Interaction-Aware Motion Planning for Automated Vehicles

Zur Erlangung des akademischen Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau des
Karlsruher Instituts für Technologie (KIT)
angenommene

DISSERTATION

von

Christoph Burger, M.Sc.

Tag der mündlichen Prüfung: 04.11.2022

Hauptreferent: Prof. Dr.-Ing. Christoph Stiller
Korreferent: Prof. Miguel Ángel Sotelo

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Mess- und Regelungstechnik (MRT) des Karlsruher Instituts für Technologie (KIT). Ohne die vielfältige Unterstützung meiner Kollegen, Freunde und Familie gäbe es an dieser Stelle heute nichts zu lesen :)

Zunächst möchte ich Herrn Prof. Christoph Stiller für die Betreuung meiner Arbeit und darüber hinaus für die Schaffung einer Arbeitsatmosphäre, die viel Raum für wissenschaftliches Arbeiten bietet, bedanken. Ebenfalls möchte ich ein Dank an Herrn Prof. Miguel Ángel Sotelo für die Übernahme meines Korreferats und das damit verbundene Interesse an meiner Arbeit aussprechen. Die konstruktiven Diskussionen aber auch die interessanten Gespräche die über das fachliche hinaus gingen habe ich sehr geschätzt.

Ich möchte mich auch bei allen Kolleginnen und Kollegen des MRTs für die äußerst angenehme und entspannte Arbeitsatmosphäre bedanken. Die interessanten und bereichernden Diskussionen während den Kaffeerunden, den gemeinsamen Aktivitäten wie Sommerseminare, Konferenzbesuche, Skiausflüge und Kneipenaufenthalte haben mir immer viel Freude beim Arbeiten und auch außerhalb der Arbeit bereitet. Ein besonderer Dank gilt meinen Kollegen Marc, Hendrik, Frank, Flo, Johannes, Sascha, Sven, Sahin, Jannik, Annika, Jan, Max, Pio, Kevin - Meine Zeit am Institut wurde durch euch in vielerlei Hinsicht bereichert. Zudem möchte ich mich bei Johannes, Frank, Sahin und Hendrik für das Korrekturlesen meiner Arbeit bedanken - Durch euer Feedback konnte die Qualität aber vor allem die Lesbarkeit der Arbeit nochmals gesteigert werden. Ebenfalls möchte ich mich bei dem Sekretariat des MRTs, allen voran bei Erna, die bei bürokratischen Angelegenheiten stets eine große Hilfe war, bedanken.

Abschließend möchte ich meiner Familie und ganz besonders meiner Frau Vanessa danken. Ihr habt mir so vieles ermöglicht und ohne eure Unterstützung würde es diese Arbeit nicht geben - Ihr gebt mir Rückhalt und auf euch kann ich immer zählen.

Vielen Dank!

Karlsruhe im Juni 2022, *Christoph Burger*

Kurzfassung

Die Bewegungsplanung für automatisierte Fahrzeuge (AVs) in gemischtem Verkehr ist eine herausfordernde Aufgabe. Hierbei bezeichnet gemischter Verkehr, Verkehr bestehend aus von Menschen gefahrenen Fahrzeugen sowie automatisierten Fahrzeugen. Um die Komplexität der Aufgabe zu reduzieren, verwenden state-of-the-art Planungsansätze oft die vereinfachende Annahme, dass das zukünftige Verhalten umliegender Fahrzeuge unabhängig vom Plan des AVs vorhergesagt werden kann. Während die Trennung von Prädiktion und Planung für viele Verkehrssituationen eine hilfreiche Vereinfachung darstellt, werden hierbei Interaktionen zwischen den Verkehrsteilnehmern ignoriert, was besonders in interaktiven Verkehrssituationen zu suboptimalem, übermäßig konservativem Fahrverhalten führen kann.

In dieser Arbeit werden zwei interaktionsbewusste Bewegungsplanungsalgorithmen vorgeschlagen, die in der Lage sind übermäßig konservatives Fahrverhalten zu reduzieren. Der Kernaspekt dieser Algorithmen ist, dass Prädiktion und Planung gleichzeitig gelöst werden. Mit diesen Algorithmen können anspruchsvolle Fahrmanöver, wie z. B. das Reißverschlussverfahren in dichtem Verkehr, durchgeführt werden, die mit state-of-the-art Planungsansätzen nicht möglich sind.

Der erste Algorithmus basiert auf Methoden der Multi-Agenten-Planung. Interaktionen zwischen Verkehrsteilnehmern werden durch Optimierung gekoppelter Trajektorien mittels einer gemeinsamen Kostenfunktion approximiert. Das Kernstück des Algorithmus ist eine neuartige Multi-Agenten-Trajektorienplanungsformulierung, die auf gemischt-ganzzahliger quadratischer Programmierung (MIQP) basiert. Die Formulierung garantiert global optimale Lösungen und ist somit in der Lage das kombinatorische Problem zu lösen, welches kontinuierliche Methoden auf lokal optimale Lösungen beschränkt. Desweiteren kann durch den vorgestellten Ansatz ein manöverneutrales Verhalten erzeugt werden, das Manöverentscheidungen in ungewissen Situationen aufschieben kann.

Der zweite Ansatz formuliert Interaktionen zwischen einem menschlichen Fahrer und einem AV als ein Stackelberg-Spiel. Im Gegensatz zu bestehenden Arbeiten kann der Algorithmus allgemeine nichtlineare Zustands- und Eingabebeschränkungen berücksichtigen. Desweiteren führen wir Mechanismen zur Integration von Kooperation und Rücksichtnahme in die Planung ein. Damit wird übermäßig aggressives Fahrverhalten verhindert, was in der Literatur als ein Problem interaktionsbewusster Planungsmethoden identifiziert wurde. Die Wirksamkeit, Robustheit und Echtzeitfähigkeit des Algorithmus wird durch numerische Experimente gezeigt.

Abstract

Motion planning for automated vehicles (AVs) in mixed traffic, where AVs share the road with human-driven vehicles, is a challenging task. To reduce the complexity, state-of-the-art planning approaches often utilize the simplifying assumption that the future motion of surrounding vehicles can be predicted independently of the AV's plan. While separating prediction from panning is a good approximation for many traffic situations, the mutual influence between traffic participants is ignored, which can lead to suboptimal, overly conservative behavior, especially in highly interactive traffic situations.

This thesis proposes two interaction-aware motion planning algorithms that are able to overcome this overly conservative behavior. The key aspect of these algorithms is that the prediction and the planning problem are solved simultaneously. These algorithms can be used to perform challenging driving maneuvers, such as merging in dense traffic, that are not possible with state-of-the-art planning approaches.

The first algorithm is based on multi-agent planning methods. Interactions among traffic participants are approximated by optimizing coupled trajectories using a joint cost function. A core element of this algorithm is a novel multi-agent trajectory planning formulation based on mixed-integer quadratic programming (MIQP). The formulation guarantees globally optimal solutions and is thus able to solve the combinatorial problem, which commonly restricts continuous methods to locally optimal solutions. Furthermore, the proposed algorithm enables maneuver-neutral driving to postpone maneuver decisions in highly uncertain situations.

The second approach formulates the interaction between a human driver and an AV as a Stackelberg game. In contrast to existing works, the algorithm can account for general nonlinear state and input constraints. Further, we introduce mechanisms to integrate cooperation and courtesy into motion planning. This prevents overly aggressive driving behavior, which has been identified in the

literature as a problem of interaction-aware planning methods. The efficacy, robustness, and real-time capability of the algorithm are shown via numerical experiments.

Table Of Contents

Abbreviations and Notations

Abbreviations

AV	Automated vehicle
COG	Center of gravity
CQ	Constraint qualifiers
HD	Human driver
IDM	Intelligent driver model
KKT	Karush Kuhn Tucker
LICQ	Linear independence constraint qualification
LP	Linear program
MINLP	Mixed-integer nonlinear programs
MIP	Mixed-integer programming
MIQP	Mixed-integer quadratic programming
MPC	Model predictive control
MPCC	Mathematical program with complementarity constraints
NLP	Nonlinear program
OCP	Optimal control problem
QP	Quadratic program

Notations

v	Scalar variable
$\boldsymbol{v}$	Vector variable
$\boldsymbol{V}$	Matrix or tensor variable
f	Scalar-valued function
$\mathbf{f}, \mathbf{F}$	Vector-, matrix- or tensor-valued function

1 Introduction

Today academia and industry are rapidly advancing towards deploying automated vehicles onto public roads, bringing the vision of self-driving cars closer to reality. While enormous progress in sensor development, computing power, environment perception and algorithm design has been made in recent years, significant challenges still need to be solved before automated vehicles (AVs) can enter our daily lives. In addition to safety and reliability aspects, generating behavior to drive naturally among humans is still an open field of research. The work presented in this thesis aims to provide solutions for this open issue by presenting motion planning algorithms that are able to generate interaction-aware behavior and therefore enable more human-like driving behavior. The motivation, the problem statement of motion planning in the vicinity of humans, as well as the thesis approach are presented in the following.

1.1 Motivation

When automated vehicles first enter traffic, they will not drive in isolation but share the road with predominantly human drivers. Thus, interacting with them and driving in a similar way to humans is crucial for smooth and efficient operation. This is especially important in situations where the actions of multiple drivers are tidy coupled. For instance, a driver on a highway might decide to slow down or switch lanes so that another driver on the on-ramp can merge. Similar, in dense traffic, a driver might start to nudge into the adjacent lane, hoping that the driver behind will slow down and open a gap. Slowing down at intersections to signalize that another car can go first is another example.

While humans intuitively handle such highly interactive scenarios, generating an interaction-aware behavior for AVs to master such scenarios is still an open problem. Considering the mutual influence among traffic participants,

e.g., anticipating how humans might react onto actions of an AV, dealing with uncertain reactions of humans, and communicating and, if necessary, negotiating the intention in unregulated traffic scenarios, makes interactive driving particularly difficult for AVs. However, AVs will eventually need to master such interactive scenarios to drive in everyday traffic.

The demand to generate interactive behavior raises several fundamental questions we want to analyze in this thesis. How can we model interactions? What is a suitable framework to express dependencies among traffic participants? How can we generate motion planes which incorporate interactive behavior? Furthermore, how can we leverage interactions to drive more naturally among humans and prevent overly conservative driving?

1.2 Problem Statement

A key aspect to achieve a human-like driving behavior is to design algorithms that can consider interactions between human drivers and the AV. However, to reduce the computational complexity of motion planning, most state-of-the-art planners follow a structure that overlooks these mechanisms. In particular, most motion planning algorithms in robotics follow a *predict-then-plan* scheme. Here, the motion planning is separated into a prediction step, where the future motion of surrounding drivers is predicted, and a subsequent planning step, where the motion of the AV is determined. During the planning, surrounding vehicles are treated as moving objects with an immutable trajectory. Due to their structure, these planning methods are also referred to as *pipeline* planners.

While this separation poses a useful simplification for many traffic scenarios, planners utilizing this separation struggle to produce desirable behavior in highly interactive scenarios. This can even lead to situations similar to the *frozen robot* problem [TK10], a state in which the predictions of other traffic participants block all paths, and thus the planner is not able to find a safe trajectory to the goal anymore.

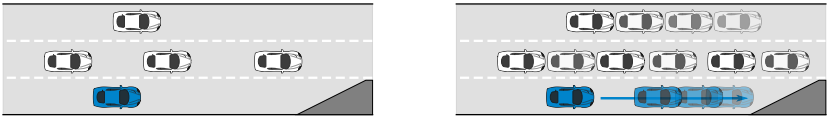
To illustrate this, the merge scenario in Fig. 1.1 is investigated. We first consider the merge scenario in low traffic. Following a pipeline approach, a separate module will provide the predictions of surrounding vehicles used to plan the AV's motion. Depending on the speed and the gap size, the AV decides either

to accelerate and merge in front or to adapt its velocity and merge behind a target vehicle on the highway, see Fig. 1.1.

(a) (b)

Figure 1.1: Following a pipeline planner, the automated vehicle, in blue, first predicts the future motion of the white vehicle. In a subsequent step the AV plans its trajectory avoiding collisions with the predicted states of the white vehicle. As illustrated, the AV is able to perform a merge onto the highway in low traffic.

Next, we consider the same scenario during rush hour, where the traffic density is significantly higher, as shown in Fig. 1.2. Following the same procedure, first predict then plan, will result in a lane blocked by the predictions of others, making it impossible for the motion planner to find a collision-free trajectory onto the highway. As a result, the automated vehicle will decelerate and stop at the end of its lane.



(a) (b)

Figure 1.2: Merge scenario in high traffic. Following a pipeline scheme, the AV is unable to find a collision-free trajectory onto the highway and stops at the end of the lane. While in low traffic, separating prediction and planning is a useful simplification, in high traffic it can lead to suboptimal, overly conservative driving behavior of the AV.

In such situations, a more desirable behavior is to actively approach a gap hoping that these actions might trigger other drivers into opening a gap. However, due to the separation of prediction and planning, pipeline planners are incapable of capturing the influence their own actions have on the actions of others, e.g., how will the prediction of the other vehicle change if the AV tries to merge? This separation restricts the solution space which can lead to undesirable, overly conservative behavior, e.g., stopping at the end of the on-ramp.

1.3 Thesis approach

Following a predict-then-plan scheme, instead of interacting with surrounding drivers, the AV's task is reduced to simple collision avoidance. Further, with this structural decomposition, the AV is incapable of capturing the effect its own actions will have on others, thereby neglecting the interactive nature of traffic. To solve this, expressive methods that are able to capture interactions are needed.

The goal of this thesis is to develop such models and motion planning algorithms that can handle interactions among traffic participants, enabling the AV to behave more naturally among humans in everyday traffic. The key insight to capture the interactive nature of traffic is to design algorithms that overcome the structural limitation of pipeline approaches and solve prediction and planning simultaneously as a joint problem.

This thesis presents two approaches that are able to generate interaction-aware motion for AVs. Both algorithms do not rely on external predictions of other traffic participants but plan collective trajectories. Further, both approaches utilize derivative-based methods, which consider continuous state and input spaces.

The first approach is based on multi-agent-system methods. It is assumed that every traffic participant is working towards a joint objective. Interactions are modeled implicitly in this formulation by assuming that traffic participants cooperatively work towards this joint objective.

The second approach utilizes ideas of game theory and models interactive traffic as a Stackelberg game. In this formulation, it is explicitly considered how own actions can influence the actions of others, which makes it especially suited to purposefully initiate interactions, e.g., starting to nudge in the adjacent lane when doing a lane change.

1.4 Contributions

The contributions presented in this thesis are the following:

- A novel multi-agent trajectory planning formulation based on mixed-integer quadratic programming (MIQP). This formulation is able to overcome the typical limitations of optimization-based methods, namely the combinatorial problem, which originates from the non-convexity of the motion planning task. The presented method guarantees a globally optimal solution for an ensemble of AVs without the restriction of a discretized search space. In contrast to existing approaches, the method exploits the entire configuration space, which enables the generation of cooperative trajectories that are not possible otherwise.
- A cooperative interaction-aware trajectory planning framework based on the aforementioned multi-agent formulation. The main contributions of this extended approach are a planning framework that can generate interaction-aware trajectories and is able to cope with uncertain human driving behavior. Further, a formulation of different intentions, which is independent of the traffic scenario or map layout, is presented. Opposed to most optimization-based methods, the trajectory of the AV is optimized for multiple future outcomes, which ables the AV to postpone a decision for a certain maneuver until the scene becomes more certain.
- A method to generate interaction-aware behavior based on a Stackelberg game formulation. The method enables anticipating how surrounding drivers might react to the AV's future motion. This can be leveraged to plan more efficient trajectories by exploiting interaction, to overcome overly conservative behavior produced by pipeline planners. Further, the formulation provides a direct link between the actions of the AV and the response of a human driver, giving the possibility to purposefully influence the state of the human. One of the main novelties of this game-theoretic algorithm is its ability to consider general nonlinear constraints, thereby ensuring feasibility of the solution. Further, to overcome the overly-aggressive behavior generated by existing interaction-aware planners, methods to introduce cooperativity and courtesy into the motion planning of the AV are proposed. Finally, a method that provides the ability to easily adjust the degree to which the impact on others is considered during planning is presented. This enables the design of planning methods that generate driving behavior ranging from overly conservative to overly aggressive.

1.5 Overview

The remainder of this thesis is structured as follows. In Chapter 2, a survey of motion planning techniques and algorithms is provided. Apart from presenting the basic terminology of motion planning, the focus is on derivative-based methods, which consider continuous state and input spaces. Chapter 3 follows by presenting the fundamentals necessary for this thesis. Concepts of mathematical optimization and special optimization classes such as MIQP, mathematical program with complementarity constraints (MPCC), bi-level problems, and ways to solve them are presented. Chapter 4 consists of two major parts. First, the novel multi-agent trajectory planning algorithm based on MIQP is presented. This formulation is already a stand-alone contribution and can be used to control an ensemble of fully automated vehicles. An interaction-aware planning framework for mixed traffic is presented in Section 4.2. At the core of this framework is the novel MIQP formulation. In Chapter 5, the game-theoretic formulation for interaction-aware trajectory optimization is presented. Finally, Chapter 6 summarizes the thesis and gives an outlook on future work.

2 Survey Motion Planning

This chapter provides an introduction into motion planning, focusing on techniques used for automated driving. After presenting common categories, a review of optimization-based methods, emphasizing interaction-aware planning approaches, is provided.

2.1 Problem Statement

The task of motion planning is to generate a path or trajectory from an initial state to a goal state or region. The resulting path or trajectory has to avoid collisions with obstacles and need to satisfy various types of constraints, e.g., state and input constraints or dynamics bounds. We adopt the widely used naming convention that a path is a time-independent, purely geometric curve, whereas a trajectory additionally requires velocity information at which a curve should be traversed.

In dynamic environments, motion planning is best described in the framework of trajectory optimization. The goal is to obtain a trajectory, i.e., a time-parametrized function $\xi(t) : [0, T] \rightarrow \mathcal{X}$ that describes the evolution of the vehicle's state in time. $\mathcal{X}$ is the set that contains all possible states. Further, let $\Xi(\mathcal{X}, [0, T])$ be the set of all continuous functions that map $[0, T] \rightarrow \mathcal{X}$ and let $\xi(0) = \mathbf{x}_{\text{init}}$ be the initial state of the vehicle and $\mathcal{X}_{\text{goal}} \subseteq \mathcal{X}$ the goal region. The set $\mathcal{X}_{free}(t) \subseteq \mathcal{X}$ contains all allowed states at time $t \in [0, T]$ and encodes, e.g., collision avoidance. Additionally, dynamic constraints on the trajectory can be enforced by $F(\xi(t), \dot{\xi}(t), \ddot{\xi}(t), \dots) = 0$. Finally, let $J(\xi) : \Xi(\mathcal{X}, [0, T]) \rightarrow \mathbb{R}$ denote the cost functional used to evaluate the quality of a trajectory $\xi(t)$.

With the introduced expressions, the trajectory optimization problem can be formulated as:

$$\begin{aligned} \xi^* := \arg\min_{\xi \in \Xi} \quad & J(\xi) \\ s.t. \quad & \xi(0) = \mathbf{x}_{\text{init}} \\ & \xi(T) \in \mathcal{X}_{\text{goal}} \\ & \xi(t) \in \mathcal{X}_{\text{free}}(t) && \forall t \in [0, T] \\ & F(\xi(t), \dot{\xi}(t), \ddot{\xi}(t), \dots) = 0 && \forall t \in [0, T] \end{aligned} \tag{2.1}$$

Note that without loss of generality, we can also address path planning problems since path planning can be considered as trajectory optimization over the unit time interval.

2.2 Motion Planning for Automated Vehicles

Motion planning is a broad field of research and a central aspect of robotics. While many methods have been developed, this review focuses on motion planning techniques for automated vehicles (AVs). A comprehensive overview and general introduction to motion planning can be found in [Cho+05; LaV06].

2.2.1 Important Properties

The most relevant properties to describe motion planning algorithms for this work are explained below.

***Feasible* vs. *optimal* methods** A property that is often used to categorize planning methods is the type of solution they yield. It is distinguished between *feasible* and *optimal* solutions.

If a solution satisfies all constraints of a problem, it is considered a feasible solution. Algorithms that only search for such solutions are also referred to as feasible planning methods. Here, the quality of the solution is of minor importance and is typically not further evaluated. Most path planning

approaches follow this principle, since the major challenge is finding a feasible path at all. A common use case is driving on parking lots.

In optimal planning, there are typically many feasible solutions, and the focus is to find an optimal among them. In order to evaluate and compare the quality of feasible solutions, a cost function is used. This function can consist of multiple measures, which might even be contradictory. E.g., driving fast but also being energy efficient.

***Global* vs. *local* methods** Most trajectory planning approaches can be categorized into combinatorial – *global*, or continuous – *local* methods.

Combinatorial approaches usually guarantee a globally optimal solution in the discretized search space, even for complex problems. However, these approaches have the limitation that they are limited in accuracy by their discretization, and the actual globally optimal solution to the original problem may not be contained in the search space. They are also particularly affected by the curse of dimensionality and suffer in high-dimensional spaces.

Local continuous approaches, on the other hand, start from an initial trajectory that is locally improved. This can avoid the limitation of discretization. In complex search spaces, however, these approaches can at best guarantee local optima.

2.2.2 Motion Planning Methods

While there are numerous ways to categorize algorithms for path and trajectory planning, [Pad+16; Gon+16] identified the following major subfields:

Graph methods The goal of these algorithms is to search for an optimal solution in a constructed graph. Due to their structure, these methods are well suited for complex nonconvex optimization problems, but at the same time, rely on a discretized state and or input space which limits the space of possible solutions. How the graph is constructed and which methods are used to search the graph are essential properties to distinguish existing algorithms.

Two popular examples of graph search algorithms are the well-known Dijkstra algorithm [Dij59] and its extension A^* [HNR68].

Sampling methods The idea of sampling-based methods is to sample configurations in the state space and check for connectivity [ES14]. If the sampled state is collision-free, it is connected to neighboring samples, and a graph is constructed. This graph is then used to find a solution utilizing graph search methods in a subsequent step. Instead of building a graph by sampling first and searching the graph in a second step, incremental methods like *RRT* [LK99] and *RRT** [LK99] combine the sampling and the search by incrementally growing a reachability graph, e.g., a tree structure in the configuration space until the graph is large enough and a node is in the goal region.

Interpolation methods Interpolation methods generate smooth trajectories from a set of reference points, e.g., a sequence of waypoints in a map [Lab+08]. As a result, the original set of reference points is altered and or augmented in favor of attributes like passenger comfort and dynamic feasibility. Different interpolation schemes are applied to path and trajectory planning, ranging from line and circular to spline-based interpolation methods [Gon+16].

Optimization-based methods The goal of optimization-based methods is to formulate the trajectory optimization problem as a mathematical optimization problem which is then optimized with respect to a cost function or reward function. One advantage of these methods is that the generated trajectory is, in general, smoother than those of other methods [SAR18]. Additionally, optimization-based methods provide an elegant way to systematically consider constraints, e.g., arising from vehicle dynamics, in the problem formulation. Compared to other methods, optimization-based methods use local information about the problem to solve for the optimal trajectories. This becomes important, especially in high-dimensional spaces, where sampling-based methods are affected by the combinatorial explosion.

Learning methods Besides classical approaches to motion planning, learning-based methods gained popularity in recent years [BKO19; Zen+20; Zen+19;

Wei+21]. Two of the most common techniques are based on reinforcement and imitation learning. An approach that gained particular attention was the end-to-end imitation learning approach presented in [Boj+16].

The approaches presented in this work belong to the area of optimization-based methods. A review of representative works is presented in Section 2.4. For a more general review of existing methods the reader is referred to the following surveys [Kat+15; Gon+16; Pad+16; Pen+17; SAR18; Ara22].

2.3 Architectures

To put the motion planner into context with other parts of an automated vehicle, a typical system architecture is introduced in this section.

The goal of an automated vehicle is to safely drive from an initial position towards a destination. To do so, the motion planner needs information about surrounding static and dynamic objects as well as the state of the vehicle. This information is provided by sensing the environment. The collected data is then used to identify objects and localize the vehicle.

A common way to structure the system of an automated vehicle is to separate different functional domains into dedicated modules. While there are various concepts [SPV20; Taş+16; Taş+17], a representative system architecture is shown in Fig. 2.1.

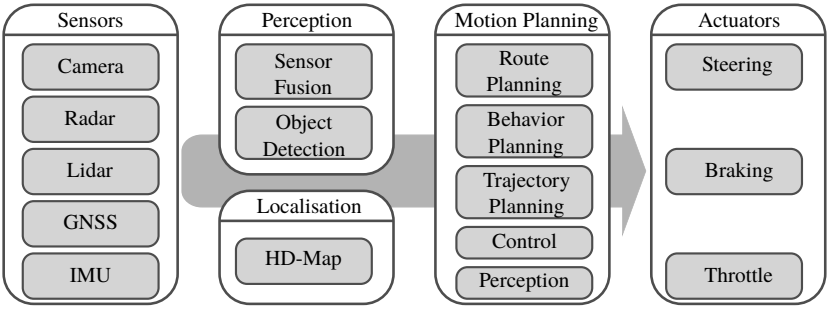


Figure 2.1: Common system architecture for AV.

It is worth noting that also less structured system architectures exist, e.g., in the case of end-to-end learning [Boj+16], where no separate modules exist.

The algorithms presented in Section 2.2 typically only address certain aspects of the motion planning task where high-level decisions, e.g., which lane to drive on or where to turn at intersections, were already made. While formulating a continuous trajectory optimization problem that addresses all these aspects simultaneously is possible, the motion planning problem is commonly divided into subproblems to keep it tractable.

A commonly used structure is to divide the problem into the following layers [Pad+16; Sch+18a]:

Route layer The route layer or route planner provides a plan that navigates from start to the goal, typically on a road topology level. The outcome is a plan through the road network and includes information like which road to take, which lanes to drive on, or where to turn at intersections.

Maneuver layer The maneuver layer addresses the combinatorial nature of driving in dynamic environments. Given a route, discrete decisions, e.g., whether to stop in front of a pedestrian crossing, passing an intersection before or after a crossing vehicle, selecting a gap to merge into during a lane change, or on which side to pass an obstacle are made here.

These discrete decisions are often interpreted as different maneuvers. While there is no common definition of a maneuver [Hub19, p.14], it is often seen as a homotopy from a mathematical viewpoint.

Since local methods are often used in the subsequent trajectory layer, the maneuver layer's goal is to limit the continuous solution space to a certain homotopy. This is especially important for optimization-based methods since the solution is generally bound to the homotopy in which the problem was initialized.

The maneuver decisions are encoded in a rough reference which is then passed to the trajectory layer. This generated reference is not necessarily collision-free, nor is it optimized regarding attributes like passenger comfort or progress towards the goal.

Trajectory layer The trajectory optimization problem is parametrized by the maneuver and route layer via a rough reference. It has no information on higher-level goals and sole optimizes for a smooth collision-free continuous trajectory respecting the dynamic constraints and criteria like passenger comfort and efficiency.

Control layer Finally, the control layer is responsible for tracking the generated trajectory by setting the appropriate control inputs for the vehicle.

Prediction The prediction module is often addressed seperatly to the before mentioned layers. It provides predictions of the future motion of surrounding objects. These predictions are essential to drive in the vicinity of dynamic objects safely.

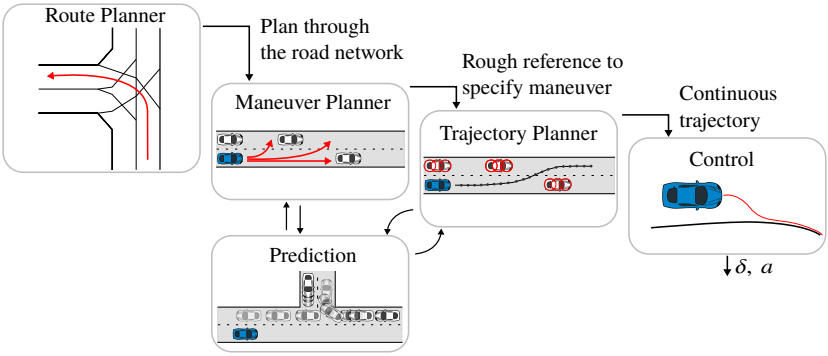


Figure 2.2: Commonly used layers for motion planning in the domain of AVs.

The presented subproblems are not necessarily structured hierarchically. In fact, the way in which the different layers are addressed and interconnected is a major characteristic to distinguish existing planning approaches for automated vehicles. Of particular interest for this work is how predictions are considered in the planning.

Although numerous approaches exist, they mostly assume that the prediction of other traffic participants can be made independently of the ego vehicle's behavior. In this case, the prediction is provided by an upstream model and

treated as immutable during planning. This clear separation between prediction and planning is often referred to as a *predict-then-plan* or *pipeline* concept.

While this separation greatly reduces the computational complexity, it also significantly limits the vehicle's ability to interact with surrounding traffic participants. Prediction and planning need to be solved simultaneously to generate an interactive behavior.

In line with the focus of this work, a detailed review of interaction-aware methods is provided in Section 2.5.

2.4 Optimization-Based Planning for Automated Vehicles

Optimization-based methods solve the motion planning problem by utilizing optimal control and numerical optimization methods [Bet98].

In the context of these methods, the problem of Eq. (2.1) is commonly stated in the following form:

$$\begin{aligned} \xi^* := \underset{\xi \in \Xi}{\arg\min} \quad & J(\xi) \\ s.t. \quad & \xi(0) = \mathbf{x}_{\text{init}} \\ & \xi(T) \in \mathcal{X}_{\text{goal}} \\ & h(\xi(t), \dot{\xi}(t), \dots) = 0 \quad \forall t \in [0, T] \\ & g(\xi(t), \dot{\xi}(t), \dots) \leq 0 \quad \forall t \in [0, T] \end{aligned} \tag{2.2}$$

where collision avoidance, dynamic, and other constraints are formulated as equality and inequality constraints.

Eq. (2.2) describes a variational problem, where the solution is a continuous function in state and time. These problems are generally hard to solve, and an analytic solution can only be found for some special cases. One example is presented in Takahashi et al. [Tak+89]. It is shown that jerk-optimal trajectories in free-space are quintic polynomials. Another example is the widely used linear quadratic regulator (LQR), which is based on the solution of the Riccati differential equation.

While analytic solutions are computationally very efficient, the necessary assumptions are often too restrictive to find these solutions for real applications. However, the results can still be useful, e.g., as approximations or as initial guesses when combined with other methods. E.g., in [Wer+10; Wer+12], jerk-optimal quintic polynomials are sampled in a road-aligned coordinate frame. These sampled trajectories are then transformed into the original frame, and their quality is evaluated.

In most cases, numerical methods are used to approximately solve Eq. (2.2). To do so, it is necessary to discretize the infinite-dimensional function space to a finite-dimensional vector space, where the solution is a vector of values instead of a continuous function in time. Different discretization methods are presented in Section 3.2. The trajectory optimization problem can then be formulated as a non-linear continuous optimization problem.

Due to their advantages stated in Section 2.2, optimization-based methods are widely used in automated vehicles [Gon+16; Pad+16; Kat+15; SAR18]. While some focus on algorithm design for specific traffic scenarios, e.g., highway driving [Liu+18], passing unsignalized intersections [KKJ17; AdL16; Qia+17; Ple17], parking [Zha+18], or cooperative driving with V2V communication [Taş+18], others try to cope with as many scenarios as possible [Zie+14; Liu+17; APdL17; Sch+18b; TS18].

A particularly interesting field for optimization-based methods is automated racing. Racetracks are often used to benchmark the performance of different approaches due to their well-defined environment. Many methods used in automated driving originate from ideas and results of this field [LDM15; AM18; PPB20; APdL17; LL20; Váz+20; Ver+14; RCB17].

The results are particularly useful for emergency scenarios where the vehicle is operated close to its dynamic limits. In these scenarios, it is crucial to fully utilize the available motion capabilities to increase the vehicle's capacity to avoid accidents [Fun+17; Jai+19; Sve+19].

Model predictive control (MPC) methods are particularly popular among optimization-based approaches. Two often-used characteristics to distinguish MPC methods are the type of vehicle model used and whether the problem is formulated in a spatial or temporal domain. While formulating the problem in the temporal domain is very common, formulating it in the spatial domain is done, e.g.,in [Gao+12; Gra+12; Gra+18; LB14; KMS16] and is especially

beneficial when spatio-temporal constraints have to be considered. E.g., when specific waypoints have to be traversed at certain times, as is the case at intersections [Ple17].

Different vehicle models are used depending on the intended application, which consider increasingly complicated lateral vehicle dynamics. Simple point-mass models, the kinematic single-track model, the single-track model, also often referred to as bicycle model, and multi-body models are some of the most popular ones. A detailed introduction to the models used for this work is given in Section 2.6.

Since most optimization-based methods are limited to finding local solutions, a major research challenge originates from the combinatorial nature of driving in dynamic environments [Ben+15]. The combinatorial aspect leads to an, in general, nonconvex cost function with multiple minima. Each minimum can be thought of as a different driving maneuver, see also Section 2.3. Especially purely optimization-based methods struggle if the problem includes such discrete decisions [Zie+14; LDM15].

One way to mitigate this issue is to a priori enumerate a set of maneuvers and solve the problem with different initializations, hoping that the global solution will be among the local solutions [Ben+15; SA17].

As explained in Section 2.3, the motion planning problem is often divided into several layers addressing different time horizons and decision problems. This separation also corresponds well with the nonconvex nature of automated driving and is often used to mitigate the effect the combinatorial problem has on local methods.

A class of continuous methods that can overcome the typical limitations of optimization-based methods is mixed-integer programming (MIP) [Sch+01; NS13; Qia+16; BL18; ES18a; MPA18; EKK20]. Despite a nonconvex cost function, these methods can find a global optimum.

2.5 Interaction-Aware Planning

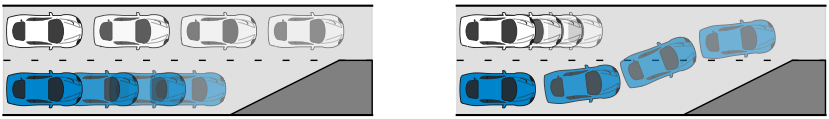
While state-of-the-art planning techniques can cope well with static environments, e.g., consisting of static road bounds and static obstacles, handling

dynamic objects is still an open field of research. However, considering dynamic objects, e.g., pedestrians and human-driven vehicles, is essential for driving in real traffic. Further, the automated vehicle should not only react to other traffic participants but also interact with them, leading to an interactive and cooperative driving behavior.

Driving in the vicinity of dynamic objects requires a prediction to describe the possible future motion of these objects. However, predicting the future motion of surrounding traffic participants is very hard, especially for longer time horizons. While for durations up to one second, the future motion can mainly be inferred from the current state, the prediction for a longer horizon strongly depends on the intention of a road user. It, therefore, becomes increasingly more difficult as the considered time horizon grows [Sch+18c]. What adds to the complexity is that possible interactions with the ego vehicle should also be considered.

As mentioned in Section 2.3, a widely used simplification is to predict the motion of other road users independently of the ego motion by an upstream module [LVL14]. This results in a pipeline approach, where the future motion of surrounding traffic participants is anticipated first, and in a second step, the ego vehicle plans its motion to stay safely out of the way of others.

These approaches provide good results in many scenarios but exhibit overly defensive behavior in interactive scenarios. A merge in dense traffic, for example, describes such an interactive scenario. Here, the behavior of other road users cannot be adequately assessed without knowledge of one's own plan. In the case of a merge, this can even lead to a standstill, as illustrated in figure 2.3



(a) predict-then-plan (b) interaction-aware

Figure 2.3: Merge in dense traffic. In a) the prediction of the white vehicle is done first, and the ego vehicle (blue) then planes around that prediction, leading to a stop at the end of the lane. b) When prediciton and planning are solved together, the ego vehicle can anticipate that the white vehicle will slow down, allowing for a merge.

Due to the separation of prediction and planning, interactions are ignored, and the ego vehicle has no way to influence other road users purposefully. Moreover, neglecting interactions can lead to the so-called *frozen robot* problem, which arises if no safe trajectory to the goal can be found by the planner [TK10]. Any interactive behavior observed in such approaches is purely reactive and based on replanning which is not intentional.

To capture the interactive nature of driving, it is essential to simultaneously predict the motion of others while planning one's own motion. A review of motion planning methods that explicitly consider interactions is given in the following. Consistent with the focus of this thesis, the emphasis is on optimization-based approaches.

Coupled Planning/ Multi Agent Planning One possible technique to generate interaction-aware behavior is coupled planning. The scene is modeled as a multi-agent problem with a joint objective. The underlying assumption is that all traffic participants, the ego-vehicle included, are on the same team, optimizing a joint cost function [CFS13; Kre+16; LKK16; Sch+17; BL18; Ban+18; BSL20]. Each agent then solves the multi-agent problem and executes its part of the plan, assuming all others will as well. To cope with the uncertain driving behavior of humans, these methods are combined with tracking approaches to estimate if the human roughly follows the same model or optimizes for something entirely different [Sch+17; BSL20].

As shown in [LKK16], the result of such methods highly depends on how the different vehicles' costs are considered in the joint cost function. Varying weights can be used to model different levels of cooperation or incorporate asymmetries in the traffic scene [BL18].

Direct Response Methods A more direct way of generating interactive behavior is to model interaction as a reaction other agents will take in response to an action taken by the ego vehicle. E.g., in [Eve+16], the reaction is modeled by the intelligent driver model (IDM) [THH00]. To generate the behavior of the ego vehicle, first, a set of jerk minimum ego candidate trajectories is generated. The reaction of other vehicles is then modeled by the IDM and obtained via a forward simulation of the scene. Finally, each candidate trajectory is evaluated, and the best one is selected.

Most sampling-based planning methods that consider interactions can be associated with this category. The reaction of other road users is obtained by a forward simulation of the traffic scene. This requires a transition model that describes how the traffic scene changes due to the action of the ego vehicle. Among the sampling-based methods, approaches based on partially observable Markov decision process (POMDP) are particularly promising [Hub+18]. In contrast to optimization-based methods, the influence that the ego vehicle exerts on others is not explicitly given, but must be determined by trying out several actions and a subsequent forward simulation.

Game-Theoretic Methods A fundamental assumption of coupled planning methods is that each agent is as interested in others reaching their goals as it is in reaching its own goal. Unfortunately, this is often not the case in real traffic since some drivers are only interested in optimizing their own driving. Therefore, in scenarios with multiple decision-makers, each maximizing their individual performance measure, potentially influencing each other, it is recommended to take a multi-agent perspective [RN21, p. 43] instead.

Dynamic games have shown to be well suited to capture the interactive nature among multiple decision-makers. Several approaches have been developed considering a range of applications with some degree of game-theoretic interactions. E.g., in automated driving, game-theoretic approaches were used for lane changes, merge scenarios, intersection crossing, traversing roundabouts, and overtaking [Sad+16a; Sad+16b; Sad+18; Fri+20; DG18; Fis+19; Wan+20; LSM21; Wan+21]. E.g., in [Sad+16b], human-like driving behavior, e.g., slowing down before intersections or nudging into the adjacent lane while doing a lane change, could be generated.

Other related applications of game-theoretic planning include agile maneuvering of multiple ground vehicles in close proximity [Wil+18], automated car racing [Wan+19; LL20; Not+20; Wan+21], as well as drone racing [Spi+20; WTS20; WSS19]. For example, in the racing scenario [Wan+19], it is shown that a game-theoretic planner yields complex strategies such as blocking and faking and significantly outperforms a baseline MPC planner.

The presented works model the prediction and planning task jointly as a non-cooperative game. In these games, there is no optimal solution in the traditional sense, but depending on the game's structure and symmetry among players,

different solutions are possible, also referred to as equilibria. Therefore, an important feature to categorize game-theoretic methods is the type of solution they are solving for. Here it is distinguished between *Nash* and *Stackelberg* equilibria.

A Nash equilibrium describes a set of strategies where no individual player can benefit from unilaterally changing its strategy, given that all other agents will stick to their strategy. This type of equilibria has been investigated, e.g., in [LSM22; Fri+20; DG18; Wan+19; BDG19; Wil+18].

Compared to a Nash equilibrium, a Stackelberg equilibrium involves turn taking and, therefore, an asymmetry in the decision-making process. It is typically modeled for a two-player game, where one player is the leader, and the other is the follower. The leader chooses its strategy first, and the follower then optimizes its strategy as the best response to the leader's strategy. In contrast, the Nash solution can be seen as the best response from everyone to everyone else without hierarchical turn-taking. Stackelberg equilibria are considered in [Sad+16b; Sad+16a; Fis+19; LL20; YL13; YL14; LL20; Sun+18].

Objective Design The cost function is a central part of a planning algorithm and provides incentives for desirable driving behavior. Traditional approaches typically design the cost function somewhat selfishly with the primary objective to optimize the ego vehicle's comfort, efficiency, and safety. These selfish incentives are not an issue in approaches that follow the predict-then-plan philosophy, since the goal is to optimize comfort and efficiency while safely staying out of the way of other drivers.

For interaction-aware planning methods, a cost function designed in a selfish way can lead to overly aggressive driving behavior [LSM22; Fis+19; Sun+18], e.g., cutting off other drivers or accelerating before intersections to go first [Sad+16b]. The reason for this aggressive behavior is that interactive methods like [Sad+16b] focus on how interaction can be handled, but it is generally not considered how this interaction influences the comfort and safety of other drivers.

Understanding how other drivers might be influenced enables the automated vehicle to exploit interactions to further optimize their own costs, e.g., changing lanes fast by cutting off other drivers. However, as pointed out in [Fis+19], a

selfishly designed cost function could even lead to trivial solutions where a leader does not consider the mutual collision constraints anymore, leading to an overly aggressive leader and overly passive follower, ignoring the interactive nature of the traffic scene.

While a more aggressive behavior is sometimes desirable, e.g., when merging into dense traffic, this should not be the default. So as planning methods get more sophisticated, special care must be taken when designing the cost function to set the right incentives. Several works have focused on modifying the cost function to mitigate the aggressive behavior.

Evestedt et al. [Eve+16], Sun et al. [Sun+18], and Bansal et al. [Ban+18] introduce additional costs that penalize any inconvenience caused to other drivers due to the behavior of the ego vehicle. E.g., in [Eve+16] additional costs occur if other drivers experience high deceleration. The additional costs enable the ego vehicle to be aware of the inconvenience caused by its behavior. These methods go in line with the concept of cooperative driving according to our previous work [Bur+17], suggesting that to achieve cooperative behavior, not only the ego vehicle's own costs should be considered but also the cost of others.

A similar idea is utilized in [Sun+18], which is based on the work of [Sad+16b]. In [Sad+16b] interactions between a human and a robot are modeled as an underactuated dynamic system, where the human plans its actions knowing that these will not only affect its state but also the actions the human will take as a response. Whereas in [Sad+16b], the focus is on how to model interactions, the focus of [Sun+18] is what should be optimized and how interactions influence other drivers. The cost function is extended by a *courteous* term. This term penalizes the increase in other drivers' costs given the ego vehicle behavior, compared to an alternative best-case scenario for the other driver. Therefore, the courteous term is a way to measure the inconvenience the ego behavior causes to other drivers. When considering the courteous term, it is also demonstrated that robot vehicles drive less aggressively in various traffic scenarios, e.g., leaving more space while performing a merge.

In [Ban+18], a selfishness factor is introduced to balance own interests with those of others. This factor determines the relative weighting of the ego vehicle and other vehicles in a joint cost function. Depending on the value, more or less cooperative behavior can be generated.

2.6 Vehicle Models

Vehicle models are a central part of motion planning algorithms. They are used to approximate the vehicle's behavior to a given control action. Several models exist which consider the lateral vehicle dynamics in different levels of complexity. While high-fidelity models like the multi-body model used in [BBL07] more accurately describe the vehicle's motion, they are commonly associated with significant additional computational efforts. On the other hand, overly simplified models might ignore important dynamical effects.

Selecting a vehicle model therefore poses a trade-off between model accuracy and computational costs and highly depends on the intended application and requirements. E.g., in emergency situations, sophisticated motion models are required to capture the behavior of a vehicle close to its dynamic limits; simpler models can be used for everyday driving situations. Even simpler models can be used in highway scenarios where the longitudinal component dominates the movement.

It is usually considered sufficient to focus on planar motion only in automated driving applications. Further, it is commonly assumed that the center of gravity (COG) is at the height of the road surface, and no roll and pitch dynamics or load transfer due to accelerations occur.

The most widely used models for car-like vehicles include the point mass model, the kinematic single track, and the (dynamic) single track model. An introduction to these models is given in the following section. For a more detailed overview, the reader is referred to [Raj12].

2.6.1 Point Mass Model

The point mass model, shown in Fig. 2.4, is the simplest, widely used vehicle model which considers lateral dynamics. It is assumed that the vehicle can be represented as a point mass and can move freely in any direction. There is no steering action; instead, time derivatives of the position in x and y direction are used as input. Depending on the order of the model, the inputs are either the velocities, accelerations, or higher-order derivatives, e.g., jerks.

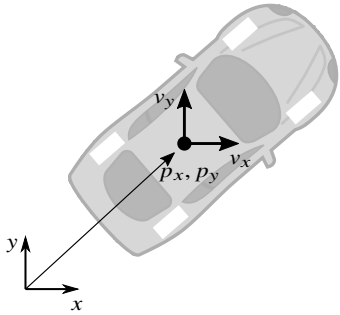


Figure 2.4: Second order point mass model.

Let us denote $\square_x$ and $\square_y$ as the components of a variable $\square$ in x and y direction. Further, let p describe the position, v the velocity, and a the acceleration in a global frame. With the state $\mathbf{x} = (p_x, p_y, v_x, v_y)^T$ and input $\mathbf{u} = (a_x, a_y)^T$ a possible representation for a 2nd order point mass model can be given as the following linear system:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{v_x} \\ \dot{v_y} \end{pmatrix} = A\mathbf{x} + B\mathbf{u}, \quad \text{with} \quad A = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{2.3}$$

Please note that in this formulation, the vehicle's heading is not explicitly given and has to be inferred by the velocity components of the velocity vector.

2.6.2 Kinematic Single-Track Model

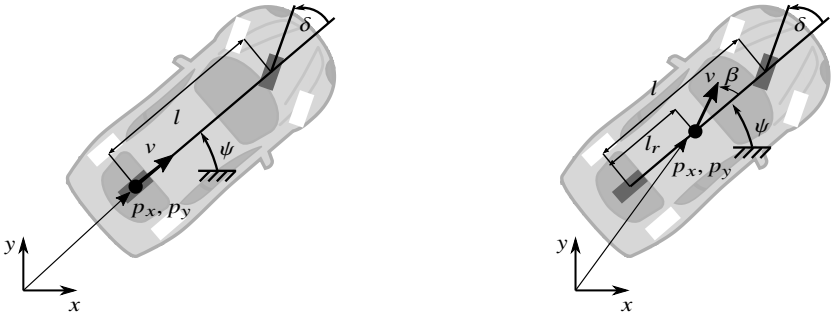
The classical kinematic single tack model models the vehicle under the assumption that each pair of wheels, rear, and front, can be represented by a single virtual wheel placed in the middle of each axis, see Fig. 2.5. The resulting model consists of two wheels connected by a rigid link similar to a bicycle, hence it is often referred to as the bicycle model. It is further assumed that there is no tire slip, and the motion is always aligned with the orientation of the wheels.

A simple example where the kinematic single track model is superior to a point mass model is parking. A point mass model would not be sufficient in this case since it does not capture the non-holonomic characteristic of the vehicle. These characteristics describe that no lateral displacement is possible without a simultaneous forward motion and is essential to calculate a minimum turning radius.

Additional to the position of the reference point p, the yaw angle ψ, which describes the vehicle's orientation with respect to the x-axis and the absolute velocity v at the reference point, are introduced. The inputs to the model are the steering angel δ at the front wheel and the longitudinal acceleration a. Furter, the wheelbase l is a necessary parameter.

The kinematic single tack model can be described by the state $\mathbf{x} = [p_x, p_y, \psi, v]^T$ and the input $\mathbf{u} = [\delta, a]^T$.

Please note that slightly different differential equations are obtained depending on the choice of the reference point. The two most common are 1) the center point of the rear axis, see Fig. 2.5a and 2) the COG, see Fig. 2.5b.



(a) Rear axis

(b) Center of gravity

Figure 2.5: Kinematic single tack model a) with the center of the rear axis as reference or b) with the center of gravity as reference.

For the center of the rear axis the model can be stated as:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{\psi} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos(\psi) \\ v\sin(\psi) \\ \frac{v}{l}\tan(\delta) \\ a \end{pmatrix} \tag{2.4}$$

Here, the direction of the velocity vector is always aligned with the vehicle's longitudinal axis.

In case the COG is chosen as a reference point, an additional variable, the slip angle β, and an additional parameter l_r are required. The slip angle β measures the angular difference between the velocity vector and the vehicle's orientation. l_r is the distance from the rear axis to the COG. The system dynamic can then be described as follows:

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{pmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{\psi} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \frac{v}{l}\tan(\delta)\cos(\beta) \\ a \end{pmatrix} \\ \beta &= \arctan\left(\frac{l_r}{l}\tan(\delta)\right) \end{aligned} \tag{2.5}$$

A major limitation of the kinematic single track model is that the generated motion is purely geometric and does not depend on the velocity. This behavior can be best described as driving as on rails. As maneuvers get more dynamic and lateral accelerations increase, the tire slip can not be neglected anymore, and the model no longer approximates the vehicle's real behavior well. Especially in higher velocities cornering scenarios, the kinematic single track model significantly underestimates the required cornering radius and produces trajectories the real vehicle cannot follow.

2.6.3 Dynamic Bicycle Model

The extension to the kinematic single track model is the (dynamic) single track model. It considers that forces in the contact point between road and tire are built up by tire slip. As a result, it is more accurate, and important driving

effects like under and over steering are captured. One example where this level of complexity is required is planning evasive maneuvers close to the vehicle's handling limit.

In addition to the variables and parameters of the kinematic single track model, the yaw rate $\dot{\psi}$, the yaw moment of inertia J and lateral and longitudianl tire forces at the front and rear wheel $F_{f_{\text{lat}}}$, $F_{r_{\text{lat}}}$, $F_{f_{\text{long}}}$, $F_{r_{\text{long}}}$, as shown in Fig. 2.6, are introduced.

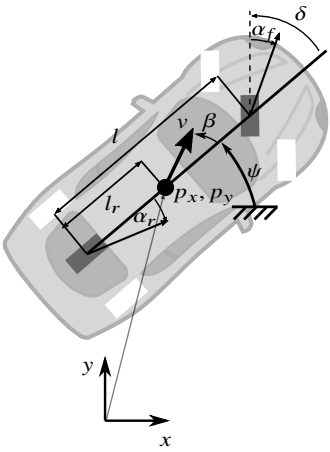


Figure 2.6: The single track model with tire forces and side-slip angles

Compared to the simpler kinematic model, the state is extended by the yaw rate $\dot{\psi}$ and the slip angle β. Given state $\mathbf{x} = [p_x, p_y, \psi, v, \dot{\psi}, \beta]^T$ and input $\mathbf{u} = [\delta, F_{\text{long}}]^T$, the dynamic model with respect to the COG can be stated as:

$$\dot{\mathbf{x}} = \begin{pmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \dot{\psi} \\ \frac{-F_{f_{\text{lat}}}\sin(\delta-\beta)+F_{f_{\text{long}}}\cos(\delta-\beta)+F_{r_{\text{lat}}}\sin\beta+F_{r_{\text{long}}}\cos\beta}{m} \\ \frac{F_{f_{\text{long}}}l_v\sin\delta-F_{r_{\text{lat}}}l_h+F_{f_{\text{lat}}}l_v\cos\delta}{J} \\ -\dot{\psi}+\frac{F_{f_{\text{lat}}}\cos(\delta-\beta)+F_{f_{\text{long}}}\sin(\delta-\beta)+F_{r_{\text{lat}}}\cos\beta-F_{r_{\text{long}}}\sin\beta}{mv} \end{pmatrix} \tag{2.6}$$

With

$$[F_{f_{\text{long}}}, F_{r_{\text{long}}}] = [\gamma, (1-\gamma)]F_{\text{long}}$$

The formulation is adopted from [Wer11, p. 74] and [Fuc05, p. 98] Note that instead of directly controlling the acceleration, the longitudinal force is chosen alongside the steering angle as an input, where γ describes the force distribution between front and rear wheels.

As mentioned, the lateral tire forces $F_{f_{\text{lat}}}, F_{r_{\text{lat}}}$ in Fig. 2.6 are functions of the side slip angles α_f, α_r which are given as:

$$\alpha_f = \delta - \arctan\left(\frac{l_f\dot{\psi}+v\sin\beta}{v\cos\beta}\right), \quad \alpha_r = \arctan\left(\frac{l_r\dot{\psi}-v\sin\beta}{v\cos\beta}\right) \tag{2.7}$$

An often used simplifying assumption is that the lateral forces increase linearly with the side slip angles $F_{f_{\text{lat}}} = C_f\alpha_f, F_{r_{\text{lat}}} = C_r\alpha_r$. Here C_f, C_r are the cornering stiffnesses coefficients. While this approximation is often used, it can lead to unbounded lateral forces and is only valid for small angles. More sophisticated models, like the Pacejka Magic Formula [PB12, p.7], consider a saturation in the tire forces and are better suited for bigger angles.

Despite its higher accuracy, the single track model is not always preferred. The two main reasons are the additional computational burden and the numerical issues at low velocities, leading to a singularity at a standstill. In order to avoid the division by zero, it is common to switch to the kinematic model at lower speeds as it is done e.g., in [Jai+19].

The dynamic single track model is used for the simulation environments in chapter 4 and 5.

3 Fundamentals of Optimization-Based Motion Planning

The methods presented in this work can be categorized as optimization-based motion planning methods. The essential concepts of mathematical optimization, optimal control and model predictive control (MPC) to understand and solve them are presented in this chapter.

3.1 Mathematical Optimization

As explained later, the methods developed in Chapter 4 and Chapter 5 are based on transforming infinite-dimensional optimal control problems (OCPs) into finite-dimensional optimization problems, which are then solved numerically. An important class of finite-dimensional optimization problems are nonlinear programs (NLPs) which in general can be stated as:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f(\mathbf{x}) \\ s.t. \quad & h(\mathbf{x}) = 0 \\ & g(\mathbf{x}) \leq 0 \end{aligned} \tag{3.1}$$

where

- $\mathbf{x} \in \mathbb{R}$ denotes the vector of decision variables.
- $f(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$ denotes the cost or objective function.
- $h(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^{n_h}$ denotes the vector of equality constraints.

- $g(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^{n_g}$ denotes the vector of inequality constraints.

Two important NLP subclasses are:

- *Linear programs* (LP): have affine constraint functions $h(\mathbf{x})$, $g(\mathbf{x})$ and a linear objective function $f(\mathbf{x}) = c^T\mathbf{x}$.
- *Quadratic programs* (QP): also have affine functions $h(\mathbf{x})$, $g(\mathbf{x})$ and a linear quadratic objective function $f(\mathbf{x}) = c^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T B\mathbf{x}$ with a symmetric matrix $B \in \mathbb{R}^{n\times n}$

If the vector of decision variables also comprises discrete or integer-valued variables, e.g.,$\mathbf{z} \in \mathbb{Z}^{n_z}$, problems like 3.1 are generally referred to as mixed-integer nonlinear programs (MINLP).

Feasible set: *The feasible set* Ω *is defined as:*

$$\Omega := \{\mathbf{x} \in \mathbb{R}^n \mid h(\mathbf{x}) = 0,\ g(\mathbf{x}) \leq 0\} \tag{3.2}$$

and contains all points that satisfy all constraints.

Among all feasible points, we are interested in $\mathbf{x}^*$, which minimizes the objective function $f(\mathbf{x})$. Generally, two types of minima are distinguished.

Global minimum: $\mathbf{x}^*$ *is a global minimizer if* $\mathbf{x}^* \in \Omega$ *and*

$$f(\mathbf{x}^*) \leq f(\mathbf{x})\ \forall \mathbf{x} \in \Omega \tag{3.3}$$

For general NLPs, the global optimum is challenging to find, and most solution algorithms are limited to finding local minimizers.

Local minimum: $\mathbf{x}^*$ *is a local minimizer if* $\mathbf{x}^* \in \Omega$ *and there exists a neighborhood N for which*

$$f(\mathbf{x}^*) \leq f(\mathbf{x})\ \forall \mathbf{x} \in \Omega \cap N \tag{3.4}$$

holds.

A special class of optimization is convex optimization. An optimization problem is convex if the following conditions hold [BV04, p.137]:

- the objective function $f(\mathbf{x})$ is convex.

- the equality constraints $h(\mathbf{x})$ are affine.
- the inequality constraints $g(\mathbf{x})$ are convex.

Convex function: *A function* $f : \Omega \to \mathbb{R}$ *is convex if* Ω *is a convex set, and if* $\forall \mathbf{x}, \mathbf{y} \in \Omega$ *and* $\lambda \in [0, 1]$ *the following holds:*

$$f(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1 - \lambda) f(\mathbf{y}) \tag{3.5}$$

Graphically speaking, this means all secants are above the graph.

Convex set: *A set* Ω *is convex if* $\forall \mathbf{x}, \mathbf{y} \in \Omega$ *and* $\lambda \in [0, 1]$ *the following holds:*

$$\mathbf{x} + \lambda(\mathbf{y} - \mathbf{x}) \in \Omega \tag{3.6}$$

This means that the connecting line between any two points in the set is also inside the set.

Convex optimization problems are of particular interest since every local minimizer $\mathbf{x}^*$ is also a global minimizer [BV04, p. 137].

For $\mathbf{x}^* \in \Omega$ to be a local minimizer of Eq. (3.1), the point has to satisfy the first order necessary optimality conditions (FONC). If, in addition, the optimization problem is convex, the necessary optimality conditions are even sufficient to guarantee that $\mathbf{x}^*$ is a global minimizer. If these conditions are not satisfied, $\mathbf{x}$ can not be a local minimizer. Due to this fact, the FONC play a key role in many solution algorithms.

The FONC for NLPs are called the Karush-Kuhn-Tucker (KKT) conditions [NW06, p.321] and can be stated as follows:

KKT conditions: *If* $\mathbf{x}^*$ *is a local minimizer of the NLP 3.1 and* $\mathbf{x}^*$ *is a regular point of the equality and all active inequality constraints, then there exist multipliers* $\mu \in \mathbb{R}^{n_h}$ and $\lambda \in \mathbb{R}^{n_g}$ *such that the following holds:*

$$\nabla f(\mathbf{x}^*) + \nabla h(\mathbf{x}^*)\lambda^* + \nabla g(\mathbf{x}^*)\mu^* = 0 \tag{3.7a}$$
$$h(\mathbf{x}^*) = 0 \tag{3.7b}$$
$$g(\mathbf{x}^*) \leq 0 \tag{3.7c}$$
$$\mu^* \geq 0 \tag{3.7d}$$
$$\mu^{*T} g(\mathbf{x}^*) = 0 \tag{3.7e}$$

The KKT-conditions can only be formulated for regular points, so not all local minimizers are KKT-points. For $\mathbf{x}^*$ to be a regular point, constraint qualifications must be satisfied. A variety of such constraint qualifications exist [NW06, p.338]. Due to its numerical benefits, the linear independence constraint qualification (LICQ) is frequently used.

LICQ holds at $\mathbf{x}^* \in \Omega$ *if the gradients of the equality constraints* $\nabla h_i(\mathbf{x}^*)$ *for* $i \in \{1, .., n_h\}$ *and the gradients of all active inequality* $\nabla g_i(\mathbf{x}^*) \forall\ g_i(\mathbf{x}^*) = 0$ *for* $i \in \{1, .., n_g\}$ *are linearly independent.*

3.1.1 Mixed-Integer Quadratic Program

Mixed-integer quadratic programmings (MIQPs) are a special class of MINLP with a linear quadratic objective function and affine constraints. MIQPs are at the core of the method presented in Chapter 4 and can, in general, be stated as:

$$\min \quad \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T Q\mathbf{x} \tag{3.8a}$$
$$s.t. \quad A\mathbf{x} - \mathbf{b} = 0 \tag{3.8b}$$
$$C\mathbf{x} - \mathbf{d} \leq 0 \tag{3.8c}$$
$$\mathbf{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p} \tag{3.8d}$$

With matrices $Q \in \mathbb{R}^{n \times n}$, $A \in \mathbb{R}^{n_h \times n}$, $C \in \mathbb{R}^{n_g \times n}$ and vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^{n_h}$, $d \in \mathbb{R}^{n_g}$. The total number of decision variables is n, of which p are integer valued.

Due to the integer variables, mixed-integer programs are well suited to model combinatorial decisions in optimization tasks. Unfortunately, optimization problems with integer variables are never convex due to the nonconvexity of the set of integers. As stated in the previous section, a global optimum can, in general, not be guaranteed for nonconvex problems, and solvers are limited to yield local solutions.

MIQPs pose an exception to this. Despite their nonconvexity, methods exist that guarantee globally optimal solutions if Q is a positive semi-definite matrix [Kal13, p.256].

While a detailed review of different solution techniques is well beyond the scope of this section, it is important to understand the core principles of mixed-integer programming (MIP) solution methods in order to understand how the integer variables affect the computational complexity.

Thus, the basics of the *branch-and-bound* method [Kar06, p.48], which is at the core of most MIP solution algorithms, is presented.

For a better illustration, the following MIQP is considered:

$$\begin{aligned} \min \quad & (x_1 - 1.5)^2 + (x_2 - 1.5)^2 && \text{(3.9a)} \\ s.t. \quad & x_1 - x_2 + 0.5 \leq 0 && \text{(3.9b)} \\ & x_1, x_2 \in \mathbb{Z}^2 && \text{(3.9c)} \end{aligned}$$

Solution methods for MIQPs or MIPs, in general, are based on a relaxation of the mixed-integer problem. Therefore, the integer restriction is removed, and several subproblems in continuous space are solved. In our example, the MIQP is relaxed to multiple QPs, see Fig. 3.1.

The feasible set of the first QP relaxation, denoted by $QP1$, is depicted in Fig. 3.2a. If the solution to $QP1$ satisfies the integer condition, the optimal solution is found. If this is not the case, the problem is split at an integer variable into two subproblems. This split, also called branching, is done by

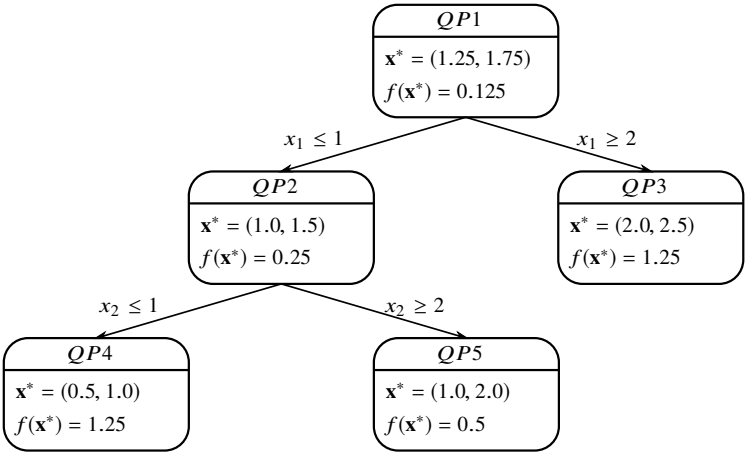


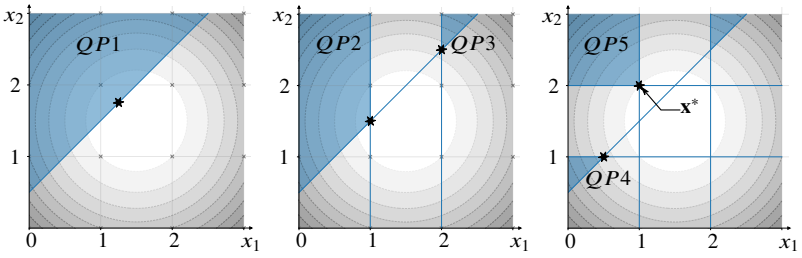
Figure 3.1: Tree structure of QPs solved in a *branch and bound* algorithm. Even though QP3 and QP4 could be branched further, since their objective value is higher than the one of QP5, no further branching is needed, and the optimal solution is found.

introducing constraints that push a fractional valued variable to the adjacent integer value.

For $QP1$, the optimal solution is $\mathbf{x}^* = (1.25, 1.75)$ which does not meet the integer condition. Thus, the problem is split into two new subproblems, $QP2$ with $x_1 \leq 1$ and $QP3$ with $x_1 \geq 2$. The feasible sets after branching on x_1 are depicted in Fig. 3.2b. Note that also a split regarding x_2 would be possible.

To find the optimal solution, the new subproblems are solved again, and a tree-like structure, see Fig. 3.1, is created where each node represents a subproblem. A node that has not been branched on is called *active*. Active nodes are explored further since they could contain a better solution than the currently best one. Each active node is branched until one of the following stop criteria applies [ST08]:

- The QP subproblem is infeasible.
- The solution of the relaxed QP is worse than the currently best solution that satisfies the integer conditions.



(a) relaxed (b) split into $x_1 \leq 1, x_1 \geq 2$ (c) split into $x_2 \leq 1, x_2 \geq 2$

Figure 3.2: In the *branch-and-bound* algorithm, mixed-integer problems are relaxed and branched into several continuous subproblems until no further split is possible or necessary. The feasible sets of the different subproblems are illustrated in blue. The individual solutions are marked with a star.

- If the integer condition is satisfied, it is compared to the currently best solution. In case it is better, it becomes the new current best, if not, the node is closed.

An optimal solution is found if there are no active nodes left thus, no further split is possible or necessary. Fig. 3.2c shows the final feasible sets, including the optimal solution for example 3.9.

An important property to consider when modeling MIPs is that, in general, the complexity grows exponentially with the number of integer variables. Even though this results in an adverse theoretic run-time, the performance in practice is good if a sophisticated solver like GUROBI [Gur22] is used [ES18b].

3.1.2 Bilevel Optimization

Bilevel optimization problems are mathematical programs where an optimization problem is nested inside another one as a constraint. The outer optimization task is commonly referred to as the leader's or upper-level problem, whereas the inner optimization task is the follower's or the lower-level problem. Bilevel problems have a hierarchical structure and are therefore not symmetric, meaning that the leader makes its decision $\mathbf{x}$ first, and the follower will choose its strategy $\mathbf{y}$ as the best response given the leader's decision. Both the leader

and the follower seek to optimize their own outcomes. In game theory, this type of problem is usually referred to as a Stackelberg game, as introduced in Section 2.5. Bilevel optimization is at the core of the method developed in Chapter 5.

Fig. 3.3 illustrates the general structure of a bilevel optimization problem. It is shown that given an upper-level decision vector $\mathbf{x}$, there is a parametric lower-level optimization problem. This lower-level optimization problem refects the optimal response of the follower given the leader's decision.

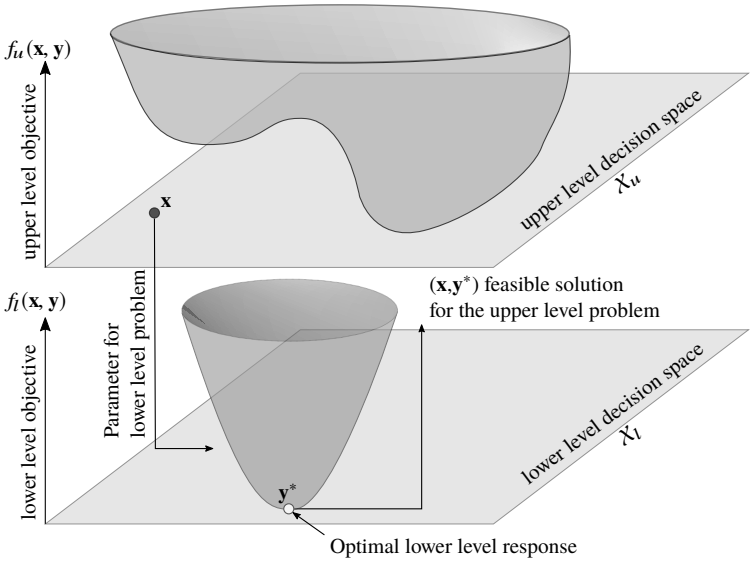


Figure 3.3: Structure of a bilevel optimization problem.

The lower-level optimization is parametrized by the decision vector of the leader $\mathbf{x}$ and is given as:

$$\min_{\mathbf{y}} \quad f_l(\mathbf{x}, \mathbf{y}) \tag{3.10a}$$

$$s.t. \quad \mathbf{g}_l(\mathbf{x}, \mathbf{y}) \leq 0 \tag{3.10b}$$

$$\mathbf{h}_l(\mathbf{x}, \mathbf{y}) = 0 \tag{3.10c}$$

Here, $f_l(\mathbf{x}, \mathbf{y})$ is the follower's objective function, $\mathbf{g}_l(\mathbf{x}, \mathbf{y})$ and $\mathbf{h}_l(\mathbf{x}, \mathbf{y})$ are the corresponding inequality and equality constraints.

The set of optimal solutions for Eq. (3.10a) can equivalently be stated as the best response mapping:

$$\Psi(\mathbf{x}) = \underset{\mathbf{y}}{\operatorname{argmin}}\{f_l(\mathbf{x}, \mathbf{y}) : \mathbf{g}_l(\mathbf{x}, \mathbf{y}) \leq 0, \mathbf{h}_l(\mathbf{x}, \mathbf{y}) = 0\} \tag{3.11}$$

The leader optimizes its objective function taking the best response of the follower into account by considering 3.11 as a constraint. This way, only optimal solutions for the follower are part of the feasible set for the leader. The optimization of the leader can then be stated as:

$$\min_{\mathbf{x},\mathbf{y}} \quad f_u(\mathbf{x}, \mathbf{y}) \tag{3.12a}$$

$$s.t. \quad \mathbf{g}_u(\mathbf{x}, \mathbf{y}) \leq 0 \tag{3.12b}$$

$$\mathbf{h}_u(\mathbf{x}, \mathbf{y}) = 0 \tag{3.12c}$$

$$\mathbf{y} \in \Psi(\mathbf{x}) \tag{3.12d}$$

Here, f_u is the upper-level objective function, and $\mathbf{g}_u$ and $\mathbf{h}_u$ are the upper-level constraints. Note that $\mathbf{y}$ is considered a decision variable of the upper-level problem, but it is actually controlled by the follower through the best response mapping $\Psi(\mathbf{x})$.

In general, $\Psi(\mathbf{x})$ can have multiple optimal solutions, which causes an ambiguity in the upper-level optimization since it is unclear to the leader which action the follower will utilize. To avoid this ambiguity, it is common to assume a certain position of the follower. In an optimistic position, the leader expects the follower to choose the solution from $\Psi(\mathbf{x})$, which leads to the best outcome for the leader. For the pessimistic case, the leader assumes that the follower will choose the action that leads to the worst outcome for the leader, which is more difficult to solve.

To solve bi-level optimization problems, they need to be transformed into a single-level representation. Various methods exist to achieve this. Here only the KKT reformulation used in this work, will be presented. The interested reader is referred to [CMS05; SMD18; DZ20] for a more comprehensive inside to the topic.

KKT-Reformulation For the following single-level reformulation, we assume that 3.10a is convex and the Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient for a global optimum. If we further assume that there is a unique solution, the best response mapping $\Psi(\mathbf{x})$ reduces to a singleton, and the lower-level problem can be replaced by its KKT conditions. This leads to the following NLP formulation of the bi-level problem:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{y}} \quad & f_u(\mathbf{x},\mathbf{y}) && \text{(3.13a)}\\
s.t. \quad & \mathbf{g}_u(\mathbf{x},\mathbf{y}) \leq 0 && \text{(3.13b)}\\
& \mathbf{h}_u(\mathbf{x},\mathbf{y}) = 0 && \text{(3.13c)}\\
& \nabla_{\mathbf{y}} L(\mathbf{x},\mathbf{y},\boldsymbol{\lambda},\boldsymbol{\mu}) = 0 && \text{(3.13d)}\\
& \mathbf{g}_l(\mathbf{x},\mathbf{y}) \leq 0 && \text{(3.13e)}\\
& \mathbf{h}_l(\mathbf{x},\mathbf{y}) = 0 && \text{(3.13f)}\\
& \boldsymbol{\mu} \geq 0 && \text{(3.13g)}\\
& \boldsymbol{\mu} \perp \mathbf{g}_l(\mathbf{x},\mathbf{y}) && \text{(3.13h)}
\end{aligned}$$

with

$$L(\mathbf{x},\mathbf{y},\boldsymbol{\lambda},\boldsymbol{\mu}) = f_l(\mathbf{x},\mathbf{y}) + \boldsymbol{\lambda}^T \mathbf{h}_l(\mathbf{x},\mathbf{y}) + \boldsymbol{\mu}^T \mathbf{g}_l(\mathbf{x},\mathbf{y})$$

Here, $L(\mathbf{x},\mathbf{y},\boldsymbol{\lambda},\boldsymbol{\mu})$ is the Lagrangian, $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the dual variables associated with the equality and inequality constraints, respectively. The obtained NLP is also known as a mathematical program with complementarity constraints (MPCC).

Note that, although the lower level problem is convex, the reformulated NLP Eq. (3.13a) is intrinsically nonconvex due to the complementary constraints Eq. (3.13h) and generally difficult to solve.

3.1.3 Mathematical Program with Complementarity Constraints (MPCC)

A general MPCC [FP97] can be stated as:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad f(x) \tag{3.14a}$$
$$s.t. \quad h(\mathbf{x}) = 0 \tag{3.14b}$$
$$g(\mathbf{x}) \leq 0 \tag{3.14c}$$
$$0 \leq Q(\mathbf{x}) \perp P(\mathbf{x}) \geq 0 \tag{3.14d}$$

The objective function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, the equality $h(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{n_h}$, and inequality constraints $g(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^{n_g}$ are assumed to be sufficiently smooth.

Eq. (3.14d) are the complementarity constraints enforcing nonnegativity and orthogonality of $Q(\mathbf{x})$ and $P(\mathbf{x})$. The notation used in 3.14d is short for:

$$0 \leq Q(\mathbf{x}) \perp P(\mathbf{x}) \geq 0 \Leftrightarrow 0 \leq Q(\mathbf{x}), \quad 0 \leq P(\mathbf{x}), \quad Q(\mathbf{x})^T P(\mathbf{x}) = 0 \tag{3.15}$$

Due to the complementarity constraints (3.14d), MPCCs are nonsmooth and nonconvex.

Because of the nonsmoothness and nonconvexity of the feasible set, MPCCs are particularly challenging to solve. Additionally, at every feasible point, ordinary constraint qualifiers like the LICQ or the Mangasarian-Fromovitz constraint qualification are violated [CKA95]. The failure to meet the constraint qualifiers makes it difficult to numerically solve problems like Eq. (3.14) directly, and nonlinear optimization solvers are likely to fail to find local optima without explicitly addressing the complementarity constraints.

Several strategies exist to solve MPCCs, the concepts most relevant to this work rely on a transformation of the complementarity constraints and will be introduced briefly. For an in-depth review, the reader is referred to [KLM20].

In order to better discuss the presented solution strategies, we consider the example 3.16 form [SS00]:

$$\min_{\mathbf{x}\in\mathbb{R}^2} \quad (x_1 - 1)^2 + (x_2 - 1)^2 \tag{3.16a}$$

$$s.t. \quad 0 \leq x_1 \perp x_2 \geq 0 \tag{3.16b}$$

A plot of Eq. (3.16) is illustrated in Section 3.1.3. The feasible set is indicated by the L-shaped solid black line. As can be seen, it is nonsmooth and nonconvex due to the kink at $x_1 = x_2 = 0$. The example contains two local minimizers at $(0, 1)$ and $(1, 0)$ and a local maximizer at the origin $(0, 0)$.

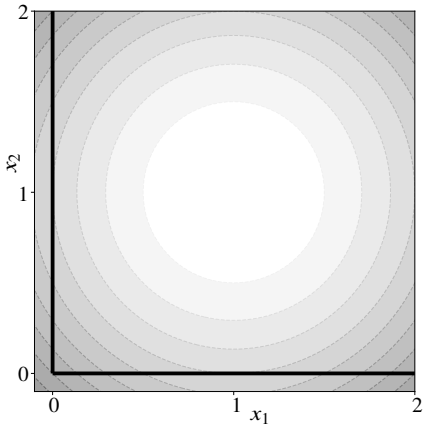


Figure 3.4: Contour lines of Eq. (3.16). The L-shaped solid black line indicates the feasible set. For this example, two local minimizers at $(0, 1)$ and $(1, 0)$ and a local maximizer at $(0, 0)$ exist.

The complementarity constraint 3.16b implies that either x_1 or x_2, must be 0 (or both). Hence, complementarity constraints can also be seen as a way how to model a combinatorial nature of the two variables.

Smoothing methods

A popular approach to solve MPCCs is to approximate the nonsmooth complementarity conditions 3.16b by smooth equality constraints [FJQ99; BS07; NAD20] as:

$$0 \leq x_1, \quad 0 \leq x_2, \quad x_1 x_2 = \epsilon \tag{3.17}$$

With $\epsilon > 0$, the constraints become smooth in x_1 and x_2.

Another widely used function to smooth the complimentarity constraint is the perturbed Fischer-Burmeister function [Kan96] which works particularly well for SQP methods [FLP98].

Relaxation method

Similar to smoothing methods, the core idea of relaxation methods [Sch01; RW04; HKS13] is to approximate the complementarity constraints. In contrast to smoothing methods, the complementarity constraints are replaced by inequality instead of equality constraints as:

$$0 \leq x_1, \quad 0 \leq x_2, \quad x_1 x_2 \leq \epsilon \tag{3.18}$$

With $\epsilon > 0$, the feasible set is enlarged, or relaxed, hence the name relaxation methods. The smaller ϵ is chosen, the closer a feasible point (x_1, x_2) is to achieving complementarity.

An advantage of the relaxation method over the smoothing method is that the original minimizers are still contained in the feasible set.

Penalty methods

For penalty methods [RW04; LLN06; Ani05; Hal+22], the complementarity constraints are removed from the set of constraints, and their violation is penalized in the objective function via a penalty term.

The transformed NLP then reads:

$$\min_{\mathbf{x}\in\mathbb{R}^2} \quad (x_1 - 1)^2 + (x_2 - 1)^2 + \rho_k \Phi(x_1, x_2) \tag{3.19a}$$

$$s.t. \quad x_1 \geq 0 \tag{3.19b}$$

$$x_2 \geq 0 \tag{3.19c}$$

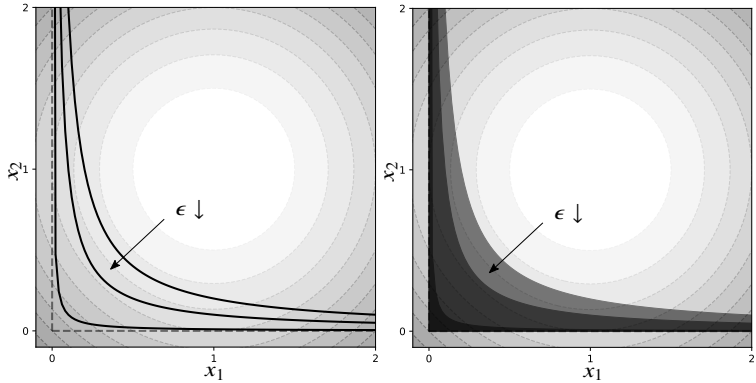
with e.g., $\Phi(x_1, x_2) = x_1 x_2$.

The penalty method generates a sequence of $\mathbf{x}_1^*, \mathbf{x}_2^*, \mathbf{x}_3^*, \ldots$ by solving Eq. (3.19) with increasing penalty weights $\rho_1 < \rho_2 < \rho_3 < \ldots$.

Since the feasible set does not change between iterations, the solution of iteration k can be used as an initial guess for $k + 1$ to warms start the NLP solver.

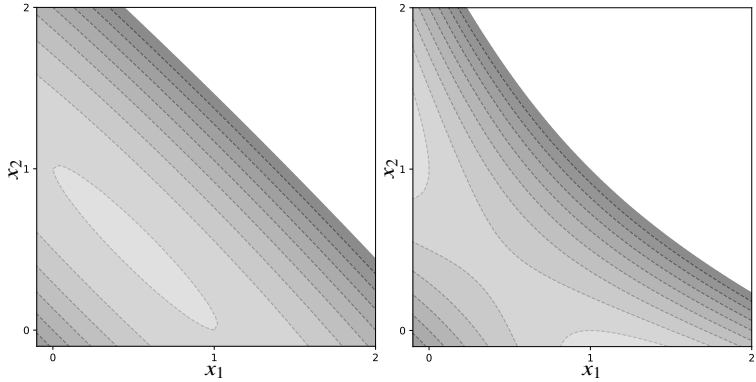
A drawback of these methods is that due to the, in general, nonconvexity of Φ the solver can get stuck in a local minimum, and increasing the penalty term ρ_k will not necessarily lead to less violation of the complementarity constraint. For more details on the convergence properties, the reader is referred to [HR04].

The presented strategies focus on reformulating the original MPCC into an approximately equivalent, regularized NLP where the constraint qualifiers are satisfied again. The reformulated problem can then be solved using common NLP solvers. In Fig. 3.5, it is illustrated how the different methods affect the objective function and feasible set of example 3.16.



(a) Feasible set for the smoothing method.

(b) Feasible set for the relaxation method.



(c) Contour lines of the objective function for a penalty weight of $\rho = 1.9$.

(d) Contour lines of the objective function for a penalty weight of $\rho = 3.0$.

Figure 3.5: Illustrated is how the presented strategies change the feasible set for the smoothing a) and the relaxation method b), as well as how the objective function changes in case of the penalty reformulation for different ρ. a) The smaller ϵ is chosen, the closer the smoothed constraint approximates the original complementarity constraint. The same holds for the relaxation method shown in b). With increased ρ, the nonconvexity of the reformulated objective function becomes visible, see c) and d).

3.2 Optimal Control

Optimization-based motion planning methods build upon optimal control problems (OCPs). An introduction into OCPs is provided in this section.

Let us consider the following dynamic system

$$\dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{3.20}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, and $\mathbf{u}(t) \in \mathbb{R}^m$ is the control vector.

The goal is to determine a control trajectory $\mathbf{u}(t)$ that brings the system from an initial state $\mathbf{x}(t = 0) = \mathbf{x}_0$ to a desired final state $\mathbf{x}(t_e)$. While there are many, possibly infinitely many, control trajectories that can achieve that, the subject of optimal control is finding $\mathbf{u}^*$ among the many feasible trajectories that minimizes a cost functional J.

In general, an OCP can be stated as:

$$\min_{\mathbf{u}(t)} J(\mathbf{u}(t)) = \min_{\mathbf{u}(t)} V(\mathbf{x}(t_e)) + \int_{t_0}^{t_e} l(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \tag{3.21a}$$

s.t.:

$$\dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{u}(t)) \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.21b}$$

$$\mathbf{h}(\mathbf{x}(t_e), t_e) = 0 \tag{3.21c}$$

$$\mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t) \leq 0 \tag{3.21d}$$

The cost functional J consists of the terminal costs for the final state $V(\mathbf{x}(t_e))$ and the running costs $l(\mathbf{x}(t), \mathbf{u}(t), t)$. The equality constraints 3.21b enforce the system dynamic 3.20 and describe how the system evolves in time for a certain choice of $\mathbf{u}(t)$. Further, 3.21c describes the constraints on the final state. Note that this also includes the special cases of a free end state and a free end time. General inequality constraints, as well as bound constraints on inputs and states, are formulated in Eq. (3.21d).

Generally, there are three approaches to solve the continuous optimal control problem stated in Eq. (3.21), see also the upper row of Fig. 3.6. We here follow the outline stated in [Die+06].

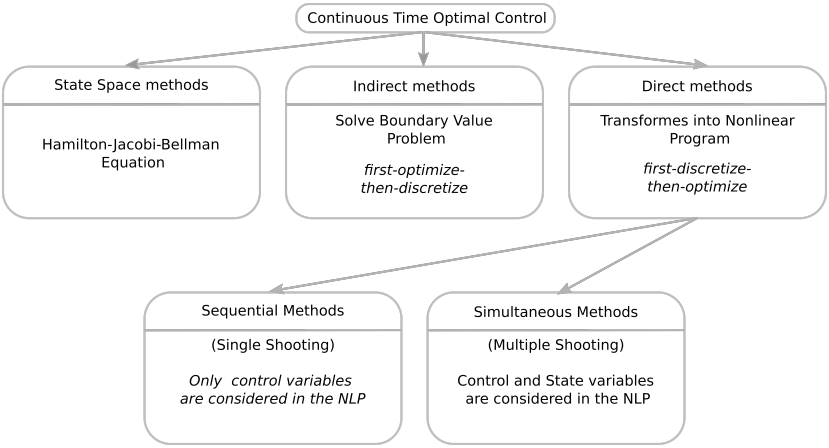


Figure 3.6: Different methods to solve the continuous optimal control problem.

State Space methods State space methods build upon the principle of optimality, stating that each subsection of the optimal trajectory also needs to be optimal. While this is the basis of dynamic programming in discrete time, this formulation leads to the so-called *Hamilton-Jacobi-Bellman* equation in continuous time. Methods to numerically solve this state space equation exist, but they heavily suffer from the curse of dimensionality and are therefore only applicable for small state spaces.

Indirect methods These methods use necessary conditions for optimality of the infinite-dimensional problem to formulate a boundary value problem, which is then solved numerically in a subsequent step. Since optimality conditions are obtained in a continuous form, and the problem is then discretized, indirect methods are often referred to as *first optimize then discretize*. Well-known formulations of indirect methods are the Euler-Lagrange differential equations and the Pontryagin Maximum Principle. Although indirect methods typically provide highly accurate solutions, a significant downside is that the underlying differential equations are often hard to solve and therefore hardly applicable for complex applications.

Direct methods Direct methods are often referred to as *first discretize, then optimize* since they transform the continuous time problem into a discrete time problem, which is then optimized. This transformation is achieved by finitely parameterizing the infinite-dimensional control trajectory such that the original problem is approximated by a NLP. The obtained NLP can then be solved efficiently by structure exploiting solvers. Roughly speaking, instead of solving an infinite-dimensional problem in function space, a finite-dimensional vector space problem is solved.

A major advantage of direct methods is that all sorts of constraints can be easily considered. This makes discrete methods by far the most widely used for real-world applications.

While all direct methods are based on a finite-dimensional parametrization of the input trajectory in some form, they differ in how the state trajectory is considered.

Two major classes of direct methods are *sequential* and *simultaneous* methods. For both methods, the continuous time interval $t \in [t_0, t_e]$ is first discretized into N steps. Further, the input function $\mathbf{u}(t)$ is parameterized on the N subintervals $[t_k, t_{k+1})$ by a finite-dimensional vector, typically polynomials or piecewise constant functions in the simplest case.

For sequential methods, the state variables are obtained by numerical integration of the input variables. This integration is not necessarily part of the OCP formulation. Thus, no explicit state variables are contained in the NLP. Sequential approaches lead to relatively small NLPs with densely populated derivative matrices.

Simultaneous methods also discretize the state trajectory $\mathbf{x}(t)$ and introduce a finite-dimensional state vector into the NLP. Opos to sequential methods, the numerical integration is part of the NLP, and equality constraints enforce the vehicle dynamics. Simultaneous approaches lead to larger NLPs with sparsely populated derivative matrices. Even though the NLPs are larger, due to structure exploding solvers, simultaneous methods often outperform sequential methods in terms of computational time.

Due to their advantages, direct simultaneous methods are used in this work. We further parametrize the control trajectory by a piecewise constant function to

transform the OCPs in chapter 4 and 5. With the planning horizon discretized into N steps, we approximate OCPs like 3.21 with the following NLP:

$$\min_{\mathbf{u}_{0:N-1},\, \mathbf{x}_{1:N}} J(\mathbf{u}_{0:N-1},\, \mathbf{x}_{1:N}) = \min_{\mathbf{u}_{0:N-1},\, \mathbf{x}_{1:N}} V(\mathbf{x}_N) + \sum_{k=0}^{N-1} l(\mathbf{x}_k, \mathbf{u}_k) \tag{3.22a}$$

$$\text{s. t.:} \tag{3.22b}$$

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{3.22c}$$

$$\mathbf{x}_0 = \hat{\mathbf{x}} \tag{3.22d}$$

$$\mathbf{g}(\mathbf{x}_k, \mathbf{u}_k) \leq 0 \tag{3.22e}$$

Here, $\hat{\mathbf{x}}$ denotes the initial state of the system.

3.3 Model Predictive Control

The solution of the OCP described in Section 3.2 is a control trajectory $\mathbf{u}(t)$. Directly applying this precomputed sequence of controls to the real system is called *open-loop control*. Doing so will likely yield unsatisfying results since, typically, the model used during optimization and the real system does not entirely coincide. Also, deviations from the desired state trajectory, e.g., due to unforeseen external disturbances during the execution, are not considered. Simply applying open-loop control, the system's final state will most likely be different from the intended one.

Consider, for example, the steering control of a car. Let us assume a straight road, and the goal is to keep following the road. Starting with the vehicle placed in the middle of the lane, the solution of the OCP would yield constant 0 steering. Appling these steering actions for a longer time will most likely cause the car to leave the road. This is due to the fact that no feedback is involved from the initial state onwards. Model errors and disturbances, e.g., uneven road or side winds, are not considered.

In contrast to *open-loop control* where the control, trajectory $\mathbf{u}(t)$ only depends on time, in *closed-loop control* or *feedback control*, the control trajectory also considers the state. Deviations of the state trajectory due to model errors

or external disturbances can then be accounted for, and the performance is improved considerably.

Ideally, we would like to compute the optimal control feedback as a function of all possible states, which can be described by:

$$\mathbf{u} = K(\mathbf{x}(t), t) \tag{3.23}$$

Except for special cases, computing the optimal input as a feedback is unfortunately very hard or even not possible. Thanks to powerful numerical methods, we can approximate a closed-loop control by repeatedly solving the OCP as an open-loop problem, starting from the current state. The control action is then used to actuate the system until the next OCP solution is available. This approximation technique is known as MPC.

Standard MPC implementations consist of the following four steps, also visualized in Fig. 3.7

1. Observe the current state $\mathbf{x}_0$
2. Predict and optimize the system's state on a limited horizon with a length N, starting at $\mathbf{x}_0$.
3. Apply the first control action $\mathbf{u}_0$ of the open-loop solution to the real system.
4. Move the horizon forward by one step and repeat the process.

Due to the moving horizon character, MPC is also known as *receding horizon control*.

Compared to other control strategies, MPC approaches are computationally demanding. To allow for fast computation, often simplified models of the system are used, and only a reduced prediction horizon is considered. However, care must be taken when a limited time horizon is used. If the length is chosen too short, the MPC might suffer from stability issues.

The presented introduction only scratches the surface of the broad research field of MPC. The interested reader is referred to [MGK07; RMD20] for a deeper inside to the topic.

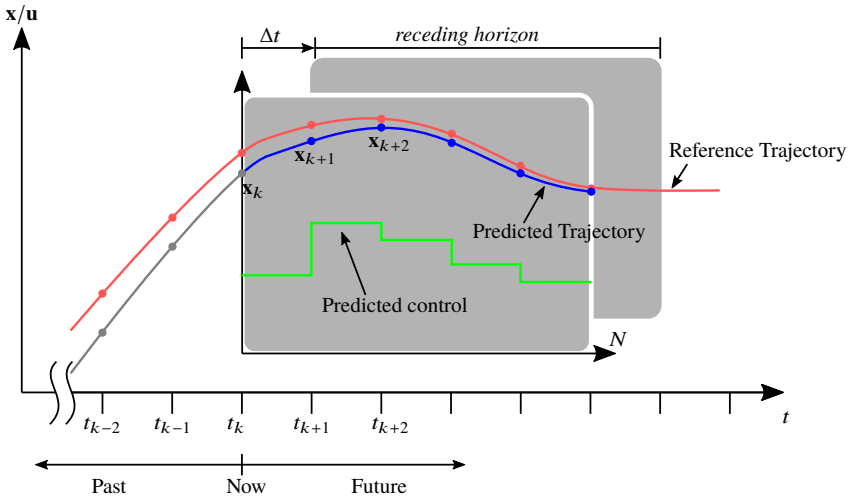


Figure 3.7: Core mechanism of MPC. First, the OCP is solved for a limited time horizon N starting from the current state. Then, only the first element of the solution vector $\mathbf{u}^*$ is applied to the system before the horizon is moved one step forward and the process is repeated.

4 Cooperative Interaction-Aware Motion Planning using Multi-Agent Trajectory Planning

This chapter presents a cooperative interaction-aware motion planner based on a multi-agent formulation. The approach aims to resolve the overly conservative behavior of pipeline approaches by planning for multiple vehicles simultaneously and thereby coupling prediction and planning. This structure enables the ego vehicle to anticipate how other drivers might react to its planed actions.

The central element of the presented planner is a novel multi-agent trajectory planning formulation, presented in Section 4.1. The major contribution of this formulation is its ability to overcome the typical limitations of optimization-based methods, namely the combinatorial problem which originates from the non-convexity of motion planning, as presented in Section 2.4. This is achieved by formulating the planning task as a mixed-integer quadratic programming (MIQP) in which a flat output of the vehicle is represented by continuous variables and the maneuver variants are represented by discrete variables via collision avoidance constraints. The resulting method guarantees a global optimum, without the restriction of a discretized search space. The approach is also particularly well suited for the insertion of logical constraints, such as those imposed by traffic regulations.

In Section 4.2, a framework based on the multi-agent trajectory planning formulation is presented, that enables an automated vehicle to generate interaction-aware trajectories, and therefore driving in dense traffic, see Fig. 4.1.

To cope with the uncertain behavior of human drivers different intention models are defined. These intentions are modeled in a generic way, which makes them independent of the traffic scenario or map layout. The concept is evaluated in a simulation environment, demonstrating its ability to plan interactive maneuvers and its ability to cope with uncertain human driving behavior.

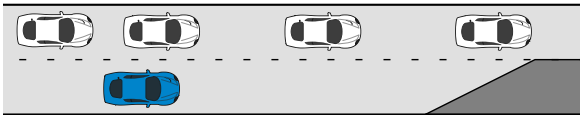


Figure 4.1: Merge scenario in dense traffic. Due to the narrow gaps a planner following a pipeline architecture would not allow for a successfully merge and the ego vehicle (blue) would stop at the end of the lane. Treating prediction and planning jointly enables the ego vehicle to consider interactions, and allows to anticipate that other drivers might open a gap.

4.1 Cooperative Multiple Vehicle Trajectory Planning with MIQP

In this section, a novel cooperative multiple vehicle trajectory planner based on [BL18] is presented. The problem is formulated as a MIQP which yields globally optimal trajectories for a group of automated vehicles $\mathcal{V} = V^1, \ldots, V^{n^v}$. Cooperative behavior among the vehicles is introduced by optimizing a joint objective function.

The key element of this planner is the formulation of the collision avoidance constraints via binary variables. This formulation allows to overcome the limitations optimization-based methods typically have, and the solution is not bound to the homotopy of its initialization.

The trajectory planning for the group of vehicles $\mathcal{V} = V^1, \dots, V^{n^v}$ is performed in a MPC fashion. Let $\mathbf{x}^n$ be the state and $\mathbf{u}^n$ be the input of vehicle $V^n \in \mathcal{V}$, the underlying OCP can be stated as:

$$\min_{\mathbf{u}_{0:N-1}^{1:n^v}, \mathbf{x}_{1:N}^{1:n^v}} J_{\text{total}} = \min_{\mathbf{u}_{0:N-1}^{1:n^v}, \mathbf{x}_{1:N}^{1:n^v}} \sum_{n=1}^{n^v} w^n \left(J_{\text{ref}}^n(\mathbf{x}_{1:N}^{1:n^v}) + J_u^n(\mathbf{u}_{0:N-1}^n) \right) \tag{4.1a}$$

s.t. :

$$\forall n \in \{1, \dots, n^v\}$$

$$\mathbf{x}_{k+1}^n = \mathbf{f}(\mathbf{x}_k^n, \mathbf{u}_k^n) \qquad k = 0, \dots, N-1 \tag{4.1b}$$

$$\mathbf{x}_0^n = \hat{\mathbf{x}}^n \tag{4.1c}$$

$$\underline{\mathbf{g}_{\text{dyn}}} \leq \mathbf{g}_{\text{dyn}}(\mathbf{x}_k^n) \leq \overline{\mathbf{g}_{\text{dyn}}} \qquad k = 1, \dots, N \tag{4.1d}$$

$$\mathbf{g}_{\text{col}}(\mathbf{x}_k^n, \mathbf{x}_k^m) \leq 0 \qquad \forall m \in \{n+1, \dots, n^v\} k = 1, \dots, N \tag{4.1e}$$

$$\mathbf{g}_{\text{obs}}(\mathbf{x}_k^n) \leq 0 \qquad k = 1, \dots, N \tag{4.1f}$$

$$\underline{\mathbf{x}} \leq \mathbf{x}_k^n \leq \overline{\mathbf{x}} \qquad k = 1, \dots, N \tag{4.1g}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k^n \leq \overline{\mathbf{u}} \qquad k = 0, \dots, N-1 \tag{4.1h}$$

Here, we utilize a direct simultaneous method to transform the general continuous OCP of motion planning, see also Section 3.2. Therefore, the optimization horizon T is discretized into N steps, assuming a piecewise constant input.

The OCP objective function J_{total} is a linear combination of the vehicle's individual costs and contains penalties for any control efforts J_u^n as well as costs for deviations from a reference state J_{ref}^n. Further details about the formulations and the definition of a suitable reference state are given in Section 4.1.3.

The equality constraints in Eq. (4.1b) are imposed by the vehicle dynamics. The underlying vehicle model is described in Section 4.1.1. Further, Eq. (4.1c) initializes the system at the current state. The inequality constraints $\mathbf{g}_{\text{dyn}}$ are introduced to ensure that the generated trajectories are dynamically feasible.

The inequality constraints in Eq. (4.1e) and Eq. (4.1f) are used for collision avoidance among the controlled vehicles $\mathbf{g}_{\text{col}}$ as well as for collision avoidance with obstacles $\mathbf{g}_{\text{obs}}$, respectively. To account for physical limitations of the real

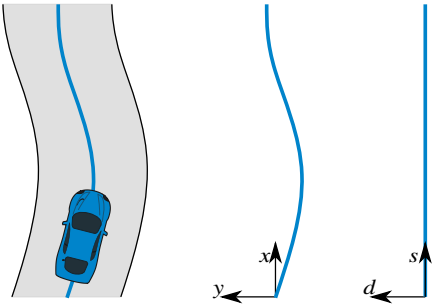


Figure 4.2: Curvy road with the center of the lane as a reference. In the middle, the path is represented in Cartesian coordinates. On the right, the same path is given in road-aligned Frenet coordiantes.

system, additional bound constraints on the states, Eq. (4.1g), and the inputs, Eq. (4.1h), are considered.

Note that except for the collision constraints $\mathbf{g}_{\text{col}}$, which couple the controlled vehicles, all costs, and constraints can be formulated for each vehicle individually.

The approach is intended for a group of connected, automated vehicles in general on-road scenarios. Moreover, ordinary driving scenarios are the focus, where the vehicles are not operated close to their dynamic limits.

The road within a planning horizon T is assumed to be sufficiently straight, so forces and dynamic effects due to the road curvature can be neglected. The coordinates of a vehicle can then be described in a road-aligned *Frenet* frame [Wer+10], where s denotes the longitudinal and d the lateral direction. Fig. 4.2 illustrates the transformation into a road-aligned frame.

In the following, the individual components of the cooperative multi-agent trajectory planning problem are explained in detail. Further, the assumptions which are necessary to obtain a MIQP are stated.

4.1.1 Vehicle Model

An important design choice of optimization-based methods is the vehicle model. The vehicle model should capture the essential characteristics of the real system without modeling unnecessary complexity. Since this approach is intended for ordinary on-road scenarios, rather than emergency situations where the vehicle is operated close to its dynamic limits, a simple point mass model is chosen, see Section 2.6.1.

In contrast to similar works [Sch+01; ES17], where first and second-order point-mass models were used, we follow the approach of [Qia+16] and choose a third order model. This formulation ensures a continuous acceleration profile and, as a result, a continuous yaw rate of the generated trajectory, which a low-level controller can track smoothly.

The motion of a vehicle is formulated separately in the longitudinal and the lateral direction, denoted by $\square_s$ and $\square_d$, respectively. The state $\mathbf{x}^n(t) \in \mathbb{R}^6$ and input $\mathbf{u}^n(t) \in \mathbb{R}^2$ of vehicle v^n are given by:

$$\mathbf{x}^n(t) = [p_s^n(t), v_s^n(t), a_s^n(t), p_d^n(t), v_d^n(t), a_d^n(t)]^T \tag{4.2}$$

$$\mathbf{u}^n(t) = [j_s^n(t), j_d^n(t)]^T \tag{4.3}$$

where p is the position, v the speed, a the acceleration, and j the jerk.

The system dynamics can be stated as:

$$\dot{\mathbf{x}}^n(t) = \begin{pmatrix} \mathbf{A} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{A} \end{pmatrix} \mathbf{x}^n(t) + \begin{pmatrix} \mathbf{B} & \mathbf{0}^{3\times1} \\ \mathbf{0}^{3\times1} & \mathbf{B} \end{pmatrix} \mathbf{u}^n(t) \tag{4.4}$$

with

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

The equation describes the evolution of the system in continuous time and is stated in the form of an ordinary differential equation $\dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{u}(t))$. To consider the vehicle dynamics in the discrete optimal control problem Eq. (4.1), a discrete transition model $\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k)$ is necessary.

With piecewise constant inputs, a continuous linear system given by $\dot{\mathbf{x}}(t) = \mathbf{A}_{\text{con}}\mathbf{x}(t) + \mathbf{B}_{\text{con}}\mathbf{u}(t)$ can be discretized by considering the analytical solution over a fixed length discretization step T. The result is a linear state-space equation of the form $\mathbf{x}_{k+1} = \mathbf{A}_{\text{dis}}\mathbf{x}_k + \mathbf{B}_{\text{dis}}\mathbf{u}_k$.

The matrices for the discrete system $\mathbf{A}_{\text{dis}}$, $\mathbf{B}_{\text{dis}}$ are obtained by:

$$\mathbf{A}_{\text{dis}} = e^{\mathbf{A}_{\text{con}}T} \tag{4.5}$$

$$\mathbf{B}_{\text{dis}} = \mathbf{B}_{\text{con}} \int_0^T e^{\mathbf{A}_{\text{con}}t}\, dt \tag{4.6}$$

With τ as the discretization step, the discrete system dynamics are given by:

$$\mathbf{x}_{k+1}^n = \begin{pmatrix} \mathbf{A}^d & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{A}^d \end{pmatrix} \mathbf{x}_k^n + \begin{pmatrix} \mathbf{B}^d & \mathbf{0}^{3\times1} \\ \mathbf{0}^{3\times1} & \mathbf{B}^d \end{pmatrix} \mathbf{u}_k^n \tag{4.7}$$

with

$$\mathbf{A}^d = \begin{pmatrix} 1 & \tau & \frac{1}{2}\tau^2 \\ 0 & 1 & \tau \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{B}^d = \begin{pmatrix} \frac{1}{6}\tau^3 \\ \frac{1}{2}\tau^2 \\ \tau \end{pmatrix}$$

To account for the dynamic limitations of the real system, the following bound constraints on the velocity v^n, the acceleration a^n, as well as the jerk j^n are introduced:

$$\mathbf{x}_k^n \in [\underline{\mathbf{x}}_k^n, \overline{\mathbf{x}}_k^n] \tag{4.8}$$

$$\mathbf{u}_k^n \in [\underline{\mathbf{u}}_k^n, \overline{\mathbf{u}}_k^n] \tag{4.9}$$

The minimum and maximum values are denoted by $\underline{\square}$ and $\overline{\square}$, respectively.

So far, the longitudinal and lateral motion are treated separately, and the non-holonomic characteristic of the real system is ignored. To avoid generating dynamically infeasible trajectories, e.g., moving sideways without moving forward, the additional constraints $\mathbf{g}_{\text{dyn}}$ are introduced. These constraints approximate the non-holonomic property by coupling the velocity components via the heading Θ of the vehicle.

Though Θ is not explicitly contained in the state, it can be reconstructed from the velocities as $\Theta = \arctan(\frac{v_d}{v_s})$. The coupling of the longitudinal and lateral dynamics can then be enforced by limiting the heading $\Theta \in [\underline{\Theta}, \overline{\Theta}]$ with:

$$v_d \in [v_s \tan(\underline{\Theta}), v_s \tan(\overline{\Theta})] \tag{4.10}$$

4.1.2 Collision Avoidance Constraints

In the following, the mixed-integer based collision avoidance constraints are introduced. Road-aligned rectangles approximate the shape of vehicles and obstacles. Further, obstacles and road boundaries are sufficiently enlarged so that the controlled vehicles can be represented by single points. More complex polygonal shape approximations are possible; however, they come with an increased computational burden and are therefore not considered in this approach.

Collisions avoidance among controlled vehicles

Let us consider any pair of vehicles V^n and $V^m \in \mathcal{V}$. Collisions can be avoided by ensuring that both vehicles do not occupy the same geometric area at the same time t_k for all $k = 1, \ldots, N$. This statement can be expressed as the following set of logical constraints:

$$\begin{aligned}
& p_s^n \leq p_s^m - l_{\min}^{n,m} && \text{(4.11a)}\\
\vee \quad & p_s^n \geq p_s^m + l_{\min}^{n,m} && \text{(4.11b)}\\
\vee \quad & p_d^n \leq p_d^m - w_{\min}^{n,m} && \text{(4.11c)}\\
\vee \quad & p_d^n \geq p_d^m + w_{\min}^{n,m} && \text{(4.11d)}
\end{aligned}$$

Here, $l_{\min}^{n,m}$ is the minimum longitudinal, and $w_{\min}^{n,m}$ is the minimum lateral distance necessary between the two vehicles. Additional safety margins can be incorporated by simply increasing $l_{\min}^{n,m}$ and $w_{\min}^{n,m}$. E.g., to cope with sensor noise or to increase the perceived safety by bigger clearances.

The set of logical OR constraints in Eq. (4.11a) can be reformulated into a conjunction of linear inequality constraints by applying the so-called Big-M method [WVG03].

A binary variable δ is introduced for each statement in Eq. (4.11a), indicating whether this statement is satisfied $\delta = 1$ or not $\delta = 0$. The corresponding linear inequality is obtained by adding or subtracting an application-specific big positive constant M^{big} depending on the value of δ. For each time step, four binary variables are introduced. The collision avoidance constraints $\mathbf{g}_{\text{col}}$ can then be written as:

$$p^n_{s,k} \leq p^m_{s,k} - l^{n,m}_{\min} + (1 - \delta^{n,m}_{1,k})M^{\text{big}} \tag{4.12a}$$

$$p^n_{s,k} \geq p^m_{s,k} + l^{n,m}_{\min} - (1 - \delta^{n,m}_{2,k})M^{\text{big}} \tag{4.12b}$$

$$p^n_{d,k} \leq p^m_{d,k} - d^{n,m}_{\min} + (1 - \delta^{n,m}_{3,k})M^{\text{big}} \tag{4.12c}$$

$$p^n_{d,k} \geq p^m_{d,k} + d^{n,m}_{\min} - (1 - \delta^{n,m}_{4,k})M^{\text{big}} \tag{4.12d}$$

$$\sum_{i=1}^{4} \delta^{n,m}_{i,k} \geq 1 \tag{4.12e}$$

The last constraint, Eq. (4.12e), ensures that at least one of the original OR conditions is always satisfied. The collision avoidance constraints are illustrated in Fig. 4.3.

To keep the formulation efficient and the computation time low, care must be taken when choosing M^{big}. The value should be chosen as small as possible such that the number of subproblems of the branch-and-bound algorithm is kept low, see Section 3.1.1. On the other hand, the value has to be big enough to capture all solutions of interest.

Collisions avoidance with obstacles

The collision avoidance constraints between controlled vehicles and static or dynamic obstacles, $\mathbf{g}_{\text{obs}}$, are treated the same way. Again, a vehicle and an obstacle cannot occupy the same space simultaneously. The major difference is that the state variables of the obstacles are not part of the optimization problem

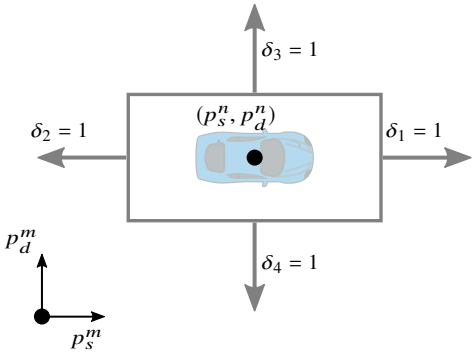


Figure 4.3: Collision avoidance constraints between V^n and V^m. The collision free regions are indicated by the arrows and the corresponding values of the binary variables δ_i.

and are assumed to be given. For each obstacle-vehicle pair, again, four binary variables $\delta_i^{o,n}$ per time step k are introduced as:

$$p_{s,k}^n \leq p_{s,k}^o - l_{\min}^{n,m} + (1 - \delta_{1,k}^{n,o})M^{\text{big}} \tag{4.13a}$$

$$p_{s,k}^n \geq p_{s,k}^o + l_{\min}^{n,m} - (1 - \delta_{2,k}^{n,o})M^{\text{big}} \tag{4.13b}$$

$$p_{d,k}^n \leq p_{d,k}^o - w_{\min}^{n,m} + (1 - \delta_{3,k}^{n,o})M^{\text{big}} \tag{4.13c}$$

$$p_{d,k}^n \geq p_{d,k}^o + w_{\min}^{n,m} - (1 - \delta_{4,k}^{n,o})M^{\text{big}} \tag{4.13d}$$

$$\sum_{i=1}^{4} \delta_{i,k}^{n,o} \geq 1. \tag{4.13e}$$

Vehicles not considered in the multi-agent optimization problem are modeled as moving obstacles. There motion $(p_{s,k}^o, p_{d,k}^o)$ is assumed to be provided by a separate prediction module.

4.1.3 Collective Cost Function

The objective of the multi-agent trajectory planning approach is to generate comfortable and collision-free trajectories for an ensemble of vehicles. Besides these requirements, the vehicles also need to make progress along their paths to

be time-efficient. These objectives are formulated in the two cost terms J_{u}^{n} and J_{ref}^{n} in Eq. (4.1), which are formulated for each vehicle $V^n \in \mathcal{V}$ individually.

J_{u}^{n} penalizes the jerk quadratically and is given by:

$$J_{u}^{n}(\mathbf{u}^n) = {\mathbf{u}^n}^T R^n \mathbf{u}^n = \|\mathbf{u}_k^n\|_{R^n}^2 \tag{4.14}$$

J_{ref}^{n} penalizes any deviation from a reference state, also quadratically, and is given by:

$$J_{\text{ref}}^{n}(\mathbf{x}^n) = (\mathbf{x}^n - \mathbf{x}_{\text{ref}}^n)^T Q^n (\mathbf{x}^n - \mathbf{x}_{\text{ref}}^n) = \|\mathbf{x}^n - \mathbf{x}_{\text{ref}}^n\|_{Q^n}^2 \tag{4.15}$$

R^n and Q^n are positive and semi-positive definite weighting matrices used to model different driving preferences. E.g., when compared to a comfortable driver, a sporty driver might choose a lower penalty for accelerations.

The reference state x_{ref} can be individual for each vehicle and is given as:

$$\begin{aligned} \mathbf{x}_{\text{ref}}^n &= [p_{s,\text{ref}}^n, v_{s,\text{ref}}^n, a_{s,\text{ref}}^n, p_{d,\text{ref}}^n, v_{d,\text{ref}}^n, a_{d,\text{ref}}^n]^T \\ &= [0, v_{s,\text{ref}}^n, 0, p_{d,\text{ref}}^n, 0, 0]^T \end{aligned} \tag{4.16}$$

The desired velocity $v_{s,\text{ref}}^n$ is context-dependent, e.g., to the speed limit or a desired maximal velocity. The driving lane is chosen via $p_{d,\text{ref}}^n$, representing the corresponding lane's centerline. The absolute value of $p_{s,\text{ref}}^n$ is irrelevant and can be set to zero.

The joint cost function is obtained by a linear combination of the vehicles' individual costs and can be written as:

$$\min_{\mathbf{u}_{0:N-1}^n, \mathbf{x}_{0:N}^n \forall n \in 1,\dots,n^v} \sum_{n=1}^{n^v} w^n \left(\sum_{k=1}^{N} \|\mathbf{x}_k^n - \mathbf{x}_{k,\text{ref}}^n\|_{Q^n}^2 + \sum_{k=0}^{N-1} \|\mathbf{u}_k^n\|_{R^n}^2 \right) \tag{4.17}$$

w^n are used to weigh the individual vehicles' costs relative to each other. This way, traffic precedence or different levels of cooperativity can be modeled.

Unlike previous works [Sch+01], a quadratic cost function is chosen such that deviations from the desired state increase more than linear. This should prevent a solution where one vehicle has a high disadvantage while multiple others

have small advantages. E.g., 101 vehicles save one second each while crossing an intersection, but therefore one vehicle has to wait for 100 seconds, which would be a valid solution for a linear cost function that naively minimizes the overall travel time.

If the desired state $\mathbf{x}_{\text{ref}}^{n}$ is independent or linearly dependent on other optimization variables, the formulation yields a MIQP, see Section 3.1.1.

Please note that the *OR* conditions in Eq. (4.11a) could also be approximated by nonlinear constraints and be solved with continuous NLP solvers. These solvers, however, only find local optima, whereas the formulation with mixed integers ensures the global optimum.

4.1.4 Evaluation

In this section, numerical experiments are provided to demonstrate the feasibility and advantages of the novel multi-agent trajectory planning formulation. A challenging overtaking scenario on a rural road with oncoming traffic, as shown in Fig. 4.4, is selected to evaluate the performance. The method is compared to priority-based and individual motion planning.

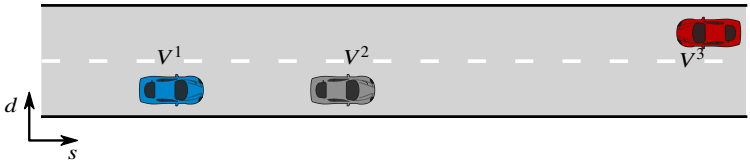


Figure 4.4: Overtaking scenario on a two-lane rural road. The blue vehicle has to perform an overtaking maneuver to continue with a higher velocity. To continue with a higher velocity the blue vehicle has to perform a overtaking maneuver. This is a challenging task due to the oncoming red vehicle.

Scenario

The considered scenario involves three vehicles on a two-lane rural road. The blue vehicle,V^1, has a higher desired velocity and is approaching the slower driving grey vehicle, V^2. Due to the difference in the reference speeds, an

overtaking maneuver would be favorable for V^1. This, however, is not possible without further ado due to the oncoming vehicle V^3 on the opposite lane.

The time horizon of the simulation is set to $T = 20$s with a discretization step size of $\tau = 0.5$s. The vehicles initially drive with their desired velocity and have a width of $w = 2\,\mathrm{m}$ and a length of $l = 5\,\mathrm{m}$. Additional parameters are listed in Table 4.1.

Ref. Velocities	Bounds $V^{1,2,3}$	Weights $V^{1,2,3}$
$v^1_{s,\mathrm{ref}} = 25\,\frac{\mathrm{m}}{\mathrm{s}}$ $v^2_{s,\mathrm{ref}} = 15\,\frac{\mathrm{m}}{\mathrm{s}}$ $v^3_{s,\mathrm{ref}} = 15\,\frac{\mathrm{m}}{\mathrm{s}}$	$\overline{\mathbf{x}} = [0, 30\,\frac{\mathrm{m}}{\mathrm{s}}, 3\,\frac{\mathrm{m}}{\mathrm{s}^2}, 6\,\mathrm{m}, 2\,\frac{\mathrm{m}}{\mathrm{s}}, 2\,\frac{\mathrm{m}}{\mathrm{s}^2}]^T$ $\underline{\mathbf{x}} = [0, 0, -4\,\frac{\mathrm{m}}{\mathrm{s}^2}, 1\,\mathrm{m}, -2\,\frac{\mathrm{m}}{\mathrm{s}}, -2\,\frac{\mathrm{m}}{\mathrm{s}^2}]^T$ $\overline{\mathbf{u}} = [3\,\frac{\mathrm{m}}{\mathrm{s}^3}, 2\,\frac{\mathrm{m}}{\mathrm{s}^3}]^T$ $\underline{\mathbf{u}} = [-6\,\frac{\mathrm{m}}{\mathrm{s}^3}, -2\,\frac{\mathrm{m}}{\mathrm{s}^3}]^T$ $\overline{\Theta} = 0.4$ $\underline{\Theta} = -\overline{\Theta}$	$diag(Q) = \{0, 1, 2, 1, 2, 4\}$ $diag(R) = \{4, 4\}$

Table 4.1: Reference velocities, bounds, and weights used for the numerical experiment.

In the following, the results of the different planning methods are explained and compared.

Individual Motion Planning

In the case of individual motion planning, the multi-agent problem reduces to a set of decoupled trajectory planning problems, where each vehicle solely minimizes an individual cost function, not considering the costs of others. Since V^2 and V^3 are already driving with their desired velocity, they will continue without any change.

V^1 has two options, which can be described by the temporal sequence the vehicles pass each other. It can either strongly accelerate to overtake V^2 before passing V^3, which might not be possible due to a lack of engine power, or it has to brake and overtake V^2 after passing V^3.

For the given initial configuration, V^1 has to decelerate sharply and wait until V^3 has passed before overtaking V^2. This deceleration and staying behind a

slower vehicle causes significant costs for V^1. The resulting trajectory is shown in Fig. 4.6a in the discussion section.

Priority-Based Planning

Priority-based planning is a common approach for multi-agent motion planning. Here, a priority list is calculated to decompose the multi-agent into a series of consecutive single-agent trajectory planning problems. The vehicle with the highest priority plans its motion first, not considering the plans of others. Then the vehicle with the second-highest priority plans its motion, only considering the plan of the first vehicle, and so. The concept is visualized in Fig. 4.5.

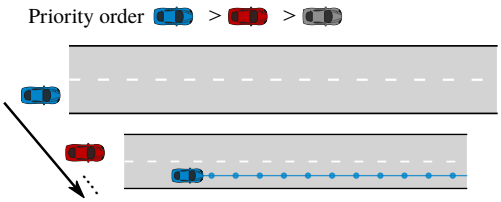


Figure 4.5: Concept of priority-based planning. The blue vehicle has the highest priority and therefore plans first, not considering other vehicles. The red vehicle plans second, only considering the plans of the blue vehicle. This pattern continues until the last vehicle has planned its motion.

While this decomposition decreases the computational complexity, it also limits the solution space. Even with an optimal priority assigned, optimality can not be guaranteed, and in some cases, the simplified problem becomes infeasible even though a solution in the full configuration space exists [FB11].

The results of the priority-based method are obtained by calculating all possible priority permutations and selecting the best one. This way, the results are independent of a specific priority selection and represent the best achievable outcome.

For the overtaking scenario, the two best priority assignments are either $V^1 > V^3 > V^2$ or $V^3 > V^1 > V^2$. Since V^2 has the lowest priority, it plans its motion last, adapting to the plans of V^1 and V^3. For the considered scenario, this means it would be beneficial for the overall costs if the slower driving grey

vehicle changes lanes, so the red car can continue to drive without slowing down. The resulting trajectories are visualized in Fig. 4.6b.

Cooperative MIQP Formulation

For the simulation, the cooperative weights w^n in Eq. (4.17) are set to one for each vehicle $V^n \in \mathcal{V}$. As a result, every vehicle's costs are considered equally, thus enabling the ensemble to reach the highest level of cooperative behavior according to [Bur+17]. In contrast to existing methods, the proposed MIQP approach exploits the entire configuration space and yields the best solution across all homotopy classes. In the case of an overtaking scenario, this can be an ad-hoc creation of a third lane, as shown in Fig. 4.6c.

4.1.5 Discussion

Fig. 4.7 shows the vehicles' individual costs and longitudinal velocity profiles generated by the different approaches. When comparing them, the individual motion planning yields the highest overall costs. On the other hand, the priority-based approach yields slightly lower collective costs, mainly because big deviations from the reference velocities of any vehicle can be mitigated, see Fig. 4.7b.

The major drawback of priority-based methods is that the vehicles do not cooperate, e.g., the vehicle with the highest priority does not adapt its behavior to any other vehicle even if small own disadvantages would enable others to improve their trajectories significantly.

By creating an ad-hoc third lane, the proposed cooperative MIQP approach yields the best solution, see Fig. 4.7c. This way, all vehicles can stick to their reference velocities, and strong acceleration or deceleration can be avoided.

By looking at the costs, one can see that for individual as well as priority-based planning, all costs were carried out by a single vehicle. In the MIQP formulation, the costs are distributed among all vehicles, which leads to a significant reduction in the overall costs for the ensemble.

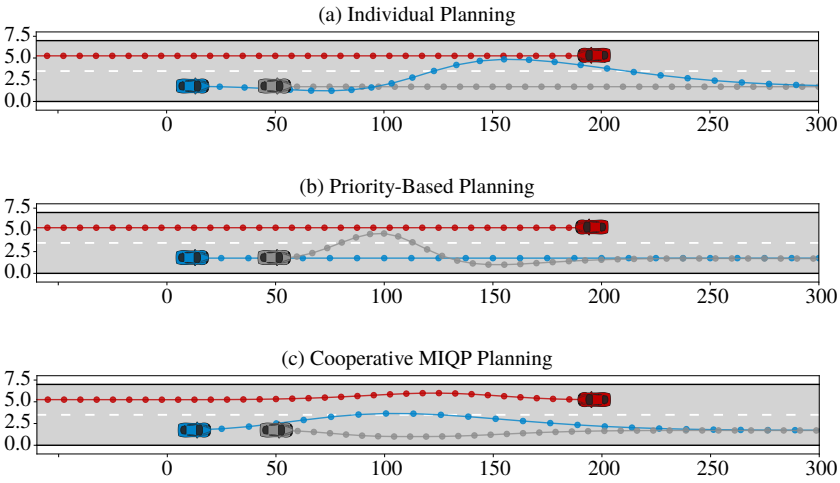
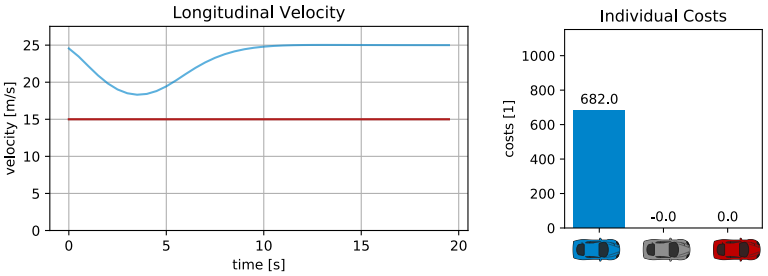


Figure 4.6: Solution trajectories of the three different motion planning approaches for the overtaking scenario considered.

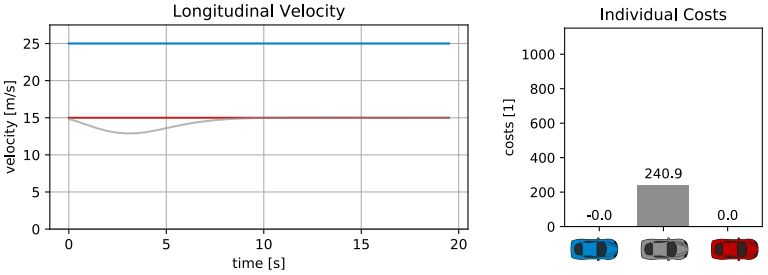
4.1.6 Summary

This section presented a cooperative multi-agent trajectory planner for general driving scenarios. The algorithm generates globally optimal trajectories for a group of automated vehicles. This is achieved by formulating the collision avoidance constraints among vehicles and other objects via a set of binary constraints. In combination with the chosen cost function and vehicle model, this leads to a mixed-integer quadratic program MIQP. The numerical experiments show that this formulation outperforms existing approaches and is able to significantly lower the costs of a group of fully automated vehicles.

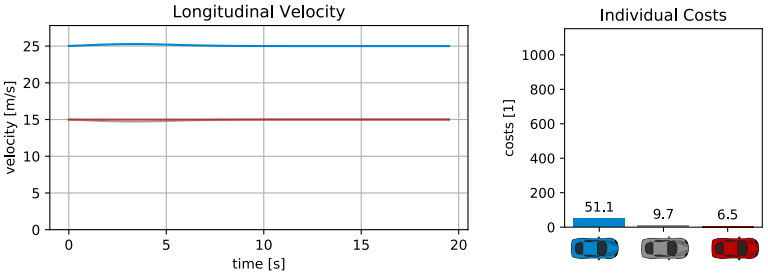
A major advantage of the presented MIQP formulation is that the entire configuration space can be considered directly and no separation of maneuver selection and trajectory optimization is required, see Section 2.3. This joint consideration enables finding cooperative trajectories that might not be found when using an a priori defined sets of maneuvers. This further makes the approach scenario independent since all possible maneuvers, or homotopies in the mathematical term, are implicitly considered.



(a) Individual Planning



(b) Priority-Based Planning



(c) Cooperative MIQP Planning

Figure 4.7: The longitudinal velocities and the individual vehicles' costs produced by the different planning methods are shown. By creating an ad-hoc third lane, the cooperative MIQP planner generates a solution that enables every vehicle to stick to its desired speeds. This allows the cooperative MIQP approach to significantly lower the overall costs and clearly outperforms the other two methods, which results in a more comfortable and more efficient behavior for the ensemble.

The MIQP formulation for the multi-agent motion planning problem thus represents a hybrid approach that combines the advantages of global and local methods. On the one hand, the globally optimal solution across all homotopies is guaranteed, and on the other hand, a smooth, locally continuous solution is obtained.

4.2 Interaction-Aware Motion Planning for Mixed Traffic

In this section, a framework to generate interaction-aware driving behavior based on [BSL20] is presented. Both, the prediction and planning, are solved simultaneously, enabling the automated vehicle to consider the effect its planned actions might have on surrounding vehicles. Further, the framework is able to cope with the uncertainty due to different ways a human driver might react during interactions. Challenging driving maneuvers, like a merge in dense traffic, see Fig. 4.8, can be performed, which would not be possible with pipeline approaches.

At the core of the presented framework is the multi-agent trajectory optimization method introduced in Section 4.1. Rather than to control all traffic participants, the method is used to generate interaction-aware predictions of surrounding vehicles.

Numerical experiments are performed to demonstrate the feasibility of the approach.

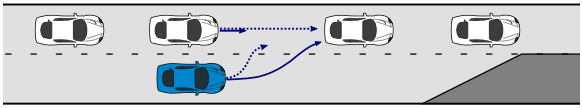


Figure 4.8: A lane change in dense traffic is a challenging driving scenario where considering interactions is crucial. Motion planners following a pipeline architecture often fail to generate satisfying solutions in such scenarios.

4.2.1 The Approach

Overview

The ability to consider the effect own actions might have on the behavior of others is crucial for driving in high traffic scenarios. The presented approach approximates this by generating interaction-aware predictions via solving multi-agent problems. The underlying assumption is that every traffic participant considers the costs of others to some extent in its planning.

In general, human drivers react differently in interactive scenarios. To capture this, we assume that the different ways a human might react are due to different underlying intentions. These intentions are modeled by differently parameterized multi-agent planning problems. The intention modeling is the subject of Section 4.2.1.

The intention of a human can not be measured directly and has to be estimated from observations of the driving behavior. The estimation is done by generating sample trajectories for the individual intention classes, which are then compared to the observations of the real system. By the comparison, conclusions about the underlying intention can be drawn. Here, the sample trajectories should represent a typical driving behavior for the different intention classes. The intention estimation is the subject of Section 4.2.1.

Based on the obtained probability distribution over the intentions, the ego vehicle's behavior is planned. The details are described in section Section 4.2.1. An overview of the algorithm is illustrated in Fig. 4.9.

Intention Modeling

To drive naturally and safely in the vicinity of humans, automated vehicles need to predict the behavior of surrounding traffic participants, including the actions they might take in response to the future behavior of the automated vehicle. Due to the complex and individual nature of human driving, these predictions are subject to uncertainties. E.g., in Fig. 4.8 a driver on the highway might slow down or close the gap as a response to a merge attempt of the automated vehicle.

A common way to deal with these behavior uncertainties is to estimate a probability distribution over a set of a-priory defined maneuvers the human could perform [Ben+15]. The maneuvers are generally defined by hand, and enumerating them is difficult, even if interactions with the automated vehicle are neglected. Further, this approach is highly scenario dependent and not easily transferable to unseen situations.

Instead of describing the predictions in terms of a set of possible maneuvers, we assume that an underlying intention of the human can describe the behavior. These intentions, $\mathcal{I} = I_1, I_2, ..., I_{|\mathcal{I}|}$, are modeled as differently parametrized

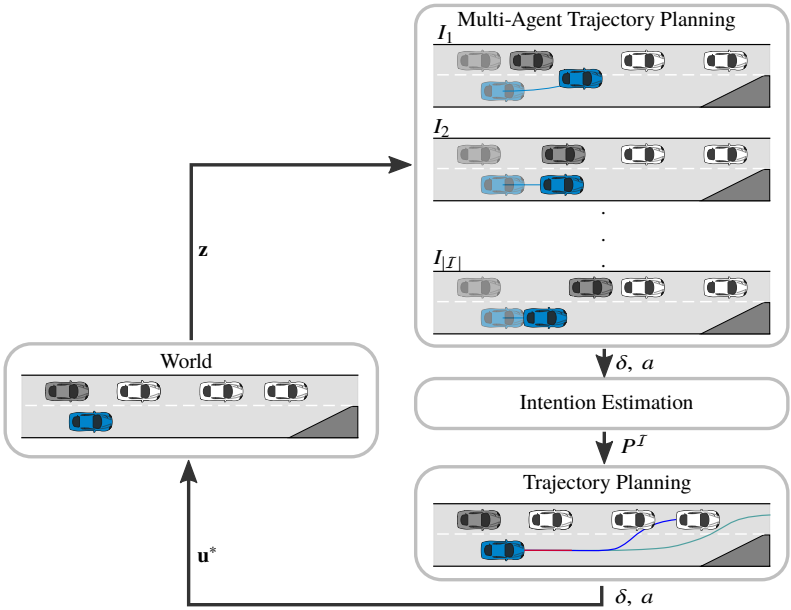


Figure 4.9: Overview of the algorithm to generate interaction-aware trajectories. Different intentions model the different ways a human might react during an interaction. The active intention can not be measured directly and must be estimated. Based on the estimation, the ego trajectory is planned.

multi-agent planning problems, more specifically, by different relative weighings of the vehicles' costs in the joint cost function Eq. (4.17). E.g., a human with a cooperative intention might weigh the costs of others equally to its own; a non-cooperative human, however, is more likely to weigh its own costs higher than the one of others. Fig. 4.10 illustrates the evolution of the scene for different intentions.

Thanks to the global solution character of the MIQP formulation developed in Section 4.1, all maneuvers are implicitly considered and hence do not have to be defined beforehand. This makes the intention formulation based on a parameterized multi-agent planning problem more general and independent of the specific traffic scenario.

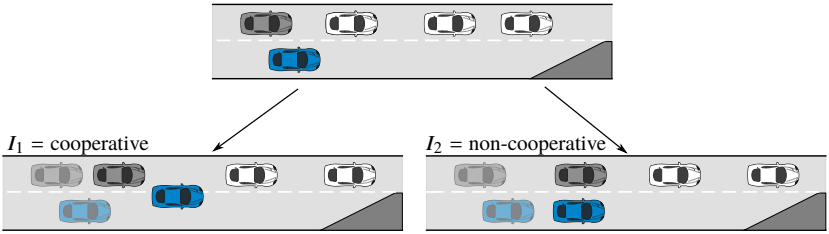


Figure 4.10: Depending on the intention of the human, different evolutions of the traffic scene are possible. On the left is the prediction in case the human has a cooperative intention. On the right, the case with a non-cooperative intention is illustrated.

Intention Estimation

Without knowing which intention model the human is following, the vehicle can not plan its trajectory. Further, the intention of the human can not be observed directly but has to be estimated from observations of the behavior. To utilize Bayesian estimation, sample trajectories are generated, representing common driving behavior for the individual intentions. A probability distribution over the underlying intention is obtained by comparison with the observations.

In the developed approach, any driver is assumed to act according to exactly one of the intention models $\mathcal{I} = I_1, I_2, \ldots, I_{|\mathcal{I}|}$ at any given time. To account for the fact that drivers continuously re-evaluate the current traffic scene, the intention can change between planning intervals. This is modeled by the intention switch probability μ. The human behavior is thus modeled as a stochastic process over $|I|$ models, exactly one of them being active at a time.

The currently active intention is estimated by an *interacting multiple model* (IMM) Kalman filter (KF), where each intention model I_i is described by a separate KF. The prediction of the subsequent state in the filter is determined by applying the control $\mathbf{u}$ generated by solving the intention-specific multi-agent trajectory planning problem. The posterior mode probabilities $P^{\mathcal{I}} = [P^{I_1}, P^{I_2}, \ldots, P^{I_{|\mathcal{I}|}}]^T$ are calculated through Bayesian statistics given the observation $\mathbf{z} = [\mathbf{x}^1, \mathbf{x}^2, \ldots]$, the last posterior and the switching probability μ.

The individual steps of an IMM-KF are illustrated in Fig. 4.11. The task is to estimate the intention of the grey vehicle. For this scenario, two intention models are considered; one results in a lane change in front, the other in a lane

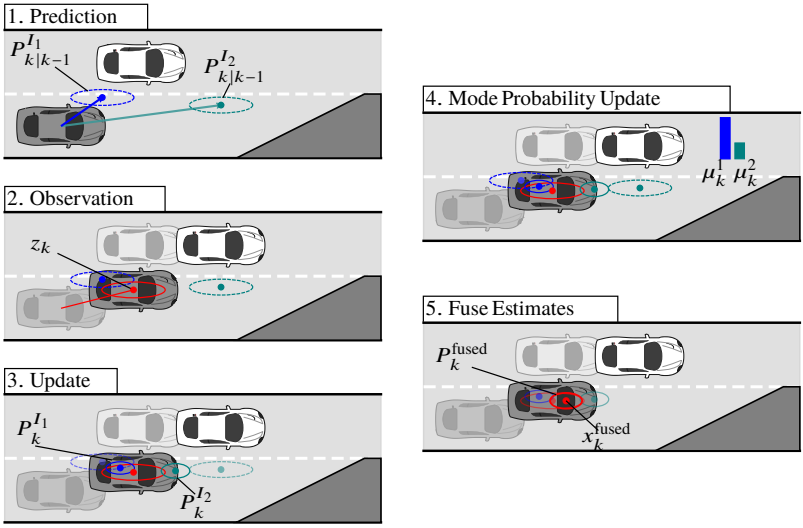


Figure 4.11: Individual steps of an IMM-KF iteration. The goal is to estimate the intention of the grey vehicle during a merge scenario. The two considered intention classes result either in a merging in front of the white vehicle or behind.

change behind the white vehicle. In the first step, the states and covariances of every intention model are predicted. In the observation step, the actual state of the vehicle is measured. Next, each prediction is updated separately according to the observation. In the fours step, the mode probabilities are calculated based on a likelihood function. Based on the probabilities and covariance of the individual filters, a fused state is obtained. The resulting state and covariance are used to initialize the next filter cycle.

Trajectory Planning

Based on the estimated probability distribution over the intentions P_{I_i}, a trajectory for the ego vehicle needs to be determined. A common strategy is only to consider the currently most likely model and optimize the ego behavior accordingly. This driving strategy, however, has its drawbacks. Besides a possibly frequent change in the ego behavior, intentions that are currently

less likely are completely ignored, which could lead to uncomfortable or even dangerous situations.

In contrast, the developed method considers all intention models I_i according to their current probabilities. This is achieved by combining the different multi-agent planning problems in a joint objective. The combination ensures that intentions with lower probabilities also affect the trajectory of the ego vehicle. E.g., if the current probability for I_1 is only slightly lower than the probability of I_2, a driving policy selecting only the most likely model could lead to high overall costs in the future. In contrast, considering all models, the trajectory is adjusted early, and the expected overall costs are decreased.

Instead of optimizing a single trajectory over the entire horizon, the trajectory is allowed to split into several trajectories associated with the different intention models after a shared time t_{shared}, or k_{shared} in discrete time, see Fig. 4.12. This separation mitigates the interdependencies between different intention models and optimizes for different evolutions of the traffic scene.

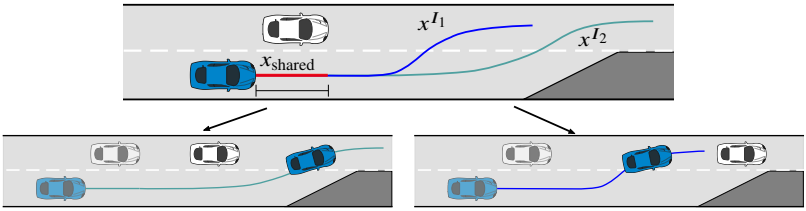


Figure 4.12: The trajectory for the ego vehicle is generated considering all possible intentions of the human driver. The planned trajectory splits after the common states x_{shared} to mitigate the interdependencies between different intention models. In the shown scenario, the green trajectory plans for the case that the human has a cooperative intention, whereas the blue trajectory optimizes for the non-cooperative case.

The underlying cost function is a weighted sum of the intention specific costs J_{I_i} multiplied by their current probability P_{I_i} and is given as:

$$J = \sum_{i=1}^{|\mathcal{I}|} P_{I_i} J_{I_i}(\mathbf{x}_{\text{shared}}, \mathbf{x}^{I_i}_{k_{\text{shared}}+1:N}, \mathbf{u}_{\text{shared}}, \mathbf{u}^{I_i}_{k_{\text{shared}}:N-1}) \tag{4.18}$$

Here, $\mathbf{x}_{\text{shared}}$ are the shared states for $k = 1, \ldots, k_{\text{shared}}$ and $\mathbf{u}_{\text{shared}}$ are the shared inputs for $k = 0, \ldots, k_{\text{shared}} - 1$.

The presented planning formulation represents a novel method to combine several, in general opposing, maneuver hypotheses. Thanks to the MIQP formulation's ability to find the optimal solution across all homotopies, the decision which maneuvers to choose is made implicitly by the optimizer based on the expected costs. Therefore, no hand-designed decision heuristics or other decision logic are needed. A more in detail evaluation is presented the following chapter.

4.2.2 Evaluation

The focus of the evaluation is on the ability to perform driving maneuvers that require the consideration of interactions as well as the ability to cope with the uncertainty due to the different ways a human might react during interactions. To showcase this, a challenging merge in dense traffic is considered.

The framework presented is a general formulation to create interaction-aware driving behavior and allows for different implementations. A suitable one, which builds upon the methods and models of Section 4.1, is presented and used to conduct numerical experiments.

Scenario

To better demonstrate and visualize the method, the merge scenario in Fig. 4.8 is reduced to three vehicles, V^1, V^2, V^3, see Fig. 4.13. The essential interaction

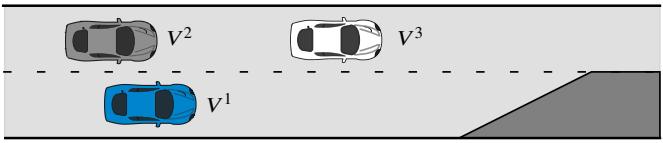


Figure 4.13: Lane change scenario considered for the evaluation. The automated vehicle, shown in blue, has to change lanes due to the ending of the right lane. The gap between V^2 and V^3 is too tight to perform a lane change right away. A joint consideration of prediction and planning is essential to anticipate that a gap might open up through interaction.

happens between the ego vehicle, V^1, and the human driver in V^2 on the target

lane. The interaction with V^3 is neglected since we assume that no interactions happen with traffic participants who are far behind.

Multi-Agent Formulation

The multi-agent planning problem used to generate the intention-specific interaction-aware predictions is presented. It builds upon the formulation presented in Section 4.1.1. The evaluation-specific adaptations and the used parameters are presented next.

State Space The automated vehicle V^1 is modeled as a third order point mass, with state $\mathbf{x}^1 = [p_s^1, v_s^1, a_s^1, p_d^1, v_d^1, a_d^1]^T \in \mathbb{R}^6$ and input $\mathbf{u}^1 = [j_s^1, j_d^1]^T \in \mathbb{R}^2$ in longitudinal, s, and lateral, d, Frenét-coordinates. Since we assume that V^2 will keep its lateral position and not perform a lane change, its dynamics can be simplified. The model can then be reduced to only consider the longitudinal direction. This leads to the state $\mathbf{x}^2 = [p_s^2, v_s^2, a_s^2]^T \in \mathbb{R}^3$, the input $\mathbf{u}^2 = j_s^2 \in \mathbb{R}$, and a constant lateral position p_d^2. Since no interaction with V^3 is considered, it is modeled as a moving obstacle in the multi-agent problem for which an independent prediction is performed, see Eq. (4.13e).

Lane ending The end of the right lane in Fig. 4.13 is enforced by the following constraints:

$$p_{s,k}^1 \leq s_{\text{end}} + (1 - \delta_k) M_s^{\text{big}} \tag{4.19}$$

$$p_{d,k}^1 \geq d_{\text{end}} - \delta_k M_d^{\text{big}} \tag{4.20}$$

The binary variable δ_k, $k = 1, \ldots, N$ couples the two equations and enforces that the automated vehicle is driving on the left lane, by excluding the right lane from the solution space after s_{end}.

Safety distances The original MIQP formulation, presented in Section 4.1, considers an ensemble of fully automated vehicles. This allows for coordinated maneuvers and small safety distances. However, when driving in the vicinity of humans, bigger safety distances are required.

Instead of simply increasing the minimum distance values, $l_{\text{min}}^{n,m}$ and $w_{\text{min}}^{n,m}$ in Eq. (4.12e), soft constraints are introduced, which cause additional costs if the distance between two vehicles decreases below a certain threshold. To utilize this, the collision avoidance constraints Eq. (4.12e) are modified by adding the slack variables $s_{i,k}^{n,m} \geq 0$. The new set of constraints are given by:

$$p_{s,k}^{n} - s_{1,k}^{n,m} \leq p_{s,k}^{m} - l_{\text{min}}^{n,m} - l_{\text{soft}}^{n,m} + (1 - \delta_{1,k}^{n,m})M^{\text{big}} \tag{4.21}$$

$$p_{s,k}^{n} + s_{2,k}^{n,m} \geq p_{s,k}^{m} + l_{\text{min}}^{n,m} + l_{\text{soft}}^{n,m} - (1 - \delta_{2,k}^{n,m})M^{\text{big}} \tag{4.22}$$

$$p_{d,k}^{n} - s_{3,k}^{n,m} \leq p_{d,k}^{m} - d_{\text{min}}^{n,m} - d_{\text{soft}}^{n,m} + (1 - \delta_{3,k}^{n,m})M^{\text{big}} \tag{4.23}$$

$$p_{d,k}^{n} + s_{4,k}^{n,m} \geq p_{d,k}^{m} + d_{\text{min}}^{n,m} + d_{\text{soft}}^{n,m} - (1 - \delta_{4,k}^{n,m})M^{\text{big}} \tag{4.24}$$

$$\tag{4.25}$$

The slack variable takes on values greater than zero if the respective constraint is not fulfilled. The values $l_{\text{soft}}^{n,m}$ and $d_{\text{soft}}^{n,m}$ define below which distance additional costs appear. The conzept is visualized in Fig. 4.14.

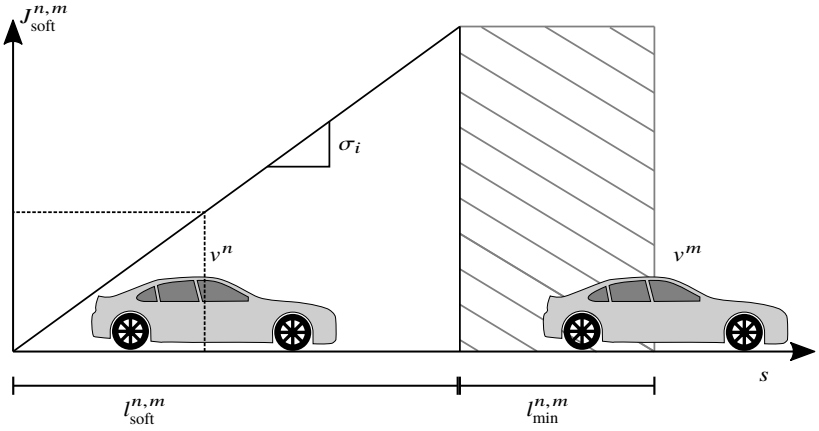


Figure 4.14: Penalty for approaching in s direction.

The penalty term considering the additional costs for each pair of vehicles is given by:

$$J_{\text{soft}}^{n,m} = \sum_{k=1}^{N} \sum_{i=1}^{4} \sigma_i s_{i,k}^{n,m} \tag{4.26}$$

The costs are added to the joint cost function, Eq. (4.17). For each of the four vehicle sides, approaching is punished independently by the penalty factor σ_i. This allows for, e.g., a stronger penalty in the lateral direction than the longitudinal direction.

A big advantage of the introduced soft constraints is that the feasible set remains unchanged, and all feasible solutions are preserved. This would not be the case when simply increasing the minimum distances, and the problem might become infeasible. Further, soft constraints have the additional benefit of increased numerical robustness towards disturbances.

With the presented adaptations, the multi-agent optimal control problem (OCP) can be stated as:

$$\min_{\mathbf{u}_{0:N-1}^{\{1,2\}}, \mathbf{x}_{0:N}^{\{1,2\}}, \mathbf{s}_{1:N}^{1,2}} \quad w^1 \left(\sum_{k=1}^{N} \|\mathbf{x}_k^1 - \mathbf{x}_{k,\mathrm{ref}}^1\|_{Q^1} + \sum_{k=0}^{N-1} \|\mathbf{u}_k^1\|_{R^1} \right) \tag{4.27a}$$

$$+ w^2 \left(\sum_{k=1}^{N} \|\mathbf{x}_k^2 - \mathbf{x}_{k,\mathrm{ref}}^2\|_{Q^2} + \sum_{k=0}^{N-1} \|\mathbf{u}_k^2\|_{R^2} \right) + J_{\mathrm{soft}}^{1,2}(\mathbf{s}_{1:N}^{1,2})$$

s.t. :

$$\mathbf{x}_{k+1}^{\{1,2\}} = \mathbf{f}(\mathbf{x}_k^{\{1,2\}}, \mathbf{u}_k^{\{1,2\}}) \qquad k = 0, \ldots, N-1 \tag{4.27b}$$

$$\mathbf{x}_0^{\{1,2\}} = \hat{\mathbf{x}}^{\{1,2\}} \tag{4.27c}$$

$$\underline{\mathbf{g}_{\mathrm{dyn}}} \leq \mathbf{g}_{\mathrm{dyn}}(\mathbf{x}_k^1) \leq \overline{\mathbf{g}_{\mathrm{dyn}}} \qquad k = 1, \ldots, N \tag{4.27d}$$

$$\mathbf{g}_{\mathrm{col}}(\mathbf{x}_k^1, \mathbf{x}_k^2, \mathbf{s}_k^{1,2}) \leq 0 \qquad k = 1, \ldots, N \tag{4.27e}$$

$$\mathbf{g}_{\mathrm{obs}}(\mathbf{x}_k^{\{1,2\}}) \leq 0 \qquad k = 1, \ldots, N \tag{4.27f}$$

$$\mathbf{g}_{\mathrm{end}}(\mathbf{x}_k^1) \leq 0 \qquad k = 1, \ldots, N \tag{4.27g}$$

$$\underline{\mathbf{x}} \leq \mathbf{x}_k^{\{1,2\}} \leq \overline{\mathbf{x}} \qquad k = 1, \ldots, N \tag{4.27h}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k^{\{1,2\}} \leq \overline{\mathbf{u}} \qquad k = 0, \ldots, N-1 \tag{4.27i}$$

$$0 \leq \mathbf{s}_k^{1,2} \qquad k = 1, \ldots, N \tag{4.27j}$$

Here, $\mathbf{s}_{\mathrm{k}}^{1,2} = [s_{1,\mathrm{k}}^{1,2}, s_{2,\mathrm{k}}^{1,2}, s_{3,\mathrm{k}}^{1,2}, s_{4,\mathrm{k}}^{1,2}]$ are the slack variables introduced in Eq. (4.25), and $J_{\mathrm{soft}}^{1,2}(\mathbf{s}_{\mathrm{k}}^{1,2})$ denotes the penalty term introduced in Eq. (4.26).

Apart from the modified collision avoidance constraints $\mathbf{g}_{\mathrm{col}}$, the constraints enforcing the lane ending $\mathbf{g}_{\mathrm{end}}$, and the non-negativity constraint on the slack variables, the constraints can be carried over from the formulation introduced in the previous section, Section 4.1. Additional parameters for the optimization are listed in Table 4.2.

$T = 20\,\mathrm{s}, \tau = 0.8\,\mathrm{s}$
$\mathrm{diag}(Q^1) = [0, 1, 2, 1, 2, 4]^T$, $\mathrm{diag}(R^1) = [2, 2]^T$
$\mathrm{diag}(Q^2) = [0, 1, 2]^T$, $R^2 = 2$
$\overline{\mathbf{x}} = [free, 10\,\frac{\mathrm{m}}{\mathrm{s}}, 3\,\frac{\mathrm{m}}{\mathrm{s}^2}, 6\,\mathrm{m}, 2\,\frac{\mathrm{m}}{\mathrm{s}}, 2\,\frac{\mathrm{m}}{\mathrm{s}^2}]^T$, $\overline{\mathbf{u}} = [3\,\frac{\mathrm{m}}{\mathrm{s}^3}, 2\,\frac{\mathrm{m}}{\mathrm{s}^3}]^T$
$\underline{\mathbf{x}} = [0, 0, -4\,\frac{\mathrm{m}}{\mathrm{s}^2}, 1\,\mathrm{m}, -2\,\frac{\mathrm{m}}{\mathrm{s}}, -2\,\frac{\mathrm{m}}{\mathrm{s}^2}]^T$, $\underline{\mathbf{u}} = [-6\,\frac{\mathrm{m}}{\mathrm{s}^3}, -2\,\frac{\mathrm{m}}{\mathrm{s}^3}]^T$
$l_{\mathrm{soft}}^{1,2} = 10\,\mathrm{m}$, $d_{\mathrm{soft}}^{1,2} = 0.5\,\mathrm{m}$
$\sigma_1 = \sigma_2 = 20$, $\sigma_3 = \sigma_4 = 100$

Table 4.2: Additional parameters used for the experiment.

Intention Models

Two different intentions $\mathcal{I} = (I_1, I_2)$ are considered for the human driven vehicle. I^1 describes a cooperative intention where all vehicles' costs are weighted equally in the joint cost function Eq. (4.27a). I^2 describes a non-cooperative intention where the human driver weights its costs significantly higher, leading to a more egoistic behavior.

The two intention-specific multi-agent problems are therefore modeled with a weight ratio of $w^2/w^1 = 1$ for I^1 and a ratio of $w^2/w^1 = 100$ for I^2.

IMM-KF Parameters

Following [LLI12], an intention switching probability $\mu = 0.1$ is used for the IMM filter. The initial probability distribution is $P_{I_1} = 0.7$ and $P_{I_2} = 0.3$, setting a slight bias towards assuming that most drivers will behave cooperatively.

The measurements $\mathbf{z} = [s^2, v^2, s^3, v^3]$ consist of the position and the velocity of V_2 and V_3.

Trajectory planning

The motion of the ego vehicle is generated by solving a linear combination of both multi-agent formulations. The cost function can be stated as:

$$J = P_{I^1}\left(J_{V^1} + J_{V^2}\right) + P_{I^2}\left(J_{V^1} + 100 J_{V^2}\right) + J^{1,2}_{\text{soft}} \tag{4.28}$$

where $J_{V^{\{1,2\}}}$ are the individual vehicles' costs, see Eq. (4.27a), and $P_{I^{\{1,2\}}}$ are the current probabilities of the intention models. For the experiments, t_{shared} is set to 4 τ.

Model of human drivers

To make conclusions about the transferability to real traffic, the Intelligent Driver Model (IDM) [THH00] is used to simulate human behavior. The IDM is a car-following model that generates a longitudinal acceleration command which depends on the distance to the vehicle in front $\Delta s = s_{\text{front}} - s - l$ and the velocity difference $\Delta v = v - v_{\text{front}}$ and can be stated as:

$$a = \left(1 - \left(\frac{v}{v_{\text{des}}}\right)^{\delta} - \left(\frac{s^*(v, \Delta v)}{\Delta s}\right)^2\right) \tag{4.29}$$

$$\text{with} \quad s^*(v, \Delta v) = s_0 + vT + \frac{v\Delta v}{2\sqrt{ab}}$$

The used parameters are stated in Table 4.3.

The intention-specific behavior is generated by the choice of the preceding vehicle. If the human driver is non-cooperative, V^3 is used as the front vehicle. In case the human is cooperative, the automated vehicle is considered the front vehicle, which leads to a yielding behavior.

Parameter	Discription	Value
v_{des}	desired velocity	$5\,\frac{\text{m}}{\text{s}}$
s_0	minimum distance	1.5 m
a	maximum acceleration	$1.0\,\frac{\text{m}}{\text{s}^2}$
b	comfortable braking deceleration	$2.0\,\frac{\text{m}}{\text{s}^2}$
T	time headway	2.5 s
δ	velocity exponent	4

Table 4.3: IDM parameters

Experiments

In the following, two experiments are presented. One where the human driver has a non-cooperative intention, and one where it is willing to cooperate. The planning of the automated vehicle is performed in a MPC fashion. As mentioned, V^2 is controlled with the IDM model. V^3 is simulated with constant velocity.

The vehicles are approximated by rectangles with length $l = 5\,\text{m}$ and width $w = 2\,\text{m}$. The initial and reference states used in the experiments are stated in Table 4.4.

	Initial state	Reference State
V^1	$\mathbf{x}_0^1 = [7.5\,\text{m}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}, 1.75\,\text{m}, 0\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T$	$\mathbf{x}_{\text{ref}}^1 = [\text{free}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}, 5.25\,\text{m}, 0\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T$
V^2	$\mathbf{x}_0^2 = [0\,\text{m}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T,\ p_d^2 = 5.25\,\text{m}$	$\mathbf{x}_{\text{ref}}^2 = [\text{free}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T,\ p_d^2 = 5.25\,\text{m}$
V^3	$\mathbf{x}_0^3 = [15\,\text{m}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T,\ p_d^3 = 5.25\,\text{m}$	$\mathbf{x}_{\text{ref}}^3 = [\text{free}, 5\,\frac{\text{m}}{\text{s}}, 0\,\frac{\text{m}}{\text{s}^2}]^T,\ p_d^3 = 5.25\,\text{m}$

Table 4.4: Initial and reference states. As the table shows, all vehicles' goal is to drive in the middle of the upper lane.

Non-cooperative In the first experiment, the case that V^2 has a non-cooperative intention throughout the entire interaction is considered. Here, the human driver will ignore the merge attempt of the automated vehicle. The simulation results are shown in Fig. 4.15. As can be seen, the automated vehicle approaches the gap initially, anticipating that a gap might open up.

Over the evolution of the scene, the active intention is estimated correctly as non-cooperative, making a merge in front not possible. As a result, V^1 slows down and performs the lane change behind V^2.

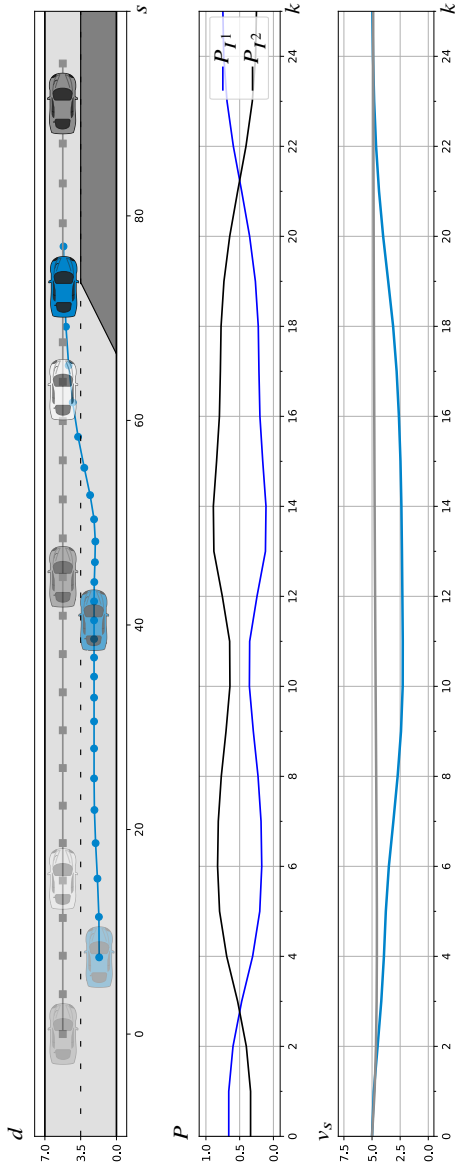


Figure 4.15: Simulation results in case the human driver is non-cooperative. The initial state, the end state, and a state in between are added for better understanding. As seen in the middle plot, the intention is estimated correctly. The ego car slows down and yields.

To give a better inside into the algorithm, the result of a single optimization is shown in Fig. 4.16. After t_{shared}, the trajectory splits representing the different evolutions of the traffic scene depending on the intention I_i of the human driver V^2. In the case of a cooperative intention, the future evolution of the scene is shown on the bottom left side. It is anticipated that the human will slow down to allow for a merge in front. On the bottom right, the predicted evolution for non-cooperative behavior is shown. In this future scenario, the ego vehicle has to slow down to merge behind V^2.

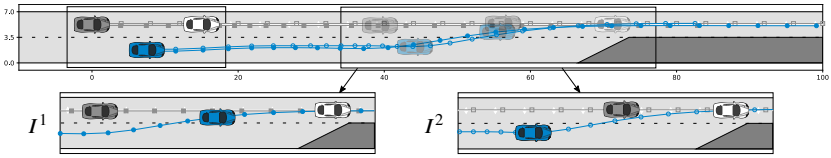


Figure 4.16: Result of a single MPC step during the execution of a lane change. In the beginning, both intention models share the same states. After t_{shared}, the trajectories deviate, reflecting different future evolutions of the scene depending on the intention I^i of V^2.

Cooperative The result of the second experiment is shown in Fig. 4.17. The automated vehicle starts with a bias towards cooperative behavior. After observing the human behavior, the at first non-cooperative intention is estimated correctly, and the ego vehicle slows down. At $k = 7$, V^2 recognizes the merge attempt, and the behavior becomes cooperative. The velocity plot shows that V^2 starts to slow down to enlarge the gap. The intention switch is detected, and the automated vehicle anticipates that V^2 will open the gap, enabling a successful merge in front.

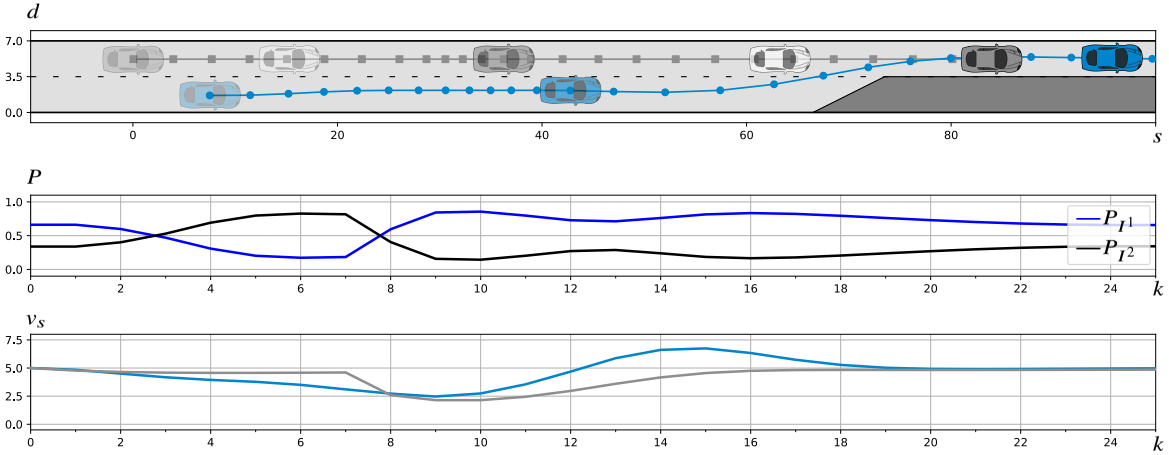


Figure 4.17: The simulation for delayed cooperative behavior is shown. In addition to the trajectories, the velocity profiles and the temporal sequence of the probability distribution over the intentions are shown. At $k = 7$, the human recognizes the merge attempt and switches towards cooperative behavior. The ego car slows down at first due to estimating non-cooperative behavior. At $k = 8$, the intention switch to cooperative behavior is noticed, and the ego vehicle reaccelerates and successfully merges in front.

Both scenarios show that the ego vehicle approaches the gap while continuously updating the belief over the possible intentions. Further, the active intention is estimated correctly in both scenarios, even though surrounding vehicles not exactly follow the predicted sample trajectory associated with their intention.

4.2.3 Summary

In this section, a framework to generate interaction-aware driving behavior for automated vehicles was presented. It builds upon the assumption that a cooperative multi-agent problem can approximate interactions among traffic participants.

Intention models based on differently parametrized multi-agent problems were introduced to capture different ways a human might react. The MIQP formulation presented in Section 4.1 makes the intention modeling scenario independent, and no a-priory definition of maneuver classes is required.

The motion of the automated vehicle is based on the estimated probability distribution over the intention models. Instead of solely optimizing for the currently most likely model, the behavior is optimized for multiple evolutions of the scene. This allows driving maneuver-neutral in case of an unclear intention. The decision, e.g., to merge in front or behind, is implicitly postponed to a later point in time. Further, the final maneuver decision emerges from the optimization, and no hand-designed heuristic is required.

The experiments show that the ability to consider interactions enables the automated vehicle to anticipate the reaction of surrounding traffic participants, which allows for less conservative driving behavior. Combined with the presented intention estimation, challenging driving maneuvers, like a merge in dense traffic, can thus be performed, which would not be possible with pipeline approaches.

5 Interaction-Aware Motion Planning as a Game

The previous chapter presented a model to plan interaction-aware behavior based on a multi-agent system. This model relies on the assumption that all traffic participants are cooperative and act towards a common objective. This implies that humans are as interested in the automated vehicle (AV) reaching its goal as they are in reaching their own. In real traffic, this might be problematic since not all traffic participants follow this principle, e.g., egoistic drivers might only optimize their own objective function. Further, due to the common cost formulation, interactions are considered implicitly rather than as a direct link between the AV's actions and the human's actions. For example, in a merging scenario, a vehicle opening a gap does so to reduce the overall cost rather than as a direct consequence of the merging vehicle approaching the gap.

In this chapter, we propose an algorithm based on a game-theoretic formulation to consider interactions explicitly [Bur+22]. Thereby, the human driver (HD) is assumed to only optimize its own objective function. We assume a turn-taking structure in interactive scenarios, where the AV initiates an interaction and the human reacts to the actions of the AV. This leads to a nested optimization problem, where the HD's optimization is an optimization inside the AV's optimization.

The presented formulation yields a direct link between the AV's actions and the actions the human will take as a response. This enables the AV to be aware of how its actions effect the human and furthermore, provides the AV with the possibility to deliberately influence the state of the human. In addition, this link can be used to leverage interactions to plan more efficient trajectories, e.g., anticipating that a human will open a gap when merging in dense traffic, which in turn allows for a smoother merge.

The chapter begins with stating the game-theoretic problem formulation for interaction in the considered AV-HD system. In Section 5.2, this game-

theoretic formulation is approximated by a bi-level optimization problem. Section 5.3 presents the single-level reformulation of the bi-level problem, which is necessary to efficiently utilize derivative-based optimization methods. In Section 5.4, the capabilities and the performance of the proposed algorithm are evaluated in different driving scenarios. A proof-of-concept model predictive control (MPC) implementation is proposed that delivers promising results and real-time performance. It is demonstrated that using the presented algorithm, an AV is able to plan interaction-aware trajectories to deliberately influence the state of the human and further, is able to plan more efficient trajectories by considering interactions. In addition, the runtime and the robustness of the algorithm are evaluated. Finally, Section 5.5 discusses the major assumptions and limitations of the proposed algorithm.

5.1 Problem Statement

This section develops a model based on a game-theoretic formulation that directly captures interactions between a AV and a HD. Therefore, we model the interaction between two agents as an asymmetric decision-making process, where one agent, considered the *leader*, decides on its action first and a second agent, the *follower*, then optimizes its response to the leader's action. This hierarchical structure is also referred to as a Stackelberg game. In the following, we will use a notation similar to the one used in [Sad+16b].

We consider a system with one AV, representing the leader L, and one HD, representing the follower F. The system's state at time t is given by the leader's and follower's state $\mathbf{x}_t^L, \mathbf{x}_t^F \in \mathcal{X}$, where $\mathcal{X}$ is the set that contains all possible states. The leader's and follower's actions are described by their trajectories $\xi_L(t), \xi_F(t) : [0, \mathcal{T}] \to \mathcal{X}$. Further, each agent has its individual objective function denoted by J_L and J_F.

The objective is minimized subject to the vehicle's initial state $\xi(0) = \mathbf{x}_0$ and the evolution of the state described by the trajectory, which is only allowed to pass through the set of feasible states $\mathcal{X}_{\text{feasible}}(t) \subseteq \mathcal{X}$. $\mathcal{X}_{\text{feasible}}(t)$ encodes, for instance, collision avoidance. Additionally, system dynamics and bound constraints can be enforced by $D(\xi(t), \dot{\xi}(t), \ddot{\xi}(t), \dots) = 0$. The set of all feasible trajectories $\xi(t)$ is denoted by Ξ.

Due to the turn-taking assumption, the follower optimizes its own trajectory as a response to the leader's trajectory. To do so, the follower predicts the leaders's future motion $\tilde{\xi}_L$ and then plans by minimizing its objective function J_F considering these predictions. Therefore, the follower's optimal trajectory can be described as:

$$\underset{\xi_F \in \Xi_F}{\arg\min}\, J_F(\mathbf{x}_0^L, \mathbf{x}_0^F, \tilde{\xi}_L, \xi_F) \tag{5.1}$$

For simplicity, we assume that for short time horizons, a human can predict the trajectory of the AV sufficiently well, such that the prediction $\tilde{\xi}_L$ can be assumed to be the actual trajectory ξ_L of the AV. Hence, the optimal trajectory of the follower as a function of the leader's actual trajectory ξ_L is given as:

$$\xi_F^*(\mathbf{x}_0^L, \mathbf{x}_0^F, \xi_L) = \underset{\xi_F \in \Xi_F}{\arg\min}\, J_F(\mathbf{x}_0^L, \mathbf{x}_0^F, \xi_L, \xi_F) \tag{5.2}$$

Equation (5.2) gives the leader the ability to reason about how its actions will influence the follower's response and therefore, provides a way to indirectly control the follower's future trajectory.

With this link between the leader's actions and the follower's actions the optimal trajectory for the AV can be stated as:

$$\xi_L^* = \underset{\xi_L \in \Xi_L}{\arg\min}\, J_L \left(\mathbf{x}_0^L, \mathbf{x}_0^F, \xi_L, \xi_F^*(\mathbf{x}_0^L, \mathbf{x}_0^F, \xi_L)\right) \tag{5.3}$$

Equation (5.3) is the fundamental model which enables the planning of interaction-aware behavior.

5.2 Bi-level Formulation

If the follower's best response to the leader's actions can be stated in closed form, Eq. (5.3) can be solved as a standard optimal control problem (OCP). However, this is, in general, not the case since ξ_F^* is the outcome of an OCP itself, which results in a nested optimization, also referred to as a bi-level optimization problem, see Fig. 5.1. Further, solving the underlying Stackelberg game would require planning until $\mathcal{T}$, which is the end of an interaction. However, the end

of an interaction is not trivial to determine and requires the consideration of a varying time horizon.

In the following, we propose an approximate solution to Eq. (5.3) based on MPC, where we solve the problem on a receding horizon with a fixed length T, execute the first action and then replan. We utilize multiple shooting methods and discretize the time horizon $t \in [0, T]$ into $N = T/\tau$ intervals, where τ denotes the duration of the time steps. To improve readability, we subsume the state and input sequences of the leader and follower as $\mathbf{x} := (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and $\mathbf{u} := (\mathbf{u}_0, \ldots, \mathbf{u}_{N-1})$. In the following, the resulting nonlinear programs (NLPs) of the follower and leader are stated. The equality constraints $\mathbf{h}$ can be used to represent constraints imposed by the system dynamics while the inequality constraints $\mathbf{g}$ collect bound constraints, collision constraints, and dynamic constraints.

5.2.1 NLP of the Follower

The follower's NLP is parametrized by the leader's states and inputs $(\mathbf{x}^L, \mathbf{u}^L)$ and can be formulated as:

$$\begin{aligned} \underset{\mathbf{x}^F, \mathbf{u}^F}{\arg\min} \quad & J_F(\mathbf{x}^L, \mathbf{x}^F, \mathbf{u}^F) && \text{(5.4a)} \\ \text{s.t.} \quad & \mathbf{h}_F(\mathbf{x}^F, \mathbf{u}^F) = 0, && \text{(5.4b)} \\ & \mathbf{g}_F(\mathbf{x}^L, \mathbf{x}^F, \mathbf{u}^F) \leq 0 && \text{(5.4c)} \end{aligned}$$

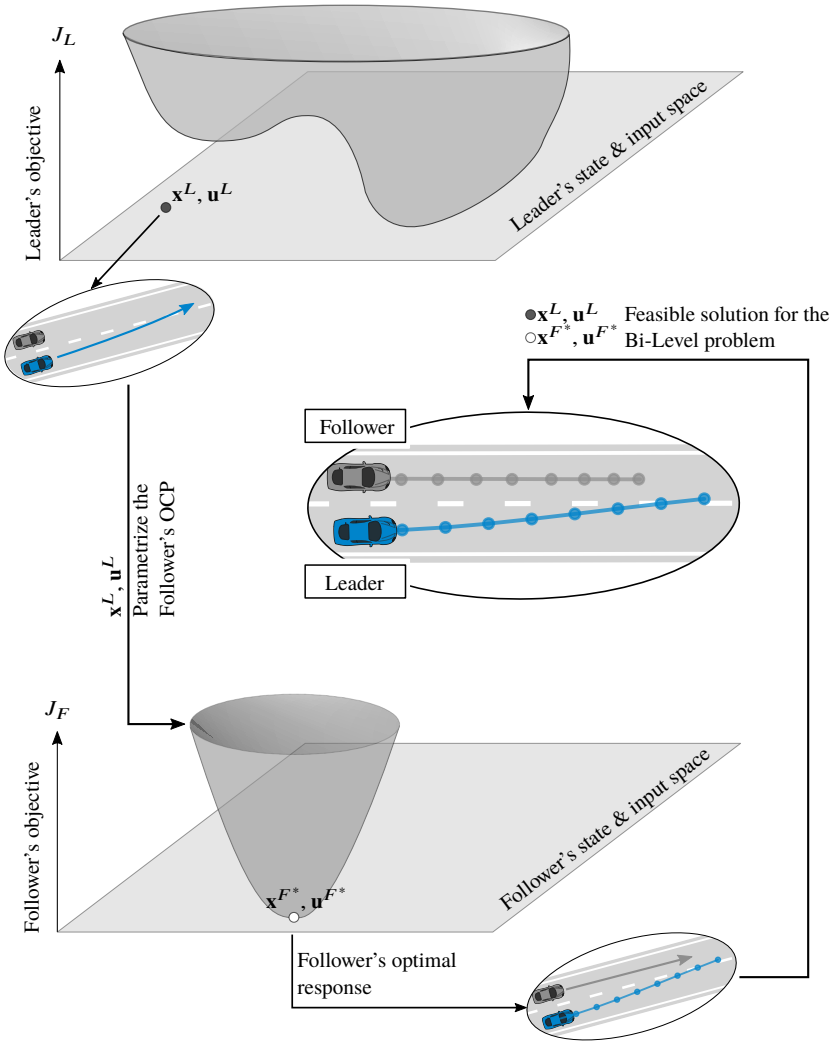


Figure 5.1: Structure of a bi-level optimization problem. Here, the follower optimizes its objective function as a response to the given actions of the leader.

5.2.2 NLP of the Leader

The leader's bi-level optimization problem can be stated as:

$$\underset{\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L,\mathbf{u}^F}{\arg\min} \quad J_L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L) \tag{5.5a}$$

$$\text{s.t.} \quad \mathbf{h}_L(\mathbf{x}^L,\mathbf{u}^L) = 0, \tag{5.5b}$$

$$\mathbf{g}_L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L) \leq 0, \tag{5.5c}$$

$$(\mathbf{x}^F,\mathbf{u}^F) \in \underset{\mathbf{x}^F,\mathbf{u}^F}{\arg\min}\{J_F(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F) : \mathbf{h}_F = 0,\ \mathbf{g}_F \leq 0\} \tag{5.5d}$$

Formulating the follower's optimization problem as a constraint, Eq. (5.5d), ensures that only optimal solutions for the follower are considered feasible solutions when optimizing the leader's trajectory.

5.3 Single-Level Representation

To efficiently solve Eq. (5.5a), we first reformulate the bi-level optimization problem into a regular, single-level problem. Therefore, we utilize the assumption that the follower will act optimally with respect to its own cost function Eq. (5.4). This allows to replace the inner optimization problem with its necessary conditions for optimality.

If the follower's problem is convex, the Karush Kuhn Tucker (KKT) conditions are necessary and sufficient for optimality. However, due to the combinatorial nature of driving it is, in general, non-convex, e.g., due to non-linear collision avoidance constraints or a non-convex cost function. To obtain a locally optimal solution, we convexify the follower's problem around an initial guess, which at the same time encodes the considered homotopy class. For the convexification, the constraints are linearized, and the cost function is approximated by a second order Tailer expansion.

Henceforth, the bi-level optimization problem Eq. (5.5) is reformulated as a single-level problem by replacing the follower's optimization problem Eq. (5.4) with its KKT conditions, resulting in:

$$\begin{aligned}
\underset{\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L,\mathbf{u}^F,\boldsymbol{\lambda},\boldsymbol{\mu}}{\arg\min} \quad & J_L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L) && \text{(5.6a)}\\
\text{s.t.} \quad & \mathbf{h}_L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L)=0, && \text{(5.6b)}\\
& \mathbf{g}_L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^L)\leq 0, && \text{(5.6c)}\\
& \nabla_{(\mathbf{x}^F,\mathbf{u}^F)} L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F,\boldsymbol{\lambda},\boldsymbol{\mu})=0, && \text{(5.6d)}\\
& \mathbf{h}_{F_{\text{lin}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F)=0, && \text{(5.6e)}\\
& \mathbf{g}_{F_{\text{lin}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F)\leq 0, && \text{(5.6f)}\\
& \boldsymbol{\mu}\geq 0, && \text{(5.6g)}\\
& \boldsymbol{\mu}\perp\mathbf{g}_{F_{\text{lin}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F) && \text{(5.6h)}
\end{aligned}$$

with the Lagrangian

$$\begin{aligned}
L(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F,\boldsymbol{\lambda},\boldsymbol{\mu}) = {} & J_{F_{\text{con}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F)\\
& +\boldsymbol{\lambda}^T\mathbf{h}_{F_{\text{lin}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F)+\boldsymbol{\mu}^T\mathbf{g}_{F_{\text{lin}}}(\mathbf{x}^L,\mathbf{x}^F,\mathbf{u}^F)
\end{aligned}$$

Here, $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the KKT multipliers and $\mathbf{h}_{F_{\text{lin}}}$, $\mathbf{g}_{F_{\text{lin}}}$ and $J_{F_{\text{con}}}$ are the constraints and objective after the convexification. For the reformulation, we assume sufficient regularity of the follower's NLP, differentiability of $\mathbf{h}_F$ and $\mathbf{g}_F$, and the cost function J_F to be twice differentiable.

5.3.1 Solving the Complementarity Constraints

The leader's NLP in Eq. (5.6) forms an instance of a mathematical program with complementarity constraints. Due to the complementarity constraints $\boldsymbol{\mu}\perp\mathbf{g}_F$, MPCCs are non-smooth and non-convex problems. They are particularly challenging to solve because at every feasible point, ordinary constraint qualifiers (CQ) such as LICQ or Mangasarian-Fromovitz CQ are violated [CKA95]. To

solve the MPCC, we reformulate it using relaxation methods [HKS13]. Here, the complementarity constraints Eq. (5.6h) are relaxed as follows:

$$-\epsilon \leq \boldsymbol{\mu}^T \mathbf{g}_F. \tag{5.7}$$

With $\epsilon > 0$ a regularized NLP is obtained, and CQ can be satisfied again. Further, the smaller ϵ is chosen, the closer any feasible solution is to achieving complementarity. However, if ϵ is chosen too small, the problem may be numerically unstable and the solver will fail to find a feasible solution.

5.4 Evaluation

In the following, the proposed algorithm is evaluated in multiple interactive scenarios. The algorithm's efficacy is evaluation in two settings, each with a different focus. As claimed at the beginning of this chapter, the proposed algorithm provides the AV's with a direct link between its actions and the HD's actions which allows the AV to be aware of how its actions affect the HD. The first part of the evaluation, Section 5.4.3, will investigate this link by demonstrating the AV's ability to deliberately influence the HD's state through its driving behavior. Since in real driving applications, the goal of the AV is to drive efficiently and comfortably rather than to influence the state of other vehicles, the focus of the second part, Section 5.4.4, is to demonstrate how the approach can be used to plan interaction-aware, cooperative driving behavior.

Apart from the efficacy, the algorithm's robustness is empirically evaluated using Monte Carlo simulations. This is followed by a runtime analysis and a discussion highlighting the advantages and limitations of the algorithm.

The section starts by stating the OCP used for trajectory optimization. This OCP contains the system dynamics, bound constraints, as well as an objective function to encode desirable driving behavior. For the purpose of this evaluation, we assume that a good approximation of a human cost function is provided, which could be obtained, e.g., via inverse reinforcement learning.

5.4.1 Trajectory Optimization for AVs

The OCP used for trajectory planning can be stated as:

$$\arg\min_{\mathbf{x},\mathbf{u}} J_{\text{base}} = \arg\min_{\mathbf{x},\mathbf{u}} J_{\mathbf{x}} + J_{\mathbf{u}} + J_{\dot{\mathbf{u}}} \tag{5.8a}$$

s.t. :

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \qquad k = 0, \ldots, N-1 \tag{5.8b}$$

$$\mathbf{x}_0 = \hat{\mathbf{x}} \tag{5.8c}$$

$$\underline{\mathbf{g}_{\text{dyn}}} \leq \mathbf{g}_{\text{dyn}}(\mathbf{x}_k) \leq \overline{\mathbf{g}_{\text{dyn}}} \qquad k = 1, \ldots, N \tag{5.8d}$$

$$\mathbf{g}_{\text{col}}(\mathbf{x}_k^F, \mathbf{x}_k^L) \leq 0 \qquad k = 1, \ldots, N \tag{5.8e}$$

$$\mathbf{g}_{\text{obs}}(\mathbf{x}_k) \leq 0 \qquad k = 1, \ldots, N \tag{5.8f}$$

$$\underline{\mathbf{x}} \leq \mathbf{x}_k \leq \overline{\mathbf{x}} \qquad k = 1, \ldots, N \tag{5.8g}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \overline{\mathbf{u}} \qquad k = 0, \ldots, N-1 \tag{5.8h}$$

The objective function J_{base} is used to generate a desirable driving behavior and consists of the three components $J_{\mathbf{x}}$, $J_{\mathbf{u}}$, $J_{\dot{\mathbf{u}}}$, penalizing deviations from a desired state, any control effort, and any changes in control, respectively.

In contrast to Chapter 4, a more accurate vehicle model is used and the restriction to a Frenet frame is released. Additional, $\mathbf{g}_{\text{dyn}}$ limit the lateral acceleration to ensure that the vehicle model stays valid. The inequality constraints Eq. (5.8e-5.8h) are used for collision avoidance and to account for physical limitations of the real system. The individual components of the OCP are described in the following.

Vehicle Model

The kinematic single-track model was chosen to describe the dynamics of the vehicles. The vehicle state at time k, $\mathbf{x}_k = (x_k, y_k, \psi_k, v_k)$, is described by the lateral and longitudinal position (x, y) of the vehicle's center of gravity, the orientation ψ, and the absolute velocity v. Together with the input $\mathbf{u}_k = (\delta_k, a_k)$

consisting of steering angle δ and acceleration a, the dynamics of a vehicle are given by:

$$\dot{\mathbf{x}}_k = \begin{pmatrix} \dot{x}_k \\ \dot{y}_k \\ \dot{\psi}_k \\ \dot{v}_k \end{pmatrix} = \begin{pmatrix} v_k \cos(\psi_k + \beta_k) \\ v_k \sin(\psi_k + \beta_k) \\ \frac{v_k}{l} \tan(\delta_k) \cos(\beta_k) \\ a_k \end{pmatrix} \tag{5.9}$$

Here, β is the slip angle which is given by $\beta = \arctan\left(\frac{l_r}{l} \tan(\delta)\right)$.

Further, l is the wheelbase, and l_r is the distance between the center of gravity and the rear axis. A discrete dynamics model $\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k)$ is obtained using a fourth-order Runge-Kutta method.

To ensure the validity of the kinematic single-track model [Pol+17], $\mathbf{g}_{\text{dyn}}(\mathbf{x}_k)$ Eq. (5.8d) are introduced to limit the lateral acceleration as follows:

$$|v_k \dot{\psi}_k| = |\frac{v_k^2}{l} \tan(\delta_k) \cos(\beta_k)| \leq a_{\text{lat,max}} = 4 \, \tfrac{\text{m}}{\text{s}^2} \tag{5.10}$$

Additionally, realistic dynamics are enforced by bound constraints on the states Eq. (5.8g) and inputs Eq. (5.8h). Further, to limit the jerk, the following constraints on the acceleration change are introduced:

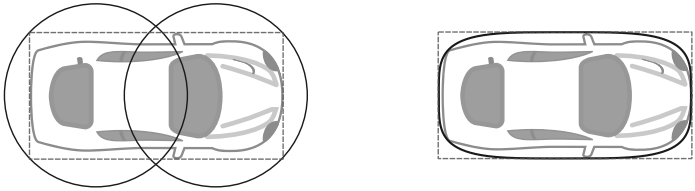
$$j_{\min} \leq \frac{a_k - a_{k-1}}{\tau} \leq j_{\max} \tag{5.11}$$

Here, $j_{\min}$ and $j_{\max}$ are the minimum and maximum allowed jerk values.

Collision Avoidance

The collision avoidance constraints between vehicles are formulated by approximating the shape of one vehicle by a finite number of circular primitives and the second vehicle with a single ellipsoidal primitive, see Fig. 5.2. Representing one vehicle by an ellipsoidal primitive significantly reduces the total number of constraints compared to a formulation that solely relies on circular primitives while still being easy to compute.

Instead of regular ellipses, superellipses are used in this work. They provide a more accurate approximation of the vehicle's rectangular shape [Ne+17].



(a) (b)

Figure 5.2: Shape approximations by a) multiple circles and b) a superellipse of order $n = 4$.

Collision avoidance between a point $\mathbf{p} = [x, y]^T$ and a superellipse defined by the semi-major a, the semi-minor b, and order $n \in \mathbb{N}$ can be formulated as:

$$\sqrt[n]{\left(\frac{x}{a}\right)^n + \left(\frac{y}{b}\right)^n} \geq 1 \tag{5.12}$$

Note that the position of the point $\mathbf{p}$ is given in the coordinate system of the superellipse.

Collision avoidance between a circle with radius r and a superellipse can similarly be formulated as a point mass constraint on the center point of the circle $\mathbf{p_c} = [x_c, y_c]^T$. This is achieved by modifying Eq. (5.12) to ensure that $\mathbf{p_c}$ is outside the Minkowski sum of the superellipse and a circle with radius $\frac{r}{2}$, as illustrated in Fig. 5.3.

The Minkowsi sum is approximated by an enlarged superellipse to maintain an efficient formulation. In the case of a superellipse of order $n = 4$, enlarging the semi-major and semi-minor by the radius r is sufficient to overapproximate the Minkowski sum, as can be seen in Fig. 5.3. Hence, the collision avoidance constraint between a circle and a superellipse can be stated as:

$$\sqrt[n]{\left(\frac{x_c}{a+r}\right)^n + \left(\frac{y_c}{b+r}\right)^n} \geq 1 \tag{5.13}$$

The center point of the circle $\mathbf{p_c}$ is, again, given in the coordinate frame of the superellipse. Note that depending on the order, the superellipse has to be enlarged further to fully contain the Minkowski sum [Bri+19].

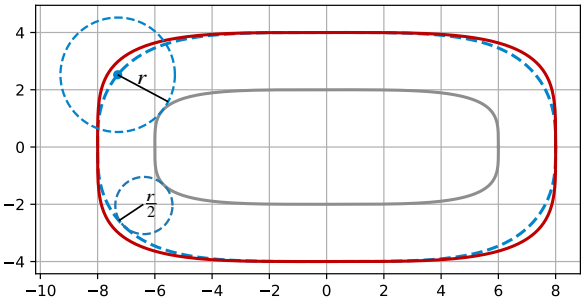


Figure 5.3: Comparison of the Minkowski sum, shown in blue, and a superellipse with the semi-major and semi-minor enlarged by r, shown in red. The original superellipse is shown in grey.

Objective Function

We use the cost function in Eq. (5.14) to penalize any control effort, changes in control, as well as deviations of the current state to a desired reference state or trajectory $\mathbf{x}_{\text{ref}} = [x_{k,\text{ref}}, y_{k,\text{ref}}, \psi_{k,\text{ref}}, v_{k,\text{ref}}]^T$.

$$J_{\text{base}}(\mathbf{x}, \mathbf{u}) = J_{\mathbf{x}} + J_{\mathbf{u}} + J_{\dot{\mathbf{u}}} \tag{5.14a}$$

$$= \sum_{k=1}^{N} \begin{pmatrix} x_k - x_{k,\text{ref}} \\ y_k - y_{k,\text{ref}} \\ \psi_k - \psi_{k,\text{ref}} \\ \Delta v \end{pmatrix}^T Q \begin{pmatrix} x_k - x_{k,\text{ref}} \\ y_k - y_{k,\text{ref}} \\ \psi_k - \psi_{k,\text{ref}} \\ \Delta v \end{pmatrix} \tag{5.14b}$$

$$+ \sum_{k=0}^{N-1} {\mathbf{u}_k}^T R_u \mathbf{u}_k \tag{5.14c}$$

$$+ \sum_{k=1}^{N-1} (\mathbf{u}_k - \mathbf{u}_{k-1})^T R_{\dot{u}} (\mathbf{u}_k - \mathbf{u}_{k-1}) \tag{5.14d}$$

$$+ (\mathbf{u}_0 - \hat{\mathbf{u}})^T R_{\dot{u}} (\mathbf{u}_0 - \hat{\mathbf{u}}) \tag{5.14e}$$

With the velocity vector $\mathbf{v} = [v\cos(\psi+\beta), v\sin(\psi+\beta)]^T$ and the road tangential unit vector $\mathbf{t}$, $\Delta v = \mathbf{v}\cdot\mathbf{t} - v_{\text{ref}}$ measures the difference between the current velocity along the road and the reference velocity v_{ref}. Further, $\hat{\mathbf{u}}$ is the control input from the previous step. Finally, R_u,$R_{\dot{u}}$ and Q are weighting matrices used to model the desired driving behavior.

5.4.2 Base Scenario

We evaluate our approach in multi-lane scenarios as shown in Fig. 5.4, where the AV is depicted in blue and the HD is depicted in gray. For the purpose of these experiments, the AV is considered the leader, and the HD is considered the follower. In the following, we will use the terms leader and AV as well as follower and HD interchangeably.

Both vehicles have a width of 2.0 m and a length of 4.0 m. Collision avoidance is implemented using a superellipse of order $n = 4$ for the leader and two circles for the follower. Further parameters are given in Table 5.1.

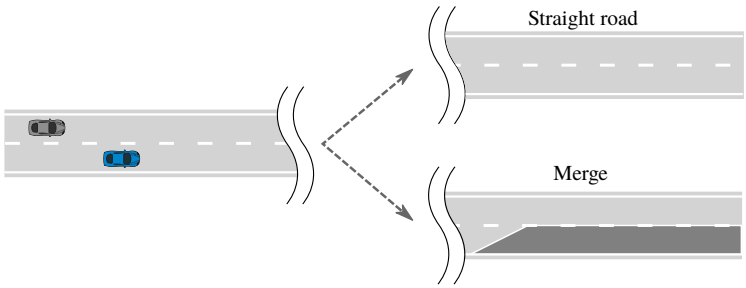


Figure 5.4: Depending on the experiment, either a multi-lane or a merging scenario, where the right lane ends, is considered.

The follower directly uses the cost function Eq. (5.14) for its trajectory planning with the weights and vehicle characteristics given in Table 5.1. The leader's OCP is also based on the objective function J_{base} but additionally considers the KKT conditions of the follower's OCP as constraints, as stated in Eq. (5.6).

Parameter	Value
N	30
T	6 s
Q	diag(0, 1, 0, 100)
R_u	diag(1, 1)
$R_{\dot{u}}$	diag(10000, 1000)
$v_{\min}$, $v_{\max}$	$0\ \frac{\text{m}}{\text{s}}$, $30\ \frac{\text{m}}{\text{s}}$
$\delta_{\max}$	30°
$a_{\min}$, $a_{\max}$	$-8\ \frac{\text{m}}{\text{s}^2}$, $3\ \frac{\text{m}}{\text{s}^2}$
$j_{\min}$, $j_{\max}$	$-10\ \frac{\text{m}}{\text{s}^3}$, $6\ \frac{\text{m}}{\text{s}^3}$
l	4 m
l_r	2 m

Table 5.1: MPC parameters

Further, the leader's objective function is augmented with additional cost terms to set scenario-specific incentives.

If not stated otherwise, the initial and reference states listed in Table 5.2 are used for the leader and follower.

Parameter	Value
$\mathbf{x}_0^L$	$[12.0\,\text{m}, 3.0\,\text{m}, 0°, 10.0\ \frac{\text{m}}{\text{s}}]^T$
$\mathbf{x}_0^F$	$[2.0\,\text{m}, 5.0\,\text{m}, 0°, 10.0\ \frac{\text{m}}{\text{s}}]^T$
$\mathbf{x}_{\text{ref}}^L = \mathbf{x}_{\text{ref}}^F$	$[0.0\,\text{m}, 5.0\,\text{m}, 0°, 10.0\ \frac{\text{m}}{\text{s}}]^T$

Table 5.2: Leader's and follower's initial and reference states.

5.4.3 Influence the Human's State

The following two experiments investigate the leader's ability to influence the follower's state. To provide the appropriate incentives, the leader's objective function is augmented with $J_{\text{influence}}$. The leader's objective is, therefore, the following weighted sum:

$$J_L = w_L J_{\text{base}} + w_{\text{influence}} J_{\text{influence}} \tag{5.15}$$

Henceforth, a ratio of $\frac{w_{\text{influence}}}{w_L} = 10^7$ is used.

Slow down the Human

In this experiment, the leader's goal is to slow down the follower to a certain velocity. To incentivize this behavior, deviations of the follower's velocity along the road to a certain reference velocity v_{ref}^F are penalized in the leader's objective function. Therefore, the scenario-specific $J_{\text{influence}}$ is set to

$$J_{\text{influence}} = \sum_{k=1}^{N} (\mathbf{v}_k^F \cdot \mathbf{t}_k - v_{\text{ref}}^F)^2 \tag{5.16}$$

with $\mathbf{v} = [v\cos(\psi + \beta), v\sin(\psi + \beta)]^T$ and $\mathbf{t}$ as the road tangential unit vector.

The resutls for a desired velocity of $v_{\text{ref}}^F = 5.0\,\frac{\text{m}}{\text{s}}$ are illustrated in Fig. 5.5. As can be seen, the leader changes to the left lane to get in front of the follower. Despite its interest in driving fast, the leader then starts to brake, forcing the follower to slow down. To prevent the follower from overtaking, the leader drives close to the center of the road. The corresponding velocity profiles are presented in Fig. 5.6.

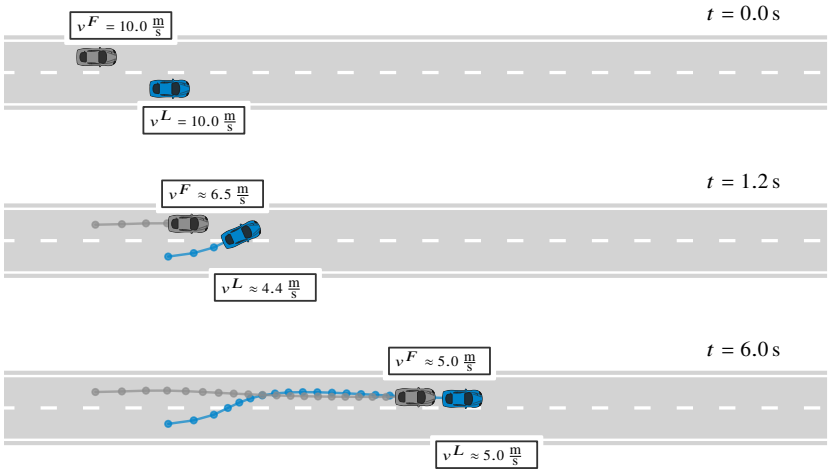


Figure 5.5: The leader drives in front and starts braking to slow down the follower.

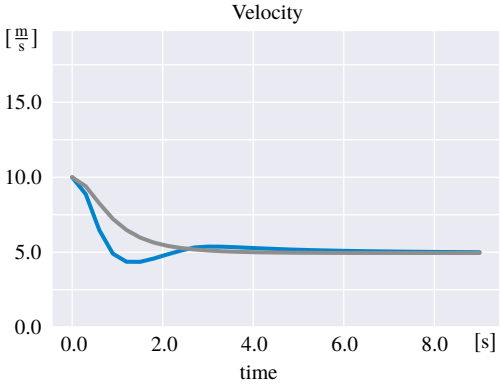


Figure 5.6: As can be seen, the leader is able to slow down the follower to the desired velocity of $v_{\text{ref}}^F = 5.0\,\frac{\text{m}}{\text{s}}$.

Push the Human to the Adjacent Lane

In this experiment, the ability to also influence the follower in the lateral direction is investigated. Therefore, a 3-lane road is considered, see Fig. 5.7.

The leader's goal is to enforce a lane change of the human to the adjacent left lane. This incentive is encoded by setting $J_{\text{influence}}$ to penalize deviations of the follower's lateral position to a reference y_{ref}^F as:

$$J_{\text{influence}} = \sum_{k=1}^{N} (y_k^F - y_{\text{ref}}^F)^2 \tag{5.17}$$

Fig. 5.7 shows the behavior for $y_{\text{ref}}^F = 8.5\,\text{m}$, which corresponds the center of the adjacent lane. To push the follower to the left, the leader changes lanes and slows down, almost comming to a full stop, see Fig. 5.8. The leader thereby blocks the middle lane, which forces the follower to also slow down to avoid a collision. To continue, the follower starts an overtaking maneuver. At the same time, the leader accelerates again to stay next to the follower, blocking him from changing back to his original lane.

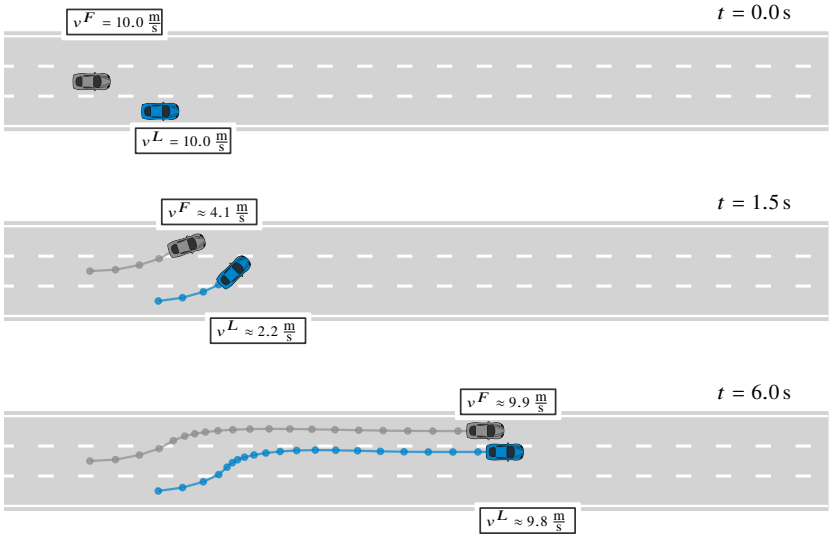


Figure 5.7: The leader pushes the follower to the leftmost lane by blocking the middle lane.

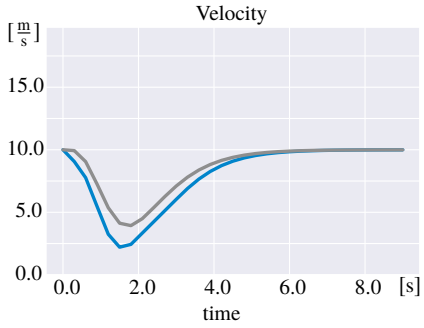


Figure 5.8: The leader changes lanes and brakes harshly to enforce an overtaking maneuver of the follower. As soon as the follower tries to overtake, the leader accelerates again, blocking the follower from changing back to the middle lane.

5.4.4 Interaction-Aware Trajectory Planning

In real traffic, the primary goal of the AV is to drive comfortably and efficiently rather than to change the state of surrounding vehicles in a certain way. Therefore, the generated behaivor when planning trajectories with the proposed interaction-aware algorithm in different lane cahnge sceaniros is investigated next. To better show the effect of the planned behavior, the desired velocity of the follower is increased to $v_{\text{ref}}^{F} = 15.0\,\frac{\text{m}}{\text{s}}$. Throughout the scenarios, the leader aims to perform a lane change to the left.

Efficient Planning

We start by formulating the leader's objective in an egocentric way, similar to how it is formulated for pipeline planners. Here, the leader solely considers attributes of its own trajectory, formulated by only optimizing J_{base}.

The resulting trajectories are shown in Fig. 5.9. As can be seen, the leader plans a very efficient lane change whithout any acceleration. However, as a response, the follower has to brake harshly to avoid a collision, see Fig. 5.10. This aggresive cut in is a result of the leader knowing that the follower will react, which the leader then exploits to optimize its own driving behavior.

This example shows that interactive behavior not only occurs when the leader is incentivized to alter the state of the follower but also emerges out of efficiency.

Cooperative Interaction-Aware Planning

The proposed interaction-aware model gives the leader the ability to anticipate the follower's reaction. When naively using an egocentric objective function, the leader exploits the follower's response and generates an overly aggressive behavior, as demonstrated in the previous example.

To mitigate this effect, the impact imposed on others must be considered in the objective function of the leader. Therefore, a formulation based on a

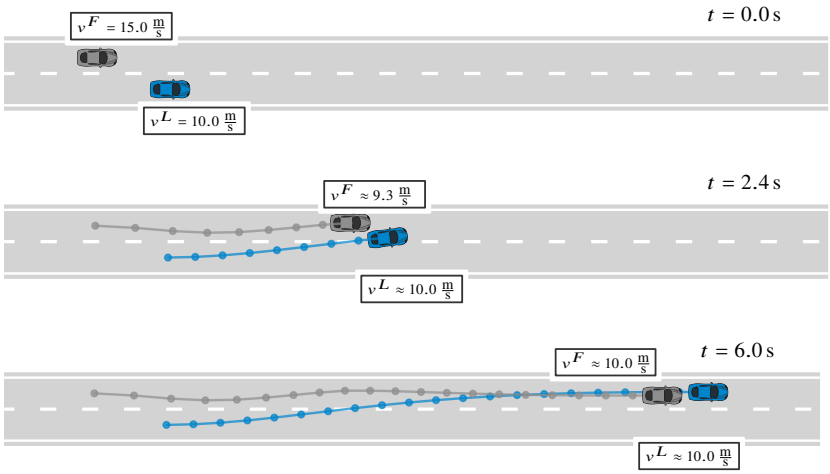


Figure 5.9: When only considering its own costs, the leader performs an aggressive lane change.

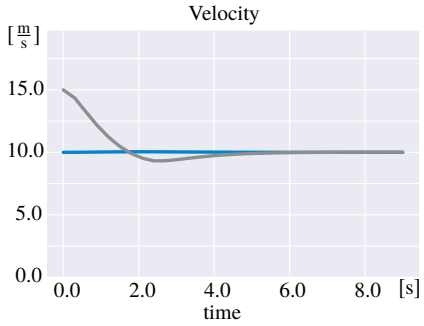


Figure 5.10: While the leader can perform a smooth lane change without accelerating, the follower has to brake harshly to avoid a collision.

cooperative cost function that includes the leader's and follwers's cost in the leader's objective is considered in the following:

$$J_{\text{cooperative}} = \alpha J_{F,\text{base}} + (1-\alpha) J_{L,\text{base}} \tag{5.18}$$

In this formulation, the variable $\alpha \in [0, 1]$ determines to which extent the leader's and the follower's cost are considered. Therefore, α provides a way to design different driving behaviors, ranging from overly aggressive to overly conservative.

The impact the parameter α has on the generated behavior is investigated in the following. Therefore, we consider a scenario including a mandatory lane change for the leader, see Fig. 5.11. The different α-dependent acceleration and velocity profiles for $\alpha = 0.0$, $\alpha = 0.5$ and $\alpha = 0.99$ are illustrated in Fig. 5.12.

In detail, for $\alpha = 0.0$, the leader does not accelerate, and all the discomfort has to be carried out by the follower. This represents the aggressive, egocentric behavior presented in the previous experiment. With a larger α, the leader increases its acceleration until reaching the acceleartion limits. In the case of $\alpha = 0.99$, the leader mostly considers the follower's cost and tries to intervene with its optimal plan as little as possible. This value of α generates a very conservative behavior similar to a predict-then-plan approach. With $\alpha = 0.5$, the leader's and the follower's cost are considered equaly, which leads to an approximtly equal distribution of discomfort. Note however, that, besides adjusting the acceleration during the lane change, the leader also adapts its stationary velocity depending on α.

Courtesy Constraints

The cooperative cost formulation presented in the previous experiment has the side effect that for $\alpha > 0.0$, the leader permanently drives faster than its desired velocity v^L_{ref}. For some scenarios, e.g., overtaking a slow-moving truck on the highway, a temporal increased velocity might be acceptable or even desirable for traffic efficiency. However, in most situations, a vehicle in front does not adapt its velocity to the desires of rear traffic.

An alternative to the cooperative cost formulation is introducing *courtesy* constraints. With these constraints, the leader's impact on others can be limited without altering the leader's objective function.

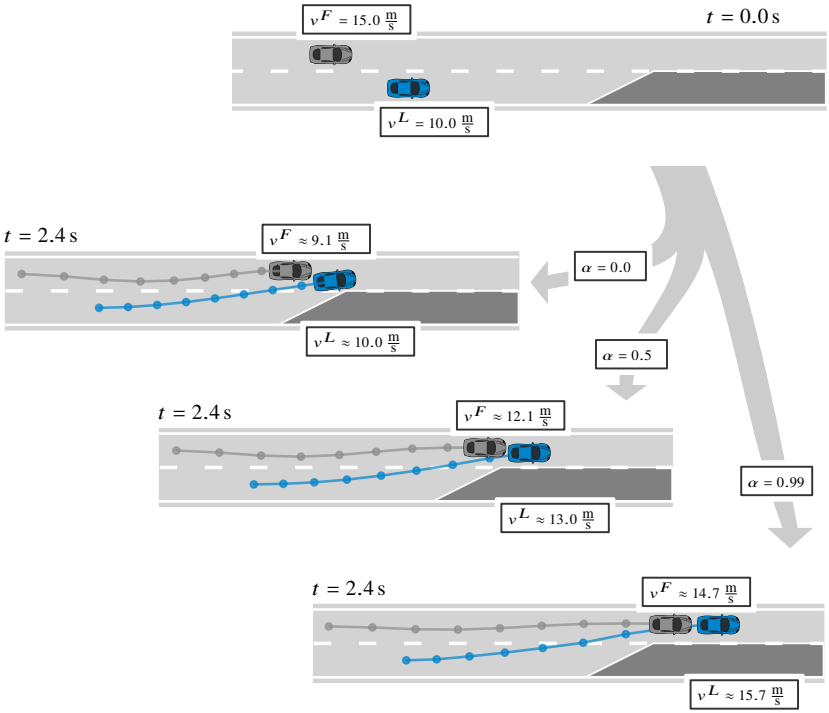


Figure 5.11: Scenario where the leader has to perform a lane change to the left. Depending on the value of α, different behaviors are generated, ranging from overly aggressive to overly conservative.

In this experiment, we introduce a constraint such that the leader is allowed to, at max, cause a deceleration of a_{limit} to the follower. To enforce this, the following constraints are added to the leader's OCP:

$$\mathbf{g}_{\text{courtesy},k} = a_k^F - a_{\text{limit}} \geq 0 \tag{5.19}$$

Here, a_k^F is the acceleration of the follower.

The effect of the courtesy constraint with $a_{\text{limit}} = -2.0$ on the considered merging scenario is illustrated in Fig. 5.13. By introducing the constraint, the leader accelerates during the lane change, which successfully limits the

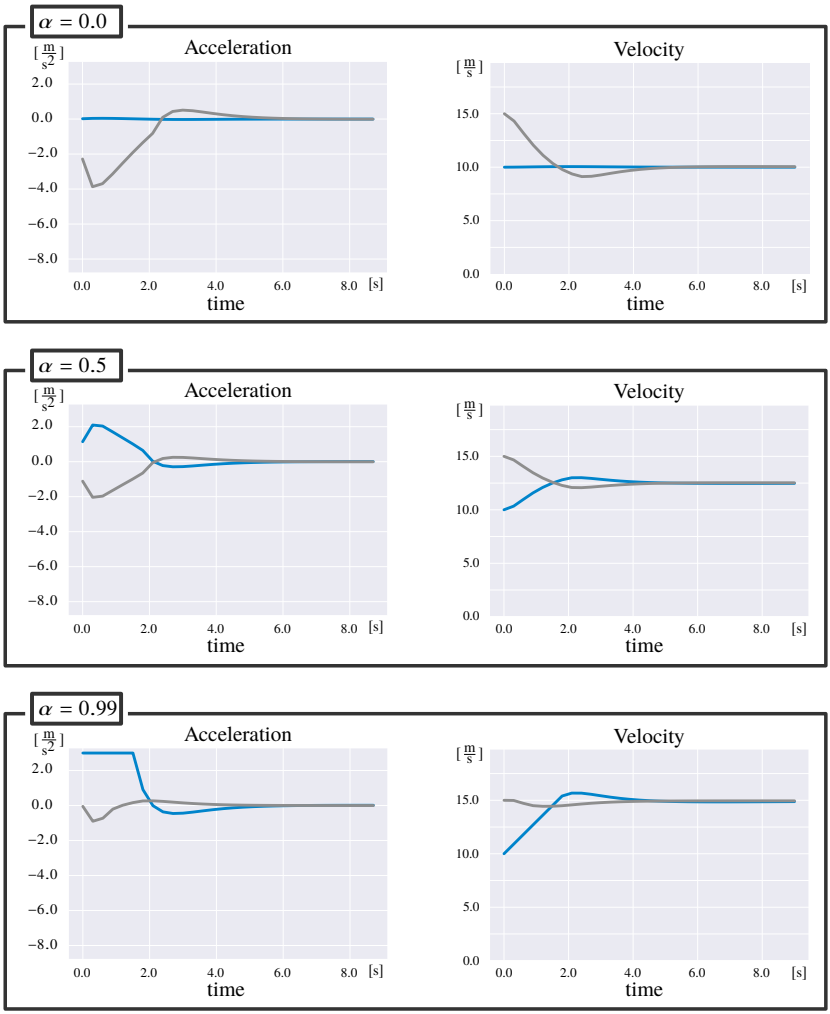


Figure 5.12: Depending on the value of α different acceleration and velocity profiles are obtained. Thereby, the longer α is, the more discompfort the leader accepts. Further, with different α the vehicles approach different stationary velocities which might significantly differ from their desired velocities.

induced deceleration to $-2.0\,\frac{\mathrm{m}}{\mathrm{s}^2}$. The velocity profiles are shown in Fig. 5.13b. Compared to the cooperative cost formulation, the leader returns to its desired velocity of $10.0\,\frac{\mathrm{m}}{\mathrm{s}}$ after the successful merge.

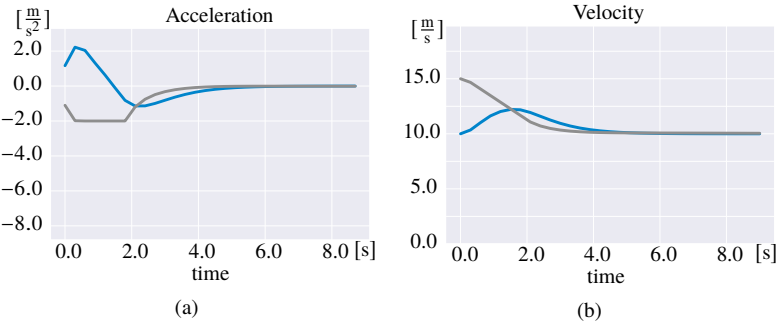


Figure 5.13: Acceleration a) and velocity b) profiles when planning with the courtesy constraints. Introduced these constraints into the leader's OCP generates a behavior that successfully limit the follower's deceleration to $a_{\text{limit}} = -2.0$. Futher, after the merge is complited, the leader returns to its desired velocitiy.

Both, the cooperative cost and the courtesy constraint method have traffic scenarios where they are particularly suited to generate a desirable driving behavior. E.g., when overtaking a slower driving vehicle on the highway, the cooperative cost formulation might be more suitable as it leads to a temporal increase in velocity. In contrast, for a merging scenario or a permanent lane change, the courteous constraint method might be the better choice since the leader returns to its desired velocity after the merge is completed.

5.4.5 Robustness Analysis

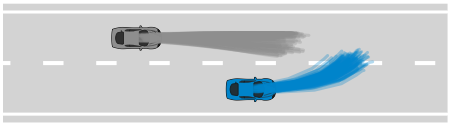
We perform Monte Carlo simulations to evaluate the robustness of the bi-level planner in different traffic scenarios. Randomness is introduced by perturbing the initial state of the leader and follower. More specifically, we uniformly sample the initial state perturbations from the intervals given in Table 5.3. For the following experiments, we solve 100 OCPs. We consider the same traffic situation and parameters as in the previous experiments. Further, we use the parameters and initial states stated in Table 5.1 and Table 5.2 if not mentioned otherwise.

Parameter	Value
x_0	$\pm 1.0\,\mathrm{m}$
y_0	$\pm 0.25\,\mathrm{m}$
ϕ_0	$\pm 5.0°$
v_0	$\pm 5.0\%$

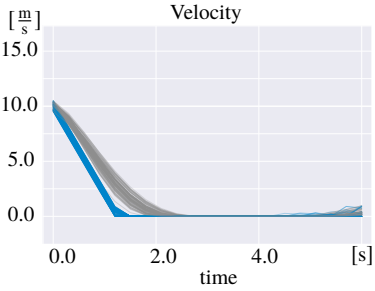
Table 5.3: Perturbation of the leader's and follower's inital states.

We first consider a scenario where the leader's focus is to alter the state of the follower. Therefore, the leader's objective is formulated to bring the follower to a full stop. This is achieved using the cost function presented in Eq. (5.16) with $v_{\mathrm{ref}}^F = 0.0$.

Fig. 5.14 shows that the leader generates a trajectory that successfully slows down the follower to a full stop for each sampled initial state.



(a) Trajectories of the leader and follower.

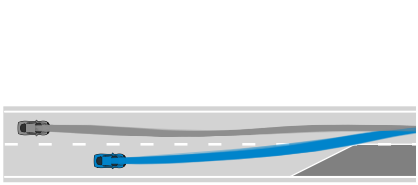


(b) Velocities of the leader and follower.

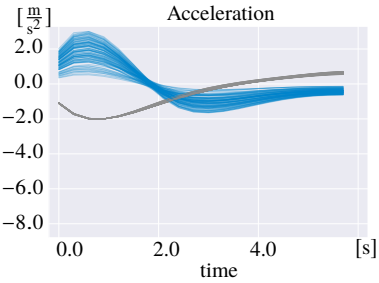
Figure 5.14: Results of the Monte Carlo simulation for 100 randomly sampled initial states. The leaders objective is to bring the follower to a full stop.

Next, a merging scenario where the leader plans its behavior applying the courtesy constraint method is considered. The leader starts from the right lane and performs a lane change to the left. The follower has an increased desired velocity of $v_{\mathrm{ref}}^F = 15.0\,\frac{\mathrm{m}}{\mathrm{s}}$. All other parameters are stated in Table 5.1. The maximum allowed induced deceleration is $a_{\mathrm{limit}} = -2.0$.

The results of the Monte Carlo simulation are illustrated in Fig. 5.15. As shown in Fig. 5.15a, the proposed algorithm reliability solves the merge problem.



(a) Trajectories of the leader and follower.



(b) Acceleration of the leader and follower.

Figure 5.15: Results of the Monte Carlo simulation for 100 randomly sampled initial states considering a merging scenario including courtesy constraints. As can be seen, the follower's deceleration is reliably limited to $a_{\text{limit}} = -2.0$.

Further, all obtained solutions satisfy the introduced courtesy constraints and limit the impact imposed on the follower to $-2.0\,\frac{\text{m}}{\text{s}^2}$, see Fig. 5.15b.

Even though the formulated bi-level optimization problem is highly nonlinear and non-smooth, the empirical data suggests that the proposed algorithm is able to robustly solve interactive traffic scenarios.

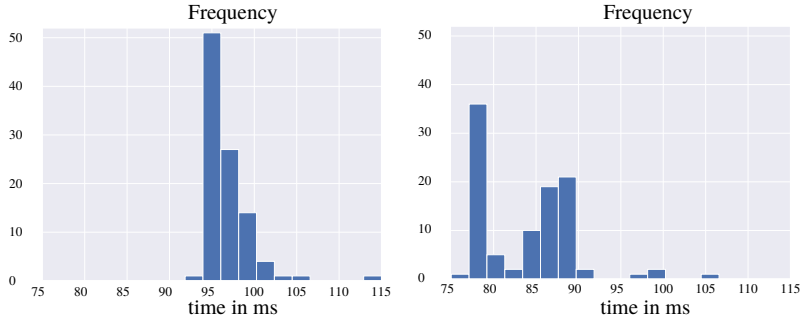
5.4.6 Runtime Experiments

The presented method for interaction-aware trajectory planning computes an open-loop solution for the AV. More precisely, the control inputs are functions of time and not of the state. To adapt to unforeseen changes in the environment, the algorithm needs to run in an MPC fashion. For MPC, a sufficiently high update rate is crucial. Therefore, we analyze the performance of the algorithm with a proof of concept MPC implementation.

The MPC was implemented in Python. All necessary derivatives were calculated using the open-source software CasADi [And+19]. CasADi utilizes automatic differentiation methods to accurately calculate the derivatives. Compared to, finite difference methods, automatic differentiation is faster and more accurate. Further, IPOPT [WB06] was used to solve the formulated NLPs. IPOPT is a general-purpose solver for large-scale nonlinear problems. We cold started the IPOPT solver with a feasible solution of the desired driving maneuver, which

we obtained by sequentially solving a single vehicle OCP, as in Eq. (5.8), first for the leader and then for the follower. This initialization was only performed for the very first iteration of the planner. All subsequent iterations were warm started with the solution of the previous iteration. To get a better initial guess, the previous solution was shifted by the duration between the planning iterations.

The timing results were obtained by considering a merging scenario, with the two most relevant methods for the application to real traffic, namely, the cooperative cost function method, with $\alpha = 0.5$ and the courtesy constraints method, with $a_{\text{limit}} = -2.0$. We simulate each method for 9.0 s. A horizon length of $N = 30$ steps is considered for the MPC. Further parameters were taken over from Table 5.1. The runtime results are obtained by running the MPC implementation 100 times on the merging scenario with both methods. The mean computation time over the 100 simulation runs are shown in the histogram in Fig. 5.16. Additionally, the mean and standard deviation of the mean computation times are listed in Table 5.4. All timing results were obtained on an Intel Core i7-8565U CPU with a clockrate of 1.80GHz.



(a) Cooperative cost method

(b) Courtesy constraint method

Figure 5.16: Mean computation times obtained by running the MPC implementation 100 times on the merging scenario with the cooperative cost and the courtesy constraint method.

Even though the experiments were conducted with an MPC implementation that leaves great potential for improvements, we could already demonstrate our algorithm's real-time capability with mean computation times of 96.82 ms and 83.85 ms, respectively. The presented results can be considered a conservative estimate of the achievable performance. However, in the future, this could be

Method	μ	σ
cooperative cost	96.82 ms	2.61 ms
courtesy constraints	83.85 ms	5.50 ms

Table 5.4: Mean and standard deviation of the mean computation times obtained by running the MPC implementation 100 times.

greatly improved by utilizing tailored solvers and implementing the approach in a high-performance programming language, e.g., C++.

5.5 Algorithm Discussion

A core assumption that we made to obtain the model for interaction-aware planning, stated in Eq. (5.3), is that the human does not try to influence the AV but rather reacts to its actions. According to [Sad+16b; Sad+18], this is a valid assumption for a wide range of interactive scenarios. Further, compared to a Nash equilibrium, it might even be the better model for how humans act in interative situations since humans typically do not solve games in their everyday lives when they are not playing chess [HZ02].

The formulated NLP, Eq. (5.6h), is a non-convex and non-smooth problem. As such, one can not expect to find globally optimal solutions. However, we use derivative-based optimization methods to find locally optimal solutions. These methods require an initial guess, which at the same time sets the considered homotopy, as the solutions of local methods typically are in the same homotopy as the initial guess. In the context of automated driving, homotopies are often thought of as maneuvers. Thus, we use the initial guess to encode the desired driving maneuver. Via the experiments, we empirically observe that initializing with a rough, but feasible initial guess of the desired maneuver is sufficient to reliably solve the problem. To take multiple maneuvers into consideration, it is advisable to combine the presented approach with a global method. E.g., a higher abstraction behavior planner based on an arbitration scheme as in [OBL20] could be used to generate good initial guesses.

The focus of the experiments was to analyze the capabilities and the performance of the proposed bi-level planner. As such, the algorithm was evaluated in a

tailored simulation environment, where one important modeling assumption was that the human driver is always attentive. However, in real traffic, this is not the case, and human drivers are sometimes distracted and do not respond to the actions of the AV. Therefore, the presented algorithm needs to be combined with an intention estimation, e.g, as presented in Section 4.2, to cope with unattentive drivers.

5.6 Summary

In this chapter, we presented an algorithm that is able to generate interaction-aware trajectories for AVs. The interaction between a HD and an AV is modeled as a Stackelberg game, where the human responds rationally to the AV's actions optimizing its own objective. This leads to a nested optimization problem which we approximate by MPC based on a bi-level optimization formulation. To solve this, we reformulated the problem into a single-level representation, exploiting the assumption that the human will act optimally with respect to its objective function. We solve the obtained NLP using derivative-based optimization methods. The presented algorithm is able to solve the interaction-aware trajectory planning problem in a continuous state and input space. Further, general nonlinear state and input constraints can be considered, which allows for an accurate dynamics model.

The algorithm enables the AV to anticipate how surrounding HDs will react to its actions. This gives the AV the possibility to deliberately influence the state of the human. Here, simply encoding the desired effect into the AV's objective function is enough to generate complex, interaction rich behavior to achieve the desired result. No hand designed decision heuristic, e.g.,

> Rule:
>
> *In order to slow down the follower one first needs to overtake the follower to be infront and then needs to brake while at the same time driving in the middle of the street to prevent the follower form overtaking again*

is needed but such a highly interactive behavior directly emerges out of the optimization. This interactive behavior does not only occur if incentivized

in the AV's objective function, but also emerges to further optimize the AV's behavior.

However, care must be taken to avoid that the AV exploits interactions to further optimize its own objective, and thereby generates an overly aggressive driving behavior. To prevent such a overly aggressive behavior, the AV's objective is extended to also consider the costs of the HD. A parameter α was introduced, which provides the ability to easily adjust the degree to which the impact on others is considered. This enables generating driving behavior ranging from overly conservative to overly aggressive.

As an alternative to modifying the AV's objective function, we presented a strategy to establish courtesy in the planning algorithm. Here, courtesy is introduced via constraints which limit the impact the AV's actions are allowed to have on a HD. These constraints allow a motion planner to utilizes an egocentric objective function, provided that a certain acceptable deceleration imposed on other vehicles is not exceeded.

The experiments demonstrated the efficacy of our algorithm and suggest that the algorithm can be used in challenging interactive driving scenarios. Further, we could achieve real-time performance even with an unoptimized proof-of-concept implementation. It is worth noting, that the presented algorithm is not limited to the context of automated driving but can be applied to other fields of robotics, where considering interaction among robots and humans is essential for efficient operation.

6 Conclusion & Outlook

State-of-the-art planning algorithms often follow a pipeline structure, where the future motion of surrounding objects is predicted first, and in a subsequent step, the automated vehicle (AV) plans its motion, treating others as immutable moving objects. These algorithms are incapable of capturing the interactive nature of traffic, i.e., how the predictions of others might change due to the AV's actions, thereby reducing interactions to simple collision avoidance. While this separation is a useful simplification in many situations, it can lead to suboptimal, overly conservative driving behavior, especially in dense traffic.

In this thesis, two algorithms to plan interaction-aware trajectories for AVs in mixed traffic were proposed. Hereby, we refer to mixed traffic as traffic in which AVs and human-driven vehicles share the road. The aim of the proposed algorithms is to enable AVs to drive more naturally among humans by overcoming the overly conservative behavior generated by state-of-the-art planning algorithms. The key inside into planning interaction-aware trajectories is to get rid of the structural limitation of state-of-the-art planners and solve both, the prediction of others and the planning of the AV's motion simultaneously as a joint problem formulation. We were able to show that challenging traffic situations could be mastered with these approaches, which would not have been possible with state-of-the-art planning approaches.

The first algorithm builds upon the assumption that interactions among traffic participants can be approximated by cooperative multi-agent planning. The underlying idea is that all traffic participants are on the same team, ensuring that each agent is driving comfortably and reaching its goal while avoiding collisions. The AV calculates coupled trajectories by optimizing a joint cost function, assuming that the human driver will also approximately follow those trajectories.

A central element of this algorithm is a novel multi-agent trajectory planning formulation. Hereby, the trajectory planning problem for multiple agents is

formulated as a mixed-integer quadratic program (MIQP) where continuous variables represent the flat output of each vehicle, and the maneuver variants are represented by discrete variables via collision avoidance constraints. The proposed formulation is able to consider the entire configuration space and globally optimal solutions can be guaranteed; thus, no separation of maneuver selection and trajectory optimization is required. Thanks to this property a major limitation of local methods, namely the combinatorial problem, which originates from the non-convexity of motion planning, can be overcome without the restriction of a discretized search space. The proposed formulation thus represents a hybrid approach that combines the advantages of global and local methods. Numerical experiments have shown that this formulation significantly outperforms existing trajectory planning approaches for ensembles consisting of multiple automated vehicles.

While the MIQP formulation for multi-agent planning poses a stand-alone contribution, it needs to be extended to cope with uncertain driving behavior of humans in mixed traffic. This was achieved by introducing multiple intention models to capture how humans might react during interactions. A generic formulation for these intention models was proposed, making them independent of the traffic scenario or map layout. Based on an estimated probability distribution over the intention models, the motion of the AV is generated. Hereby, instead of optimizing for a single intention, e.g., the most likely one, the algorithm is able to optimize the behavior for multiple different evolutions of the scene. This enables the AV to drive maneuver-neutral until a situation becomes more certain. Further, maneuver decisions directly emerge from optimization and no hand-designed heuristics are necessary. In conclusion, the proposed algorithm enables AVs to cope with uncertain human behavior in highly interactive situations and thus enables AVs to drive more naturally among humans in dense traffic.

The second algorithm is based on a reformulation of a bi-level optimization problem that frames interaction among a human driver and a AV as a Stackelberg game. This game-theoretic formulation assumes that the human will optimize its behavior in response to the AV's behavior, which leads to a nested optimization problem. Compared to the first algorithm, this formulation considers interaction as a direct link between the AV's action and the action the human will take in response. In contrast to previous works, the algorithm is able to consider general nonlinear state and input constraints, which ensures dynamic feasibility

of the generated trajectories. Further, several ways were proposed how the impact the AV's actions have on surrounding vehicles can be considered. This prevents the AV from exploiting interactions that would otherwise lead to overly aggressive driving behavior. In literature, this overly aggressive behavior has been identified as a problem of interaction-aware planning algorithms, which we mitigate by introducing methods to promote cooperation and courtesy into the planning.

The evaluation of the algorithm demonstrated its ability to deliberately influence the state of the human via interactions. Further, it has been shown that interactive behavior also emerges out of the AV's desire for efficiency. Real-time performance could be achieved with a proof-of-concept implementation, which holds the potential for further runtime improvements. The conducted Monte Carlo simulations additionally demonstrated the efficacy and the robustness of the proposed algorithm and suggested that it can reliably solve challenging interactive driving scenarios. It is worth noting that the proposed game-theoretic formulation is not limited to automated driving but can be applied to other robotic domains where considering interactions between robots and humans is essential for efficient operation.

The game-theoretic algorithm for interaction-aware trajectory optimization enables AVs to be aware of the impact their actions have on surrounding vehicles. Further, it provides AVs with a model of how their actions link to actions humans take as a response. This link lays the foundation for the development of future algorithms that are able to purposefully exploit interactions to gain information about states of human drivers which are not directly observable, e.g., their willingness to cooperate or wether they are attentive or not. For example, in a traffic situation, if it is unclear whether a human driver who needs to be interacted with is paying attention or not, the automated vehicle could perform targeted actions to provoke a response from the human. Based on the reaction, conclusions can then be drawn about the hidden states of the human.

While for the purpose of this thesis a good estimate of the cost function of a human was assumed to be given, learning such functions from naturalistic driving datasets is a future direction to improve the results of the presented algorithms. One possibility to obtain such functions is via inverse reinforcement learning.

The presented algorithms enable AVs to consider interactions in their planning which allows more natural driving among humans. However, before these algorithms can be deployed in real traffic they need to be combined with formal methods which can guarantee safety in case humans behave unexpectedly. A system architecture based on an arbitration structure, as proposed in [OBL20], is particularly suited for this.

References

[AdL16] F. Altche and A. de La Fortelle. “Analysis of Optimal Solutions to Robot Coordination Problems to Improve Autonomous Intersection Management Policies”. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. Gotenburg, Sweden: IEEE, June 2016, pp. 86–91. ISBN: 978-1-5090-1821-5. DOI: 10.1109/IVS.2016.7535369 (cit. on p. 15).

[AM18] B. Alrifaee and J. Maczijewski. “Real-Time Trajectory Optimization for Autonomous Vehicle Racing Using Sequential Linearization”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, Suzhou, China, June 2018, pp. 476–483. DOI: 10.1109/IVS.2018.8500634 (cit. on p. 15).

[And+19] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. “CasADi: A Software Framework for Nonlinear Optimization and Optimal Control”. In: *Mathematical Programming Computation* 11.1 (Mar. 2019), pp. 1–36. ISSN: 1867-2949, 1867-2957. DOI: 10.1007/s12532-018-0139-4 (cit. on p. 111).

[Ani05] M. Anitescu. “On Using the Elastic Mode in Nonlinear Programming Approaches to Mathematical Programs with Complementarity Constraints”. In: *SIAM Journal on Optimization* 15.4 (Jan. 1, 2005), pp. 1203–1236. ISSN: 1052-6234. DOI: 10.1137/S1052623402401221 (cit. on p. 41).

[APdL17] F. Altché, P. Polack, and A. de La Fortelle. “High-Speed Trajectory Planning for Autonomous Vehicles Using a Simple Dynamic Model”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. Yokohama, Japan, Oct. 2017, pp. 1–7. DOI: 10.1109/ITSC.2017.8317632 (cit. on p. 15).

[Ara22] S. Aradi. “Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.2 (Feb. 2022), pp. 740–759. ISSN: 1558-0016. DOI: 10.1109/TITS.2020.3024655 (cit. on p. 11).

[Ban+18] S. Bansal, A. Cosgun, A. Nakhaei, and K. Fujimura. “Collaborative Planning for Mixed-Autonomy Lane Merging”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, Oct. 2018, pp. 4449–4455. DOI: 10.1109/IROS.2018.8594197 (cit. on pp. 18, 21).

[BBL07] E. Bertolazzi, F. Biral, and M. D. Lio. “Real-Time Motion Planning for Multibody Systems”. In: *Multibody System Dynamics* 17.2 (Apr. 1, 2007), pp. 119–139. ISSN: 1573-272X. DOI: 10.1007/s11044-007-9037-7 (cit. on p. 22).

[BDG19] A. Britzelmeier, A. Dreves, and M. Gerdts. “Numerical Solution of Potential Games Arising in the Control of Cooperative Automatic Vehicles”. In: *2019 Proceedings of the Conference on Control and Its Applications (CT)*. Proceedings. Philadelphia, USA: Society for Industrial and Applied Mathematics, Jan. 1, 2019, pp. 38–45. DOI: 10.1137/1.9781611975758.7 (cit. on p. 20).

[Ben+15] P. Bender, Ö. Ş. Taş, J. Ziegler, and C. Stiller. “The Combinatorial Aspect of Motion Planning: Maneuver Variants in Structured Environments”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. Seoul, South Korea, June 2015, pp. 1386–1392. DOI: 10.1109/IVS.2015.7225909 (cit. on pp. 16, 69).

[Bet98] J. T. Betts. “Survey of Numerical Methods for Trajectory Optimization”. In: *Journal of Guidance, Control, and Dynamics* 21.2 (Mar. 1, 1998), pp. 193–207. DOI: 10.2514/2.4231 (cit. on p. 14).

[BKO19] M. Bansal, A. Krizhevsky, and A. Ogale. “ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst”. In: *Proceedings of Robotics: Science and Systems*. Freiburg, Germany, June 2019. DOI: 10.15607/RSS.2019.XV.031 (cit. on p. 10).

[BL18] C. Burger and M. Lauer. “Cooperative Multiple Vehicle Trajectory Planning Using MIQP”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, Hawaii, USA, Nov. 2018, pp. 602–607. DOI: 10.1109/ITSC.2018.8569776 (cit. on pp. 16, 18, 52).

[Boj+16] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. “End to End Learning for Self-Driving Cars”. Apr. 25, 2016. URL: http://arxiv.org/abs/1604.07316 (visited on 06/27/2022) (cit. on pp. 11, 12).

[Bri+19] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora. “Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments”. In: *IEEE Robotics and Automation Letters* 4.4 (Oct. 2019), pp. 4459–4466. ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2929976 (cit. on p. 98).

[BS07] G. Bouza and G. Still. “Mathematical Programs with Complementarity Constraints: Convergence Properties of a Smoothing Method”. In: *Mathematics of Operations Research* 32.2 (2007), pp. 467–483. ISSN: 0364-765X. JSTOR: 25151800 (cit. on p. 41).

[BSL20] C. Burger, T. Schneider, and M. Lauer. “Interaction Aware Cooperative Trajectory Planning for Lane Change Maneuvers in Dense Traffic”. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. Rhodes, Greece, Sept. 2020, pp. 1–8. DOI: 10.1109/ITSC45102.2020.9294638 (cit. on pp. 18, 68).

[Bur+17] C. Burger, P. F. Orzechowski, Ö. Ş. Taş, and C. Stiller. “Rating Cooperative Driving: A Scheme for Behavior Assessment”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. Yokohama, Japan, Oct. 2017, pp. 1–6. DOI: 10.1109/ITSC.2017.8317794 (cit. on pp. 21, 64).

[Bur+22] C. Burger, J. Fischer, F. Bieder, Ö. Ş. Taş, and C. Stiller. “Interaction-Aware Game-Theoretic Motion Planning for Automated Vehicles using Bi-level Optimization”. In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. Macau, China, Oct. 2022, pp. 1–6 (cit. on p. 87).

[BV04] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004. 716 pp. ISBN: 978-0-521-83378-3 (cit. on pp. 30, 31).

[CFS13] G. R. de Campos, P. Falcone, and J. Sjöberg. "Autonomous Cooperative Driving: A Velocity-Based Negotiation Approach for Intersection Crossing". In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. The Hague, Netherlands, Oct. 2013, pp. 1456–1461. DOI: 10.1109/ITSC.2013.6728435 (cit. on p. 18).

[Cho+05] H. M. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementation*. Ed. by H. M. Choset. Intelligent Robotics and Autonomous Agents. Cambridge, USA: MIT Press, 2005. 603 pp. ISBN: 978-0-262-03327-5 (cit. on p. 8).

[CKA95] H. Chen, H. Kremling, and F. Allgöwer. "Nonlinear Predictive Control of a Benchmark CSTR". In: *Proceedings of the 3rd European Control Conference, Rome,Italy.* (Jan. 1, 1995), pp. 3247–3252 (cit. on pp. 39, 93).

[CMS05] B. Colson, P. Marcotte, and G. Savard. "Bilevel Programming: A Survey". In: *4OR: A Quarterly Journal of Operations Research* 3.2 (June 1, 2005), pp. 87–107. ISSN: 1614-2411. DOI: 10.1007/s10288-005-0071-0 (cit. on p. 37).

[DG18] A. Dreves and M. Gerdts. "A Generalized Nash Equilibrium Approach for Optimal Control Problems of Autonomous Cars". In: *Optimal Control Applications and Methods* 39.1 (2018), pp. 326–342. ISSN: 1099-1514. DOI: 10.1002/oca.2348 (cit. on pp. 19, 20).

[Die+06] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber. "Fast Direct Multiple Shooting Algorithms for Optimal Robot Control". In: *Fast Motions in Biomechanics and Robotics*. Ed. by M. Diehl and K. Mombaur. Vol. 340. Lecture Notes in Control and Information Sciences. Berlin, Germany: Springer Berlin Heidelberg, 2006, pp. 65–93. ISBN: 978-3-540-36118-3. DOI: 10.1007/978-3-540-36119-0_4 (cit. on p. 44).

[Dij59] E. W. Dijkstra. “A Note on Two Problems in Connexion with Graphs”. In: *Numerische Mathematik* 1.1 (Dec. 1, 1959), pp. 269–271. ISSN: 0945-3245. DOI: 10.1007/BF01386390 (cit. on p. 10).

[DZ20] S. Dempe and A. Zemkoho, eds. *Bilevel Optimization: Advances and Next Challenges*. Springer Optimization and Its Applications. Cham, Switzerland: Springer International Publishing, 2020. ISBN: 978-3-030-52118-9. DOI: 10.1007/978-3-030-52119-6 (cit. on p. 37).

[EKK20] K. Esterle, T. Kessler, and A. Knoll. “Optimal Behavior Planning for Autonomous Driving: A Generic Mixed-Integer Formulation”. In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. Las Vegas, USA, Oct. 2020, pp. 1914–1921. DOI: 10.1109/IV47402.2020.9304743 (cit. on p. 16).

[ES14] M. Elbanhawi and M. Simic. “Sampling-Based Robot Motion Planning: A Review”. In: *IEEE Access* 2 (2014), pp. 56–77. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2014.2302442 (cit. on p. 10).

[ES17] J. Eilbrecht and O. Stursberg. “Cooperative Driving Using a Hierarchy of Mixed-Integer Programming and Tracking Control”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, USA, June 2017, pp. 673–678. DOI: 10.1109/IVS.2017.7995795 (cit. on p. 55).

[ES18a] J. Eilbrecht and O. Stursberg. “Optimization-Based Maneuver Automata for Cooperative Trajectory Planning of Autonomous Vehicles”. In: *2018 European Control Conference (ECC)*. Limassol, Cyprus, June 2018, pp. 82–88. DOI: 10.23919/ECC.2018.8550422 (cit. on p. 16).

[ES18b] J. Eilbrecht and O. Stursberg. “Optimization-Based Maneuver Automata for Cooperative Trajectory Planning of Autonomous Vehicles”. In: *2018 European Control Conference (ECC)*. Limassol, Cyprus, June 2018, pp. 82–88. DOI: 10.23919/ECC.2018.8550422 (cit. on p. 35).

[Eve+16] N. Evestedt, E. Ward, J. Folkesson, and D. Axehill. “Interaction Aware Trajectory Planning for Merge Scenarios in Congested Traffic Situations”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Rio de Janeiro,

Brazil, Nov. 2016, pp. 465–472. DOI: 10.1109/ITSC.2016.7795596 (cit. on pp. 18, 21).

[FB11] C. Frese and J. Beyerer. "A Comparison of Motion Planning Algorithms for Cooperative Collision Avoidance of Multiple Cognitive Automobiles". In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. Baden-Baden, Germany, June 2011, pp. 1156–1162. DOI: 10.1109/IVS.2011.5940489 (cit. on p. 63).

[Fis+19] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan. "Hierarchical Game-Theoretic Planning for Autonomous Vehicles". In: *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, May 2019, pp. 9590–9596. DOI: 10.1109/ICRA.2019.8794007 (cit. on pp. 19, 20).

[FJQ99] F. Facchinei, H. Jiang, and L. Qi. "A Smoothing Method for Mathematical Programs with Equilibrium Constraints". In: *Mathematical Programming* 85.1 (May 1, 1999), pp. 107–134. ISSN: 1436-4646. DOI: 10.1007/s10107990015a (cit. on p. 41).

[FLP98] M. Fukushima, Z.-Q. Luo, and J.-S. Pang. "A Globally Convergent Sequential Quadratic Programming Algorithm for Mathematical Programs with Linear Complementarity Constraints". In: *Computational Optimization and Applications* 10.1 (Apr. 1, 1998), pp. 5–34. ISSN: 1573-2894. DOI: 10.1023/A:1018359900133 (cit. on p. 41).

[FP97] M. C. Ferris and J. S. Pang. "Engineering and Economic Applications of Complementarity Problems". In: *SIAM Review* 39.4 (Jan. 1997), pp. 669–713. ISSN: 0036-1445, 1095-7200. DOI: 10.1137/S0036144595285963 (cit. on p. 39).

[Fri+20] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin. "Efficient Iterative Linear-Quadratic Approximations for Nonlinear Multi-Player General-Sum Differential Games". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. Paris, France, May 2020, pp. 1475–1481. DOI: 10.1109/ICRA40945.2020.9197129 (cit. on pp. 19, 20).

[Fuc05] S. Fuchshumer. "Algebraic Linear Identification, Modelling, and Applications of Flatness-based Control". PhD thesis. Linz, Austria: Johannes Kepler Universität, 2005 (cit. on p. 27).

[Fun+17] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes. “Collision Avoidance and Stabilization for Autonomous Vehicles in Emergency Scenarios”. In: *IEEE Transactions on Control Systems Technology* 25.4 (July 2017), pp. 1204–1216. ISSN: 1558-0865. DOI: 10.1109/TCST.2016.2599783 (cit. on p. 15).

[Gao+12] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J. K. Hedrick, and F. Borrelli. “Spatial Predictive Control for Agile Semi-Autonomous Ground Vehicles”. In: *11th international symposium on advanced vehicle control*. Vol. 2. Seoul, South Korea, 2012, pp. 1–6 (cit. on p. 15).

[Gon+16] D. González, J. Pérez, V. Milanés, and F. Nashashibi. “A Review of Motion Planning Techniques for Automated Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.4 (Apr. 2016), pp. 1135–1145. ISSN: 1524-9050. DOI: 10.1109/TITS.2015.2498841 (cit. on pp. 9–11, 15).

[Gra+12] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli. “Predictive Control for Agile Semi-Autonomous Ground Vehicles Using Motion Primitives”. In: *2012 American Control Conference (ACC)*. Montréal, Canada, June 2012, pp. 4239–4244. DOI: 10.1109/ACC.2012.6315303 (cit. on p. 15).

[Gra+18] M. Graf Plessen, D. Bernardini, H. Esen, and A. Bemporad. “Spatial-Based Predictive Control and Geometric Corridor Planning for Adaptive Cruise Control Coupled With Obstacle Avoidance”. In: *IEEE Transactions on Control Systems Technology* 26.1 (Jan. 2018), pp. 38–50. ISSN: 1558-0865. DOI: 10.1109/TCST.2017.2664722 (cit. on p. 15).

[Gur22] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2022. URL: https://www.gurobi.com (visited on 06/27/2022) (cit. on p. 35).

[Hal+22] J. Hall, A. Nurkanović, F. Messerer, and M. Diehl. “A Sequential Convex Programming Approach to Solving Quadratic Programs and Optimal Control Problems With Linear Complementarity Constraints”. In: *IEEE Control Systems Letters* 6 (2022), pp. 536–541. ISSN: 2475-1456. DOI: 10.1109/LCSYS.2021.3083467 (cit. on p. 41).

[HKS13] T. Hoheisel, C. Kanzow, and A. Schwartz. “Theoretical and Numerical Comparison of Relaxation Methods for Mathematical Programs with Complementarity Constraints”. In: *Mathematical Programming* 137.1 (Feb. 1, 2013), pp. 257–288. ISSN: 1436-4646. DOI: 10.1007/s10107-011-0488-5 (cit. on pp. 41, 94).

[HNR68] P. E. Hart, N. J. Nilsson, and B. Raphael. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”. In: *IEEE Transactions on Systems Science and Cybernetics* 4.2 (July 1968), pp. 100–107. ISSN: 2168-2887. DOI: 10.1109/TSSC.1968.300136 (cit. on p. 10).

[HR04] X. M. Hu and D. Ralph. “Convergence of a Penalty Method for Mathematical Programming with Complementarity Constraints”. In: *Journal of Optimization Theory and Applications* 123.2 (Nov. 1, 2004), pp. 365–390. ISSN: 1573-2878. DOI: 10.1007/s10957-004-5154-0 (cit. on p. 42).

[Hub+18] C. Hubmann, J. Schulz, G. Xu, D. Althoff, and C. Stiller. “A Belief State Planner for Interactive Merge Maneuvers in Congested Traffic”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, USA, Nov. 2018, pp. 1617–1624. DOI: 10.1109/ITSC.2018.8569729 (cit. on p. 19).

[Hub19] C. Hubmann. “Belief State Planning for Autonomous Driving: Planning with Interaction, Uncertain Prediction and Uncertain Perception”. PhD thesis. Karlsruhe, Germany: Karlsruher Institut für Technologie (KIT), 2019. DOI: 10.5445/IR/1000120841 (cit. on p. 12).

[HZ02] T. Hedden and J. Zhang. “What Do You Think I Think You Think?: Strategic Reasoning in Matrix Games”. In: *Cognition* 85.1 (Aug. 1, 2002), pp. 1–36. ISSN: 0010-0277. DOI: 10.1016/S0010-0277(02)00054-9 (cit. on p. 113).

[Jai+19] V. Jain, U. Kolbe, G. Breuel, and C. Stiller. “Reacting to Multi-Obstacle Emergency Scenarios Using Linear Time Varying Model Predictive Control”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. Paris, France, June 2019, pp. 1822–1829. DOI: 10.1109/IVS.2019.8813982 (cit. on pp. 15, 27).

[Kal13] J. Kallrath. *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*. Wiesbaden, Germany: Springer Fachmedien Wiesbaden, 2013. ISBN: 978-3-658-00689-1 978-3-658-00690-7. DOI: 10.1007/978-3-658-00690-7 (cit. on p. 33).

[Kan96] C. Kanzow. "Some Noninterior Continuation Methods for Linear Complementarity Problems". In: *SIAM Journal on Matrix Analysis and Applications* 17.4 (Oct. 1996), pp. 851–868. ISSN: 0895-4798, 1095-7162. DOI: 10.1137/S0895479894273134 (cit. on p. 41).

[Kar06] J. K. Karlof, ed. *Integer Programming: Theory and Practice*. Boca Raton, USA: Taylor&Francis, 2006. 316 pp. ISBN: 978-0-8493-1914-3 (cit. on p. 33).

[Kat+15] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka. "Real-Time Motion Planning Methods for Autonomous on-Road Driving: State-of-the-art and Future Research Directions". In: *Transportation Research Part C: Emerging Technologies* 60 (Nov. 1, 2015), pp. 416–442. ISSN: 0968-090X. DOI: 10.1016/j.trc.2015.09.011 (cit. on pp. 11, 15).

[KKJ17] A. Katriniok, P. Kleibaum, and M. Joševski. "Distributed Model Predictive Control for Intersection Automation Using a Parallelized Optimization Approach". In: *IFAC-PapersOnLine*. 20th IFAC World Congress 50.1 (July 1, 2017), pp. 5940–5946. ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2017.08.1492 (cit. on p. 15).

[KLM20] Y. Kim, S. Leyffer, and T. Munson. "MPEC Methods for Bilevel Optimization Problems". In: *Bilevel Optimization: Advances and Next Challenges*. Ed. by S. Dempe and A. Zemkoho. Springer Optimization and Its Applications. Cham, Switzerland: Springer International Publishing, 2020, pp. 335–360. ISBN: 978-3-030-52119-6. DOI: 10.1007/978-3-030-52119-6_12 (cit. on p. 39).

[KMS16] J. Karlsson, N. Murgovski, and J. Sjöberg. "Temporal vs. Spatial Formulation of Autonomous Overtaking Algorithms". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Rio de Janeiro, Brazil, Nov. 2016, pp. 1029–1034. DOI: 10.1109/ITSC.2016.7795682 (cit. on p. 15).

[Kre+16] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard. “Socially Compliant Mobile Robot Navigation via Inverse Reinforcement Learning”. In: *The International Journal of Robotics Research* 35.11 (Sept. 2016), pp. 1289–1307. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364915619772 (cit. on p. 18).

[Lab+08] L. Labakhua, U. Nunes, R. Rodrigues, and F. S. Leite. “Smooth Trajectory Planning for Fully Automated Passengers Vehicles: Spline and Clothoid Based Methods and Its Simulation”. In: *Informatics in Control Automation and Robotics: Selected Papers from the International Conference on Informatics in Control Automation and Robotics 2006*. Ed. by J. A. Cetto, J.-L. Ferrier, J. M. Costa dias Pereira, and J. Filipe. Lecture Notes Electrical Engineering. Berlin, Germany: Springer, 2008, pp. 169–182. ISBN: 978-3-540-79142-3. DOI: 10.1007/978-3-540-79142-3_14 (cit. on p. 10).

[LaV06] S. M. LaValle. *Planning Algorithms*. Cambridge, UK: Cambridge University Press, 2006. ISBN: 978-0-511-54687-7 978-0-521-86205-9. DOI: 10.1017/CBO9780511546877 (cit. on p. 8).

[LB14] R. Lot and F. Biral. “A Curvilinear Abscissa Approach for the Lap Time Optimization of Racing Vehicles”. In: *IFAC Proceedings Volumes*. Vol. 47. 19th IFAC World Congress 3. Cape Town, South Africa, Jan. 1, 2014, pp. 7559–7565. DOI: 10.3182/20140824-6-ZA-1003.00868 (cit. on p. 15).

[LDM15] A. Liniger, A. Domahidi, and M. Morari. “Optimization-Based Autonomous Racing of 1:43 Scale RC Cars”. In: *Optimal Control Applications and Methods* 36.5 (Sept. 2015), pp. 628–647. ISSN: 01432087. DOI: 10.1002/oca.2123. arXiv: 1711.07300 (cit. on pp. 15, 16).

[Liu+17] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng. “Path Planning for Autonomous Vehicles Using Model Predictive Control”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, USA, June 2017, pp. 174–179. DOI: 10.1109/IVS.2017.7995716 (cit. on p. 15).

[Liu+18] K. Liu, J. Gong, A. Kurt, H. Chen, and U. Ozguner. “Dynamic Modeling and Control of High-Speed Automated Vehicles for Lane Change Maneuver”. In: *IEEE Transactions on Intelligent*

Vehicles 3.3 (Sept. 2018), pp. 329–339. ISSN: 2379-8904. DOI: 10.1109/TIV.2018.2843177 (cit. on p. 15).

[LK99] S. LaValle and J. Kuffner. "Randomized Kinodynamic Planning". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*. Vol. 1. Detroit, MI, USA, May 1999, 473–479 vol.1. DOI: 10.1109/ROBOT.1999.770022 (cit. on p. 10).

[LKK16] D. Lenz, T. Kessler, and A. Knoll. "Tactical Cooperative Planning for Autonomous Highway Driving Using Monte-Carlo Tree Search". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. Gothenburg, Sweden, June 2016, pp. 447–453. DOI: 10.1109/IVS.2016.7535424 (cit. on p. 18).

[LL20] A. Liniger and J. Lygeros. "A Noncooperative Game Approach to Autonomous Racing". In: *IEEE Transactions on Control Systems Technology* 28.3 (May 2020), pp. 884–897. ISSN: 1558-0865. DOI: 10.1109/TCST.2019.2895282 (cit. on pp. 15, 19, 20).

[LLI12] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. "Risk Assessment at Road Intersections: Comparing Intention and Expectation". In: *2012 IEEE Intelligent Vehicles Symposium*. Madrid, Spain, June 2012, pp. 165–171. DOI: 10.1109/IVS.2012.6232198 (cit. on p. 78).

[LLN06] S. Leyffer, G. López-Calva, and J. Nocedal. "Interior Methods for Mathematical Programs with Complementarity Constraints". In: *SIAM Journal on Optimization* 17.1 (Jan. 1, 2006), pp. 52–77. ISSN: 1052-6234. DOI: 10.1137/040621065 (cit. on p. 41).

[LSM21] S. Lecleac'h, M. Schwager, and Z. Manchester. "LUCIDGames: Online Unscented Inverse Dynamic Games for Adaptive Trajectory Prediction and Planning". In: *IEEE Robotics and Automation Letters* (2021), pp. 1–1. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2021.3074880 (cit. on p. 19).

[LSM22] S. Le Cleac'h, M. Schwager, and Z. Manchester. "ALGAMES: A Fast Augmented Lagrangian Solver for Constrained Dynamic Games". In: *Autonomous Robots* 46.1 (Jan. 1, 2022), pp. 201–215. ISSN: 1573-7527. DOI: 10.1007/s10514-021-10024-7 (cit. on p. 20).

[LVL14] S. Lefèvre, D. Vasquez, and C. Laugier. “A Survey on Motion Prediction and Risk Assessment for Intelligent Vehicles”. In: *ROBOMECH Journal* 1.1 (July 23, 2014), p. 1. ISSN: 2197-4225. DOI: 10.1186/s40648-014-0001-z (cit. on p. 17).

[MGK07] J. Maciejowski, P. Goulart, and E. Kerrigan. “Constrained Control Using Model Predictive Control”. In: *Advanced Strategies in Control Systems with Input and Output Constraints*. Ed. by S. Tarbouriech, G. Garcia, and A. H. Glattfelder. Lecture Notes in Control and Information Sciences. Berlin, Germany: Springer, 2007, pp. 273–291. ISBN: 978-3-540-37010-9. DOI: 10.1007/978-3-540-37010-9_9 (cit. on p. 48).

[MPA18] C. Miller, C. Pek, and M. Althoff. “Efficient Mixed-Integer Programming for Longitudinal and Lateral Motion Planning of Autonomous Vehicles”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. Changshu, China, June 2018, pp. 1954–1961. DOI: 10.1109/IVS.2018.8500394 (cit. on p. 16).

[NAD20] A. Nurkanović, S. Albrecht, and M. Diehl. “Limits of MPCC Formulations in Direct Optimal Control with Nonsmooth Differential Equations”. In: *2020 European Control Conference (ECC)*. St. Petersburg, Russia, May 2020, pp. 2015–2020. DOI: 10.23919/ECC51009.2020.9143593 (cit. on p. 41).

[Ne+17] S. Ne, A. Cd, C. Rg, and R. Ja. “Implementing Conditional Inequality Constraints for Optimal Collision Avoidance”. In: *Journal of Aeronautics & Aerospace Engineering* 06.03 (2017). ISSN: 21689792. DOI: 10.4172/2168-9792.1000195 (cit. on p. 97).

[Not+20] G. Notomista, M. Wang, M. Schwager, and M. Egerstedt. “Enhancing Game-Theoretic Autonomous Car Racing Using Control Barrier Functions”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. Paris, France: IEEE, May 2020, pp. 5393–5399. ISBN: 978-1-72817-395-5. DOI: 10.1109/ICRA40945.2020.9196757 (cit. on p. 19).

[NS13] J. Nilsson and J. Sjöberg. “Strategic Decision Making for Automated Driving on Two-Lane, One Way Roads Using Model Predictive Control”. In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. Gold Coast, Australia, June 2013, pp. 1253–1258. DOI: 10.1109/IVS.2013.6629638 (cit. on p. 16).

[NW06] J. Nocedal and S. Wright. *Numerical Optimization*. 2nd ed. Springer Series in Operations Research and Financial Engineering. New York, USA: Springer-Verlag, 2006. ISBN: 978-0-387-30303-1. DOI: 10.1007/978-0-387-40065-5 (cit. on pp. 31, 32).

[OBL20] P. F. Orzechowski, C. Burger, and M. Lauer. "Decision-Making for Automated Vehicles Using a Hierarchical Behavior-Based Arbitration Scheme". In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. Las Vegas, NV, USA, Oct. 2020, pp. 767–774. DOI: 10.1109/IV47402.2020.9304723 (cit. on pp. 113, 120).

[Pad+16] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli. "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles". In: *IEEE Transactions on Intelligent Vehicles* 1.1 (Mar. 2016), pp. 33–55. ISSN: 2379-8858. DOI: 10.1109/TIV.2016.2578706 (cit. on pp. 9, 11, 12, 15).

[PB12] H. B. Pacejka and I. Besselink. *Tire and Vehicle Dynamics*. 3d edition. Oxford, UK: Butterworth-Heinemann Elsevier, 2012. ISBN: 978-0-08-097016-5 (cit. on p. 27).

[Pen+17] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang. "Perception, Planning, Control, and Coordination for Autonomous Vehicles". In: *Machines* 5.1 (1 Mar. 2017), p. 6. DOI: 10.3390/machines5010006 (cit. on p. 11).

[Ple17] M. G. Plessen. "Trajectory Planning of Automated Vehicles in Tube-like Road Segments". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. Yokohama, Japan, Oct. 2017, pp. 1–6. DOI: 10.1109/ITSC.2017.8317585 (cit. on pp. 15, 16).

[Pol+17] P. Polack, F. Altché, B. d'Andréa-Novel, and A. d. L. Fortelle. "The Kinematic Bicycle Model: A Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles?" In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, USA, June 2017, pp. 812–818. DOI: 10.1109/IVS.2017.7995816 (cit. on p. 96).

[PPB20] E. Pagot, M. Piccinini, and F. Biral. "Real-Time Optimal Control of an Autonomous RC Car with Minimum-Time Maneuvers and a Novel Kineto-Dynamical Model". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, USA, Oct. 2020, pp. 2390–2396. DOI: 10.1109/IROS45743.2020.9340640 (cit. on p. 15).

[Qia+16] X. Qian, F. Altché, P. Bender, C. Stiller, and A. d. L. Fortelle. "Optimal Trajectory Planning for Autonomous Driving Integrating Logical Constraints: An MIQP Perspective". In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. Rio de Janeiro, Brazil, Nov. 2016, pp. 205–210. DOI: 10.1109/ITSC.2016.7795555 (cit. on pp. 16, 55).

[Qia+17] X. Qian, F. Altché, J. Grégoire, and A. de La Fortelle. "Autonomous Intersection Management Systems: Criteria, Implementation and Evaluation". In: *IET Intelligent Transport Systems* 11.3 (2017), pp. 182–189. ISSN: 1751-9578. DOI: 10.1049/iet-its.2016.0043 (cit. on p. 15).

[Raj12] R. Rajamani. *Vehicle Dynamics and Control*. Mechanical Engineering Series. New York, USA: Springer New York, 2012. ISBN: 978-1-4614-1432-2 978-1-4614-1433-9. DOI: 10.1007/978-1-4614-1433-9 (cit. on p. 22).

[RCB17] U. Rosolia, A. Carvalho, and F. Borrelli. "Autonomous Racing Using Learning Model Predictive Control". In: *2017 American Control Conference (ACC)*. Seattle, USA, May 2017, pp. 5115–5120. DOI: 10.23919/ACC.2017.7963748 (cit. on p. 15).

[RMD20] J. Rawlings, D. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation and Design*. 2nd edition. Santa Barbara, USA: Nob Hill Publishing, LLC, 2020. ISBN: 978-0-9759377-5-4 (cit. on p. 48).

[RN21] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Fourth edition. Pearson Series in Artificial Intelligence. Upper Saddle River, New Jersey, USA: Pearson, 2021. ISBN: 978-0-13-461099-3 (cit. on p. 19).

[RW04] D. Ralph * and S. J. Wright. “Some Properties of Regularization and Penalization Schemes for MPECs”. In: *Optimization Methods and Software* 19.5 (Oct. 1, 2004), pp. 527–556. ISSN: 1055-6788. DOI: 10.1080/10556780410001709439 (cit. on p. 41).

[SA17] S. Söntges and M. Althoff. “Computing Possible Driving Corridors for Automated Vehicles”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, USA, June 2017, pp. 160–166. DOI: 10.1109/IVS.2017.7995714 (cit. on p. 16).

[Sad+16a] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan. “Information Gathering Actions over Human Internal State”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea, Oct. 2016, pp. 66–73. DOI: 10.1109/IROS.2016.7759036 (cit. on pp. 19, 20).

[Sad+16b] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan. “Planning for Autonomous Cars That Leverage Effects on Human Actions”. In: *Robotics: Science and Systems XII*. Michigan, USA: Robotics: Science and Systems Foundation, 2016. ISBN: 978-0-9923747-2-3. DOI: 10.15607/RSS.2016.XII.029 (cit. on pp. 19–21, 88, 113).

[Sad+18] D. Sadigh, N. Landolfi, S. S. Sastry, S. A. Seshia, and A. D. Dragan. “Planning for Cars That Coordinate with People: Leveraging Effects on Human Actions for Planning and Active Information Gathering over Human Internal State”. In: *Autonomous Robots* 42.7 (Oct. 1, 2018), pp. 1405–1426. ISSN: 1573-7527. DOI: 10.1007/s10514-018-9746-1 (cit. on pp. 19, 113).

[SAR18] W. Schwarting, J. Alonso-Mora, and D. Rus. “Planning and Decision-Making for Autonomous Vehicles”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 1.1 (May 28, 2018), pp. 187–210. ISSN: 2573-5144. DOI: 10.1146/annurev-control-060117-105157 (cit. on pp. 10, 11, 15).

[Sch+01] T. Schouwenaars, B. D. Moor, E. Feron, and J. How. “Mixed Integer Programming for Multi-Vehicle Path Planning”. In: *2001 European Control Conference (ECC)*. Porto, Portugal, Sept. 2001, pp. 2603–2608 (cit. on pp. 16, 55, 60).

[Sch+17] J. Schulz, K. Hirsenkorn, J. Löchner, M. Werling, and D. Burschka. "Estimation of Collective Maneuvers through Cooperative Multi-Agent Planning". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, USA, June 2017, pp. 624–631. DOI: 10.1109/IVS.2017.7995788 (cit. on p. 18).

[Sch+18a] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka. "Multiple Model Unscented Kalman Filtering in Dynamic Bayesian Networks for Intention Estimation and Trajectory Prediction". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, USA, Nov. 2018, pp. 1467–1474. DOI: 10.1109/ITSC.2018.8569932 (cit. on p. 12).

[Sch+18b] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus. "Safe Nonlinear Trajectory Generation for Parallel Autonomy With a Dynamic Vehicle Model". In: *IEEE Transactions on Intelligent Transportation Systems* 19.9 (Sept. 2018), pp. 2994–3008. ISSN: 1558-0016. DOI: 10.1109/TITS.2017.2771351 (cit. on p. 15).

[Sch+18c] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus. "Safe Nonlinear Trajectory Generation for Parallel Autonomy With a Dynamic Vehicle Model". In: *IEEE Transactions on Intelligent Transportation Systems* 19.9 (Sept. 2018), pp. 2994–3008. ISSN: 1558-0016. DOI: 10.1109/TITS.2017.2771351 (cit. on p. 17).

[Sch01] S. Scholtes. "Convergence Properties of a Regularization Scheme for Mathematical Programs with Complementarity Constraints". In: *SIAM Journal on Optimization* 11.4 (Jan. 1, 2001), pp. 918–936. ISSN: 1052-6234. DOI: 10.1137/S1052623499361233 (cit. on p. 41).

[SMD18] A. Sinha, P. Malo, and K. Deb. "A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications". In: *IEEE Transactions on Evolutionary Computation* 22.2 (Apr. 2018), pp. 276–295. ISSN: 1941-0026. DOI: 10.1109/TEVC.2017.2712906 (cit. on p. 37).

[Spi+20] R. Spica, E. Cristofalo, Z. Wang, E. Montijano, and M. Schwager. "A Real-Time Game Theoretic Planner for Autonomous Two-Player Drone Racing". In: *IEEE Transactions on Robotics* 36.5 (Oct. 2020), pp. 1389–1403. ISSN: 1941-0468. DOI: 10.1109/TRO.2020.2994881 (cit. on p. 19).

[SPV20] A. Serban, E. Poll, and J. Visser. “A Standard Driven Software Architecture for Fully Autonomous Vehicles”. In: *Journal of Automotive Software Engineering* 1 (Jan. 1, 2020). DOI: 10.2991/jase.d.200212.001 (cit. on p. 11).

[SS00] H. Scheel and S. Scholtes. “Mathematical Programs with Complementarity Constraints: Stationarity, Optimality, and Sensitivity”. In: *Mathematics of Operations Research* 25.1 (2000), pp. 1–22. ISSN: 0364-765X. JSTOR: 3690420 (cit. on p. 40).

[ST08] J. Smith and Z. Taskin. “A Tutorial Guide to Mixed-Integer Programming Models and Solution Techniques”. In: *Optimization in medicine and biology* (2008), pp. 521–548. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.464.6182&rep=rep1&type=pdf (visited on 06/27/2022) (cit. on p. 34).

[Sun+18] L. Sun, W. Zhan, M. Tomizuka, and A. D. Dragan. “Courteous Autonomous Cars”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain, Oct. 2018, pp. 663–670. DOI: 10.1109/IROS.2018.8593969 (cit. on pp. 20, 21).

[Sve+19] L. Svensson, M. Bujarbaruah, N. R. Kapania, and M. Törngren. “Adaptive Trajectory Planning and Optimization at Limits of Handling”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Macau, China, Nov. 2019, pp. 3942–3948. DOI: 10.1109/IROS40897.2019.8967679 (cit. on p. 15).

[Tak+89] A. Takahashi, T. Hongo, Y. Ninomiya, and G. Sugimoto. “Local Path Planning And Motion Control For Agv In Positioning”. In: *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems '. (IROS '89) 'The Autonomous Mobile Robots and Its Applications*. Tsukuba, Japan, Sept. 1989, pp. 392–397. DOI: 10.1109/IROS.1989.637936 (cit. on p. 14).

[Taş+16] Ö. Ş. Taş, F. Kuhnt, J. M. Zöllner, and C. Stiller. “Functional System Architectures towards Fully Automated Driving”. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. Gothenburg, Sweden, June 2016, pp. 304–309. DOI: 10.1109/IVS.2016.7535402 (cit. on p. 11).

[Taş+17] Ö. Ş. Taş, S. Hörmann, B. Schäufele, and F. Kuhnt. “Automated Vehicle System Architecture with Performance Assessment”. In: *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC)*. Yokohama, Japan, Oct. 2017, pp. 1–8. DOI: 10.1109/ITSC.2017.8317862 (cit. on p. 11).

[Taş+18] Ö. Ş. Taş, N. O. Salscheider, F. Poggenhans, S. Wirges, C. Bandera, M. R. Zofka, T. Strauss, J. M. Zöllner, and C. Stiller. “Making Bertha Cooperate–Team AnnieWAY’s Entry to the 2016 Grand Cooperative Driving Challenge”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.4 (2018), pp. 1262–1276. DOI: 10.1109/TITS.2017.2749974 (cit. on p. 15).

[THH00] M. Treiber, A. Hennecke, and D. Helbing. “Congested Traffic States in Empirical Observations and Microscopic Simulations”. In: *Physical Review E* 62.2 (Aug. 1, 2000), pp. 1805–1824. ISSN: 1063-651X, 1095-3787. DOI: 10.1103/PhysRevE.62.1805 (cit. on pp. 18, 79).

[TK10] P. Trautman and A. Krause. “Unfreezing the Robot: Navigation in Dense, Interacting Crowds”. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Taipei, Taiwan, Oct. 2010, pp. 797–803. DOI: 10.1109/IROS.2010.5654369 (cit. on pp. 2, 18).

[TS18] Ö. Ş. Taş and C. Stiller. “Limited Visibility and Uncertainty Aware Motion Planning for Automated Driving”. In: *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*. Changshu, China, June 2018, pp. 1171–1178. DOI: 10.1109/ivs.2018.8500369 (cit. on p. 15).

[Váz+20] J. L. Vázquez, M. Brühlmeier, A. Liniger, A. Rupenyan, and J. Lygeros. “Optimization-Based Hierarchical Motion Planning for Autonomous Racing”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, USA, Oct. 2020, pp. 2397–2403. DOI: 10.1109/IROS45743.2020.9341731 (cit. on p. 15).

[Ver+14] R. Verschueren, S. De Bruyne, M. Zanon, J. V. Frasch, and M. Diehl. “Towards Time-Optimal Race Car Driving Using Nonlinear MPC in Real-Time”. In: *53rd IEEE Conference on Decision and Control*. Los Angeles, CA, USA: IEEE, Dec. 2014, pp. 2505–2510.

ISBN: 978-1-4673-6090-6 978-1-4799-7746-8 978-1-4799-7745-1. DOI: 10.1109/CDC.2014.7039771 (cit. on p. 15).

[Wan+19] M. Wang, Z. Wang, J. Talbot, J. Christian Gerdes, and M. Schwager. "Game Theoretic Planning for Self-Driving Cars in Competitive Scenarios". In: *Robotics: Science and Systems XV*. Freiburg, Germany: Robotics: Science and Systems Foundation, June 22, 2019. ISBN: 978-0-9923747-5-4. DOI: 10.15607/RSS.2019.XV.048 (cit. on pp. 19, 20).

[Wan+20] M. Wang, N. Mehr, A. Gaidon, and M. Schwager. "Game-Theoretic Planning for Risk-Aware Interactive Agents". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA: IEEE, Oct. 24, 2020, pp. 6998–7005. ISBN: 978-1-72816-212-6. DOI: 10.1109/IROS45743.2020.9341137 (cit. on p. 19).

[Wan+21] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager. "Game-Theoretic Planning for Self-Driving Cars in Multivehicle Competitive Scenarios". In: *IEEE Transactions on Robotics* 37.4 (Aug. 2021), pp. 1313–1325. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2020.3047521 (cit. on p. 19).

[WB06] A. Wächter and L. T. Biegler. "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming". In: *Mathematical Programming* 106.1 (Mar. 1, 2006), pp. 25–57. ISSN: 1436-4646. DOI: 10.1007/s10107-004-0559-y (cit. on p. 111).

[Wei+21] B. Wei, M. Ren, W. Zeng, M. Liang, B. Yang, and R. Urtasun. "Perceive, Attend, and Drive: Learning Spatial Attention for Safe Self-Driving". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. Xi'an, China, May 2021, pp. 4875–4881. DOI: 10.1109/ICRA48506.2021.9561904 (cit. on p. 11).

[Wer+10] M. Werling, J. Ziegler, S. Kammel, and S. Thrun. "Optimal Trajectory Generation for Dynamic Street Scenarios in a Frenét Frame". In: *2010 IEEE International Conference on Robotics and Automation*. Anchorage, USA, May 2010, pp. 987–993. DOI: 10.1109/ROBOT.2010.5509799 (cit. on pp. 15, 54).

[Wer+12] M. Werling, S. Kammel, J. Ziegler, and L. Gröll. “Optimal Trajectories for Time-Critical Street Scenarios Using Discretized Terminal Manifolds”. In: *The International Journal of Robotics Research* 31.3 (Mar. 2012), pp. 346–359. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364911423042 (cit. on p. 15).

[Wer11] M. Werling. *Ein neues Konzept für die Trajektoriengenerierung und -stabilisierung in zeitkritischen Verkehrsszenarien*. Karlsruhe: KIT Scientific Publishing, Apr. 2011, p. 164. ISBN: 978-3-86644-631-1. DOI: 10.5445/KSP/1000021738 (cit. on p. 27).

[Wil+18] G. Williams, B. Goldfain, P. Drews, J. M. Rehg, and E. A. Theodorou. “Best Response Model Predictive Control for Agile Interactions Between Autonomous Ground Vehicles”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia, May 2018, pp. 2403–2410. DOI: 10.1109/ICRA.2018.8462831 (cit. on pp. 19, 20).

[WSS19] Z. Wang, R. Spica, and M. Schwager. “Game Theoretic Motion Planning for Multi-robot Racing”. In: *Distributed Autonomous Robotic Systems*. Ed. by N. Correll, M. Schwager, and M. Otte. Vol. 9. Springer Proceedings in Advanced Robotics. Cham,Switzerland: Springer International Publishing, 2019, pp. 225–238. ISBN: 978-3-030-05815-9 978-3-030-05816-6. DOI: 10.1007/978-3-030-05816-6_16 (cit. on p. 19).

[WTS20] Z. Wang, T. Taubner, and M. Schwager. “Multi-Agent Sensitivity Enhanced Iterative Best Response: A Real-Time Game Theoretic Planner for Drone Racing in 3D Environments”. In: *Robotics and Autonomous Systems* 125 (Mar. 1, 2020), p. 103410. ISSN: 0921-8890. DOI: 10.1016/j.robot.2019.103410 (cit. on p. 19).

[WVG03] W. L. Winston, M. A. Venkataramanan, and J. B. Goldberg. *Introduction to Mathematical Programming. Volume 1: Operations Research*. 4th ed. Pacific Grove, CA, USA: Thomson/Brooks/Cole, 2003. 924 pp. ISBN: 978-0-534-35964-5 (cit. on p. 58).

[YL13] J. H. Yoo and R. Langari. “Stackelberg Game Based Model of Highway Driving”. In: ASME 2012 5th Annual Dynamic Systems and Control Conference Joint with the JSME 2012 11th Motion and Vibration Conference. Florida, USA: American Society of

Mechanical Engineers Digital Collection, Sept. 17, 2013, pp. 499–508. DOI: 10.1115/DSCC2012-MOVIC2012-8703 (cit. on p. 20).

[YL14] J. H. Yoo and R. Langari. “A Stackelberg Game Theoretic Driver Model for Merging”. In: ASME 2013 Dynamic Systems and Control Conference. Palo Alto, California, USA: American Society of Mechanical Engineers Digital Collection, Mar. 6, 2014. DOI: 10.1115/DSCC2013-3882 (cit. on p. 20).

[Zen+19] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun. “End-to-End Interpretable Neural Motion Planner”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. California, USA, June 2019 (cit. on p. 10).

[Zen+20] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun. “DSDNet: Deep Structured Self-driving Network”. In: *Computer Vision - ECCV 2020*. Ed. by A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm. Lecture Notes in Computer Science. Glasgow, UK: Springer International Publishing, 2020, pp. 156–172. ISBN: 978-3-030-58589-1. DOI: 10.1007/978-3-030-58589-1_10 (cit. on p. 10).

[Zha+18] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli. “Autonomous Parking Using Optimization-Based Collision Avoidance”. In: *2018 IEEE Conference on Decision and Control (CDC)*. Miami, USA, Dec. 2018, pp. 4327–4332. DOI: 10.1109/CDC.2018.8619433 (cit. on p. 15).

[Zie+14] J. Ziegler, P. Bender, T. Dang, and C. Stiller. “Trajectory Planning for Bertha — A Local, Continuous Method”. In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. Dearborn, USA, June 2014, pp. 450–457. DOI: 10.1109/IVS.2014.6856581 (cit. on pp. 15, 16).