Using Reachable Sets for Trajectory Planning of Automated Vehicles

Stefanie Manzinger, Christian Pek, and Matthias Althoff

***Abstract*—The computational effort of trajectory planning for automated vehicles often increases with the complexity of the traffic situation. This is particularly problematic in safety-critical situations, in which the vehicle must react in a timely manner. We present a novel motion planning approach for automated vehicles, which combines set-based reachability analysis with convex optimization to address this issue. This combination makes it possible to find driving maneuvers even in small and convoluted solution spaces. In contrast to existing work, the computation time of our approach typically decreases, the more complex situations become. We demonstrate the benefits of our motion planner in scenarios from the CommonRoad benchmark suite and validate the approach on a real test vehicle.**

I. Introduction

EXISTING motion planning techniques for automated vehicles may still fail to obtain comfortable and safe motions in complex traffic scenarios with small and convoluted solution spaces (i.e., the portion of the state space enclosing feasible solutions is reduced by a large number of obstacles [1]). Yet, automated vehicles have to cope with arbitrarily complex scenarios in the real world. Why do planning techniques still struggle in complex scenarios?

The complexity mainly arises from the non-convexity of the motion planning problem: the presence of obstacles in the environment partitions the search space of the motion planning problem into different homotopy classes [1]–[3] (see Fig. 1). Homotopy classes describe “sets of trajectories that can be transformed into each other by gradual bending and stretching without colliding with obstacles” [4]. Figuratively speaking, planners have to decide when and how to pass obstacles, e.g., on the left or right side. The temporal orders of such tactical decisions can be represented by driving corridors (spatio-temporal constraints) and are a crucial element in generating collision avoidance constraints.

Given strict real-time constraints, many motion planning techniques encounter difficulties in determining suitable driving corridors for three reasons: a) scenarios in which α tactical decisions can be taken to avoid σ obstacles may already have up to α^{σ} possible driving corridors. This circumstance is particularly problematic in safety-critical situations, in which

Manuscript received Month Date, Year; revised Month Date, Year. This work was supported by the German Research Foundation (grants AL 1185/4-2 and AL 1185/3-2).

Stefanie Manzinger and Matthias Althoff are with the Department of Informatics, Technical University of Munich, D-85748 Garching, Germany (e-mail: stefanie.manzinger@in.tum.de, althoff@in.tum.de)

Christian Pek is with the Division of Robotics, Perception and Learning, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden (e-mail: pek2@kth.se); his contribution was carried out at the Department of Informatics, Technical University of Munich, D-85748 Garching, Germany.

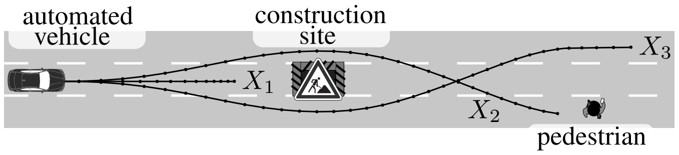


Fig. 1: Each trajectory $X_i, i \in \{1, 2, 3\}$, belongs to a different homotopy class. The automated vehicle needs to decide whether to pass the construction site (X_2 or X_3) or not (X_1). In the former case, the automated vehicle also has to decide on a passing side (left: X_2, right: X_3).

the automated vehicle must react in a timely manner to avoid a collision. b) Besides the sheer number of corridors, identifying which ones are drivable in reasonable time may often be intractable [5]. Deciding whether a driving corridor with a drivable trajectory exists can be reduced to a mixed-integer problem [6] and is therefore NP-complete [7]. c) Multiple driving corridors may exist that may not reach desired goal regions of the vehicle.

Set-based reachability analysis is particularly well suited for determining driving corridors in complex scenarios, since the number of set operations often decreases if the solution space becomes smaller [8]. The reachable set of an automated vehicle is the set of states the vehicle is able to reach over time starting from an initial set of states. Consequently, reachable sets make it possible to explore search spaces and homotopy classes efficiently in continuous space and to detect even narrow passages [9].

In this work, we show that by combining reachability analysis with an optimization-based trajectory planner, we are able to compute driving corridors and trajectories online in arbitrary scenarios. Our experiments demonstrate that our approach becomes faster with increased complexity of the motion planning problem. Below, we review existing planning techniques and present their shortcomings for the motion planning of automated vehicles in complex scenarios.

A. Literature Overview

1) Motion Planning Techniques: Various motion planning approaches are discussed in [10], [11]. We briefly review the most relevant publications for our work. Discretization-based planners, such as *rapidly exploring random trees* [12]–[14] or *state lattices* [15]–[18], connect partial motions toward a goal region in a kinematically feasible way. However, because the search space is discretized, solutions may not be found in small and convoluted solution spaces.

Continuous optimization techniques are applied to overcome discretization effects [19]. They minimize a cost function with respect to a set of state and input constraints [20]–[23]. Since many motion planning problems are non-convex, optimization problems may get stuck in local minima and may not be solvable in a computationally efficient way [24]. Convex optimization problems, however, have global convergence, for which efficient solving techniques exist [24]–[26]. Yet, convex planning problems can only be formulated for most traffic scenarios if the motion is separated into longitudinal and lateral components [27]–[30]. As a result, collision avoidance constraints can usually only be obtained if the driving corridor is already known [6].

Machine learning approaches have also been successfully applied to motion planning, e.g., [31]–[34]; however, these techniques are not yet suitable in safety-critical situations, since they are difficult to verify [35].

2) Driving Corridor Techniques: The extraction of driving corridors has been studied for some time. Naive approaches address the problem of determining driving corridors through sampling [36]–[39], combinatorial enumerations [5], [40], or support vector machines [41], [42]. However, sampling-based approaches usually struggle to detect narrow passages due to discretization effects, and enumerating all possible sequences of tactical decisions is not feasible under hard real-time constraints for complex situations. Pre-defining rules for tactical decision-making, e.g., using ontologies [43]–[45] or state-machines [46]–[49], may fail in unforeseen traffic situations. Some approaches reduce the complexity of maneuver planning by considering only a discrete set of admissible actions [50]–[53], but this diminishes the expressiveness of the planner.

Reachable sets have already been used to obtain driving corridors and to determine the non-existence of maneuvers [9], [54], [55]. The approaches in [56]–[58] combine reachability analysis with planning approaches to obtain collision-free motions. Nevertheless, they may not be applicable in arbitrary traffic situations, cannot constrain corridors to end in certain terminal states, or cannot extract high-level maneuvers with certain desired properties, such as comfort.

B. Contribution and Outline

This paper proposes a novel method for motion planning of automated vehicles by combining reachability analysis with convex optimization. Unlike previous work, our approach simultaneously

1) handles small and convoluted solution spaces. Our experiments reveal that the computational effort of our approach decreases with increased complexity of the scenario;
2) identifies collision-free driving corridors in arbitrary traffic situations using reachable sets;
3) plans optimal trajectories in candidate driving corridors with low computational effort; and
4) constrains driving corridors to end in certain terminal states, e.g., standstill in safe areas.

This paper is structured as follows: after presenting preliminaries in Sec. II, we introduce our framework in Sec. III. The computation of reachable sets is described in Sec. IV, followed by the determination of driving corridors and collision avoidance constraints in Sec. V-VI. We evaluate our concept in Sec. VII and finish with conclusions in Sec. VIII.

II. Preliminaries and Problem Statement

A. System Dynamics

We introduce different discrete-time linear systems to model the dynamics of the automated vehicle, subsequently denoted as ego vehicle. The linear models are a trade-off between the required computational time for the planning task and the accuracy compared to the real behavior. To formally verify planned trajectories despite control disturbances and model uncertainties, we refer to the approaches presented in [59], [60], which is not discussed in this work since we focus on motion planning. Additionally, in Sec. VII, we demonstrate by vehicle tests that planned trajectories are drivable despite the use of different models.

Let us introduce the subscript $\mathrm{sys} \in \{\mathrm{lon}, \mathrm{lat}, \mathcal{R}\}$ to distinguish the discrete-time linear systems for the longitudinal and lateral trajectory planning, and reachability analysis:

$$x_{\mathrm{sys},k+1} = A_{\mathrm{sys},k} x_{\mathrm{sys},k} + B_{\mathrm{sys},k} u_{\mathrm{sys},k}, \tag{1}$$

where $x_{\mathrm{sys},k} \in \mathbb{R}^{n_{\mathrm{sys},x}}$ is the state, $u_{\mathrm{sys},k} \in \mathbb{R}^{n_{\mathrm{sys},u}}$ is the input, and $k \in \mathbb{N}_0$ is the discrete time step corresponding to the time $t_k = k\Delta t$, where $\Delta t \in \mathbb{R}^+$ is the time increment. Without loss of generality, the initial time step is $k_0 = 0$ and the final time step is k_f. $A_{\mathrm{sys},k} \in \mathbb{R}^{n_{\mathrm{sys},x} \times n_{\mathrm{sys},x}}$ is the system matrix and $B_{\mathrm{sys},k} \in \mathbb{R}^{n_{\mathrm{sys},x} \times n_{\mathrm{sys},u}}$ is the input matrix. Systems in the form of (1) are subject to the convex sets of admissible states $\mathcal{X}_{\mathrm{sys},k} \subset \mathbb{R}^{n_{\mathrm{sys},x}}$ and admissible control inputs $\mathcal{U}_{\mathrm{sys},k} \subset \mathbb{R}^{n_{\mathrm{sys},u}}$, each at time step k. We use X_{sys} and U_{sys} to denote a possible state and input trajectory. Subsequently, we assume that the chosen reference point for describing the systems (1) coincides for all $\mathrm{sys} \in \{\mathrm{lon}, \mathrm{lat}, \mathcal{R}\}$.

B. Coordinate Systems and Reference Path

Let us introduce the global Cartesian coordinate frame as $F^{\mathtt{G}}$, the local curvilinear coordinate frame as $F^{\mathtt{L}}$, and the vehicle-fixed coordinate frame as $F^{\mathtt{V}}$, as shown in Fig. 2. The frame $F^{\mathtt{L}}$ is aligned with a given reference path $\Gamma : \mathbb{R} \to \mathbb{R}^2$, which is the centerline of a lane and can be provided by a route planning module, for example. In $F^{\mathtt{L}}$, a global position $(s_x, s_y)^T$ is expressed in terms of the arc length s_ζ and the orthogonal deviation s_η from $\Gamma(s_\zeta)$ (see Fig. 2). The orientation and curvature of $\Gamma(s_\zeta)$ are denoted by $\theta_\Gamma(s_\zeta)$ and $\kappa_\Gamma(s_\zeta)$, respectively. The transformation from $F^{\mathtt{L}}$ to $F^{\mathtt{G}}$ is denoted by $T^{\mathtt{G}}_{\mathtt{L}}(s_\zeta)$ (see [61]). The transformation from $F^{\mathtt{V}}$ to $F^{\mathtt{G}}$ is $T^{\mathtt{G}}_{\mathtt{V}}(\theta)$, where θ is the heading of the ego vehicle. We use a left-sided superscript $\{\mathtt{G}, \mathtt{L}, \mathtt{V}\}$ to indicate the reference coordinate system of a variable, e.g., ${}^{\mathtt{G}}v$ means velocity v of the ego vehicle in $F^{\mathtt{G}}$; for brevity, we omit the superscript if the coordinate system can be inferred from the context.

C. Reachability Analysis

As motivated in Sec. I, we use reachability analysis for identifying driving corridors. We therefore describe the dynamics of the ego vehicle with the discrete-time linear system $x_{\mathcal{R},k+1} = A_{\mathcal{R}} x_{\mathcal{R},k} + B_{\mathcal{R}} u_{\mathcal{R},k}$. Before introducing the one-step reachable set of the ego vehicle dynamics, we define the set $\mathcal{F}_k$ of forbidden states.

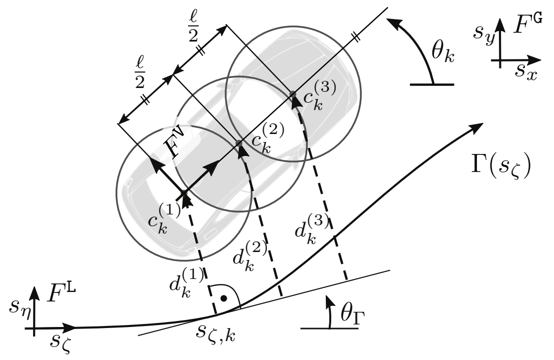


Fig. 2: The shape of the ego vehicle is approximated by three circles with center $c_k^{(i)}$, $i \in \{1,2,3\}$, to obtain collision avoidance constraints.

Definition 1 (Set of Forbidden States) *The occupied space of the ego vehicle in state $x_{\mathcal{R},k}$ is denoted by ${}^{\mathtt{G}}\mathcal{Q}(x_{\mathcal{R},k}) \subset \mathbb{R}^2$ and the set of occupied positions of all obstacles (e.g., other cars and pedestrians), as well as the space outside the road by ${}^{\mathtt{G}}\mathcal{O}_k$. The set of forbidden states at time step k is*

$$\mathcal{F}_k = \left\{ x_{\mathcal{R},k} \in \mathcal{X}_{\mathcal{R},k} \mid {}^{\mathtt{G}}\mathcal{Q}(x_{\mathcal{R},k}) \cap {}^{\mathtt{G}}\mathcal{O}_k \neq \emptyset \right\}.$$

Definition 2 (One-step Reachable Set) *The initial reachable set is $\mathcal{R}^{\mathtt{e}}_{k_0} = \mathcal{X}_{\mathcal{R},k_0}$, where $\mathcal{X}_{\mathcal{R},k_0}$ is the set of initial states of the ego vehicle including measurement uncertainties. The one-step reachable set $\mathcal{R}^{\mathtt{e}}_{k+1}$ is then defined as the set of all states that can be reached from an initial set of states $\mathcal{R}^{\mathtt{e}}_{k} \subseteq \mathcal{X}_{\mathcal{R},k}$ within one time step without intersecting $\mathcal{F}_{k+1}$:*

$$\mathcal{R}^{\mathtt{e}}_{k+1} = \Big\{ x_{\mathcal{R},k+1} \in \mathcal{X}_{\mathcal{R},k+1} \,\Big|\, \exists x_{\mathcal{R},k} \in \mathcal{R}^{\mathtt{e}}_k,\, \exists u_{\mathcal{R},k} \in \mathcal{U}_{\mathcal{R},k}: x_{\mathcal{R},k+1} = A_{\mathcal{R}} x_{\mathcal{R},k} + B_{\mathcal{R}} u_{\mathcal{R},k} \wedge x_{\mathcal{R},k+1} \notin \mathcal{F}_{k+1} \Big\}.$$

Before defining the drivable area representing the collision-free, reachable positions of the ego vehicle, let us introduce the projection operator $\mathrm{proj}_{\Diamond}(x)$.

Definition 3 (Projection) *The operator $\mathrm{proj}_{\Diamond}(x)$ maps the state $x \in \mathcal{X}$ to its elements $\Diamond$, e.g., $\mathrm{proj}_{(m_1,m_3)}(x) = (m_1, m_3)^T$ for $x = (m_1, m_2, m_3)^T$. Using the same notation, we project a set of states $\mathcal{X}$: $\mathrm{proj}_{\Diamond}(\mathcal{X}) = \{\mathrm{proj}_{\Diamond}(x) \mid x \in \mathcal{X}\}$.*

Definition 4 (Drivable Area) *We obtain the drivable area $\mathcal{D}^{\mathtt{e}}_k$ of the ego vehicle as the projection of its reachable set $\mathcal{R}^{\mathtt{e}}_k$ onto the position domain: $\mathcal{D}^{\mathtt{e}}_k = \mathrm{proj}_{(s_\zeta, s_\eta)}(\mathcal{R}^{\mathtt{e}}_k)$.*

The superscript $\mathtt{e}$ of $\mathcal{R}^{\mathtt{e}}_k$ and $\mathcal{D}^{\mathtt{e}}_k$ denotes the exact reachable set and drivable area, respectively. However, it is generally computationally intractable to compute the exact reachable set of a system [62]. For this reason, we aim to compute accurate approximations $\mathcal{R}_k \approx \mathcal{R}^{\mathtt{e}}_k$ and $\mathcal{D}_k \approx \mathcal{D}^{\mathtt{e}}_k$ (see Sec. IV).

D. Convex Trajectory Planning

For trajectory planning, we describe the kinematics of the ego vehicle along $\Gamma(s_\zeta)$ using the rear axis center as the reference point, e.g., as shown in [29], [30]. In principle, the reference point can be chosen differently; however, our choice has the advantage that the slip angle can be assumed to be zero at low speeds and generally neglected at high speeds [29]. To formulate our trajectory planning problems as convex optimization problems—which can be efficiently solved with global convergence [24]—we separate the longitudinal and lateral motion of the ego vehicle.

Following [29], we assume that the orientation θ of the ego vehicle is close to the reference path's orientation $\theta_\Gamma(s_\zeta)$, consequently, the trigonometric functions can be approximated as $\sin(\Delta) \approx \Delta$ and $\cos(\Delta) \approx 1$, and $s_\eta \kappa_\Gamma(s_\zeta) \ll 1$. In this way, we linearize the kinematics of the ego vehicle as in [29]:

$$\dot{s}_\zeta = v\,\frac{\cos(\theta - \theta_\Gamma(s_\zeta))}{1 - s_\eta \kappa_\Gamma(s_\zeta)} \approx v, \tag{2}$$

$$\dot{s}_\eta = v \sin(\theta - \theta_\Gamma(s_\zeta)) \approx v(\theta - \theta_\Gamma(s_\zeta)). \tag{3}$$

Using (2) and (3), we can derive discrete-time linear systems of structure (1), which is detailed in [29], [30] and Sec. VII-A.

To model collision avoidance for the full-dimensional vehicle, we over-approximate its shape by three circles with equal radius r similar to [29], [30], [63]. The centers $c_k^{(i)}$, $i \in \{1,2,3\}$, of the circles are selected such that $c_k^{(1)}$ and $c_k^{(3)}$ coincide with the rear and front axis centers, respectively (see Fig. 2). The distance between the centers of adjacent circles is $\ell/2$, where ℓ is the wheelbase. With this, we formally define the longitudinal and lateral trajectory planning problems.

Definition 5 (Longitudinal Trajectory Planning Problem) *The longitudinal trajectory planning problem is to minimize the convex cost function $J_{\mathrm{lon}}(x_{\mathrm{lon}}, u_{\mathrm{lon}})$ for all $k \in \{k_0, \dots, k_f\}$:*

$$\min_{u_{\mathrm{lon}}} \; J_{\mathrm{lon}}(x_{\mathrm{lon}}, u_{\mathrm{lon}}) \tag{4a}$$

$$\text{s.t.} \quad x_{\mathrm{lon},k+1} = A_{\mathrm{lon},k}\, x_{\mathrm{lon},k} + B_{\mathrm{lon},k}\, u_{\mathrm{lon},k}, \tag{4b}$$

$$u_{\mathrm{lon},k} \in \mathcal{U}_{\mathrm{lon},k}, \quad x_{\mathrm{lon},k} \in \mathcal{X}_{\mathrm{lon},k}, \tag{4c}$$

$$\underline{s}_{\zeta,k+1} \leq \; \mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k+1}) \; \leq \overline{s}_{\zeta,k+1}, \tag{4d}$$

where (4b) *describes the longitudinal dynamics of the ego vehicle subject to the convex constraints* (4c)-(4d)*. The linear constraint* (4d) *models collision avoidance, where $s_{\zeta,k} = \mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$ is the longitudinal position of ${}^{\mathtt{L}}c_k^{(1)}$ (see Fig. 2). The minimum and maximum bounds on $s_{\zeta,k}$ are denoted by $\underline{s}_{\zeta,k}$ and $\overline{s}_{\zeta,k}$, respectively.*

Definition 6 (Lateral Trajectory Planning Problem) *Given that the longitudinal trajectory X_{lon} is obtained a priori by solving* (4)*, the lateral trajectory planning problem is to minimize the convex cost function $J_{\mathrm{lat}}(x_{\mathrm{lat}}, u_{\mathrm{lat}})$ for all $k \in \{k_0, \dots, k_f\}$:*

$$\min_{u_{\mathrm{lat}}} \; J_{\mathrm{lat}}(x_{\mathrm{lat}}, u_{\mathrm{lat}}) \tag{5a}$$

$$\text{s.t.} \quad x_{\mathrm{lat},k+1} = A_{\mathrm{lat},k}\, x_{\mathrm{lat},k} + B_{\mathrm{lat},k}\, u_{\mathrm{lat},k}, \tag{5b}$$

$$u_{\mathrm{lat},k} \in \mathcal{U}_{\mathrm{lat},k}, \quad x_{\mathrm{lat},k} \in \mathcal{X}_{\mathrm{lat},k}, \tag{5c}$$

$$\forall i \in \{1,2,3\}: \; \underline{d}^{(i)}_{k+1} \leq d^{(i)}_{k+1}(x_{\mathrm{lat},k+1}, x_{\mathrm{lon},k+1}) \leq \overline{d}^{(i)}_{k+1}, \tag{5d}$$

where (5b) *describes the lateral dynamics of the ego vehicle subject to the convex constraints* (5c)-(5d)*, and $x_{\mathrm{lon},k} \in X_{\mathrm{lon}}$. To model lateral collision avoidance, we introduce the lateral distance $d_k^{(i)}(x_{\mathrm{lat},k}, x_{\mathrm{lon},k})$ of the i-th circle's center to $\Gamma(s_{\zeta,k})$ at time step k (see Fig. 2) and restrict the minimum and maximum deviations $\underline{d}_k^{(i)}$ and $\overline{d}_k^{(i)}$ from $\Gamma(s_{\zeta,k})$, respectively.*

E. Problem Statement

The drivable area $\mathcal{D}_k$ is used to explore the collision-free solution space for trajectory planning. However, the drivable area is generally non-convex and may be disconnected due to the presence of obstacles (see Fig. 3a), which renders the direct usage of the drivable area unsuitable to obtain bounds for the convex collision avoidance constraints (4d) and (5d). To overcome this issue, we determine driving corridors that are subsets of the drivable area, particularly, a driving corridor for the longitudinal and lateral trajectory planning problems referred to as longitudinal and lateral driving corridor, respectively. To this end, we exploit that the projection of a connected set $\mathcal{C} \subset \mathbb{R}^2$ onto either the longitudinal or lateral position domain yields an interval from which we can obtain the bounds in (4d) and (5d).

Definition 7 (Connected Set [64]) *A set $\mathcal{C} \subset \mathbb{R}^2$ is connected if there do not exist open sets $\mathcal{W}_1, \mathcal{W}_2 \subseteq \mathbb{R}^2$ such that $\mathcal{W}_1 \cup \mathcal{W}_2$ contains $\mathcal{C}$ and with $\mathcal{C} \cap \mathcal{W}_1$ and $\mathcal{C} \cap \mathcal{W}_2$ disjoint and non-empty.*

A longitudinal driving corridor is composed of connected sets (see Fig. 3b).

Definition 8 (Longitudinal Driving Corridor) *A longitudinal driving corridor C_{lon} is a sequence of connected sets $\mathcal{C}_k \subseteq \mathcal{D}_k$ over time steps $k \in \{k_0, \dots, k_f\}$. We refer to the longitudinal driving corridor at time step k as $C_{\text{lon},k}$.*

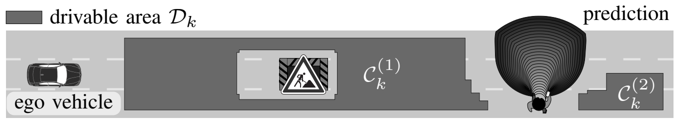
A lateral driving corridor is also composed of connected sets. However, a connected set may have holes, e.g., $\mathcal{C}_k^{(1)}$ in Fig. 3b; thus, the passing sides for obstacles may be ambiguous. Therefore, we demand that lateral driving corridors must provide a single passing side for each obstacle at the longitudinal positions $\text{proj}_{s_\zeta}(x_{\text{lon},k})$ of the longitudinal trajectory X_{lon} (see Fig. 3d).

Definition 9 (Lateral Driving Corridor) *A lateral driving corridor C_{lat} is a sequence of connected sets $\mathcal{C}_k \subseteq \mathcal{D}_k$ over time steps $k \in \{k_0, \dots, k_f\}$. For each $\mathcal{C}_k$, it must hold that $\{s_{\eta,k} \,|\, (s_{\zeta,k}, s_{\eta,k})^T \in \mathcal{C}_k, s_{\zeta,k} = \text{proj}_{s_\zeta}(x_{\text{lon},k}), x_{\text{lon},k} \in X_{\text{lon}}\}$ is a non-empty interval (unique passing side at $\text{proj}_{s_\zeta}(x_{\text{lon},k})$, see Fig. 3d). We refer to the lateral driving corridor at time step k as $C_{\text{lat},k}$.*

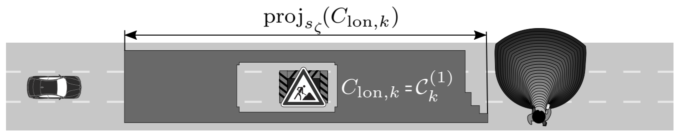
The goal of this work is to present an efficient algorithm to extract longitudinal and lateral driving corridors according to Def. 8 and 9 from the drivable area. Furthermore, we elaborate on the extraction of collision avoidance constraints for the full-dimensional vehicle using the proposed driving corridors.

III. Framework

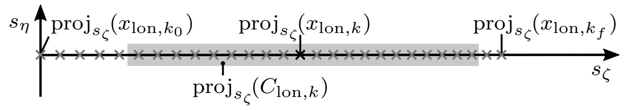
Let us introduce our motion planning approach summarized in Alg. 1 using the scenario in Fig. 3, in which the ego vehicle approaches a construction site and a pedestrian. Our motion planning approach is integrated between the behavioral layer and the control layer of the ego vehicle, e.g., see planning framework in [11]. We assume that our method receives as input the environment model including the admissible lanes leading to the specified target, a desired reference path through



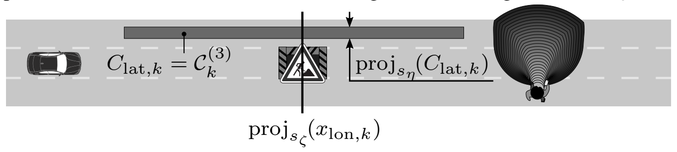
(a) Computation of the drivable area for consecutive time steps k. We depict the drivable area $\mathcal{D}_k$ at time step k.


$\text{proj}_{s_\zeta}(C_{\text{lon},k})$
$C_{\text{lon},k} = \mathcal{C}_k^{(1)}$

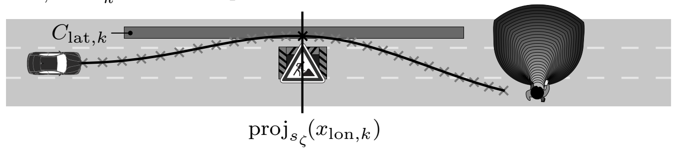
(b) Identification of the longitudinal driving corridors C_{lon} as a time series of connected sets. We depict the longitudinal driving corridor $C_{\text{lon},k} = \mathcal{C}_k^{(1)}$ at time step k.


s_η
$\text{proj}_{s_\zeta}(x_{\text{lon},k_0})$
$\text{proj}_{s_\zeta}(x_{\text{lon},k})$
$\text{proj}_{s_\zeta}(x_{\text{lon},k_f})$
s_ζ
$\text{proj}_{s_\zeta}(C_{\text{lon},k})$

(c) Optimization of the longitudinal trajectory X_{lon} subject to the longitudinal position constraints obtained from the longitudinal driving corridor C_{lon}.


$C_{\text{lat},k} = \mathcal{C}_k^{(3)}$
$\text{proj}_{s_\eta}(C_{\text{lat},k})$
$\text{proj}_{s_\zeta}(x_{\text{lon},k})$

(d) Identification of the lateral driving corridors C_{lat} as a time series of connected sets with unique passing side. We depict the lateral driving corridor $C_{\text{lat},k} = \mathcal{C}_k^{(3)}$ at time step k.



(e) Optimization of the lateral trajectory X_{lat} subject to the lateral position constraints obtained from the lateral driving corridor C_{lat}.

Fig. 3: Motion planning with reachable sets.

the road network, and all safety-relevant traffic participants and their predicted future behavior. Our algorithm outputs a reference trajectory for the ego vehicle that is passed to a vehicle controller.

As a first step, we compute the one-step reachable sets of the ego vehicle for consecutive time steps (see Alg. 1, line 2) and store them in a data structure $\mathcal{G}_{\mathcal{R}}$ (see Sec. IV). Since our trajectory planner requires position constraints to determine drivable trajectories, we project the reachable set onto the position domain to obtain the drivable area (see Fig. 3a).

We continue with identifying possible longitudinal driving corridors within the drivable area according to Def. 8 (see Alg. 1, line 3). The drivable area $\mathcal{D}_k$ can be disconnected, e.g., $\mathcal{C}_k^{(1)}$ is not connected with $\mathcal{C}_k^{(2)}$ in Fig. 3a, since its computation requires the removal of colliding states (see Def. 2 and 4). Because of this, there may be multiple longitudinal driving corridors (stored in a data structure $\mathcal{G}_{\mathcal{C},\text{lon}}$, see Sec. V). We select a suitable one for trajectory optimization by ranking the longitudinal driving corridors according to a user-defined criterion (see Alg. 1, line 4). The highest-ranked driving corridor C_{lon}, e.g., $C_{\text{lon},k}$ in Fig. 3b, is selected for planning the longitudinal trajectory X_{lon} as proposed in Sec. II-D (see Alg. 1, line 5). If the optimization problem is infeasible, we select the next available longitudinal driving corridor. The

Algorithm 1 ConvexTrajectoryPlanning

```
Input: set of initial states $\mathcal{X}_{\mathcal{R},k_0}$, set of obstacles $\mathcal{O}_k$ for $k_0$ to $k_f$
Output: longitudinal and lateral trajectories $X_{\mathrm{lon}}$ and $X_{\mathrm{lat}}$
 1: $X_{\mathrm{lon}},\ X_{\mathrm{lat}} \leftarrow \emptyset$
 2: $\mathcal{G}_{\mathcal{R}} \leftarrow$ computeReachableSet($\mathcal{X}_{\mathcal{R},k_0}, \mathcal{O}_k$)
 3: $\mathcal{G}_{\mathcal{C},\mathrm{lon}} \leftarrow$ IdentifyCorridors($\mathcal{G}_{\mathcal{R}}$)
 4: for all $C_{\mathrm{lon}}$ in $\mathcal{G}_{\mathcal{C},\mathrm{lon}}$.assessCorridor( ) do
 5:     $X_{\mathrm{lon}} \leftarrow$ LongitudinalOptimization($C_{\mathrm{lon}}$)
 6:     if $X_{\mathrm{lon}} \neq \emptyset$ then
 7:         $\mathcal{G}_{\mathcal{C},\mathrm{lat}} \leftarrow$ IdentifyCorridors($\mathcal{G}_{\mathcal{R}}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
 8:         for all $C_{\mathrm{lat}}$ in $\mathcal{G}_{\mathcal{C},\mathrm{lat}}$.assessCorridor( ) do
 9:             $X_{\mathrm{lat}} \leftarrow$ LateralOptimization($C_{\mathrm{lat}}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
10:             if $X_{\mathrm{lat}} \neq \emptyset$ then
11:                 return $X_{\mathrm{lon}}, X_{\mathrm{lat}}$
12:             end if
13:         end for
14:     end if
15: end for
```

longitudinal trajectory X_{lon} (see Fig. 3c) is used to identify lateral driving corridors (see Alg. 1, line 7).

Analogous to longitudinal planning, multiple lateral driving corridors may exist (stored in a data structure $\mathcal{G}_{\mathcal{C},\mathrm{lat}}$, see Sec. V). We also rank them according to a user-defined criterion (see Alg. 1, line 8). The highest-ranked lateral driving corridor C_{lat}, e.g., $C_{\mathrm{lat},k}$ in Fig. 3d, is used to optimize the lateral trajectory X_{lat} as proposed in Sec. II-D (Alg. 1, line 9). If the optimization problem is infeasible, we use the next available driving corridor C_{lat} to obtain X_{lat}. If the optimization is successful, the final trajectory is returned (see Fig. 3e and Alg. 1, line 11). Trajectories can be planned for each available driving corridor as long as time permits. Thus, it is possible to obtain several maneuver options to choose from. In the rare event that no drivable trajectories are found, we always keep a fail-safe trajectory available [30], [65].

IV. Reachable Set Computation

After presenting the vehicle model for reachability analysis in Sec. IV-A, we introduce the representation of reachable sets (see Sec. IV-B) and our proposed algorithm (see Sec. IV-C) inspired by [9].

A. Vehicle Model for Reachability Analysis

We deliberately select a simple but realistic dynamic model for the reachable set computation to reduce computational complexity. The dynamics of the ego vehicle are described by two second-order integrator models in the curvilinear coordinate frame $F^{\mathtt{L}}$ with ${}^{\mathtt{L}}c_k^{(1)}$ as the reference point (see Fig. 2). The state $x_{\mathcal{R}} = (s_\zeta, v_\zeta, s_\eta, v_\eta)^T$ is composed of the position s and velocity v in the longitudinal ζ-direction and the lateral η-direction. The input $u_{\mathcal{R}} = (a_\zeta, a_\eta)^T$ is given by the acceleration a in ζ- and η-directions. Adding bounds on the accelerations and velocities, the dynamics are

$$x_{\mathcal{R},k+1} = \begin{pmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{pmatrix} x_{\mathcal{R},k} + \begin{pmatrix} \frac{1}{2}\Delta t^2 & 0 \\ \Delta t & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ 0 & \Delta t \end{pmatrix} u_{\mathcal{R},k}, \quad (6a)$$

$$\underline{v}_\zeta \leq v_{\zeta,k} \leq \overline{v}_\zeta, \quad \underline{v}_\eta \leq v_{\eta,k} \leq \overline{v}_\eta, \quad (6b)$$

$$\underline{a}_\zeta \leq a_{\zeta,k} \leq \overline{a}_\zeta, \quad \underline{a}_\eta \leq a_{\eta,k} \leq \overline{a}_\eta. \quad (6c)$$

The formulation of the dynamics in $F^{\mathtt{L}}$ makes it possible to consider speed limits and to avoid driving backwards. This entails that model (6a) deviates increasingly from a real vehicle for larger curvatures $\kappa_\Gamma(s_\zeta)$, e.g., the ego vehicle would be able to make a turn with an arbitrarily high velocity. However, we compensate for this by setting appropriate constraints (6b)-(6c), e.g., we use a conservative parametrization of the admissible accelerations and velocities.

B. Reachable Set Representation

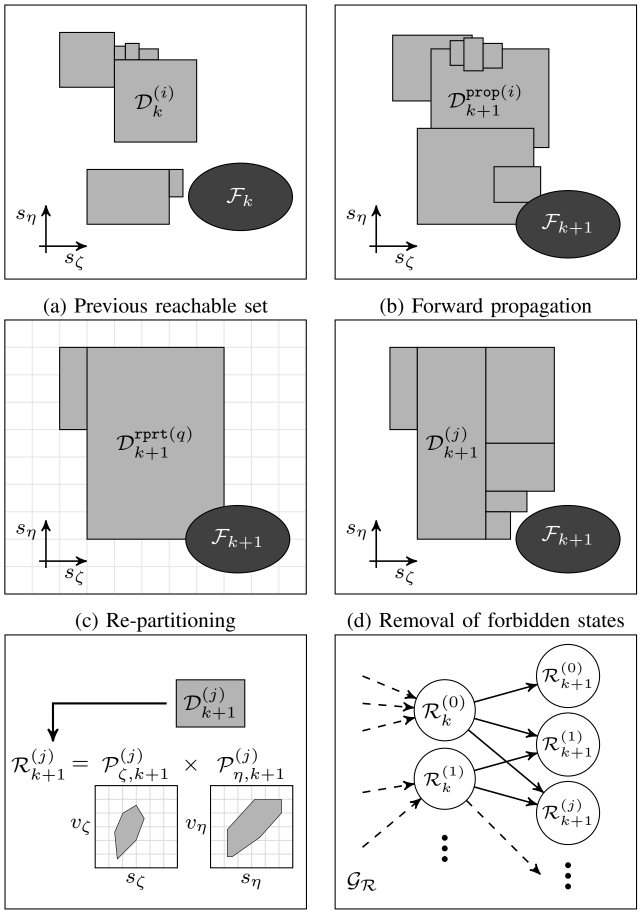
To improve computational efficiency, the reachable set (see Def. 2) of (6) is approximated by the union $\mathcal{R}_k$ of base sets $\mathcal{R}_k^{(i)} = \mathcal{P}_{\zeta,k}^{(i)} \times \mathcal{P}_{\eta,k}^{(i)}$, $i \in \mathbb{N}_0$, which are the Cartesian products of two convex polytopes $\mathcal{P}_{\zeta,k}^{(i)}$ and $\mathcal{P}_{\eta,k}^{(i)}$ in the (s_ζ, v_ζ) and (s_η, v_η) plane (see Fig. 4e), respectively [9]. Hence, each base set represents pairs of reachable positions and velocities. We select polytopes for the representation of the reachable set, because the required set operations can be efficiently performed on polytopes. The projection of sets $\mathcal{R}_k^{(i)}$ onto the position domain yields axis-aligned rectangles $\mathcal{D}_k^{(i)}$ (see Fig. 4a) whose union represents the drivable area $\mathcal{D}_k$ (see Def. 4).

C. Algorithm

The initial base set $\mathcal{R}_{k_0}^{(0)}$ encloses the set of initial states $\mathcal{X}_{\mathcal{R},k_0}$ of the ego vehicle including measurement uncertainties, such that $\mathcal{X}_{\mathcal{R},k_0} \subseteq \mathcal{R}_{k_0}^{(0)}$. The reachable set $\mathcal{R}_k$ of consecutive time steps is computed as explained below and illustrated in Fig. 4. To simplify the notation, we also denote the collection of base sets $\mathcal{R}_k^{(i)}$ with $\mathcal{R}_k$ (not only the union of base sets), i.e., $\mathcal{R}_k = \{\mathcal{R}_k^{(0)}, \mathcal{R}_k^{(1)}, \ldots\}$. Similarly, we simplify the notation for the drivable area $\mathcal{D}_k$.

1) Forward Propagation: The base sets $\mathcal{R}_k^{(i)} \in \mathcal{R}_k$ of time step k (see Fig. 4a) are propagated forward in time according to the system model (6) considering all admissible inputs. The forward propagation is computed similarly to [9] and a detailed description can be found in the Appendix A. We denote the propagated reachable set with $\mathcal{R}_{k+1}^{\mathtt{prop}}$, and refer to the propagated drivable area as $\mathcal{D}_{k+1}^{\mathtt{prop}}$, see Fig. 4b.

2) Re-Partitioning: With the aim to reduce the number of axis-aligned rectangles, the propagated drivable area $\mathcal{D}_{k+1}^{\mathtt{prop}}$ is merged and re-partitioned into a set $\mathcal{D}_{k+1}^{\mathtt{rprt}}$ of new axis-aligned rectangles with disjoint interiors using sweep line algorithms [66] (see Fig 4c). To avoid generating a large number of new axis-aligned rectangles with only slightly displaced edges, we enlarge each propagated set $\mathcal{D}_{k+1}^{\mathtt{prop}(i)}$ to a grid prior to merging and re-partitioning as presented in [9] (see Fig 4c).


$\mathcal{D}_k^{(i)}$
$\mathcal{F}_k$
s_η
s_ζ
$\mathcal{D}_{k+1}^{\mathtt{prop}(i)}$
$\mathcal{F}_{k+1}$
$\mathcal{D}_{k+1}^{\mathtt{rprt}(q)}$
$\mathcal{D}_{k+1}^{(j)}$
$\mathcal{R}_{k+1}^{(j)} = \mathcal{P}_{\zeta,k+1}^{(j)} \times \mathcal{P}_{\eta,k+1}^{(j)}$
v_ζ
v_η
$\mathcal{R}_k^{(0)}$
$\mathcal{R}_k^{(1)}$
$\mathcal{R}_{k+1}^{(0)}$
$\mathcal{R}_{k+1}^{(1)}$
$\mathcal{R}_{k+1}^{(j)}$
$\mathcal{G}_\mathcal{R}$

(a) Previous reachable set (b) Forward propagation

(c) Re-partitioning (d) Removal of forbidden states

(e) Creation of base sets (f) Update of reachability graph

Fig. 4: Steps in the reachable set computation.

3) Removal of Forbidden States: The forbidden positions are removed from $\mathcal{D}_{k+1}^{\mathtt{rprt}}$ to obtain only collision-free configurations of the ego vehicle as shown in Fig. 4d. Since $\mathcal{D}_{k+1}^{\mathtt{rprt}} \setminus \mathrm{proj}_{(s_\zeta,s_\eta)}(\mathcal{F}_{k+1})$ cannot be represented by axis-aligned rectangles in general, we under-approximate the result by $\mathcal{D}_{k+1}$. The under-approximation ensures that the remaining drivable area $\mathcal{D}_{k+1}$ is collision-free. A detailed description of this computation step is provided in Sec. IV-D.

4) Creation of Base Sets: The reachable set $\mathcal{R}_{k+1}$ is obtained by determining the reachable velocities for the collision-free drivable area $\mathcal{D}_{k+1}$ (see Fig. 4e). From the propagated base sets $\mathcal{R}_{k+1}^{\mathtt{prop}(i)}$, we can determine the velocities for which the ego vehicle reaches a set of positions $\mathcal{D}_{k+1}^{(j)}$. After introducing the set $\mathrm{overlap}(\mathcal{D}_{k+1}^{(j)}) = \{i \in \mathbb{N}_0 \,|\, \exists \mathcal{R}_{k+1}^{\mathtt{prop}(i)} \in \mathcal{R}_{k+1}^{\mathtt{prop}} : \mathcal{D}_{k+1}^{(j)} \cap \mathrm{proj}_{(s_\zeta,s_\eta)}(\mathcal{R}_{k+1}^{\mathtt{prop}(i)}) \neq \emptyset\}$, the new polytopes $\mathcal{P}_{\zeta,k+1}^{(j)}$ are

$$\mathcal{P}_{\zeta,k+1}^{(j)} = \mathrm{convexhull}\left(\bigcup_{i \in \mathrm{overlap}(\mathcal{D}_{k+1}^{(j)})} \hat{\mathcal{P}}_{\zeta,k+1}^{(i)} \right),$$

with

$$\hat{\mathcal{P}}_{\zeta,k+1}^{(i)} = \mathcal{P}_{\zeta,k+1}^{\mathtt{prop}(i)} \cap \left\{ \begin{pmatrix} s_\zeta \\ v_\zeta \end{pmatrix} \in \mathbb{R}^2 \,\middle|\, s_\zeta \geq \inf\left(\mathrm{proj}_{s_\zeta}(\mathcal{D}_{k+1}^{(j)})\right), \right.$$
$$\left. s_\zeta \leq \sup\left(\mathrm{proj}_{s_\zeta}(\mathcal{D}_{k+1}^{(j)})\right) \right\}.$$

$\mathcal{P}_{\eta,k+1}^{(j)}$ is determined similarly. Note that we compute the convex hull to ensure that we obtain convex polytopes.

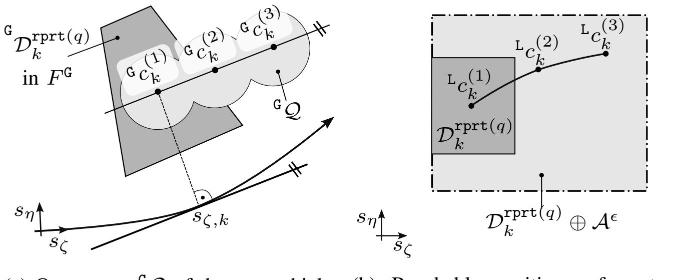
5) Update of Reachability Graph: The base sets $\mathcal{R}_{k+1}^{(j)}$ are stored in a directed graph $\mathcal{G}_\mathcal{R}$ to trace back the temporal and spatial sequence of reachable states (see Fig. 4f). In $\mathcal{G}_\mathcal{R}$, each base set $\mathcal{R}_{k+1}^{(j)}$ is assigned to exactly one node and an edge indicates that $\mathcal{R}_{k+1}^{(j)}$ is reachable from $\mathcal{R}_k^{(i)}$ for consecutive time steps (this also holds for $\mathcal{D}_k^{(i)}$ and $\mathcal{D}_{k+1}^{(j)}$, since $\mathcal{D}_k^{(i)} = \mathrm{proj}_{(s_\zeta,s_\eta)}(\mathcal{R}_k^{(i)})$). Note that a set $\mathcal{R}_k^{(i)}$ may reach several sets $\mathcal{R}_{k+1}^{(j)}$ in the next point in time.

D. Removal of Forbidden States

To remove the forbidden states (see Fig. 4d), we need to consider the occupancy ${}^{\mathtt{G}}\mathcal{Q}(x_{\mathcal{R},k})$ of the ego vehicle that is over-approximated by three circles with radius r (see Sec. II). Let us therefore introduce the Minkowski sum of two sets $\mathcal{W}_1$ and $\mathcal{W}_2$ as $\mathcal{W}_1 \oplus \mathcal{W}_2 = \{w_1 + w_2 \,|\, w_1 \in \mathcal{W}_1, w_2 \in \mathcal{W}_2\}$. By dilating the occupancies ${}^{\mathtt{G}}\mathcal{O}_k$ with a circle with radius r using the Minkowski sum, it is sufficient to test only ${}^{\mathtt{G}}c_k^{(i)}$, $i \in \{1,2,3\}$, for collisions [63]. To determine the center ${}^{\mathtt{G}}c_k^{(i)}$ of the i-th circle for ${}^{\mathtt{L}}c_k^{(1)} = (s_{\zeta,k}, s_{\eta,k})^T$, we assume that the heading θ_k of the ego vehicle is aligned with $\theta_\Gamma(s_{\zeta,k})$:

$${}^{\mathtt{G}}c_k^{(i)} = T_{\mathtt{L}}^{\mathtt{G}}(s_{\zeta,k}) \begin{pmatrix} s_{\zeta,k} \\ s_{\eta,k} \end{pmatrix} + T_{\mathtt{V}}^{\mathtt{G}}(\theta_\Gamma(s_{\zeta,k})) \begin{pmatrix} \frac{i-1}{2}\ell \\ 0 \end{pmatrix}. \quad (7)$$

Computing (7) for all $(s_{\zeta,k}, s_{\eta,k})^T \in \mathcal{D}_k^{\mathtt{rprt}}$ to determine the collision-free drivable area is intractable, since the transformation matrix $T_{\mathtt{V}}^{\mathtt{G}}(\theta_\Gamma(s_{\zeta,k}))$ continuously varies along the reference path $\Gamma(s_\zeta)$ with changing $s_{\zeta,k}$ (see Fig. 5a). To reduce computational complexity, we strictly over-approximate the reachable positions of centers ${}^{\mathtt{L}}c_k^{(i)}$ by enlarging each $\mathcal{D}_k^{\mathtt{rprt}(q)} \in \mathcal{D}_k^{\mathtt{rprt}}$ with a rectangular shape $\mathcal{A}^\epsilon$ (see Fig. 5b): $\mathcal{D}_k^{\mathtt{rprt}(q)} \oplus \mathcal{A}^\epsilon$. The size of $\mathcal{A}^\epsilon$ depends on the local curvature κ_Γ of the reference path and $\mathcal{D}_k^{\mathtt{rprt}(q)}$; however, we omit the


${}^{\mathtt{G}}\mathcal{D}_k^{\mathtt{rprt}(q)}$ in $F^{\mathtt{G}}$
${}^{\mathtt{G}}c_k^{(1)}$
${}^{\mathtt{G}}c_k^{(2)}$
${}^{\mathtt{G}}c_k^{(3)}$
${}^{\mathtt{G}}\mathcal{Q}$
s_η
s_ζ
$s_{\zeta,k}$
${}^{\mathtt{L}}c_k^{(1)}$
${}^{\mathtt{L}}c_k^{(2)}$
${}^{\mathtt{L}}c_k^{(3)}$
$\mathcal{D}_k^{\mathtt{rprt}(q)}$
$\mathcal{D}_k^{\mathtt{rprt}(q)} \oplus \mathcal{A}^\epsilon$

(a) Occupancy ${}^{\mathtt{G}}\mathcal{Q}$ of the ego vehicle for an arbitrary position ${}^{\mathtt{G}}c_k^{(1)}$ within ${}^{\mathtt{G}}\mathcal{D}_k^{\mathtt{rprt}(q)}$ given that $\theta_k = \theta_\Gamma(s_{\zeta,k})$.

(b) Reachable positions of centers ${}^{\mathtt{L}}c_k^{(i)}$, $i \in \{1,2,3\}$, are enclosed by enlarging $\mathcal{D}_k^{\mathtt{rprt}(q)}$ with $\mathcal{A}^\epsilon$ in $F^{\mathtt{L}}$.

Fig. 5: Consideration of the shape of the ego vehicle in the reachable set computation.

dependency in the notation for brevity. A detailed explanation on the computation of $\mathcal{A}^{\epsilon}$ is given in the Appendix B. Using $\mathcal{A}^{\epsilon}$, the removal of forbidden states from $\mathcal{D}_k^{\mathtt{rprt}}$ can be either performed in the global coordinate frame $F^{\mathtt{G}}$ or in the curvilinear frame $F^{\mathtt{L}}$. As a result, we under-approximate $\mathcal{D}_k^{\mathtt{rprt}} \setminus \mathrm{proj}_{(s_\zeta, s_\eta)}(\mathcal{F}_k)$ with a new set of axis-aligned rectangles $\mathcal{D}_k$, such that for all $\mathcal{D}_k^{(j)} \in \mathcal{D}_k$: $\mathcal{D}_k^{(j)} \oplus \mathcal{A}^{\epsilon}$ is collision-free.

V. Identification of Driving Corridors

After introducing necessary preliminaries, we elaborate on the identification of longitudinal and lateral driving corridors.

A. Preliminaries

As stated in Def. 8-9, a driving corridor is a temporal sequence of connected sets in the position domain. We denote the n-th connected set at time step k by $\mathcal{C}_k^{(n)} = \{\mathcal{D}_k^{(i)}, \mathcal{D}_k^{(j)}, \ldots\}$, $n \in \mathbb{N}$ (see Fig. 3a). To determine longitudinal and lateral driving corridors, we create corresponding graphs $\mathcal{G}_{\mathcal{C}}$, in which connected sets $\mathcal{C}_k^{(n)}$ are stored as a node denoted by $\mathtt{n}(\mathcal{C}_k^{(n)})$. An edge between nodes $\mathtt{n}(\mathcal{C}_k^{(n)})$ and $\mathtt{n}(\mathcal{C}_{k+1}^{(m)})$ in $\mathcal{G}_{\mathcal{C}}$ indicates that at least one set $\mathcal{D}_k^{(i)} \in \mathcal{C}_k^{(n)}$ reaches a set $\mathcal{D}_{k+1}^{(j)} \in \mathcal{C}_{k+1}^{(m)}$ within one time step so that the connected set $\mathcal{C}_k^{(n)}$ reaches $\mathcal{C}_{k+1}^{(m)}$. A path in $\mathcal{G}_{\mathcal{C}}$ from k_0 to k_f is a possible driving corridor.

B. Longitudinal Driving Corridors

Alg. 2 identifies longitudinal driving corridors backwards in time using the reachability graph $\mathcal{G}_{\mathcal{R}}$. Let us first explain the procedure, afterwards we apply it to the example in Fig. 6: we obtain the drivable area $\mathcal{D}_{k_f}$ of the final time step k_f (see Alg. 2, line 2) through the reachability graph $\mathcal{G}_{\mathcal{R}}$. As an option, we can exclude states of the reachable set beforehand to constrain driving corridors to end in a predefined set of terminal states (e.g., a standstill in safe areas). Then, we identify connected sets $\mathcal{C}_{k_f}^{(n)}$ within the drivable area $\mathcal{D}_{k_f}$ using a sweep line algorithm [67] (see Alg. 2, line 6). For each $\mathcal{C}_{k_f}^{(n)}$, we add a new node $\mathtt{n}(\mathcal{C}_{k_f}^{(n)})$ to $\mathcal{G}_{\mathcal{C}}$ (see Alg. 2, line 7) and determine the temporally preceding connected sets by calling the FindPaths function (see Alg. 2, line 8).

The FindPaths function updates graph $\mathcal{G}_{\mathcal{C}}$ by determining connected sets $\mathcal{C}_{k-1}^{(s)}$, $s \in \mathbb{N}$, reaching the provided set $\mathcal{C}_k^{(n)}$ within one time step (see Alg. 2, lines 11-22). To this end, we obtain the union of parents $\mathcal{D}_{k-1}^{\text{parents}} = \{\mathcal{D}_{k-1}^{(q)}, \mathcal{D}_{k-1}^{(p)}, \ldots\}$ for all provided sets $\mathcal{D}_k^{(i)} \in \mathcal{C}_k^{(n)}$ through the reachability graph $\mathcal{G}_{\mathcal{R}}$ (see Alg. 2, line 12). Next, we also cluster the parent sets $\mathcal{D}_{k-1}^{\text{parents}}$ to connected sets $\mathcal{C}_{k-1}^{(s)}$ (see Alg. 2, line 16). For each connected parent set $\mathcal{C}_{k-1}^{(s)}$:

1) a new node $\mathtt{n}(\mathcal{C}_{k-1}^{(s)})$ is added to $\mathcal{G}_{\mathcal{C}}$ (see Alg. 2, line 17),
2) an edge from $\mathtt{n}(\mathcal{C}_{k-1}^{(s)})$ to $\mathtt{n}(\mathcal{C}_k^{(n)})$ is created to store the fact that $\mathcal{C}_{k-1}^{(s)}$ reaches $\mathcal{C}_k^{(n)}$ (see Alg. 2, line 18),
3) the FindPaths function is called (see Alg. 2, line 19).

The FindPaths function terminates as soon as all paths from the provided set $\mathcal{C}_{k_f}^{(n)}$ to the initial set $\mathcal{C}_{k_0}^{(1)}$ are found. One may also set a timeout or other criteria for early stopping, e.g., a maximum number of driving corridors.

Algorithm 2 IdentifyCorridors

Input: reachability graph $\mathcal{G}_{\mathcal{R}}$
Optional Input: X_{lon} and C_{lon} to obtain lateral driving corridors (otherwise longitudinal driving corridors are obtained).
Output: graph $\mathcal{G}_{\mathcal{C}}$ containing possible driving corridors

```
1: $\mathcal{G}_{\mathcal{C}}$.init( )
2: $\mathcal{D}_{k_f} \leftarrow \mathcal{G}_{\mathcal{R}}$.drivableArea($k_f$)
3: if $X_{\mathrm{lon}} \neq \emptyset$ then          ▷ only for lateral driving corridors
4:     $\mathcal{D}_{k_f} \leftarrow$ ExcludeSets($\mathcal{D}_{k_f}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
5: end if
6: for all $\mathcal{C}_{k_f}^{(n)}$ in connectedSets($\mathcal{D}_{k_f}$) do
7:     $\mathcal{G}_{\mathcal{C}}$.addNode($\mathtt{n}(\mathcal{C}_{k_f}^{(n)})$)
8:     $\mathcal{G}_{\mathcal{C}} \leftarrow$ FindPaths($\mathcal{G}_{\mathcal{R}}$, $\mathcal{G}_{\mathcal{C}}$, $\mathcal{C}_{k_f}^{(n)}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
9: end for
10: return $\mathcal{G}_{\mathcal{C}}$

11: function FindPaths($\mathcal{G}_{\mathcal{R}}$, $\mathcal{G}_{\mathcal{C}}$, $\mathcal{C}_k^{(n)}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
12:     $\mathcal{D}_{k-1}^{\text{parents}} \leftarrow \mathcal{G}_{\mathcal{R}}$.getParents($\mathcal{C}_k^{(n)}$)
13:     if $X_{\mathrm{lon}} \neq \emptyset$ then          ▷ only for lateral driving corridors
14:         $\mathcal{D}_{k-1}^{\text{parents}} \leftarrow$ ExcludeSets($\mathcal{D}_{k-1}^{\text{parents}}$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
15:     end if
16:     for all $\mathcal{C}_{k-1}^{(s)}$ in connectedSets($\mathcal{D}_{k-1}^{\text{parents}}$) do
17:         $\mathcal{G}_{\mathcal{C}}$.addNode($\mathtt{n}(\mathcal{C}_{k-1}^{(s)})$)
18:         $\mathcal{G}_{\mathcal{C}}$.addEdge($\mathtt{n}(\mathcal{C}_{k-1}^{(s)}), \mathtt{n}(\mathcal{C}_k^{(n)})$)
19:         $\mathcal{G}_{\mathcal{C}} \leftarrow$ FindPaths($\mathcal{G}_{\mathcal{R}}$, $\mathcal{G}_{\mathcal{C}}$, $\mathcal{C}_{k-1}^{(s)}, X_{\mathrm{lon}}, C_{\mathrm{lon}}$)
20:     end for
21:     return $\mathcal{G}_{\mathcal{C}}$
22: end function

23: function ExcludeSets($\mathcal{D}'_k$, $X_{\mathrm{lon}}$, $C_{\mathrm{lon}}$)
24:     $\mathcal{D}'_k \leftarrow \mathcal{D}'_k \cap C_{\mathrm{lon},k}$
25:     $\mathcal{D}'_k \leftarrow$ filter($\mathcal{D}'_k$, $X_{\mathrm{lon}}$.at($k$))
26:     return $\mathcal{D}'_k$
27: end function
```

Example *At time step $k_f = 2$, the connected sets are $\mathcal{C}_2^{(1)}$, $\mathcal{C}_2^{(2)}$, and $\mathcal{C}_2^{(3)}$ (see Fig. 6b). Let us assume that* FindPaths *is invoked with $\mathcal{C}_2^{(2)}$. The obtained parents of $\mathcal{C}_2^{(2)}$ are $\mathcal{D}_1^{\text{parents}} = \{\mathcal{D}_1^{(0)}, \ldots, \mathcal{D}_1^{(4)}\}$ (see Fig. 6a-6b) that are partitioned into two connected sets $\mathcal{C}_1^{(1)}$ and $\mathcal{C}_1^{(2)}$ (see Fig. 6b). Assume that we continue with $\mathcal{C}_1^{(1)}$: we first add the node $\mathtt{n}(\mathcal{C}_1^{(1)})$ to $\mathcal{G}_{\mathcal{C}}$ (see Fig. 6c), followed by creating an edge from $\mathtt{n}(\mathcal{C}_1^{(1)})$ to $\mathtt{n}(\mathcal{C}_2^{(2)})$. Then, the* FindPaths *function is first invoked for $\mathcal{C}_1^{(1)}$. After the* FindPaths *function terminates for $\mathcal{C}_1^{(1)}$, it is invoked for $\mathcal{C}_1^{(2)}$. Eventually, we obtain the paths $(\mathcal{C}_0^{(1)}, \mathcal{C}_1^{(1)}, \mathcal{C}_2^{(2)})$ and $(\mathcal{C}_0^{(1)}, \mathcal{C}_1^{(2)}, \mathcal{C}_2^{(2)})$ for $\mathcal{C}_2^{(2)}$ (see Fig. 6c).*

C. Lateral Driving Corridors

Lateral driving corridors are obtained in a similar way to longitudinal driving corridors using Alg. 2, but with the addition that sets $\mathcal{D}_k^{(i)}$ are removed (see Alg. 2, ExcludeSets function, lines 23-27)

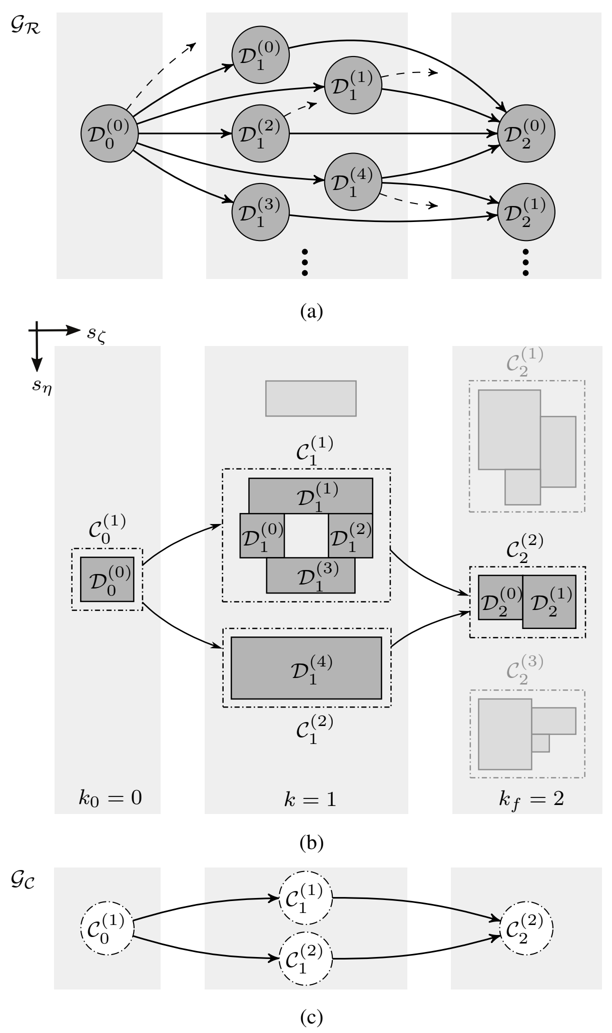


Fig. 6: Identification of longitudinal driving corridors. (a) Example of a reachability graph $\mathcal{G}_{\mathcal{R}}$. We only depict nodes and edges that are relevant for our example. (b) Corresponding drivable area (gray rectangles) of the ego vehicle for different time steps k. As an example, we highlight sets $\mathcal{D}_k^{(i)}$ in dark gray that are part of a longitudinal driving corridor reaching $\mathcal{C}_2^{(2)}$ at k_f. (c) Excerpt from the graph $\mathcal{G}_{\mathcal{C}}$. We only show paths that end in $\mathcal{C}_2^{(2)}$.

A1) that are not part of the longitudinal driving corridor C_{lon} (see Alg. 2, line 24),
A2) for which $\text{proj}_{s_\zeta}(x_{\text{lon},k}) \notin \text{proj}_{s_\zeta}(\mathcal{D}_k^{(i)})$, $x_{\text{lon},k} \in X_{\text{lon}}$, holds (see Alg. 2, line 25).

Addition A1 ensures that each identified lateral driving corridor is a subset of the longitudinal driving corridor. Since we do not update the reachability graph $\mathcal{G}_{\mathcal{R}}$ for computational efficiency, all parents, including those that are not part of the longitudinal driving corridor, are returned in Alg. 2, line 12. Addition A2 ensures that we obtain connected sets with a unique passing side at all states $x_{\text{lon},k}$ of the longitudinal trajectory X_{lon}.

Example *We continue the example in Fig. 6. Let us assume that the selected longitudinal driving corridor is $C_{\text{lon}} = (\mathcal{C}_0^{(1)}, \mathcal{C}_1^{(1)}, \mathcal{C}_2^{(2)})$. In Fig. 7, the longitudinal positions of X_{lon} are highlighted. For brevity, we only concentrate on the modifications in Alg. 2 for determining lateral driving*

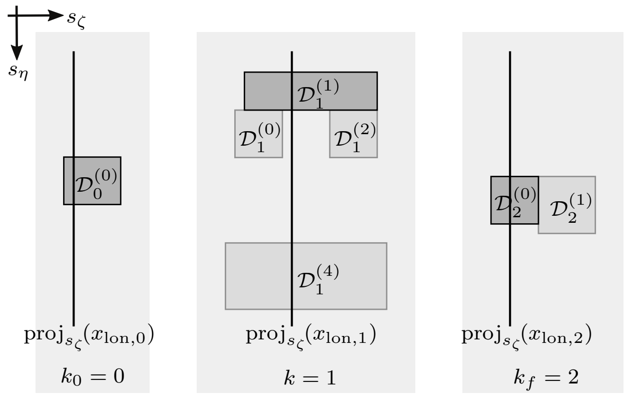


Fig. 7: Identification of lateral driving corridors. As an example, we highlight sets $\mathcal{D}_k^{(i)}$ in dark gray that are part of the lateral driving corridor reaching $\mathcal{D}_2^{(0)}$ at the final time step k_f.

corridors: at time step $k_f = 2$, the connected sets $\mathcal{C}_2^{(1)}$ and $\mathcal{C}_2^{(3)}$ are removed as they are not part of C_{lon} (addition A1, see Fig. 6b). We also exclude $\mathcal{D}_2^{(1)}$ from $\mathcal{C}_2^{(2)}$ due to addition A2 (see Fig. 7 and Alg. 2, line 4). Then, the FindPaths *function is invoked from $\{\mathcal{D}_2^{(0)}\}$. The parent sets of $\mathcal{D}_2^{(0)}$ are $\mathcal{D}_1^{\text{parents}} = \{\mathcal{D}_1^{(0)}, \mathcal{D}_1^{(1)}, \mathcal{D}_1^{(2)}, \mathcal{D}_1^{(4)}\}$ (see Fig. 6a). We remove $\mathcal{D}_1^{(4)}$ and $\{\mathcal{D}_1^{(0)}, \mathcal{D}_1^{(2)}\}$ due to addition A1 and A2, respectively (see Fig. 7 and Alg. 2, lines 13-15).*

VI. Determining Collision Avoidance Constraints from Driving Corridors

The longitudinal trajectory planner (see Def. 5) optimizes the longitudinal motion of the ego vehicle along the reference path $\Gamma(s_\zeta)$, i.e., $\forall k:\ \theta_k = \theta_\Gamma(s_{\zeta,k})$ and $s_{\eta,k} = 0$. The lateral trajectory planner (see Def. 6) optimizes the lateral motion of the ego vehicle, so that the ego vehicle is able to deviate from the reference path, i.e., for some k, it may hold that $\theta_k \neq \theta_\Gamma(s_{\zeta,k})$ or $s_{\eta,k} \neq 0$. To guarantee collision avoidance, we constrain all possible trajectories of the ego vehicle to stay within the longitudinal and lateral driving corridor C_{lon} and C_{lat}, which is described below.

A. Longitudinal Constraints

Longitudinal collision avoidance is realized by enforcing a minimum and maximum bound $\underline{s}_{\zeta,k}$ and $\overline{s}_{\zeta,k}$ on the longitudinal position $s_{\zeta,k}$ of the reference circle ${}^{\mathtt{L}}c_k^{(1)}$ (center of rear circle, see Fig. 2) at each time step k. As the shape of the ego vehicle is considered during the reachable set computation, we conclude that the ego vehicle is collision-free if ${}^{\mathtt{L}}c_k^{(1)}$ is located within the drivable area $\mathcal{D}_k$. Since $C_{\text{lon},k} \subseteq \mathcal{D}_k$, we obtain $\underline{s}_{\zeta,k}$ and $\overline{s}_{\zeta,k}$ for the longitudinal position constraint (4d) from the longitudinal driving corridor $C_{\text{lon},k}$:

$$\underline{s}_{\zeta,k} = \inf\left\{\text{proj}_{s_\zeta}(C_{\text{lon},k})\right\},\ \ \overline{s}_{\zeta,k} = \sup\left\{\text{proj}_{s_\zeta}(C_{\text{lon},k})\right\}.$$

B. Lateral Constraints

To model lateral collision avoidance, we introduce the linearized lateral deviation $d_k^{(i)}$ of the i-th circle's center to the reference path at time step k (see Fig. 2):

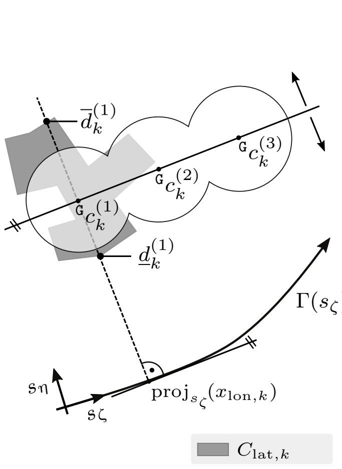
$$d_k^{(i)} = s_{\eta,k} + \frac{i-1}{2}\ell\left(\theta_k - \theta_\Gamma(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))\right), \qquad (8)$$

where $x_{\mathrm{lon},k}$ is the k-th state of the longitudinal trajectory X_{lon} and $i \in \{1,2,3\}$. We are able to guarantee collision avoidance in the lateral direction by limiting the deviation (8) of centers ${}^{\mathtt{G}}c_k^{(i)}$ at each time step k (see (5d)). Below, we assume that $\theta_k = \theta_\Gamma(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))$ to obtain the limits on (8). Note that the limits obtained are still valid for $\theta_k \neq \theta_\Gamma(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))$ due to the small-angle approximation (see Sec. II). Below, we use these findings to obtain the admissible lateral deviations for the circles ${}^{\mathtt{G}}c_k^{(i)}$, $i \in \{1,2,3\}$.

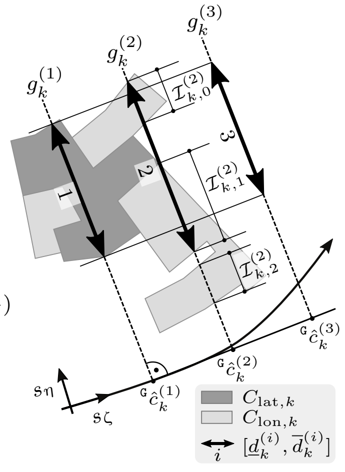
1) Admissible Deviations for Reference Circle: Using a similar reasoning as for the longitudinal position constraints in Sec. VI-A, we obtain the minimum and maximum admissible deviations $\underline{d}_k^{(1)}$ and $\overline{d}_k^{(1)}$ of ${}^{\mathtt{G}}c_k^{(1)}$ from the reference path $\Gamma(s_\zeta)$ directly from C_{lat}. Figuratively speaking, we move the ego vehicle perpendicular to the reference path at $\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$ and determine all positions of the lateral driving corridor $C_{\mathrm{lat},k}$ intersecting with ${}^{\mathtt{G}}c_k^{(1)}$ (see Fig. 8a):

$$\underline{d}_k^{(1)} = \inf\left\{\mathrm{proj}_{s_\eta}(C_{\mathrm{lat},k})\right\}, \quad \overline{d}_k^{(1)} = \sup\left\{\mathrm{proj}_{s_\eta}(C_{\mathrm{lat},k})\right\}.$$

2) Admissible Deviations for Center and Front Circles: The maximum and minimum admissible deviations $\underline{d}_k^{(i)}$ and $\overline{d}_k^{(i)}$, $i \in \{2,3\}$, for the center and front circle cannot be extracted directly from the lateral driving corridor C_{lat}. The reason is that $C_{\mathrm{lat},k}$ only contains collision-free, reachable positions of ${}^{\mathtt{L}}c_k^{(1)}$ in the neighborhood of $\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$, $x_{\mathrm{lon},k} \in X_{\mathrm{lon}}$ (see addition A2 in Sec. V-C). Thus, ${}^{\mathtt{G}}c_k^{(2)}$ and ${}^{\mathtt{G}}c_k^{(3)}$ may be even located outside of $C_{\mathrm{lat},k}$ (see Fig. 8a). Yet, we can set the bounds $\underline{d}_k^{(i)}$ and $\overline{d}_k^{(i)}$, $i \in \{2,3\}$, to $\underline{d}_k^{(1)}$ and $\overline{d}_k^{(1)}$, respectively, since we have already considered the shape of the ego vehicle in the reachable set computation (see Sec. IV-D) for $\theta_k = \theta_\Gamma(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))$.


$\overline{d}_k^{(1)}$
${}^{\mathtt{G}}c_k^{(3)}$
${}^{\mathtt{G}}c_k^{(2)}$
${}^{\mathtt{G}}c_k^{(1)}$
$\underline{d}_k^{(1)}$
$\Gamma(s_\zeta)$
s_η
s_ζ
$\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$
$C_{\mathrm{lat},k}$

(a) Admissible deviations $d_k^{(1)}$ of center ${}^{\mathtt{G}}c_k^{(1)}$ from $\Gamma(s_\zeta)$ are determined with the lateral driving corridor C_{lat}.


$g_k^{(3)}$
$g_k^{(2)}$
$g_k^{(1)}$
$\mathcal{I}_{k,0}^{(2)}$
$\mathcal{I}_{k,1}^{(2)}$
$\mathcal{I}_{k,2}^{(2)}$
1
2
3
${}^{\mathtt{G}}\hat{c}_k^{(3)}$
${}^{\mathtt{G}}\hat{c}_k^{(2)}$
${}^{\mathtt{G}}\hat{c}_k^{(1)}$
s_η
s_ζ
$C_{\mathrm{lat},k}$
$C_{\mathrm{lon},k}$
$[\underline{d}_k^{(i)}, \overline{d}_k^{(i)}]$
i

(b) Admissible deviations $d_k^{(i)}$ of centers ${}^{\mathtt{G}}c_k^{(i)}$, $i \in \{2,3\}$, from $\Gamma(s_\zeta)$ are determined with C_{lon}.

Fig. 8: We obtain the minimum and maximum admissible deviation $\underline{d}_k^{(i)}$ and $\overline{d}_k^{(i)}$ through driving corridors C_{lon} and C_{lat}.

However, this choice might be overly conservative for large time steps k, since we restrict the limits $\underline{d}_k^{(i)}$ and $\overline{d}_k^{(i)}$, $i \in \{2,3\}$, to the bounds of $d_k^{(1)}$. We aim to use the longitudinal driving corridor C_{lon} to extend the deviation limits of the center and front circles based on the following observation: the connected sets of the longitudinal driving corridor C_{lon} usually increase in size for greater time steps k (since more states are reachable), so that ${}^{\mathtt{L}}c_k^{(2)}$, ${}^{\mathtt{L}}c_k^{(3)} \in C_{\mathrm{lon},k}$ for ${}^{\mathtt{L}}c_k^{(1)} \in C_{\mathrm{lat},k}$. We can infer that the center and front circle are collision-free within $C_{\mathrm{lon},k}$, since a circle with radius r is guaranteed to be collision-free in the longitudinal driving corridor (see Sec. IV-D). We therefore continue with the determination of the admissible deviations from the reference path for which the center and front circle are contained in the longitudinal driving corridor.

Let us introduce the normal vector $\eta(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))$ of the reference path at the longitudinal position $\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$. We further introduce the straight line $g_k^{(i)}$ that is parallel to $\eta(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))$ passing through ${}^{\mathtt{G}}\hat{c}_k^{(i)}$, where ${}^{\mathtt{G}}\hat{c}_k^{(i)}$ is the Cartesian position of the circle centers if the ego vehicle is located on $\Gamma(s_\zeta)$ at $\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k})$ (see Fig. 8b). The point ${}^{\mathtt{G}}\hat{c}_k^{(i)}$ is obtained with (7) for ${}^{\mathtt{L}}c_k^{(1)} = (\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}), 0)^T$. The distances from ${}^{\mathtt{G}}\hat{c}_k^{(i)}$ to all positions in $C_{\mathrm{lon},k}$ intersecting $g_k^{(i)}$ are:

$$\mathcal{Y}_k^{(i)} = \Bigg\{ \overbrace{\eta(\mathrm{proj}_{s_\zeta}(x_{\mathrm{lon},k}))^T \left(T_{\mathtt{L}}^{\mathtt{G}}(s_{\zeta,k}) \begin{pmatrix} s_{\zeta,k} \\ s_{\eta,k} \end{pmatrix} - {}^{\mathtt{G}}\hat{c}_k^{(i)}\right)}^{\text{distance}} \in \mathbb{R} \,\Bigg|$$

$$\begin{pmatrix} s_{\zeta,k} \\ s_{\eta,k} \end{pmatrix} \in C_{\mathrm{lon},k},\; \underbrace{T_{\mathtt{L}}^{\mathtt{G}}(s_{\zeta,k}) \begin{pmatrix} s_{\zeta,k} \\ s_{\eta,k} \end{pmatrix} \in g_k^{(i)}}_{\text{intersection points}} \Bigg\}.$$

If $\mathcal{Y}_k^{(i)} \neq \emptyset$, there exist sets $\mathcal{D}_k^{(j)} \in C_{\mathrm{lon},k}$ intersecting $g_k^{(i)}$ in $F^{\mathtt{G}}$ (see Fig. 8b, $g_k^{(2)}$). We unify the intersection distances $\mathcal{Y}_k^{(i)}$ to intervals $\mathcal{I}_{k,q}^{(i)}$, $q \in \mathbb{N}_0$, as illustrated in Fig. 8b, and consider only those intervals $\mathcal{I}_{k,q}^{(i)}$ where $[\underline{d}_k^{(1)}, \overline{d}_k^{(1)}] \cap \mathcal{I}_{k,q}^{(i)} \neq \emptyset$. The collection of the intervals considered is denoted with $\mathcal{I}_{\mathrm{v}}^{(i)}$. As illustrated in Fig. 8b, interval $\mathcal{I}_{k,2}^{(2)}$ is omitted and we only consider intervals $\mathcal{I}_{k,0}^{(2)}$ and $\mathcal{I}_{k,1}^{(2)}$, i.e., $\mathcal{I}_{\mathrm{v}}^{(2)} = \{\mathcal{I}_{k,0}^{(2)}, \mathcal{I}_{k,1}^{(2)}\}$. The deviation limits are then determined by

$$[\underline{d}_k^{(i)}, \overline{d}_k^{(i)}] = [\underline{d}_k^{(1)}, \overline{d}_k^{(1)}] \cup \mathcal{I}_{\mathrm{v}}^{(i)}, \quad i \in \{2,3\}.$$

VII. Experiments

After introducing implementation details in Sec. VII-A, we validate the drivability of our planned motions by real-world experiments in Sec. VII-B. In Sec. VII-C, we compare our method to motion planners based on discretization and continuous optimization. Subsequently, we demonstrate that our approach works in arbitrarily complex scenarios. A video of the experiments is attached to this paper.

A. Implementation Details

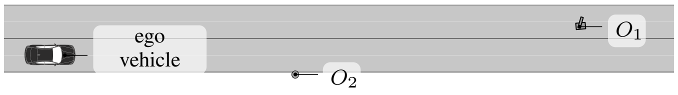
Our approach is implemented partly in Python and C++ on a computer with an Intel Core i7-6700HQ CPU and 16 GB of memory. We use the CommonRoad benchmark suite to model our scenarios [68]. The assessment criteria for driving corridor selection can be chosen arbitrarily. In this work, we select the paths in $\mathcal{G}_\mathcal{C}$ with the greatest cumulated area of connected sets, since connected areas of greater size generally yield less restrictive position constraints for trajectory optimization.

For longitudinal and lateral trajectory planning (see Def. 5 and 6), we apply the approach presented in [30], where system (4b) is a fourth-order integrator model with jounce as input and bounded velocity, acceleration, and jerk. System (5b) is a linearized kinematic single-track model with constraints on the steering actuators. We select a quadratic cost function for both the longitudinal and lateral optimization. The longitudinal cost function J_{lon} punishes deviations from a desired velocity, high accelerations, jerk, and inputs. The lateral cost function J_{lat} minimizes the lateral distance to $\Gamma(s_\zeta)$, deviations from $\theta_\Gamma(s_\zeta)$ and $\kappa_\Gamma(s_\zeta)$, as well as high curvature rates and inputs to obtain smooth trajectories.

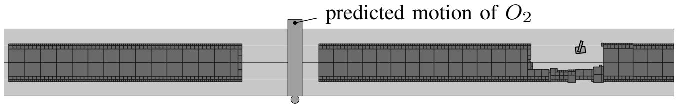
B. Drivability of Planned Motions

We have integrated our approach in a BMW 7-series test vehicle and applied it online to determine the driving corridor. Fig. 9a illustrates the real-world scenario with two lanes, in which we placed a static obstacles O_1 and a simulated pedestrian O_2 in the driveway of the vehicle. The obstacles are detected by LiDAR sensors and we use the set-based prediction *SPOT* [69] to predict the future motion of the pedestrian over time. Note that the prediction method can be replaced and is not part of our algorithm.

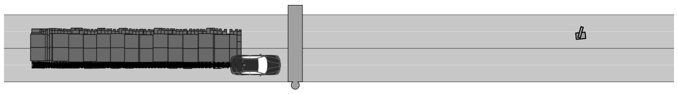
Fig. 9b shows the drivable area at the final time step, from which we infer that two maneuvers exist: a) stopping in front of the pedestrian, and b) swerving. The trajectory that ends in front of the pedestrian and its driving corridor are visualized in Fig. 9c. The swerving trajectory is illustrated in Fig. 9d, in which the vehicle first passes the pedestrian O_2 on the left and then evades the static obstacle O_1 by changing to the original lane. Fig. 9e shows the drivable area of the scenario at $t_{30} = 6.0\,\mathrm{s}$. Even though the solution space is exceedingly small, we are able to detect even narrow passageways by using reachable sets for driving corridor identification. This is of utmost importance in safety-critical scenarios, in which evading may be the last possible feasible maneuver. Even though our approach uses different models for the reachability analysis and trajectory planning, we obtain drivable trajectories as illustrated in Fig. 9f and 9g, which show the nominal and measured velocity and curvature profiles of the executed double lane change maneuver.

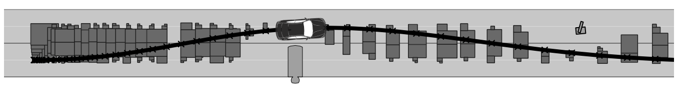


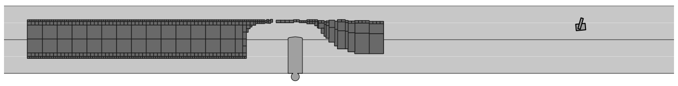
(a) Initial scenario with a static obstacle O_1 and a pedestrian O_2 at $t = 0\,\mathrm{s}$.

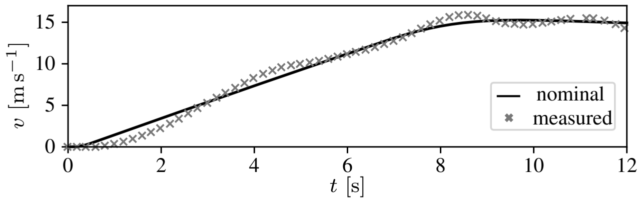


(b) Drivable area $\mathcal{D}_{70}$ at the final time step $t_{70} = 14\,\mathrm{s}$.

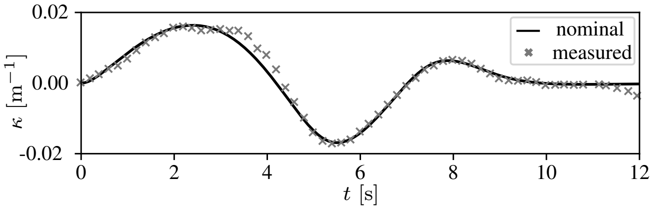

(c) Planned trajectory that ends in front of the pedestrian and the corresponding driving corridor C_{lat}.


(d) Planned evasive trajectory and the corresponding driving corridor C_{lat} at $t_{27} = 5.4\,\mathrm{s}$.


(e) Small solution space within the drivable area $\mathcal{D}_{30}$ at $t_{30} = 6.0\,\mathrm{s}$.



(f) Nominal and measured velocity profile.



(g) Nominal and measured curvature profile.

(h) Images of the vehicle tests with a BMW 7-series.

Fig. 9: Validation of our approach on a BMW 7-series test vehicle (scenario ZAM_Urban-8_1_S-1:2018b).

C. Comparison

1) Sampling-based Motion Planner: We compare our method with a popular sampling-based trajectory planner presented in [15] on the previous scenario to demonstrate the efficacy of our method to detect narrow passages in the solution space. The motion planner in [15] samples a finite set of trajectories that connect the initial state of the ego vehicle with different goal states. The longitudinal and lateral motions are planned separately using quartic and quintic polynomials in F^{L}. The Cartesian product of all longitudinal and lateral motions yields the set of candidate trajectories that are checked for drivability and collisions. Due to the pre-defined s-shape of the trajectories of the sampling-based planner, a double lane change maneuver that is similar to the one of our method (see Fig. 9d) can only be planned in an anticipatory way. We therefore remove the static obstacle O_1 and aim to sample at least one feasible trajectory that evades the pedestrian, which

TABLE I: Parameters for Sampling.

scenario	goal state	min. value	max. value	step size
baseline & critical	t in [s]	0.40	10.0	0.20
	$d^{(2)}$ in [m]	-4.00	4.00	1.00
baseline	v_ζ in [m/s]	6.00	18.00	1.5
critical	v_ζ in [m/s]	6.00	18.00	0.2

is particularly challenging due to the small solution space (see Fig. 9e). This scenario is referred to as baseline scenario. Then, we add an additional constraint to the motion planning to increase the difficulty and demand that the ego vehicle must keep a distance of $0.2\,\mathrm{m}$ to the pedestrian. We therefore dilate the predicted occupancies of the pedestrian with a circle with radius $0.2\,\mathrm{m}$ and refer to this scenario as critical scenario. Subsequently, we compare the performance of both planners.

Following [15], we compute 3969 trajectories using the sampling-based planner in the baseline scenario. The sampling parameters are uniformly distributed and can be found in Tab. I. From 3969 trajectories, we found two feasible trajectories evading the pedestrian. However, using the same sampling parameters, no feasible trajectory has been found in the critical scenario. We therefore gradually raise the number of goal states by approximately bisecting the step size for velocity sampling, i.e., we used step sizes $\{1.5, 0.8, 0.4, 0.2\}\mathrm{m/s}$, until one feasible trajectory is found. The final sampling parameters are listed in Tab. I. Overall, we needed 26901 trajectories to obtain one feasible trajectory evading the pedestrian in the critical scenario (see Fig. 10). The computation times of the baseline and critical scenario are $3.2\,\mathrm{s}$ and $20.8\,\mathrm{s}$, respectively; thus, the runtime increases by factor 6.5. In contrast, our motion planning approach is able to solve both scenarios without the need for adapting parameters and the computation times do not substantially differ (baseline scenario: $0.078\,\mathrm{s}$, critical scenario: $0.076\,\mathrm{s}$).

This scenario particularly demonstrates that sampling-based motion planners struggle to detect narrow passage ways in the solution space for trajectory planning due to the discretization of the state space. In contrast, our method can automatically detect narrow passages without increased computational effort. One might argue that we have used a naiv sampling strategy, however, manually tuning or learning adaptive sampling strategies for different traffic situations is error-prone and difficult. In comparison, our method for driving corridor identification can be applied immediately without parameter tuning.

2) Mixed-Integer Quadratic Programming: We analyze the influence of an increasing number of obstacles on the runtime behavior of our method. For comparison, we select the motion planner proposed in [70] based on mixed-integer quadratic programming (MIQP). The reason for selecting a MIQP planner is two-fold: 1) it is a continuous optimization method, and thus, does not suffer from discretization effects, and 2) mixed-integer programming has gained increasing interest over the past years, e.g., see [6], [20], [70]–[72], as binary variables enable to incorporate discrete tactical decisions into the trajectory optimization.

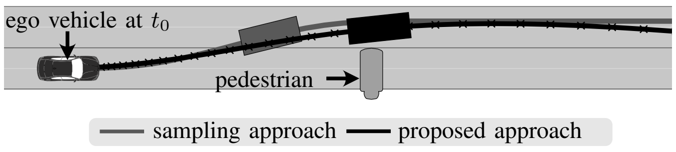
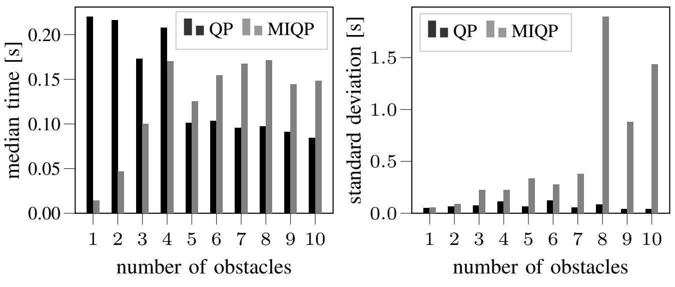


Fig. 10: Critical scenario at $t_{27} = 5.4\,\mathrm{s}$.



(a) Median comp. times (b) Standard deviation of comp. times

Fig. 11: Runtime behavior of our approach (label QP) and a MIQP planner [70] (label MIQP) in randomly generated scenarios with up to ten obstacles.

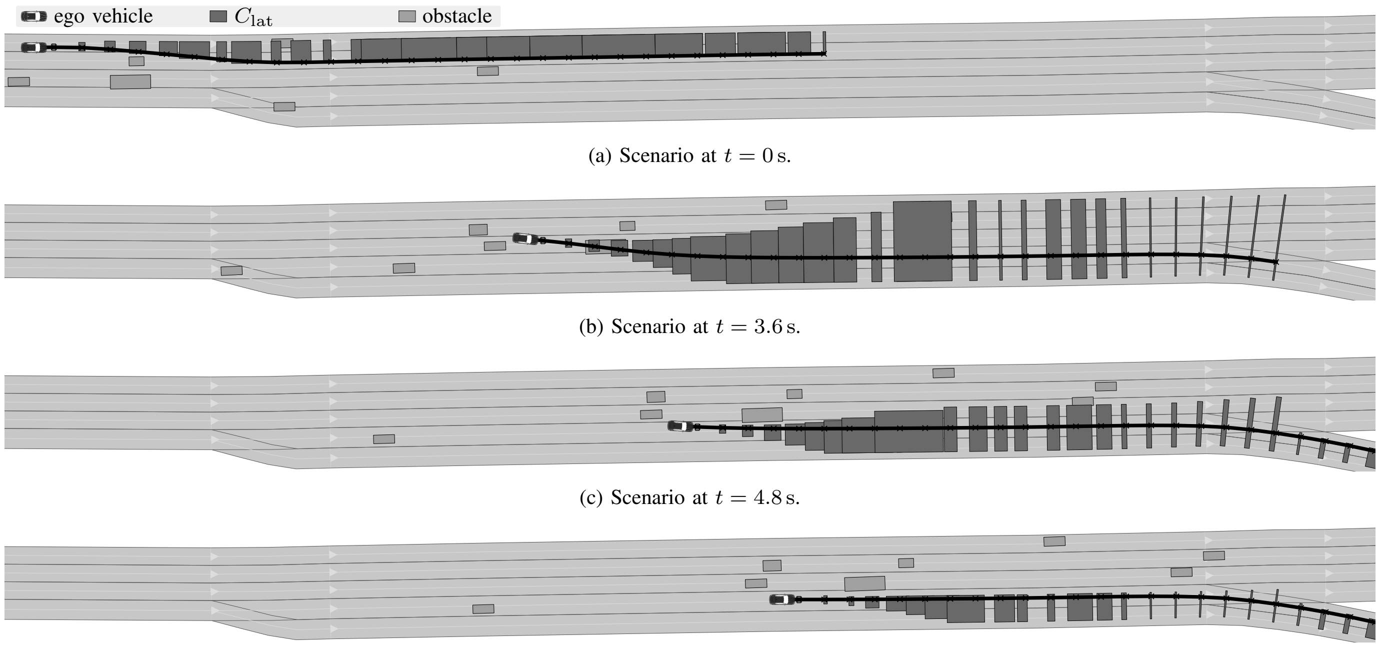
We consider the same road as in Fig. 10 and gradually increase the number $\sigma \in \{1, \dots, 10\}$ of static obstacles in the environment. The obstacles are represented by rectangles, i.e., $[s_x - {l_{\mathrm{obs}}}/2, s_x + {l_{\mathrm{obs}}}/2] \times [s_y - {w_{\mathrm{obs}}}/2, s_y + {w_{\mathrm{obs}}}/2]$, that are randomly sampled, where $s_x \in [25, 180]\,\mathrm{m}$, $s_y \in [-1.75, 5.25]\,\mathrm{m}$, $l_{\mathrm{obs}} \in [1.0, 20.0]\,\mathrm{m}$, $w_{\mathrm{obs}} \in [0.5, 4.0]\,\mathrm{m}$. Since our motion planner uses a kinematic single-track model for optimization that is more restrictive compared to the third-order integrator models of [70], we only consider scenarios for the analysis that both planners could solve. Overall, we generated 50 scenarios for each σ. In all scenarios, trajectories are planned for 50 time steps with $\Delta t = 0.2\,\mathrm{s}$ and the initial velocity of the ego vehicle was $15\,\mathrm{m/s}$. To compute the solution for the MIQP, we use the solver Gurobi [73], version 9.0.2.

As illustrated in Fig. 11a, the median computation times for both planners are comparable. While the MIQP planner has lower computation times for scenarios with up to 4 obstacles, our approach is faster for scenarios with more than 4 obstacles. Our method is already applicable for complex road geometries and obstacles with arbitrary shapes, whereas the MIQP planner is tailored to our experimental setup, i.e., lane boundaries are constant and obstacles have rectangular shapes. Modeling complex obstacle geometries, e.g., arbitrary polygonal shapes, increases the computation times of the MIQP planner [70].

The MIQP planner solves many scenarios rather fast, however, the runtime behavior is highly volatile. As shown in Fig. 11b, the standard deviation of the computation times are particularly high for scenarios with multiple obstacles and have the tendency to increase with a raising number of obstacles. In contrast, the computation times of our method vary only slightly over the different scenarios. The experiment indicates that the runtime behavior of our method is reliable, which is of high importance for real-time applications.

D. Motion Planning in a Complex Scenario

Let us consider a complex scenario, where the ego vehicle is in the far left lane and plans to take the highway off-



(a) Scenario at $t = 0\,\mathrm{s}$.

(b) Scenario at $t = 3.6\,\mathrm{s}$.

(c) Scenario at $t = 4.8\,\mathrm{s}$.

(d) Scenario at $t = 5.6\,\mathrm{s}$.

Fig. 12: Planned trajectories of the ego vehicle for different planning cycles (scenarios DEU_A9-3_{1,...,15}_T-1:2018b).

ramp (see Fig. 12a). Even for experienced human drivers, this maneuver is challenging as multiple lane changes need to be planned. In order to account for realistic movements by other traffic participants, this scenario is based on real traffic that we recorded using a BMW 7-series vehicle. Again, we use *SPOT* [69] to predict the future motion of the other traffic participants and assume that traffic participants will remain in their current lanes, and accelerate or decelerate only slightly. Since this maneuver requires longer planning horizons, we replan the trajectory for consecutive planning cycles. In each planning cycle, trajectories are planned for 30 time steps using $\Delta t = 0.2\,\mathrm{s}$; a new planning cycle starts every $0.4\,\mathrm{s}$.

Fig. 12 illustrates the planned trajectories and the corresponding lateral driving corridors at selected times t. Other traffic participants are shown in their measured state without uncertainties at time t. As can be seen in Fig. 12b, our method enables the ego vehicle to merge into small gaps while finally reaching the highway off-ramp (see Fig. 12d). The median computation times of the reachability analysis and of trajectory planning, including the driving corridor identification, are $\approx 131\,\mathrm{ms}$ and $\approx 154\,\mathrm{ms}$, respectively. This scenario highlights that driving corridors allow the ego vehicle to maneuver even in dense traffic with multiple road users.

E. Scenarios with a Small Solution Space

Next, we analyze the influence of a decreasing solution space. We therefore group the computation times of our approach (reachability analysis and trajectory planning including driving corridor identification) obtained in Sec. VII-C2 according to the size of the drivable area cumulated over all time steps. As shown in Fig. 13, the computation times tend to decrease given that the cumulative drivable area shrinks. Since the number of driving corridors can be limited (thus, the number of planned trajectories), the main factor for the decrease in computation time is the reachability analysis that we further examine below.

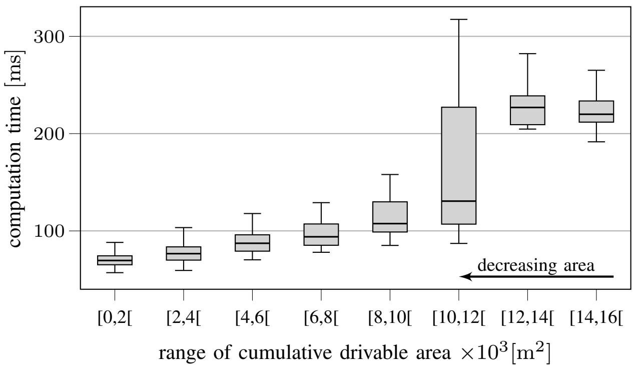


Fig. 13: The computation times of our approach have the tendency to decrease when the cumulative drivable area decreases. For better visibility, outliers of the boxplots are not shown.

We increase the criticality of the CommonRoad scenario USA_US101-6_1_T-1:2018b (see Fig. 14) featuring 29 dynamic obstacles. By gradually raising the initial velocity of the ego vehicle by $1.4\,\mathrm{m/s}$, the traffic situations becomes more dangerous compared to the baseline scenario, when using, e.g., the Time-to-Collision or the Inter-Vehicle-Time, as thread-measure [74]. We emphasize that criticality measures are not part of the presented algorithm; an overview can be found in [74]. For each scenario, the reachable set is computed for 50 time steps with $\Delta t = 0.1\,\mathrm{s}$ and repeated 20 times to account for fluctuations in the computation time. As shown in Tab. II, the median computation time of the reachable set decreases with increasing criticality of the traffic situation, since fewer set operations have to be performed, i.e., the number of base sets cumulated over all time steps k decreases. Also, the size of the cumulated drivable area is reduced by roughly 75%

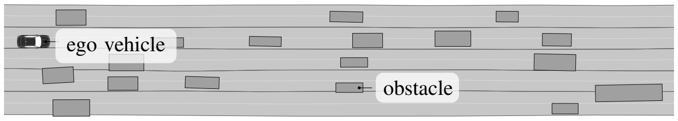


Fig. 14: Scenario USA_US101-6_1_T-1 of the CommonRoad benchmark suite. The obstacles and the ego vehicle are depicted at the initial time step.

TABLE II: Decreasing median computation times, number of base sets, and size of the drivable area cumulated over all time steps k, when increasing the initial velocity in scenario USA_US101-6_1_T-1.

Init. Vel.	**Comp. Time**	**No. Base Sets**	**Red. Drivable Area**
16.79 m/s	74.76 ms	2025	0.00 %
18.19 m/s	69.51 ms	1843	8.65 %
19.59 m/s	65.65 ms	1712	17.55 %
20.99 m/s	58.90 ms	1561	27.25 %
22.39 m/s	52.93 ms	1346	37.95 %
23.79 m/s	45.46 ms	1112	49.00 %
25.19 m/s	37.79 ms	899	59.47 %
26.59 m/s	31.30 ms	678	67.97 %
27.99 m/s	27.19 ms	535	75.40 %

when comparing the baseline scenario with the most-critical scenario. Thus, our approach is particularly suited for complex situations with small solution spaces, where fast reaction times are a necessity to avoid collisions.

VIII. Conclusions

This paper presents a novel motion planning approach for automated vehicles by combining reachability analysis with convex optimization. In contrast to most existing work on motion planning, our approach can be applied in arbitrarily complex traffic situations. We showed that set-based techniques have the potential to reduce the computation time of motion planners when the criticality of the scenario increases, i.e., the solution space becomes smaller and more convoluted. In real vehicle tests, we validated the extraction of collision avoidance constraints from the obtained driving corridors to plan drivable trajectories. Our results indicate that the presented approach can drastically enhance the safety of automated vehicles and their ability to determine complex driving maneuvers. Additionally, we are able to constrain driving corridors to end in predefined terminal states. Thus, automated vehicles can choose the most appropriate maneuver to achieve the desired driving task.

Appendix A
Forward Propagation

The forward propagation is performed similar to [9]. However, in [9], the absolute limits of the acceleration are restricted to be equal ($|\underline{a}_\zeta| = |\overline{a}_\zeta|$ and $|\underline{a}_\eta| = |\overline{a}_\eta|$), whereas we allow the maximum braking deceleration to be different from the maximum acceleration. According to [9], the lower and upper bounds of the reachable positions and velocities for system model (6) at $t = \Delta t$ are obtained by applying a bang-bang input with switching time $\gamma\Delta t$ for $\gamma \in [0, 1]$. Below, we present the solution only in the longitudinal direction, the solution in the lateral direction is obtained similarly. The lower bounds $s_\zeta^{\mathrm{lo}}(\gamma)$ and $v_\zeta^{\mathrm{lo}}(\gamma)$ on the position and velocity in the longitudinal direction are obtained by applying full acceleration until time $\gamma\Delta t$ followed by full deceleration:

$$\begin{aligned} s_\zeta^{\mathrm{lo}}(\gamma) = s_\zeta(0) + v_\zeta(0)\Delta t + \frac{1}{2}\underline{a}_\zeta \Delta t^2(1 - 2\gamma + \gamma^2) \\ + \frac{1}{2}\overline{a}_\zeta \Delta t^2 (2\gamma - \gamma^2), \end{aligned} \tag{9a}$$

$$v_\zeta^{\mathrm{lo}}(\gamma) = v_\zeta(0) + \underline{a}_\zeta \Delta t(1-\gamma) + \overline{a}_\zeta \gamma \Delta t. \tag{9b}$$

The upper bounds $s_\zeta^{\mathrm{hi}}(\gamma)$ and $v_\zeta^{\mathrm{hi}}(\gamma)$ on the position and velocity in the longitudinal direction are obtained by applying full deceleration until time $\gamma\Delta t$ followed by full acceleration for $(1-\gamma)\Delta t$:

$$\begin{aligned} s_\zeta^{\mathrm{hi}}(\gamma) = s_\zeta(0) + v_\zeta(0)\Delta t + \frac{1}{2}\overline{a}_\zeta \Delta t^2(1 - 2\gamma + \gamma^2) \\ + \frac{1}{2}\underline{a}_\zeta \Delta t^2 (2\gamma - \gamma^2), \end{aligned} \tag{10a}$$

$$v_\zeta^{\mathrm{hi}}(\gamma) = v_\zeta(0) + \overline{a}_\zeta \Delta t(1-\gamma) + \underline{a}_\zeta \gamma \Delta t. \tag{10b}$$

From this, we can infer that a polytope $\mathcal{P}_{\zeta,k-1}^{(i)}$ can be propagated as shown in [9]:

$$\mathcal{P}_{\zeta,k}^{(i)} = \underbrace{\left(\begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \mathcal{P}_{\zeta,k-1}^{(i)} \oplus \mathcal{P}_{u,\zeta}(\Delta t)\right)}_{\text{solution of (6a) + (6c)}} \cap \underbrace{\left(\mathbb{R} \times [\underline{v}_\zeta, \overline{v}_\zeta]\right)}_{\text{velocity constraints (6b)}},$$

where $\oplus$ denotes the Minkowski sum and $\mathcal{P}_{u,\zeta}(\Delta t)$ is the set of all states that can be reached from the initial state $(s_\zeta, v_\zeta)^T = (0,0)^T$ when all admissible inputs are applied (6c). For computational reasons, $\mathcal{P}_{u,\zeta}(\Delta t)$ is over-approximated with a predefined number of halfspaces. Using (9), the linear equation for one halfspace at some γ is [9]:

$$\begin{pmatrix} \frac{\mathrm{d}s_\zeta^{\mathrm{lo}}}{\mathrm{d}\gamma} \\ \frac{\mathrm{d}v_\zeta^{\mathrm{lo}}}{\mathrm{d}\gamma} \end{pmatrix}^T \Bigg|_\gamma \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}^T \begin{pmatrix} s_\zeta \\ v_\zeta \end{pmatrix} \leq \begin{pmatrix} \frac{\mathrm{d}s_\zeta^{\mathrm{lo}}}{\mathrm{d}\gamma} \\ \frac{\mathrm{d}v_\zeta^{\mathrm{lo}}}{\mathrm{d}\gamma} \end{pmatrix}^T \Bigg|_\gamma \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}^T \begin{pmatrix} s_\zeta^{\mathrm{lo}} \\ v_\zeta^{\mathrm{lo}} \end{pmatrix} \Bigg|_\gamma .$$

Similarly, we obtain the halfspace for (10).

Appendix B
Removal of Forbidden States

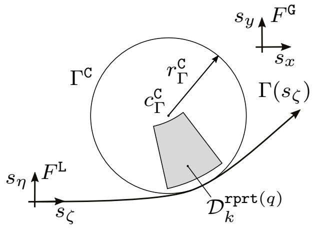
This section elaborates on the determination of the rectangular set $\mathcal{A}^\epsilon$ that over-approximates the reachable positions of centers ${}^{\mathtt{L}}c_k^{(i)}$ for the removal of forbidden states in the reachable set computation (see Sec. IV-D). Particularly, we present a solution for the following problem:

Problem 1 *Given that* $\theta_k = \theta_\Gamma(\mathrm{proj}_{s_\zeta}({}^{\mathtt{L}}c_k^{(1)}))$*, we aim to find the rectangular set* $\mathcal{A}^\epsilon = [0, \overline{\epsilon}_\zeta] \times [-\underline{\epsilon}_\eta, \overline{\epsilon}_\eta]$ *such that* $\forall {}^{\mathtt{L}}c_k^{(1)} \in \mathcal{D}_k^{\mathtt{rprt}(q)} \forall i \in \{1,2,3\}$*:* ${}^{\mathtt{L}}c_k^{(i)} \in \mathcal{D}_k^{\mathtt{rprt}(q)} \oplus \mathcal{A}^\epsilon$*.*

Once the necessary assumptions have been introduced for computing $\mathcal{A}^\epsilon$ in Appendix B-A, the longitudinal enlargement $\overline{\epsilon}_\zeta$ is derived in Appendix B-B, followed by the lateral enlargements $\overline{\epsilon}_\eta$ and $\underline{\epsilon}_\eta$.

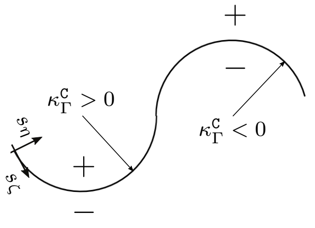
A. Notation and Assumptions

We denote the respective minimum and maximum lateral position of $\mathcal{D}_k^{\mathtt{rprt}(q)}$ by $\underline{s}_{\eta,k}^{\mathtt{rprt}(q)} = \inf\left\{\mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})\right\}$



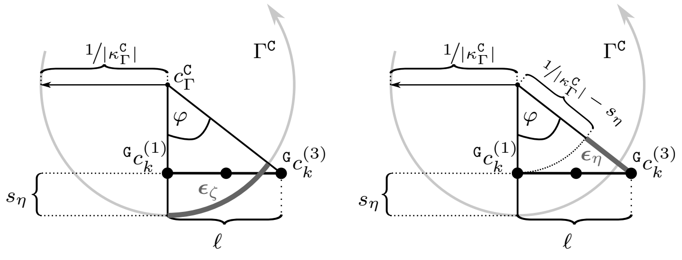
s_y $F^{\mathtt{G}}$ s_x
$\Gamma^{\mathtt{C}}$
$r_\Gamma^{\mathtt{C}}$
$c_\Gamma^{\mathtt{C}}$
$\Gamma(s_\zeta)$
s_η $F^{\mathtt{L}}$ s_ζ
$\mathcal{D}_k^{\mathtt{rprt}(q)}$

Fig. 15: Local approximation of reference path $\Gamma(s_\zeta)$ with circle $\Gamma^{\mathtt{C}}$.



+
−
$\kappa_\Gamma^{\mathtt{C}} > 0$
$\kappa_\Gamma^{\mathtt{C}} < 0$
s_η
s_ζ
+
−

Fig. 16: Positive and negative domain of lateral coordinate s_η.



$1/|\kappa_\Gamma^{\mathtt{C}}|$
$c_\Gamma^{\mathtt{C}}$
$\Gamma^{\mathtt{C}}$
φ
${}^{\mathtt{G}}c_k^{(1)}$
${}^{\mathtt{G}}c_k^{(3)}$
ϵ_ζ
s_η
ℓ
$1/|\kappa_\Gamma^{\mathtt{C}}|$
$1/|\kappa_\Gamma^{\mathtt{C}}| - s_\eta$
$\Gamma^{\mathtt{C}}$
φ
${}^{\mathtt{G}}c_k^{(1)}$
ϵ_η
${}^{\mathtt{G}}c_k^{(3)}$
s_η
ℓ

(a) Longitudinal enlargement ϵ_ζ. (b) Lateral enlargement ϵ_η.

Fig. 17: Computation of longitudinal and lateral enlargements.

and $\overline{s}_{\eta,k}^{\mathtt{rprt}(q)} = \sup\left\{\mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})\right\}$. The minimum and maximum curvature of $\Gamma(s_\zeta)$ are denoted by $\underline{\kappa}_\Gamma$ and $\overline{\kappa}_\Gamma$. We assume that we can locally approximate the reference path $\Gamma(s_\zeta)$ with a circle $\Gamma^{\mathtt{C}}$ in proximity of $\mathcal{D}_k^{\mathtt{rprt}(q)}$ as shown in Fig. 15. The center, curvature, and radius of $\Gamma^{\mathtt{C}}$ are denoted as $c_\Gamma^{\mathtt{C}}$, $\kappa_\Gamma^{\mathtt{C}}$, and $r_\Gamma^{\mathtt{C}} = 1/|\kappa_\Gamma^{\mathtt{C}}|$, respectively. For $\kappa_\Gamma^{\mathtt{C}} \geq 0$, the lateral coordinates s_η are positive on the inner side of the curve (see Fig. 16) and for $\kappa_\Gamma^{\mathtt{C}} < 0$, the lateral coordinates s_η are negative on the inner side of the curve (see Fig. 16). Based on $\Gamma^{\mathtt{C}}$ and $\mathcal{D}_k^{\mathtt{rprt}(q)}$, we compute $\overline{\epsilon}_\zeta$, $\overline{\epsilon}_\eta$ and $\underline{\epsilon}_\eta$ to determine $\mathcal{A}^\epsilon$. We further assume that $\mathcal{D}_k^{\mathtt{rprt}(q)} \oplus \mathcal{A}^\epsilon$ lies completely inside the unique projection domain of $\Gamma(s_\zeta)$ [75].

B. Longitudinal Enlargement

We wish to enclose all longitudinal positions of centers ${}^{\mathtt{L}}c_k^{(i)}$, $i \in \{1,2,3\}$, for all ${}^{\mathtt{L}}c_k^{(1)} \in \mathcal{D}_k^{\mathtt{rprt}(q)}$ using $\mathcal{A}^\epsilon$. First, we derive the function $\epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ for computing the longitudinal enlargement based on a single position ${}^{\mathtt{L}}c_k^{(1)}$. Second, we determine the over-approximative longitudinal enlargement $\overline{\epsilon}_\zeta$ for a set of positions based on $\epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$.

1) Single Position: Given ${}^{\mathtt{L}}c_k^{(1)}$, we aim to determine the longitudinal position coordinate $\hat{s}_{\zeta,k}$ of ${}^{\mathtt{L}}c_k^{(3)}$ relative to ${}^{\mathtt{L}}c_k^{(1)}$, such that $\hat{s}_{\zeta,k} = s_{\zeta,k} + \epsilon_\zeta$. Note that we are interested in the maximum longitudinal enlargement, therefore, it is sufficient to only consider ${}^{\mathtt{L}}c_k^{(3)}$ (${}^{\mathtt{L}}c_k^{(3)}$ is farther away from ${}^{\mathtt{L}}c_k^{(1)}$ than ${}^{\mathtt{L}}c_k^{(2)}$). As illustrated in Fig. 17a, ϵ_ζ is the arc length of a circle sector with radius $r_\Gamma^{\mathtt{C}} = 1/|\kappa_\Gamma^{\mathtt{C}}|$ and central angle φ:

$$\epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) = \frac{\varphi(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})}{|\kappa_\Gamma^{\mathtt{C}}|}, \tag{11}$$

where the central angle is computed as follows:

$$\varphi(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) = \begin{cases} \arctan\left(\frac{\ell}{1/|\kappa_\Gamma^{\mathtt{C}}| - s_{\eta,k}}\right) & \text{for } \kappa_\Gamma^{\mathtt{C}} \geq 0 \\ \arctan\left(\frac{\ell}{1/|\kappa_\Gamma^{\mathtt{C}}| + s_{\eta,k}}\right) & \text{for } \kappa_\Gamma^{\mathtt{C}} < 0. \end{cases} \tag{12}$$

As the lateral coordinates are positive on the inner side of the curve for $\kappa_\Gamma^{\mathtt{C}} > 0$ (see Fig. 16), we need to subtract $s_{\eta,k}$ from $1/|\kappa_\Gamma^{\mathtt{C}}|$ to compute $\varphi(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ (see Fig. 17a). For $\kappa_\Gamma^{\mathtt{C}} < 0$, the lateral coordinates are negative on the inner side of the curve. Therefore, we add $s_{\eta,k}$ to $1/|\kappa_\Gamma^{\mathtt{C}}|$ for computing $\varphi(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$. Note that $\lim\limits_{|\kappa_\Gamma^{\mathtt{C}}| \to 0} \epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) \to \ell$.

2) Set of Positions: For computational efficiency, we over-approximate (11) for a set of positions $\mathcal{D}_k^{\mathtt{rprt}(q)}$. Thus, the question arises, for which $s_{\eta,k} \in \mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})$ and $\kappa_\Gamma^{\mathtt{C}} \in [\underline{\kappa}_\Gamma, \overline{\kappa}_\Gamma]$ does the longitudinal enlargement (11) reach its maximum.

Lemma 1 *For $\kappa_\Gamma^{\mathtt{C}} \geq 0$, $\epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ (11) is monotonically increasing in $s_{\eta,k}$. Furthermore, for $\kappa_\Gamma^{\mathtt{C}} < 0$, $\epsilon_\zeta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ (11) is monotonically decreasing in $s_{\eta,k}$.*

Proof 1 *Given that $s'_{\eta,k} > s^*_{\eta,k}$ and $\kappa_\Gamma^{\mathtt{C}} \geq 0$, we have:*

$$\begin{gathered} \epsilon_\zeta(s'_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) > \epsilon_\zeta(s^*_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) \\ \arctan\left(\frac{\ell}{1/|\kappa_\Gamma^{\mathtt{C}}| - s'_{\eta,k}}\right) > \arctan\left(\frac{\ell}{1/|\kappa_\Gamma^{\mathtt{C}}| - s^*_{\eta,k}}\right) \\ 1/|\kappa_\Gamma^{\mathtt{C}}| - s'_{\eta,k} < 1/|\kappa_\Gamma^{\mathtt{C}}| - s^*_{\eta,k} \\ s'_{\eta,k} > s^*_{\eta,k}. \end{gathered}$$

Using the same reasoning, we obtain for $\kappa_\Gamma^{\mathtt{C}} < 0$ that $\epsilon_\zeta(s'_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) < \epsilon_\zeta(s^*_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$. □

Lemma 2 *For $\kappa_\Gamma^{\mathtt{C}} \geq 0$ and $(s_{\zeta,k}, s_{\eta,k}) \in \mathcal{D}_k^{\mathtt{rprt}(q)}$, (11) reaches its maximum for $s_{\eta,k} = \overline{s}_{\eta,k}^{\mathtt{rprt}(q)}$. For $\kappa_\Gamma^{\mathtt{C}} < 0$, (11) reaches its maximum for $s_{\eta,k} = \underline{s}_{\eta,k}^{\mathtt{rprt}(q)}$.*

Proof 2 *From Lemma 1, it follows that (11) is monotonically increasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} \geq 0$. Therefore, we select $\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}$ which is the maximum lateral position in $\mathcal{D}_k^{\mathtt{rprt}(q)}$ to compute (11). Similarly, as (11) is monotonically decreasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} < 0$, we select the minimum lateral coordinate $\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}$ of $\mathcal{D}_k^{\mathtt{rprt}(q)}$ to compute (11).* □

Lemma 3 *For any $s_{\eta,k} \in \mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})$ and $\kappa_\Gamma^{\mathtt{C}} \in [\underline{\kappa}_\Gamma, \overline{\kappa}_\Gamma]$, (11) is bounded by*

$$\overline{\epsilon}_\zeta = \sup\left\{\epsilon_\zeta\left(\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}, [0, \overline{\kappa}_\Gamma]\right) \cup \epsilon_\zeta\left(\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}, [\underline{\kappa}_\Gamma, 0]\right)\right\}. \tag{13}$$

Proof 3 *According to Lemma 2, $\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}$ and $\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}$ maximize (11) for $\kappa_\Gamma^{\mathtt{C}} \geq 0$ and $\kappa_\Gamma^{\mathtt{C}} < 0$, respectively. In contrast to $s_{\eta,k}$, (11) is not monotonic in $\kappa_\Gamma^{\mathtt{C}}$. Using interval arithmetics [76], we obtain the upper bound on (11) over the intervals $[0, \overline{\kappa}_\Gamma]$ and $[\underline{\kappa}_\Gamma, 0]$.* □

As (13) becomes increasingly over-approximative for larger intervals of curvatures, we pre-compute (13) for smaller intervals and store the values in a look-up table generated offline.

C. Lateral Enlargement

Similarly to the longitudinal enlargement, we wish to enlarge $\mathcal{D}_k^{\mathtt{rprt}(q)}$ in the lateral direction such that all lateral positions of centers ${}^{\mathtt{L}}c_k^{(i)}$, $i \in \{1,2,3\}$, are enclosed for all

${}^{\mathtt{L}}c_k^{(1)} \in \mathcal{D}_k^{\mathtt{rprt}(q)}$. Similarly to the previous section, we first derive the function $\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ for determining the lateral enlargements based on a single position ${}^{\mathtt{L}}c_k^{(1)}$. Second, we determine the over-approximative lateral enlargements $\underline{\epsilon}_\eta$ and $\overline{\epsilon}_\eta$ based on $\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$.

1) Single Position: Given ${}^{\mathtt{L}}c_k^{(1)}$, we aim to determine the lateral position coordinate $\hat{s}_{\eta,k}$ of ${}^{\mathtt{L}}c_k^{(3)}$ relative to ${}^{\mathtt{L}}c_k^{(1)}$, such that $\hat{s}_{\eta,k} = s_{\eta,k} - \epsilon_\eta$ (again, it is sufficient to only compute it for ${}^{\mathtt{L}}c_k^{(3)}$). As illustrated in Fig. 17b, we use the Pythagorean theorem to determine ϵ_η:

$$\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) = \begin{cases} \sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| - s_{\eta,k})^2 + \ell^2} \\ -(1/|\kappa_\Gamma^{\mathtt{C}}| - s_{\eta,k}) \end{cases}, \text{ for } \kappa_\Gamma^{\mathtt{C}} \geq 0, \quad \begin{cases} \sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| + s_{\eta,k})^2 + \ell^2} \\ -(1/|\kappa_\Gamma^{\mathtt{C}}| + s_{\eta,k}) \end{cases}, \text{ for } \kappa_\Gamma^{\mathtt{C}} < 0. \tag{14}$$

As the sign of the lateral coordinate s_η changes on the inner side of the curve depending on the sign of $\kappa_\Gamma^{\mathtt{C}}$ (see Fig. 16), we distinguish two cases for (14) similar to (12). Note that $\lim\limits_{|\kappa_\Gamma^{\mathtt{C}}| \to 0^+} \epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) \to 0$.

2) Set of Positions: For computational efficiency, we over-approximate (14) for a set of positions $\mathcal{D}_k^{\mathtt{rprt}(q)}$. Thus, the question arises for which $s_{\eta,k} \in \mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})$ and $\kappa_\Gamma^{\mathtt{C}} \in [\underline{\kappa}_\Gamma, \overline{\kappa}_\Gamma]$ the lateral enlargement (14) reaches its maximum.

Lemma 4 *For $\kappa_\Gamma^{\mathtt{C}} \geq 0$, $\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ (14) is monotonically increasing in $s_{\eta,k}$. Furthermore, for $\kappa_\Gamma^{\mathtt{C}} < 0$, $\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ (14) is monotonically decreasing in $s_{\eta,k}$.*

Proof 4 *The partial derivative of (14) for $\kappa_\Gamma^{\mathtt{C}} \geq 0$ with respect to $s_{\eta,k}$ is*

$$\frac{\partial \epsilon_\eta}{\partial s_{\eta,k}}(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) = \begin{cases} +1 - \frac{1/|\kappa_\Gamma^{\mathtt{C}}| - s_{\eta,k}}{\sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| - s_{\eta,k})^2 + \ell^2}}, & \text{for } \kappa_\Gamma^{\mathtt{C}} \geq 0, \\ -1 + \frac{1/|\kappa_\Gamma^{\mathtt{C}}| + s_{\eta,k}}{\sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| + s_{\eta,k})^2 + \ell^2}}, & \text{for } \kappa_\Gamma^{\mathtt{C}} < 0. \end{cases} \tag{15}$$

Since $1/|\kappa_\Gamma^{\mathtt{C}}| \mp s_{\eta,k} > 0$ must hold such that $s_{\eta,k}$ is within the unique projection domain of $\Gamma^{\mathtt{C}}$ and $1/|\kappa_\Gamma^{\mathtt{C}}| \mp s_{\eta,k} < \sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| \mp s_{\eta,k})^2 + \ell^2}$, we have:

$$0 < \frac{1/|\kappa_\Gamma^{\mathtt{C}}| \mp s_{\eta,k}}{\sqrt{(1/|\kappa_\Gamma^{\mathtt{C}}| \mp s_{\eta,k})^2 + \ell^2}} < 1.$$

Thus, it holds that

$$\forall \kappa_\Gamma^{\mathtt{C}} \geq 0 : \frac{\partial \epsilon_\eta}{\partial s_{\eta,k}}(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) > 0,$$
$$\forall \kappa_\Gamma^{\mathtt{C}} < 0 : \frac{\partial \epsilon_\eta}{\partial s_{\eta,k}}(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}}) < 0.$$

Therefore, (14) is monotonically increasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} \geq 0$ and monotonically decreasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} < 0$. □

Lemma 5 *For $\kappa_\Gamma^{\mathtt{C}} \geq 0$ and $(s_{\zeta,k}, s_{\eta,k}) \in \mathcal{D}_k^{\mathtt{rprt}(q)}$, (14) reaches its maximum for $s_{\eta,k} = \overline{s}_{\eta,k}^{\mathtt{rprt}(q)}$. For $\kappa_\Gamma^{\mathtt{C}} < 0$, (14) reaches its maximum for $s_{\eta,k} = \underline{s}_{\eta,k}^{\mathtt{rprt}(q)}$.*

Proof 5 *From Lemma 4, it follows that (14) is monotonically increasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} \geq 0$. Therefore, we select $\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}$, which is the maximum lateral position in $\mathcal{D}_k^{\mathtt{rprt}(q)}$, for computing (14). Similarly, as (14) is monotonically decreasing in $s_{\eta,k}$ for $\kappa_\Gamma^{\mathtt{C}} < 0$, we select the minimum lateral coordinate $\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}$ of $\mathcal{D}_k^{\mathtt{rprt}(q)}$ to compute (14).* □

The remaining task is to determine the curvature $\kappa_\Gamma^{\mathtt{C}}$, for which $\epsilon_\eta(s_{\eta,k}, \kappa_\Gamma^{\mathtt{C}})$ reaches its maximum for a given $\mathcal{D}_k^{\mathtt{rprt}(q)}$ and reference path $\Gamma(s_\zeta)$.

Lemma 6 *(14) is monotonically increasing in $|\kappa_\Gamma^{\mathtt{C}}|$.*

Proof 6 *We substitute $1/|\kappa_\Gamma^{\mathtt{C}}|$ with $r_\Gamma^{\mathtt{C}}$ in (14) and compute the partial derivative of (14) with respect to $r_\Gamma^{\mathtt{C}}$:*

$$\frac{\partial \epsilon_\eta}{\partial r_\Gamma^{\mathtt{C}}}(s_{\eta,k}, r_\Gamma^{\mathtt{C}}) = \begin{cases} -1 + \frac{r_\Gamma^{\mathtt{C}} - s_{\eta,k}}{\sqrt{(r_\Gamma^{\mathtt{C}} - s_{\eta,k})^2 + \ell^2}}, & \text{for } \kappa_\Gamma^{\mathtt{C}} \geq 0, \\ -1 + \frac{r_\Gamma^{\mathtt{C}} + s_{\eta,k}}{\sqrt{(r_\Gamma^{\mathtt{C}} + s_{\eta,k})^2 + \ell^2}}, & \text{for } \kappa_\Gamma^{\mathtt{C}} < 0. \end{cases} \tag{16}$$

Since $\frac{\partial \epsilon_\eta}{\partial r_\Gamma^{\mathtt{C}}}(s_{\eta,k}, r_\Gamma^{\mathtt{C}}) < 0$, (14) is monotonically decreasing in $r_\Gamma^{\mathtt{C}}$. As a result, (14) is monotonically increasing in $|\kappa_\Gamma^{\mathtt{C}}|$, because $|\kappa_\Gamma^{\mathtt{C}}|$ is the reciprocal of $r_\Gamma^{\mathtt{C}}$. □

Lemma 7 *For any $s_{\eta,k} \in \mathrm{proj}_{s_\eta}(\mathcal{D}_k^{\mathtt{rprt}(q)})$ and $\kappa_\Gamma^{\mathtt{C}} \in [\underline{\kappa}_\Gamma, \overline{\kappa}_\Gamma]$, (14) is bounded by*

$$\underline{\epsilon}_\eta = \begin{cases} \epsilon_\eta\left(\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}, \overline{\kappa}_\Gamma\right), & \text{for } \overline{\kappa}_\Gamma \geq 0, \\ 0 & \text{for } \overline{\kappa}_\Gamma < 0, \end{cases} \tag{17}$$

$$\overline{\epsilon}_\eta = \begin{cases} \epsilon_\eta\left(\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}, \underline{\kappa}_\Gamma\right), & \text{for } \underline{\kappa}_\Gamma < 0, \\ 0 & \text{for } \underline{\kappa}_\Gamma \geq 0. \end{cases} \tag{18}$$

Proof 7 *Since a straight line in the curvilinear coordinate system $F^{\mathtt{L}}$ is bent toward the inner side of the circle $\Gamma^{\mathtt{C}}$ when it is transformed to the Cartesian frame $F^{\mathtt{G}}$, we need to enlarge $\mathcal{D}_k^{\mathtt{rprt}(q)}$ only to the circle's outer side. For $\kappa_\Gamma^{\mathtt{C}} \geq 0$, we enlarge $\mathcal{D}_k^{\mathtt{rprt}(q)}$ in the negative η-direction with $\underline{\epsilon}_\eta = \epsilon_\eta\left(\overline{s}_{\eta,k}^{\mathtt{rprt}(q)}, \overline{\kappa}_\Gamma\right)$ (see Lemma 5 and 6). For $\kappa_\Gamma^{\mathtt{C}} < 0$, we enlarge $\mathcal{D}_k^{\mathtt{rprt}(q)}$ in the positive η-direction with $\overline{\epsilon}_\eta = \epsilon_\eta\left(\underline{s}_{\eta,k}^{\mathtt{rprt}(q)}, \underline{\kappa}_\Gamma\right)$ (see Lemma 5 and 6).* □

Theorem 1 *By computing $\mathcal{A}^\epsilon$ according to (13), (17), and (18), Problem 1 is solved.*

Proof 8 *The proof follows directly from Lemma 3 and 7.* □

Acknowledgment

The authors thank BMW for supporting our driving experiments, particularly, Dr. Sebastian Gnatzig and Dr. Tobias Rehder for their assistance during our experiments. Moreover, the authors thank Vitaliy Rusinov for his contribution to fast collision checking in the reachability analysis.

References

[1] S. Bhattacharya and R. Ghrist, "Path homotopy invariants and their application to optimal trajectory planning," *Annals of Mathematics and Artificial Intelligence*, vol. 84, pp. 139–160, 2018.

[2] J. Park, S. Karumanchi, and K. Iagnemma, "Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1101–1115, 2015.

[3] A. Varava, K. Hang, D. Kragic, and F. T. Pokorny, "Herding by caging: A topological approach towards guiding moving agents via mobile robots," in *Robotics: Science and Systems*, 2017, pp. 1–9.

[4] S. Bhattacharya, V. Kumar, and M. Likhachev, "Search-based path planning with homotopy class constraints," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2010, pp. 1230–1237.

[5] P. Bender, Ö. S. Taş, J. Ziegler, and C. Stiller, "The combinatorial aspect of motion planning: Maneuver variants in structured environments." in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2015, pp. 1386–1392.

[6] C. Miller, C. Pek, and M. Althoff, "Efficient mixed-integer programming for longitudinal and lateral motion planning of autonomous vehicles," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018, pp. 1954–1961.

[7] R. M. Karp, "On the computational complexity of combinatorial problems," *Networks*, vol. 5, no. 1, pp. 45–68, 1975.

[8] M. Althoff, "Reachability analysis of nonlinear systems using conservative polynomialization and non-convex sets," in *Hybrid Systems: Computation and Control*, 2013, pp. 173–182.

[9] S. Söntges and M. Althoff, "Computing the drivable area of autonomous road vehicles in dynamic road scenes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1855–1866, 2018.

[10] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[11] 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," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[12] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[13] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," in *Proc. of the American Control Conference*, 2001, pp. 43–49.

[14] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[15] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *Int. Journal of Robotics Research*, vol. 31, no. 3, pp. 346–359, 2012.

[16] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2009, pp. 1879–1884.

[17] F. von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H.-J. Wuensche, "Driving with tentacles: Integral structures for sensing and motion," *Journal of Field Robotics*, vol. 25, no. 9, pp. 640–673, 2008.

[18] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 4889–4895.

[19] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 489–494.

[20] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proc. of the IEEE European Control Conference*, 2001, pp. 2603–2608.

[21] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha – a local, continuous method," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2014, pp. 450–457.

[22] F. Molinari, Nguyen Ngoc Anh, and L. Del Re, "Efficient mixed integer programming for autonomous overtaking," in *Proc. of the American Control Conference*, 2017, pp. 2303–2308.

[23] B. Yi, S. Gottschling, J. Ferdinand, N. Simm, F. Bonarens, and C. Stiller, "Real time integrated vehicle dynamics control and trajectory planning with MPC for critical maneuvers," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 584–589.

[24] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[25] D. P. Bertsekas, *Nonlinear Programming*, ser. Athena scientific optimization and computation series. Athena Scientific, 2016.

[26] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *Proc. of the IEEE European Control Conference*, 2013, pp. 3071–3076.

[27] J. Nilsson, M. Brännström, J. Fredriksson, and E. Coelingh, "Longitudinal and lateral control for automated yielding maneuvers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1404–1414, 2016.

[28] W. Zhan, J. Chen, C.-Y. Chan, C. Liu, and M. Tomizuka, "Spatially-partitioned environmental representation and planning architecture for on-road autonomous driving," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 632–639.

[29] B. Gutjahr, L. Gröll, and M. Werling, "Lateral vehicle trajectory optimization using constrained linear time-varying MPC," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1586–1595, 2017.

[30] C. Pek and M. Althoff, "Computationally efficient fail-safe trajectory planning for self-driving vehicles using convex optimization," in *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, 2018, pp. 1447–1454.

[31] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 3530–3538.

[32] M. Riedmiller, M. Montemerlo, and H. Dahlkamp, "Learning to drive a real car in 20 minutes," in *Proc. of the IEEE Int. Conf. on Frontiers in the Convergence of Bioscience and Information Technologies*, 2007, pp. 645–650.

[33] P. Wolf, K. Kurzer, T. Wingert, F. Kuhnt, and J. M. Zöllner, "Adaptive behavior generation for autonomous driving using deep reinforcement learning with compact semantic states," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018, pp. 993–1000.

[34] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, pp. 1–13, 2016.

[35] S. A. Seshia, D. Sadigh, and S. S. Sastry, "Towards verified artificial intelligence," *arXiv preprint arXiv:1606.08514*, pp. 1–13, 2017.

[36] T. Gu, J. M. Dolan, and J. Lee, "Automated tactical maneuver discovery, reasoning and trajectory planning for autonomous driving," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 5474–5480.

[37] J. Schlechtriemen, K. P. Wabersich, and K.-D. Kuhnert, "Wiggling through complex traffic: Planning trajectories constrained by predictions," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 1293–1300.

[38] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. H. Reif, "Narrow passage sampling for probabilistic roadmap planning," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1105–1115, 2005.

[39] H. Mohy-ud-Din and A. Muhammad, "Detecting narrow passages in configuration spaces via spectra of probabilistic roadmaps," in *Proc. of the ACM Symposium on Applied Computing*, 2010, pp. 1294–1298.

[40] K. Esterle, P. Hart, J. Bernhard, and A. Knoll, "Spatiotemporal motion planning with combinatorial reasoning for autonomous driving," in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2018, pp. 1053–1060.

[41] Q. H. Do, S. Mita, and K. Yoneda, "Narrow passage path planning using fast marching method and support vector machine," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2014, pp. 630–635.

[42] H. Liu, F. Xiao, and C. Wang, "A predictive model for narrow passage path planner by using support vector machine in changing environments," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 2991–2996.

[43] M. Buechel, G. Hinz, F. Ruehl, H. Schroth, C. Gyoeri, and A. Knoll, "Ontology-based traffic scene modeling, traffic regulations dependent situational awareness and decision-making for automated vehicles," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 1471–1476.

[44] R. Kohlhaas, D. Hammann, T. Schamm, and J. M. Zöllner, "Planning of high-level maneuver sequences on semantic state spaces," in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2015, pp. 2090–2096.

[45] L. Zhao, R. Ichise, T. Yoshikawa, T. Naito, T. Kakinami, and Y. Sasaki, "Ontology-based decision making on uncontrolled intersections and narrow roads," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2015, pp. 83–88.

[46] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knöppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, "Making Bertha drive – an autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[47] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski,

B. Salesky, Y. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, “Autonomous driving in urban environments: Boss and the Urban Challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[48] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebl, F. von Hundelshausen, O. Pink, C. Frese, and C. Stiller, “Team AnnieWAY’s autonomous system for the 2007 DARPA Urban Challenge,” *Journal of Field Robotics*, vol. 25, no. 9, pp. 615–639, 2008.

[49] A. Furda and L. Vlacic, “Enabling safe autonomous driving in real-world city traffic using multiple criteria decision making,” *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 1, pp. 4–17, 2011.

[50] S. Ulbrich and M. Maurer, “Probabilistic online POMDP decision making for lane changes in fully automated driving,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2013, pp. 2063–2067.

[51] C. Hubmann, J. Schulz, G. Xu, D. Althoff, and C. Stiller, “A belief state planner for interactive merge maneuvers in congested traffic,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2018, pp. 1617–1624.

[52] S. Brechtel, T. Gindele, and R. Dillmann, “Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2014, pp. 392–399.

[53] D. Lenz, T. Kessler, and A. Knoll, “Tactical cooperative planning for autonomous highway driving using Monte-Carlo Tree Search,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 447–453.

[54] S. Söntges and M. Althoff, “Computing possible driving corridors for automated vehicles,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 160–166.

[55] H. Ahn, K. Berntorp, and S. Di Cairano, “Reachability-based decision making for city driving,” in *Proc. of the American Control Conference*, 2018, pp. 3203–3208.

[56] A. Shkolnik, M. Walter, and R. Tedrake, “Reachability-guided sampling for planning under differential constraints,” in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2009, pp. 2859–2865.

[57] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan, “Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots,” *arXiv preprint arXiv:1809.06746*, pp. 1–58, 2018.

[58] M. Gerdts and I. Xausa, “Avoidance trajectories using reachable sets and parametric sensitivity analysis,” in *IFIP Conf. on System Modeling and Optimization*, 2013, pp. 491–500.

[59] B. Schürmann, D. Heß, J. Eilbrecht, O. Stursberg, F. Köster, and M. Althoff, “Ensuring drivability of planned motions using formal methods,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2017, pp. 1–8.

[60] M. Althoff and J. M. Dolan, “Online verification of automated road vehicles using reachability analysis,” *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[61] E. Héry, S. Masi, P. Xu, and P. Bonnifait, “Map-based curvilinear coordinates for autonomous vehicles,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2017, pp. 1–7.

[62] G. Lafferriere, G. J. Pappas, and S. Yovine, “Symbolic reachability computation for families of linear vector fields,” *Journal of Symbolic Computation*, vol. 32, no. 3, pp. 231 – 253, 2001.

[63] J. Ziegler and C. Stiller, “Fast collision checking for intelligent vehicle motion planning,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2010, pp. 518–522.

[64] H. T. Croft, K. Falconer, and R. K. Guy, *Unsolved problems in geometry: unsolved problems in intuitive mathematics*. Springer Science & Business Media New York, 1991, vol. 2.

[65] S. Magdici and M. Althoff, “Fail-safe motion planning of autonomous vehicles,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2016, pp. 452–458.

[66] W. Lipski and F. P. Preparata, “Finding the contour of a union of iso-oriented rectangles,” *Journal of Algorithms*, vol. 1, no. 3, pp. 235 – 246, 1980.

[67] H. Edelsbrunner, J. Van Leeuwen, T. Ottmann, and D. Wood, “Computing the connected components of simple rectilinear geometrical objects in d-space,” *RAIRO, Informatique théorique*, vol. 18, no. 2, pp. 171–183, 1984.

[68] M. Althoff, M. Koschi, and S. Manzinger, “CommonRoad: Composable benchmarks for motion planning on roads,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 719–726.

[69] M. Koschi and M. Althoff, “Set-based prediction of traffic participants considering occlusions and traffic rules,” *IEEE Transactions on Intelligent Vehicles*, 2020, [in press].

[70] X. Qian, F. Altché, P. Bender, C. Stiller, and A. de La Fortelle, “Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2016, pp. 205–210.

[71] C. Burger and M. Lauer, “Cooperative multiple vehicle trajectory planning using MIQP,” in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2018, pp. 602–607.

[72] J. Eilbrecht and O. Stursberg, “Cooperative driving using a hierarchy of mixed-integer programming and tracking control,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 673–678.

[73] Gurobi Optimization, LLC, “Gurobi optimizer reference manual,” 2020. [Online]. Available: http://www.gurobi.com

[74] J. Dahl, G. R. de Campos, C. Olsson, and J. Fredriksson, “Collision avoidance: A literature review on threat-assessment techniques,” *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 1, pp. 101–113, 2019.

[75] J. Pegna and F.-E. Wolter, “Surface curve design by orthogonal projection of space curves onto free-form surfaces,” *Journal of Mechanical Design*, vol. 118, no. 1, pp. 45–52, 1996.

[76] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to interval analysis*. Society for Industrial and Applied Mathematics, 2009.

Stefanie Manzinger is currently a Ph.D. candidate and joined the Cyber-Physical Systems Group at the Technical University of Munich under Prof. Dr.-Ing. Matthias Althoff in 2016. She received her B.Sc. degree in Mechatronics and Information Technology in 2014 and her M.Sc. degree in Robotics, Cognition, Intelligence in 2016, both from the Technical University of Munich, Germany. Her research interests include automated vehicles, reachability analysis, and motion planning.

Christian Pek is a postdoctoral researcher in the Division of Robotics, Perception and Learning at KTH Royal Institute of Technology. Before joining KTH, he was a PhD student in the Cyber-Physical Systems Group at the Technical University of Munich under Prof. Dr.-Ing. Matthias Althoff. He was a research assistant in the motion planning group at BMW Group from 2015 until 2019. Christian graduated with the Master of Science degree in computer science and robotics from the Technical University of Braunschweig, Germany, and the University of Auckland, New Zealand, in 2015. His vision is a future of robots which robustly and safely accomplish tasks with and around humans.

Matthias Althoff is an associate professor in computer science at the Technical University of Munich, Germany. He received his diploma engineering degree in Mechanical Engineering in 2005, and his Ph.D. degree in Electrical Engineering in 2010, both from the Technical University of Munich, Germany. From 2010 to 2012 he was a postdoctoral researcher at Carnegie Mellon University, Pittsburgh, USA, and from 2012 to 2013 an assistant professor at Ilmenau University of Technology, Germany. His research interests include formal verification of continuous and hybrid systems, reachability analysis, planning algorithms, nonlinear control, automated vehicles, and power systems.