KTH Electrical Engineering

Predictive control for autonomous driving

with experimental evaluation on a heavy-duty construction truck

PEDRO F. LIMA

Licenciate Thesis
Stockholm, Sweden 2016

KTH Royal Insitute of Technology
School of Electrical Engineering
Department of Automatic Control
SE-100 44 Stockholm
SWEDEN

TRITA-EE 2016:064
ISSN 1653-5146
ISBN 978-91-7595-984-9

Akademisk avhandling som med tillstånd av Kungliga Tekniska högskolan framlägges till offentlig granskning för avläggande av licentiatexamen i elektro- och systemteknik måndagen den 23 maj 2016 klockan 14.00 i Q2, Kungliga Tekniska högskolan, Osquldas väg 10, Stockholm, Sweden.

Tryck: Universitetsservice US AB

Abstract

Autonomous vehicles is a rapidly expanding field, and promise to play an important role in society. The goal of autonomous vehicles is to improve the traffic system in terms of higher efficiency, fewer accidents, lower fuel consumption, less pollution, and shorter travel times. In more isolated environments, such as mines and ports, vehicle automation can bring significant efficiency and production benefits and it eliminates repetitive jobs that can lead to inattention and accidents.

The results of this thesis are developed within the iQMatic project, which is lead by Scania CV AB in collaboration with Saab AB, Autoliv, Linköping University, and KTH, with the goal of developing a fully autonomous truck for mining applications by 2018. The thesis addresses the problem of lateral and longitudinal dynamics control of autonomous ground vehicles with the purpose of accurate and smooth path following. Clothoids are curves with linearly varying curvature, which are widely used in road design due to their smoothness properties. In the thesis, clothoids are used in the design of optimal predictive controllers aimed at minimizing the lateral forces and jerks in the vehicle.

First, a clothoid-based path sparsification algorithm is proposed to efficiently describe the reference path. This approach relies on a sparseness regularization technique such that a minimal number of clothoids is used to describe the reference path.

Second, a clothoid-based model predictive controller (MPCC) is proposed. This controller aims at producing a smooth driving by taking advantage of the clothoid properties. The motion of a vehicle traveling at low speeds is assumed to define a clothoid segment if the steering angle is chosen to vary piecewise linearly with respect to the traveled distance.

Third, an alternative formulation of a controller for smooth driving is presented. In this case, we formulate the problem as an economic model predictive controller (EMPC). In EMPC the objective function contains an economic cost (here represented by comfort or smoothness), which is described by the second and first derivatives of the curvature. This results in that the obtained curvature is approximately linear.

Fourth, the generation of feasible speed profiles, and the longitudinal vehicle control for following these, is studied. The speed profile generation is formulated as an optimization problem with two contradictory objectives: to drive as fast as possible (close to the speed limit) while accelerating as little as possible. Constraints are given by vehicle limitations and road geometry. The longitudinal controller is formulated in a similar way, but is implemented in a receding horizon fashion, and provides feasible reference speeds to a cruise controller.

Finally, experimental and simulation evaluation are performed. The EMPC is deployed on a Scania construction truck. The experimental evaluation with the EMPC demonstrates its good performance, since the deviation from the path never exceeds 30 cm and in average is 6 cm. A simulation environment is developed allowing evaluation and validation of the control design before deploying it in the experimental platform. The EMPC and the MPCC are compared with a pure-pursuit controller (PPC) and a standard MPC. The results demonstrate the effectiveness of both the MPCC and the EMPC. Furthermore, the EMPC clearly outperforms the PPC in terms of path accuracy and the standard MPC in terms of driving smoothness.

Sammanfattning

Autonoma fordon är ett forskingsområde som växer snabbt och det utlovar stora förändringar på samhället. Målet med självkörande fordon är att förbättra transportsystemet i termer av ökad effektivitet, färre olyckor, lägre bränsleförbrukning, mindre utsläpp och kortare restider. I mer slutna arbetsmiljöer, som till exempel gruvor och hamnar, kan självkörande fordon ge stora effektivitets- och produktivitetsvinster och dessutom försvinner monotona arbeten som kan leda till ouppmärksamhet och olyckor.

Resultaten i denna avhandling har tagits fram inom projektet iQMatic, som är ett projekt som leds av Scania CV AB i samarbete med Saab AB, Autoliv, Linköpings Universitet och KTH, med mål att utveckla och demonstrera självkörande lastbilar för gruvtillämpningar senast 2018. Avhandlingen handlar om lateral och longitudinell reglering av självkörande lastbilar för att uppnå noggrann och mjuk banföljning. En klotoid är en kurva med linjär kurvaturändring som till exempel används för att konstruera vägar som är behagliga att köra. I avhandlingen används klotoider för att ta fram optimala prediktiva regulatorer som minimerar sidokrafter och -ryckningar i fordonet.

Först presenteras en klotoidbaserad algoritm för att ta fram en effektiv gles beskrivning av referensbanan. Metoden bygger på en regulariseringsteknik för att beskriva banan med så få klotoidsegment som möjligt.

Därefter presenteras en klotoidbaserad modellprediktiv regulator (MPCC). Regulatorn eftersträvar mjuk körning genom att utnyttja klotoidernas mjuka kurvform. Ett fordons rörelser vid låg fart kan beskrivas med klotoidsegment om styrvinkeln ändras styckvis linjärt med körsträckan.

Det tredje bidraget är en alternativ formulering av en regulator för mjuk körning. I det här fallet formuleras problemet som en ekonomisk modellprediktiv regulator (EMPC). I EMPC så har man en ekonomisk term i kostnadsfunktionen som här representeras av komfort eller jämnhet och som beskrivs som första och andraderivatan av kurvaturen. Detta resulterar i att den erhållna kurvaturen blir ungefärligt linjär.

Vidare studeras genereringen av körbara hastighetsprofiler, samt longitudinell reglering för att följa dessa. Hastighetsprofilgenereringen formuleras som ett optimeringsproblem med två motstridiga mål: att köra så fort det går (nära hastighetsgränsen), men med så låg acceleration som möjligt. Bivillkor ges av fordonsbegränsingar och av vägens geometri. Den longitudinell hastighetsregleringen formuleras på ett liknande sätt, men implementeras med en ändlig rullande horisont (receding horizon) och ger körbara referenser till en farthållare.

Slutligen görs en utvärdering med hjälp av simulering och experiment på en anläggningslastbil. En experimentell utvärdering av EMPCn visar att den har väldigt bra prestanda då avvikelsen från referensbanan aldrig överstiger 30 cm och i medel endast är 6 cm. En simuleringsmiljö har utvecklats som möjliggör utvärdering och validering av reglerdesignen innan den implementeras på experimentplattformen. EMPCn och MPCCn jämförs med en pure-pursuitregulator (PPC) och med en standard-MPC. Resultaten visar effektiviteten av både MPCCn och EMPCn. EMPCn ger mycket bättre prestanda än PPCn när det gäller hur noggrannt den följer banan, och bättre än standard-MPCn när det gäller mjuk körning.

To my parents and girlfriend.

Acknowledgements

There are many who have contributed to the work presented in this thesis. First, I would like to thank to my main advisor Bo Wahlberg, for giving me the chance to work at the automatic control department and for his guidance and unending knowledge. A heartfelt thanks goes to Jonas Mårtensson for his advices, support, good mood, enthusiasm, and dedication. A special thanks goes to Marco Trincavelli for helping me finding a suitable research topic to work on and for all the priceless advice at every stage of my research. I would like to show my gratitude to Mattias Nilsson for all the hours we spent together to make things work. The work presented in this thesis would not have been possible without you.

At KTH, I would like to thank Antonio, Bart, Demia, Kuo-Yun, Miguel, Olle, Pedro, Rui, Sadegh, Sebastian, and Valerio for proofreading parts of the manuscript. Thanks to all my officemates, for making it the best office of the department. Thanks to all the Master's thesis students I have supervised for your fresh ideas and all the fruitful discussions. Thanks to the administrators Hanna, Anneli, Gerd, Kristina, Silvia and Karin for being always helpful and making everything easier.

At Scania, I would like to thank to the iQMatic project team, particularly to Johan, Jon, Kristian, Lars, Magnus, Marco, Marcello, Mattias, Samuel, and Tom for believing in me and for all the great work.

I want to thank to the Portuguese crew living in Stockholm for making life easier and funnier.

I am grateful to Vinnova (FFI) and to Scania CV AB for financing the iQMatic project and this research.

Last but certainly not least, there are no words to express my gratitude to my family and my girlfriend Madalena. Your love, care, dedication and patience were key to overcome the difficulties of this journey.

Pedro Lima
Stockholm, May 2016.

Contents

Chapter 1

Introduction

Autonomous vehicles are currently a reality. Today's society is vehicle-centered, as many have their own personal car, and others use public transportation. At the same time, trucks are indispensable to transport basic goods to the population. The enormous demand for more efficient transportation systems and increased need for goods creates congestions, as well as safety and environmental concerns. Autonomous vehicles have emerged to fulfill the need of more efficient and safer systems. In the past five years, many major vehicle manufacturers, suppliers and technology companies have started projects and collaborations around the autonomous vehicles theme. Autonomous vehicles is a rapidly expanding field, and promise to play an important role in society.

In this chapter, we motivate the need for autonomous vehicles and briefly describe their history, which goes back to the 1980s. Also, we define the concept of autonomous vehicle, in which we distinguish the different levels of vehicle autonomy. We support the motivation for autonomous vehicles in recent statistics about the economic and social impact of these systems.

The outline of this chapter is as follows. Section 1.1 provides the motivation for the need of autonomous vehicles. In Section 1.2, we define autonomous vehicle and discuss the associated challenges. In Section 1.3, we provide a brief history of autonomous vehicles. In Section 1.4, we mathematically formulate the problem tackled in this thesis. Finally, Section 1.5 provides the outline and contributions of the thesis.

1.1 Motivation

The increasing demand for economic growth permanently pressures transportation systems and primary products producers. At the same time, world institutions target increased efficiency in road utilization, fuel consumption and goods transportation [1]. A European Commission study concludes that traffic congestions cost Europe about 1% of the European Union's gross domestic product every year [2].

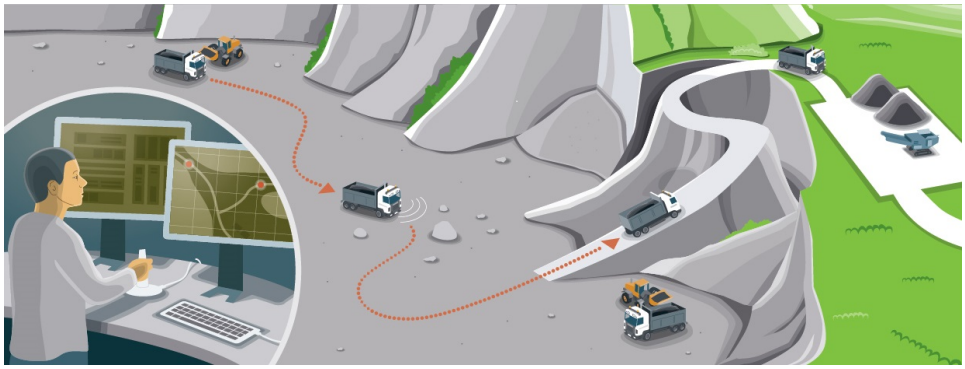


Figure 1.1: iQMatic project ambition (courtesy of Scania CV AB). In a mining site, autonomous trucks are able to accomplish complex tasks, such as the loading and unloading of debris. Remotely, in a command central, specialized humans are responsible for supervising all the actions and intervene if necessary.

Since the traffic intensity will continue increasing, decision makers face major challenges to solve this issue. The world health organization predicts that road traffic injuries will be the third most common cause of disability by 2020 [3]. According to USA's national highway traffic safety administration (NHTSA) research, 94% of accidents are caused by human error [4].

In gravel pits, mining areas, construction sites and loading terminals, workers are subject to dangerous environments and tedious tasks. Also, due to the increasing demand, the production sites are rapidly moving to remote locations and experiencing more complicated working conditions. Additionally, many companies have recently faced high administrative costs due to increased regulation about environmental compliance. These challenges make the development of autonomous vehicles an attractive idea. Autonomous driving will target more efficient fuel consumption and harder safety requirements. Traffic jams will be reduced since vehicles will be able to drive closer to each other and probably at higher speeds. Hence, inefficient fuel consumption, wasted commuting time and people frustration are translated into economic and financial benefits. Also, the automation of vehicles in isolated environments is a huge opportunity, since it is a cost efficient option and it eliminates repetitive jobs that can lead to inattention and accidents, resulting in an efficiency and production increase. Furthermore, comparing with an urban environment, a not so substantial investment and expansion in infrastructure and law adaption are required for autonomous driving [5–8]. By eliminating the human-in-the-loop factor, we expect to eliminate the number of human fatalities and increase productivity and efficiency in these working sites [9].

In this thesis, we address the problem of deploying autonomous trucks in isolated environments, such as mining sites. iQMatic is a project led by Scania CV AB,

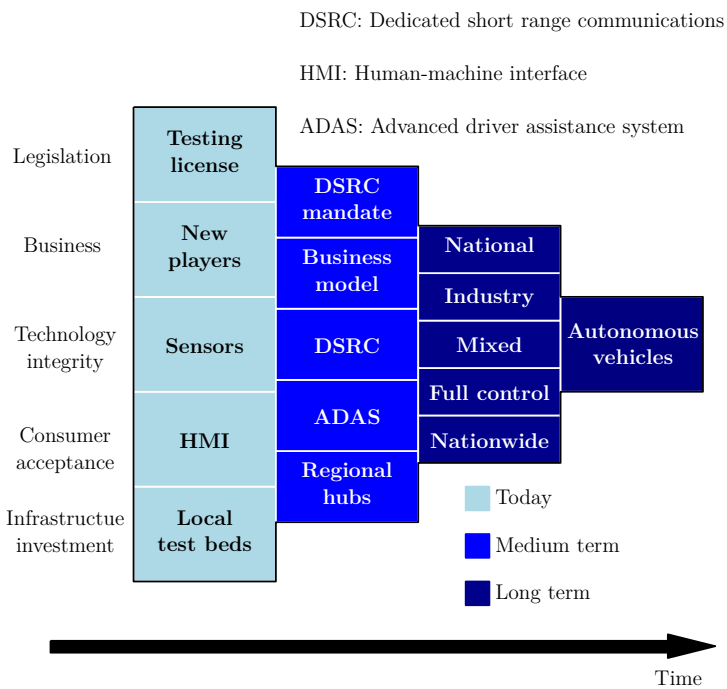


Figure 1.2: The journey to autonomous vehicles (inspired by [10]). The are five different challenges before autonomous vehicles is a reality: legislation, business, technology integrity, consumer acceptance and infrastructure investment. Each one of them plays a significant role in the autonomous vehicles outcome.

one of the most important truck manufacturers in Europe (16.5% market share). The project is funded by the Swedish government and automotive industry within the "fordonsstrategisk forskning och innovation" (FFI) program - strategic vehicle research and innovation. The iQMatic project partners are Saab AB, Autoliv, Linköping University, and KTH. An illustration of the project ambition is shown in Figure 1.1. The idea is to develop a fully autonomous truck for mining applications by 2018. At the deploying site, the fleet of autonomous vehicles navigates and cooperates to complete a set of missions given by a command central. There, humans supervise the entire fleet of autonomous vehicles and intervene in emergency and deadlock situations.

1.2 The journey to fully autonomous vehicles

Defining autonomous vehicle is not an easy task. It has been defined by the Nevada State law as "a motor vehicle that uses artificial intelligence, sensors and global

positioning system coordinates to drive itself without the active intervention of a human operator". However, NHTSA has also defined different vehicle automation levels [11]:

- **No-automation (level 0):** The driver is in complete control of the primary controls – vehicle brake, steering and throttle – at all times.
- **Function-specific automation (level 1):** The vehicle is capable of assisting the driver to regain control of the vehicle or stop faster than possible by acting alone. Examples are the electronic stability control (ESC), lane keeping and adaptive cruise controller (ACC).
- **Combined function automation (level 2):** The vehicle is capable of automating two primary functions. For example, using simultaneously lane keeping and the adaptive cruise controller.
- **Limited self-driving automation (level 3):** The vehicle is able to have full control of all safety – critical functions in specific traffic and environmental conditions. However, the driver is expected to be available for occasional control.
- **Full self-driving automation (level 4):** The vehicle is in complete control of the primary controls – vehicle brake, steering and throttle – at all times.

As of today, the great majority of commercially available vehicles have an autonomy level between 1 and 3. It is mandatory by law, in most countries, to have some advanced driver assistant system (ADAS), such as ESC, and most of the vehicles already combined several different ADAS. More recently, Tesla Motors launched the Model S [12], which is considered an autonomous vehicle with autonomy level 3, since the driver is still expected to be available for occasional control. The work described in this thesis is intended to be part of a fully autonomous heavy-duty vehicle (autonomy level 4) as the vehicle is suppose to be driverless.

Figure 1.2 illustrates the journey to autonomous vehicles. Autonomous vehicles face essentially five challenges: technology integrity, infrastructure investment, consumer acceptance, legislation and business. The current development of autonomous vehicles is built on merging sensor- and communication-based approaches. The automated highway idea, proposed at Futurama, the General Motors exhibition at 1939 world's fair in New York [13], did not succeed; however, 10 years later, the first steps towards autonomous driving were being made with the introduction of the cruise controller. Until today, other ADAS have been developed, such as anti-lock braking system, electronic stability control, adaptive cruise control, lane departure warning system and automatic parking [14]. This technology has been slowly replacing the human driver by taking the wheel in both critical and tedious situations. Typically, these situations can lead to human losses or decrease driving efficiency. ADAS combine the information perceived by a set of sensors ranging

from stereo cameras to long- and short-range RADARs in order to respond accordingly to the environmental dynamics. However, the available technology has not proven itself to be completely reliable, in the sense that the vehicle is still not able to accurately perceive its surroundings as humans do. For instance, while typical traffic situations can be repeated and learned using artificial intelligence techniques, there are always unforeseen scenarios that are difficult to handle in real time with current approaches. A considerable shortcoming of the sensor-based approach is the high-cost of integrating the available sensor technology in a commercial vehicle. Although the cost is currently decreasing, the entry price of an autonomous vehicle may be a barrier to the general public.

To help overcoming these challenges, vehicles are increasingly connected in order to cooperate with each other and to easily gather more information. Vehicles can communicate in real-time with each other – vehicle-to-vehicle communication (V2V) – and with the infrastructure – vehicle-to-infrastructure communication (V2I). Decentralizing the vehicle perception would decrease the need of standalone solutions, complex sensing devices, and artificial intelligence technology, while it would increase the redundancy of the system. A drawback is, again, the need of investment in both vehicles and infrastructures to be able to use these systems in practice.

Also, public acceptance is one of the greatest barriers to autonomous vehicles. Many drivers may take a slightly increased chance of accident in exchange for maintaining their ability to avoid an accident. When autonomous vehicles will start to be available to the general public, there will be for sure a transition period where driverless vehicles and human drivers will have to live together. In case of accidents between autonomous and non-autonomous vehicles there will be liability issues. This will affect the way insurance companies will look at the transportation business as well as how decision makers will influence the vehicle manufacturers and vehicle owners.

Also for taxi, bus or truck drivers, the idea of having autonomous vehicles may not be appealing. It is crucial to convince the part of the society that feels that jobs are lost due to the existence of machines. Automation replaces repetitive, unhealthy and dangerous jobs that humans should not do. On the one hand, not just the people that make a living driving will lose their jobs but probably some maintenance jobs will disappear and traffic police will no longer be as needed as nowadays. On the other hand, this will be compensated with the creation of intelligence- and information-based jobs. Lastly, from the business point of view, autonomous vehicles are estimated to burst new opportunities. For example, parking spaces will be much tighter since there is no need for the driver to get out of the vehicle; vehicle sharing will substitute vehicle owners as we know today; autonomous fleets of mining trucks will depend on the efficiency of the vehicle dispatcher algorithms.

The work developed in this thesis targets mainly isolated environments and the replacement of these type of jobs by autonomous vehicles.

1.3 A brief history of autonomous vehicles

The first boom in the autonomous vehicles field occurred in the decades of 1980 and 1990. Carnegie Mellon University (CMU, Pittsburgh, PA) presented its Navlab vehicles that operated in structured environments [15], UniBw Munich and Daimler-Benz's demonstrated autonomous driving at the European project PROMETHEUS [16], and the work done at University of Parma with the ARGO project was promising. PROMETHEUS was indeed a great milestone, since autonomous driving was demonstrated in a three-lane French Autoroute with speeds up to 130 km/h, where the vehicles were able to change lane and perform lane and vehicle tracking. In parallel, the idea of automated highway systems was being developed within the California partners for advanced transportation technology (PATH) program [17–19]. At the time, the PATH program main goal was to show technical feasibility rather than product development. In 1991, the PATH program was one of the first to demonstrate platooning at highway speed and using wireless communication [20]. Vehicle platooning consists in driving several vehicles close to each other in a convoy. This leads to decreased aerodynamical forces acting on the vehicles, which results in a more efficient fuel consumption and less greenhouse emissions [21]. Also, densely packing the vehicles results in an increased transportation capacity of the road network. Naturally, these effects are more outstanding when vehicles drive with inter-vehicle distances that are not considered safe for human drivers. Hence, it is crucial to have vehicle-to-vehicle communication and accurate control.

After that, several other competitions and demonstrations have taken place. It has been over 10 years since the first DARPA grand challenge in 2004 was won by the Stanford's robot, Stanley [22]. It autonomously completed an unrehearsed 230 km long course through the Mojave desert in less than 7 hours. After that, in 2007, DARPA organized a new competition, the urban challenge [23]. At the time, this event was spearheading, since the participants had to deal with other moving traffic and obstacles while respecting traffic rules. More recently, cooperative driving has captured the interest in the grand cooperative driving challenge GCDC 2011 [24] and soon in GCDC 2016 [25]. In these, autonomous driving is more focused on cooperative driving and vehicle platooning.

The PROUD project [26] presents an autonomous vehicle test on Italian public urban roads and freeways. The authors claim that it was the first test in open public urban roads with a driverless vehicle that had to cope with roundabouts, junctions, pedestrian crossings, freeway junctions, traffic lights and regular traffic. Also the Stadtpilot project [27] achieved successful urban autonomous driving. This time, the tests are performed in German roads and the paper presents not only technical details about the work but also guides the reader through some of the bureaucratic aspects of these tests.

Other countless competitions have taken place around the world to test and demonstrate driverless vehicles capabilities (e.g., [28–31]). In the last couple of years, autonomous vehicles have taken the next step by being released to the general public [12, 32, 33].

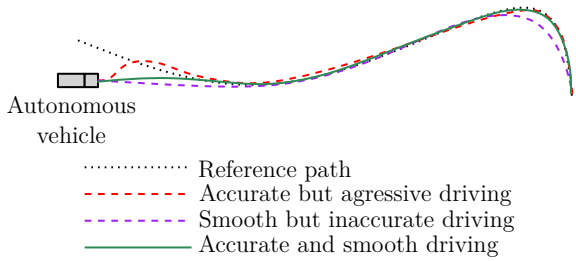


Figure 1.3: Illustration of the problem addressed in this thesis, which is the development of controllers for autonomous path following. These take into account the path tracking accuracy and the driving smoothness.

An extensive literature review is done in the background chapter (Chapter 2).

1.4 Problem formulation

The problem tackled in this thesis is the control of the lateral and longitudinal dynamics of an autonomous ground vehicle with the purpose of accurate and smooth path following. The autonomous vehicle is assumed to be equipped with a wide range of sensing technologies allowing self-localization and situation awareness to be done accurately. Furthermore, the vehicle has a clear objective (i.e., it knows its destination and the path to drive there). Figure 1.3 illustrates the path following problem. We intend to design a controller that steers the vehicle to a reference path under some constraints. Typically, the vehicle inputs are limited due to actuator limitations, and physical constraints are necessary to ensure safety, for example. Moreover, the vehicle dynamics can be described by a discrete nonlinear function. Also, we consider that the given path is obstacle free and feasible. Moreover, Figure 1.3 illustrates that a trade-off between accuracy and smoothness exists regarding the path following problem. Clothoids are widely used in road design fundamentally because the curvature varies linearly with the path arc-length, which produces low lateral jerk and acceleration. Consequently, we study the properties of clothoids and how this type of curves can be used in control design.

In short, the problem that we address in this thesis is to design controllers that solve the following generic optimal control problem:

$$\begin{aligned} \max \quad & \text{Tracking accuracy} + \text{driving smoothness} && \text{(1.1a)} \\ \text{s.t.} \quad & \text{Vehicle dynamics} && \text{(1.1b)} \\ & \text{Bounds on inputs} && \text{(1.1c)} \\ & \text{Safety constrains} && \text{(1.1d)} \end{aligned}$$

1.5 Thesis outline and contributions

In this section, we outline the thesis and its contributions (see Figure 1.4).

Chapter 2: Background

This chapter provides the background for the remainder of the thesis. We start by explaining the autonomous vehicle system architecture and introducing the main modules. We also provide a brief literature review for each module. We focus on the control architecture and detail our approach to integrate planner- and vehicle-agnostic controllers in an autonomous vehicle. Furthermore, we explain some of the key concepts that are used throughout the thesis, such as the concept of clothoid, the pure-pursuit algorithm and the model predictive controller.

Chapter 3: Modeling

This chapter introduces the vehicle modeling for simulation analysis and control design. For the simulation model, we have developed a 4-axle bicycle model to describe the lateral dynamics of a real truck. Besides the vehicle lateral dynamics, we also had to model the steering column dynamics. The model parameters are identified and validated using real data. The simulation model contains a longitudinal model that consists of a cruise controller and vehicle longitudinal dynamics provided by Scania CV AB. The model used for control design is a car-like kinematic bicycle model. Although the model is simple, since it does not contain any model of dynamics, it has proven to be sufficient for the type of application we target in the following chapters.

Chapter 4: Clothoid-based path sparsification

In this chapter, we address the path sparsification problem (i.e., describing a path with few waypoints). We present a novel approach to the path sparsification problem where a reduced number of clothoids is used to describe the reference path. This approach relies on a reweighed l_1-norm approximation of the l_0-norm. The clothoid-based path has few points, where each point has embedded the clothoid segment properties, resulting in a computationally inexpensive path description. The number of clothoids used to describe the path is dependent on the parameter that defines the maximum deviation allowed between the original and the clothoid-based path. The smaller the maximum allowed deviation, the greater the number of clothoids. The algorithm performance is compared for different maximum allowed deviations resorting to an illustrative example. Finally, we provide results based on real recorded paths.

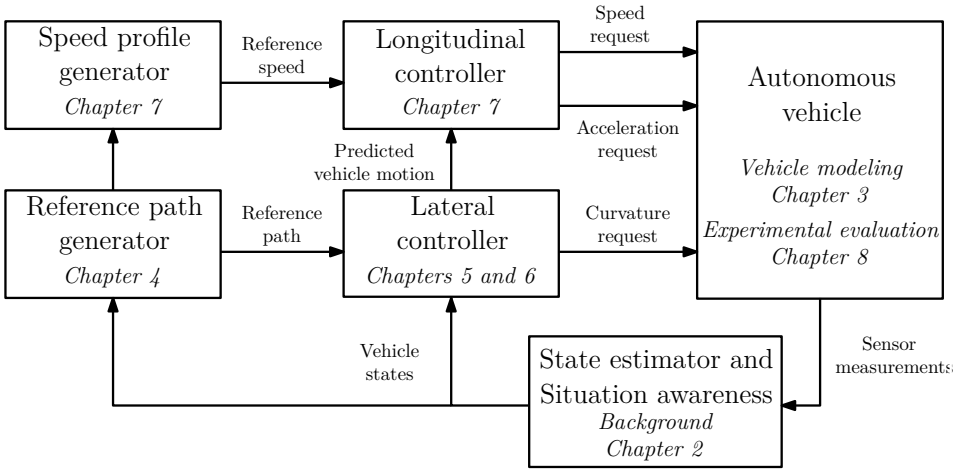


Figure 1.4: Thesis outline and simplified system overview.

Chapter 5: Clothoid-based model predictive controller

In this chapter, we propose a clothoid-based model predictive controller (MPCC) for vehicle lateral control. This controller provides a smooth driving for the vehicle by taking advantage of the clothoid properties. We use the path representation that results from using the method in Chapter 4. With our approach, we are able to compute smooth driving controls while predicting the vehicle behavior along a long prediction distance. We assume that the motion of a vehicle traveling at low speeds (such that the lateral vehicle dynamics are negligible) defines a clothoid segment if the steering angle is chosen to vary piecewise linearly with respect to the traveled distance. To describe the vehicle behavior between each pair of waypoints, it is enough to evaluate the clothoid parameters at those waypoints. We only need to compute the input of the vehicle at each clothoid starting point, allowing us to have a long prediction distance (with a small prediction horizon) with no significant increase of computational time and still maintain a good accuracy. We discuss the waypoints spacing influence on the vehicle motion prediction accuracy. Although it is unfeasible to use only the kink-points (i.e., the beginning or endpoints of a clothoid), it is still possible to have long prediction distances with good tracking accuracy.

Chapter 6: Model predictive controller for smooth driving

In this chapter, we present another method to deal with the vehicle lateral control – an economic model predictive controller (EMPC). The fundamental difference with the standard tracking MPC and the MPCC is that the driving economic cost (e.g., comfort or smoothness) is directly addressed in the objective function. Also, while a clothoid model is used to predict the vehicle motion in the MPCC, we now use a kinematic vehicle model. The economic cost is based on the minimization of the second and first derivatives of the curvature so that the obtained curvature is approximately linear. Since the curvature in clothoids varies linearly with the path arc-length, we use the smoothness and comfort characteristics of clothoid-driving to obtain a compact and intuitive controller formulation. We enforce convergence of the controller to the reference path with soft constraints that avoid deviations from the reference path. We analyze the controller tuning parameters influence on the performance of the controller and demonstrate the controller performance through illustrative examples.

Chapter 7: Longitudinal speed profiler and controller

In this chapter, we propose a method for optimal speed profile generation in specified paths with known semantic (e.g., speed limit and geometric information). We discuss the applicability of the speed profiler in clothoid-based paths. Furthermore, we translate the speed profile problem to a receding-horizon problem in order to produce feasible reference speeds that are given to a low-level vehicle speed controller. With our method, we do not target time optimal speeds but speed feasibility and safety. The speed profile generation is formulated as a convex optimization problem where the cost function has two contradictory terms, and the constraints are given by the vehicle limitations and road geometry. The first term of the cost function is related to the fact that we want to drive as close as possible to the speed limit. The second term of the cost function is used to prevent sudden accelerations. One can choose to penalize more or less the acceleration by regulating one single constant parameter. The goal of the longitudinal controller is to produce feasible speed requests for a low-level controller, such as a cruise controller. Also, it is responsible for tracking the speed profile as precisely as possible by adjusting the speed requests sent to the low-level controller. The controller is formulated in a similar way as the speed profiler. In this case, we optimize over a finite horizon instead of the full path, and we compute the optimal inputs in a receding-horizon fashion every sampling time. This controller is able to get the most recent vehicle behavior predictions from the lateral controller and sensor information, such that the speed is adapted accordingly.

Chapter 8: Experimental evaluation and discussion

This chapter intends to evaluate the performance of the MPCC and the EMPC proposed in Chapters 5 and 6. This comparison is performed resorting to a simulation environment that strongly resembles the experimental platform and also to experimental tests in a Scania construction truck. We benchmark the proposed controllers against a standard MPC approach and a pure-pursuit (PPC) controller. The evaluation is performed in three different paths: an artificially generated double S-curve that allows the analysis to focus on the controllers main aspects, a precision track that resembles a mining scenario, and a high-speed test track that reminds of a highway situation. The last two reference paths are real tracks from Scania's test tracks facilities near Södertälje, Sweden. The main goal is to evaluate the controllers path tracking accuracy and their driving behavior in terms of smoothness. We discuss the influence that non-clothoid-based and clothoid-based paths have on the MPCC performance. Moreover, we relate the formulations of the standard MPC with the EMPC to determine the influence of the economic cost on the controller performance.

Chapter 9: Conclusions and future work

The final chapter of the thesis summarizes the work presented and emphasizes potential future research directions.

Publications by the author

- **P. F. Lima**, M. Trincavelli, J. Mårtensson, and B. Wahlberg.
  Clothoid-based model predictive control for autonomous driving.
  In *IEEE European Control Conference*, July 2015.

- **P. F. Lima**, M. Trincavelli, J. Mårtensson, and B. Wahlberg.
  Clothoid-based speed profiler and control for autonomous driving.
  In *IEEE Intelligent Transportation Systems Conference*, September 2015.

- **P. F. Lima**, M. Trincavelli, M. Nilsson, J. Mårtensson, and B. Wahlberg.
  Experimental evaluation of predictive lateral control for an autonomous truck.
  In *IEEE Intelligent Vehicles Symposium*, June, 2016.

- **P. F. Lima**, M. Nilsson, M. Trincavelli, J. Mårtensson, and B. Wahlberg.
  Economic model predictive control for autonomous driving.
  Submitted for journal publication, 2016.

The order of the author names reflects the workload, where the first had the most important contribution. In all the publications, the thesis author is the main author and the results are mostly his own original work.

Chapter 2

Background

The idea of autonomous vehicles has been preconized for more than 50 years [13]. Yet, only in the last few decades technology has made it possible, and a considerable progress has been made in order to help the driver in the process of decision making. In this chapter, we describe the main modules of an autonomous vehicle system architecture with major focus on the vehicle control module. We provide a brief literature review regarding each module of the system architecture. Furthermore, we go into detail on the literature related to lateral and longitudinal control for autonomous vehicles. We establish the background for the rest of the thesis, including the concept of clothoids, model predictive control (MPC) and pure-pursuit control.

The outline of this chapter is as follows. In Section 2.1, an autonomous vehicle system architecture is proposed and each module is described. The concept of clothoid is explained in Section 2.2, while the model predictive control framework is the topic of Section 2.3. Lastly, Section 2.4 summarizes the chapter.

2.1 System architecture

In this section, we explain a possible autonomous vehicle system architecture and detail our contribution to its development. The architecture in Figure 2.1 provides an overview of an autonomous vehicle guidance system. We highlight four modules, namely, the environment fusion, the perception and localization, the motion planning, and the motion controller. Next, we describe these modules in detail.

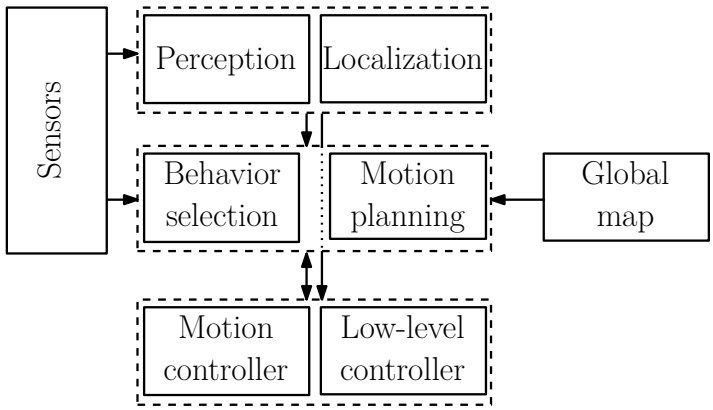


Figure 2.1: Simplified system overview of an autonomous vehicle. An autonomous vehicle perceives the environment using its sensors. At the same time, a global map helps the system in the process of self-localization. This information is processed and reorganized in the environment fusion block that provides the information about the state of the autonomous vehicle, and its surroundings, to the behavior selection and motion planning block. There, the autonomous vehicle decides how to proceed, providing a local plan to the motion controller. In this final step, the vehicle is actuated and guided precisely through this local obstacle-free path.

Perception and localization

This module takes data generated from a set of sensors and fuses them into a consistent representation of the world that is suitable for autonomous driving. The typical sensor platform usually consists of several RADARs, LIDARs, global positioning system (GPS) antennas, inertial measurement units (IMUs), and monocular and stereo cameras. Hence, the benefits of accurate angular resolution of cameras together with the precise distance information of RADARs and LIDARs is fused. An accurate environment perception is challenging due to the fact that autonomous vehicles may operate in dynamic environments. Besides having both static and moving obstacles, the environmental conditions are constantly changing (e.g., visibility conditions). Common approaches to overcome obstacle detection include the creation of an occupancy grid map [34], where each grid cell stores the probability for the existence of an obstacle [22, 35, 36]. Besides deciding whether there is an obstacle or not, these cells may be used to characterize the environment in other ways. For example, in determining the terrain type, the obstacle type and helping self-localizing the vehicle [37]. The biggest challenges are concerned with loop

closure and the ability to free cells if the obstacle is not static. These challenges typically arise the when the goal is to obtain a 360° map of the environment. In the case of dynamic objects, Bayesian filters, extended Kalman filters and particle filters are used to perform tracking. Then, the object is classified according to how it moves [38] or what it looks like [39, 40]. In case of other traffic, the ability to predict the intentions of other vehicles is essential to properly assign a behavior to the autonomous vehicle (e.g., to decide whether the autonomous vehicle will yield or go out into an intersection) [41]. This is a challenging problem, which has been approached by applying probabilistic and machine learning methods [42].

The main task of the localization module is to accurately estimate the autonomous vehicle state. An inaccurate localization may jeopardize the entire system and it is crucial for correct decision making and for a precise, safe and comfortable motion. This problem has been addressed many times. Typically, the localization of the vehicle is performed by fusing all available sensor information. Sensors gather measurements ranging from odometry to acceleration and can be combined with offline saved maps. Furthermore, including knowledge about the vehicle motion in the measurement data provides smoother state estimations. For instance, Thrun et al. [22] have implemented in Stanley, the first autonomous car to finish a DARPA competition, an unscented Kalman filter, which incorporates observations from the GPS, the GPS compass, the IMU and the wheel encoders. This has proven itself successful both when the GPS is available and when it is not. Other approaches rely on the existence of a detailed digital map where the vehicle is relatively positioned. In [32], a Mercedes S-class 500 localizes itself using feature-based localization and lane-marking-based localization. The first system detects landmarks around the ego position of the vehicle and manipulates them to match the map saved offline. The latter is used mainly in urban environments where lane markings are common. The observed lane markings are also compared with the expected lane markings in the map [43].

Behavior selection

This module is responsible for selecting the autonomous vehicle behavior (e.g., lane keeping, overtaking, yielding, parking, off-road driving, standing still, among other behaviors). It receives the processed sensor data from the perception and localization module and computes the appropriate vehicle behavior. Hierarchical finite state machines and hierarchical concurrent finite state machines are popular when performing this task and they are successfully used in [32, 35, 44]. Hierarchy allows the development of super- and sub-states, which allows a clear and comprehensible top-down modeling of reactive systems. Concurrency allows setting up multiple state charts that react to the same events independently. For example, a generic behavior (e.g., "drive") can be separated in several substates (e.g., "drive start", "drive stop", and "lane change"). More recently, behavior trees have been the focus to model these behavior selection algorithms [45]. Behavior trees were originally used in video games [46] to control artificial intelligence controlled players. Behav-

ior trees overcome the limitations of the (hierarchical) (finite) state machines in the sense that their implementation and formulation is usually simpler, more scalable and modular.

Motion planner

Motion planning is fundamental to cope with dynamic environments that might require computation of new paths. Motion planning problems can be divided in on-road and off-road driving. Also, the planning framework can be split in two planning layers. A global planner produces a high-level plan with long term goals, while a local planner ensures that the generated paths respect the vehicle limits and avoids possible obstacles. Early path planning work was not focused on path smoothness or planning respecting non-holonomic constraints. Nevertheless, in the past two decades, autonomous driving has triggered research that targets those issues, and the problem of path planning has been addressed typically resorting to sampling-based methods, such as rapidly-exploring random trees (RRT) [47], lattice-based motion planners [48], optimal control theory [49] and hybrid approaches [36].

RRTs are based on incrementally growing a tree rooted in the current state by randomly sampling the search space. In each iteration, the closest tree node to a sample is used as a starting point for simulating a system trajectory that targets that sample. The trajectory is checked for collision and the end point added to the tree. Eventually, a node reaches the goal point and the best trajectory is sent to the motion controller. RRTs were successfully used in autonomous driving in [50]. In fact, the authors extend the concept of RRT by proposing the so called closed-loop RRT (CL-RRT) algorithm. The framework makes use of a low-level controller that, together with a system model, plans over the closed-loop dynamics. The need for fast re-planning motivated the extension in [51], where a sampling recovery technique using brake profile regeneration is proposed.

Lattice planners also expand a tree rooted in the current state. However, the expansion is done resorting to offline pre-computed motion primitives, which allow the state space to be explored *a posteriori* by classical artificial intelligent graph-search algorithms, such as A* or D*. For example, [44] uses a lattice planner together with D* for autonomous driving, while a hybrid A* is used in [36] to generate smooth paths in an unknown environment. A modified A* algorithm is proposed that, instead of using discrete nodes, captures the continuous state of the vehicle motion. Afterwards, a non-linear optimization is also used to improve the quality of the solution.

In the case of autonomous vehicles for mining applications, the need for a local planner is crucial since the landscape is constantly being altered as a consequence of the mining works, mission end points might be dependent on external factors, and the maneuvering areas are typically small.

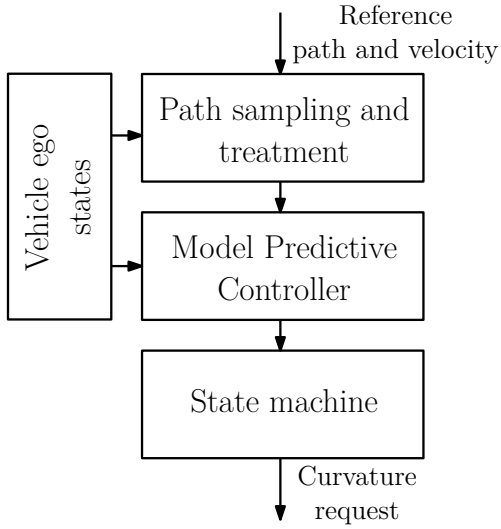


Figure 2.2: Simplified controller block diagram. A controller harness is developed to allow the deployment of a controller in an autonomous vehicle. In this way, the deployed controller, in this case an MPC, is totally planner-agnostic on the upper level, as well as vehicle-agnostic on the lower lever.

Motion controller

The main topic of this thesis is the motion controller. The motion controller receives an obstacle-free and feasible reference path from the motion planner. It also receives the vehicle ego states estimated by the perception and localization module. In fact, to handle the delay present in the system[1], the controller uses a forward-prediction of the vehicle states. The controller main task is to stabilize and guide the vehicle along the path.

The problem of path following can be divided into two control problems, namely, the lateral and the longitudinal control. Although the vehicle lateral and longitudinal dynamics are coupled, the control can be decoupled [52]. The overall problem has been addressed in different ways (e.g., fuzzy control [53], feedback control [54] and pure-pursuit algorithms [55]). Recently, MPC has become quite popular due to its ability to predict the vehicle behavior for a given set of inputs [56]. Moreover, it handles nonlinear time-varying models and constraints in a systematic way [57,58]. In MPC, a chosen cost function is minimized and the optimal sequence of inputs is

[1]We kindly refer the reader to Chapter 3, where we describe the vehicle modeling and parameter identification, namely the steering dynamics.

computed in order to better follow a specified trajectory under known constraints on states and inputs. An input subset of the optimal sequence is applied to the vehicle and the process is then repeated.

MPC has been widely used to address the autonomous vehicle guidance problem (e.g., [59–66]). A particular area where MPC has been applied is active front steering (AFS). AFS modifies the vehicle steering angle without changing the steering wheel position, helping the driver to drive around difficult corners and improving stability. In [59], the authors combine differential braking and AFS to improve yaw stability. There, an MPC controller is designed to track a driver-requested yaw rate using offline computed optimal solutions by multiparametric programming [67]. The combination of AFS with braking is also explored in [60]. In this case, the intended contribution is to develop an autonomous vehicle capable of avoiding obstacles, rather than helping a human driver. In this work, the controller relies on a nonlinear MPC (NMPC), introduced in [61]. There, an NMPC controller is presented in order to stabilize a vehicle along a desired path while fulfilling its physical constraints. Since a nonlinear vehicle model is considered, the resulting MPC controller has a considerable computational burden, which makes it difficult to be implemented in real-time. MPC controllers limit experimental validation due to the use of complex models for predicting the vehicle behavior, which makes the optimization complex and slow. Also, adding constraints, such as imposing certain comfort measures, increases the problem complexity and the computational burden. To overcome these problems, a linear time-varying (LTV) MPC for AFS is proposed in [64]. There, the discrete time linear system is based on successive online linearizations of the nonlinear vehicle model, and the stability conditions for LTV-MPC controllers are also presented. Following the same reasoning, a control architecture based on a LTV-MPC is presented in [65] to address the lane keeping and obstacle avoidance problems for a passenger car driving on low curvature and low friction roads. A study about the use of kinematic and dynamic vehicle models for MPC control design is detailed in [63]. The authors show that the kinematic model has better forecast errors when discretized at 200 ms, compared to 100 ms. Therefore, the authors motivate the MPC design based on a kinematic vehicle model since it is simpler than the dynamical model and still considerably accurate. In [68], the authors integrate an environment model with a stochastic predictive control approach to estimate and predict the state of surrounding vehicles. Thereafter, the authors demonstrate the ability of a constrained MPC to achieve certain objectives while avoiding collisions with other vehicles. Another approach to vehicle stabilization using MPC is presented in [66]. In this work, the vehicle motion is bounded within the region of the state space that does not contain unstable vehicle dynamics. In [62], a hierarchical framework based on MPC for autonomous vehicles is presented. At the high-level MPC, a desired trajectory is computed online using simplified models, which is then provided to a low-level MPC that computes the vehicle inputs using a detailed nonlinear vehicle model.

In the field of speed planning and control, Velenis et al. have investigated the optimal velocity profile generation with given acceleration limits [69–72]. The au-

thors propose a semi-analytical method to generate optimal velocity profiles on a specified path for a vehicle with known acceleration limitations. The problem of obtaining the optimal velocity profile is formulated as an optimal control problem using a maximum allowed centripetal force as a constraint and solved it with the Pontryagin's maximum principle. In [69], a theoretical analysis of the method, with a formal proof regarding optimality, is presented for a point mass vehicle. Moreover, in [70], the work is extended to include a half-car model and consequently constraining both front and rear axis centripetal forces. Afterwards, in [71], the infinite-horizon optimal control problem is implemented in a receding-horizon fashion. A dynamic scheme that determined the planning and execution horizon length is used to guarantee robustness.

Furthermore, within the same idea of minimum-time optimal speed profile, [73] proposes a robot path planning technique for differential drive robots generating a spline that minimizes the time of reaching an end point with desired heading and velocity. The proposed method calculates the optimal position of the spline control points by minimizing the time needed to perform the path. The robot speed is computed using the geometry of the path and constraining the lateral and longitudinal accelerations. Moreover, in [74] the problem of minimum-time optimal speed profile is also addressed but, in this case, a single-track model including tire models and load transfer is considered. In addition, the strategy is based on the combination of an equivalent formulation of the optimal control problem with a continuation method to find a candidate minimum-time trajectory.

Another active related research is the one considering eco-driving (i.e., concerns with fuel consumption in autonomous driving). In [75], the vehicle efficiency is optimized using a dynamic programming approach and the authors claim that fuel savings up to 16% are possible by applying this method. Finally, in [76], the authors developed a smooth path and speed planner for an automated public transport vehicle in unstructured environments. In this work, the focus is on efficiency and comfort rather than time optimality. Besides producing a smooth path with bounded continuous curvature, the velocity planner uses the semantic information of the road – maximum speed, longitudinal and lateral accelerations, and jerk – and the constraints on the lateral acceleration and jerk to produce comfortable speeds. The results of the proposed strategy were contrasted with real driving maneuvers performed by human drivers in an automated public transport vehicle.

Figure 2.2 depicts a controller architecture that we developed in order to integrate the motion controller in the overall system architecture. The two most outer layers, the path sampling and treatment and the state machine, were developed to create a vehicle- and planner-agnostic controller in the middle.

The reference path contains more than one waypoint where at least one has non-zero velocity. Otherwise, either there is no path, since only one waypoint is created, or the path is to be performed with null speed, which is not possible. At the path sampling and treatment module, the path is manipulated such that the path coordinates are set with respect to the vehicle local coordinate system. Also, the controller does not depend on the spatial or time sampling of the path received

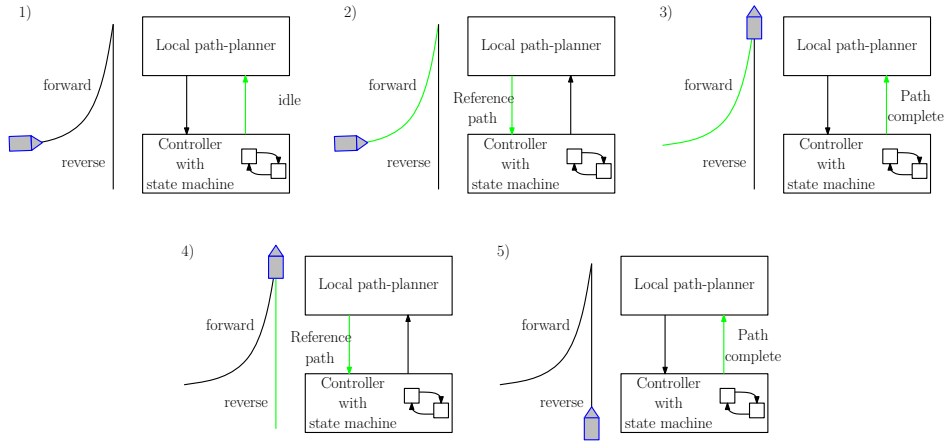


Figure 2.3: Interaction between the planner and controller. Example of a simplified forward-stop-reverse scenario messaging between the planner and the controller. Although it is not represented, the autonomous vehicle controller interchanges more information with the planner, such as the portion of the planned path that is being executed and what is the tracking deviation.

from the planner, since in this module the path is velocity-dependent sampled (i.e., the spacing between waypoints depend on a predefined time gap). In the center, an MPC controller is depicted. The length of the MPC prediction horizon has to be set *a priori*, and when the number of waypoints received is less than that length (but still larger than one), we extrapolate the path with the same curvature as the last segment, to assure that the MPC has enough points to work with.

On the other end of the diagram, the state machine is essential for the cooperation between the controller and the local planner. Figure 2.3 sketches a typical maneuvering scenario, where the vehicle has to move forward, stop, move backwards and stop again. The state machine is responsible for making sure that the controller does not end in a deadlock and that the vehicle inputs are state-dependent. For instance, the vehicle must stand still if the path is too far away or if it is infeasible to be tracked. Furthermore, the motion planner is required to stamp each waypoint with unique IDs. These are used to keep track of which part of the path the vehicle has executed. The vehicle is projected on the path through the interception between its normal and the linear interpolation between the two closest waypoints. Moreover, the ID of the closest waypoint behind the vehicle (i.e., the waypoint previously executed), is sent back to the planner. When the ID corresponds to a stopping point, a path completion flag is activated. Also, we record the distance from the path, specifically the Euclidean distance between the vehicle and its projection on the path.

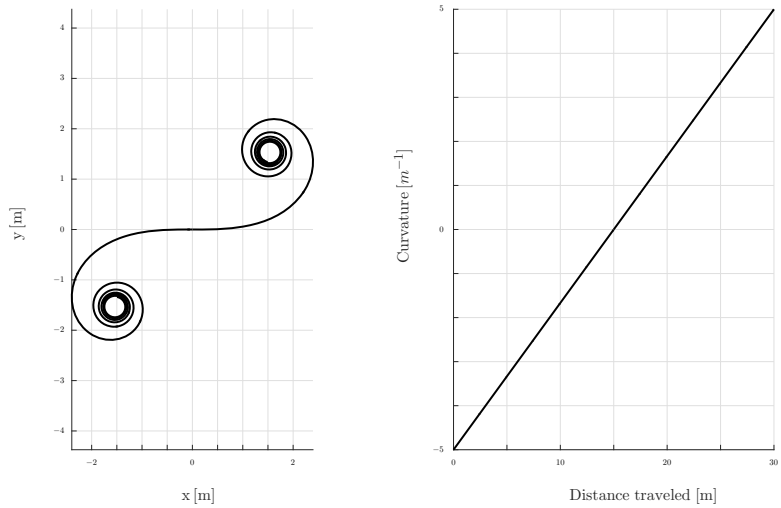


Figure 2.4: Example of a clothoid. Its curvature grows linearly from -5 m^{-1} to 5 m^{-1} and the arc-length is 30 meters.

2.2 Clothoids

One of the core ideas of this thesis is the use and exploitation of clothoids properties in the controller design. Clothoids are also found in the literature under the names of "Euler spiral" and "Cornu spiral". Mathematically speaking, the underlying equation describing a clothoid is known as a Fresnel integral. The many different names are due to the historical background of this curve, which was reinvented many times with completely different purposes. There is, however, one discovery that directly connects clothoids with autonomous driving. In 1890, Arthur Talbot derived the equation of clothoids to serve as a transition curve in railway tracks [77]. In a clothoid, the curvature varies linearly with the path arc-length (see Figure 2.4). In this way, straight lines and circular arcs are connected to minimize the shock and unpleasant lurch of trains due to the curvature discontinuity in transition curves. Currently, clothoids are the standard transition curve in road design [78]. In the case of ground vehicles, clothoidal road design provides a linear change in the vehicle steering angle when performing a turn, preventing sudden changes in lateral acceleration and therefore, yielding low values of lateral jerk [79]. We also use the smoothness and comfort characteristics of clothoid-driving in our controller design.

Early work on path planning for nonholonomic vehicles focused on the generation of Dubins' curves (i.e., paths obtained by connecting circular arcs and straight lines) [80, 81]. These type of paths do not have a continuous curvature. Therefore,

to follow them, a nonholonomic vehicle must stop and reorient its steering wheels; otherwise, the path tracking is not possible without major errors in position and in orientation at the curvature discontinuities since instantaneous changes in steering mechanisms are physically impossible. This problem triggered the emergence of smooth path planning [82], namely, path planners that produce paths with B-splines [83], polar splines [84], cubic spirals [82] and clothoids [85]. Clothoids have the disadvantage of not having a closed-form expression, but the advantage of being the curve with the simplest curvature function that provides the smoothest transition curves making them easy to track. Clothoids have also been widely used in model based detection systems for determining road boundaries [86] and lane recognition [87, 88].

The general parametric form of a clothoid curve is given by

$$x(s) = x_0 + \int_0^s \cos(\theta(\zeta))\ \mathrm{d}\zeta, \tag{2.1a}$$

$$y(s) = y_0 + \int_0^s \sin(\theta(\zeta))\ \mathrm{d}\zeta, \tag{2.1b}$$

$$\theta(s) = \theta_0 + \int_0^s \kappa(\zeta)\ \mathrm{d}\zeta, \tag{2.1c}$$

$$\kappa(s) = \kappa_0 + cs, \tag{2.1d}$$

where $x(s)$ and $y(s)$ are the Cartesian coordinates of the curve, and $\theta(s)$ is the angle defined by the tangent along the curve. The curvature κ varies linearly with respect to the traveled distance with a constant curvature change rate c. The curve is parameterized in the space domain $s \in \mathbb{R}^+$, which represents the distance along the path. Integrating the right-hand side of (2.1a) and (2.1b) yields the so called Fresnel integrals, which do not have a closed-form expression [89].

2.3 Model predictive control

MPC is an optimal control framework in which the current control input is obtained by iteratively solving an open loop optimal control problem. The current state of the plant is used as the initial state and the optimization is based on the predicted state of the system, which relies on the system model. The solution of the optimal control problem is a control input sequence and an input subset of this sequence is applied to the plant. The process is then repeated at constant time intervals. The biggest advantages of MPC are the systematic way of designing the controller and its ability to cope with both input and state constraints. The latter represent limitations on actuators or limitations that arise from physical, economic or safety constraints. On the other hand, the optimization problem may be computationally demanding. Consequently, MPC was originally used to improve the performance of petrochemical industries in the 70's [90]. At the time, the low computation power

available limited the use of MPC to slow dynamics processes. More recently, MPC has been subject to intensive research both in theoretical and practical aspects. Feasibility and closed-loop stability may be guaranteed [57] and MPC has been successfully applied to fast systems, such as autonomous vehicles [64].

In this section we introduce the concept of MPC and its variants, closely following [91] but opting for another notation. We kindly refer the interested reader to [92–94] for a thorough description of the topic.

Nonlinear model predictive control

Consider the discrete-time nonlinear system

$$z(t+1) = f(z(t), u(t)), \tag{2.2}$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ with $f \in \mathcal{C}^1$ describes the discretized system dynamics at every discrete-time instant $t \in \mathbb{Z}^+$, $z(t) \in \mathbb{R}^n$ is the system state vector and $u(t) \in \mathbb{R}^m$ is the system control input at time instant t. The system (2.2) is subject to constraints, particularly

$$z(t) \in \mathcal{Z}, \quad u(t) \in \mathcal{U}, \quad \forall t \in \mathbb{Z}^+, \tag{2.3}$$

where $\mathcal{Z} \subseteq \mathbb{R}^n$ and $\mathcal{U} \subseteq \mathbb{R}^m$ are polytopes [95].

Let $H \in \mathbb{Z}^+$ be the prediction horizon. We can formulate the problem of steering the system (2.2) to the origin as a finite-time optimal control problem with the cost function $C : \mathbb{R}^n \times \mathbb{R}^{H\times m} \to \mathbb{R}^+$ as

$$\mathscr{C}(z(k), U(k)) = \sum_{k=t}^{t+H-1} l(z(k), u(k)), \tag{2.4}$$

where $U(t) = [u(t), ..., u(t+H-1)]$ is a sequence of inputs over the time horizon H at time instant t, $z(k)$ for $k = t, ..., t+H$ is the state trajectory obtained by applying the control sequence $U(t)$ to the system (2.2), starting from the initial state $z(t)$, $l \in \mathcal{C}^1$ and $l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^+$ is the stage cost.

With (2.4) in mind, at each sampling time, we solve the optimization problem

$$\begin{aligned}
\min_{U(k)} \quad & \mathscr{C}(z(k), U(k)) && \text{(2.5a)}\\
\text{s.t.} \quad & z(k+1) = f(z(k), u(k)) \quad k = t, ..., t+H-1, && \text{(2.5b)}\\
& u(k) \in \mathcal{U} \quad k = t, ..., t+H-1, && \text{(2.5c)}\\
& z(k) \in \mathcal{Z} \quad k = t, ..., t+H-1, && \text{(2.5d)}\\
& z(k) = z(t). && \text{(2.5e)}
\end{aligned}$$

Then, we apply $u(k)$ to the system and repeat the optimization with the new measured states.

Linear time-varying model predictive control

The nonlinear problem formulation (2.5), depending on the functions l and f, can be a non-convex optimization problem and therefore, computational expensive. Also, in this case, there is no guarantee that a solver converges to a global minimum and, typically, the time to solve these type of problems makes it unfeasible for real-time applications. Therefore, an LTV-MPC is presented, which reduces the computation complexity of the problem by approximating the nonlinear system model with a linear time-varying model.

Consider the state $z_0 \in \mathcal{Z}$ and the input $u_0 \in \mathcal{U}$. Let $z_0(k)$ be the state trajectory obtained by applying a constant input sequence $u_0(k) = u_0$ for $k \geq 0$ to the system (2.2) with $z_0(0) = z_0$. Then, the linearization of (2.2) around $z_0(k)$ and $u_0(k)$ yields the system

$$\delta z(k+1) = A_0(k)\delta z(k) + B_0(k)\delta u(k), \tag{2.6}$$

where $\delta z(k) = z(k) - z_0(k)$ and $\delta u(k) = u(k) - u_0$. $A_0(k) \in \mathbb{R}^{n\times n}$ and $B_0(k) \in \mathbb{R}^{n\times m}$ are defined as

$$A_0(k) = \left.\frac{\partial f(z,u)}{\partial z}\right|_{\substack{z\,=\,z_0(k),\\ u\,=\,u_0(k)}}, \quad B_0(k) = \left.\frac{\partial f(z,u)}{\partial u}\right|_{\substack{z\,=\,z_0(k),\\ u\,=\,u_0(k)}}, \tag{2.7}$$

The LTV system (2.6) is a first-order approximation of the nonlinear system (2.2). It describes the deviations from the state trajectory $z_0(k)$ when a constant sequence of amplitude u_0 is applied to the plant.

Thus, at each sampling time we solve the following problem with respect to $\delta U(k) = U(k) - u_0$:

$$\begin{aligned}
\min_{\delta U(k)} \quad & \mathscr{C}(z(k), U(k)) && \text{(2.8a)}\\
\text{s.t.} \quad & \delta z(k+1) = A(k)\delta z(k) + B(k)\delta u(k) \quad k = t, ..., t+H-1, && \text{(2.8b)}\\
& u(k) \in \mathcal{U} \quad k = t, ..., t+H-1, && \text{(2.8c)}\\
& z(k) \in \mathcal{Z} \quad k = t, ..., t+H-1, && \text{(2.8d)}\\
& z(k) = z(t). && \text{(2.8e)}
\end{aligned}$$

Linear time-varying model predictive control for reference tracking

The MPC defined in (2.8) steers the system to the origin. Let $z_{\text{ref}}(t) \in \mathcal{Z}\ \forall t \in \mathbb{Z}^+$ be a time-varying reference trajectory. Also, let $u_{\text{ref}}(t) \in \mathcal{U}\ \forall t \in \mathbb{Z}^+$ be the sequence of inputs that generated the reference trajectory. We modify the cost function (2.4) to steer the system to the given reference. Also, note that, by construction,

$$0 = l(0,0) \leq l(\tilde{z}(k), \tilde{u}(k)) \quad \forall z \in \mathcal{Z},\ \forall u \in \mathcal{U}.$$

To be specific,

$$\mathscr{C}(\tilde{z}(t), \tilde{U}(t)) = \sum_{k=t}^{t+H-1} l(\tilde{z}(k), \tilde{u}(k)), \tag{2.9}$$

where $\tilde{z}(t) = z(t) - z_{\text{ref}}(t)$ and $\tilde{u}(t) = u(t) - u_{\text{ref}}(t)$.

In case of unknown model errors or disturbances, zero offset may be achieved by designing an MPC with integral action by appropriately including the disturbance and reference dynamics in the prediction model [96, 97].

Economic model predictive controller

Economic MPC (EMPC) [98] is a recent variant of MPC, which was born to efficiently combine the hierarchical separation between economic and dynamic performance. Typically, an upper layer provides the optimal set-point, from the economic point of view, to the MPC, which is responsible for stabilizing the system and reject any disturbances, from the dynamical point of view. Although recent, it has been applied, for example, to building temperature and smart grid management [99,100]. To our best knowledge, the EMPC framework has never been applied to autonomous driving.

In the standard MPC cost function, the cost function is, by construction, zero at the optimal reference trajectory. The same is not true in the economic MPC, since the operating cost of the system is included in the cost function. Let $e(z, u) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^+$ be the economic cost. Thus, we can modify the cost function (2.9) even further to steer the system to the given reference while minimizing the operating cost of the plan, yielding

$$\mathscr{C}(\tilde{z}(t), \tilde{U}(t)) = \sum_{k=t}^{t+H-1} l(\tilde{z}(k), \tilde{u}(k)) + e(z(k), u(k)). \tag{2.10}$$

2.4 Summary

An autonomous vehicle system may be decomposed in different modules, namely, perception and localization, behavior selection, motion planning, and motion controller. The environment fusion is responsible for estimating the vehicle position with respect to the world environment, determining also the state of the vehicle surroundings. The behavior selection and the motion planner module receive a detailed map with this information. These modules are responsible for appropriately deciding which behavior the vehicle should have and which path the vehicle should take. Then, the motion controller is responsible for accurately executing the planned path. In the last few decades, the field of autonomous vehicles has been subject to intensive research. We covered a part of the extensive available literature, where we described the significant progress made in the last years with the maturation of systems, such as ADAS that are available in the majority of the commercial vehicles. This technology helps the driver intervening whenever a critical situation is triggered, and therefore, mitigating the risk of accidents. Lastly, we provided an overview on the main concepts to be exploited in this thesis, particularly clothoids for path representation and MPC for reference tracking.

Chapter 3

Vehicle modeling

Model-based control design is highly dependent on the quality of the models provided. On one hand, an accurate system model provides good predictions. However, these models are typically complex and computationally expensive. On the other had, simple models are less computationally demanding but are unable to provide good predictions. In this chapter, we describe the vehicle modeling for two distinct cases: simulation analysis and control design. The purpose of this thesis is to design and evaluate controllers on a heavy-duty vehicle. Therefore, we base our modeling on a Scania G480 construction truck shown in Figure 3.1. The vehicle's lateral dynamics are modeled as a 4-axles bicycle model with two steering axles. Also, the vehicle longitudinal dynamics are included in the model making use of an in-production simulation model of a cruise controller. We focus the lateral dynamics modeling since the longitudinal dynamics model is provided by Scania CV AB. On the other hand, the movement of a car-like nonholonomic vehicle, such as a truck that performs curvy paths at low speeds (i.e., when the lateral dynamics have little influence) and drives straight at high-speeds can be described by its kinematics without significant loss of prediction accuracy [63]. So, we use a kinematic vehicle model for control design.

The outline of this chapter is as follows. In Section 3.1, the simulation used in simulation is described, namely a 4-axles bicycle model. In Section 3.2 is presented the kinematic vehicle model used for control design. Section 3.3 provides a comparison between both models behavior when driving at different speeds and subject to different curvature requests. A summary of the presented models concludes this chapter.

Figure 3.1: Modified Scania G480 construction truck used as experimental and research platform.

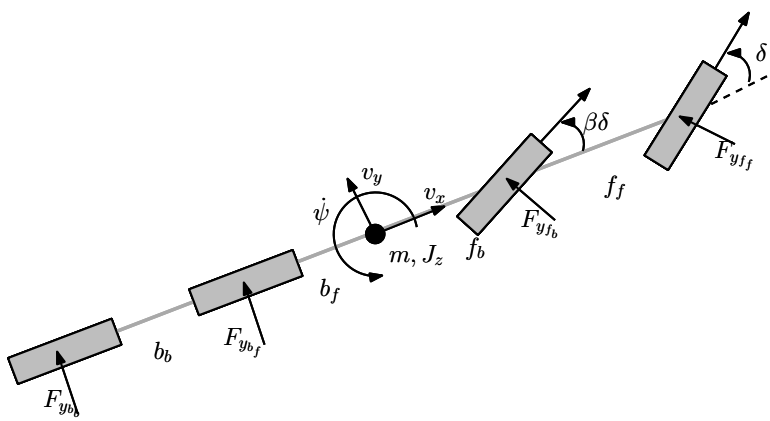


Figure 3.2: 4-axles bicycle model illustration.

3.1 Simulation model: 4-axles bicycle model

The lateral dynamics of the simulation model is described by a 4-axles nonlinear bicycle model with 2 steering axles in the front and 2 traction axles in the back. It is based on a modified Scania G480 construction truck shown in Figure 3.1.

The notation used in the 4-axles bicycle model is shown in Figure 3.2. The vehicle lateral dynamics are modeled by the following set of differential equations describing the lateral force and the momentum [101],

$$m(\dot{v}_y + \dot{\psi} v_x) = F_{y_{b_b}} + F_{y_{b_f}} + F_{y_{f_b}} \cos\delta_b + F_{y_{f_f}} \cos\delta_f \tag{3.1a}$$

$$J_z \ddot{\psi} = f_f F_{y_{f_f}} \cos\delta_f + f_b F_{y_{f_b}} \cos\delta_b - b_f F_{y_{b_f}} - b_b F_{y_{b_b}}, \tag{3.1b}$$

where v_x and v_y denote the longitudinal and the lateral speed of the vehicle, and $\dot{\psi}$ denotes the yaw rate. The constants m and J_z denote the vehicle mass and moment of inertia about the yaw axis, respectively, and $f_{b/f}$ and $b_{b/f}$ represent the distances from the center of gravity to the front and rear axles respectively. Finally, $\delta_{f/b}$ are the steering angles of the front axles. From Figure 3.2, $\delta = \delta_f$ and $\delta_b = \beta\delta_f$, where β is the wheel angle ratio between the two front axles. The estimation of β is made using real data containing the wheel angles from both front axles. Since the model focuses on the lateral dynamics, v_x is considered known. In this model, the side forces are assumed to be linear functions of the slip angles

$$F_{y_{f_f}} = -C_{f_f}\alpha_{f_f} \tag{3.2a}$$

$$F_{y_{f_b}} = -C_{f_b}\alpha_{f_b} \tag{3.2b}$$

$$F_{y_{b_f}} = -C_{b_f}\alpha_{b_f} \tag{3.2c}$$

$$F_{y_{b_b}} = -C_{b_b}\alpha_{b_b}, \tag{3.2d}$$

where the constants $C_{b_{f/b}}$ and $C_{f_{f/b}}$ are the cornering stiffnesses of the rear and the front axles respectively. The axles cornering stiffnesses have been computed by linearizing Pacejka's formula [101] for no longitudinal slip, no camber, small slip angles and using the steady-state vertical loads on each axle obtained through Adams simulation [102]. Adams is a simulation environment that allows building systems in detail, such as a vehicle, to the dynamics of moving parts and how loads and forces are distributed. The slip angles are given by

$$\alpha_{f_f} = \arctan\left(\frac{v_y + \dot{\psi} f_f}{|v_x|}\right) - \delta_f \tag{3.3a}$$

$$\alpha_{f_b} = \arctan\left(\frac{v_y + \dot{\psi} f_b}{|v_x|}\right) - \delta_b \tag{3.3b}$$

$$\alpha_{b_f} = \arctan\left(\frac{v_y - \dot{\psi} b_f}{|v_x|}\right) \tag{3.3c}$$

$$\alpha_{b_b} = \arctan\left(\frac{v_y - \dot{\psi} b_b}{|v_x|}\right). \tag{3.3d}$$

Also, we derive a formula that allows us to relate the vehicle curvature with its steering angle (input to the vehicle model (3.1)) assuming steady-state conditions

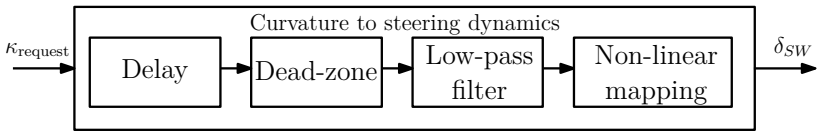


Figure 3.3: Curvature to steering wheel angle dynamics.

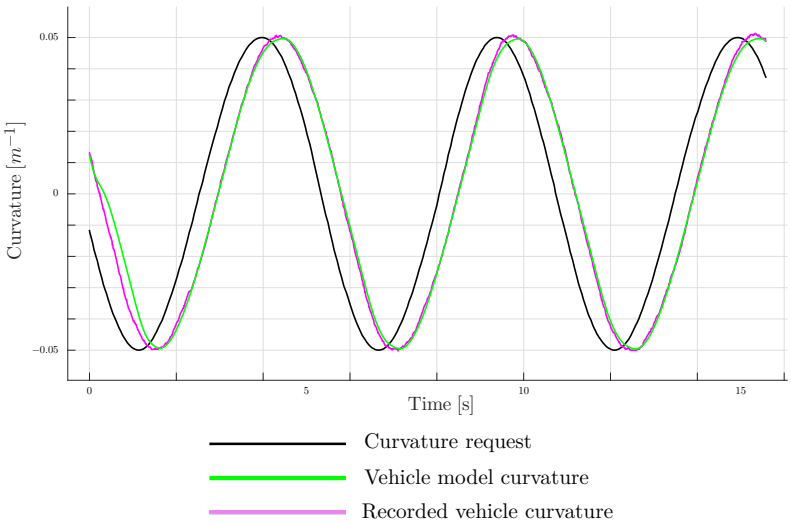
and that the vehicle has no slip (i.e., $v_x \gg v_y$). Vehicle curvature is defined as being the ratio between yaw rate and longitudinal speed at each instant. Equation (3.4) uses the curvature information to compute the steering angle that makes it dependent on the square of the vehicle velocity and on the stiffnesses of both rear and front wheels

$$\delta_f = \frac{(C_{f_f} + C_{f_b}) \arctan\left((D + K_{us} v_x^2)\kappa\right)}{C_{f_f} + C_{f_b}\beta}, \tag{3.4}$$

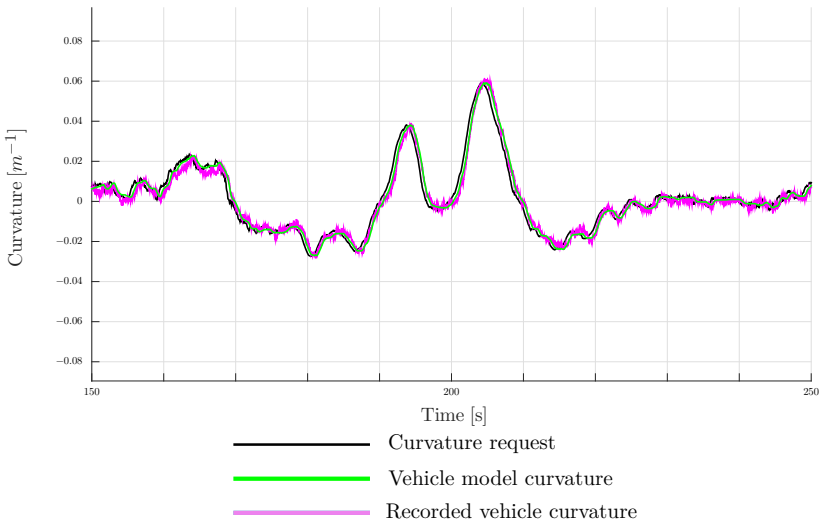
where K_{us} is the understeering gradient of the vehicle, which depends on the vehicle mass and on its center of mass, and D is the wheelbase of the truck.

The interface between the controller and the vehicle is a curvature request. Thus, an additional modeling step is necessary. The curvature to steering wheel angle dynamics are composed by a constant delay, a low-pass filter and a dead-zone, as shown in Figure 3.3. The vehicle model was validated using experimental data where the truck, shown in Figure 3.1, was subject to different curvature requests functions, such as sinusoids and square waves, at low speeds (between 5 and 10 m/s). Then, the vehicle model was subject to the same steering wheel angle and the same longitudinal speed as the recorded measurements. One of those experiments is depicted in Figure 3.4a. Moreover, Figure 3.4b shows the model accuracy even when subject to more dynamic curvature request profiles. It is clear that the vehicle model correctly captures the vehicle curvature dynamics.

The longitudinal dynamics model are described in two steps and it was developed by Scania CV AB. First, the longitudinal model receives a speed request or an acceleration request and outputs a engine torque or a brake demand. It consists of three controllers, namely a cruise controller that outputs engine torque, a downhill speed controller that outputs auxiliary (or retarder) torque, and an external brake interface that takes acceleration as input and outputs wheel brake torque. The cruise controller is a proportional-integral (PI) controller that takes into account the vehicle characteristics, such as mass and maximum engine torque, as well as the road profile, to produce a engine torque that makes the vehicle track the speed request. The parameters used in the simulation are the same as the ones commercially available in the truck cruise controller. Secondly, the powertrain illustrated in Figure 3.5 is modeled. The engine receives a torque request and determines the amount of diesel that needs to be combined with air in a high pressurised chamber.



(a) Comparison between the vehicle model and real data using as input a sinusoid wave with curvature amplitude of 0.05 m^{-1} and period of 6 seconds.



(b) Comparison between the vehicle model and real data in a real road with two sharp turns.

Figure 3.4: Two examples of recorded experiments with the truck where different curvature request functions were inputted.

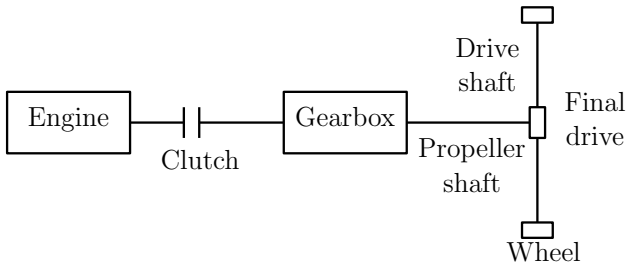


Figure 3.5: Illustration of the powertrain model.

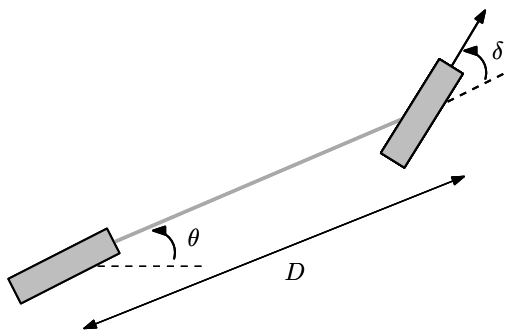


Figure 3.6: Kinematic model illustration.

The clutch is responsible for decoupling the engine from the drivetrain to enable gear shifts. The gear box connects the clutch to the propeller shaft and determines the conversion ratio between the engine torque and the propeller shaft torque. Finally, these torques are converted in the final drive and drive shafts to wheel torque. For a detailed description of each one of these elements of the powertrain, please refer to [103].

3.2 Control design model: car-like kinematic model

Figure 3.6 illustrates the kinematic vehicle model. The movement of a car-like nonholonomic vehicle at low speeds (i.e., when the lateral dynamics have little influence) can approximately be described by its time-domain kinematic equations [54] given by

$$\frac{dx(t)}{dt} = v(t)\cos(\theta(t)) \tag{3.5a}$$

$$\frac{dy(t)}{dt} = v(t)\sin(\theta(t)) \tag{3.5b}$$

$$\frac{d\theta(t)}{dt} = \frac{v(t)}{D}\tan(\delta(t)), \tag{3.5c}$$

where x and y are the coordinates of the vehicle in the global coordinate system, θ is the yaw angle, D is the distance between the front and rear axle, v is the longitudinal velocity in the vehicle coordinate system, and δ is the steering angle of the front wheels. We can translate these equations to space-domain where $v(t) \cdot \mathrm{d}t = \mathrm{d}s$ and assuming that $v(t) \neq 0$ and $v(t)$ is a continuous function, which yields

$$\frac{dx(s)}{ds} = \cos(\theta(s)) \tag{3.6a}$$

$$\frac{dy(s)}{ds} = \sin(\theta(s)) \tag{3.6b}$$

$$\frac{d\theta(s)}{ds} = \frac{1}{D}\tan(\delta(s)). \tag{3.6c}$$

3.3 Analysis

In this section, we compare the behavior of the dynamic and the kinematic model when subject to the same input.

Figure 3.7 shows simulation results using the dynamic and the kinematic models. There, the vehicle models are subject to different speeds and, at 15 seconds of simulation, a curvature request is given to both models. As expected, the dynamic and the kinematic models are similar at low speeds (5 m/s). The slip angles and the tire forces modeled in the dynamic model have neglectable influence on the vehicle motion at this speed. However, at high speeds (15 m/s), the behavior of the models is totally distinct. Therefore, at high speeds, it is not possible to accurately predict the motion of a realistic vehicle using a kinematic model. In this case, the dynamical model presents a considerable understeering behavior, while this is not modeled by the kinematic model. Finally, at moderate speeds (10 m/s), the behavior of both vehicle models is similar when the curvature is small. Nevertheless, when the curvature is considerable, the kinematic model is unable to replicate the lateral dynamics of the vehicle.

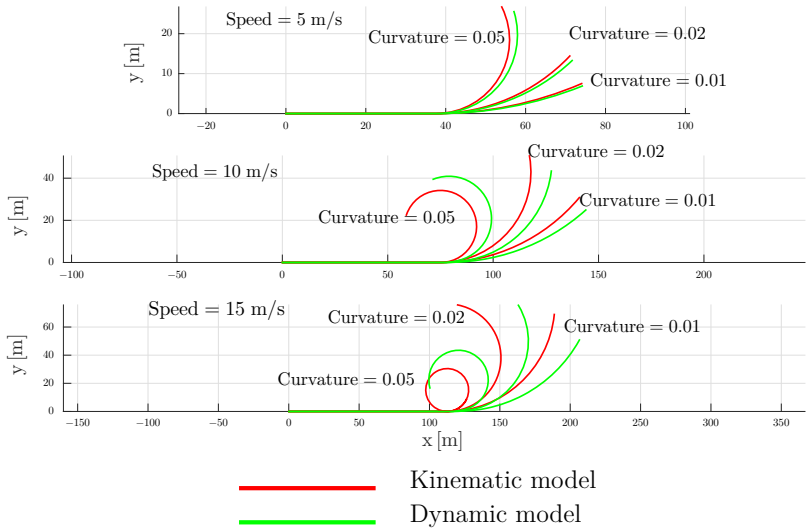


Figure 3.7: Comparison between the dynamic and the kinematic models for different velocities and curvatures.

In this thesis, we propose control design using the kinematic model. Therefore, we assume that the lateral dynamics of the vehicle are neglectable when driving straight or with small curvature. In addition, we enforce low speed when turning to mitigate the vehicle lateral slip. With these two conditions, the kinematic model is able to accurately predict the vehicle motion.

3.4 Summary

In this chapter, we described two types of vehicle models. A 4-axle bicycle model integrated with steering column dynamics was presented to describe the vehicle lateral dynamics. Furthermore, we compared the behavior of the vehicle model with real vehicle data where a truck was subject to different curvature requests and its movement recorded. We briefly detailed the longitudinal model, which contains cruise controller and vehicle longitudinal dynamics. This model was provided by Scania CV AB. In addition, we presented a kinematic car-like vehicle model, which is used as a model for control design in the next chapters. In the end, we compared the behavior of both models at different speeds and subject to different curvature requests. The kinematic model is able to reproduce accurately the dynamical vehicle model behavior at low speeds or with small curvatures. However, the models are totally distinct when the speed is too high due to the non-neglectable lateral dynamics is this case.

Chapter 4

Clothoid-based path sparsification

The need of a proper path representation is crucial for autonomous driving. Clothoids are used in road design due to their smoothness properties. In fact, the curvature of a clothoid varies linearly with the traveled distance. This provides a linear change in the vehicle steering angle when performing a turn and preventing sudden changes in lateral acceleration. Therefore, clothoid transition curves yield reduced values of lateral jerk. Thus, we describe a reference path by means of clothoids. Furthermore, we describe the path using the minimum number of clothoid segments. The problem of clothoid fitting has been a subject of research in recent years (e.g., [104, 105]). In [104], an algorithm is proposed for fitting a clothoid sequence with G^2 continuity (i.e., with continuous curvature) to a curve where a clothoid rational approximation is used. In [105], an algorithm is proposed for clothoid fitting between two given points using the Newton-Raphson method in a fast and robust fashion.

In this chapter, we formulate mathematically the problem of clothoid-based path sparsification in Section 4.1. We introduce the problem of path sparsification and detail how it can be solved using l_0-norm minimization techniques. In Section 4.2, we provide simulation and experimental results using the proposed algorithm. We conclude this chapter with a summary of its content.

4.1 Problem formulation

In many application, a reference path is a set of dense of linearly interpolated waypoints containing position, orientation and speed information. In this section, a strategy that allows sparsifying the path description with almost no loss of detail describing it with a set of clothoids is presented.

The problem of sparsification consists of describing an original path using a set of clothoids and is illustrated in Figure 4.1. The original path consists of a vector of N_o waypoints containing their Cartesian coordinates. Let $[\mathbf{x}_\mathrm{o}, \mathbf{y}_\mathrm{o}]^\intercal$ be a pair of vectors containing (x_i^o, y_i^o), $i = 1, ..., N_o$. The path described by these set of clothoids must

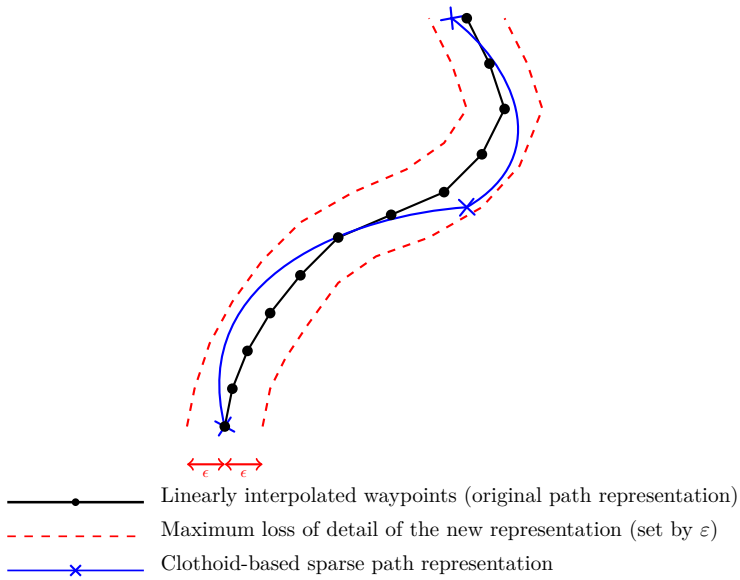


Figure 4.1: The reference path is a dense set of linearly interpolated waypoints, while the intended representation is a sparse set of clothoid segments that can deviate at most ε from the original path.

not deviate more than a certain small tolerance ε from the original path. Clothoids are uniquely described by their kink-points (i.e., the beginning and the end points of a clothoid). Let a clothoid-based path be described by $N_k \leq N_o$ kink-points, consisting of position, (x_j^k, y_j^k), orientation, θ_j^k, curvature, κ_j^k and traveled distance since the first kink-point, s_j^k, $j = 1, ..., N_k$. Furthermore, we intend to describe the original path with as few clothoid segments as possible.

We assume that the distances between the waypoints $\Delta s_i = s_{i+1} - s_i$ are known. The discretization of the clothoid parametric equations (2.1a) and (2.1b) is based on the fact that integrals can be approximated by Riemann sums. Thus, the position (x_n, y_n) with $n = 1, ..., N_o - 1$ after applying $\boldsymbol{\kappa}_n = [\kappa_0, ..., \kappa_n]^\intercal$ is described by

$$x_n = \sum_{i=0}^{n} \cos\left(\sum_{j=0}^{i} \kappa_j \Delta s_j\right) \Delta s_i \tag{4.1a}$$

$$y_n = \sum_{i=0}^{n} \sin\left(\sum_{j=0}^{i} \kappa_j \Delta s_j\right) \Delta s_i. \tag{4.1b}$$

The equivalent vectorial version for $\boldsymbol{\kappa} = [\kappa_0, ..., \kappa_{N_o}]^\intercal$ is

$$\mathbf{x}(\boldsymbol{\kappa}) = \mathbf{S}\cos(\mathbf{S}\kappa) \tag{4.2a}$$
$$\mathbf{y}(\boldsymbol{\kappa}) = \mathbf{S}\sin(\mathbf{S}\kappa), \tag{4.2b}$$

where $\mathbf{S} \in \mathbb{R}^{N_o \times N_o}$ is a lower triangular matrix that performs the cumulative sum operation along a vector

$$\mathbf{S} = \begin{bmatrix} \Delta s_0 & & & & \\ \Delta s_0 & \Delta s_1 & & & \\ \Delta s_0 & \Delta s_1 & \Delta s_2 & & \\ \vdots & \vdots & \vdots & \ddots & \\ \Delta s_0 & \Delta s_1 & \Delta s_2 & \dots & \Delta s_N \end{bmatrix}. \tag{4.3}$$

Note that the coordinates of the reconstructed path $[\mathbf{x}(\boldsymbol{\kappa}), \mathbf{y}(\boldsymbol{\kappa})]^\intercal$ are only dependent on the clothoid curvature κ assuming known initial orientation and curvature.

The problem of fitting a sparse piecewise linear function to data involves minimizing the l_0-norm. For a piecewise linear function, the second-order derivative of the curvature is exactly zero everywhere except in a few points, the so called kink-points. Minimizing the l_0-norm of the second-order differences of the curvature vector, is equivalent to minimizing the number of kink-points. Thus, we propose to solve the optimization problem

$$\min_{\boldsymbol{\kappa}} \quad ||\mathbf{D_2}\boldsymbol{\kappa}||_0 \tag{4.4a}$$
$$\text{s. t.} \quad |\mathbf{x}(\boldsymbol{\kappa}) - \mathbf{x}_\text{o}| \leq \varepsilon \tag{4.4b}$$
$$|\mathbf{y}(\boldsymbol{\kappa}) - \mathbf{y}_\text{o}| \leq \varepsilon, \tag{4.4c}$$

where $\boldsymbol{\kappa}$ is the vector of curvatures to be estimated, $\varepsilon \in \mathbb{R}^+$ is the maximum allowed error of the reconstructed curve, the inequalities (4.4b) and (4.4c) are performed elementwise, and $\mathbf{D}_2 \in \mathbb{Z}^{(N_o-2)\times N_o}$is the matrix operator that calculates second-order differences of a vector,

$$\mathbf{D_2} = \begin{bmatrix} 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & 1 \end{bmatrix}. \tag{4.5}$$

Note that the orientation of the reconstructed curve is given by $\boldsymbol{\theta}(\boldsymbol{\kappa}) = \mathbf{S}\boldsymbol{\kappa}$

The problem (4.4) is non-deterministic polynomial-time hard (NP-hard) [106] and therefore, difficult to solve due to its computational complexity. These type of problems are usually relaxed using the l_1-norm, which result in convex relaxations of the original problem. As proposed in [107], a method to solve these problems

consists of applying an algorithm from the family of majorization-minimization algorithms, where the minimization of a generic function (l_0-norm) is performed by iteratively minimizing a convex majorizer function (l_1-norm). As discussed in [107], there are several properties of this algorithm that need more research and understanding. Here, the algorithm is analyzed from a practical point of view and we do not claim any specific property, such as convergence or optimality. Thus, we have to rewrite the problem (4.4) as

$$\begin{aligned} \min_{\boldsymbol{\kappa}} \quad & ||\mathbf{W} \odot \mathbf{D_2}\boldsymbol{\kappa}||_1 \\ \text{s.t.} \quad & |\mathbf{x}(\boldsymbol{\kappa}) - \mathbf{x}_\text{o}| \leq \varepsilon \\ & |\mathbf{y}(\boldsymbol{\kappa}) - \mathbf{y}_\text{o}| \leq \varepsilon, \end{aligned} \tag{4.6}$$

where $\mathbf{W}$ is a vector that contains weights that are initially set to 1 and $\odot$ is the Hadamard product. $\mathbf{W}$ is updated in each iteration as

$$\mathbf{W}_{m+1} = \frac{1}{|\mathbf{W}_m \odot \mathbf{D}_2\boldsymbol{\kappa}|}. \tag{4.7}$$

At each iteration, the weighting vector is normalized so that the weights sum up to the number of samples. In other words, the weights are updated such that the entries of the smoothness term close to zero receive a higher weight, while the others obtain a smaller weight resulting in a reweighed l_1-norm that is more similar to the l_0-norm.

However, problem (4.6) is not in convex form due to the fact that the reconstructed path $[\mathbf{x}(\boldsymbol{\kappa}), \mathbf{y}(\boldsymbol{\kappa})]^\intercal$ is computed using (4.2) that are clearly nonlinear equations. A linear approximation is done using a first-order Taylor approximation of both cosine and sine around an estimated curvature $\hat{\boldsymbol{\kappa}}$. To be able to apply this approximation, we have to estimate the original path curvature and discretize its function in Δs_i. The linearization takes the form

$$\mathbf{x}(\kappa) = \mathbf{S}\left(\cos(\mathbf{S}\hat{\boldsymbol{\kappa}}) - \sin(\mathbf{S}\hat{\boldsymbol{\kappa}})\mathbf{S}(\boldsymbol{\kappa} - \hat{\boldsymbol{\kappa}})\right) \tag{4.8a}$$

$$\mathbf{y}(\kappa) = \mathbf{S}\left(\sin(\mathbf{S}\hat{\boldsymbol{\kappa}}) + \cos(\mathbf{S}\hat{\boldsymbol{\kappa}})\mathbf{S}(\boldsymbol{\kappa} - \hat{\boldsymbol{\kappa}})\right), \tag{4.8b}$$

where estimated curvature $\hat{\boldsymbol{\kappa}}$ is found by approximating a circle passing by three consecutive points p_{k-1}, p_k and p_{k+1} of the raw data, as illustrated in Figure 4.2. Then the inverse of the circle radius is the curvature approximation. Thus, as described in [108], the discrete curvature approximation is given by

$$\hat{\kappa} = \frac{4A_\Delta}{P_\Delta}, \tag{4.9}$$

where A_Δ and P_Δ are the area and perimeter of the triangle formed by the considered three points, respectively.

Finally, we formulate the original problem (4.4) as a convex problem

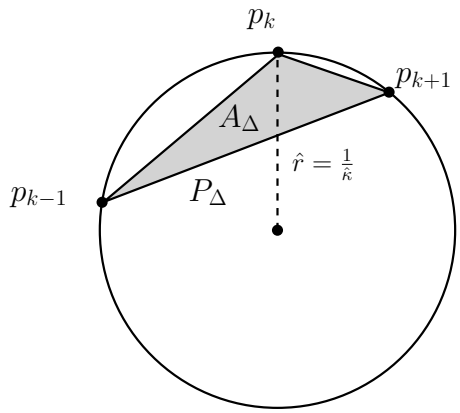


Figure 4.2: Curvature estimation $\hat{\kappa}$ by circle approximation where p_{k-1}, p_k and p_{k+1} are three consecutive points.

$$\min_{\kappa} \quad ||\mathbf{W} \odot \mathbf{D_2}\boldsymbol{\kappa}||_1 \tag{4.10a}$$

$$\text{s.t.} \quad |\mathbf{S}\left(\cos(\mathbf{S}\hat{\boldsymbol{\kappa}}) - \sin(\mathbf{S}\hat{\boldsymbol{\kappa}})\mathbf{S}(\boldsymbol{\kappa} - \hat{\boldsymbol{\kappa}})\right) - \mathbf{x}_{\text{raw}}| \leq \varepsilon \tag{4.10b}$$

$$|\mathbf{S}\left(\sin(\mathbf{S}\hat{\boldsymbol{\kappa}}) + \cos(\mathbf{S}\hat{\boldsymbol{\kappa}})\mathbf{S}(\boldsymbol{\kappa} - \hat{\boldsymbol{\kappa}})\right) - \mathbf{y}_{\text{raw}}| \leq \varepsilon. \tag{4.10c}$$

Let $\boldsymbol{\rho} = ||\mathbf{W} \odot \mathbf{D_2}\boldsymbol{\kappa}^\star||_1$, where $\boldsymbol{\rho} = [\rho_i, ..., \rho_{N_o}]^\intercal$. The kink-points are the elements of the optimal curvature vector $\boldsymbol{\kappa}^\star$ where the smoothness term $\boldsymbol{\rho}$ is bigger than a small threshold δ. These are the elements with index j that belong to

$$\{j \in \{2, .., N_o - 1\} : \rho_j > \delta\} \cup \{1, N_o\}.$$

Figure 4.3 illustrates the idea of choosing the kink-points only where the second-order differences of the curvature function is bigger than a specified threshold.

The path reconstruction is possible to do from the sparsification information. The path reconstruction is accomplished by using only the kink-points and their underlying information in an efficient and computational inexpensive fashion. A pair of kink-points uniquely defines a clothoid since it is possible to compute the clothoid arc-length, $l_j^k = s_{j+1}^k - s_j^k$, and the curvature change rate, $c_j^k = \frac{\kappa_{j+1}^k - \kappa_j^k}{l_j^k}$. In order to avoid numeric integration as in (4.2) we use the rational approximation given in [89] to approximate the Fresnel integrals in (2.1a) and (2.1b) when reconstructing the curve $(x(s), y(s))$. The rational approximation is given by

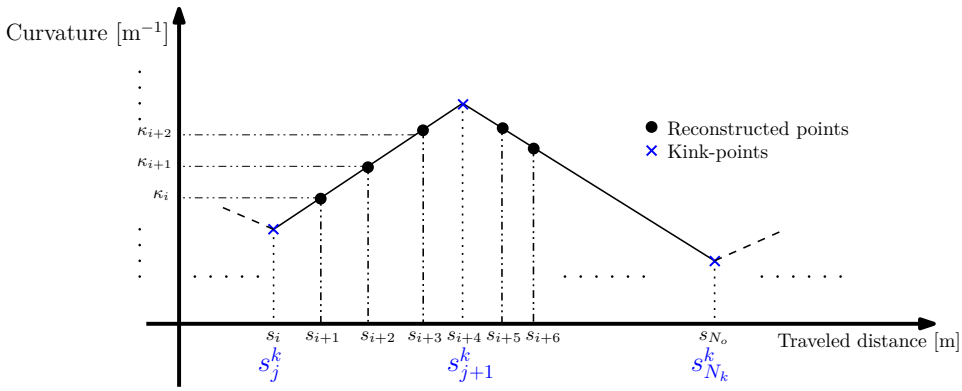


Figure 4.3: Illustration of the reconstructed path curvature function $\boldsymbol{\kappa}$. The points where the second-order differences of $\boldsymbol{\kappa}$ are bigger than a certain threshold are kink-points. These are points where two linear functions with different slopes meet and represent the beginning or end of a clothoid.

$$x(s) \approx \frac{1}{2} - R(s)\sin(\frac{1}{2}\pi(A(s) - s^2)), \tag{4.11a}$$

$$y(s) \approx \frac{1}{2} - R(s)\cos(\frac{1}{2}\pi(A(s) - s^2)), \tag{4.11b}$$

where

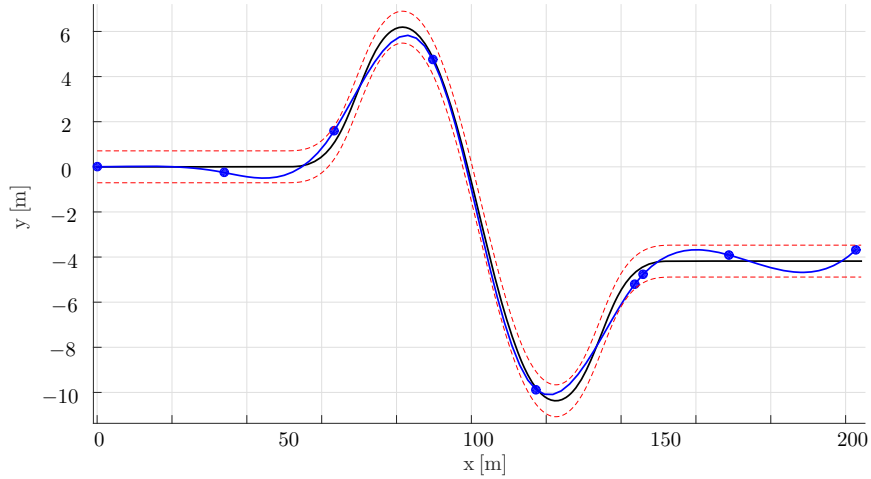
$$R(s) = \frac{0.506s + 1}{1.79s^2 + 2.054s + 1.414}, \tag{4.12a}$$

$$A(s) = \frac{1}{0.803s^3 + 1.886s^2 + 2.524s + 2}. \tag{4.12b}$$

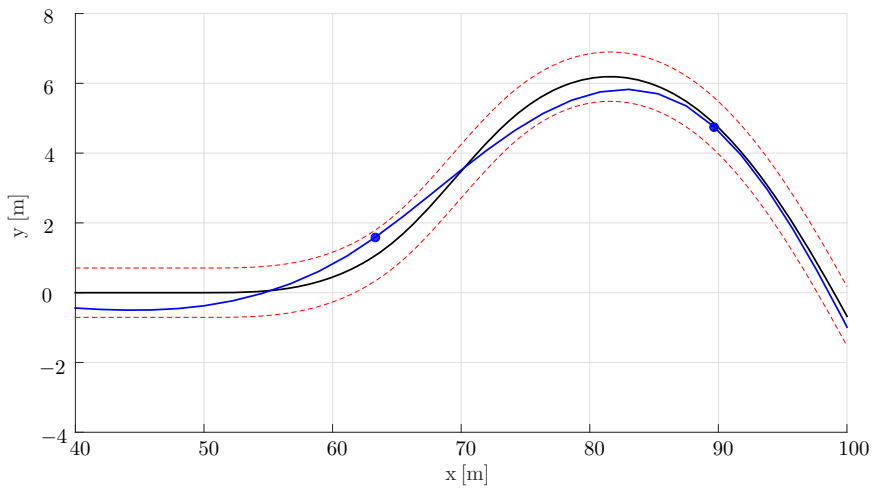
The maximum error of this approximation is 1.7×10^{-3} [89].

4.2 Simulation examples and discussion

In this section, we present the results obtained with the clothoid-based path sparsification algorithm described in Section 4.1. We start by showing an illustrative example that exemplifies the usefulness of the method. This illustrative example is based on a double S-curve, which is generated from nine clothoids connected together. Then, we apply the algorithm to two real paths: a precision test track and a high-speed test track. The paths were recorded using a real-time kinematic (RTK) GPS installed in a truck. The data is from Scania's test track located in Södertälje, Sweden, south of Stockholm.



(a) Clothoid-based path sparsification algorithm output with $\varepsilon = 0.5$.



(b) Zoomed section of the clothoid-based path sparsification algorithm output with $\varepsilon = 0.5$.

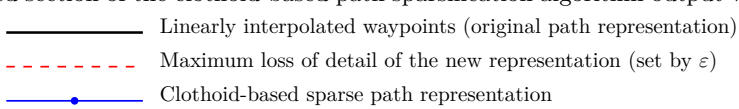


Figure 4.4: The reference path is described with a clothoid-based path using only a few kink-points.

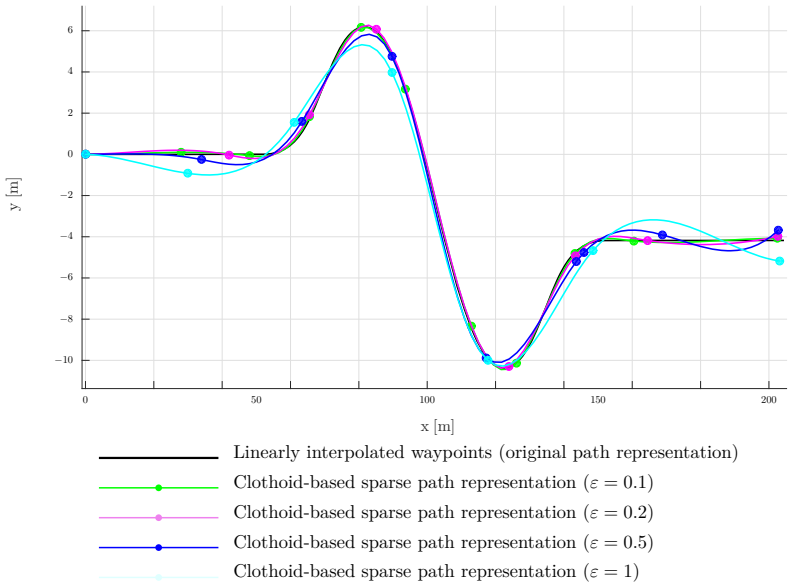


Figure 4.5: Illustrative example that demonstrates the effect of ε in the result of the clothoid-based path sparsification algorithm.

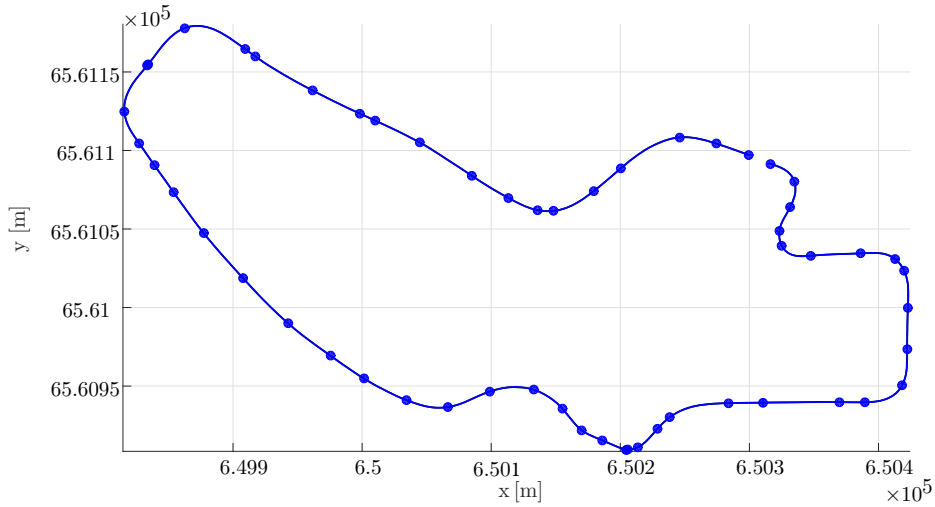
Table 4.1: Reference path information for $\varepsilon = 0.1$

Path	Precision track	High-speed track
Number of points of the original path	1459	3024
Number of points of the sparse path	55	59

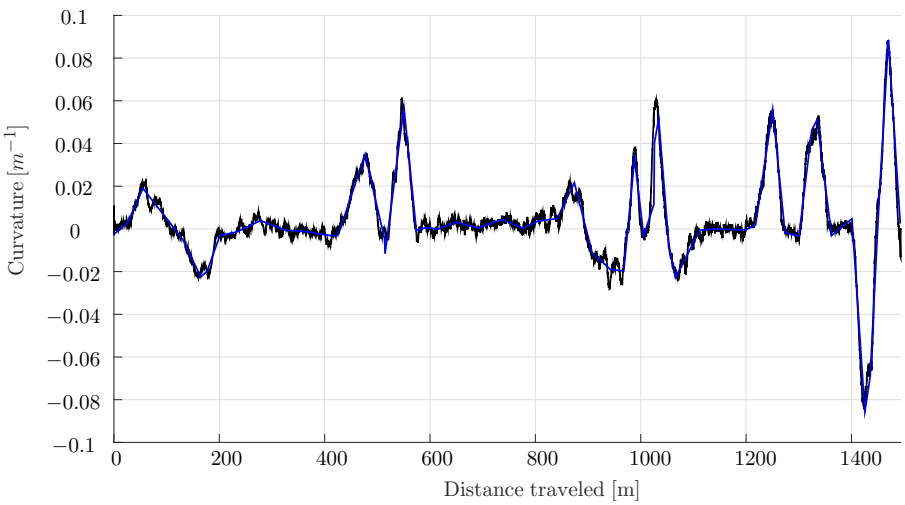
Figure 4.4 depicts an illustration of the clothoid-based path algorithm output. We see that the original path is reconstructed with only few points where the reconstructed path does not violate the imposed ε constraint. Furthermore, Figure 4.5 illustrates the influence that the single parameter ε has on the solution retrieved by the algorithm. As expected, the bigger the constraints the more the reconstructed path deviates from the original and fewer kink-points are needed to reconstruct the path.

We show the obtained paths when running the clothoid-based path sparsification algorithm in Figures 4.6 and 4.7 with $\varepsilon = 0.1$ m. Once again, the algorithm demonstrates its ability to represent the path with few kink-points.

Information, such as the number of points of the original and the sparsified paths, is summarized in Table 4.1. While the compression ratio between the sparsified path and the original path is about 2%-3% the sparsified path has almost no loss of detail since the maximum deviation allowed is 0.1 m.



(a) The clothoid-based path sparsification algorithm applied on the precision track path.



(b) Curvature of the original path is fitted with a piecewise linear curvature function to be able to describe the original path with a set of clothoids.

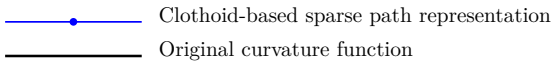


Figure 4.6: The precision track path is described with a clothoid-based path using only a few kink-points.

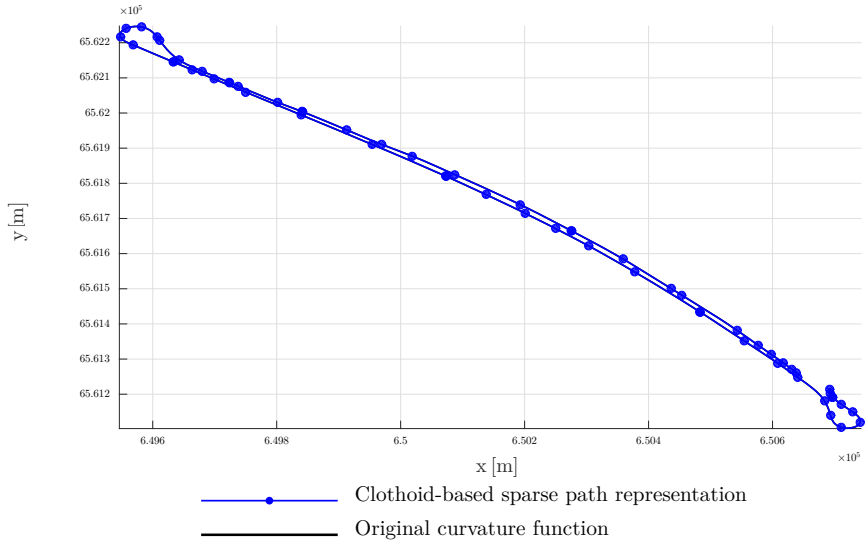


Figure 4.7: Clothoid-based path sparsification is used on the high-speed test track path.

4.3 Summary

In this chapter, a clothoid-based path sparsification algorithm was presented. We formulated the problem of path sparsification as a l_0-norm curvature regularization problem (i.e., an optimization problem that consists of finding a piecewise linear curvature function with the smallest number of kink-points allowing us to reconstruct the path with almost no loss of detail). The property of having linear varying curvature with the traveled distance is present in clothoids. The l_0-norm induces sparsity in the solution, since we want the cost function (i.e., the second derivative of the curvature function) to always be zero except at a few points, the so called kink-points. However, optimization problems using the l_0-norm are computationally intractable since they are NP-hard and therefore, we solved the problem using a majorization-minimization algorithm where we solve iteratively a weighted l_1-norm optimization problem. In the end of the chapter, we presented results using the clothoid-based path sparsification algorithm. The results showed that it is possible to represent a path with few kink-points with a compression ratio of about 2%-3%.

Chapter 5

Clothoid-based model predictive controller

In this chapter, we approach the problem of path following by merging the MPC framework with the concept of clothoids: we introduce a clothoid-based model predictive controller (MPCC). While predictive control performance is dependent on the model accuracy, its performance is also highly affected by the model complexity. Although a complex model can be really accurate, it can also endanger the real-time applicability due to limited computational power and real time behavior. Therefore, we propose that the vehicle motion is predicted using the geometric model of a clothoid instead of a vehicle model. Even though the kinematic vehicle model and the clothoid model are essentially the same, we constrain the former to have a piecewise linearly varying curvature such that the vehicle is predicted as a set of clothoids. In fact, instead of using the steering wheel angle as the model input, the MPC optimizes over the curvature change rate, how much the vehicle curves and for how long. Then, the MPC is formulated in the standard form as a tracking MPC.

We address the problem of clothoid-based path following using an MPC framework by developing an linear-time varying (LTV) MPC for the lateral control of a vehicle in Section 5.1. Section 5.2 discusses the accuracy of the clothoid model as a prediction model and demonstrates the performance of the proposed controller through illustrative simulations. In the end, Section 5.3 highlights the main topics addressed in the chapter.

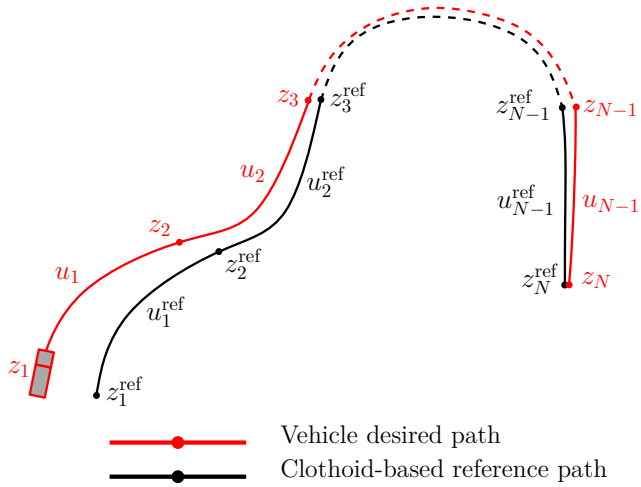


Figure 5.1: The path is represented by $N-1$ clothoids that are described by N kink-points (i.e., the beginning or end of a clothoid segment). The clothoid-based MPC controller results in a sequence of inputs that guide the vehicle along the reference path.

5.1 Problem formulation

In this section, we propose a method for lateral control of an autonomous vehicle by formulating the MPCC.

The movement of a car-like nonholonomic vehicle at low speeds (i.e., when the lateral dynamics have little influence), can approximately be described by its space-domain kinematic equations

$$\frac{dx(s)}{ds} = \cos(\theta(s)), \tag{5.1a}$$

$$\frac{dy(s)}{ds} = \sin(\theta(s)), \tag{5.1b}$$

$$\frac{d\theta(s)}{ds} = \kappa(s), \tag{5.1c}$$

where $\kappa(s)$ represents the vehicle curvature along the path. If we limit $\kappa(s)$ to be a linearly varying function given by

$$\kappa(s) = cs + \kappa(0),$$

where c is the curvature change rate, then (5.1) describes a clothoid as in (2.1).

Let the reference path be generated by the clothoid-based path sparsification algorithm presented in Chapter 4. Thus, the reference path is represented by $N-1$

clothoid segments that are described by N kink-points defined by $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}, \theta_i^{\text{ref}}, \kappa_i^{\text{ref}}]^\intercal$, where $i = 1, ..., N$. For each kink-point $\mathbf{z}_i^{\text{ref}}$, $[x_i^{\text{ref}}, y_i^{\text{ref}}]^\intercal$ represents the position, θ_i^{ref} represents the orientation, and κ_i^{ref} represents the curvature. Also, we have access to the clothoid model input that generated the path $\mathbf{u}_i^{\text{ref}} = [c_i^{\text{ref}}, l_i^{\text{ref}}]^\intercal$, where $i = 1, ..., N-1$, which represent the curvature change rate (constant by definition), c_i^{ref}, and arc-length, l_i^{ref} of the clothoid delimited by the kink-points i and $i+1$.

The vehicle state at each kink-point is similarly defined as $\mathbf{z}(s_i) = \mathbf{z}_i = [x_i, y_i, \theta_i, \kappa_i]^\intercal$, where $i = 1, ..., N$ and its input as $\mathbf{u}(s_i) = \mathbf{u}_i = [c_i, l_i]^\intercal$, where $i = 1, ..., N-1$. Here, we use the subscript i intentionally since we sample the predicted vehicle motion only at the kink-points of the reference path as idealized in Figure 5.1.

Using (5.1) and assuming $\kappa(s)$ to be a piecewise linearly varying function, the vehicle state evolution at each kink-point is computed using the discrete function

$$\mathbf{z}_{i+1} = f(\mathbf{z}_i, \mathbf{u}_i),$$

which is defined as

$$x_{i+1} = x_i + \int_0^{l_i} \cos(\theta_i + \kappa_i s + c_i \frac{s}{2}^2) \, \mathrm{d}s, \tag{5.2a}$$

$$y_{i+1} = y_i + \int_0^{l_i} \sin(\theta_i + \kappa_i s + c_i \frac{s}{2}^2) \, \mathrm{d}s, \tag{5.2b}$$

$$\theta_{i+1} = \theta_i + \kappa_i l_i + c_i \frac{l_i}{2}^2, \tag{5.2c}$$

$$\kappa_{i+1} = \kappa_i + c_i l_i. \tag{5.2d}$$

We are interested in the problem of following a reference path with an autonomous vehicle only using the kink-points of the path. We can formulate this problem as a standard tracking MPC problem

$$\min_{\tilde{\mathbf{u}}_1, ..., \tilde{\mathbf{u}}_H} \quad \sum_{i=1}^{H} \tilde{\mathbf{z}}_i^\intercal \mathbf{Q} \tilde{\mathbf{z}}_i + \tilde{\mathbf{u}}_{i-1}^\intercal \mathbf{R} \tilde{\mathbf{u}}_{i-1} \tag{5.3a}$$

$$\text{s.t.} \quad \mathbf{z}_{i+1} = f(\mathbf{z}_i, \mathbf{u_i}), \quad i = 1, ..., H-1, \tag{5.3b}$$

$$\mathbf{z}_0 = \mathbf{z}(0), \tag{5.3c}$$

$$\mathbf{u_i} \in \mathbb{U}, \quad i = 1, ..., H-1 \tag{5.3d}$$

where $H \in \mathbb{Z}^+$ is the MPC prediction horizon and $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{z}_i^{\text{ref}}$, $\tilde{\mathbf{u}}_i = \mathbf{u}_i - \mathbf{u}_i^{\text{ref}}$ represent the state and input tracking errors, respectively. The penalization matrices $\mathbf{Q} \in \mathbb{R}^{4\times4}$ and $\mathbf{R} \in \mathbb{R}^{2\times2}$ are diagonal positive definite and weigh the state and the input cost, respectively. The vehicle motion prediction, represented in (5.3b)

by function f, is done using (5.2) and (5.3d) describes the input constraints. Intuitively, we want to predict the set of clothoids that guide the vehicle to the desired path while minimizing the state error (weighted difference in position, orientation, and curvature) at the kink-points and the input error (weighted difference in curvature slope and arc-length).

The integrals in (5.2a) and (5.2b) are known as the Fresnel integrals and do not have a closed form expression [89]. Since we want to be able to compute the vehicle motion path kink-points explicitly, a good approximation is to substitute the integral by its corresponding arc length as in a Riemann integral [109],

$$x_{i+1} = x_i + l_{i+1}\cos(\theta_i + \kappa_i l_i + c_i \frac{l_i}{2}^2) \tag{5.4a}$$

$$y_{i+1} = y_i + l_i \sin(\theta_i + \kappa_i l_i + c_i \frac{l_i}{2}^2) \tag{5.4b}$$

$$\theta_{i+1} = \theta_i + \kappa_i l_i + c_i \frac{l_i}{2}^2 \tag{5.4c}$$

$$\kappa_{i+1} = \kappa_i + c_i l_i, \tag{5.4d}$$

and then to linearize the model around the reference path

$$\tilde{\mathbf{z}}_{i+1} = \mathbf{A}_i \tilde{\mathbf{z}}_i + \mathbf{B}_i \tilde{\mathbf{u}}_i, \tag{5.5}$$

where

$$\mathbf{A}_i = \frac{\partial f(\mathbf{z},\mathbf{u})}{\partial \mathbf{z}}\bigg|_{\substack{\mathbf{z}=\mathbf{z_i}^{\text{ref}},\\ \mathbf{u}=\mathbf{u_i}^{\text{ref}}}} = \begin{bmatrix} 1 & 0 & -l_i^{\text{ref}}\sin\theta_{i+1}^{\text{ref}} & -{l_i^{\text{ref}}}^2\sin\theta_{i+1}^{\text{ref}} \\ 0 & 1 & l_i^{\text{ref}}\cos\theta_{i+1}^{\text{ref}} & -{l_i^{\text{ref}}}^2\cos\theta_{i+1}^{\text{ref}} \\ 0 & 0 & 1 & l_i^{\text{ref}} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.6}$$

and

$$\mathbf{B}_i = \frac{\partial f(\mathbf{z},\mathbf{u})}{\partial \mathbf{u}}\bigg|_{\substack{\mathbf{z_i}=\mathbf{z}^{\text{ref}},\\ \mathbf{u}=\mathbf{u_i}^{\text{ref}}}} = \begin{bmatrix} -\frac{{l_i^{\text{ref}}}^3}{2}\sin\theta_{i+1}^{\text{ref}} & \cos\theta_{i+1}^{\text{ref}} - l_i^{\text{ref}}\kappa_{i+1}^{\text{ref}}\sin\theta_{i+1}^{\text{ref}} \\ \frac{{l_i^{\text{ref}}}^3}{2}\cos\theta_{i+1}^{\text{ref}} & \sin\theta_{i+1}^{\text{ref}} + l_i^{\text{ref}}\kappa_{i+1}^{\text{ref}}\cos\theta_{i+1}^{\text{ref}} \\ \frac{{l_i^{\text{ref}}}^2}{2} & \kappa_{i+1}^{\text{ref}} \\ l_i^{\text{ref}} & c_i^{\text{ref}}. \end{bmatrix} \tag{5.7}$$

In Appendix A, the final formulation of the controller is cast as a quadratic program (QP), for which there are efficient methods to solve the optimization problem (e.g., *cvxgen* [110]).

Table 5.1: MPCC parameters tuning.

Parameters	MPCC
Prediction horizon, H	10
Controller frequency	50 Hz
State penalization matrix, $\mathbf{Q}$	$\text{diag}(1, 1, 10, 10)$
Input penalization matrix, $\mathbf{R}$	$\text{diag}(100, 1000)$

5.2 Simulation example and discussion

In this section, we present simulation results obtained with the MPCC controller introduced in Section 5.1. We use the same illustrative example as the one used in Section 4.2, and shown in Figure 5.2a. The reference path is a double S-curve that consists of nine clothoids connected together.

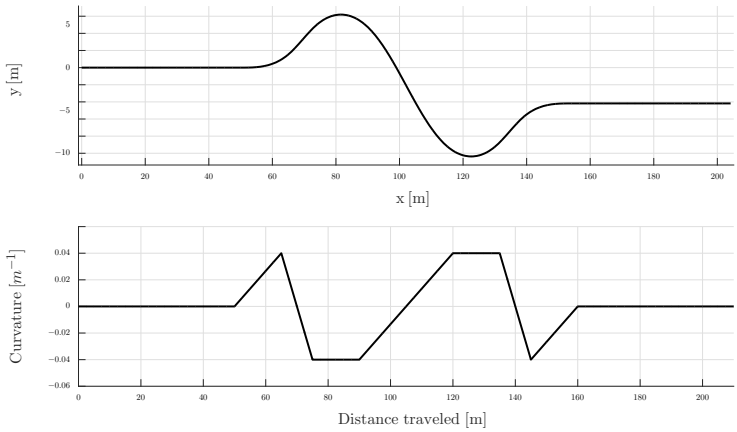
When the clothoid prediction model linearization is performed, we approximate the Fresnel integrals, which do not have closed form solution, by the Riemann integral approximation (5.4). In this case, the the upper bound of the approximation error depends on the clothoid arc-length L_i and is given by

$$e_{\text{approx}} = \left| \int_0^{L_i} f(s)\mathrm{d}s - \hat{f}(s) \right| < \frac{L_i^2}{2n} \max_{[0,L_i]} \left| \frac{\mathrm{d}f(s)}{\mathrm{d}s} \right|. \tag{5.8}$$

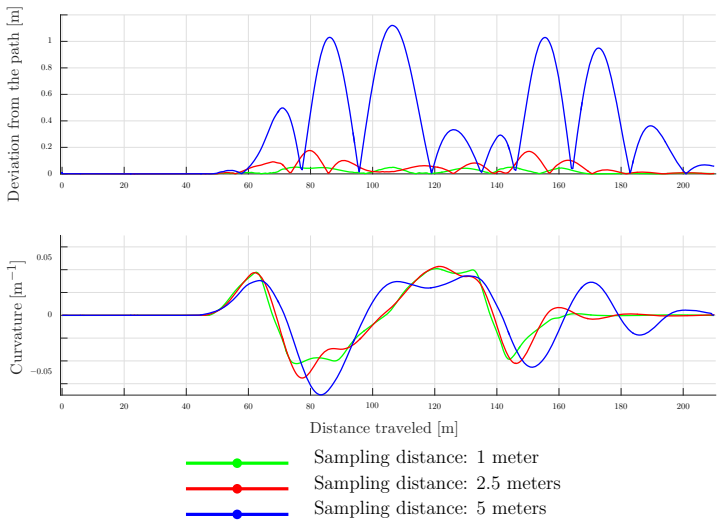
where $\hat{f}(s)$ is the Riemann integral approximation of the integral of f from 0 to L_i with respect to s and n is the number of subintervals in $[0, L_i]$. It is clear from that when $n \to \infty$ then $e_{\text{approx}} \to 0$ (i.e., the more samples we take the less erroneous the approximation is). As before, L_i represent a clothoid arc-length (i.e., the arc-length between two consecutive kink-points). Note that $L_i = nl_i$, where l_i in (5.2) is the arc-length of smaller clothoids consequence of dividing the original clothoid in n subintervals. Also, f is either a cosine or a sine, and therefore, the term $\max_{[0,L_i]} |\frac{\mathrm{d}f(s)}{\mathrm{d}s}|$ is 1 in every possible interval. Therefore, it is not reasonable to use waypoints too far apart, otherwise the prediction model can be inaccurate. Consider the case where the Riemann integral is calculated over 20 m, then the error upper-bound is 200 m. However, if we divide the interval in 10 smaller intervals of 2 m each the error-bound is reduced 100 times to 2 m.

We use three different sampling distances in the evaluation of the controller: 1, 2.5 and 5 m. Each clothoid is cut into smaller clothoids using the sampling distances. The results are shown in Figure 5.2b. The vehicle drives at constant speed of 5 m/s. Note that, although this might be a fairly low speed, the reference path includes two sharp turns. The simulation uses the vehicle dynamics described in Chapter 3. All the optimization problems are solved using *cvxgen*, which is a custom optimization solver [110].

Table 5.1 presents the MPCC parameter tuning used in simulation. The $\mathbf{Q}$ trades off position accuracy for driving smoothness, while $\mathbf{R}$ strongly penalizes deviations from the reference input. Controller aggressiveness is not desired if the



(a) Reference path used to illustrate the MPCC performance. In the figure, the reference path is represented on the upper plot, while the reference curvature is represented in the lower plot. The reference is a double S-curve that consists of nine clothoid segments.



(b) Illustrative example of the MPCC performance. The controller is tested in a clothoid-based reference path, where each clothoid is oversampled with different sampling distances. Above the vehicle deviation from the path is depicted, while below the controllers curvature requests are shown.

Figure 5.2: Double S-curve and MPCC performance.

deviation from the path is small and therefore, we desire to maintain low errors on heading and curvature as well. We also penalize deviations from the input, specially we do not intend to predict the vehicle motion with significantly different arc-lengths.

We can see, as expected from the reasoning about the approximation error, that the performance of the MPCC gets worse as the sampling distance increases. Therefore, if the sampling is short enough, the clothoid model used for the vehicle prediction has good accuracy allowing the performance of the controller to achieve good results. For instance the average deviation from the path is less than 0.02 m for both the 1 m and 2.5 m sampling. Furthermore, the maximum deviation is less than 0.2 m for these cases also.

In Chapter 8, we produce a more complete demonstration of the MPCC performance. There we compare the controller with standard approaches, such as the pure-pursuit controller and a standard tracking MPC approach. We also compare the MPCC with the controller described in Chapter 6. The comparison is done in simulation but still using the accurate dynamic vehicle model and real reference paths both with high and low speeds.

5.3 Summary

In this chapter, the problem of autonomous vehicle path following was addressed. We proposed a new approach to solve the problem by combining the MPC framework with clothoidal vehicle prediction. Therefore, a clothoid-based model predictive controller (MPCC) was introduced. The controller formulation is identical to a standard tracking MPC formulation, though the main novelty is the vehicle prediction being made using the clothoid model instead of a vehicle model. Typically, the model input to control the lateral position is the steering angle. In this case, we consider two inputs, namely the curvature change rate and the arc-length. The first one defines how sharply the vehicle curves, while the latter defines the length of the prediction until the next kink-point. We provided results showing the controller performance with an illustrative example. The reference path is a double S-curve that consists of several clothoids interconnected. The issue of approximating the clothoid parametric equations using Riemann integrals was discussed and the controller was evaluated using reasonable sampling distances due to the inaccuracy of the prediction for long distances. The controller was able to follow a clothoid-based reference path in a good fashion with deviations below 0.2 m in all cases.

Chapter 6

Model predictive controller for smooth driving

In autonomous driving, combining economic performance, such as comfort, with accurate path tracking accuracy is a challenging task due to their contradictory objectives. To overcome this, we propose an economic model predictive controller (EMPC) for autonomous driving. In standard tracking MPC approaches, the cost function of the optimization problem is zero at the optimal reference trajectory. However, that is not the case in the EMPC due to the inclusion of an economic cost associated with the system operation [98]. In our case, the economic cost imposes smooth and comfortable driving. We explicitly include the vehicle curvature in the cost function to influence its shape and characteristics and, to enforce convergence to the reference path, we also include the vehicle distance from the path. Once again, clothoid properties have a crucial role in the controller formulation. The economic cost introduced in the objective function leads to a quasi-clothoid driving since we minimize the first and second derivatives of the curvature function (i.e., we encourage linear curvature profiles). The underlying nonlinearities in the vehicle model lead to a linear time-varying (LTV) MPC formulation. In this chapter, we analyze some of the controller properties through illustrative examples that demonstrate the influence of the tuning parameters. Furthermore, we illustrate the controller performance with a simulation in a simple example.

This chapter is structured as follows. In Section 6.1, we address the problem of path following using a receding-horizon framework by developing an EMPC controller for an autonomous vehicle. In Section 6.2, we analyze the influence of the controller tuning parameters and evaluate the performance of the controller in simulation. Finally, Section 6.3 summarizes the chapter.

6.1 Problem formulation

In this section, we address the problem of lateral control of an autonomous vehicle. We formulate an EMPC in order to combine path tracking accuracy and driving smoothness.

Let a reference path be constituted by N waypoints of the form $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}, \theta_i^{\text{ref}}, s_i]^\intercal$, $i = 1, ..., N$ representing the path Cartesian coordinates, the orientation and the distance traveled along the path since the first point, respectively.

Let the economic cost of the EMPC be interpreted as the smoothness (or the comfort) of the vehicle motion and its predictions. To achieve that, the economic cost is designed in order to penalize the first and the second-order derivative of the curvature function. The first-order derivative is related to the vehicle lateral jerk, for which high values are perceived by a human as uncomfortable. On the other hand, the second derivative of the curvature is related to the angular acceleration of the steering wheel. Therefore, it is desired to maintain both the first and second derivatives as low as possible while tracking the reference path in order to achieve a smooth driving. For instance, a truck designed for mining applications weights around 100 tons when loaded. As a consequence, a high curvature request change rate can seriously damage the tires and the vehicle's steering servo. The other goal of the EMPC is to accurately follow a given reference path. Therefore, soft constraints are added to the problem in order to avoid large deviations from the path. Moreover, the vehicle maximum curvature and change rate are also limited.

As described in Chapter 4 for the clothoid-based path sparsification algorithm, we discretize the vehicle model (3.6) by letting the vehicle state at each sampling point be defined as $\mathbf{z}(s_i) = \mathbf{z}_i = [x_i, y_i, \theta_i]^\intercal$ and the vehicle input be $\kappa(s_i) = \kappa_i$. The vehicle state is sampled every $\Delta s_i = s_{i+1} - s_i$, which is the traveled distance between two consecutive points. The curvature is assumed to be constant between $\kappa(s_i)$ and $\kappa(s_{i+1})$. The integral discretization of (3.6a) and (3.6b) is based on the fact that integrals can be approximated by Riemann sums [109]. Thus, the position $[x(\boldsymbol{\kappa}_n), y(\boldsymbol{\kappa}_n)]^\intercal$, with $n = 1, ..., H-1$, where $H \in \mathbb{Z}^+$ is the prediction horizon of the MPC, after applying $\boldsymbol{\kappa}_n = [\kappa_0, ..., \kappa_n]^\intercal = [\kappa(s_o), ..., \kappa(s_n)]^\intercal$ is described as

$$x(\boldsymbol{\kappa}_n) = \sum_{i=0}^{n} \cos\left(\sum_{j=0}^{i} \kappa(s_j)\Delta s_j\right) \Delta s_i \tag{6.1a}$$

$$y(\boldsymbol{\kappa}_n) = \sum_{i=0}^{n} \sin\left(\sum_{j=0}^{i} \kappa(s_j)\Delta s_j\right) \Delta s_i. \tag{6.1b}$$

The equivalent vectorial version for $\boldsymbol{\kappa} = [\kappa_0, ..., \kappa_H]^\intercal$ is

$$\mathbf{x}(\boldsymbol{\kappa}) = \mathbf{S}\cos(\mathbf{S}\boldsymbol{\kappa}) \tag{6.2a}$$

$$\mathbf{y}(\boldsymbol{\kappa}) = \mathbf{S}\sin(\mathbf{S}\boldsymbol{\kappa}), \tag{6.2b}$$

where $\mathbf{S} \in \mathbb{R}^{(H+1)\times(H+1)}$ is a lower triangular matrix that performs the cumulative sum operation along a vector

$$\mathbf{S} = \begin{bmatrix} \Delta s_0 & & & & \\ \Delta s_0 & \Delta s_1 & & & \\ \Delta s_0 & \Delta s_1 & \Delta s_2 & & \\ \vdots & \vdots & \vdots & \ddots & \\ \Delta s_0 & \Delta s_1 & \Delta s_2 & \dots & \Delta s_{l-1} \end{bmatrix}.$$

The EMPC is formulated as the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\kappa}} \quad & ||\mathbf{D}_2\boldsymbol{\kappa}||_2^2 + \alpha||\mathbf{D}_1\boldsymbol{\kappa}||_2^2 && \text{(6.3a)} \\ \text{s. t.} \quad & |\mathbf{x}(\kappa) - \mathbf{x}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_x && \text{(6.3b)} \\ & |\mathbf{y}(\kappa) - \mathbf{y}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_y && \text{(6.3c)} \\ & |\mathbf{D}_1\boldsymbol{\kappa}| \leq \mathbf{1}c_{\max} && \text{(6.3d)} \\ & |\boldsymbol{\kappa}| \leq \mathbf{1}\kappa_{\max} && \text{(6.3e)} \\ & \kappa_0 = \kappa_{\text{vehicle}}, && \text{(6.3f)} \end{aligned}$$

where $\boldsymbol{\kappa} \in \mathbb{R}^{H+1}$ is the curvature vector to be optimized, $\boldsymbol{\varepsilon}_x \in \mathbb{R}^H$ and $\boldsymbol{\epsilon}_y \in \mathbb{R}^H$ are the maximum allowed deviations from the path waypoints in x and y directions, $\mathbf{D}_2 \in \mathbb{Z}^{(H-1)\times(H+1)}$ and $\mathbf{D}_1 \in \mathbb{Z}^{H\times(H+1)}$ are the matrix operators that calculate the second- and first-order differences of a vector, repectively, and $\alpha \in \mathbb{R}$ penalizes the curvature change rate magnitude. The constants $\kappa_{\max}$ and $c_{\max}$ denote the maximum curvature and change rate that the vehicle can perform, respectively, and $\mathbf{1}$ is a column vector filled with ones. Note that the coordinates of the clothoid-based path $[\mathbf{x}, \mathbf{y}]^\intercal$ are only dependent on the curvature $\boldsymbol{\kappa}$, since we can accurately approximate the arc-lengths between points to be equal to the arc-lengths between the reference path waypoints. The inequalities (6.3b) to (6.3e) are performed element-wise, and the equality constraint sets the initial curvature value of the optimization problem.

To calculate the second derivative of the curvature, we use a second-order difference operator, $\mathbf{D}_2$, weighted with the distance between the waypoints. The operator $\mathbf{D}_2$ used to approximate the second-order derivatives of the curvature is based on finite differences. The first derivative of the curvature function with respect to traveled distance is approximated as

$$\frac{\mathrm{d}\kappa(s)}{\mathrm{d}s}\Big|_{s_i} \approx \frac{\kappa(s_{i+1}) - \kappa(s_i)}{|s_{i+1} - s_i|} = \frac{\kappa(s_{i+1}) - \kappa(s_i)}{|\Delta s_i|} \tag{6.4}$$

and consequently the second discrete derivative is

$$\frac{\mathrm{d}^2\kappa(s)}{\mathrm{d}s^2}|_{s_i} \approx \frac{\frac{\kappa(s_{i+1})-\kappa(s_i)}{\Delta s_i} - \frac{\kappa(s_i)-\kappa(s_{i-1})}{\Delta s_{i-1}}}{\Delta s_{i-1,i}} \tag{6.5a}$$

$$= \frac{\left(\kappa(s_{i+1}) - \kappa(s_i)\right)\Delta s_{i-1} - \left(\kappa(s_i) - \kappa(s_{i-1})\right)\Delta s_i}{\Delta s_{i-1,i}\ \Delta s_{i-1}\ \Delta s_i}, \tag{6.5b}$$

where $\Delta s_{i,j} = \frac{\Delta s_i + \Delta s_j}{2}$. Since $\mathbf{D}_2$ is a matrix operator over a vector, we stack (6.5b) from $i = 1, ..., H$ where H is the prediction horizon and get

$$\mathbf{D}_2 = \begin{bmatrix} -\frac{\Delta s_1}{\Delta s_{0,1}\Delta s_0\Delta s_1} & \frac{\Delta s_0+\Delta s_1}{\Delta s_{0,1}\Delta s_0\Delta s_1} & -\frac{\Delta s_0}{\Delta s_{0,1}\Delta s_0\Delta s_1} & & \\ & \ddots & \ddots & \ddots & \\ & -\frac{\Delta s_H}{\Delta s_{H-1,H}\Delta s_{H-1}\Delta s_H} & \frac{\Delta s_{H-1}+\Delta s_H}{\Delta s_{H-1,H}\Delta s_{H-1}\Delta s_H} & -\frac{\Delta s_{H-1}}{\Delta s_{H-1,H}\Delta s_{H-1}\Delta s_H} & \end{bmatrix}. \tag{6.6}$$

Similarly to $\mathbf{D}_2$, $\mathbf{D}_1$ is obtained by stacking (6.4),

$$\mathbf{D}_1 = \begin{bmatrix} \frac{1}{\Delta s_0} & -\frac{1}{\Delta s_0} & & & \\ & \frac{1}{\Delta s_1} & -\frac{1}{\Delta s_1} & & \\ & & \ddots & \ddots & \\ & & & \frac{1}{\Delta s_{H-1}} & -\frac{1}{\Delta s_{H-1}} \end{bmatrix}. \tag{6.7}$$

The problem formulation (6.3) is not in convex form due to the fact that the reconstructed path $[\mathbf{x}(\boldsymbol{\kappa}), \mathbf{y}(\boldsymbol{\kappa})]^\intercal$ is computed using (6.2), which are clearly nonlinear equations. A linear approximation is done using a first-order Taylor approximation of both cosine and sine around the path orientation $\boldsymbol{\theta}^{\text{ref}}$. The linearization around $\boldsymbol{\theta}^{\text{ref}}$ takes the form

$$\mathbf{x}(\boldsymbol{\kappa}) = \mathbf{S}\left(\cos(\boldsymbol{\theta}^{\text{ref}}) - \sin(\boldsymbol{\theta}^{\text{ref}}) \odot (\mathbf{S}\boldsymbol{\kappa} - \boldsymbol{\theta}^{\text{ref}})\right) \tag{6.8a}$$

$$\mathbf{y}(\boldsymbol{\kappa}) = \mathbf{S}\left(\sin(\boldsymbol{\theta}^{\text{ref}}) + \cos(\boldsymbol{\theta}^{\text{ref}}) \odot (\mathbf{S}\boldsymbol{\kappa} - \boldsymbol{\theta}^{\text{ref}})\right), \tag{6.8b}$$

where $\odot$ is the Hadamard product. Also, the problem may be infeasible when there is no solution that respects the constraints (6.3b) and (6.3c). Therefore, we add a slack variable $\boldsymbol{\Delta} = [\boldsymbol{\Delta}_x^\intercal\ \boldsymbol{\Delta}_y^\intercal]^\intercal$ and minimize the cost function taking the slack variable into account as well. Also, with the inclusion of the slack variable $\boldsymbol{\Delta}$, the deviations from the path are explicitly included in the cost function that enforces convergence to the reference path. In conclusion, the EMPC problem is formulated to yield the linear function with the smallest slope that produces a clothoid-like vehicle motion respecting the constraints. The final formulation is

$$\min_{\boldsymbol{\kappa},\boldsymbol{\Delta}} \quad ||\mathbf{D}_2\boldsymbol{\kappa}||_2^2 + \alpha||\mathbf{D}_1\boldsymbol{\kappa}||_2^2 + \boldsymbol{\Delta}^\intercal\boldsymbol{\Lambda}\boldsymbol{\Delta} \tag{6.9a}$$

$$\text{s. t.} \quad |\mathbf{x}(\boldsymbol{\kappa}) - \mathbf{x}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_x + \boldsymbol{\Delta}_{\boldsymbol{x}} \tag{6.9b}$$

$$|\mathbf{y}(\boldsymbol{\kappa}) - \mathbf{y}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_y + \boldsymbol{\Delta}_{\boldsymbol{y}} \tag{6.9c}$$

$$|\mathbf{D}_1\boldsymbol{\kappa}| \leq \mathbf{1}c_{\max} \tag{6.9d}$$

$$|\boldsymbol{\kappa}| \leq \mathbf{1}\kappa_{\max} \tag{6.9e}$$

$$\boldsymbol{\Delta} \geq 0 \tag{6.9f}$$

$$\kappa_0 = \kappa_{\text{vehicle}}, \tag{6.9g}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{2H\times 2H}$ is a diagonal matrix of the form $\text{diag}(\lambda_1, ..., \lambda_H, \lambda_1, ..., \lambda_H)$ that penalizes constraint violations.

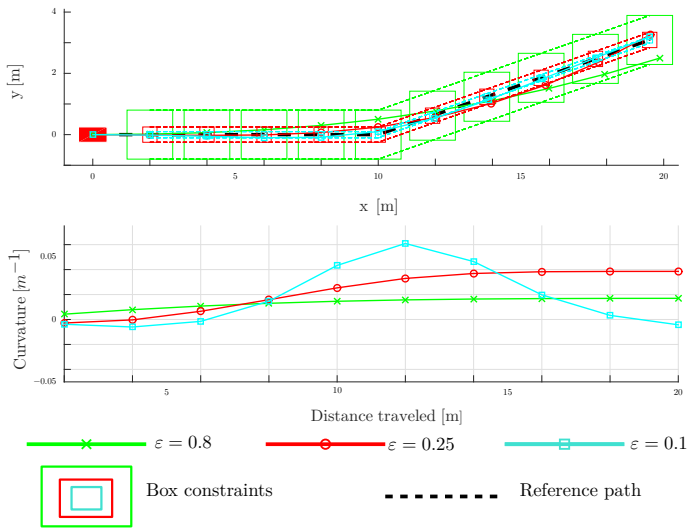
In Appendix B, the final formulation of the controller is cast as a QP by defining the optimization variable $\mathbf{u} = [\boldsymbol{\kappa}, \boldsymbol{\Delta}]^\intercal$.

It is also interesting to compare the path sparsification algorithm optimization problem (4.4), in Chapter 4, with the EMPC formulation (6.9). In the path sparsification algorithm the l_0-norm is used to achieve a piecewise linear curvature profile with as few kink-points (i.e., the points where the first derivative of the curvature is not continuous). In the EMPC formulation, the intended curvature function is approximately a linear function with a small slope. In this case, there is no strict requirement regarding sparsity or linearity due to the utilization of the l_2-norm. In a single clothoid path, the second-order derivative of the curvature is zero. In opposition to the path sparsification problem, this is not imposed in the EMPC cost function (6.9a) and consequently, the EMPC does not lead to a clothoidal driving. However, since the EMPC objective is to minimize the second-order derivative of the curvature function, a quasi-clothoid driving is the desired and expected behavior. As described in Chapter 2, clothoids are widely used in road design due to their property of having a piecewise linearly varying curvature, and therefore, a quasi-clothoid driving provides a smooth driving.

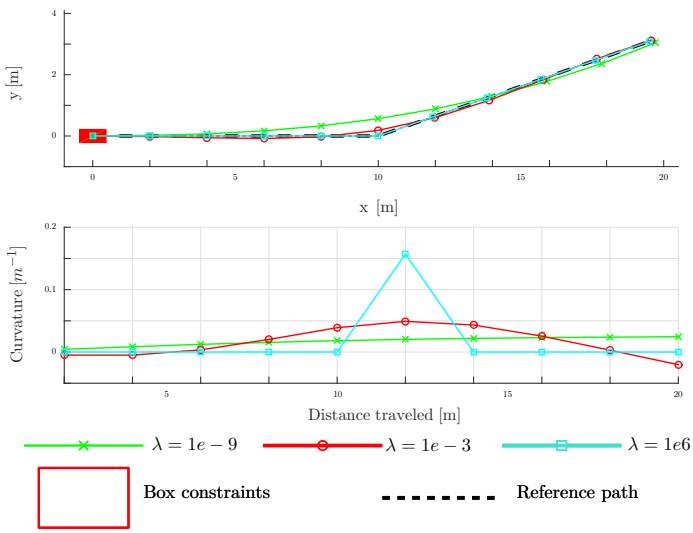
6.2 Simulation example and discussion

In this section, we present simulation results using the EMPC presented in Section 6.1. We analyze the tuning parameters influence on the optimization solution with an illustrative example. Also, we demonstrate the controller ability to accurately follow a path in simulation.

We constrain the vehicle's predicted position to be inside of a box around the waypoint. By designing an off-road driving controller we can interpret the dimension of such constraint as the vehicle's desired maximum deviation from the path. In case of road driving, this is equivalent to the lane width minus half of the vehicle width. Hence, it is crucial to understand how the tuning parameters $\boldsymbol{\varepsilon}$ and $\boldsymbol{\lambda}$ influence the driving experience. The first one determines the size of the box con-



(a) Simulation of one run of the controller with $H = 10$, where the size box constraint $\boldsymbol{\varepsilon}$ is varied.



(b) Simulation of one run of the controller with $H = 10$, where the penalization to violate the constraint $\boldsymbol{\lambda}$ is varied. In this case, $\boldsymbol{\lambda}$ is constant along the horizon.

Figure 6.1: EMPC position accuracy parameters influence.

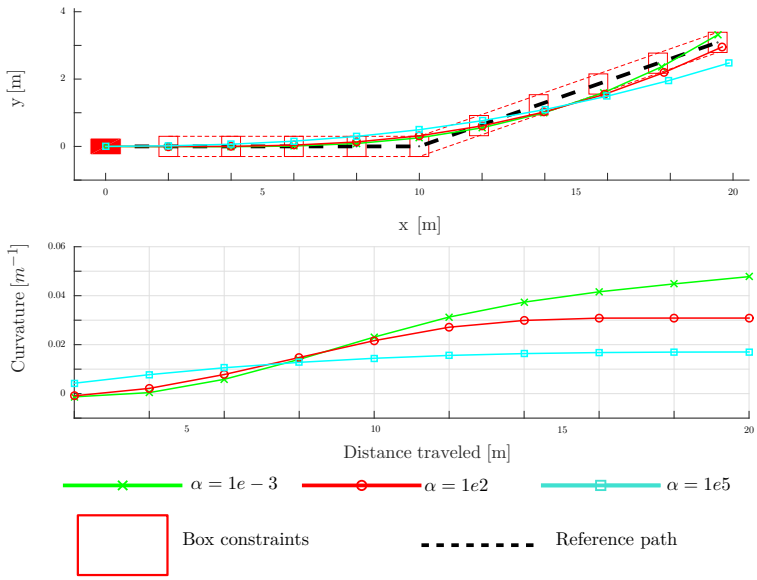
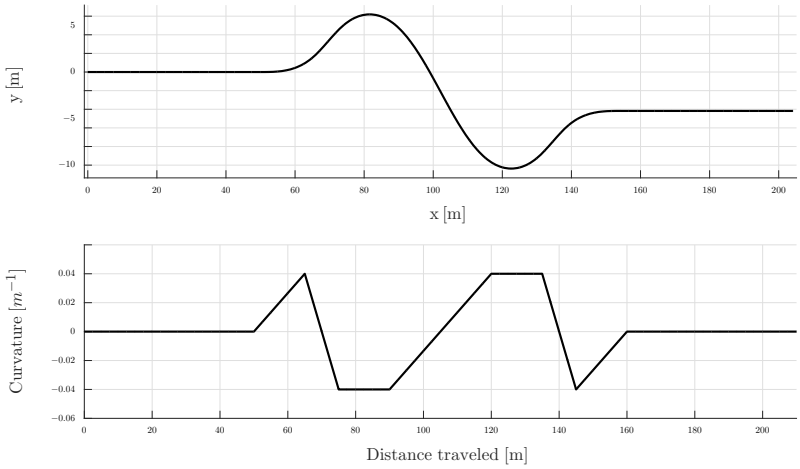


Figure 6.2: Simulation of one run of the controller with $H = 10$, where the importance given to the first derivative of curvature α is varied.

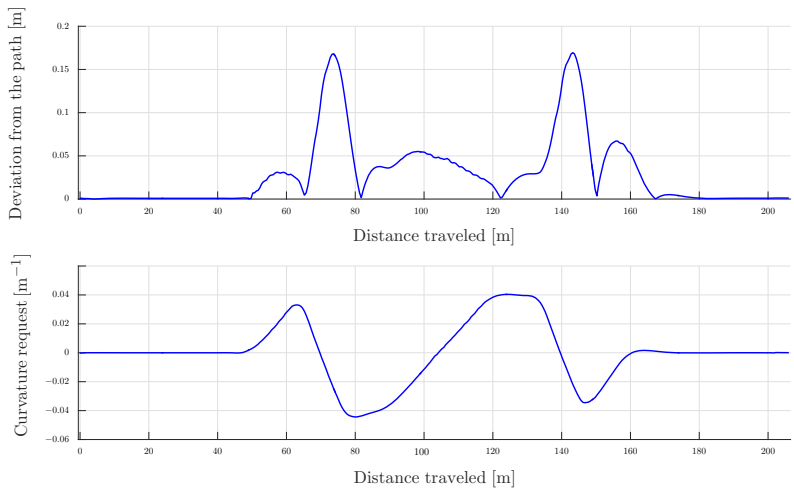
straints and consequently how much freedom the controller has to choose a suitable curvature function such that the predicted path is within these constraints. The latter parameter sets how flat (or how sharp) the quadratic penalization $||\boldsymbol{\Delta}||_2^2$ is when such predicted path violates the box constraints.

Figures 6.1a and 6.1b depict the influence of the parameters $\boldsymbol{\varepsilon}$ and $\boldsymbol{\lambda}$, respectively. In Figure 6.1a, we see the effect of the size of the box constraints. The larger the box constraint, the closer to linear the curvature function is. In Figure 6.1b, $\boldsymbol{\lambda}$ is constant over the horizon and we study how the curvature function changes for different values of $\boldsymbol{\lambda}$. The smaller $\boldsymbol{\lambda}$ is, the closer to linear the curvature function is. However, the predicted vehicle path can lay outside the constraint boxes with low penalization. In the extreme case $\boldsymbol{\lambda} = 0$, the curvature function is the current vehicle curvature throughout the horizon.

Another parameter to consider is α, which sets the importance of maintaining the current constant curvature throughout the horizon. Figure 6.2 depicts the influence of α on the curvature function. The larger α is, the more constant the curvature function is. We can see that if we penalize $||\mathbf{D}_1\boldsymbol{\kappa}||_2^2$ more, the variation in the curvature decrease. This is an intended property of the controller, since the term was included to regularize the amount of steering that we need in order to follow the path. This way it is possible to favour changing the curvature as little as possible.



(a) Reference path used to illustrate the EMPC performance. In the figure, the reference path is represented on the plot above, while the reference curvature is represented in the figure below. The reference is a double S-curve that consists of nine clothoid segments.



(b) Illustrative example of the EMPC performance. Above is depicted the vehicle deviation from the path, while below is shown the controller curvature request.

Figure 6.3: Reference path and EMPC performance.

We study the EMPC path following performance resorting the same reference path as in Chapter 5. The reference is a double S-curve that consists of nine

Table 6.1: EMPC parameters tuning.

Parameters	EMPC
Prediction horizon, H	10
Path sampling time, T_s	200 ms
Controller frequency	50 Hz
Slack penalization, λ_i	200
First derivative penalization, α	200
Box constraints sizes, ε_i	0

clothoid segments. We show it again in Figure 6.3a for the reader's convenience. Figure 6.3b depicts one run of the EMPC in the same reference path, where the vehicle drives at constant speed of 5 m/s. As previously stated, although this might be a fairly low speed, the reference path includes two sharp curves. The simulation uses the vehicle dynamics described in Chapter 3. The tuning parameters used in the EMPC are summarized in Table 6.1. All the optimization problems are solved using *cvxgen*, which is a custom optimization solver [110]. From Figure 6.3b, it is clear that the vehicle follows the path accurately, since the maximum deviation from the path is only about 16 cm.

In Chapter 8, we present a more complete demonstration of the EMPC performance. The controller is deployed in a real Scania construction truck, allowing us to provide real experimental results demonstrating the EMPC performance. We also compare it with a pure-pursuit controller. Moreover, in simulation, using the accurate dynamic vehicle model and real reference paths both with high and low speeds, we compare the EMPC with the MPCC, presented in Chapter 5, using a standard tracking MPC approach and a pure-pursuit controller.

6.3 Summary

In this chapter, we proposed an economic model predictive controller (EMPC) for autonomous path following. While in standard reference tracking MPC the cost function is zero at the reference path, in the EMPC the cost function is included an economic cost associated with the plant operation. In this case, the economic cost is driving comfort and smoothness. With this purpose, we minimize the second and first derivatives of the vehicle curvature. So, we encourage the vehicle to be guided using linear-varying curvature in order to provide a smooth driving. Nevertheless, we added soft constraints to the formulation in order to always drive in the reference path vicinity. In the end, we illustrated the controller performance and the influence of the parameter tuning in simulation. We concluded that the EMPC performs well with a deviation never exceeding 16 cm.

Chapter 7

Longitudinal speed profiler and controller

Full autonomy can only be achieved combining lateral and longitudinal control. So far in this thesis, we have described methods to deal with the lateral control of an autonomous vehicle. The longitudinal controller is responsible for accurately tracking a given speed profile. In this chapter, we address the problem of speed profile tracking. Additionally, the interface with the autonomous vehicle is done through the cruise controller and the brake controller as they receive speed and acceleration requests, respectively. Just like the lateral controller needs a reference path, the longitudinal controller needs a speed profile to track. Therefore, we also address the problem of generating a speed profile on paths with known semantic (e.g., speed limit and geometric information). We formulate the problem of speed profile generation as a simple and intuitive optimization problem that has only one tuning parameter. The speed profiler is formulated as an offline optimal control problem where the intended goal is to drive as close as possible to the speed limit, while respecting safety and comfort constraints. The longitudinal controller is a receding horizon controller and its formulation resembles the speed profiler. In this case, the objective is to track the speed profile as precisely as possible by adjusting the speed and acceleration requests sent to the cruise controller of the vehicle.

We address the optimal speed profile generation in Section 7.1 and in Section 7.2, we introduce the longitudinal controller. In Section 7.3, the performance of both the speed profiler and the longitudinal controller is evaluated. Finally, Section 7.4 summarizes the chapter.

7.1 Speed profile generation

The generation of a speed profile is a crucial step for autonomous driving. It defines the desired vehicle speed at each point of the reference path. For instance, it has to respect speed regulation and avoid dangerous maneuvers that can lead to tire slipping or rollover. The reference path, as described in the previous chapters, does not include a speed profile to be followed. In this section, we address the problem of generating a feasible discrete speed profile in space-domain that takes into account road regulation, comfort, safety constraints and vehicle limitations.

Let a reference path be constituted by N waypoints of the form $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}, \theta_i^{\text{ref}}, \kappa_i^{\text{ref}}, s_i]^\intercal$, $i = 1, ..., N$, which represent the path Cartesian coordinates, $[x_i^{\text{ref}}, y_i^{\text{ref}}]^\intercal$, the orientation, θ_i^{ref}, the curvature, κ_i^{ref}, and the distance traveled along the path since the first point, s_i. Also, let $v_i^{\max}$ be the maximum speed that the vehicle can have at the i-th waypoint, $a_i^{\max}$ and $a_i^{\min}$ be the vehicle's maximum and minimum acceleration at the i-th waypoint respectively, and f be a function that relates speed with acceleration in space-domain. Regarding the derivation of the function f, in continuous-time we know that

$$\frac{\mathrm{d}v(t)}{\mathrm{d}t} = a(t), \tag{7.1}$$

but since $v(t) \cdot \mathrm{d}t = \mathrm{d}s$, assuming that $v(t) \neq 0$ and $v(t)$ is a continuous function, then

$$\frac{\mathrm{d}v(s)}{\mathrm{d}s} = \frac{a(s)}{v(s)}. \tag{7.2}$$

Integrating both sides of the expression (7.2) and assuming constant acceleration between waypoints, we can represent $a(s)$ as a function of $v(s)$ using a quadratic expression. Thus,

$$\frac{\mathrm{d}v(s)}{\mathrm{d}s} = \frac{a(s)}{v(s)} \tag{7.3a}$$

$$\Leftrightarrow \quad v(s)\mathrm{d}v = a(s)\mathrm{d}s \tag{7.3b}$$

$$\Rightarrow \quad \int_{v_i}^{v_{i+1}} v(s)\,\mathrm{d}v = \int_{s_i}^{s_{i+1}} a(s)\,\mathrm{d}s \tag{7.3c}$$

$$\Rightarrow \quad v_{i+1}^2 - v_i^2 = 2(s_{i+1} - s_i)a_i \tag{7.3d}$$

$$\Leftrightarrow \quad v_{i+1}^2 - v_i^2 = 2l_i a_i \tag{7.3e}$$

$$\Leftrightarrow \quad a_i = \frac{v_{i+1}^2 - v_i^2}{2l_i}, \tag{7.3f}$$

where l_i, is the distance between two consecutive waypoints. So, the speed description in discrete space is

$$v_{i+1}^2 = v_i^2 + 2l_i a_i. \tag{7.4}$$

Let the speed limit of the road (or user-specified) be $v_{\max}^{\text{law}}$, and the maximum speed due to comfort and safety reasons be $v_{\max}^{\text{road}}$. In particular, since the road contains, on each waypoint, the curvature information κ_i^{ref}, we can relate that with the maximum lateral acceleration a_y allowed and consequently the maximum speed $v_{\max}^{\text{road}}$ allowed on each stretch [111]. Hence, $v_{\max}^{\text{road}}$ can be expressed as

$$v_{\max}^{\text{road}} = \sqrt{\frac{a_y^{\max}}{\kappa_i^{\text{ref}}}}, \tag{7.5}$$

where $a_y^{\max}$ is the maximum allowed lateral acceleration given the road and the vehicle properties. In [112], a second-order transfer function relating the lateral load transfer and lateral acceleration is analytically derived. To constrain the speed profile using a constant maximum lateral acceleration, then a conservative value has to be chosen to accommodate for the peak resonance of the second-order dynamics.

Consider that the speed profiler can access information about the road speed limit and that the limit is given for each path waypoint, $v_{\max}^{\text{law}}$. Also, the path geometry makes it to compute the maximum speed allowed $v_{\max}^{\text{road}}$ depending on the path curvature, for instance. Therefore, the maximum speed allowed on the road is

$$v_i^{\max} = \min\{v_{\max_i}^{\text{law}}, v_{\max_i}^{\text{road}}\}, \tag{7.6}$$

where $i = 1, .., N$.

With this in mind, we formulate a space-domain speed profiler over the entire path described by N waypoints as the solution of the following optimization problem:

$$\min_{v_i^{\text{profile}}} \quad \sum_{i=1}^{N}(v_i^{\text{profile}} - v_i^{\max})^2 + \alpha \sum_{i=1}^{N-1} a_i^2 \tag{7.7a}$$

$$\text{s.t.} \quad a_i^{\text{profile}} = f(v_{i+1}^{\text{profile}}, v_i^{\text{profile}}, l_i), \quad i = 1, ..., N-1, \tag{7.7b}$$

$$a_i^{\min} \leq a_i^{\text{profile}} \leq a_i^{\max}, \quad i = 1, ..., N-1, \tag{7.7c}$$

$$0 \leq v_i^{\text{profile}} \leq v_i^{\max}, \quad i = 1, ..., N, \tag{7.7d}$$

$$v_1^{\text{profile}} = v_{\text{init}}, \tag{7.7e}$$

$$v_N^{\text{profile}} = v_{\text{final}}, \tag{7.7f}$$

where, α is a constant smoothing parameter that penalizes accelerations different from zero and $v_{\text{init}} \in \mathbb{R}$ and $v_{\text{final}} \in \mathbb{R}$ are predefined initial and final speed values respectively. Therefore, there are two contradictory objectives in the speed profiler. The first one is to compute the fastest speed profile. The second one is to accelerate as few times as possible. If α is chosen sufficiently large, the speed profile is to maintain a constant speed throughout the path. The speed profile obtained

by (7.7) is a linear function with respect to the traveled distance, since (7.3) assumes constant acceleration. If the path is described by clothoid segments, the maximum velocity allowed, which is only influenced by the curvature at each waypoint, varies linearly with the traveled distance. Therefore, when the speed profile matches the maximum speed at all waypoints, the maximum speed is not violated between waypoints.

The problem of computing a speed profile that minimizes the difference to the maximum speed allowed and penalizes acceleration changes can be solved using l_2-norm regularization techniques. We can rewrite problem (7.7) in matrix form and minimize with respect to $[v_1^2, v_2^2, ..., v_N^2]^\intercal$ to be compliant with the speed discretization (7.4) and make a change of variables $\mathbf{w} = [w_1, w_2, ..., w_N]^\intercal = [v_1^2, v_2^2, ..., v_N^2]^\intercal$ as

$$\min_{\mathbf{w}} \quad ||\mathbf{w}_{\text{profile}} - \mathbf{w}_{\max}||_2^2 + \alpha||\mathbf{D}_1\mathbf{w}||_2^2 \tag{7.8a}$$

$$\text{s.t.} \quad \mathbf{a}_{\min} \leq \mathbf{D}_1\mathbf{w}_{\text{profile}} \leq \mathbf{a}_{\max}, \tag{7.8b}$$

$$\mathbf{0} \leq \mathbf{w}_{\text{profile}} \leq \mathbf{w}_{\max} \tag{7.8c}$$

$$w_1^{\text{profile}} = w_{\text{init}}, \tag{7.8d}$$

$$w_N^{\text{profile}} = w_{\text{final}}, \tag{7.8e}$$

where the subscripts on $\mathbf{w}$ are self-explanatory and have the same meaning as in (7.7) and $\mathbf{D}_1 \in \mathbb{Z}^{(N-1)\times N}$ is the matrix operator that calculates first-order differences of a vector and includes the distances between waypoints l_i:

$$\mathbf{D}_1 = \begin{bmatrix} \frac{-1}{2l_1} & \frac{1}{2l_1} & & & & & \\ & \frac{-1}{2l_2} & \frac{1}{2l_2} & & & & \\ & & \ddots & \ddots & & & \\ & & & \frac{-1}{2l_{N-1}} & \frac{1}{2l_{N-1}} & & \\ & & & & \frac{-1}{2l_N} & \frac{1}{2l_N} \end{bmatrix}.$$

The speed profile generation is done offline and it is tracked by a longitudinal controller as described in the next section.

7.2 Longitudinal controller

In this section, we present a longitudinal controller that is similarly formulated as the speed profiler. The longitudinal controller computes speed requests that are fed to a low-level speed controller, such as a cruise controller, that is responsible to track them. Figure 7.1 depicts the proposed system structure, where the controller consists of a decoupled longitudinal and lateral controller. The longitudinal controllers receives a reference speed profile and the curvature predictions computed by the lateral controller. While the speed profiler uses the maximum speed depending on the road geometry, the longitudinal controller uses an up-to-date curvature

prediction and constrains the maximum speed allowed depending on that. The controller outputs a set of requests to the low-level controllers of the vehicle. In particular, the longitudinal controller sends a speed request to the cruise controller or an acceleration request to the brake controller.

The longitudinal controller is formulated similarly to (7.8). Nevertheless, it works in a receding-horizon fashion and so, it computes the optimal input for a certain horizon taking into account the speed profile and the predicted curvature provided by the lateral controller. The problem is formulated as a tracking problem where we want to minimize the deviations from the reference speed profile and at the same time using the most recent lateral controller predictions to compute a desired speed within the maximum lateral acceleration limits (7.5). Using the same change of variable $\mathbf{w} = [w_1, w_2, ..., w_H]^\intercal = [v_1^2, v_2^2, ..., v_H^2]^\intercal$, where $H \in \mathbb{Z}^+$ is the prediction horizon, the controller is formulated as

$$\min_{\mathbf{w}_{\text{vehicle}}, \Delta} \quad ||\mathbf{w}_{\text{vehicle}} - \mathbf{w}_{\text{profile}}||_2 + \beta\Delta \tag{7.9a}$$

$$\mathbf{a}^{\min} \leq \mathbf{D}_1 \mathbf{w}_{\text{vehicle}} \leq \mathbf{a}^{\max} \tag{7.9b}$$

$$\mathbf{0} \leq \mathbf{w}_{\text{vehicle}} \leq \mathbf{w}_{\max} + \Delta, \tag{7.9c}$$

$$w_{\text{vehicle}_1} = w_{\text{current}}, \tag{7.9d}$$

$$\Delta \geq 0, \tag{7.9e}$$

where a slack variable $\Delta \in \mathbb{R}_0^+$ is included to allow the optimization problem feasibility even if the vehicle overspeeds the maximum allowed speed and the weight $\beta \in \mathbb{R}_0^+$ penalizes this overspeed. For example, if the slack variable was not included, we could face an infeasible problem when the vehicle enters a descent and exceeds the maximum allowed speed or when an obstacle appears suddenly and it is impossible to brake on time. The constraint (7.9d) represents the initial condition of the optimization problem.

Assuming that the vehicle low-level controller is able to track the speed requests within the acceleration limits defined in the constraints, the speed predictions are sufficiently accurate to allow finding the the speed request, which is sent to the low-level cruise controller, to be found by computing the predicted vehicle speed in T_p seconds, where T_p is approximately the response time of the vehicle and the cruise controller to a speed request. In parallel, the acceleration required to request that speed is computed (i.e., $a_{\text{request}} = \frac{w_2 - w_1}{2l_1}$). If $a_{\text{request}} < a_b < 0$, where a_b is constant threshold, the output of the controller is an acceleration request instead of a speed request. The transition between acceleration and speed request is subject to a small hysteresis to avoid flickering.

7.3 Simulation examples and discussion

In this section, we present the main results obtained with the speed profiler and afterwards the longitudinal controller. The reference path is similar to the one

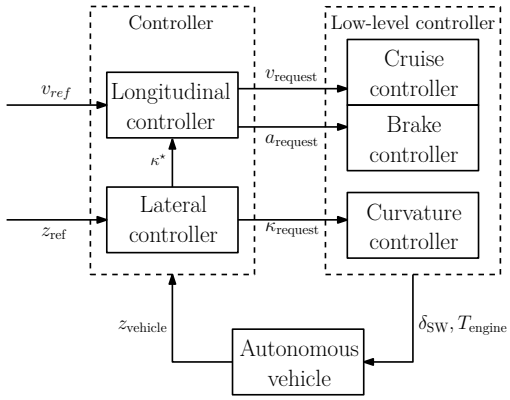


Figure 7.1: The interfaces between the lateral and the longitudinal controller as well as with the low-level controllers. The overall controller receives information about the reference path and speed and outputs reference curvature, speed and acceleration values. The low-level controllers are responsible for tracking the requests as well as possible by actuating on the steering wheel angle and on the engine torque.

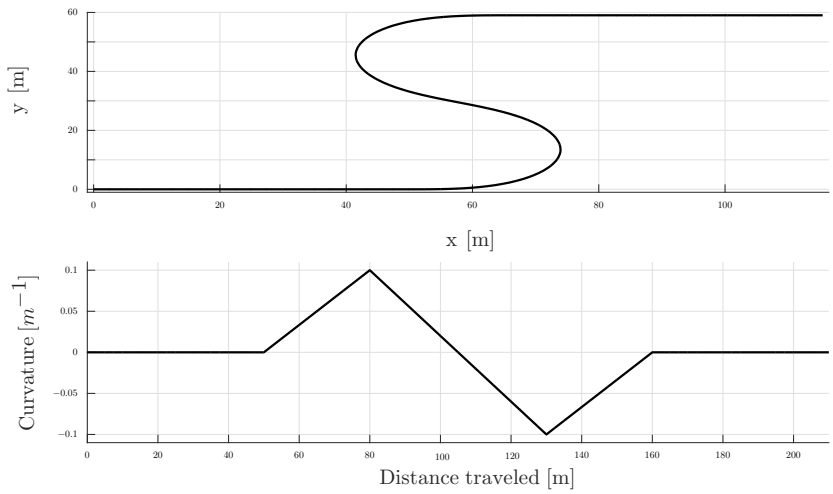
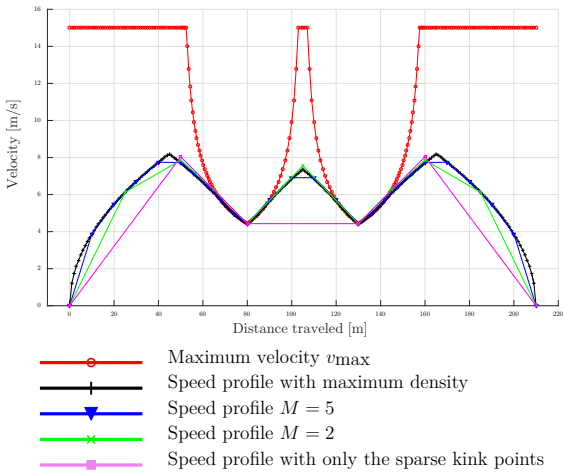
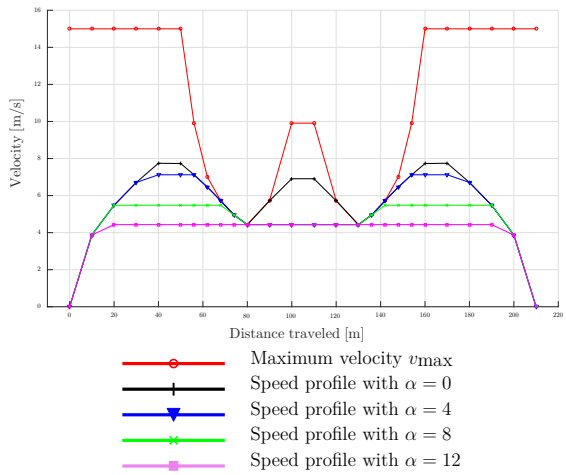


Figure 7.2: Single S-curve reference path used to illustrate the speed profiler performance. The reference path is represented on the plot above, while the reference curvature is represented in the plot below. The reference path is a single S-curve that consists of five clothoid segments.



(a) Different speed profiles for different path representation densities for $\alpha = 0$. The maximum velocity $v_{\max}$ is obtained using (7.6) in a dense path representation. By varying the number of waypoints (M) describing each clothoid the speed profiler produces different solutions.



(b) Different speed profiles for different acceleration penalization values α that penalizes non-zero accelerations. The reference path is oversampled with $M = 5$ (i.e., dividing each clothoid segment in 5 smaller segments). The bigger α the flatter the speed profile is.

Figure 7.3: Influence of the path representation density and acceleration penalization on the speed profile.

used in the previous chapters, but in this case it is a single S-curve that consists of a set of five clothoids connected together as show in Figure 7.2. The desired curvature values and arc-lengths are user-defined. The reference path is ideal to eliminate possible sources of errors and provide a proper proof of concept, since the curvature function is piecewise linear as assumed in the problem formulation. All the optimization problems are solved using *cvxgen*, which is a custom optimization solver [110].

Speed profiler

In Chapter 5 we discuss the influence of the waypoint sampling distance on the lateral controller accuracy. Hence, we discuss the performance of the longitudinal controller for different path representation densities, where we divide each clothoid in $M+1$ segments (i.e., sampling M times between the kink-points).

The speed profiler (7.8) is applied to the reference path with parameters

- Vehicle acceleration limits, $a_{\max}^{\text{profile}} = -a_{\min}^{\text{profile}} = 0.75$ m/s^2;
- Maximum allowed speed on the road, $v_{\max}^{\text{law}} = 25$ m/s;
- Maximum allowed lateral acceleration, $a_y^{\max} = 0.15g$ m/s^2, where $g = 9.82$ m/s^2 is the gravitational acceleration;
- Speed profile initial and final speed, $v_{\text{init}}^{\text{profile}} = v_{\text{final}}^{\text{profile}} = 0$ m/s,

and we obtain the speed profiles shown in Figure 7.3a and 7.3b.

In Figure 7.3a, the influence of different path densities on the speed profile is analyzed where $\alpha = 0$ is fixed. The maximum velocity $v_{\max}$ is represented using a dense path representation. We intend to analyze the suboptimality properties of the speed profile obtained when the path representation is sparser. We assume that the optimal speed profile is the one obtained with a continuous path. Thus, we compute it using a dense path representation, which we assume to be a good approximation. We can see that, except when the path is too sparse, the speed profile is is similar to the optimal solution. Actually, note that with $M = 5$, the speed profile is already similar with the optimal one although it is a considerable sparser path representation. We conclude that the speed profiler can handle a sparse path representation and give suboptimal feasible speed references. In cases of hard time constraints or the need for fast speed profile recalculation, the speed profiler can use sparser path representations and still produce a feasible speed profile.

In Figure 7.3b, the influence of the penalization factor α in the speed profile is studied where the path representation density $M = 5$ is fixed. In this case, we do not analyze optimality but smoothness. In compliance with our design, we clearly see that the larger the acceleration penalization, the smoother the speed profile is. It depends on the application whether a speed profile should be more or less aggressive. For instance, for a heavily loaded truck the design may favour the least change of speeds, while when the truck is unloaded it might be favourable to go as

quickly as possible to the next loading site. The proposed speed profiler allows this kind of parameter tuning.

Longitudinal controller

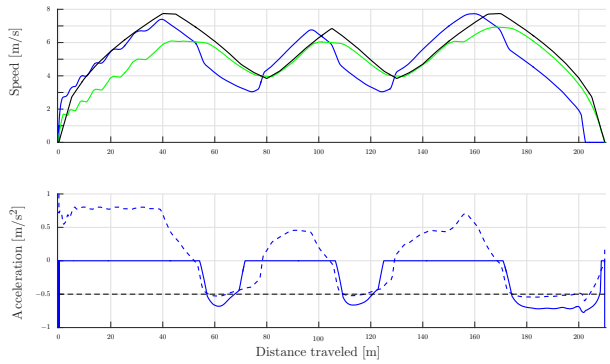
The purpose of the longitudinal controller is to provide a speed or acceleration request to the cruise controller or brake controller of the vehicle such that it tracks the speed profile computed offline. The longitudinal controller computes an optimal speed or acceleration request over a certain horizon, and taking into account the curvature predictions made by the lateral controller. For example, in case of an unexpected event, the vehicle may need to avoid an obstacle making a sharp turn, and in this case the speed request would be significantly reduced since the predicted curvature requests are high.

We tested the longitudinal controller on the same reference path as the speed profiler to evaluate its performance. The controller parameters were chosen as

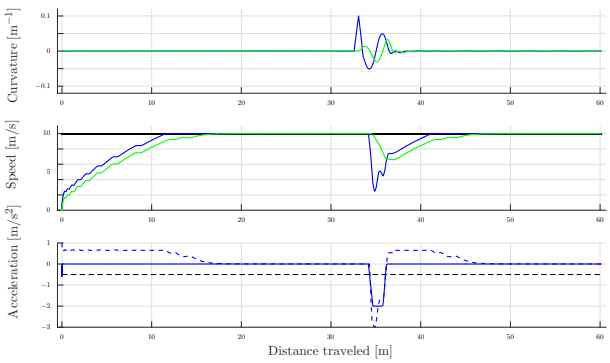
- Vehicle acceleration limits, $a_{\max}^{\text{vehicle}} = -a_{\min}^{\text{vehicle}} = 2$ m/s^2;
- Maximum allowed speed on the road, $v_{\max}^{\text{law}} = 25$ m/s;
- Maximum allowed lateral acceleration, $a_y^{\max} = 0.15g$ m/s^2, where $g = 9.82$ m/s^2 is the gravitational acceleration;
- Braking acceleration threshold, $a_b = -0.5$ m/s^2;
- Speed request time-gap, $T_p = 2.5$ s;
- Overspeed penalization weight, $\beta = 10^3$.

Figure 7.4a depicts the performance of the longitudinal controller in the single S-curve shown in Figure 7.2. We can see that the vehicle smoothly and precisely tracks the speed profile by applying the cruise control or the brakes when needed. We note that, in the beginning, the speed tracking is not accurate due to the gear changing. In a truck, the gear changing can take 1-2 seconds, and during that time there is no torque applied on the vehicle, causing it to lose speed on those instants. We do not take into account, neither on the speed profiler or on the longitudinal controller, gear changes and their impact on the maximum vehicle acceleration. We can see, though, that after the first curve (80 m) the vehicle precisely tracks the speed profile since it does not need to change gear. The speed profile is well tracked both with the cruise controller and with the brake controller. However, we can see that in speed profile peaks (for instance at 110 m) the vehicle is unable to reach that speed. Since the vehicle and the cruise controller dynamics are slow, the time T_p used to request the speed is large, and therefore, the vehicle does not achieve the desired speed in this situations.

To demonstrate the ability of adapting the speed depending on the dynamic environment, we present a situation where the vehicle is subject to an unexpected disturbance. At some point, for about 1 second, the curvature request is overridden



(a) The longitudinal controller tracking a speed profile. The plot above represents the reference, requested and vehicle speeds. The plot below shows the acceleration request.



(b) The longitudinal controller deviates from the speed profile in the case of an unexpected incident. The plot above represents the requested and vehicle curvatures. Around the distance traveled of 31 m the vehicle is subject to an external disturbance, which consists in overriding the curvature request for 1 second with 0.1 m^{-1}. The longitudinal controller receives the updated predicted curvature from the lateral controller and slows down the vehicle by braking, as shown in the middle and lower plot respectively. This provides a safe and smooth way to come back to the reference path.

Reference speed profile
Curvature/speed/acceleration request
Vehicle curvature/speed
Acceleration required to achieve the speed request
Braking acceleration threshold, a_b

Figure 7.4: Reference path and EMPC performance.

that obligates the lateral control to compensate. Because of that, the speed profile is no longer feasible since a lower maximum speed is allowed to be compliant with comfort or safety reasons. Figure 7.4b shows the result of the simulation. The reference path consists of a straight line with a constant speed profile of 10 m/s. Around the distance traveled of 32 m, the vehicle is subject to a curvature request of 0.1 m^{-1}. Also, since the predicted vehicle curvature is sent to the longitudinal controller the vehicle brakes, lowering the vehicle speed, returning safely and smoothly to the path.

7.4 Summary

In this chapter, we addressed two problems how to create a space-based speed profile for a given path and how to track it using the cruise and brake controller available in the vehicle. Our approach to the speed profile generation problem is to solve an offline optimization problem with two contradictory objectives, namely to drive as fast as possible while penalizing big accelerations. The optimization problem constraints respect the vehicle limitations and road regulations. The longitudinal controller is formulated in a similar fashion as the speed profiler. In this case, the controller objective is to track the speed profile generated *a priori*, with respect to identical constraints. One of the main properties of the controller is that it receives the updated predicted vehicle curvature from the lateral controller, allowing it to adapt the vehicle speed in the case of an unexpected incident. In the end, we provided illustrative examples to demonstrate both the speed profiler and the longitudinal controller. We analyzed the speed profiler's main properties by changing the acceleration penalization factor, α, which penalizes non-zero accelerations. As expected, the larger the acceleration penalization, the smoother the speed profile is.

Also, we compare the performance of the method with different path representation densities, showing that, with few points, the algorithm is able to generate close to optimal speed profiles. We simulated the longitudinal controller with a generated speed profile. We concluded that, except where the gear changes have noticeable effect, the longitudinal controller is able to track the speed profile both with the cruise and brake controller. We also exemplified the case when the longitudinal controller needs to adapt to an unexpected event, where the vehicle curvature predictions limit the vehicle maximum speed. For instance, in the example provided, the vehicle is subject to an external disturbance and the lateral controller reacts in opposite direction to guide the vehicle towards the reference path again. In this situation, the longitudinal controller slows the vehicle down by braking, which makes the vehicle return smoothly and safely to the path.

Chapter 8

Experimental controller benchmarking

Experimental evaluation is the final step for controller validation. Also, our goal is to demonstrate the control performance in a real vehicle. Therefore, we deployed the economic MPC (EMPC), described in Chapter 6, on a Scania construction truck. We compare the performance of the EMPC with a pure-pursuit controller (PPC) that was previously implemented on the truck. Furthermore, we develop a simulation environment that resembles the real system by using the vehicle model described in Chapter 3. This simulation environment allows evaluating and validating the control design before deploying it in the experimental platform. Consequently, the EMPC and the clothoid-based MPC (MPCC), presented in Chapter 5, are compared with a PPC and a standard MPC in simulation. We analyze the performance of the controllers in terms of path tracking accuracy and driving smoothness. We discuss the simulation environment reliability as a test environment keeping in mind that no matter how detailed the simulation environment is, it will never be identical to reality. There are always unmodeled dynamics, noise, disturbances and delays that cannot be represented in the simulation environment.

The main idea behind the experimental controller benchmarking performed in this chapter is to determine the quality of the designed controllers when compared with standard approaches. For instance, the MPCC formulation is based on clothoid-based reference paths and predictions. Thus, the MPCC is evaluated on non-clothoid-based and clothoid-based reference paths. Furthermore, the EMPC formulation intends to give smoothness to the driving experience. Therefore, the EMPC is compared with a standard MPC to understand the effect that the smoothing term has on path tracking accuracy. Finally, the EMPC, the MPCC, and the standard MPC are compared with a PPC. Although being a simple controller, the PPC is widely used in industry due to its ease of implementation and satisfactory results.

This chapter is organized as follows. In Section 8.1, the reference paths used in the chapter are presented. In Section 8.2, the pure-pursuit controller is briefly

explained and the standard MPC, the MPCC and the EMPC are revisited. In Section 8.3, the performance of the MPCC and the EMPC is benchmarked against a PPC and standard MPC. Moreover, we compare the performance of the MPCC in both non-clothoid-based and clothoid-based reference paths. We provide a statistical analysis of both the deviation from the path and the curvature change rate of the controllers. In Section 8.4, the EMPC is compared with a PPC both deployed in a Scania construction truck, and the validity of the simulation environment is discussed by comparing experimental and simulation results. Section 8.5 summarizes the main results of the chapter.

8.1 Reference paths

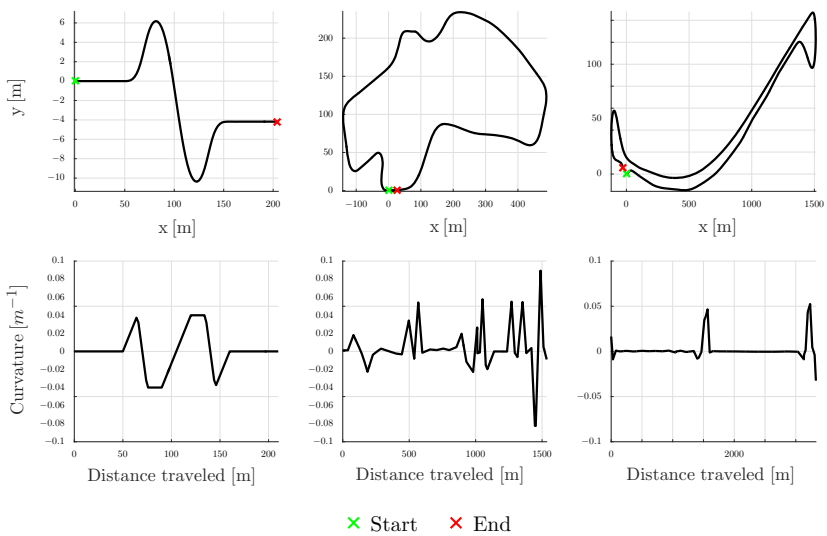


Figure 8.1: From left to right: double S-curve, precision track, and high-speed test track. Above are show the reference paths, while below are represented the reference curvatures. The double S-curve is an artificial reference path intended to analyze the controller properties. The precision track and the high-speed test track are real-world paths. The first one gives an idea of the challenges present in a mining situation, while the second is a typical highway situation.

This section introduces the reference paths on which the controllers developed in this thesis are evaluated and compared. These reference paths are depicted in Figure 8.1.

The double S-curve is constructed of nine clothoid segments. It is an artificial reference path that allows the analysis to be focused solely on the controller per-

formance. This is also the reason why the double S-curve and similar were used in Chapters 5,6, and 7 to illustrate the controllers performance. The double S-curve is only used in simulation.

Both in simulation and experimentally, the reference paths used are generated by recording a GPS trace using a RTK-GPS installed in the truck. The reference paths are recorded at Scania's test tracks facilities near Södertälje, Sweden. The paths are then subsampled and consist of a waypoint sequence, in which the waypoints are two meters apart. Without being further processed, these paths are referred to as non-clothoid-based paths. Also, clothoid-based versions of these paths are computed by oversampling the result from the clothoid-based path sparsification algorithm, presented in Chapter 4. We use a precision track, which is a challenging gravel road designed to perform autonomous mining truck tests, since it resembles a mining site. It is a narrow road with an approximate length of 1.5 km. It has sharp turns where consequently the vehicle speed can not be high. Therefore, it is the perfect scenario to evaluate the performance of control algorithms designed to be extremely accurate at low speeds. Also, to widen the scope of the designed controller applicability, a high-speed test track is also used. It consists of two long straights and one sharp U-turn. In one of the straights (0-1500 m) the speed limit is 70 km/h, while in the other straight (1500-3000 m) the speed limit is 90 km/h. The performance of the MPCC is compared on non-clothoid-based and clothoid-based reference path versions of the precision track and high-speed test track.

8.2 Controllers

In this section, we introduce the pure-pursuit controller (PPC) and revisit the standard MPC, the clothoid-based MPC (MPCC), and the economic MPC (EMPC). All the controllers described in this section are evaluated in simulation. The EMPC and the PPC are also evaluated experimentally.

Pure-pursuit controller

Let a reference path be constituted by a finite number of points $N > 1$ of the form $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}]^\intercal$, $i = 1, ..., N$, which represent the path Cartesian coordinates $x_i^{\text{ref}}, y_i^{\text{ref}}$. Also, the path coordinates and orientation are relative to the current vehicle position and orientation.

Let the desired curvature be the vehicle input. Figure 8.2 depicts an illustration of the pure-pursuit controller algorithm [55]. In fact, this method is inspired by the way humans drive. Humans tend to look some distance in front of the car and head towards that spot. The algorithm has only one user-defined parameter (i.e., look-ahead distance $L_{\text{ahead}} > 0$) with which the goal position is chosen. With the goal position defined, simple geometric relations are used to determine the curvature required, represented by κ in Figure 8.2, to move the vehicle from its current position to the goal position, denoted by $(0, 0)$ and by (x, y), respectively, in Figure 8.2.

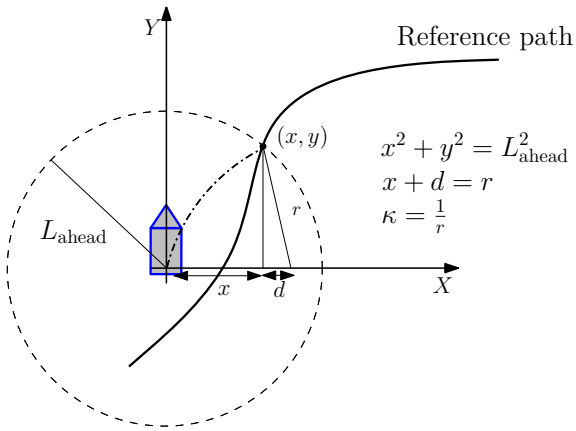


Figure 8.2: The pure-pursuit controller algorithm illustration (inspired from [55]). The controller has only one parameter, the look-ahead distance that gives, as illustrated, the target point. Then, the curvature from the vehicle ego position to the target point is computed using a simple geometric analysis.

The pure-pursuit controller is often applied due to its versatility and ease of implementation. However, a fine tuning is hard to accomplish and it is vehicle specific. Furthermore, the main parameter, the look-ahead distance, has a large influence on the path following accuracy, which can never obtain null deviation in curvy roads, resulting in a major drawback of the method.

Standard MPC

In this case, the reference path also includes the path orientation θ_i^{ref}, $i = 1, ..., N$, such that it is given in the form $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}, \theta_i^{\text{ref}}]^\intercal$.

Let $\mathbf{z}_i = [x_i, y_i, \theta_i]^\intercal$ be the vehicle states where index i ranges from 1 to H, and H is the MPC prediction horizon. Again, the control input to the vehicle is the curvature, $\mathbf{u}_i = \kappa_i$. Typically, a standard MPC is formulated as

$$\min_{\tilde{\mathbf{u}}_1,\ldots,\tilde{\mathbf{u}}_H} \quad \sum_{i=1}^{H} \tilde{\mathbf{z}}_i{}^\intercal \mathbf{Q} \tilde{\mathbf{z}}_i + \tilde{\mathbf{u}}_i^\intercal \mathbf{R} \tilde{\mathbf{u}}_i \tag{8.1a}$$

$$\text{s.t.} \quad \mathbf{z}_{i+1} = f_1(\mathbf{z}_i, \mathbf{u_i}), \tag{8.1b}$$

$$\mathbf{z}_0 = \mathbf{z}(s_0), \tag{8.1c}$$

$$\tilde{\mathbf{u}}_\mathbf{i} \in \mathbb{U}, \tag{8.1d}$$

where $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{z}_i^{\text{ref}}$, $\tilde{\mathbf{u}}_i = \mathbf{u}_i - \mathbf{u}_i^{\text{ref}}$, $\mathbf{Q} \in \mathbb{R}^{3\times3}$ and $\mathbf{R} \in \mathbb{R}$ are the state penalization and the input penalization matrices, respectively and they are generally chosen

to be diagonal and positive definite and $\mathbb{U}$ is the set of feasible inputs. The cost function (8.1a) penalizes deviations from the reference path, (8.1b) represents the kinematic vehicle model (3.6) that is used to predict the vehicle motion, (8.1c) is the initial condition, and (8.1d) describes the input constraints.

The vehicle model (3.6) is discretized and linearized around the reference path using the sample interval $\Delta s = vT_s$, where v is the current vehicle speed and $T_s = 200$ ms is the sampling rate of the prediction model (motivated by the results from [63]).

In the end, (8.1) is rewritten as a LTV-MPC and cast as a QP.

Clothoid-based MPC (MPCC)

In Chapter 4, a clothoid-based path sparsification method is introduced, which allows for describing a path through a small number of waypoints, the so called kink-points. For the MPCC, let the reference path be described by N kink-points containing position $x_i^{\text{ref}}, y_i^{\text{ref}}$, orientation θ_i^{ref}, κ_i^{ref}, and traveled distance s_i. Also, we have access to the clothoid model reference input $\mathbf{u}_i^{\text{ref}} = [c_i^{\text{ref}}, L_i^{\text{ref}}]^\intercal$, where c_i^{ref} and L_i^{ref} are the i-th clothoid curvature slope (constant by definition) and arc-length respectively.

As described in Chapter 5, the vehicle motion can be predicted using clothoids assuming that its curvature changes linearly with the traveled distance. Therefore, the vehicle input is $\mathbf{u}_i = [c_i, l_i]^\intercal$. We can formulate the MPCC as a standard MPC,

$$\min_{\tilde{\mathbf{u}}_1,\dots,\tilde{\mathbf{u}}_H} \quad \sum_{i=1}^{H} \tilde{\mathbf{z}}_i^\intercal \mathbf{Q} \tilde{\mathbf{z}}_i + \tilde{\mathbf{u}}_{i-1}^\intercal \mathbf{R} \tilde{\mathbf{u}}_{i-1} \tag{8.2a}$$

$$\text{s.t.} \quad \mathbf{z}_{i+1} = f_2(\mathbf{z}_i, \mathbf{u_i}), \tag{8.2b}$$

$$\mathbf{z}_0 = \mathbf{z}(s_0), \tag{8.2c}$$

$$\mathbf{u} \in \mathbb{U}, \tag{8.2d}$$

where H is the MPC prediction horizon, $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{z}_i^{\text{ref}}$, $\tilde{\mathbf{u}}_i = \mathbf{u}_i - \mathbf{u}_i^{\text{ref}}$ and $\mathbb{U}$ is the set of feasible inputs. The penalization matrices $\mathbf{Q} \in \mathbb{R}^{4\times4}$ and $\mathbf{R} \in \mathbb{R}^{2\times2}$ are diagonal positive definite and weight the state and the input cost, respectively. The vehicle model used in (8.2b) is a linearized version of (5.4), which makes a clear distinction between the MPCC and the standard MPC (8.1).

Again, (8.2) is rewritten as a LTV-MPC and cast as a QP.

Economic MPC (EMPC)

As before, let the vehicle be described solely by its position $\mathbf{z}_i = [x_i, y_i]^\intercal$, $i = 1, ..., H$ and $\mathbf{u}_i = \kappa_i$ be its desired curvature and input. Also, the path is only described by it Cartesian coordinates $\mathbf{z}_i^{\text{ref}} = [x_i^{\text{ref}}, y_i^{\text{ref}}]^\intercal$.

As described in Chapter 6, the EMPC formulation accounts for the operating cost of the system. We define the economic cost as the driving smoothness and comfort. These concepts are assumed to be achieved if the cost function includes a term that encourages predictions using piecewise linear curvature function. Thus, let $||\mathbf{D}_2\boldsymbol{\kappa}||_2^2 + \alpha||\mathbf{D}_1\boldsymbol{\kappa}||_2^2$ be the economic cost associated with the vehicle, where $\mathbf{D}_2 \in \mathbb{Z}^{(H-1)\times(H+1)}$ and $\mathbf{D}_1 \in \mathbb{Z}^{H\times(H+1)}$ are the matrix operators that calculate the second- and first-order differences of a vector. We formulate the EMPC as

$$\begin{aligned}
\min_{\mathbf{u},\boldsymbol{\Delta}} \quad & \mathbf{u}^\intercal\mathbf{R}\mathbf{u} + \boldsymbol{\Delta}^\intercal\boldsymbol{\Lambda}\boldsymbol{\Delta} && \text{(8.3a)}\\
\text{s. t.} \quad & |\mathbf{x} - \mathbf{x}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_x + \boldsymbol{\Delta}_{\boldsymbol{x}}, && \text{(8.3b)}\\
& |\mathbf{y} - \mathbf{y}_{\text{ref}}| \leq \boldsymbol{\varepsilon}_y + \boldsymbol{\Delta}_{\boldsymbol{y}}, && \text{(8.3c)}\\
& \mathbf{u} \in \mathbb{U}, && \text{(8.3d)}\\
& \boldsymbol{\Delta} \geq 0, && \text{(8.3e)}
\end{aligned}$$

where $\mathbf{R} = \mathbf{D}_2^\intercal\mathbf{D}_2 + \alpha\mathbf{D}_1^\intercal\mathbf{D}_1$. Moreover, $\alpha \in \mathbb{R}$ penalizes the curvature change rate magnitude, $\boldsymbol{\Lambda} \in \mathbb{R}^{2H\times 2H}$ is a diagonal matrix of the form $\text{diag}(\lambda_1, ..., \lambda_H, \lambda_1, ..., \lambda_H)$ that penalizes how much the solution can violate the constraints and $\mathbb{U}$ is the set of feasible inputs. The cost function (8.3a) is minimized with respect to the input, which is the vehicle desired curvature, and to a slack variable $\boldsymbol{\Delta} = \text{diag}(\boldsymbol{\Delta}_{\boldsymbol{x}}, \boldsymbol{\Delta}_{\boldsymbol{y}})$. This slack variable is included in the box constraints that limit the vehicle motion prediction (8.3b) and (8.3c) to avoid problem infeasibility.

Note that (8.3) is written in epigraph form. If $\boldsymbol{\varepsilon}_x = \boldsymbol{\varepsilon}_y = 0$, the optimization problem can be rewritten as

$$\begin{aligned}
\min_{\mathbf{u}} \quad & \mathbf{u}^\intercal\mathbf{R}\mathbf{u} + \tilde{\mathbf{z}}^\intercal\boldsymbol{\Lambda}\tilde{\mathbf{z}} && \text{(8.4a)}\\
\text{s. t.} \quad & \mathbf{u} \in \mathbb{U}, && \text{(8.4b)}
\end{aligned}$$

where $\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}_{\text{ref}}$ describes the deviations from the reference path.

Comparing the EMPC (8.4) with the standard MPC (8.1) we can highlight several similarities and differences. In the EMPC there is no penalization for deviations from the reference input $\mathbf{u}_{\text{ref}}$, while in the standard MPC it is present. However, in the EMPC there is an economic cost associated, which is absent in the standard MPC. Moreover, in both the EMPC and in the standard MPC there is a cost associated with deviations from the reference path.

Again, (8.4) is rewritten as an LTV-MPC and cast as a QP.

Table 8.1: Tuning of the controllers parameters for simulation evaluation.

Parameters / Controllers	PPC	MPC	MPCC	EMPC
Look ahead time, L_{ahead}	1.2 s			
Prediction horizon, H		10	10	10
Path sampling time, T_s		200 ms	200 ms	200 ms
Controller frequency	50 Hz	50 Hz	50 Hz	50 Hz
State pen. matrix, $\mathbf{Q}$		$\text{diag}(50, 50, 0.1)$	$\text{diag}(1, 1, 10, 10)$	
Input pen. matrix, $\mathbf{R}$		500	$\text{diag}(100, 1000)$	
Slack penalization, λ_i				200
First derivative pen., α				200
Box constraints sizes, ε_i				0

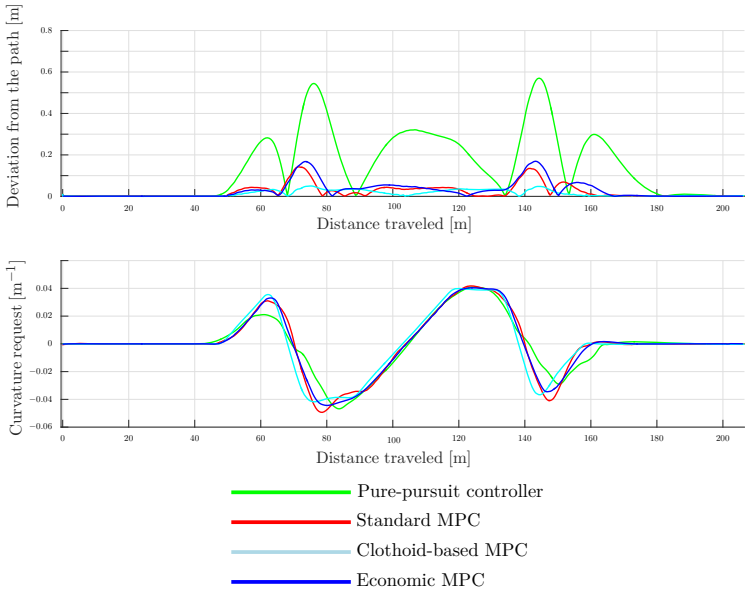
8.3 Simulation results

Experimental evaluation requires some level of prior controller testing and validation. Therefore, we set up a simulation environment in MATLAB/Simulink resembling as much as possible the real system with the same modules and interfaces as in the vehicle. The system and the controller diagrams are described in detail in Chapter 2. Our intention is to demonstrate that the simulation environment is sufficiently accurate to allow, for example, controller tuning before deploying it on the vehicle. Also, in simulation we can extend our analysis further than experimentally since we are not bounded by implementation issues and platform availability. We simulate the vehicle dynamics using the model described in Chapter 3. In this environment, we compare the pure-pursuit controller, the standard MPC, the MPCC and the EMPC.

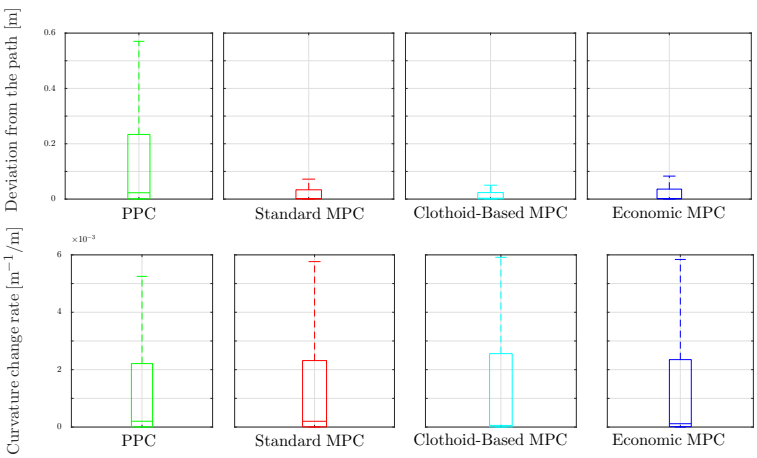
The simulation results are shown in Figures 8.3 to 8.8 for the three scenarios discussed in Section 8.1. Figures 8.3a to 8.8a represent the curvature requests from the controllers and their lateral deviation from the reference path. The results are summarized in Tables 8.2 and 8.3. Figures 8.3b to 8.8b provide a statistical analysis of the distance to the reference path and the curvature change rate. In the boxes that are depicted in these figures, the central mark is the median, the edges of the box are the 25th and 75th percentiles and the whiskers extend to the most extreme data points not considered outliers. The meaning of the boxes is detailed in Figure 8.6.

Table 8.1 summarizes the tuning of the controllers parameters for each controller. The parameters tuning are found mixing empirical analysis and desired controller behavior.

The double S-curve is performed at a vehicle speed of 5 m/s by all controllers. Analyzing Figure 8.3a, it is evident that the MPC strategies outperform the PPC. Also, all three MPC approaches have extraordinary performance with particular highlight to the MPCC that never deviates from the path more than 5 cm, see Table 8.2. As expected and desired, a clothoid-based reference path induces extremely good performance on a clothoid-based MPC, which has the same underlying princi-

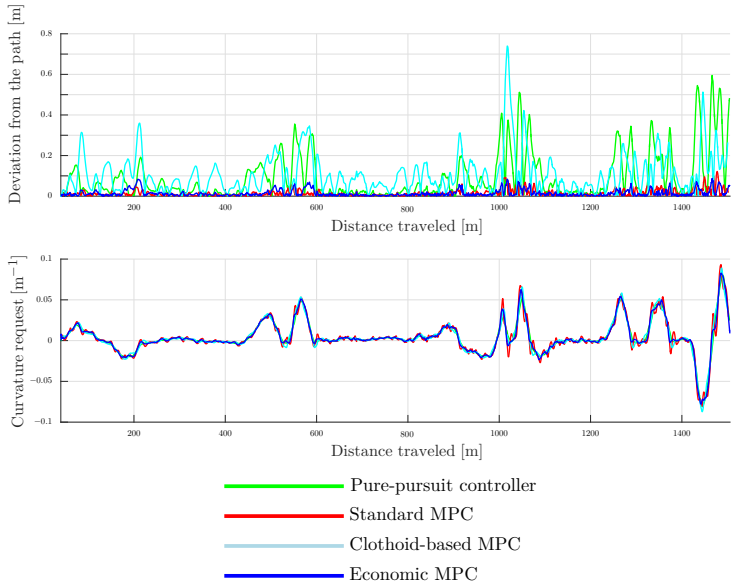


(a) Above, the deviation from the path of the controllers is shown. Below, the controllers curvature requests are depicted.

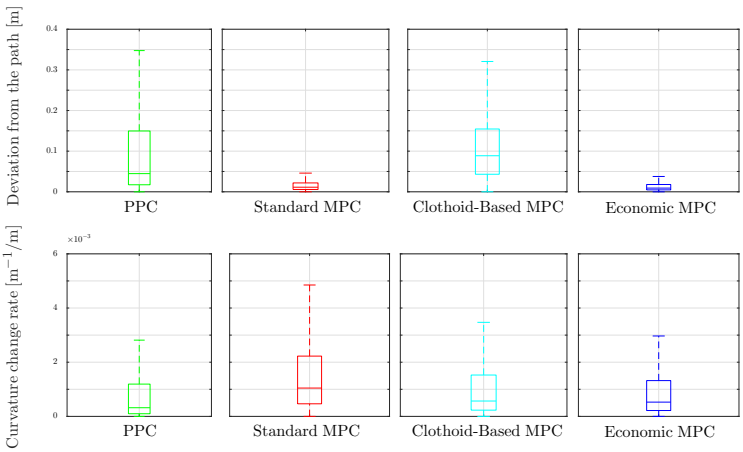


(b) Statistical analysis of the deviation from the path for all the controllers. The legend of the figure can be seen in Figure 8.6.

Figure 8.3: Simulation run of the PPC, MPC, MPCC and EMPC on the double S-curve shown in Figure 8.1. The controllers parameters are presented in Table 8.1.

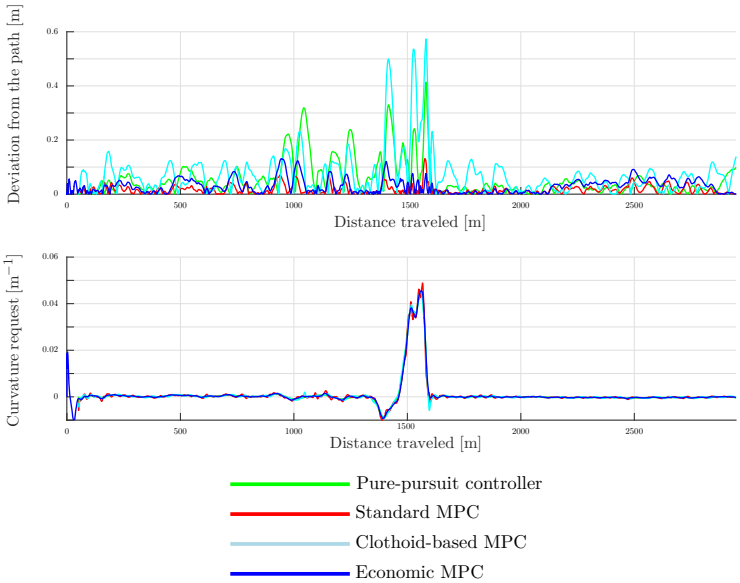


(a) Above is shown the deviation from the path of the controllers. Below, it is depicted the controllers curvature requests.

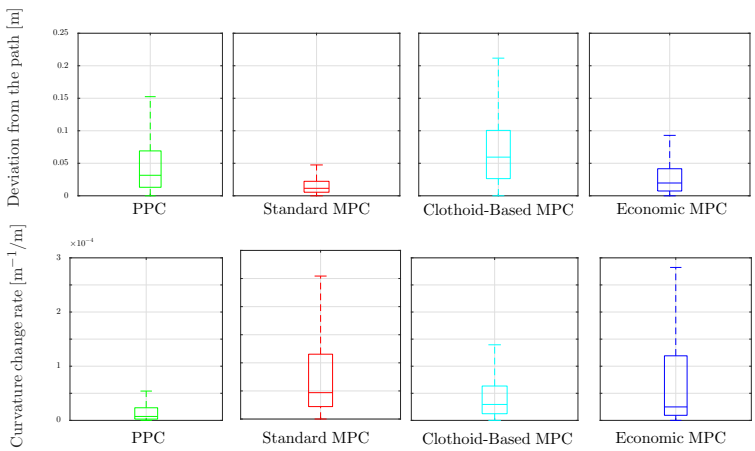


(b) Statistical analysis of the deviation from the path for all the controllers. The legend of the figure can be seen in Figure 8.6.

Figure 8.4: Simulation run of the PPC, MPC, MPCC and EMPC on the precision track shown in Figure 8.1. The controllers parameters are presented in Table 8.1.



(a) Above is shown the deviation from the path of the controllers. Below, it is depicted the controllers curvature requests.



(b) Statistical analysis of the deviation from the path for all the controllers. The legend of the figure can be seen in Figure 8.6.

Figure 8.5: Simulation run of the PPC, MPC, MPCC and EMPC on the high-speed test track shown in Figure 8.1. The controllers parameters are presented in Table 8.1.

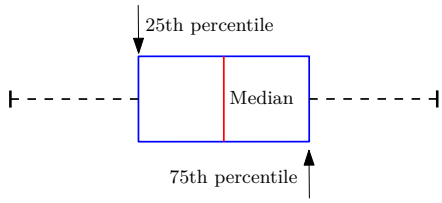


Figure 8.6: Box plot legend. The central mark is the median, the edges of the box are the 25th and 75th percentiles and the whiskers extend to the most extreme data points not considered outliers.

ples. Moreover, carefully interpreting the curvature change rate data in Figure 8.3b, we conclude that there is no significant difference between all the controllers in this case. In fact, the PPC is the smoothest controller, but since it is the less accurate controller it is not a good trade-off.

The conclusions are different when analyzing the other reference paths in Figures 8.4a and 8.5a. On the precision track, the vehicle speed is typically between 4 m/s and 10 m/s, while on the high-speed test track the speed can reach 25 m/s in one of the straights. Also, in the simulation runs with MPCC the vehicle does not exceed 10 m/s since the controller formulation assumes low speeds to avoid non-neglectable dynamics and inaccurate vehicle motion predictions (see Chapter 5). Since both the precision and the high-speed test tracks are not clothoid-based representations, the MPCC is not expected to perform as in a clothoid-based reference path. The need for a clothoid-based reference path can be seen as a major drawback of the controller. In this case, both the standard MPC and EMPC approaches outperform the PPC and the MPCC. To be able to draw conclusions about the performance of the controller, we also take into account the curvature change rate data in Figures 8.4b and 8.5b. Both the EMPC and the standard MPC approaches have similar and good performance, since they both have a low average deviation and a maximum deviation from the path below 20 cm, see Table 8.2. However, the standard MPC approach is more aggressive if we compare curvature request change rate. First of all, we can see that a pure-pursuit strategy is, for the scenarios considered, the smoothest controller, followed by the MPCC. In contrast, the standard MPC approach is the most aggressive but also typically one of the most accurate. This is expected from the standard MPC parameter tuning, which can be seen in Table 8.1. The controller can achieve smoother behavior, but at the cost of a significant loss of accuracy. For understandable benchmarking, it is interesting to have the best path tracking accuracy possible, which we are interested to maximize, and compare driving behaviors. Therefore, there is a clear tradeoff between accuracy and smoothness, which is challenging to balance. Nevertheless, the EMPC is able to effectively combine a smooth control action with an accurate path following.

Table 8.2: Controller deviation from the path comparison in simulation.

		Max. [m]	Average [m]	Std. deviation [m]
Double S-curve	PPC	0.57	0.12	0.15
	MPC	0.14	0.02	0.03
	MPCC	0.05	0.02	0.01
	EMPC	0.17	0.02	0.04
Precision track	PPC	0.59	0.10	0.12
	MPC	0.12	0.02	0.02
	MPCC	0.73	0.11	0.10
	EMPC	0.09	0.02	0.02
High-speed test track	PPC	0.41	0.06	0.08
	MPC	0.13	0.02	0.02
	MPCC[1]	0.57	0.08	0.09
	EMPC	0.13	0.03	0.02

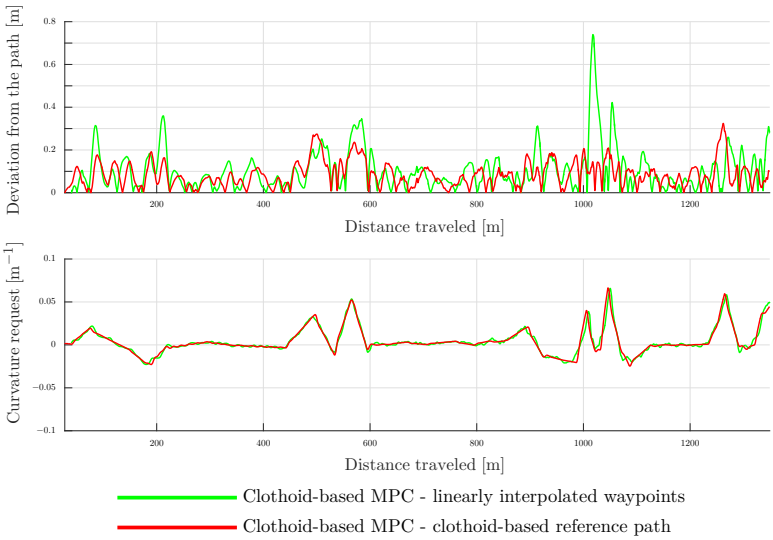
Table 8.3: Controller deviation from the path comparison in simulation. Comparison between non-clothoid-based and clothoid-based (C.B.) path representations.

		Max. [m]	Average [m]	Std. deviation [m]
Precision track	MPCC	0.73	0.11	0.10
C.B. precision track	MPCC	0.32	0.09	0.06
High-speed test track	MPCC[1]	0.57	0.08	0.09
C.B. high-speed test track	MPCC[1]	0.42	0.08	0.07

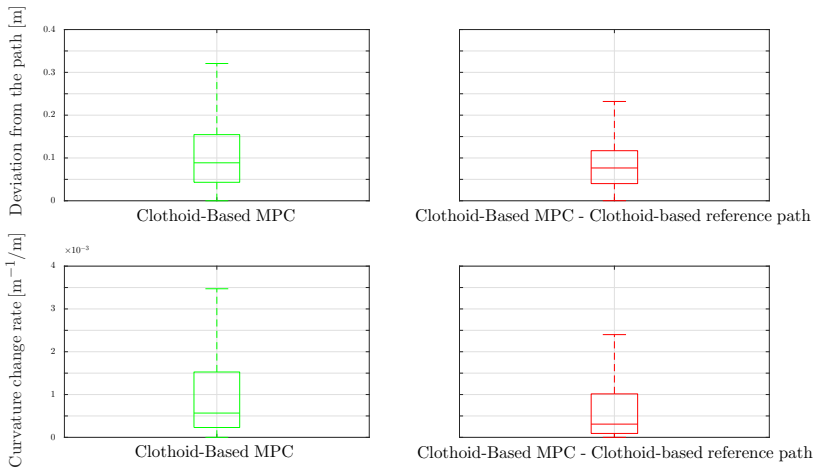
Clothoid-based *vs* non-clothoid-based path description

In the non-clothoid-based case, the MPCC does not have a good performance. Therefore, to have a fair comparison, we describe the precision track and the high-speed test track using a clothoid-based representation by oversampling the output of the clothoid-based path sparsification algorithm described in Chapter 4. The results are presented in Figures 8.7 and 8.8, and summarized in Table 8.3. Again, in the simulation runs with the MPCC controller the vehicle does not exceed a speed of 10 m/s. There is a clear improvement when using a clothoid-based representation. For instance, the maximum deviation is 56% less in the precision track and 26% less in the high-speed test track. Also, the average deviation is 18% less in the precision track. Although the MPCC performance is not as good as the standard MPC and the EMPC, we conclude that in the presence of clothoid-based reference path the MPCC has an acceptable performance.

[1] Maximum speed of 10 m/s.

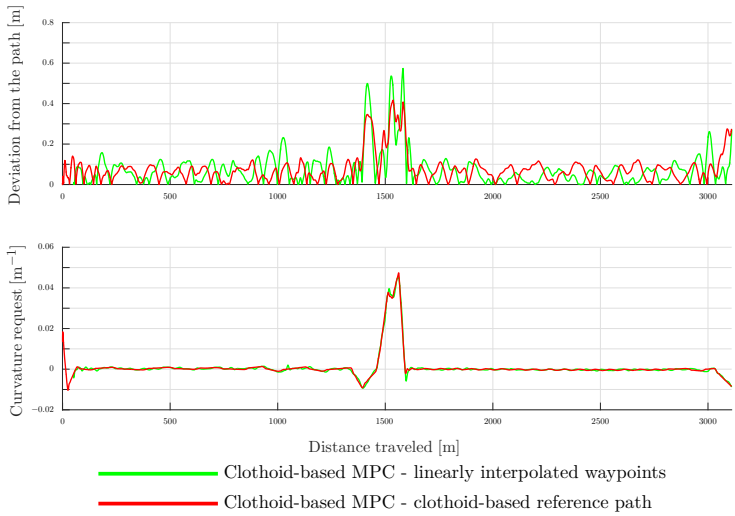


(a) Above is shown the deviation from the path of the controllers. Below, it is depicted the controllers curvature requests.

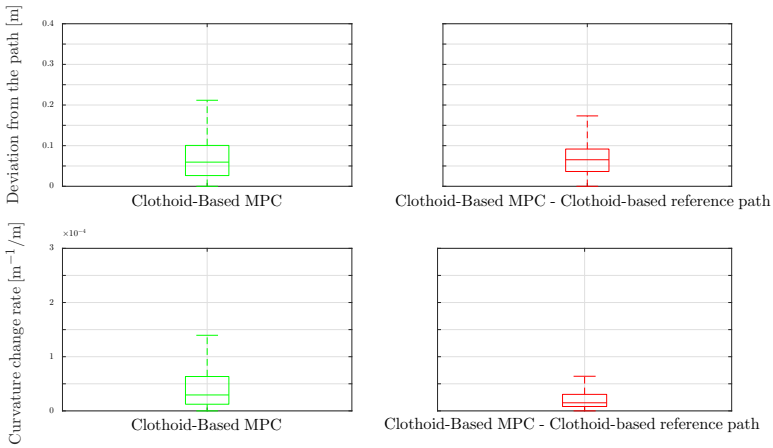


(b) Statistical analysis of the deviation from the path for the MPCC. The legend of the figure can be seen in Figure 8.6.

Figure 8.7: Simulation run of the MPCC on the precision track shown in Figure 8.1. The controllers parameters are presented in Table 8.1. The precision track is either a linear interpolation between waypoints or a clothoid-based reference path.



(a) Above is shown the deviation from the path of the controllers. Below, it is depicted the controllers curvature requests.



(b) Statistical analysis of the deviation from the path for the MPCC. The legend of the figure can be seen in Figure 8.6.

Figure 8.8: Simulation run of the MPCC on the high-speed test track shown in Figure 8.1. The controllers parameters are presented in Table 8.1. The high-speed test track is either a linear interpolation between waypoints or a clothoid-based reference path.

Figure 8.9: Modified Scania G480 construction truck used as experimental and research platform.

8.4 Experimental results

The truck used in the experimental evaluation is a modified Scania G480 construction truck and it is shown in Figure 8.9. The vehicle is equipped with a sensor platform and a servo motor for automated control of the steering column. We refer the interested read to Chapter 2 for a detailed description of the system and the controller architecture. To perform these experimental tests, a GPS track is previously recorded and the vehicle position and orientation is recorded. The GPS track is post-processed to include path curvature information and a speed profile. Then, the resultant waypoints are added to a database that is available to be queried at run time (i.e., an algorithm constantly determines if the vehicle is close to a GPS track previously recorded feeding the waypoints to the controller). Another option would be to use a local path planner. The controller implementation is not affected by the origin of the path (see Chapter 2). Therefore, the results described in this section are also applicable to this case, since the experiments intend to give the controller performance, in practice, when tracking a generic path. However, two fixed reference paths are chosen to allow for a proper comparison between the PPC and the EMPC. We choose to implement the EMPC on the truck, as it is the controller with the best performance in the simulation environment, while the PPC was already deployed and provided a good benchmark.

The experimental evaluation is performed on the precision track and the high-speed test track, see Figure 8.1. Figure 8.10 shows a video frame from the experiment performed at the precision track. The road is extremely narrow at some

Figure 8.10: Right front wheel of the truck while performing the precision track. The lack of space between the vehicle and the side of the road demonstrates how narrow the road is and how accurate the controller needs to be.

points, as at the position represented in Figure 8.10, where the vehicle has just little more space than its own width. In mining sites, the roads are expected to be similar to this one, which motivates this experiment and the crucial need for high accuracy. Figure 8.11 shows a video frame from the experiment done at the high-speed test track. At the time instant shown in the figure, the vehicle drives at 90 km/h, the maximum speed allowed for trucks. We intend to demonstrate the capabilities of the controller outside the mining site application as the high-speed test track replicates a highway situation.

The experimental results are shown in Figures 8.12 and 8.13. In Figures 8.12a and 8.13a, we compare the curvature requests from the PPC and the EMPC and their lateral deviation from the path. The results are summarized in the Table 8.5. We also provide a statistical analysis of the distance to path in Figures 8.12b and 8.13b. The boxes in the figures have the same meaning as in the simulation, see Figure 8.6.

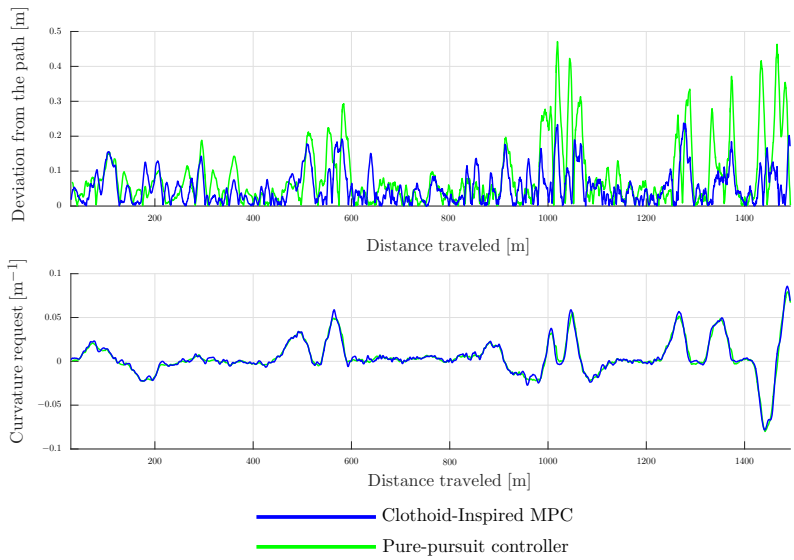
In both situations, it is clear that the EMPC outperforms the PPC (see Figures 8.12a and 8.13a). Also, we can see that the performance of the EMPC on the high-speed test track (see Table 8.5) is similar to the one expected from simulation results (see Table 8.2), which supports our claim of having a good simulation environment that can be used prior to experimental evaluation. This is not contra-

Figure 8.11: Performing one of the high-speed test track straights at 90 km/h. The "driver" keeps his hands close to the steering wheel for safety reasons.

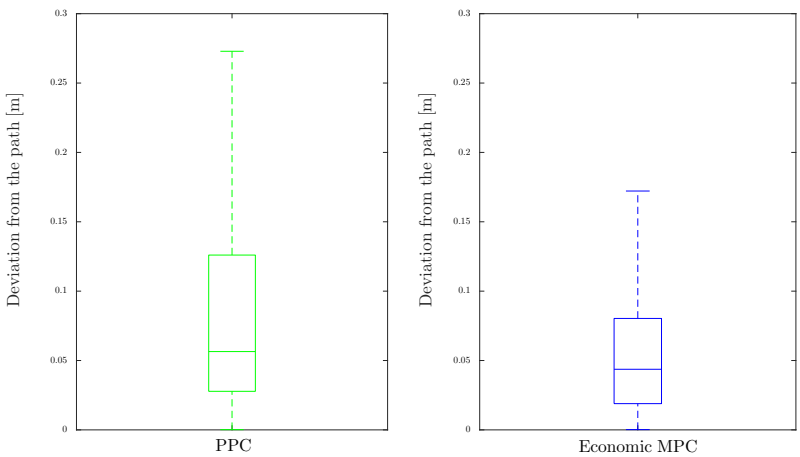
Table 8.4: Tuning of the controllers parameters for experimental evaluation.

Parameters / Controllers	PPC	EMPC
Look ahead time gap, L_{ahead}	1.2 s	
Prediction horizon, H		10
Path sampling time, T_s		200 ms
Controller frequency	50 Hz	50 Hz
Slack penalization, λ_i		20
First derivative penalization, α		9
Box constraints sizes, ε_i		0

dicted by the fact that the PPC performs quite differently at high speeds in reality when comparing with the simulation. Namely, it means that there are high-speed dynamics that are not considered in the simulation vehicle model, which was expected. Also, it means that the PPC is not able to handle these dynamics in a real vehicle, while an MPC approach is intrinsically more capable. At low-speeds, the performance of both controllers is good with an average deviation from the path of less than 10 cm (see Table 8.5). Still, the EMPC outperforms the PPC specially on the sharp curves. Note that the EMPC is able to run at both high and low speeds with the same parameter configuration and with a high performance score.

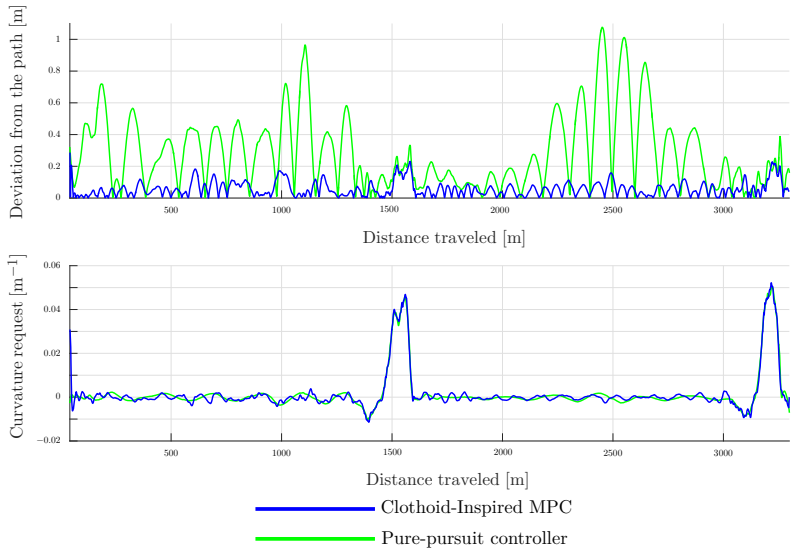


(a) Above, it is depicted the controllers curvature requests. Below is shown the deviation from the path of both controllers.

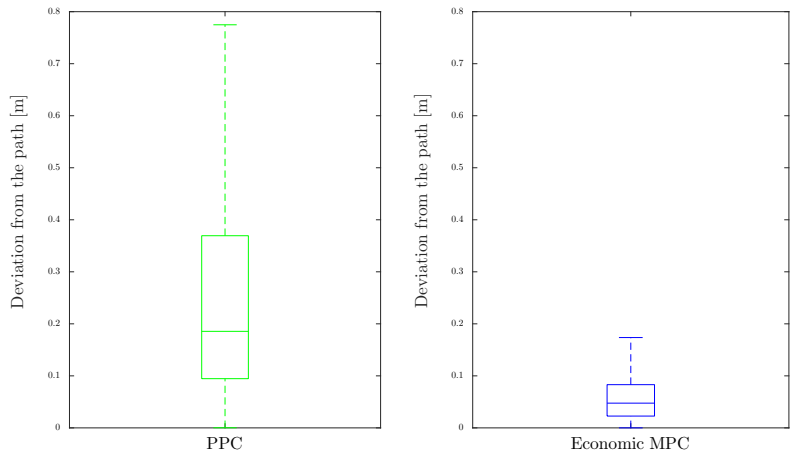


(b) Statistical analysis of the deviation from the path for both controllers. The legend of the figure can be seen in Figure 8.6.

Figure 8.12: Experimental run of the PPC and EMPC on the precision track shown in Figure 8.1. The controllers parameters are presented in Table 8.1.



(a) Above, it is depicted the controllers curvature requests. Below is shown the deviation from the path of the controllers.



(b) Statistical analysis of the deviation from the path for both controllers. The legend of the figure can be seen in Figure 8.6.

Figure 8.13: Experimental run of the PPC and EMPC on the high-speed test track shown in Figure 8.1. The controllers parameters are presented in Table 8.4.

Table 8.5: Controller deviation from the path comparison in real experiments.

		Max. [m]	Average [m]	Std. deviation [m]
Precision track	PPC	0.48	0.10	0.10
	EMPC	0.24	0.06	0.05
High-speed test track	PPC	1.08	0.25	0.21
	EMPC	0.29	0.06	0.05

8.5 Summary

In this chapter, we provided simulation and experimental results using the EMPC and the MPCC. We compared them to existing approaches, such as the pure-pursuit controller and the standard MPC. In simulation, we evaluated the performance of these four controllers both in clothoid-based and non-clothoid-based path representations. The performance was measured in terms of deviation from the path and curvature change rate (i.e., driving smoothness). The pure-pursuit controller is always the controller that provides the smoothest driving. However, it was generally the least accurate controller. Second, the MPCC has the best performance in the clothoid-based double S-curve, while when the reference paths are non-clothoid-based its performance is poor. Thus, the need for a clothoid-based path representation is a major drawback for the MPCC. We also compared the MPCC performance in non-clothoid-based and clothoid-based reference paths, and concluded that there was a clear improvement when using the clothoid-based representations. Nevertheless, the performance was still not as good as the performance of the EMPC and the standard MPC in the same non-clothoid-based paths. Nevertheless, the MPCC performs well in these specific conditions. Finally, the EMPC and the standard MPC are the most accurate controllers, as expected. In order to point out which one is the best we also analyzed the curvature change rate provided by the controllers and concluded that the EMPC is smoother than the standard MPC with a similar path accuracy. The EMPC formulation plays a crucial role in the driving smoothness, since unlike the standard MPC it encourages linear curvature predictions. Consequently, the EMPC is the controller that provides a better trade-off smooth driving with accurate path following. Experimentally, we compared the EMPC with a PPC, both deployed in a Scania construction truck. As predicted from the simulations, the EMPC clearly outperformed the PPC and its performance is outstanding since the maximum deviation from the path never exceeds 30 cm and in average is 6 cm. Also, we concluded that the tuning of the controllers parameters using the simulation environment provided a good approximation of the controller performance in reality.

Chapter 9

Conclusions and future work

Autonomous driving is a rapidly expanding research field. Even though autonomous vehicles are becoming available to the general public, there are still countless open problems to address. In this thesis, we proposed solutions for specific problems with good results. First, we described the modeling of the lateral and longitudinal dynamics vehicle for the development of an accurate simulation environment. Second, we proposed an algorithm that allows for a clothoid-based path description using a small number of points. Third, we addressed the problem of autonomous vehicle path following. We proposed two different controllers, combining the model predictive control framework and the properties of clothoids to provide smooth and comfortable driving. Fourth, we studied the problem of speed profiling and autonomous vehicle longitudinal control. Furthermore, we experimentally evaluated the performance of one of the controllers by deploying it in a Scania construction truck for mining applications.

In the following, we discuss the main conclusions and present open questions for future research directions.

9.1 Conclusions

This thesis focuses on the control design for autonomous driving of a heavy-duty vehicle for mining applications. We used the concept of clothoids and their properties to provide comfort and smoothness to the autonomous driving.

First, we presented the vehicle models used in the simulation environment and in the control design. The lateral dynamics of the vehicle model derived for simulation are described by a 4-axle bicycle model validated against real truck data. We identified the curvature request to steering angle dynamics of the truck and determined some vehicle parameters, such as the wheel angle ratio between the two front axles and the understeering gradient. Moreover, we described the longitudinal and the cruise controller model developed by Scania CV AB and we integrated them in the overall vehicle model. This model was found to be accurate enough to allow controller tuning in simulation. We also presented the vehicle model used

for control design. This is the kinematic model of a car-like vehicle that does not include any dynamics but is fundamentally valid at low speeds (i.e., when vehicle dynamics are negligible). Although simple, this model has proven useful because at high speeds the vehicle mostly drives straight and in curves the vehicle mostly drives slowly. In both cases, the lateral dynamics are considered negligible.

Second, we presented an algorithm that produces a sparse path representation, where a reduced number of clothoids is used to describe the reference path. This approach relies on a reweighed l_1-norm approximation of the l_0-norm. The clothoid-based path has few points where each point, the so called kink-point, has embedded the clothoid segment properties, resulting in a computationally efficient path description. The number of clothoids used to describe the path is dependent on the maximum deviation allowed between the original and the clothoid-based path. The smaller maximum allowed deviation, the greater the number of clothoids.

Third, we tackled the problem of autonomous lateral control. We presented a clothoid-based model predictive controller (MPCC) that addresses the problem of clothoid-based path following. This controller formulation is based on the fact that the path of a vehicle traveling at low speeds defines a segment of clothoids if the steering angle is chosen to vary piecewise linearly with the traveled distance. Therefore, instead of a vehicle model, we used the clothoid model to predict the vehicle motion. The main difference is that the inputs to this model are the curvature change rate and the arc-length of the clothoid. Since the corresponding equations are non-linear and do not have closed-form, we linearized the vehicle/clothoid predictions around the reference path, which is assumed to be clothoid-based and approximated the integrals with Riemman sums. We discussed the influence of this strategy in the prediction error upper-bound and concluded that it is not possible to use only the kink-points when predicting the vehicle motion. Still, we achieved good results when each waypoint was separated by 2 meters in a clothoid-based path representation.

Furthermore, we used the concept of economic model predictive controller (EMPC) for a vehicle lateral control. We formulated the controller using the EMPC framework, where the cost function explicitly minimizes an operating cost of the vehicle. In this case, the operating cost can be considered as a comfort or smooth driving costs and is related to the second and the first derivative of the curvature. The problem formulation was inspired by clothoids, which have a null curvature second derivative (i.e., linear varying curvature function). In addition, we constrained the optimization problem such that the vehicle always stays in a small box of size ϵ around the reference path. These are soft constraints and therefore, a slack variable that penalizes deviations from the path is also minimized in the cost function. We analyzed the influence of the controller parameter tuning and concluded that smaller constraint slack penalizations $\boldsymbol{\lambda}$ and larger box constraints sizes $\boldsymbol{\epsilon}$ contribute to a smoother driving. Moreover, the regularization parameter α (i.e., the curvature first derivative weighing, defines the curvature magnitude). For the MPCC and the EMPC we provided illustrative examples to evaluate their performance.

Fourth, we dealt with the problem of speed profiling and longitudinal control. The generation of an optimal speed profile was formulated as a convex optimization problem, where we minimized the deviations between the speed profile and the maximum speed allowed on the road. The maximum speed has two components, one is the maximum speed allowed by law (or user defined) and the other considers safety issues depending on the road geometry. Also, the speed profile can be tuned in order to produce more aggressive or smoother speed profiles with respect to the changes in acceleration. Furthermore, we developed a longitudinal controller that works in a receding-horizon fashion by solving a similar problem as the one formulated for the speed profiler. In this case, the longitudinal controller only takes into account a finite horizon ahead of the vehicle and receives the most recent vehicle state predictions. In doing so, the longitudinal controller produces feasible and safe speed requests for the low-level speed controller. We analyzed the influence of different path description densities and the influence of the acceleration penalization factor in the speed profile generation. We concluded that the sparser the path description, the further away the resultant speed is from the optimal. However, we could see that the path description does not need to be dense to achieve a good sub-optimal solution and is faster to compute. Also, by varying the acceleration penalization factor we could modify the smoothness of the speed profile. In this case, we have a clear trade-off between an aggressive speed profile, which tends to deviate as little as possible from the maximum speed allowed, or a smooth speed profile that provides smaller accelerations. We presented simulation results combining the generation of a speed profile for a single S-curve and the longitudinal controller. We concluded that when gear changes have no impact on the vehicle acceleration, the controller is able to correctly track the speed profile. Moreover, we exemplified the effect of an unexpected event in the longitudinal control. Since the lateral controller sends the most recent vehicle predictions, the longitudinal controller can adapt the speed or acceleration requests depending on those predictions without necessarily tracking the original speed profile.

Finally, we presented an experimental benchmarking of the lateral controllers. We compared through simulations the MPCC and the EMPC with a standard MPC and a pure-pursuit controller (PPC). Also, we deployed the EMPC on a real Scania construction truck and compared it with a PPC. The performance of the controllers was evaluated in terms of path following accuracy and curvature request change rate. For that, we provided a statistical analysis of the data. We concluded that the PPC is the smoothest controller but is also the least accurate one. Furthermore, we showed that the MPCC has great performance in clothoid-based reference paths, while its performance in non-clothoid-based paths is poor. The need of a clothoid-reference path is a drawback in the controller. Moreover, the EMPC and the standard MPC had similar path tracking accuracy performances. However, the EMPC was smoother than the standard MPC, which was intended from its formulation. In the end, we showed that the EMPC has an outstanding performance in real experiments with an average deviation from the path of 6 cm and never exceeding 30 cm in challenging tracks, in both high and low speeds.

9.2 Future work

In this section we discuss possible future work directions on control design for autonomous driving.

Extend the experimental benchmarking

Experimental validation is one of the most challenging and important tasks of control design. Although the controller implementation process is time consuming, it is a rewarding and a great learning experience. Therefore, the completion of our experimental benchmarking is necessary with the implementation in practice of the MPCC and the standard MPC. This would allow a fruitful comparison with EMPC, MPCC and other controllers that may be developed in the future.

Analyze controller performance (stability and convergence)

The theoretical analysis in control design is one of the most important aspects. Thus, we would like to analytically demonstrate stability and convergence properties of the designed controllers. In doing so, it is necessary to deepen our knowledge about the controllers properties and their performance limits to be able to get the most out of them.

Improve the longitudinal control

Currently, the longitudinal controller is a reference generator for the cruise controller. However, there are no guarantees of speed profile tracking without knowing the underlying cruise controller model, since there is no direct way of controlling the vehicle acceleration without controlling directly the engine torque. One way to improve the control design is to include a control-oriented cruise controller model in the longitudinal controller to improve the vehicle speed predictions and the speed profile tracking. Also, since the controller does not include any vehicle longitudinal model, it is not possible to correctly predict the vehicle speed in the case of substantial road slope. Thus, the development of a control-oriented longitudinal model is also necessary. Another way is to develop a new cruise-controller that would directly translate the speed profile, given as input, into engine torques.

Develop algorithms for model uncertainties robustness and model parameters online tuning

So far, the utilization of a simple model has proven to be sufficient for a good controller performance. However, the controller still requires a significant amount of tuning and the prediction accuracy can be further improved. For this reason, we will consider the development of a procedure to systematically tune the parameters of the controller, possibly online while driving. One possible, interesting approach

would be to combine machine learning techniques with model predictive control to adapt the controller or model parameters depending on the environment. Also, it would be engaging to extend the formulation in the light of stochastic MPC to mitigate the influence of model uncertainties in the controller performance.

Abridge the gap between planning and control

The control design described in this thesis assumed the existence of an upper layer responsible for planning. Obstacles are taken into account and a local path is fed to the controller. As of today, the controller assumes that the received path is obstacle free and that the path planner is in charge of replanning in case of a sudden environment change. An interesting research direction is the combination between planning and control. In emergency situations, in fact, there might be no time for replanning in an upper layer. Therefore, the controller should be designed such to be aware of the environment and act accordingly to the different situations.

Explore extreme driving situations

In a mining scenario, roads are slippery, have high slopes and banking angles, and are full of debris. Also, the vehicle is subject to different loads that influence the driving experience. A simple model is not able to reproduce the vehicle motion is these cases. Therefore, there is a need to investigate and design more complex vehicle models providing better motion predictions in more elaborated scenarios.

Acronyms

ADAS	Advanced driver assistance system
AFS	Active front steering
CL-RRT	Closed-loop RRT
EKF	Extended Kalman filter
EMPC	Economic MPC
FFI	Fordonsstrategisk forskning och innovation
GPS	Global positioning system
IMU	Inertial measurement units
LIDAR	Light detection and ranging
LTV	Linear-time varying
MPC	Model predictive controller
MPCC	Clothoid-based MPC
NHTSA	National highway traffic safety administration
NMPC	Nonlinear MPC
PPC	Pure-pursuit controller
QP	Quadratic program
RADAR	Radio detection and ranging
RRT	Rapidly-exploring random trees
RTK-GPS	Real-time kinematic GPS
V2V	Vehicle-to-vehicle communication
V2I	Vehicle-to-infrastructure communication

Bibliography

[1] European Road Transport Research Advisory Council. Multi-annual implementation plan for Horizon 2020, 2013.

[2] European Commission. Roadmap to a single european transport area. Electronic, 2011.

[3] World Health Organization. Global status report on road safety: Time for action, Geneva, 2009.

[4] National Highway Traffic Safety Administration. Critical reasons for crashes investigated in the national motor vehicle crash causation, 2015. USA.

[5] J. Marshall, T. Barfoot, and J. Larsson. Autonomous underground tramming for center-articulated vehicles. *Journal of Field Robotics*, 25(6-7):400–421, 2008.

[6] S. Scheding, G. Dissanayake, E. Nebot, and H. Durrant-Whyte. An experiment in autonomous navigation of an underground mining vehicle. *IEEE Transactions on Robotics and Automation*, 15(1):85–95, 1999.

[7] L. Thrybom, J. Neander, E. Hansen, and K. Landernas. Future challenges of positioning in underground mines. In *Proceedings of the IFAC Conference on Embedded Systems, Computer Intelligence and Telematics*, pages 222–226, June 2015.

[8] H. Andreasson, A. Bouguerra, M. Cirillo, D. Dimitrov, D. Driankov, L. Karlsson, A. Lilienthal, F. Pecora, J. Saarinen, and A. Sherikov. Autonomous transport vehicles: where we are and what is missing. *IEEE Robotics & Automation Magazine*, 22(1):64–75, 2015.

[9] D. Bellamy and L. Pravica. Assessing the impact of driverless haul trucks in Australian surface mining. *Resources Policy*, 36(2):149–158, 2011.

[10] R. Wallace and P. Silberg. Self-driving cars: The next revolution. Technical report, Center for Automotive Research, Transportation Systems Analysis Group, 2012.

[11] National Highway Traffic Safety Administration. U. S. department of transportation releases policy on automated vehicle development, May 2013.

[12] M. Mchugh. Tesla's cars now drive themselves, kinda. *Wired*, 2015.

[13] R. Ferlis. The dream of an automated highway. Technical report, Federal Highway Administration - U. S. Department of Transportation, 2007.

[14] P. Ross. Robot, you can drive my car. *IEEE Spectrum*, 51(6):60–90, 2014.

[15] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer. Vision and navigation for the carnegie-mellon navlab. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):362–373, 1988.

[16] E. Dickmanns, R. Behringer, D. Dickmanns, T. Hildebrandt, M. Maurer, F. Thomanek, and J. Schiehlen. The seeing passenger car VaMoRs-P. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 68–73, October 1994.

[17] P. Varaiya. Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, 38(2):195–207, 1993.

[18] J. Hedrick, M. Tomizuka, and P. Varaiya. Control issues in automated highway systems. *IEEE Control Systems*, 14(6):21–32, 1994.

[19] S. Shladover, C. Desoer, J. Hedrick, M. Tomizuka, J. Walrand, W. Zhang, D. McMahon, H. Peng, S. Sheikholeslam, and N. McKeown. Automated vehicle control developments in the path program. *IEEE Transactions on Vehicular Technology*, 40(1):114–130, 1991.

[20] K. Chang, W. Li, P. Devlin, A. Shaikhbahai, P. Varaiya, J. Hedrick, D. McMahon, V. Narendran, D. Swaroop, and J. Olds. Experimentation with a vehicle platoon control system. In *Proceedings of Vehicle Navigation and Information Systems Conference*, volume 2, pages 1117–1124, October 1991.

[21] A. Assad, J. Mårtensson, and K. Johansson. An experimental study on the fuel reduction potential of heavy duty vehicle platooning. *Control Engineering Practice*, 38:11–25, 2015.

[22] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney. Stanley: The robot that won the DARPA grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.

[23] C. Urmson, J. Bagnell, C. Baker, M. Hebert, A. Kelly, R. Rajkumar, P. Rybski, S. Scherer, R. Simmons, and S. Singh. Tartan racing: A multi-modal approach to the DARPA urban challenge. Technical report, Carnegie Mellon University, 2007.

[24] J. Mårtensson, A. Alam, S. Behere, M. Khan, J. Kjellberg, K. Liang, H. Pettersson, and D. Sundman. The development of a cooperative heavy-duty vehicle for the GCDC 2011: Team Scoop. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1033–1049, 2012.

[25] TNO, Idiada TU/e, and Viktoria Swedish ICT. Grand cooperative driving challenge. Homepage: http://www.gcdc.net/en/, 2016.

[26] A. Broggi, P. Cerri, S. Debattisti, M.C. Laghi, P. Medici, M. Panciroli, and A. Prioletti. PROUD - public road urban driverless test: Architecture and results. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 648–654, June 2014.

[27] T. Nothdurft, P. Hecker, S. Ohl, F. Saust, M. Maurer, A. Reschka, and J. Bohmer. Stadtpilot: First fully autonomous test drives in urban traffic. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 919–924, October 2011.

[28] P. Cerri, G. Soprani, P. Zani, J. Choi, J. Lee, D. Kim, K. Yi, and A. Broggi. Computer vision at the Hyundai autonomous challenge. In *Proceedings of the International IEEE Intelligent Transportation Systems Conference*, pages 777–783, October 2011.

[29] J. Funke, P. Theodosis, R. Hindiyeh, G. Stanek, K. Kritatakirana, C. Gerdes, D. Langer, M. Hernandez, B. Muller-Bessler, and B. Huhnke. Up to the limits: Autonomous Audi TTS. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 541–547, June 2012.

[30] M. Harris. Documents confirm Apple is building self-driving car. *The Guardian*, 2015.

[31] J. Markoff. Google cars drive themselves, in traffic. *New York Times*, 2010. October 9.

[32] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, and C. Keller. Making Bertha drive 2014 - An autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6(2):8–20, 2014.

[33] C. Ziegler. Volvo will run a public test of self-driving cars with 100 real people in 2017. *The Verge*, 2015.

[34] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[35] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebl, and F. Hundelshausen. Team AnnieWAY's autonomous system for the 2007 DARPA urban challenge. *Journal of Field Robotics*, 25(9):615–639, 2008.

[36] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun. Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597, 2008.

[37] K. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proceedings of the IEEE Conference on Robotics and Automation*, volume 2, May 2010.

[38] M. Darms, P. Rybski, and C. Urmson. Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 1197–1202, June 2008.

[39] M. Enzweiler and D. Gavrila. A multilevel mixture-of-experts framework for pedestrian classification. *IEEE Transactions on Image Processing*, 20(10):2967–2979, 2011.

[40] F. Lindner, U. Kressel, and S. Kaelberer. Robust recognition of traffic signals. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 49–53, June 2004.

[41] E. Ward and J. Folkesson. Multi-classification of driver intentions in yielding scenarios. In *Proceedings of the International IEEE Intelligent Transportation Systems Conference*, pages 678–685, September 2015.

[42] S. Lefèvre, D. Vasquez, and C. Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *Robomech Journal*, 1(1):1, 2014.

[43] M. Schreiber, C. Knoppel, and U. Franke. Laneloc: Lane marking based localization using highly accurate maps. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 449–454, June 2013.

[44] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, and C. Geyer. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[45] A. Marzinotto, M. Colledanchise, C. Smith, and P. Ogren. Towards a unified behavior trees framework for robot control. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 5420–5427, June 2014.

[46] C. Lim, R. Baumgarten, and S. Colton. Evolving behaviour trees for the commercial game defcon. In *Applications of evolutionary computation*, pages 100–110. Springer, 2010.

[47] S. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, Iowa State University, 1998.

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

[49] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Kolter, D. Langer, O. Pink, and V. Pratt. Towards fully autonomous driving: Systems and algorithms. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 163–168, June 2011.

[50] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How, and G. Fiore. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, 2009.

[51] N. Evestedt, D. Axehill, M. Trincavelli, and F. Gustafsson. Sampling recovery for closed loop rapidly expanding random tree using brake profile regeneration. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 101–106, June 2015.

[52] R. Attia, R. Orjuela, and M. Basset. Combined longitudinal and lateral control for automated vehicle guidance. *Vehicle System Dynamics*, 52(2):261–279, 2014.

[53] M. Sugeno and M. Nishida. Fuzzy control of model car. *Fuzzy sets and systems*, 16(2):103–113, 1985.

[54] A. De Luca, G. Oriolo, and C. Samson. *Feedback control of a nonholonomic car-like robot*, volume 229, chapter 4, pages 171–253. Springer Berlin Heidelberg, 1998.

[55] R. Coulter. Implementation of the pure-pursuit path tracking algorithm. Technical report, Carnegie Mellon University, 1992.

[56] A. Bemporad. Model predictive control design: New trends and tools. In *Proceedings of the IEEE Annual Conference on Decision and Control*, pages 6678–6683, December 2006.

[57] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[58] C. Garcia, D. Prett, and M. Morari. Model predictive control: theory and practice - a survey. *Automatica*, 25(3):335–348, 1989.

[59] S. Di Cairano, H. Tseng, D. Bernardini, and A. Bemporad. Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain. *IEEE Transactions on Control Systems Technology*, 21(4):1236–1248, 2013.

[60] P. Falcone, E. Tseng, F. Borrelli, J. Asgari, and D. Hrovat. MPC-based yaw and lateral stabilisation via active front steering and braking. *Vehicle System Dynamics*, 46(S1):611–628, 2008.

[61] F. Borrelli, P. Falcone, and T. Keviczky. MPC-based approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*, 3(2):265–291, 2005.

[62] P. Falcone, F. Borrelli, E. Tseng, J. Asgari, and D. Hrovat. A hierarchical model predictive control framework for autonomous ground vehicles. In *Proceedings of the American Control Conference*, pages 3719–3724, June 2008.

[63] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 1094–1099, June 2015.

[64] P. Falcone, F. Borrelli, H. Tseng, J. Asgari, and D. Hrovat. Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation. *International journal of robust and nonlinear control*, 18(8):862–875, 2008.

[65] V. Turri, A. Carvalho, E. Tseng, K. Johansson, and F. Borrelli. Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads. In *Proceedings of the International IEEE Intelligent Transportation Systems Conference*, pages 378–383, October 2013.

[66] C. Beal and J. Gerdes. Model predictive control for vehicle stabilization at the limits of handling. *IEEE Transactions on Control Systems Technology*, 21(4):1258–1269, 2013.

[67] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[68] A. Carvalho, Y. Gao, S. Lefevre, and F. Borrelli. Stochastic predictive control of autonomous vehicles in uncertain environments. In *Proceedings of the International Symposium on Advanced Vehicle Control*, September 2014.

[69] E. Velenis and P. Tsiotras. Optimal velocity profile generation for given acceleration limits: Theoretical analysis. In *Proceedings of the American Control Conference*, pages 1478 – 1483, June 2005.

[70] E. Velenis and P. Tsiotras. Optimal velocity profile generation for given acceleration limits: The half-car model case. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, pages 361 – 366, June 2005.

[71] E. Velenis and P. Tsiotras. Optimal velocity profile generation for given acceleration limits: Receding horizon implementation. In *Proceedings of the American Control Conference*, pages 2147 – 2152, June 2005.

[72] S. Jain, P. Tsiotras, and E. Velenis. Optimal feedback velocity profile generation for a vehicle with given acceleration limits: A level set implementation. In *Proceedings of the Mediterranean Conference on Control and Automation*, pages 451 – 456, June 2008.

[73] M. Lepetič, G. Klančar, I. Škrjanc, D. Matko, and B. Potočnik. Time optimal path planning considering acceleration limits. *Robotics and Autonomous Systems*, 45:199–210, 2003.

[74] A. Rucco, G. Notarstefano, and J. Hauser. Computing minimum lap-time trajectories for a single-track car with load transfer. In *Proceedings of the IEEE Annual Conference on Decision and Control*, pages 6321 – 6326, December 2012.

[75] F. Mensing, R. Trigui, and E. Bideaux. Vehicle trajectory optimization for application in eco-driving. In *Proceedings of the IEEE Vehicle Power and Propulsion Conference*, pages 1 – 6, September 2011.

[76] J. Villagra, V. Milanés, J. Pérez, and J. Godoy. Smooth path and speed planning for an automated public transport vehicle. *Robotics and Autonomous Systems*, 2012.

[77] A. Talbot. *The railway transition spiral*. Engineering News Publishing Company, 1904.

[78] W. Kühn. *Fundamentals of road design*, volume 20. WIT Press, 2013.

[79] H. Marzbani, R. Jazar, and M. Fard. Better road design using clothoids. In *Sustainable Automotive Technologies 2014*, pages 25–40. Springer International Publishing, 2015.

[80] L. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):497–516, 1957.

[81] J. Laumond, P. Jacobs, M. Taix, and R. Murray. A motion planner for nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 10(5):577–593, 1994.

[82] Y. Kanayama and B. Hartman. Smooth local path planning for autonomous vehicles. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1265–1270, May 1989.

[83] K. Komoriya and K. Tanie. Trajectory design and control of a wheel-type mobile robot using b-spline curve. In *Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems*, pages 398–405, September 1989.

[84] W. Nelson. Continuous-curvature paths for autonomous vehicles. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1260–1264, May 1989.

[85] A. Scheuer and T. Fraichard. Planning continuous-curvature paths for car-like robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1304–1311, November 1996.

[86] A. Kirchner and T. Heinrich. Model based detection of road boundaries with a laser scanner. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 93–98, June 1998.

[87] A. Takahashi and Y. Ninomiya. Model-based lane recognition. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 201–206, September 1996.

[88] J. Goldbeck, B. Hürtgen, S. Ernst, and L. Kelch. Lane following combining vision and DGPS. *Image and Vision Computing*, 18(5):425–433, 2000.

[89] M. Heald. Rational approximations for the fresnel integrals. *Mathematics of Computation*, 44:459–461, 1985.

[90] J. Richalet, J. Testud, and J Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14(5):413–428, 1978.

[91] P. Falcone. *Nonlinear model predictive control for autonomous vehicles*. PhD thesis, University of Sannio, Benevento, June 2007.

[92] J. Maciejowski. *Predictive control with constraints*. Pearson education, 2002.

[93] J. Rawlings and D. Mayne. *Model predictive control: theory and design*. Nob Hill Publishing, 2009.

[94] F. Borrelli, A. Bemporad, and M. Morari. *Predictive control for linear and hybrid systems*. 2015.

[95] F. Borrelli. *Constrained optimal control of linear and hybrid systems*, volume 290. Springer, 2003.

[96] U. Maeder and M. Morari. Offset-free reference tracking with model predictive control. *Automatica*, 46(9):1469–1476, 2010.

[97] G. Pannocchia and A. Bemporad. Combined design of disturbance model and observer for offset-free model predictive control. *IEEE Transactions on Automatic Control*, 52(6):1048–1053, 2007.

[98] J. Rawlings, D. Angeli, and C. Bates. Fundamentals of economic model predictive control. In *Proceedings of the IEEE Annual Conference on Decision and Control*, pages 3851–3861, December 2012.

[99] J. Ma, J. Qin, T. Salsbury, and P. Xu. Demand reduction in building energy systems based on economic model predictive control. *Chemical Engineering Science*, 67(1):92–100, 2012.

[100] T. Hovgaard, K. Edlund, and J. Jorgensen. Economic MPC for power management in the smartgrid. In *Proceedings of the European Symposium on Computer Aided Process Engineering*, volume 29, pages 1839–1843, May 2011.

[101] H. Pacejka. *Tyre and vehicle dynamics*. Elsevier, 2nd edition, 2005.

[102] MSC Software Corporation. *ADAMS Car documentation*, 2015.

[103] A. Alam. *Fuel-efficient heavy-duty vehicle platooning*. PhD thesis, KTH Royal Institute of Technology, Stockholm, 2014.

[104] J. McCrae and K. Singh. Sketch-based interfaces and modeling (sbim): Sketching piecewise clothoid curves. *Computers and Graphics*, 33(4):452–461, 2009.

[105] E. Bertolazzi and M. Frego. Fast and accurate clothoid fitting, September 2012.

[106] D. Ge, X. Jiang, and Y. Ye. A note on the complexity of l_p minimization. *Mathematical programming*, 129(2):285–299, 2011.

[107] E. J. Candes, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

[108] A. Belyaev. A note on invariant three-point curvature approximations. Technical report, Heriot-Watt University, 1999.

[109] P. Davis and P. Rabinowitz. *Methods of numerical integration*. Courier Corporation, 2007.

[110] J. Mattingley and S. Boyd. CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, 13:1–27, 2012.

[111] P. Sahlholm, H. Jansson, M. Östman, and K. Johansson. Automated speed selection for heavy-duty vehicles. Technical report, Scania CV AB and KTH Royal Institute of Technology, 2007.

[112] K. Lundahl, C. Lee, E. Frisk, and L. Nielsen. Analyzing rollover indices for critical truck maneuvers. *SAE International Journal of Commercial Vehicles*, 8(1):189–196, 2015.

Appendix A

Quadratic problem formulation of MPCC

It is possible to formulate the MPC problem (5.3) as a QP. We introduce the following vectors

$$\tilde{\mathbf{Z}}_{i+1} = \begin{bmatrix} \tilde{\mathbf{z}}_{i+1|i} \\ \tilde{\mathbf{z}}_{i+2|i} \\ \vdots \\ \tilde{\mathbf{z}}_{i+H|i} \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{U}}_i = \begin{bmatrix} \tilde{\mathbf{u}}_{i|i} \\ \tilde{\mathbf{u}}_{i+1|i} \\ \vdots \\ \tilde{\mathbf{u}}_{i+H-1|i} \end{bmatrix}, \tag{A.1}$$

where the notation $\tilde{\mathbf{z}}_{j|i}$ indicates the estimated state error at instant j predicted at instant i, and $\tilde{\mathbf{u}}_{j|i}$ indicates the optimal input error at instant j predicted at instant i.

Thus, it is possible to write the MPC cost function (5.3a) as

$$\tilde{\mathbf{Z}}_{i+1}^{\intercal} \bar{\mathbf{Q}} \tilde{\mathbf{Z}}_{i+1} + \tilde{\mathbf{U}}_i^{\intercal} \bar{\mathbf{R}} \tilde{\mathbf{U}}_i, \tag{A.2}$$

with $\bar{\mathbf{Q}} = \text{diag}(\mathbf{Q}, \ldots, \mathbf{Q})$ and $\bar{\mathbf{R}} = \text{diag}(\mathbf{R}, \ldots, \mathbf{R})$.

Furthermore, (5.5) is rewritten as a function of the initial state error $\tilde{\mathbf{z}}_{i|i}$ and the predicted input vector $\tilde{\mathbf{U}}_i$,

$$\tilde{\mathbf{Z}}_{i+1} = \tilde{\mathbf{A}}_i \tilde{\mathbf{z}}_{i|i} + \tilde{\mathbf{B}}_i \tilde{\mathbf{U}}_i, \tag{A.3}$$

where

$$\tilde{\mathbf{A}}_i = \begin{bmatrix} \mathbf{A}_i \\ \mathbf{A}_{i+1}\mathbf{A}_i \\ \vdots \\ \mathbf{A}_{H-1} \ldots \mathbf{A}_{i+1}\mathbf{A}_i \\ \mathbf{A}_H \mathbf{A}_{H-1} \ldots \mathbf{A}_{i+1}\mathbf{A}_i \end{bmatrix} \tag{A.4}$$

and

$$\tilde{\mathbf{B}}_i = \begin{bmatrix} \mathbf{B}_i & 0 & \dots & 0 \\ \mathbf{A}_{i+1}\mathbf{B}_i & \mathbf{B}_{i+1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{H-1}\dots\mathbf{A}_{i+1}\mathbf{B}_i & \mathbf{A}_{H-1}\dots\mathbf{A}_{i+2}\mathbf{B}_{i+1} & \dots & 0 \\ \mathbf{A}_H\mathbf{A}_{H-1}\dots\mathbf{A}_{i+1}\mathbf{B}_i & \mathbf{A}_H\mathbf{A}_{H-1}\dots\mathbf{A}_{i+2}\mathbf{B}_{i+1} & \dots & \mathbf{B}_{H-1} \end{bmatrix}. \tag{A.5}$$

Manipulating algebraically (A.2), we rewrite the objective function into a QP form

$$\frac{1}{2}\tilde{\mathbf{U}}_i^\intercal \mathbf{H}_i \tilde{\mathbf{U}}_i + \mathbf{f}_i^\intercal \tilde{\mathbf{U}}_i + \mathbf{d}_i, \tag{A.6}$$

with

$$\mathbf{H}_i = 2(\tilde{\mathbf{B}}_i^\intercal \bar{\mathbf{Q}} \tilde{\mathbf{B}}_i + \bar{\mathbf{R}}), \tag{A.7a}$$

$$\mathbf{f}_i = 2\tilde{\mathbf{B}}_i^\intercal \bar{\mathbf{Q}} \tilde{\mathbf{A}}_i \tilde{\mathbf{z}}_{i|i}, \tag{A.7b}$$

$$\mathbf{d}_i = \tilde{\mathbf{z}}_{i|i}^\intercal \tilde{\mathbf{A}}_i^\intercal \bar{\mathbf{Q}} \tilde{\mathbf{A}}_i \tilde{\mathbf{z}}_{i|i}. \tag{A.7c}$$

Finally, the MPC problem (5.3) is formulated as an LTV-MPC

$$\min_{\tilde{\mathbf{U}}_\mathbf{i}} \quad \frac{1}{2}\tilde{\mathbf{U}}_i^\intercal \mathbf{H}_i \tilde{\mathbf{U}}_i + \mathbf{f}_i^\intercal \tilde{\mathbf{U}}_i \tag{A.8a}$$

$$\text{s.t.} \quad \tilde{\mathbf{U}}_\mathbf{i} \in \mathbb{U}. \tag{A.8b}$$

The constraint $\tilde{\mathbf{U}}_\mathbf{i} \in \mathbb{U}$ expresses the existence of upper and lower bounds on the input values. In this problem we consider that $l_{\min} \leq l_i \leq l_{\max}$ and $c_{\min} \leq c_i \leq c_{\max}$. These constraints are written in matrix form

$$\begin{bmatrix} \mathbf{I} \\ \mathbf{-I} \end{bmatrix} \tilde{\mathbf{U}}_\mathbf{i} \leq \begin{bmatrix} \mathbf{u}_{\max} - \mathbf{u}_i^{\text{ref}} \\ -\mathbf{u}_{\min} + \mathbf{u}_i^{\text{ref}} \end{bmatrix}, \tag{A.9}$$

where $\mathbf{u}_{\min} = [l_{\min}, c_{\min}]^\intercal$ and $\mathbf{u}_{\max} = [l_{\max}, c_{\max}]^\intercal$.

Appendix B

Quadratic problem formulation of EMPC

We cast the problem (6.3) as a QP by defining the optimization variable $\mathbf{u} = [\boldsymbol{\kappa}, \boldsymbol{\Delta}]^\intercal$.

$$\min_{\mathbf{u}} \quad \frac{1}{2}\mathbf{u}^\intercal \mathbf{H}\mathbf{u} \tag{B.1a}$$

$$\text{s. t.} \quad \mathbf{A}\mathbf{u} \leq \mathbf{b}, \tag{B.1b}$$

$$\mathbf{A}_{\text{eq}}\mathbf{u} = \mathbf{b}_{\text{eq}}, \tag{B.1c}$$

where $\mathbf{H} \in \mathbb{R}^{(3H+1)\times(3H+1)}$, $\mathbf{A} \in \mathbb{R}^{(8H+1)\times(3H+1)}$, $\mathbf{b} \in \mathbb{R}^{8H+1}$, $\mathbf{A}_{\text{eq}} \in \mathbb{R}^{3H+1}$, and $\mathbf{b}_{\text{eq}} \in \mathbb{R}$. These matrices are defined as

$$\mathbf{H} = \begin{bmatrix} 2(\mathbf{D}_2^\intercal \mathbf{D}_2 + \alpha \mathbf{D}_1^\intercal \mathbf{D}_1) & \bar{\mathbf{0}} \\ \bar{\mathbf{0}} & 2\boldsymbol{\lambda} \end{bmatrix},$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{\text{box}} \\ \mathbf{A}_{\text{sharp}} \\ \mathbf{A}_{\text{sat}} \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{b}_{\text{box}} \\ \mathbf{b}_{\text{sharp}} \\ \mathbf{b}_{\text{sat}} \end{bmatrix},$$

$$\mathbf{A}_{\text{eq}} = [1\ \mathbf{0}\ \mathbf{0}], \quad \mathbf{b}_{\text{eq}} = \kappa_{\text{vehicle}},$$

where

$$\mathbf{A}_{\text{box}} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_{\text{box}_x} & -\mathbf{I} & \bar{\mathbf{0}} \\ \mathbf{0} & -\mathbf{A}_{\text{box}_x} & -\mathbf{I} & \bar{\mathbf{0}} \\ \mathbf{0} & \mathbf{A}_{\text{box}_y} & \bar{\mathbf{0}} & -\mathbf{I} \\ \mathbf{0} & -\mathbf{A}_{\text{box}_y} & \bar{\mathbf{0}} & -\mathbf{I} \end{bmatrix},$$

$$\mathbf{A}_{\text{box}_x} = -\mathbf{S}([\sin(\mathbf{S}\hat{\kappa})]_{H\times H} \odot \mathbf{S}),$$

$$\mathbf{A}_{\text{box}_y} = \mathbf{S}([\cos(\mathbf{S}\hat{\kappa})]_{H\times H} \odot \mathbf{S}),$$

$$\mathbf{A}_{\text{sharp}} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \bar{\mathbf{0}} & \bar{\mathbf{0}} \\ \mathbf{0} & -\mathbf{I} & \bar{\mathbf{0}} & \bar{\mathbf{0}} \end{bmatrix},$$

$$\mathbf{A}_{\text{sat}} = \begin{bmatrix} \mathbf{D}_1 & \bar{\mathbf{0}} & \bar{\mathbf{0}} \\ -\mathbf{D}_1 & \bar{\mathbf{0}} & \bar{\mathbf{0}} \end{bmatrix},$$

and

$$\mathbf{b}_{\text{box}} = \begin{bmatrix} \boldsymbol{\varepsilon}_x + \mathbf{b}_{\text{box}_x} \\ -\boldsymbol{\varepsilon}_x - \mathbf{b}_{\text{box}_x} \\ \boldsymbol{\varepsilon}_y + \mathbf{b}_{\text{box}_y} \\ -\boldsymbol{\varepsilon}_y - \mathbf{b}_{\text{box}_y} \end{bmatrix},$$

$$\begin{aligned} \mathbf{b}_{\text{box}_x} &= \mathbf{x}_{\text{ref}} - \mathbf{S}\left(\cos(\mathbf{S}\hat{\kappa}) + \sin(\mathbf{S}\hat{\kappa}) \odot (\mathbf{S}\hat{\kappa})\right), \\ \mathbf{b}_{\text{box}_y} &= \mathbf{y}_{\text{ref}} + \mathbf{S}\left(\sin(\mathbf{S}\hat{\kappa}) + \cos(\mathbf{S}\hat{\kappa}) \odot (\mathbf{S}\hat{\kappa})\right), \end{aligned}$$

$$\mathbf{b}_{\text{sharp}} = \begin{bmatrix} \mathbf{1}c_{\max} \\ \mathbf{1}c_{\max} \end{bmatrix},$$

$$\mathbf{b}_{\text{sat}} = \begin{bmatrix} \mathbf{1}\kappa_{\max} \\ \mathbf{1}\kappa_{\max} \end{bmatrix},$$

where $\mathbf{0}$ is a column vector of zeros and $\bar{\mathbf{0}}$ is a block of zeros of appropriate dimensions for well defined matrices. Furthermore, $[(\cdot)]_{H\times H}$ represents a matrix of dimension $H \times H$ defined by H column vectors $(\cdot)$.