is feasible. If this time instance is passed without any interaction from the driver, the lane change will become infeasible and an escape trajectory will be generated instead.

The latest possible brake point is calculated based on the current relative velocity, $v_{rel} = v_E - v_{desired}$, and the safety distance to the car which the host vehicle aims to move in behind, l_r. With the more aggressive deceleration limit, a_{high} and the jerk limit j_{lim} the time needed to reach the desired velocity can be calculated. The acceleration from the current velocity consists of three parts. t_1, the time from applying the brakes until the vehicle reaches the acceleration constraint. $t_1 = \frac{a_{high}}{j_{lim}}$. t_3, the time it takes to release the brakes, which is equal to t_1. Between t_1 and t_3 the vehicle accelerates at the acceleration constraint

$$t_2 = \frac{\frac{a_{high}}{2}t_1 - v_{rel} + \frac{a_{high}}{2}t_3}{a_{high}} \tag{3.17}$$

During the deceleration time $t_1 + t_2 + t_3$, the travelled distance in relation to the leading car will be

$$d_{brake} = t_1 \left(v_{\text{rel}} + \frac{a_{\text{high}}\, t_1}{4}\right) + t_2 \left(v_{\text{rel}} + \frac{a_{\text{high}}\, t_1}{2} + \frac{a_{\text{high}}\, t_2}{2}\right) + t_3 \left(v_{\text{rel}} + \frac{3\, a_{\text{high}}\, t_1}{4} + a_{\text{high}}\, t_2\right) \tag{3.18}$$

Thus, at the current speed, the latest point that the car needs to start decelerating is d_{brake} before the safety distance to the car in front. The time it takes to travel this distance, assuming constant speed, is easily calculated, $t_{brake} = \frac{l_r(S_j) - d_{brake}}{v_{rel}}$. This time is then matched to the closest lower control point which will act as the switch between the two different acceleration constraints. In (3.16), one constraint can be set for $t_i \leq t_{brake}$ and another for $t_i > t_{brake}$.

3.2.5 Knot interval and receding planning horizon

The trajectory is defined by a number of equidistant control points, τ_i, over a time interval $[t_0, t_m]$. The number of control points and the length of the time interval thus decides the distance between the points. In a receding horizon system the knots will be shifted forward in every iteration. If the sensor data is ideal and the shift between every iteration is a multiple of $\frac{t_m - t_0}{m}$, perfect tracking will occur. With a shift of exactly $\frac{t_m - t_0}{m}$, t_0 will be shifted to where t_1 were and so on, only t_m will be placed on a completely new position.

3.2.6 Length of planning horizon

Since the B-spline method optimizes the trajectory for the full lane change in each time instant, the full lane change needs to fit within the planning horizon. The higher the velocity difference between the host and the desired velocity, the longer the planning horizon needs to be. In this thesis, a fixed planning horizon will be used, chosen so that velocity differences over 50 km/h is possible.

4

Model predictive control

This chapter presents the trajectory planning algorithm using MPC, convex optimization and QP. It begins with an introduction part of MPC, and then follows with the actual path planning formulation, this section ends up with an implementation details part that provides the mathematic formulation of the QP problem.

4.1 Background

In this section, we provide a brief description of MPC. For a full introduction see e.g. [18].

MPC is an optimization-based control technique which has received an increasing interest since its discovery in the 1970s. Its popularity is largely stemming from the natural way in which constraints can be incorporated in the control law. This means that the controlled system can be operated close to its physical limitations which often is economically beneficial [18].

However, the benefits come at the expense of solving an optimization problem at every sampling instant. This can be a drawback, because solving an optimization problem can be demanding if short sampling times are required and only embedded hardware is available. As a remedy, a significant research effort has been devoted into the development of tailormade algorithms to solve the underlying optimization problems [19].

Let us now provide a description of the main idea behind MPC. To that end, we consider a linear discrete time system,

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k), \tag{4.1}$$

where $\mathbf{x}(k) \in \mathbf{R}^{n_x}$ and $\mathbf{u}(k) \in \mathbf{R}^{n_u}$ represent the state and control variables at the time instant k. Moreover, we assume that the state vector is known at each sampling instant, i.e. $\mathbf{x}(0) = \mathbf{x}_0$. This can be achieved either by direct measurements or by state estimates.

Let us now note that after receiving $\mathbf{x}_0$, the state trajectory can be predicted over a horizon N as:

$$\mathbf{x}(1) = A\mathbf{x}_0 + B\mathbf{u}(0) \tag{4.2a}$$

$$\mathbf{x}(2) = A\mathbf{x}(1) + B\mathbf{u}(1) \tag{4.2b}$$

$$\vdots \tag{4.2c}$$

$$\mathbf{x}(N) = A\mathbf{x}(N-1) + B\mathbf{u}(N-1) \tag{4.2d}$$

for some sequence of control inputs $\{\mathbf{u}(0)\cdots\mathbf{u}(N-1)\}$.

Since the control inputs can be manipulated, it is natural to choose them in order to optimize the state trajectory. Here, we measure optimality by a quadratic performance criteria,

$$\begin{aligned} J(\mathbf{x},\mathbf{u}) = \sum_{i=1}^{N-1} \Big(\frac{1}{2}(\mathbf{x}(i)-\mathbf{x}_r(i))^T Q(\mathbf{x}(i)-\mathbf{x}_r(i)) + \frac{1}{2}(\mathbf{u}(i)-\mathbf{u}_r(i))^T R(\mathbf{u}(i)-\mathbf{u}_r(i)) \Big) \\ + \frac{1}{2}(\mathbf{x}(N)-\mathbf{x}_r(N))^T P(\mathbf{x}(N)-\mathbf{x}_r(N)) \end{aligned} \tag{4.3}$$

where $Q \in \mathbf{S}_{++}^{n_x}$, $P \in \mathbf{S}_{++}^{n_x}$ and $R \in \mathbf{S}_{++}^{n_u}$, here $\mathbf{S}_{++}$ betyder Symmetric and positiv definit. $\mathbf{x}_r$ and $\mathbf{u}_r$ denote the reference for the states and controls.

Now, by combining (4.2) and (4.3), with the physical limitations of the system, the following optimization problem can be formed:

$$\min_{\mathbf{x},\mathbf{u}} \quad J(\mathbf{x},\mathbf{u}) \tag{4.4a}$$

$$\text{s.t.} \quad \mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k),\ k = 1,\dots,N-1 \tag{4.4b}$$

$$\mathbf{x}(k) \in \mathcal{X}_k, \quad k = 1,\dots,N \tag{4.4c}$$

$$\mathbf{u}(k) \in \mathcal{U}_k, \quad k = 0,\dots,N-1 \tag{4.4d}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \tag{4.4e}$$

Note that for affine constraints (4.4c) and (4.4d), (4.4) is a highly structured Quadratic Program (QP). This structure is heavily exploited in solvers for MPC [20].

In MPC, the problem (4.4) is solved at every sampling instant. The first control input $\mathbf{u}(0)$ is then applied to the plant. A new problem is then solved at the succeeding sampling instant, and feedback is thus incorporated in the scheme. Stability and recursive feasibility of MPC controllers have been extensively studied in the literature, and can generally be guaranteed by enforcing some technical assumptions on the terminal cost P on the terminal set $\mathcal{X}_N$ [20].

4.2 MPC-based longitudinal trajectory planning

In this section, we provide a description of the MPC-based longitudinal trajectory planner for the lane change assist.

4.2.1 MPC implementation idea

As was described in Section 2.6.1, the lane change maneuver is divided into three phases. Although the MPC problem formulation in each phase is slightly different from the other phases, it takes the following form regardless of the phase:

$$\begin{aligned} \min_{u} \quad & \textit{Cost} && \text{(4.5a)} \\ \text{s.t.} \quad & \textit{Vehicle dynamics} && \text{(4.5b)} \\ & \textit{Physical and design constraints} && \text{(4.5c)} \\ & \textit{Collision avoidance constraints} && \text{(4.5d)} \end{aligned}$$

where the vehicle dynamics in (4.5b) is a forward simulation of the point mass model (2.2a), and the physical and design constraints (4.5c) ensure that the physical limitations and the traffic rules are respected. The difference between the MPC problems lies in the cost (4.5a) and in the collision avoidance constraint (4.5d).

The MPC problem formulations in the phases are detailed in Section 4.2.3, and the implementation details are summarized in Section 4.2.4.

4.2.2 Physical and design constraints

Regardless of phase, the MPC problem should respect the traffic rules and the physical limitations of the vehicle. In this subsection, we provide a description of the constraints that are enforcing this.

Due to power and brake limitations of the vehicle, the longitudinal acceleration is bounded. Since the longitudinal acceleration is a control variable in (4.5a), this is trivially modelled as:

$$\underline{a}_x \leq a_x(k) \leq \overline{a}_x, \ \forall k \tag{4.6}$$

for some lower bound $\underline{a}_x < 0$ and upper bound $\overline{a}_x > 0$. Moreover, to increase the comfort for the passengers in the vehicle, the longitudinal jerk is also restricted as:

$$\underline{j}_x \leq j_x(k) \leq \overline{j}_x, \ \forall k \tag{4.7}$$

for a lower bound $\underline{j}_x < 0$ and upper bound $\overline{j}_x > 0$. Note that (4.6) and (4.7) take the form of so-called box-constraints which can be handled very efficiently in many MPC solvers [21].

Compliance with the (longitudinal) traffic rules is enforced by limiting the longitudinal velocity,

$$\underline{v}_x \leq v_x(k) \leq \overline{v}_x, \ \forall k \tag{4.8}$$

for some lower bound $\underline{v}_x$ and upper bound $\overline{v}_x$.

Finally, to ensure feasibility, the so-called soft constraints were introduced to accelerations and jerks in all phases.

$$a_x(k) + s(k) \leq 0, \quad \forall k \tag{4.9}$$

$$j_x(k) + c(k) \leq 0, \quad \forall k \tag{4.10}$$

where the slack variables $s = [s(1) \dots s(N)]^T$ and $c = [c(1) \dots c(N)]^T$ are heavily penalized in the objective function.

The cost function is thus formulated as

$$\begin{aligned} J(\mathbf{x}, \mathbf{u}) = \sum_{i=1}^{N-1} \Big(&\frac{1}{2}(\mathbf{x}(i) - \mathbf{x}_r(i))^T Q(\mathbf{x}(i) - \mathbf{x}_r(i)) + \frac{1}{2}(\mathbf{u}(i) - \mathbf{u}_r(i))^T R(\mathbf{u}(i) - \mathbf{u}_r(i)) \\ &+\gamma(\|s(i)\|_2^2 + \|c(i)\|_2^2)\Big) + \frac{1}{2}(\mathbf{x}(N) - \mathbf{x}_r(N))^T P(\mathbf{x}(N) - \mathbf{x}_r(N)) \end{aligned} \tag{4.11}$$

where $Q \in \mathbf{S}_{++}^{n_x}$, $P \in \mathbf{S}_{++}^{n_x}$ and $R \in \mathbf{S}_{++}^{n_u}$, $\mathbf{x}_r$ and $\mathbf{u}_r$ denote the reference for the states and controls, and γ is the penalty for the slack variables s and c. The resulting optimization problem can thus be formulated as follows

$$\min_{u} \quad (4.11) \tag{4.12a}$$

$$\text{s.t.} \quad (4.4b), (4.4c), (4.4d), (4.4e), (4.6), (4.7), (4.8), (4.9), (4.10), (4.5d) \tag{4.12b}$$

4.2.3 Cost function and collision avoidance constraints

In this subsection, we provide a description of the cost function and the collision avoidance constraints in the different phases.

4.2.3.1 Phase 1

In this phase, the speed and position of the vehicle are adjusted to track the empty slot in the adjacent lane. Collision avoidance constraints are incorporated to avoid collisions with vehicles in the host lane.

The cost function is selected as (4.11). The reference states $\mathbf{x}_r$ is calculated as follows and the reference control $\mathbf{u}_r$ is 0.

$$\mathbf{x}_r(i) = x_{s_3}(0) + \max(misalignment, \rho) + v_{s_4} t(i) \tag{4.13}$$

Misalignment is a fixed safety design distance that the host vehicle should keep. ρ is the relative distance between the host vehicle and vehicle s_4.

$$\rho = \tau(v_x - v_{s_4}) \tag{4.14}$$

τ is the time to collision(TTC). The final cost reference states $x_r(N)$ is the same as $\mathbf{x}_r(i)$.

The collision avoidance constraints are selected as,

$$x(k) \leq \bar{p}_{s_2}(k), \quad \forall k \tag{4.15}$$

where the upper bound is given by:

$$\bar{p}_{s_2}(k) = x_{s_2}(k) - l_r(s_2) \tag{4.16}$$

where $l_r(s_2)$ is the safety distance behind the vehicle s_2, calculated according to equation (2.13).

A designed transition from Phase 1 to Phase 2 is performed when the host vehicle is placed in between s_4 and s_3, i.e.,

$$x_{s_3} + l_f(s_3) \leq x(k) \leq x_{s_4} - l_r(s_4) \tag{4.17}$$

and the following conditions are fulfilled, that is when the host vehicle has passed 5 meters of s_3 and velocity differences between the host vehicle and s_4 within 40 km/h, these are also design parameters.

$$x(k) - x_{s_3} - l_f(s_3) \geq 5 \tag{4.18}$$

$$v(k) - v_{s_4} \leq 40/3.6 \tag{4.19}$$

where $l_f(s_3)$ is the safety distance distance in front of the vehicle s_3, i.e. when the reference is tracked close enough and the safety constraints are fulfilled.

4.2.3.2 Phase 2

This is the actual lane change phase, that is lateral motion is introduced for the lane change purpose. But since we are designing a longitudinal path planning algorithm, lateral control is not relevant here. The algorithm in this phase has the same cost function as the previous phase, with the same reference states and controls, but in this phase only the final speed states matter, so the position reference state is 0.

To avoid collisions with vehicles in the adjacent lane, the collision avoidance constraints are extended as:

$$x(k) \leq \bar{p}_{s_4}(k), \quad \forall k \tag{4.20a}$$

$$x(k) \leq \bar{p}_{s_2}(k), \quad \forall k \tag{4.20b}$$

where

$$\bar{p}_{s_4}(k) = x_{s_4}(k) - l_r(s_4) \tag{4.21a}$$

$$\bar{p}_{s_2}(k) = x_{s_2}(k) - l_r(s_2) \tag{4.21b}$$

A transition from Phase 2 to Phase 3 is performed when a successful lane change has been performed. This means that the host vehicle is placed laterally in the new lane and between the surrounding vehicles s_4 and s_3, i.e.,

$$x_{s_3} + l_f(s_3) \leq x(k) \leq x_{s_4} - l_r(s_4) \tag{4.22}$$

4.2.3.3 Phase 3

At this point, the lane change maneuver has been performed, the purpose here is to adjust the final speed to the actual speed and keep a safe distance with surrounding vehicles in this lane within a series of constraints, i.e position, velocity, acceleration and jerk and the same cost function with the same reference states, controls and final states as the previous phase.

In this phase, the collision avoidance that we need to consider is the front vehicle in this new lane, that is:

$$x(k) \leq \bar{p}_{s_4}(k), \quad \forall k, \tag{4.23}$$

where

$$\bar{p}_{s_4}(k) = x_{s_4}(k) - l_r(s_4) \tag{4.24}$$

4.2.4 MPC implementation details

The sets $\mathcal{X}_k$ and $\mathcal{U}_k$ in equations (4.4c) and (4.4d) are convex, therefor the MPC problem can be equivalently rewritten as a standard QP problem

$$\begin{aligned} \min_{z} \quad & \frac{1}{2} z^T H_M z + f^T z && (4.25\text{a}) \\ \text{s.t.} \quad & A_{eq} z = b_{eq} && (4.25\text{b}) \\ & A_{in} z \leq b_{in} && (4.25\text{c}) \end{aligned}$$

With states $X = \begin{bmatrix} x & v_x \end{bmatrix}^T$, $u = a_x$ and $z = \begin{bmatrix} X^T(1) & \cdots & X^T(N) & u(0) & \cdots & u(N-1) \end{bmatrix}^T$. The matrix H_M is symmetric and positive semi-definite and the QP problem (4.25a) is convex.

The control horizon and the prediction horizon are $N = 80$ and the final cost penalty is $P_f = Q$. The hessian matrix and equality constraints for the optimization problem are:

$$H_M = 2 \underbrace{\begin{bmatrix} Q & 0 & & \cdots & & & 0 \\ 0 & \ddots & & & & & \\ & & Q & \ddots & & & \vdots \\ \vdots & & & P_f & & & \\ & & & & R & & \\ & & & & & \ddots & 0 \\ 0 & & & \cdots & & 0 & R \end{bmatrix}}_{\text{Dimension of matrice } (3 \cdot N \times 3 \cdot N)}, \qquad f = \underbrace{\begin{bmatrix} -2X_r(1) \\ \vdots \\ -2X_r(N) \\ -2u_r(0) \\ \vdots \\ -2u_r(N-1) \end{bmatrix}}_{\text{Dimension of matrice } (3 \cdot N \times 1)} \tag{4.26a}$$

4.2.4.1 Constraints

Rewrite the equations (4.4b), (4.4c), (4.4d), (4.4e) to the form as equation (4.25b) gives the following equality constraints:

$$\underbrace{\begin{bmatrix} I & & & & -B & & \\ -A & \ddots & & & & \ddots & \\ & \ddots & \ddots & & & & \ddots \\ & & -A & I & & & & -B \end{bmatrix}}_{A_{eq} \text{ Dimension of matrice } (2 \cdot N \times 3 \cdot N)} \underbrace{\begin{bmatrix} X(1) \\ \vdots \\ X(N) \\ u(0) \\ \vdots \\ u(N-1) \end{bmatrix}}_{z \ (3 \cdot N \times 1)} = \underbrace{\begin{bmatrix} AX(0) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{b_{eq} \ (2 \cdot N \times 1)} \tag{4.27}$$

I is a 2×2 identity matrix, if there are two states in the dynamic.

Inequality constraints from equations (4.6), (4.7), (4.8), (4.9), (4.10) and collision avoidance constraints in each phase can be formulated in form as (4.25c), that is soft constraints on acceleration and jerk, longitudinal speed constraints and safety critical zone constraints. Safety critical zone constraints are different in different phases, in Phase 1, the host vehicle needs to take the leading vehicle s_2 into consideration, in phase two with respect to both s_2 and s_4 and the last phase with respect to s_4. Since we have assumed the host vehicle is not fully automated and the vehicles behind will never get too close to the host vehicle. The detail matrices are formulated in the same way as (4.27) and implemented in the algorithm, here we will not illustrate it.

5

Results

This chapter presents the results of the two trajectory generation methods. In the first section below the evaluation method is described as well as how the performance of each method is measured and compared. In Section 5.2 the results for each method are presented.

5.1 Evaluation method

As mentioned in Section 2.6, the algorithms will be evaluated in 5 different but similar scenarios. The difference between them will be the initial velocity of the host vehicle and the positions and velocities of the vehicles in the host lane. For the vehicles in the adjacent lane the initial position and velocities will be the same for all 5 scenarios. In the host lane, the vehicles are assumed to be travelling at the same speed as the host vehicle with a time gap of 3 seconds both to the car in front, s_1, and to the car behind the host, s_2. The two vehicles in the adjacent lane, s_3 and s_4 are travelling at 80 km/h with a distance of 70 meters between them, this corresponds to a time gap of 3.15 seconds at their velocity. The initial velocity of the host vehicle and the vehicles in the host lane will range from $90 - 130$ km/h in increments of 10 km/h. In Figure 2.3, an overview of the scenario is shown and full details of the initial positions and velocities are shown in Appendix B.

The algorithms are designed to work for all kinds of variations on the general scenario as described in Section 2.6 and are not tuned specifically for the test cases below. This means that for example acceleration and jerk limits will be the same for all scenarios, just like the individual tuning parameters associated to the two methods. All parameters that are constant and not method specific for all scenarios are presented in Table 5.1.

The 5 different scenarios will all be run in a simulation where the host driver starts the lateral lane change when he has passed the safety distance in front of vehicle s_3 with 5 meters. This is assuming that the lane change is still feasible at that point. This gives a trajectory for a successful lane change where acceleration and jerk levels can be analyzed. To measure how long each method deems a lane change feasible without the lateral motion being started, the same scenarios are run but without driver interaction.

Parameter	Phase 1	Phase 2	Phase 3	Explanation
$\bar{v}$ (km/h)	v_0	v_0	v_0	Maximum velocity
$\underline{v}$ (km/h)	72	72	72	Minimum velocity
$\bar{a}$ (m/s^2)	0.1	0.1	0.1	Maximum acceleration
$\underline{a}$ (m/s^2)	-0.3	-2	-2	Minimum acceleration
$\bar{j}$ (m/s^3)	2	2	2	Maximum jerk
$\underline{j}$ (m/s^3)	-2	-2	-2	Minimum jerk
		Phase 1-3		
t_{sample} (s)		0.2		Sampling time
$t_{simulation}$ (s)		60		Simulation time

Table 5.1: Physical constraints in the different phases.

5.1.1 Performance indicators

From the simulations described above, a number of performance indicators are extracted. Since the LCA system is a comfort feature, good performance is very objective, but there are measurements associated to comfort, such as jerk and acceleration that can be measured. Comfort features are also required not to interfere with traffic safety or rules, this will be measured as the interference with the traffic in the original lane, particularly the vehicle behind the host. Although computation time is an important factor for algorithms which will run in real time on embedded hardware, it would be irrelevant to compare the two method's run-time before any run-time optimization has been carried out. The following performance indicators will be considered in the result:

- j_{RMS} - root mean square (RMS) value of the jerk
- a_{RMS} - RMS value of the acceleration
- $t_{interrupt}$ - time gap to the following vehicle at the point where a lane change becomes infeasible
- $t_{feasible}$ - the time window in a simulation from a lane change becomes feasible to it being infeasible when there is no lateral input from the driver

The two RMS values are direct measurements of how comfortably the lane change can be realized, the higher the jerk, the more uncomfortable, the same goes for acceleration. The lowest value the RMS can take will be the mean acceleration or jerk necessary to reach the desired velocity.

As the host vehicle slows down to prepare for a lane change, the following vehicle will get closer. The distance between the two vehicles at the point where a lane change becomes infeasible will act as a measurement on how much the LCA system impacts the following traffic. In the real world, the driver behind would most likely slow down if necessary, but $t_{interrupt}$ is the time gap to the car behind if it keeps a constant velocity. At the start of the simulation, the vehicle is 3 seconds behind, it is desired to keep $t_{interrupt}$ as high as possible (but not above 3 seconds).

An important aspect of the LCA system is to let the driver know when it's no longer feasible to do a lane change. It is desired to keep the lane change feasible as long as possible, but this will on the other hand interfere more with the following vehicles. $t_{feasible}$ can't be evaluated without also looking at $t_{interrupt}$, but together they will give a comprehensive look on how successful the algorithm is for the specific scenario.

With these performance indicators, it will be possible to compare the results of the two methods and determine which is the better one. Note that the two RMS values are from a simulation when the driver completes the lane change while $t_{feasible}$ and $t_{interrupt}$ are measurements from a simulation without any lane change.

5.2 Performance evaluation

The performance and simulation results will be presented separately for the two methods. For each method two of the scenarios will be presented in more detail, while the performance indicators of the remaining three scenarios will be presented in tables.

The two chosen scenarios are Scenario 2 and 3. Scenario 2 i a normal scenario where the host vehicle travel at a typical highway speed, 110 km/h, see Table B.3 for further details. Scenario 3 is a more extreme scenario where the host vehicle travels at 130 km/h (Appendix B Table B.5) and higher jerk and acceleration are required to find a feasible solution.

5.2.1 Performance of B-spline interpolation

When using smoothing splines as in an algorithm for the LCA system, the tuning is mainly done using the weighting between acceleration and tracking performance in the optimization problem. But the number of control points, the length of the planning horizon and the sample time will also affect the behaviour. Specific parameters for the B-spline trajectory algorithm are presented in Table 5.2, they are the same during the whole simulation.

For all scenarios except Scenario 5, a lane change is feasible. For the two lowest initial velocities, in Scenario 1 and 2, the difference in velocity compared to the desired velocity is very low. This allows the vehicle to slow down to the desired velocity even without entering Phase 2 and releasing the acceleration constraints. This is portrayed in Table 5.3 as $t_{feasible} = \infty$, since the vehicle will place itself next to the gap and stay there. For the higher initial velocities, the vehicle will eventually pass the gap, and a lane change will become infeasible.

Parameter	Value	Explanation
k	3	B-spline degree
m	150	Number of control points
λ	500	Penalty on acceleration compared to tracking performance
$t_{horizon}$ (s)	30	Planning horizon

Table 5.2: Parameters for simulation with B-spline method

Scenario	1	2	3	4	5
j_{RMS}	0.0210	0.0717	0.1770	0.2519	–
a_{RMS}	0.0980	0.1783	0.3040	0.4678	–
$t_{interrupt}$	0	0	1.8776	2.5755	2.8088
$t_{feasible}$	∞	∞	8.1900	2.6100	0

Table 5.3: Performance measures for the 5 scenarios with B-spline interpolation

5.2.1.1 Scenario 3

With an initial velocity of 110 km/h, the algorithm adjusts the velocity to let the driver complete a safe lane change. As can be seen in Figure 5.1, the position of the host is well within the safety distances once the lane change is initiated. The velocity drops to 0.57 m/s below the desired velocity for a while to position the vehicle at the desired position and then slowly accelerates. Initially the host vehicle travels at the same velocity for 5 seconds, after which it decelerates at the maximum allowed deceleration until the lateral motion is initiated. At $t = 13.02$ seconds, the vehicle is travelling at 28.23 m/s and the the deceleration increase with the maximum amount of jerk to a deceleration peak at -1.15 m/s^2.

The simulation result for Scenario 3 without the driver completing the lane change is shown in Figure 5.2. In this simulation, the first 13 seconds are identical to the one described above, the difference comes after the lane change in the previous simulation is initiated. In Figure 5.2 it can be seen that the vehicle keeps decelerating as much as allowed until $t = 20.40$ seconds when the large speed difference and the proximity to the safety distance make a lane change infeasible. At this moment, the driver is notified that a lane change is no longer possible and an escape trajectory is generated. The escape trajectory is not shown in the figure, but it is designed to let the vehicle accelerate up to its initial velocity.

5.2.1.2 Scenario 2

If a lane change is completed at a lower initial velocity, such as the 100 km/h in Scenario 2, the result is still successful as can be seen in Figure 5.3. Compared to Scenario 3, the initial deceleration starts much later, both in regard of time and distance. At 100 km/h, the algorithm decides it is sufficient to start braking after 12.4 seconds and 69 meters, 7.4 seconds and 27 meters later than at 110 km/h. The

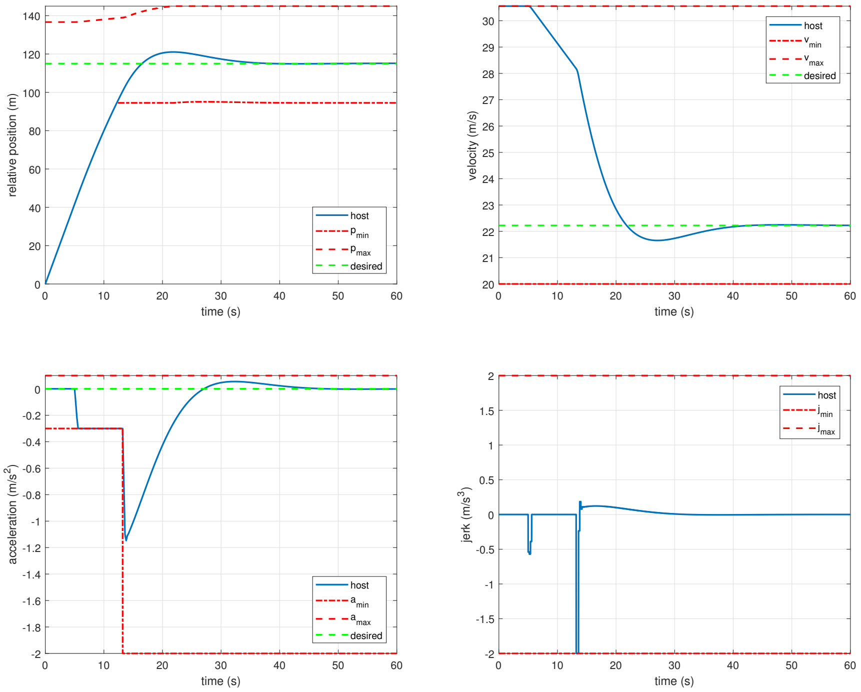


Figure 5.1: Plots Scenario 3 with lane change

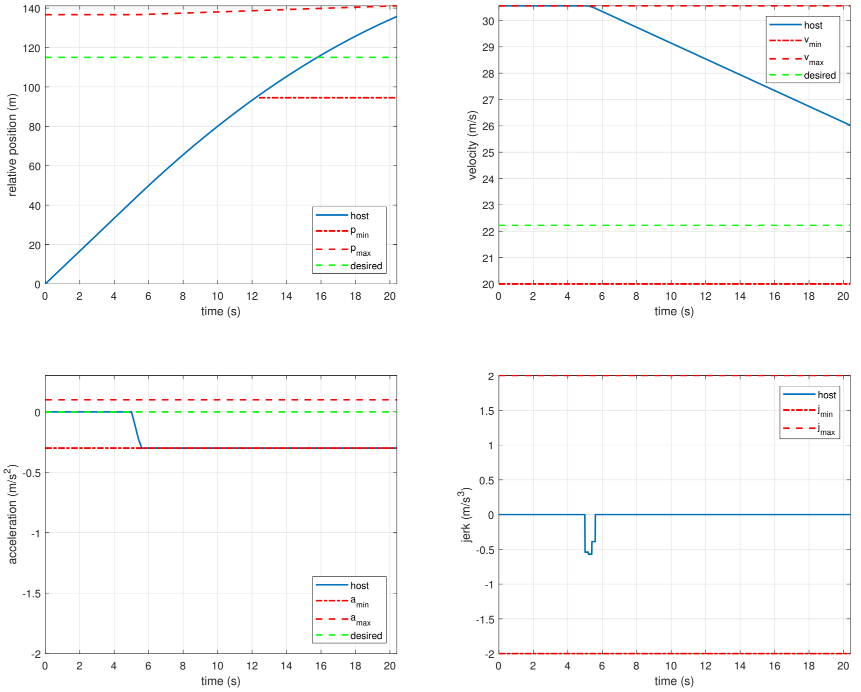


Figure 5.2: Plots Scenario 3 when no lane change occurs

lower difference in velocity between the host and the desired velocity also leads to a much lower acceleration and jerk. The acceleration peaks at $-0.49\ m/s^2$ and the jerk at $-0.95\ m/s^3$. Still a few seconds of positive acceleration is needed to reach the desired position and velocity, but the acceleration is so small that it should barely be noticeable.

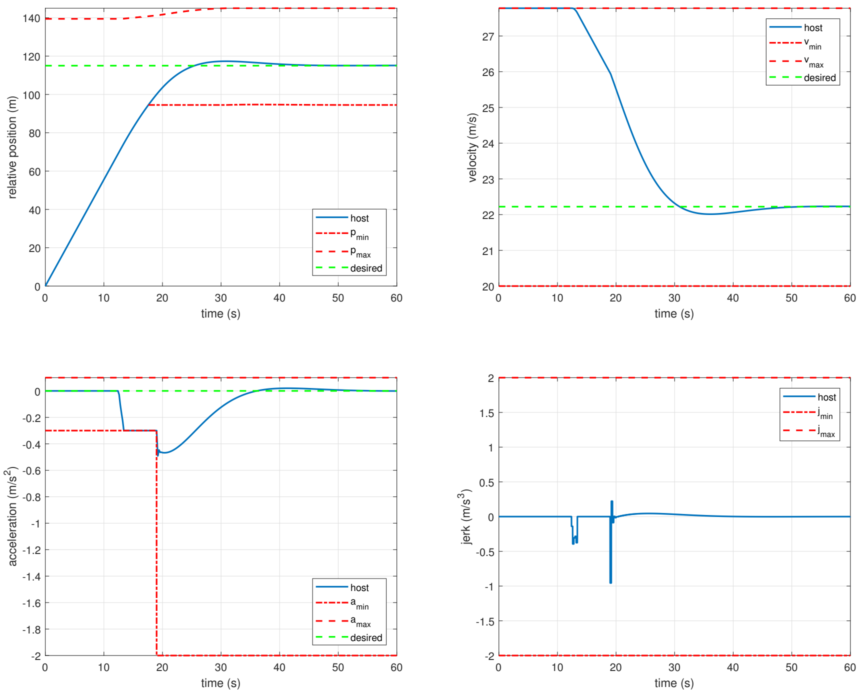


Figure 5.3: Plots Scenario 2 with lane change

The main difference at lower initial velocity is when no lane change occurs. As explained earlier, the low velocity difference leads to the host vehicle reaching the desired velocity even without releasing the acceleration constraints.

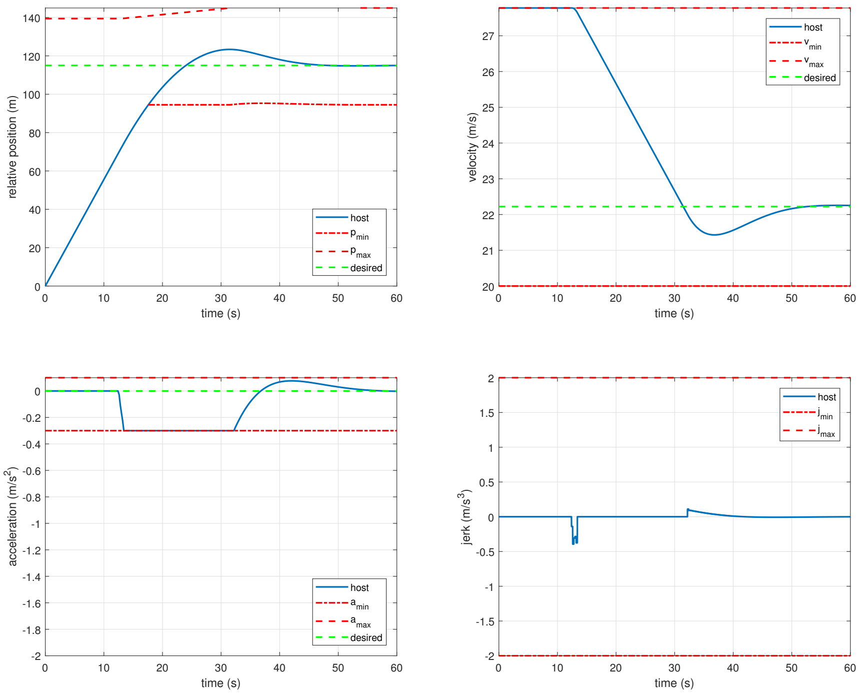


Figure 5.4: Plots Scenario 2 when no lane change occurs

5.2.2 Performance of MPC with hard constraints

The tuning parameters in the MPC algorithm are mainly the weighting matrices Q and R in the three scenarios, the horizon N and the transition condition between the scenarios. The tuning of the weighting matrices is summarized in Table 5.4.

	Phase 1	Phase 2	Phase 3
States weights (x,v,a) Q	diag(100,1000,10)	diag(0,1,1)	diag(0,100,1)
Control input weights R	10000	1	100000
Final cost weights P	diag(100,1000,10)	diag(0,1,1)	diag(0,100,1)
Acceleration slack penalty r	1e9	1e9	1e9
Jerk slack penalty q	1e9	1e9	1e9
Prediction horizon(samples) N	80	80	80

Table 5.4: Tuning of MPC controller with hard constraints.

For all scenarios except Scenario 5, the MPC algorithm is able to perform a successful lane change. The performance of the algorithm with respect to the performance indicators is summarized in Table 5.5. However, it should be noted that the algorithm has been tuned for an overall (subjectively) good performance and not for performing well with respect to the indicators.

Scenario	1	2	3	4	5
j_{RMS}	0.4596	0.5320	0.6481	0.7302	–
a_{RMS}	0.1596	0.2246	0.3384	0.4533	–
$t_{interrupt}$	∞	∞	1.607	1.150	–
$t_{feasible}$	∞	∞	14.8000	6.8000	–

Table 5.5: Performance measures for the 5 scenarios with MPC

5.2.2.1 Scenario 2 and 3

In Figure 5.5 and 5.6, successful lane change manuevers are shown for Scenario 2 and Scenario 3 respectively. Observe that the behavior is very similar for the two scenarios and that the velocity is monotonously decreasing during the lane changes.

We also note that most of the constraints are inactive during the lane changes, except of the jerk limits during short time periods and the lower acceleration limit in Phase 1. The constraints are active during slightly longer time periods in Scenario 3. This is aligned with our intuition, since the velocity difference between the lanes is larger, and therefore also the required acceleration.

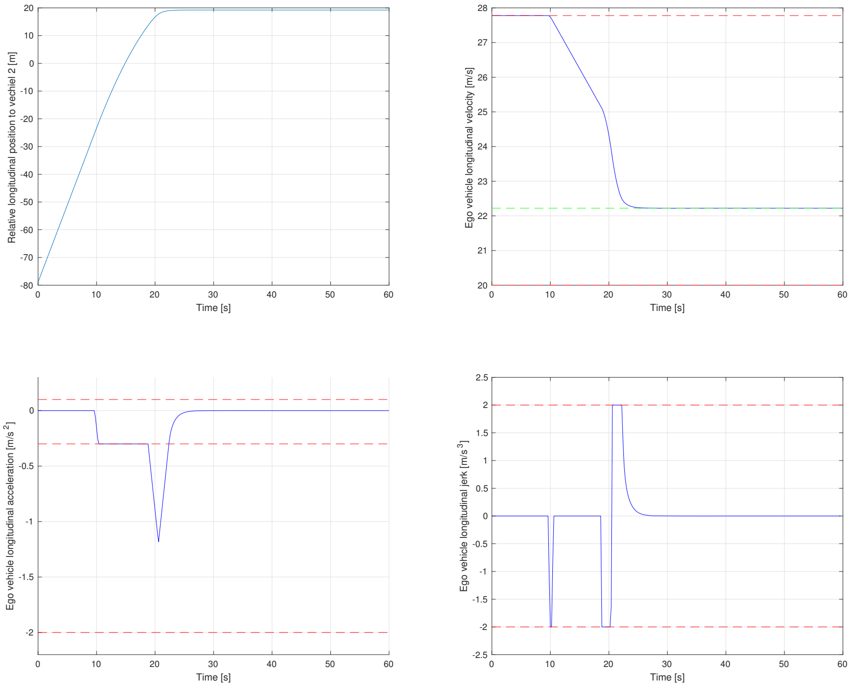


Figure 5.5: Overview of Scenario 2. Phase 2 and Phase 3 are entered at 19.4 s and 28.0 s respectively.

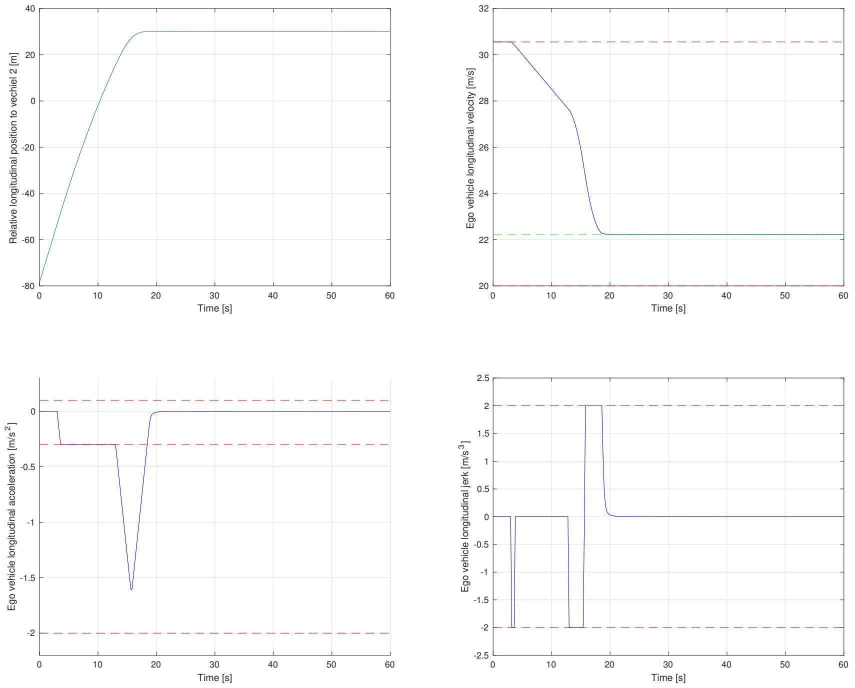


Figure 5.6: Overview of Scenario 3. Phase 2 and Phase 3 are entered at 14.2 s and 22.6 s respectively.

5.2.3 Comparison of differences between soft and hard constraints in the MPC algorithm

As mentioned above, the MPC algorithm is able to perform a successful lane change in all scenarios except Scenario 5 under the hard constrained conditions. In Scenario 5, the host vehicle travels with a initial speed at 130 km/h, the algorithm can not successfully find a trajectory for lane change that is infeasible at this speed. But with a relatively soft penalty for acceleration and jerk, the simulation can actually go through to avoid infeasibility for this application. The result is shown in Figure 5.7 with acceleration slack penalty $r = 10^6$ and jerk slack penalty $q = 10^6$ in Phase 1.

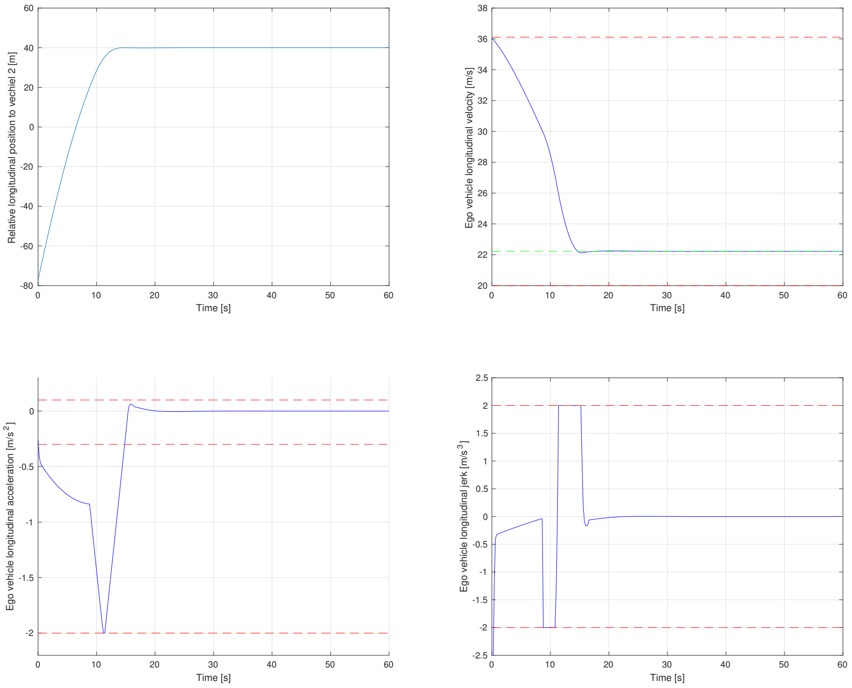


Figure 5.7: Plots Scenario 5 with soft constraints. Phase 2 and Phase 3 are entered at 9.2 s and 17.2 s respectively.

6

Discussion

This chapter discusses the two developed trajectory generating algorithms, as well as their resulting trajectories. It also elaborates on the disadvantages and advantages found for each method during the development and evaluation. Suggestions to improvements and future work will be stated as well.

6.1 Similarities and differences

It has been shown that for the given scenarios, both the MPC and the B-spline method succeed in generating longitudinal trajectories suitable for a lane change. In theory it will be possible to implement any of the two algorithms in a vehicle after some more work. Functions such as target selection and the HMI interface needs to be sorted out first. Since both methods depend heavily on the QP-solver used, the choice of solver will have a major impact on the computation times. The solver used in this thesis is not suitable for a car implementation since it is only available in Matlab.

The resulting trajectories from test simulations are similar but not identical. Part of the differences depends on the tuning for each individual method. But more interesting are the fundamental differences between the two methods.

One fundamental difference is in the way the two methods divides the scenario into different phases. Even though they go through the same phase, the MPC-method handles one phase at a time, while the B-spline method considers the full scenario in each time instant. The benefit of optimizing the trajectory for one phase at a time is the ability to use shorter planning horizons without increased risk of infeasibility. A shorter planning horizon with the same sampling time leads to fewer decision variables and thus a shorter computation time. The downside is only knowing if the current phase is feasible, not the following. The B-spline method optimizes for a trajectory for the full scenario, as long as it is feasible within its longer planning horizon. Thus it won't start to slow down for a lane change that was never feasible from the start. This is a major benefit and a necessity for the driver to trust the system.

Comparing the performance indicators, it is clear that the B-spline method performs better in all scenarios in regard of comfort. The RMS values for both jerk and acceleration are significantly lower than for the MPC method. Looking at the indicator of how much the vehicle behind the host vehicle is interrupted, $t_{interrupt}$, the B-spline method performs slightly better, while the MPC method performs much better in regard of for how long the lane change is feasible, according to $t_{feasible}$.

6.2 Regarding the B-spline method

The B-spline method is loosely based on earlier proven work for path planning [5] and has been reworked to be suitable for local longitudinal trajectory planning. Some of the key changes made are:

- **Trajectory in one dimension**
  Whereas [5] used smoothing splines to plan a 2 dimensional path from start to finish, this thesis uses it to produce a one dimensional trajectory. This means the constraints can be rewritten in a way that allows them to be changed between different time intervals in the planning horizon.
- **Receding horizon possibility**
  Since the problem stated in this thesis is regarding trajectory planning, it was necessary to be able to repeat the optimization of the trajectory at fixed time intervals as explained in 2.8. With the help of equality constraints, the initial conditions could be forced to match the current state and with the sampling time set to a multiple of the knot interval, optimal tracking was possible.
- **Variable constraints**
  With acceleration constraints that are not constant for the full planning horizon, the point where the constraints need to be lifted for a feasible lane change can be calculated. The B-spline method optimizes the trajectory for the full scenario in each iteration instead of having one optimization problem for each phase as in the MPC-based method. This has the benefit of already from the first iteration knowing if a lane change will be feasible or not, and the risk of the driver starting to change lanes when it's infeasible is reduced.

6.2.1 Improved stability with soft constraints

Using the B-spline method to generate a trajectory proved to work well for most of the given lane change scenarios with the exception being Scenario 5 where the difference in velocity between the host vehicle and the desired final velocity was too large. It is natural that a lane change will become infeasible once the initial velocity becomes too high. A car can typically brake at accelerations as high as 10 m/s^2, but as a feature that is designed to be comfortable and safe, the acceleration constraints were set much lower. Just as shown on the MPC method in 5.2.3, soft constraints can also be used on the B-spline method. Since the physical limitations of a vehicle's braking force is much higher than the constraint at $-2\ m/s^2$, there is room for for soft constraints. Soft constraints should be used to make the method more robust against measurement noise, not to force the QP-solver to find a feasible trajectory when the velocity difference is unreasonably high.

6.2.2 Control points, planning horizon and computation time

One drawback of optimizing the trajectory for the full lane change in every iteration is the long planning horizon required. For optimal tracking the sampling time needs to be a multiple of the knot interval as stated in 3.2.5. The knot interval depends on the number of control knots and the length of the planning horizon. If a high sampling frequency is desired, it requires a lot of control points which leads to a larger QP-problem and longer computation times. For the simulations in this thesis the sampling time was chosen to get a reasonable ratio between sampling and computation time.

6.2.3 Is a reference position necessary?

The cost function in the B-spline method penalizes deviation from the desired position, which is centered between the two vehicles in the target lane. It can be argued that it is unnecessary to have a desired position as long as the position constraints are fulfilled. Since the position constraints takes the safety distance to each car into account, every position that is within these constraints should be an acceptable position. Removing the reference position from the cost function will give an end position that is always as close to the preceding car as is allowed to minimize the acceleration. Removing the reference position will eliminate the overshoot seen in 5.2.1, where the host vehicle slows down more than desired to reach the reference position, and then accelerates again to reach the desired speed.

6.2.4 Improve performance at low Δv

In Scenario 1 and 2, where the difference between the initial velocity and the desired velocity is low, we get a somewhat unwanted behavior. As shown in Figure 5.4, the host vehicle reaches the desired velocity without entering Phase 2. This forces the vehicle behind the host to also slow down to the same velocity and disturbing the traffic more than desired. Given the size of the gap in the target lane and the low velocity, it is possible to complete the lane change without any braking in Phase 1 and thus without interfering with the traffic in the original lane. Trying to solve this by tuning will worsen the performance in the higher velocity scenarios. There is a trade off between braking early and late, where braking early improves the performance in high velocities but worsen the performance in low velocities, braking late have the opposite effect. This could possibly be solved by an adaptive tuning which changes the tuning as the speed decreases. Another alternative is to set a lower limit on the velocity in Phase 1, which takes the length of the gap and the desired velocity into account. This limit could be designed so that the host vehicle never reaches the desired velocity in Phase 1 but still starts to decelerate early when the initial velocity is high.

6.3 Regarding the MPC method

6.3.1 About the motion model

As mentioned before, different motion models give different simulation accuracy and simplicity. In the MPC algorithm, a point mass model was used in longitudinal estimation to simplify the mathematical formulation but with the drawback of less accuracy as a motion model for a vehicle. Thus good future work can be, for example, to try to implement the algorithm with a single-tracke model or a double track model.

6.3.2 About the tuning parameters

The robustness of the algorithm depends highly on the tuning. To achieve a good performance, the MPC algorithm needs a velocity depended tuning, but Aptiv preferred a velocity independed tuning in this work, thus the performance is restricted with this hard constraint. So a direction of future work can be to investigate a velocity depended tuning method such as gain scheduling.

6.3.3 About augmented system

During the algorithm implementation, it was noticed that the algorithm between phases has no memory, i.e when it comes into Phase 2 and Phase 3, there will be a sudden jump of the acceleration in the beginning of these phases, so the constraints are only active within phases but not in between them. To solve this problem, an augmented system was formulated instead, the results were shown in the previous section.

6.3.4 About the feasibility

One of the most fundamental problems in MPC is the lack of guaranteed feasibility. Thus to be able to step up this driver assist system to a fully autonomous vehicle, one future work that would be interesting is to explore a stability method that guarantees the safety on the road. Recursive feasibility can be interesting to investigate in this case, assume $x \in \chi_N$ is a feasible state, it means that there is a solution to the optimization problem for that x. The question is, after applied the computed control action, will the next state x^+ also belong to the feasible set χ_N? If it is so, the control action would be defined also at the next sampling instant. This is known as recursive feasibility. And another alternative is to calculate invariant set to solve this problem.

Back to this thesis, to divide the whole optimization problem into three phases can easily solve the non-convexity problem and also significantly reduce the simulation time since it does not need to predict the whole optimization problem but only within phases. But for some initial states, it maybe impossible to solve the MPC problem while respecting constraints on states and/or outputs. The problem can thus be infeasible. In practice, infeasibility may occur by too strict performance requirements, a large disturbance, model uncertainty or by an unstable system. It is always necessary to have a backup controller in this case. The MPC controller appears that can not guarantee feasibility for Phase 2 while in Phase 1 with hard constraints for acceleration and jerk in this algorithm. To ensure feasibility in Phase 2, we can either calculate the time invariant set, that is for some initial state that belongs to the set, it will be guaranteed to stay in the set for ever, or we need so-called "soft constraints". Soft constraints can be implemented in several ways, the easiest and most commonly used is to introduce slack variables that are severely punished in the cost function. This method has been successfully implemented for the MPC algorithm, as seen in Section 5.2.3.

6.3.5 About the computational efficiency

Many factors may affect the algorithm computational efficiency, such as the sampling time, the size of the control and prediction horizon, the representations and ordering of the variables, a different optimization solver or even a different computer. We have not put much efforts on this part and we used Matlab quadprog as our simulation solver. To realize this algorithm, a better solver is needed.

7
Conclusion

This project concludes that a longitudinal trajectory suitable for a lane change maneuver can be generated successfully using either B-spline interpolation or the Model predictive control framework. With continuous work, both of the presented methods are possible to implement in a vehicle for more evaluation and tuning.

The requirements for the function to be safe and follow current traffic regulations are fulfilled by using constraints on position and velocity. The more objective requirement of comfort and usability are quantified by a number of performance indicators taking the jerk, acceleration and time gap to other road users into account. According to these indicators, the B-spline method performed best in regard of comfort while the Model predictive control method performed best regard of usability.

Bibliography

[1] Planing, P., (2013), *Innovation Acceptance - The Case of Advanced Driver-Assistance Systems*, Springer Fachmedien Wiesbaden, Wiesbaden

[2] Jurgen, R., (2013), *Autonomous Vehicles for Safer Driving*, SAE International

[3] Cartwright, S., Cooper, C. & Barron, A., (1996) *The Company Car Driver, Occupational Stress as a Predictor of Motor Vehicle Accident Involvement*, Human Relations Vol 49, Issue 2, pp. 195 - 208

[4] Nilsson, J., (2016), *Automated Driving Maneuvers- Trajectory via Convex Optimization in the Model Predictive Control Framework*, Institution for signals and system, Mechatronics, Chalmers University of Technology

[5] Kano, H. & Fujioka, H., (2017), *Velocity and acceleration constrained trajectory planning by smoothing splines*, IEEE, pp. 1167

[6] Sciciliano, B., et al., *Robotics – Modelling, Planning and Control*, (2010), Springer-Verlag, London

[7] Schubert R, Richter E, Wanielik G, (2008), *Comparison and Evaluation of Advanced Motion Models for Vehicle Tracking*, Proc.international conference on information fusion, pp. 1-6

[8] Rajamani, R., (2006), *Vehicle Dynamics and Control*, Springer Science+Business Media, Inc., 233 Spring Street, New York, NY 10013, USA

[9] IBM ILOG, (2010), *V12.2: User's Manual for CPLEX*,

[10] Mathworks, (2017), *Optimization Toolbox User's Guide*, The MathWorks, Inc.

[11] S.Boyd, L.Vandenberghe, (2004), *Convex Optimization*, Cambridge: University Press.

[12] Egardt, B., (2017), *Model Predictive Control Lecture Notes*, Chalmers University of Technology

[13] F.Borrelli, A.Bemporad, M.Morari, (July 2017), *Predictive Control for linear and hybrid systems*, Cambridge University Press, 2017

[14] Prautzsch, H., Boehm, W. & Paluszny, M., (2002), *Bézier and B-spline techniques*, Springer, Berlin

[15] Fujioka, H., Kano, H., Egerstedt, M. & Martin, C. F., (2005), *Smoothing spline curves and surfaces for sampled data*, Int. J. of Innovative Computing, Information and Control, 1(3)

[16] Katrakazas, C., Quddus, M., Chen, W. H., & Deka, L., (2015), *Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions*, Transportation Research Part C: Emerging Technologies, 60, 416-442.

[17] Nocedal, J., Wright, S., (2006), *Numerical Optimization*, Springer, New York

[18] J.B.Rawlings, D.Q. Mayne, (2009), *Model Predictive Control: Theory and Design*, Madison(WI): Nob Hill Publishing.

[19] J.M. Maciejowski, (2009), *Predictive Control with Constraints*, Harlow: Pearson Education.

[20] B.Kouvaritakis, M. Cannon (2016), *Model Predictive Control: Classical,Robust and Stochastic*, Springer International Publishing AG Switzerland

[21] P. De Angelis, P. Pardalos, G. Toraldo, (1997), *Quadratic Programming with Box Constraints*, Publisher: Springer US

A

Q_0-matrix

$$Q_0 = \frac{1}{\alpha} \begin{pmatrix} \frac{1}{250} & \frac{16}{625} & \frac{119}{10000} & \frac{1}{5000} & & & & & \\ \frac{16}{625} & \frac{2397}{10000} & \frac{2107}{10000} & \frac{119}{5000} & \frac{1}{5000} & & & & \\ \frac{119}{10000} & \frac{2107}{10000} & \frac{2377}{5000} & \frac{2363}{10000} & \frac{119}{5000} & \frac{1}{5000} & & & \\ \frac{1}{5000} & \frac{119}{5000} & \frac{2363}{10000} & \frac{2397}{5000} & \frac{2363}{10000} & \frac{119}{5000} & \frac{1}{5000} & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & \frac{1}{5000} & \frac{119}{5000} & \frac{2363}{10000} & \frac{2397}{5000} & \frac{2363}{10000} & \frac{119}{5000} & \frac{1}{5000} \\ & & & \frac{1}{5000} & \frac{119}{5000} & \frac{2363}{10000} & \frac{2377}{5000} & \frac{2107}{10000} & \frac{119}{10000} \\ & & & & \frac{1}{5000} & \frac{119}{5000} & \frac{2107}{10000} & \frac{2397}{10000} & \frac{16}{625} \\ & & & & & \frac{1}{5000} & \frac{119}{10000} & \frac{16}{625} & \frac{1}{250} \end{pmatrix} \tag{A.1}$$

B

Data for simulations

Vehicle	x_0 (m)	y_0 (m)	v_0 (km/h)	a_0 (m/s^2)
Host	0	0	90	0
Vehicle 1	-75	0	90	0
Vehicle 2	75	0	90	0
Vehicle 3	80	3.75	80	0
Vehicle 4	150	3.75	80	0

Table B.1: Initial position and velocities of the vehicle in Scenario 1

Vehicle	x_0 (m)	y_0 (m)	v_0 (km/h)	a_0 (m/s^2)
Host	0	0	100	0
Vehicle 1	-83.33	0	100	0
Vehicle 2	83.33	0	100	0
Vehicle 3	80	3.75	80	0
Vehicle 4	150	3.75	80	0

Table B.2: Initial position and velocities of the vehicle in Scenario 2

Vehicle	x_0 (m)	y_0 (m)	v_0 (km/h)	a_0 (m/s^2)
Host	0	0	110	0
Vehicle 1	-91.67	0	110	0
Vehicle 2	91.67	0	110	0
Vehicle 3	80	3.75	80	0
Vehicle 4	150	3.75	80	0

Table B.3: Initial position and velocities of the vehicle in Scenario 3

Vehicle	x_0 (m)	y_0 (m)	v_0 (km/h)	a_0 (m/s^2)
Host	0	0	120	0
Vehicle 1	-100	0	120	0
Vehicle 2	100	0	120	0
Vehicle 3	80	3.75	80	0
Vehicle 4	150	3.75	80	0

Table B.4: Initial position and velocities of the vehicle in Scenario 4

Vehicle	x_0 (m)	y_0 (m)	v_0 (km/h)	a_0 (m/s^2)
Host	0	0	130	0
Vehicle 1	-108.33	0	130	0
Vehicle 2	108.33	0	130	0
Vehicle 3	80	3.75	80	0
Vehicle 4	150	3.75	80	0

Table B.5: Initial position and velocities of the vehicle in Scenario 5