Computing the Drivable Area of Autonomous Road Vehicles in Dynamic Road Scenes

Sebastian Söntges and Matthias Althoff

***Abstract*— This paper presents an algorithm for overapproximating the drivable area of road vehicles in the presence of time-varying obstacles. The drivable area can be used to detect whether a feasible trajectory exists and in which area one can limit the search of drivable trajectories. For this purpose, we abstract the considered road vehicle by a point mass with bounded velocity and acceleration. Our algorithm calculates the reachable occupancy at discrete time steps. At each time step, the set is represented by a union of finitely many sets, which are each the Cartesian product of two 2-D convex polytopes. We demonstrate our method with three examples: i) a traffic situation with identical dynamic constraints in the x- and y-directions; ii) a highway scenario with different lateral and longitudinal constraints of the dynamics; and iii) a highway scenario with different traffic predictions. The examples demonstrate that we can compute the drivable area quickly enough to deploy our approach in real vehicles.**

***Index Terms*— Autonomous cars, motion prediction, road vehicle safety, reachable set.**

I. Introduction

HIGHLY autonomous cars and advanced driver assistance systems provide a great opportunity for increasing safety and comfort in transportation. These systems make decisions automatically and choose the driving path themselves without relying on a human driver. However, the decision making and trajectory generation still pose a major challenge. Usually, the planning yields high-dimensional search problems, which must be solved with limited computational resources and hard real-time constraints. These limitations demand appropriate heuristics to accelerate the search and the need for anytime algorithms which are able to produce solutions in a timely manner.

In this work, we present a method for computing an overapproximation of the set of all possible trajectories of an automated vehicle. Our method constructs the set of all states that can be reached by a vehicle considering speed and acceleration bounds while moving in a two-dimensional plane with static and dynamic obstacles.

One basic task of an automated vehicle is the trajectory generation from the current position to some goal position without causing any collision. A popular approach to generate trajectories is to cast the trajectory generation problem into a graph search problem [1]. A set of states in the continuous state space of the vehicle is selected as nodes of a graph. Trajectories connecting these states are added as edges in the graph. Depending on the motion model, the trajectories are solutions to a boundary value problem of open or closed-loop dynamical systems, kinematic models or purely geometric curves. The nodes can be selected in a random fashion [2] (e.g. RRT) or according to an appropriate deterministic scheme [3], [4] (e.g. along the lane). Selecting states and trajectories tailored towards the specific driving task, the structure in the environment and the traffic situation can reduce the size of the search graph [5]. A trajectory is found when a path in the graph is found that connects the initial node to one in the goal region. One major drawback of graph-based approaches is that the size of the graph scales exponentially with respect to the number of search dimensions. This may quickly lead to restrictive, long search times. Suppose a vehicle can move in a two-dimensional plane with time-dependent obstacles. A vehicle model with two position variables, two velocity variables and one time variable yields a 5-dimensional search space. An exhaustive search for an appropriate, dense selection of trajectories is therefore difficult. An early approach for reducing complexity involves decomposing the trajectory generation to a path planning problem in a static environment and a velocity planning problem in path-time space [6]. In dynamic scenarios, this decomposition can be too restrictive. A more general approach is to use heuristics to reduce the searched subspace of the graph [7] and to accept suboptimal solutions [8]. Besides graph-based approaches, other methods such as artificial potential fields [9] and cell decompositions [10] are proposed to generate collision free trajectories for automated vehicles.

Our method calculates the possible search space under consideration of static and dynamic obstacles and thus significantly prunes the search space of the search methods reviewed above. The dynamic obstacles can be arbitrary, time-varying occupied regions. The computed set is an overapproximation of the set of all states which can be reached by the host vehicle. This set is often referred to as the reachable set. It is guaranteed that all solution trajectories lie within this set. All trajectories leaving the reachable set during planning can be proven to either violate the constraints on the dynamics or eventually collide. A related approach considering sets of reachable states instead of trajectories is presented in [11] which computes the backward reachable set of a goal region using a method based on the Hamilton-Jacobi-Bellman equation and involves solving partial differential equations.

Manuscript received July 5, 2016; revised April 3, 2017 and July 14, 2017; accepted August 5, 2017. Date of publication September 20, 2017; date of current version May 29, 2018. This work was supported by the German Research Foundation Grants AL 1185/3-1 and Graduiertenkolleg 1480 (PUMA). The Associate Editor for this paper was H. Van Lint. *(Corresponding author: Sebastian Söntges.)*

The authors are with the Department of Computer Science, Technische Universität München, 85748 Garching, Germany (e-mail: soentges@in.tum.de; matthias.althoff@tum.de).

Digital Object Identifier 10.1109/TITS.2017.2742141

An approximation of the projected reachable set on a grid in spatial domain using optimal control is introduced in [12]. However, it is difficult to include arbitrary shaped time varying obstacles in this method. In [13] the set of reachable positions is represented as a time varying polygon but the velocity domain is neglected.

A second application of our method is to trigger driving assistant systems, in particular collision mitigation systems. In order to provide the driver with the possibility to avoid a collision as long as possible, most collision mitigation systems only intervene when a collision is no longer avoidable [13], [14]. Many approaches only consider sampled candidate trajectories, i.e. they check whether no safe solution from a set of finitely many solutions still exists. However, since infinitely many possible trajectories exist, approaches based on sampling cannot guarantee that a safe trajectory still exists and hence are only resolution or probabilistically complete. However, if our approach returns an empty drivable area, we can prove the non-existence of any evasive maneuver for the given traffic prediction and can appropriately trigger the collision mitigation system [15].

A third application of our method is to assess the risk of a traffic situation. Risk assessment has a huge impact on maneuver selection and planning strategies since risk is an essential part for determining optimal trajectories. An overview of risk assessment methods can be found in [16]. The drivable area of a vehicle as obtained from our work can be used to deduce risk measures, such as using the area of the drivable region as a measure to compare alternative planned trajectories.

The computation of the drivable area has a lot of resemblance with classical reachability analysis, but also significant differences, which require special choices in terms of the applied algorithms and the set representation. One main difference is that the computation of the drivable area requires set difference of the drivable area between the ego vehicle and other traffic participants, which results in non-convex and sometimes even non-connected sets. In classical reachability analysis, however, reachable sets are typically represented by connected and convex sets, including polytopes [17], zonotopes [18], rectangular grids [19], ellipsoids [20], support functions [21], oriented rectangular hulls [22], and axis-aligned boxes [23]. We therefore propose a set representation that borrows ideas from approximative rectangular cell decomposition in the position domain [24] to consider set differences, while we use convex polytopes in the velocity domain to faithfully consider the vehicle dynamics. A second main difference is for the application in road vehicles, the available computation time is strictly limited.

The presented algorithm is based on our previous work [15]. The novelty of this work is as follows:

- Results of preceding and subsequent time steps are used for the calculations of the reachable set. In our previous work, the algorithm uses information exclusively from the previous step to calculate the next step. In this work, the calculated set is tightened by including information from subsequent steps i.e. excluding states that eventually enter forbidden regions.

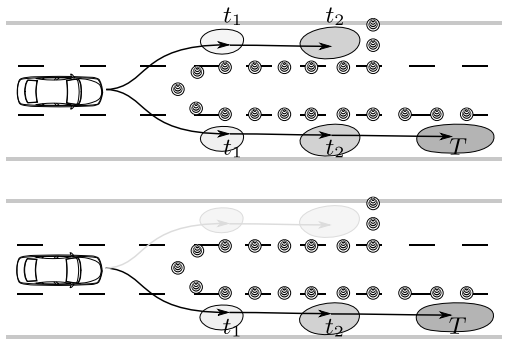

Fig. 1. Reachable set considering collision free trajectories until the given time step (top). Anticipated reachable set considering only collision free trajectories for the whole planning horizon $[t_0, T]$ (bottom).

- The set representation is refined. In our previous work, we approximate the reachable set at each time step by the union of four-dimensional intervals which may overlap. Here, we approximate the reachable set by an union of sets, of which each set is the Cartesian product of two convex 2-dimensional polytopes. The interior of the sets are disjunct and the boundaries may intersect. The algorithmic realization is more efficient despite improved accuracy.
- In contrast to our previous work, velocity limits are considered. This is particularly important when the chosen time horizon is so large, that otherwise meaningless speeds are reached.

II. Definitions and Problem Formulation

A. Reachable Set

Given an initial state s_0, an input signal $u(t)$ and the system dynamics $\dot{s}(t) = f(s(t), u(t))$, the trajectory of the state is:

$$s(t) = s_0 + \int_{t_0}^{t} f(s(\tau), \quad u(\tau))\ \mathrm{d}\tau.$$

We assume a set of forbidden states $\mathcal{F}(t)$ to be given. The reachable set $\mathbf{reach}(t; \mathcal{X}_0)$ is defined as the set of all states that can be reached from an initial set $\mathcal{X}_0$ at time t:

$$\mathbf{reach}(t; \mathcal{X}_0) = \{s(t; u, s_0) \,|\exists u \in \mathcal{U},\ \exists s_0 \in \mathcal{X}_0, \\ s(\tau; u, s_0) \notin \mathcal{F}(\tau) \text{ for } \tau \in [t_0, t]\} \quad (1)$$

where we use the notation $s(t; u, s_0)$ to refer to the trajectory with initial state s_0 and input $u(t)$. In the above definition, the reachable set is restricted to states which can only be reached without entering the forbidden states (see Fig. 1).

We additionally define the *anticipated* reachable set

$$\mathbf{reach}_{\text{ant}}(t; \mathcal{X}_0, T) = \{s(t; u, s_0) \,|\exists u \in \mathcal{U},\ \exists s_0 \in \mathcal{X}_0, \\ s(\tau; u, s_0) \notin \mathcal{F}(\tau) \text{ for } \tau \in [t_0, T]\} \quad (2)$$

where $t \in [t_0, T]$. This set is a set of states which can be reached at time t and their trajectories can be continued until time $T \geq t$ without entering the forbidden region (see Fig. 1). It holds that

$$\mathbf{reach}_{\text{ant}}(t; \mathcal{X}_0, T) \subseteq \mathbf{reach}(t; \mathcal{X}_0).$$

B. Problem Statement

We aim at calculating an overapproximation (i.e. a superset) of the anticipated reachable set of a vehicle for a given set of obstacles which corresponds to a forbidden region $\mathcal{F}(t)$. Common vehicle models $f(s,u)$ possess nonlinear behavior and several dimensions in the state space, which makes it difficult to calculate the reachable set efficiently. We overapproximate the vehicle behavior by a moving point mass with bounded acceleration and speed.

The system model used is an abstraction of a realistic vehicle. We motivate this through the following implication: Given a model M of a dynamical system, the model M_i is an abstraction of M if the reachable set contains the other reachable set, i.e. $\mathbf{reach}_M \subseteq \mathbf{reach}_{M_i}$. We assume the acceleration and speed to be bounded, which holds for all realistic vehicles. Therefore the reachable set of realistic vehicles is a subset of our computed reachable set.

The systems dynamics $f(s,u)$ of the abstract system is:

$$\begin{aligned}\begin{pmatrix}\dot{x}\\ \ddot{x}\\ \dot{y}\\ \ddot{y}\end{pmatrix} &= \begin{pmatrix}0&1&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0\end{pmatrix}\begin{pmatrix}x\\ \dot{x}\\ y\\ \dot{y}\end{pmatrix} + \begin{pmatrix}0&0\\1&0\\0&0\\0&1\end{pmatrix}\begin{pmatrix}u_x\\ u_y\end{pmatrix}\\ |u_x| &\leq a_{max,x}\\ |u_y| &\leq a_{max,y}\\ v_{min,x} &\leq \dot{x} \leq v_{max,x}\\ v_{min,y} &\leq \dot{y} \leq v_{max,y}\end{aligned} \tag{3}$$

The occupied region of the vehicle in the workspace is modeled by a disc $\mathcal{A}(s(t))$. We choose the diameter of the disc to be the diameter of the inner circle of the vehicle shape, since an underapproximation of the shape yields an overapproximation of the reachable set. The obstacle set $\mathcal{O}(t)$ is supposed to be given and defines the forbidden region of the ego vehicle. At all times in the planning horizon, the occupied region of the vehicle $\mathcal{A}(s(t))$ in the workspace at state $s(t)$ must not intersect with $\mathcal{O}(t)$:

$$\forall t:\ \mathcal{A}(s(t)) \cap \mathcal{O}(t) = \emptyset.$$

The forbidden region is deduced from the obstacles:

$$\mathcal{F}(t) = \{s(t)\,|\mathcal{A}(s(t)) \cap \mathcal{O}(t) \neq \emptyset\}.$$

In the case of no obstacles, there is a closed-form solution of the reachable set for our model (3). However, in the presence of arbitrary obstacles, the set can only be computed numerically since obstacles of arbitrary shape and arrangement interfere with the closed-form solution. The actual reachable set thus becomes a set that cannot be computed and represented efficiently. The objective of this paper is to present an algorithm that tightly overapproximates this set, i.e. all approximations we apply are strict in the sense that the actual reachable set is a subset.

III. Mathematical Modeling

The dynamics of a state along the x- and y-direction of the system (3) is modeled as independent and can be computed separately. However, to check whether a state lies in the forbidden region $\mathcal{F}$, the pair of positions (x,y) of a state must be considered jointly. In this section the mathematical solution for the reachable set of the one-dimensional motion along one direction is shown first. The reachable set of several of these one-dimensional motions are then merged in our proposed algorithm shaping an overapproximation of the reachable set of the joined two-dimensional motions.

A. Reachable Set of One-Dimensional Motion

We consider the subproblem of one-dimensional motion in (3), but initially neglect the velocity constraint. In addition, we assume that the system must pass at time $t_1, t_2, \ldots, t_n$ some position interval $I_1, I_2, \ldots, I_n$:

$$\begin{aligned}\begin{pmatrix}\dot{x}\\ \ddot{x}\end{pmatrix} &= \underbrace{\begin{pmatrix}0&1\\0&0\end{pmatrix}}_{A}\begin{pmatrix}x\\ \dot{x}\end{pmatrix} + \underbrace{\begin{pmatrix}0\\1\end{pmatrix}}_{B}u_x\\ |u_x| &\leq a_{max,x}\\ x(t_1) &\in I_1,\ x(t_2) \in I_2, \ldots,\ x(t_n) \in I_n\end{aligned} \tag{4}$$

Suppose the initial states are within a convex initial set $\mathcal{X}_0$ at $t_0 = 0$. The reachable set without any position constraint is given by

$$\begin{aligned}&\mathbf{reach}(t;\mathcal{X}_0)\\ &= \left\{s \,\middle|\, \exists s_0 \in \mathcal{X}_0, \exists u \in \mathcal{U}, s = e^{At}s_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)\mathrm{d}\tau\right\}\\ &= e^{At}\mathcal{X}_0 \oplus \underbrace{\left\{s \,\middle|\, \exists u \in \mathcal{U}, s = \int_0^t e^{A(t-\tau)}Bu(\tau)\mathrm{d}\tau\right\}}_{=:\mathcal{P}_{u,x}(t)}\\ &= e^{At}\mathcal{X}_0 \oplus \mathcal{P}_{u,x}(t)\end{aligned} \tag{5}$$

where $\oplus$ denotes the Minkowski sum, which is defined as:

$$e^{At}\mathcal{X}_0 \oplus \mathcal{P}_{u,x}(t) = \left\{a + b \,\middle|\, a \in e^{At}\mathcal{X}_0, b \in \mathcal{P}_{u,x}(t)\right\}.$$

$\mathcal{P}_{u,x}(t)$ is the set of all states that can be reached from an initial state $(x,\dot{x})^T = (0,0)^T$ for all possible acceleration constrained inputs. This region is bounded by the minimum and maximum speeds, that can be reached at each position after time t. Given a specific terminal position x_t at time t, the minimum/maximum speed at this position can be determined using optimal control theory. Pontryagin's principle yields a bang-bang input candidate function with switching time γt ($\gamma \in [0 \ldots 1]$).

The upper velocity bound (full braking until time γt, then full acceleration) is obtained by

$$\begin{aligned}x_t^{(h)}(\gamma) &= x_0 + \dot{x}_0\, t + a_0\, t^2\left(\frac{1}{2} - 2\gamma + \gamma^2\right)\\ \dot{x}_t^{(h)}(\gamma) &= \dot{x}_0 + a_{max,x}t\,(1 - 2\gamma)\end{aligned}$$

and the lower bound (full acceleration until time γt, then full braking) is obtained by

$$\begin{aligned}x_t^{(l)}(\gamma) &= x_0 + \dot{x}_0\, t - a_0\, t^2\left(\frac{1}{2} - 2\gamma + \gamma^2\right)\\ \dot{x}_t^{(l)}(\gamma) &= \dot{x}_0 - a_{max,x}t\,(1 - 2\gamma).\end{aligned}$$

The upper bound is concave and the lower bound convex. The intersection of both is convex (see Fig. 3).

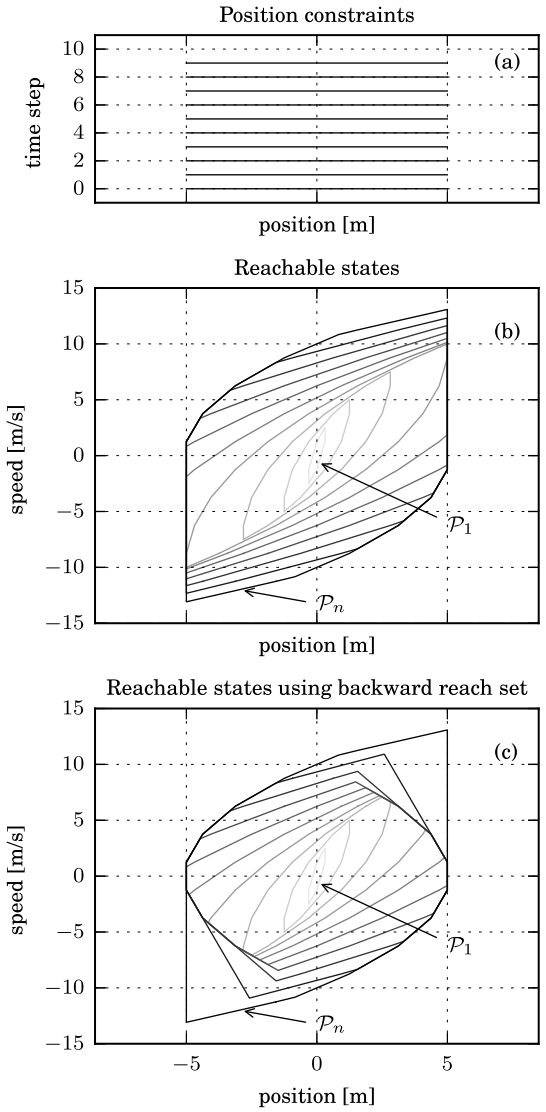

Fig. 2. Reachable set of one-dimensional motion of a point mass with bounded acceleration. The position of the point mass is constrained to the intervals $I_1 = I_2 = \cdots = I_n$ at time steps $1, 2, \ldots, n$ (a). Forward solution of the reachable set (b) and solution using the backward minimal reachable set approximation (c).

We calculate the set of possible states $\mathcal{P}_i$ at each time step $t_1, t_2, \ldots, t_n$ in a successive manner. First, the set is propagated one time step further, then it is intersected with the allowed position interval:

$$\mathcal{P}_i^* \subseteq \left(e^{A(t_i - t_{i-1})}\mathcal{P}_{i-1} \oplus \mathcal{P}_{u,x}\,(t_i - t_{i-1})\right) \cap (I_i \times [-\infty, \infty])\,.$$

If additional speed constraints are considered:

$$\mathcal{P}_i \subseteq \left(e^{A(t_i - t_{i-1})}\mathcal{P}_{i-1} \oplus \mathcal{P}_{u,x}\,(t_i - t_{i-1})\right) \cap \left(I_i \times [v_{min,x}, v_{max,x}]\right). \quad (6)$$

The last equation is an overapproximation, since the speed constraint may be violated in between the time interval.

The set $e^{A(t_i - t_{i-1})}\mathcal{P}_{i-1} \oplus \mathcal{P}_{u,x}(t_i - t_{i-1})$ is convex, for convex $\mathcal{P}_{i-1}$. However, the curved boundary of $\mathcal{P}_{u,x}$ complicates the computation of the Minkowski sum of the sets and the intersections with halfspaces. To perform these calculations efficiently, we overapproximate each set by a polytope $\mathcal{P}_{u,x}$ which has a fixed number of supporting halfspaces. Each state on the set boundary corresponds to a switching time of the bang-bang input. The linear equation for one halfspace at some γ is given by (see Fig. 3):

$$\left(\begin{matrix}\frac{\mathrm{d}x_t^{(l)}}{\mathrm{d}\gamma} \\ \frac{\mathrm{d}\dot{x}_t^{(l)}}{\mathrm{d}\gamma}\end{matrix}\right)^T\Bigg|_\gamma \begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}^T \begin{pmatrix}x_t \\ \dot{x}_t\end{pmatrix} \leq \left(\begin{matrix}\frac{\mathrm{d}x_t^{(l)}}{\mathrm{d}\gamma} \\ \frac{\mathrm{d}\dot{x}_t^{(l)}}{\mathrm{d}\gamma}\end{matrix}\right)^T\Bigg|_\gamma \begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}^T \begin{pmatrix}x_t^{(l)} \\ \dot{x}_t^{(l)}\end{pmatrix}\Bigg|_\gamma$$

and similarly for $x_t^{(h)}$. Fig. 2 (b) shows the calculated reachable sets $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_n$ for an example where we choose the intervals $I_1 = I_2 = \cdots = I_n$ to be equal.

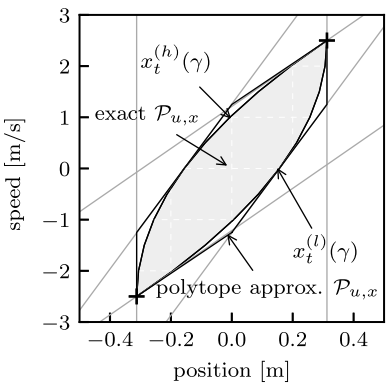

Fig. 3. Closed-form solution $\mathcal{P}_{u,x}$ of the reachable set of an acceleration-bounded point mass with initial state $(0, 0)^T$ and a polytope approximation using supporting halfspaces at switching times $\gamma = 0, 0.5, 1$.

B. Minimal Backward Reachable Set

Considering the same one-dimensional motion (4) from the previous subsection, some states may propagate over several time steps through position intervals but eventually cannot reach an allowed interval in a future time step. Such states are denoted inevitable collision states (ICS) [25]. The region of inevitable collision states is determined by the backward minimal reachable set of the forbidden region [26]. States that inevitably collide within $[0, T]$ can contribute to $\mathbf{reach}(t; \mathcal{X}_0)$ but do not contribute to $\mathbf{reach}_{\text{ant}}(t; \mathcal{X}_0, T)$ and may therefore be removed during the iterative computation.

The exact computation of ICS yields again a reachability analysis and can therefore in general not be computed efficiently [27]. Instead we determine an underapproximative set $\mathcal{ICS}$ of states which inevitably collide and remove these from the iterative computation of $\mathbf{reach}_{\text{ant}}$:

$$\mathcal{P}_i' := \mathcal{P}_i \setminus \mathcal{ICS} = \mathcal{P}_i \cap \overline{\mathcal{ICS}},$$

with the complement set (see Appendix A)

$$\overline{\mathcal{ICS}}(\overline{I_j}, t_i, t_j) = \left\{(x_i, \dot{x}_i)^T \;\middle|\; \begin{pmatrix}-1 & -\Delta t_{ij}\\ 1 & \Delta t_{ij}\end{pmatrix}\begin{pmatrix}x_i \\ \dot{x}_i\end{pmatrix} \leq \begin{pmatrix}-I_{j,min,x} + \frac{1}{2}\,a_{max}\Delta t_{ij}^2 \\ I_{j,max,x} + \frac{1}{2}\,a_{max}\Delta t_{ij}^2\end{pmatrix}\right\}$$

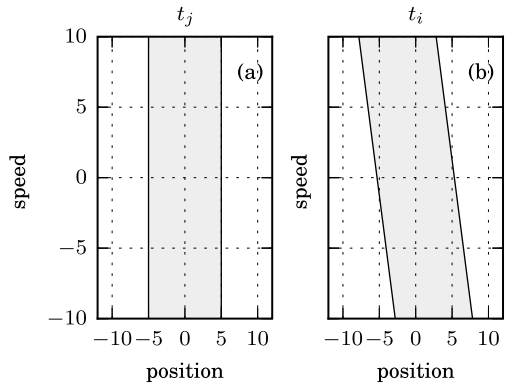

Fig. 4. Position constraint for an acceleration-bounded point mass at time t_j (a) and the corresponding allowed states at $t_i < t_j$ (b).

with $\Delta t_{ij} = t_j - t_i$. Fig. 4 shows the set of feasible states t_j for the constraint $x(t_j) \in I_j$. The set $\overline{\mathcal{ICS}}(\overline{I_j}, t_i, t_j)$ at $t_j < t_i$ contains all states which can reach the feasible states at t_i.

This can be repeated for all points in time $t_i < t_j \leq t_n$ to bound the set of all allowed initial states. Thus, the anticipated reachable set at time t_i can be tightened by the constraints after t_i:

$$\mathcal{P}_i' := \mathcal{P}_i \bigcap \left(\bigcap_{j=i+1}^{n} \overline{\mathcal{ICS}}(\overline{I_j}, t_i, t_j) \right).$$

$\mathcal{P}_i'$ stays convex, since it is an intersection of a convex set with halfspaces. Fig. 2 (c) shows the tightened reachable set for the example scenario of the previous section.

IV. Algorithm - Reachable Set of Two-Dimensional Motion

So far we have only considered one-dimensional motion. Our proposed algorithm calculates an overapproximation of the anticipated reachable set of the two-dimensional motion (3) at discrete points in time. The approximated set is represented as the union $\mathcal{R}_i$ of base sets $\mathcal{B}_i^{(q)}$, which can be represented efficiently. Each $\mathcal{B}_i^{(q)}$ is the Cartesian product of two convex 2-dimensional polytopes $\mathcal{P}_{i,x/y}^{(q)}$. Each polytope corresponds to one direction of motion along the x-axis and the y-axis. One axis of each polytope corresponds to the position, the other to speed:

$$\begin{aligned} \mathbf{reach}_{\text{ant}}(t_i; \mathcal{X}_0, T) \subseteq \ \mathcal{R}_i &= \bigcup_q \mathcal{B}_i^{(q)} \\ \text{with} \quad \mathcal{B}_i^{(q)} &= \mathcal{P}_{i,x}^{(q)} \times \mathcal{P}_{i,y}^{(q)}, \\ \text{int}(\mathrm{B}_i^{(r)}) \cap \text{int}(\mathcal{B}_i^{(l)}) &= \emptyset \text{ for } r \neq l. \end{aligned} \tag{7}$$

The $\mathcal{B}_i^{(q)}$ are selected such that the interior is pairwise disjoint.

The algorithm computes the sets $\mathcal{R}_i$ iteratively (see Fig. 5 and Algorithm 1 Steps 3-7; we use the notation $\{\mathcal{B}_i^{(q)}\}_q = \{B_i^{(1)}, \mathcal{B}_i^{(2)}, \dots\}$ to denote a list of elements). First, the previous set $\mathcal{R}_{i-1} = \cup_q \mathcal{B}_{i-1}^{(q)}$ is propagated by one time step according to the dynamical system (6) and ignoring the obstacles $\mathcal{O}(t)$. Second, all colliding states are removed from the propagated set. The resulting set is overapproximated by a set with representation (7). Third, the new base sets $\mathcal{B}_i^{(q)}$ are stored as nodes in a graph $\mathcal{G}$. An edge is inserted into $\mathcal{G}$ between $\mathcal{B}_{i-1}^{(l)}$ and $\mathcal{B}_i^{(s)}$ if one is reachable by the other. Each of these three steps are explained in the following Sec. IV-A to Sec. IV-C.

Algorithm 1

Input: Initial set $\mathcal{X}_0 = \cup_q B_0^{(q)}$, collision detection for axis-aligned rectangles with $\mathcal{F}(t)$.

Output: Graph $\mathcal{G}$ with nodes storing $\{\mathcal{B}_i^{(q)}\}_q$ for $i = 1, \dots, n$ time steps.

1: **function** COMPUTEREACHSET($\cup_q \mathcal{B}_0^{(q)}$)
2: $\mathcal{G}$.INIT($\cup_q \mathcal{B}_0^{(q)}$)
3: **for** $i = 1$ to n **do**
4: $\cup_q \hat{\mathcal{B}}_i^{(q)} \leftarrow$ PROPAGATE($\cup_q \mathcal{B}_{i-1}^{(q)}$)
5: $\cup_q \mathcal{B}_i^{(q)}, \mathcal{E} \leftarrow$ OVERAPPROXIMATE($\cup_q \hat{\mathcal{B}}_i^{(q)} \setminus \mathcal{F}(t_i)$)
6: $\mathcal{G}$.UPDATEGRAPH($\cup_q \mathcal{B}_i^{(q)}$, $\mathcal{E}$)
7: **end for**
8: **return** $\mathcal{G}$
9: **end function**

10: **function** PROPAGATE($\cup_q \mathcal{B}_{i-1}^{(q)}$)
11: **for all** $\mathcal{B}_{i-1}^{(q)}$ in $\{\mathcal{B}_{i-1}^{(q)}\}_q$ **do**
12: $\hat{\mathcal{P}}_{i,x}^{(q)} \leftarrow$ PROPAGATEX($\mathcal{P}_{i-1,x}^{(q)}$)
13: $\hat{\mathcal{P}}_{i,y}^{(q)} \leftarrow$ PROPAGATEY($\mathcal{P}_{i-1,y}^{(q)}$)
14: $\hat{\mathcal{B}}_i^{(q)} \leftarrow (\hat{\mathcal{P}}_{i,x}^{(q)} \times \hat{\mathcal{P}}_{i,y}^{(q)})$
15: **end for**
16: **return** $\cup \hat{\mathcal{B}}_i^{(q)}$
17: **end function**

18: **function** OVERAPPROXIMATE($\cup_q \hat{\mathcal{B}}_i^{(q)} \setminus \mathcal{F}(t_i)$)
19: $\{Rect_d^{(\cdot)}\} \leftarrow$ DISCRETIZE($\{\text{proj}_{xy}(\hat{\mathcal{B}}_i^{(\cdot)})\}$)
20: $\{Poly_m^{(\cdot)}\} \leftarrow$ MERGE($\{Rect_d^{(\cdot)}\}$)
21: $\{Rect_p^{(\cdot)}\} \leftarrow$ PARTITION($\{Poly_m^{(\cdot)}\}$)
22: $\{Rect_s^{(\cdot)}\} \leftarrow$ SPLIT($\{Rect_p^{(\cdot)}\}$)
23: $\mathcal{E} \leftarrow$ OVERLAP($\{Rect_s^{(\cdot)}\}$, $\{\text{proj}_{xy}(\hat{\mathcal{B}}_i^{(\cdot)})\}$)
24: **for all** $Rect_s^{(l)}$ in $\{Rect_s^{(l)}\}_l$ **do**
25: $\mathcal{B}_i^{(l)} \leftarrow$ CREATEBASESET($Rect_s^{(l)}$, $\{\hat{\mathcal{B}}_i^{(k)}\}_{(l,k)\in\mathcal{E}}$)
26: **end for**
27: **return** $\cup_q \mathcal{B}_i^{(q)}$, $\mathcal{E}$
28: **end function**

A. Flow of the Reachable Set Ignoring Obstacles

Both polytopes $\mathcal{P}_{i-1,x/y}^{(q)}$ of each $\mathcal{B}_{i-1}^{(q)}$ are propagated according to the solution of the LTI system with the overapproximation (6) and ignoring any obstacles:

$$\begin{aligned} \hat{\mathcal{P}}_{i,x}^{(q)} := \Bigg(\left\{ \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} s \,\middle|\, s \in \mathcal{P}_{i-1,x}^{(q)} \right\} \oplus \mathcal{P}_{u,x}(\Delta t) \Bigg) \\ \bigcap \left\{ \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \,\middle|\, v_{min,x} \leq \dot{x} \leq v_{max,x} \right\} \end{aligned}$$

and similarly for the y-direction (Algorithm 1 Steps 12-13).

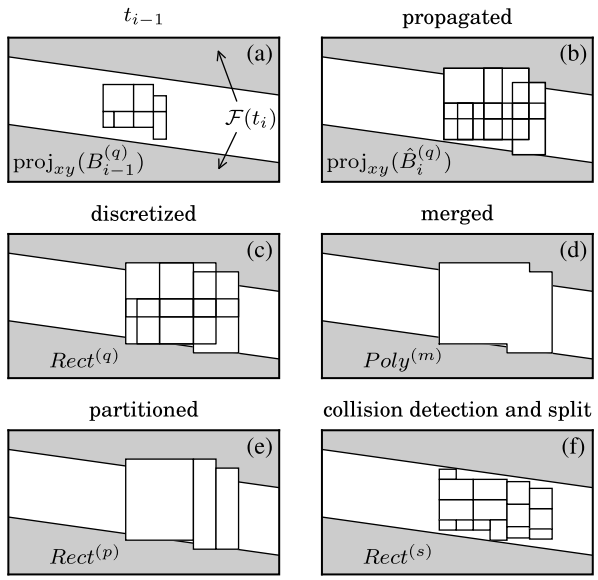

Fig. 5. Propagation and overapproximation of the reachable set after one time step. Shown is the projected reachable set in xy-domain. The reachable set of the previous time step (a) is propagated according to the dynamical system as described in Sec. IV-A. The propagated set (b) is overapproximated (c)-(f) considering collisions with obstacles as described in Sec. IV-B.

The propagated set $\cup_q \hat{\mathcal{B}}_i^{(q)} = \cup_q (\hat{\mathcal{P}}_{i,x}^{(q)} \times \hat{\mathcal{P}}_{i,y}^{(q)})$ may intersect with the set of forbidden states $\mathcal{F}(t_i)$, and $\hat{\mathcal{B}}_i^{(q)}$ may overlap with each other.

B. Overapproximation of the Reachable Set Considering Obstacles

In order to consider obstacles and obtain an $\mathcal{R}_i$ with the set representation (7) for the next time step, the forbidden states are removed from $\cup_q \hat{\mathcal{B}}_i^{(q)}$ and the set is overapproximated such that:

$$\mathcal{R}_i = \cup_q \mathcal{B}_i^{(q)} \supseteq \ (\cup_q \hat{\mathcal{B}}_i^{(q)} \setminus \mathcal{F}(t_i)) \tag{8}$$

where $\mathcal{B}_i^{(q)}$ are of the form (7).

We create this set $\mathcal{R}_i$ by partitioning the projection of $\cup_q \hat{\mathcal{B}}_i^{(q)} \setminus \mathcal{F}(t_i)$ in the position domain (see Fig. 5) and assign an overapproximation of the velocity region to each subset of the partition.

1) Discretize and Merge: The merge step computes the union of the projection of $\{\hat{\mathcal{B}}_i^{(q)}\}_q$ in the position domain. The boundaries in the position domain are axis-aligned rectangles (Algorithm 1 Step 19). We denote these rectangles by $Rect_d^{(\cdot)}$. The union of the rectangles are orthogonal polygons (Algorithm 1 Step 20). The computation of the polygons is related to Klee's measure problem and can be done efficiently using a sweep line algorithm and a segment tree [28]. In general, the coordinates of the vertices of the polygon are real numbers. Thus, the calculation would yield orthogonal polygons with a large number of corners even for slightly displaced edges. Therefore we discretize the boundary by enlarging the rectangles to a grid with segment length k_g in advance:

$$Rect_{d,\,\min \text{ x}}^{(q)} := \left\lfloor \frac{\min(\text{proj}_x\ (\hat{\mathcal{P}}_{i,x}^{(q)}))}{k_g} \right\rfloor \cdot k_g$$

$$Rect_{d,\,\max \text{ x}}^{(q)} := \left\lceil \frac{\max(\text{proj}_x\ (\hat{\mathcal{P}}_{i,x}^{(q)}))}{k_g} \right\rceil \cdot k_g$$

and similarly along the y-direction.

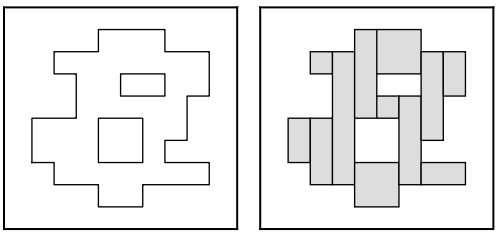

Fig. 6. Partitioning of an orthogonal polygon into rectangles.

2) Partition: The orthogonal polygons of Sec. IV-B1 are split into axis-aligned rectangles (Algorithm 1 Step 21). Through a sweep line algorithm the polygons are cut at each vertical edge into a rectangle and the remaining polygon. This continues until the remaining polygon is fully partitioned into rectangles (Fig. 6).

3) Collision Detection and Splitting: So far, workspace obstacles have been ignored. In this step (Algorithm 1 Step 22), each axis-aligned rectangle is split into two equally sized rectangles if it intersects an obstacle. The split is done by cutting the axis-aligned rectangle along the middle x- or y-coordinate depending on which one is longer. This is repeated until each rectangle is collision free or it collides and its diameter is less than the radius of the occupied region of the vehicle $\mathcal{A}(s(t))$ (cf. Sec. II-B). In the latter case, all states lying in the axis-aligned rectangle intersect an obstacle, and the states can be excluded. The set of computed $Rect_s^{(\cdot)}$ covers the projection of the desired set (8):

$$\text{proj}_{xy}(\cup_q \hat{\mathcal{B}}_i^{(q)} \setminus \mathcal{F}(t_i)) \subseteq \left(\cup_q\ Rect_s^{(q)}\right).$$

4) Overlap: In order to extend the two-dimensional set $\cup_q\ Rect_s^{(q)}$ of reachable positions to a four-dimensional set with velocity information, it is determined which $Rect_s^{(l)}$ is reachable by which $\hat{\mathcal{B}}_i^{(q)}$. The reachable pairs (l, q) are stored in the set $\mathcal{E}$.

5) Create New Base Sets: For each $Rect_s^{(l)}$ a set $\mathcal{B}_i^{(l)} = \mathcal{P}_{i,x}^{(l)} \times \mathcal{P}_{i,y}^{(l)}$ with an overapproximation of the reachable velocities is created (Algorithm 1 Step 25). The polytopes $\mathcal{P}_{i,x}^{(l)}$ and $\mathcal{P}_{i,y}^{(l)}$ are constructed from the convex hull of the propagated polytopes of all parent $\hat{\mathcal{B}}_i^{(k)}$ intersected with the spatial bounds of the axis-aligned rectangle $Rect_s^{(l)}$:

$$\mathcal{P}_{i,x}^{(l)} = \text{convexhull}\left(\bigcup_{(l,k)\in\mathcal{E}} \left(\hat{\mathcal{P}}_{i,x}^{(k)} \bigcap \left\{ \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \middle| \ Rect_{s,\,\min \text{ x}}^{(l)} \le x \le Rect_{s,\,\max \text{ x}}^{(l)} \right\}\right)\right).$$

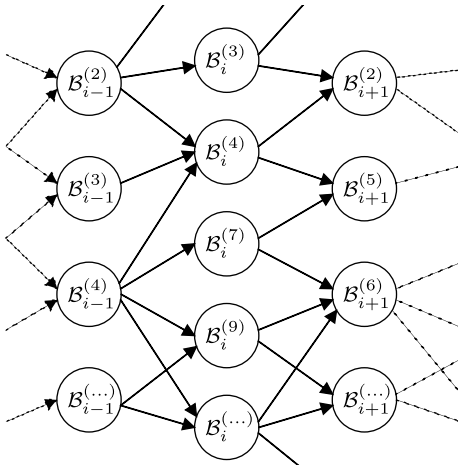

Fig. 7. Relationship between $\mathcal{B}_i^{(q)}$ of different time steps.

The union of the polytopes from the previous time step is in general not convex. We use the convex hull to restore convexity. Similarly, the polytope for the y-direction is assigned.

C. Reachability Between Base Sets of Succeeding Time Steps

Usually not all $\mathcal{B}_{i-1}^{(\cdot)}$ can reach all $\mathcal{B}_i^{(\cdot)}$ in two succeeding time steps. We express the reachability between sets $\mathcal{B}_{i-1}^{(k)}$ and $\mathcal{B}_i^{(l)}$ through a graph $\mathcal{G}$ (see Fig. 7). Each node in $\mathcal{G}$ contains one set $\mathcal{B}$. An edge is added between two nodes if one can reach the other in one time step (Algorithm 1 Step 6).

V. Multiple Runs to Refine Drivable Area

In Alg. 1, the reachable set at each time step is constructed from the result of the previous time step. This corresponds to an overapproximation of $\mathbf{reach}(t; \mathcal{X}_0)$ (cf. (1)) which neglects information about subsequent steps. For the anticipated reachable set $\mathbf{reach}_{\text{ant}}(t; \mathcal{X}_0, T)$, it is additionally required that there is a trajectory which continues each state until time T without entering $\mathcal{F}$. This property is difficult to check because it leads again to a reachability analysis.

Instead we use two approaches to tighten the overapproximation of $\mathbf{reach}_{\text{ant}}$: i) we remove $\mathcal{B}_i^{(q)}$ if it does not have any descendant in $\mathcal{G}$ and i is not the last time step (there are no descendants in the last step) and ii), we calculate additional constraints to apply the simplified concept of inevitable collision states from Sec. III-B.

The first approach can be readily implemented on the sets $\{\mathcal{B}_1^{(q)}\}_q, \dots, \{\mathcal{B}_{n-1}^{(q)}\}_q$. The second approach is explained in the following. For each time step we construct an axis-aligned rectangle $Rect_i^C$ in position domain which must be passed by all trajectories. The rectangle constraints $Rect_{i=1,\dots,n}^C$ are constructed by running Alg. 1 once and calculate the bounding box of $\{\text{proj}_{xy}(\mathcal{B}_i^{(q)})\}_q$ at each time t_i. In Sec. III-B it is shown for the one-dimensional case that given a position interval, which must be passed at time t_i, the set of possible initial states at some time $t < t_i$ is restricted. We apply the same technique here. If the state along the x- or y- direction cannot reach either the x- or y- interval of the axis-aligned rectangle of any $Rect_i^C$ in subsequent time steps, it is removed.

The propagation step of Alg. 1 is extended to exclude all these states:

$$\hat{\mathcal{P}}_{i,x}^{(q)} = \left(\left\{ \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} s \,\middle|\, s \in \mathcal{P}_{i-1,x}^{(q)} \right\} \oplus \mathcal{P}_{u,x}(\Delta t) \right) \bigcap \left\{ \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \,\middle|\, v_{min,x} \leq \dot{x} \leq v_{max,x} \right\} \bigcap \left(\bigcap_{j=i}^{n} \overline{\mathcal{ICS}}(\overline{I}_{j,x}, t_i, t_j) \right)$$

where $I_{j,x}$ denotes the interval of $Rect_j^C$ along the x-axis. The propagation along the y-direction is extended similarly.

In some cases a goal region is provided by a high-level planner, e.g. an area corresponding to a selected lane at some time $t_n = T$. The $\mathbf{reach}_{\text{ant}}$ can be used to narrow down the set of initial states which possibly can reach this goal region. After a first run, all $\mathcal{B}_n^{(q)}$ which intersect the goal region are selected. All of their ancestors up to initial time t_0 are selected. For this subset of $\{\mathcal{B}_1\}, \dots, \{\mathcal{B}_n\}$ the boundary boxes are calculated and used as constraints for a second run of the algorithm as described above.

VI. Experiments and Evaluation

A. Identical Dynamical Constraints in x- and y-Direction

As a first example we consider the scenario shown in Fig. 8. Two cars A and B are passing each other on two separate lanes in opposite direction. The ego vehicle drives behind car A.

The parameters to compute the drivable area can be chosen depending on the considered application. For emergency situations high acceleration bounds shall be selected. Lower values can be used to compute a bound of a comfortable drivable area. The following parameters are used for the ego vehicle in this examples:

time step Δt	0.15s
absolute maximum acceleration $a_{max,x/y}$	$10.0\frac{\text{m}}{\text{s}^2}$
minimum speed $v_{min,x/y}$	$-30.0\frac{\text{m}}{\text{s}}$
maximum speed $v_{max,x/y}$	$30.0\frac{\text{m}}{\text{s}}$
grid size k_g	0.5m
radius of vehicle	0.9m
time horizon T	3.0s

We suppose that a prediction of car A and car B for the given time horizon is provided. In this example, a prediction of the occupied region is based on the following assumptions [29] that i) each car stays in the lane and ii) each car may accelerate or brake with $10.0\frac{\text{m}}{\text{s}^2}$ in longitudinal direction, if the speed is below some parameterized speed v_s. Above v_s, acceleration is inversely proportional to speed, up to a maximum speed $v_{\max}$. We emphasize that the prediction is not part of the presented algorithm and is used here only for the sake of the example. Any prediction that provides a collision detection with axis-aligned rectangles at any given point in time can be used in combination with the presented algorithm.

Fig. 8 shows the reachable set at different points in time. The two dark regions are those forbidden due to the prediction of the other two cars. The boundary of the street is modeled using

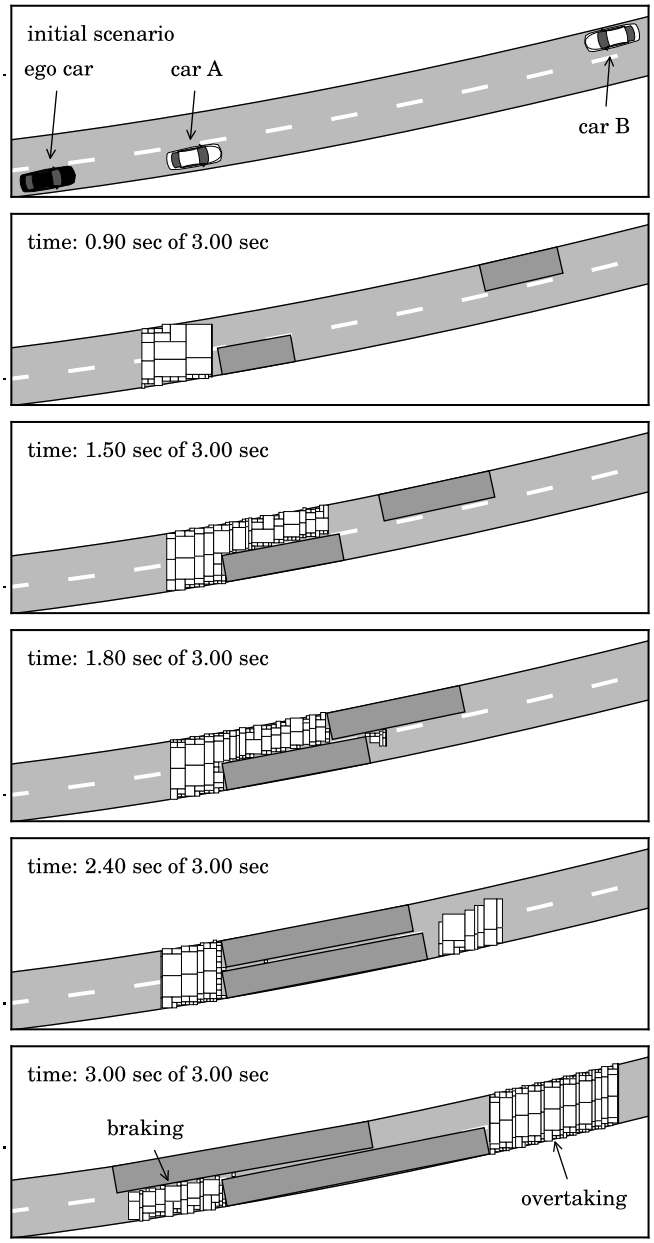

Fig. 8. Drivable area at different time steps in the overtaking scenario.

oriented rectangles (not shown in the figure). The evolution of the reachable set shows that two maneuvers are implicitly covered—braking and overtaking. The total number of sets $\mathcal{B}$ is 1447. The computation takes approx. 75ms for the 3.0s time horizon in a single-threaded C++ implementation on a 2.6 GHz Core i5 (I5-4288U).

The calculated set is 5-dimensional $(x, \dot{x}, y, \dot{y}, t)$. Fig. 9 shows the covered velocity region at time step 1.5 sec. To simplify the plot, only the maximum and minimum speed for each $\mathcal{B}$ in both directions are shown.

In a further step, we select the goal region to be the area ahead of the overtaken car and run the algorithm again for a refinement (cf. Sec. V). Fig. 10 shows the improvement after the second run of the algorithm. The region shown for the first run are all $\mathcal{B}_n$ of the last time step lying in the goal region and all their ancestors. For the second run, the bounding boxes of

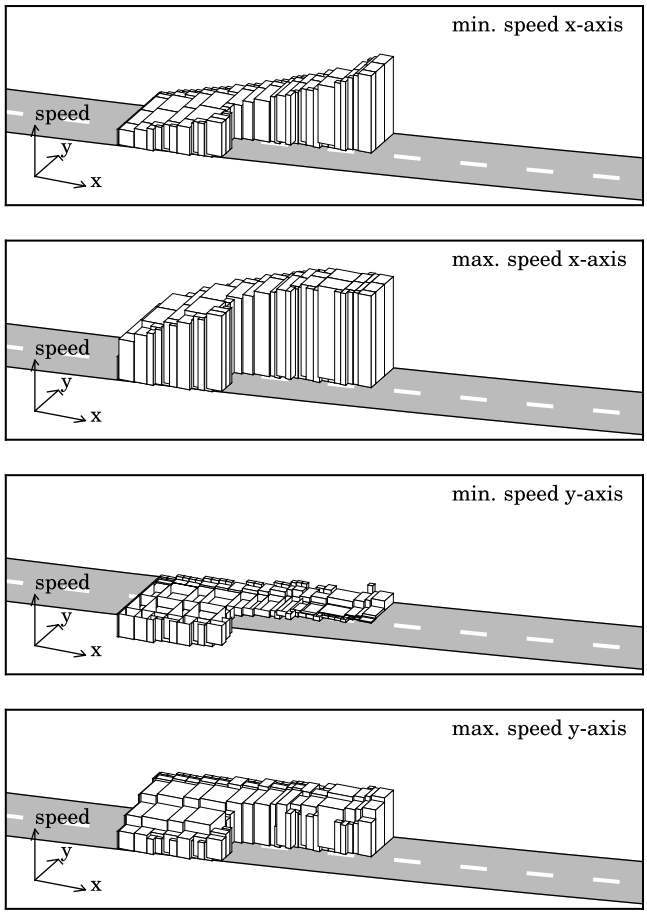

Fig. 9. Boundaries of the reachable velocities at time step 1.5 s of the overtaking scenario.

each time step are used to construct constraints. This tightens the reachable set, particularly in the early time steps. There, a larger number of subsequent constraints apply which shrink the area more strongly.

The information gained by the calculated reachable set might be used for a subsequent trajectory planner. For example, the narrow passage during overtaking can be easily identified. Since any trajectory must lie within the determined regions, intermediate goal regions at given time steps can be constructed for a planning algorithm.

B. Different Dynamical Constraints in Lateral and Longitudinal Direction

At higher speeds, for example on highways, it is often reasonable to consider the lateral and longitudinal direction separately. On curved roads we choose a coordinate system, which aligns along the center of the road [30]. The longitudinal positions are located along the arc length of the curved road center, the lateral position is given along the normal vector of the road center at a given longitudinal position. This makes it possible to map a Cartesian coordinate system to the curved road. In this example, the reachable set of lateral-longitudinal position and speeds is determined on a part of a highway as shown in Fig. 11. The cars' future positions are predicted using the assumption that i) each car stays in the lane and ii) each

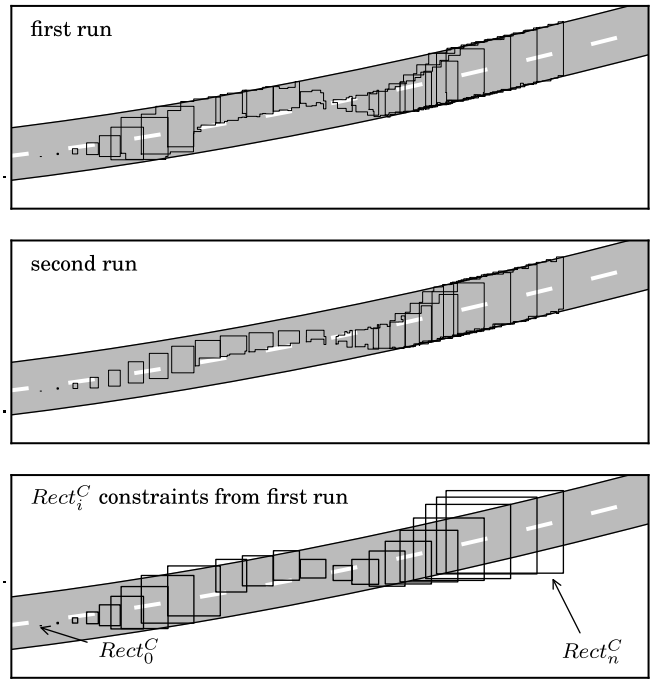

Fig. 10. Refinement of the drivable area of the overtaking maneuver after a second run of the algorithm.

car brakes. A safety margin is added behind and in front of all cars. Again, any prediction may be used as long it provides collision checks with axis-aligned rectangles at a given point in time.

The following parameters are chosen for the ego vehicle in this examples:

time step Δt	0.15s
time horizon T	3.0s
absolute maximum acceleration (long.) $a_{max,x}$	$10.0\frac{\text{m}}{\text{s}^2}$
minimum speed (long.) $v_{min,x}$	$0.0\frac{\text{m}}{\text{s}}$
maximum speed (long.) $v_{max,x}$	$45.0\frac{\text{m}}{\text{s}}$
absolute maximum acceleration (lat.) $a_{max,y}$	$3.0\frac{\text{m}}{\text{s}^2}$
minimum speed (lat.) $v_{min,y}$	$-3.0\frac{\text{m}}{\text{s}}$
maximum speed (lat.) $v_{max,y}$	$+3.0\frac{\text{m}}{\text{s}}$
grid size (lat. and long.) k_g	0.5m
radius of vehicle	0.9m

We compare the overapproximation of the reachable set with a strict underapproximation. The underapproximation is calculated using a rapidly-exploring random tree [31]. The states are connected using an analytical solution for the reachable set of the speed and acceleration bounded system [32]. The exact drivable area of our model is therefore larger than the one computed by sampling, but smaller than the one computed by our approximative set-based method. The distance of the samples to the border of the reachable set indicates the maximum overapproximation.

Two different initial states for the host vehicle are simulated. These two scenarios show the impact of the initial state on the calculated reachable set. In the first scenario, the initial longitudinal speed is 35 m/s. In the second scenario, the initial longitudinal speed is 43 m/s. The initial position and lateral speed are the same in both cases. The total number of sets $\mathcal{B}$ is

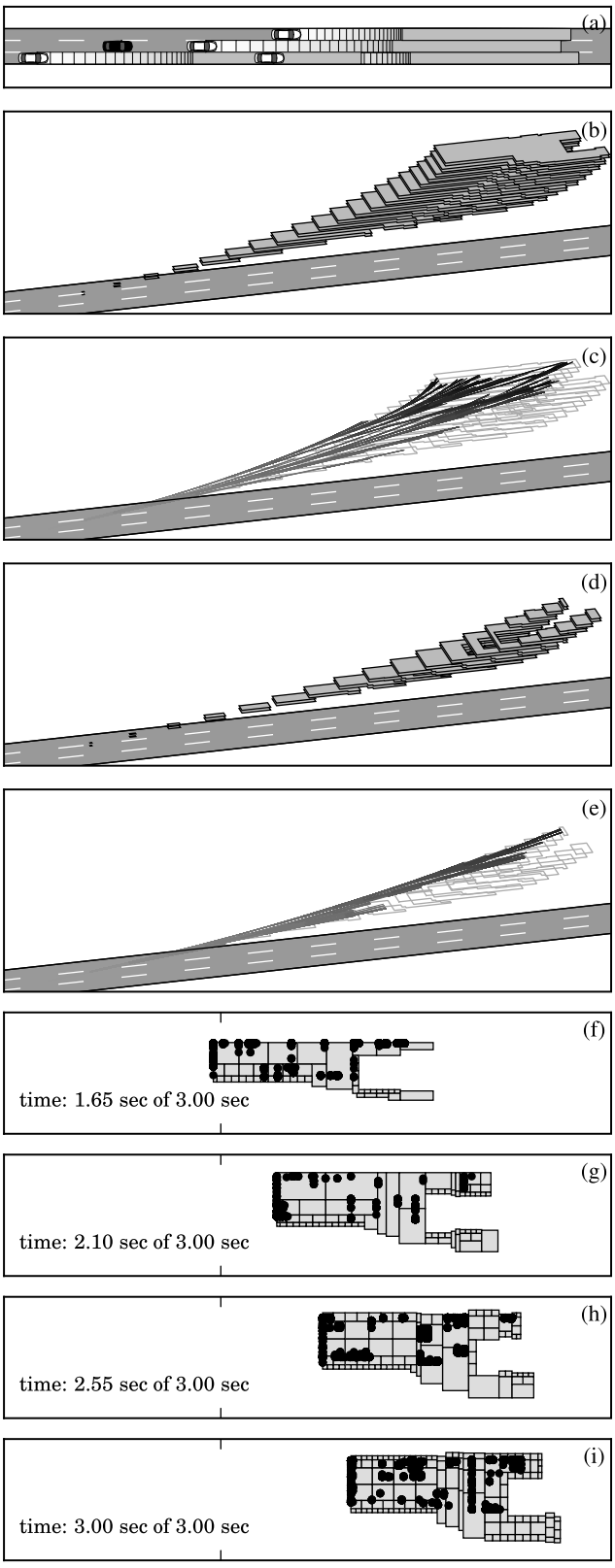

Fig. 11. Reachable set for the same scenario (a) with two different initial longitudinal speeds of 35 m/s (b) and 43 m/s (d). Underapproximation of the reachable set through a randomly sampled tree of trajectories (c,e-i).

1074 and 362, respectively. The computation times are 50ms and 20ms. The slower initial speed shows a clearly larger bound of maneuverability region. The reachable set of the

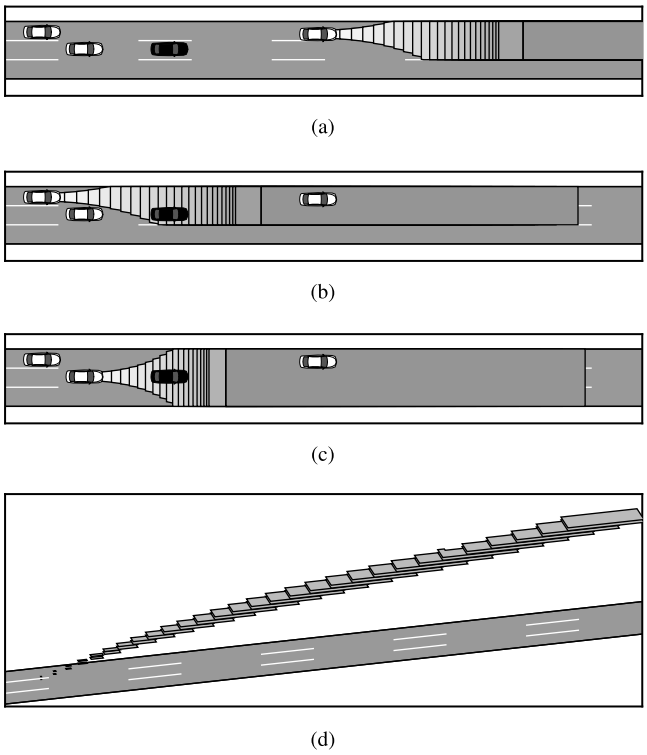

Fig. 12. Conservative prediction of the three surrounding vehicles A, B and C (a)-(c). Computed drivable area of the ego vehicle (d).

faster initial speed vanishes at prediction time 3.0 sec. Since all evasive maneuvers must lie in the reachable set, it proves that in this case and in the given traffic prediction, no such maneuver exists. Possibly, new measures for risk assessment can be constructed from properties of the reachable set, for example its volume or its change of size over different time steps.

C. Computation With Different Prediction Models

In this example, we show that our approach can be easily combined with different traffic prediction methods. The drivable area for two traffic predictions is computed and both areas are compared. One prediction covers all possible occupancies of surrounding traffic with formal guarantees [29], the second prediction uses a constant velocity model. In this scenario, the ego vehicle and three surrounding vehicles drive on a highway. The parameters for the drivable area computation are chosen as in the previous Sec. VI-B. The prediction of each traffic participant following the method in [29] is shown in Fig. 12 (a)-(c). The computed drivable area of the ego vehicle is shown in Fig. 12 (d). Since in this prediction uncertain behavior is considered by enlarging the occupied region, the drivable area is reduced compared to a less conservative prediction. For example, car A can either stay in its current lane or merge into the middle lane and block the ego vehicle. Therefore, the ego vehicle is forced to merge into the right lane as shown by the computed drivable area in Fig. 12 (d). In contrast, the constant velocity prediction as shown in Fig. 13 leaves considerably more space for the ego vehicle. The cars A, B and C stay in their lanes and may not accelerate or brake. Thus, the ego vehicle can either stay in its lane or merge into the left or right lane. The computed drivable area can immediately show the influence of different prediction models on the feasibility of possible maneuvers of the ego vehicle.

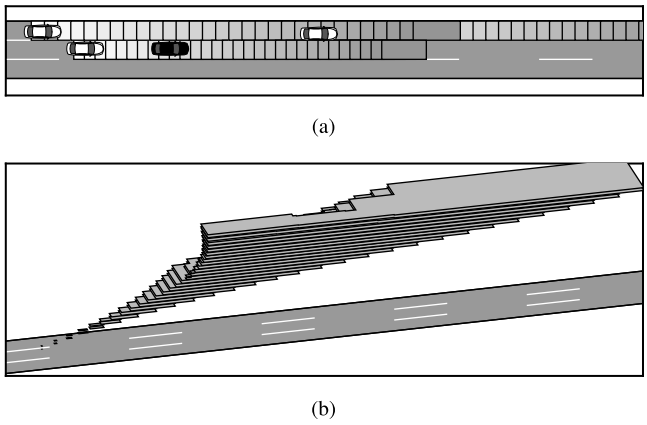

Fig. 13. Constant velocity prediction of the three surrounding vehicles A, B and C (a). Computed drivable area of the ego vehicle (b).

VII. Conclusion

We present a novel approach for efficiently computing the drivable area of road vehicles with formal guarantees. Our approach opens up new application areas that were previously impossible. First we can prune the search space of possible trajectories since we can guarantee that certain combinations of positions and velocities are impossible and will result in a crash or the traversal of road boundaries—this would not be possible with non-formal results. Second, we can determine with certainty that no action exists for avoiding a collision and thus can appropriately trigger collision mitigation systems. Third, the size of the drivable area can be used to determine the risk of a traffic situation, e.g. by analyzing the drivable area over time and determining whether times exist for which the solution space is tight, which corresponds to a dangerous situation. Another possibility to assess the risk of a situation is to compute the drivable area for an uncertain prediction with probabilistic occupancies with different level sets of probability to determine at which level a collision cannot be avoided anymore.

Our method is particularly applicable to problems in dynamic environments with a short planning horizon of a few seconds. For these short time horizons, the motion of other vehicles can be predicted reasonably. Our set representation and the construction of our algorithm are tailored towards an efficient approach despite the necessity to operate in five dimensions. We have demonstrated in this paper that our approach is fast enough to be embedded in automated vehicles. In the future, we plan to exploit the benefits of this approach for efficient trajectory planning. In particular our approach is able to detect narrow passages between obstacles which are usually difficult to find by trajectory planners.

APPENDIX A
DERIVATION OF THE APPROXIMATIVE MINIMAL BACKWARD REACHABLE SET

Consider the one-dimensional motion with position constraints (4). We denote the position x at time t_j for an initial state $(x_i, \dot{x}_i)^T$ at time t_i and an input $u(t)$ by

$$x_j(u; x_i, \dot{x}_i) = x_i + \dot{x}_i \cdot (t_j - t_i) + \int_{t_i}^{t_j} \int_{t_i}^{t} u(\tau) \mathrm{d}\tau \mathrm{d}t. \quad (9)$$

If we set the forbidden region to the complement of the position constraints $\mathcal{F}(t_i) := \overline{I_i}$ for $t_0, \ldots, t_n$, we obtain from (2):

$$(x_i, \dot{x}_i)^T \in \mathbf{reach}_{\text{ant}} \Rightarrow \exists u,\ x_j(u; x_i, \dot{x}_i) \in [I_{j,min}, I_{j,max}].$$

If a state only reaches the forbidden region $\mathcal{F}$ for any input u, it is an ICS and does not belong to $\mathbf{reach}_{\text{ant}}$:

$$\forall u,\ x_j(u; x_i, \dot{x}_i) \notin [I_{j,min}, I_{j,max}] \Rightarrow (x_i, \dot{x}_i)^T \notin \mathbf{reach}_{\text{ant}}. \quad (10)$$

We denote by $\mathcal{ICS}$ an underapproximation of the ICS set. One approximate $\mathcal{ICS}$ deduced from (10) is the set of all states $(x_i, \dot{x}_i)$ which are accelerated by $+a_{max}$ but their position at t_j is $< I_{j,min}$, and all states which are accelerated by $-a_{max}$ but their position at t_j is $> I_{j,max}$:

$$\mathcal{ICS}(\overline{I_j}, t_i, t_j) := \{(x_i, \dot{x}_i)^T \mid x_j(+a_{max}; x_i, \dot{x}_i) < I_{j,min} \vee x_j(-a_{max}; x_i, \dot{x}_i) > I_{j,max}\}. \quad (11)$$

The complement set $\overline{\mathcal{ICS}}$ is obtained from (11) and (9):

$$\overline{\mathcal{ICS}}(\overline{I_j}, t_i, t_j) = \left\{ (x_i, \dot{x}_i)^T \,\middle|\, \begin{pmatrix} -1 & -\Delta t_{ij} \\ 1 & \Delta t_{ij} \end{pmatrix} \begin{pmatrix} x_i \\ \dot{x}_i \end{pmatrix} \leq \begin{pmatrix} -I_{j,min,x} + \frac{1}{2}\, a_{max} \Delta t_{ij}^2 \\ I_{j,max,x} + \frac{1}{2}\, a_{max} \Delta t_{ij}^2 \end{pmatrix} \right\}$$

with $\Delta t_{ij} = t_j - t_i$.

APPENDIX B
COMPUTATIONAL COMPLEXITY

This section gives a brief summary of the algorithms we use for the geometric calculations in our implementation. The number of newly created sets $\mathcal{B}_i$ in each iteration generally depends on the obstacle set $\mathcal{O}(t)$ and cannot be predicted. We therefore give the complexity in terms of b, which denotes the maximum number of elements $\mathcal{B}_i$ of all iterations. Similarly, we use the maximum number p of vertices of $\mathcal{P}_{i,x/y}$ in all iterations. The number of time steps is n.

Two polytope representations are distinguished. The $\mathcal{V}$-representation of a polytope through the convex hull of a set of points, and the $\mathcal{H}$-representation through the bounded intersection of a set of halfspaces:

$\mathcal{P}_{i,x/y}$	$\mathcal{V}$-representation	p points
$\mathcal{P}_{u,x/y}$	$\mathcal{V}$-representation	u points
$\overline{\mathcal{ICS}}$	$\mathcal{H}$-representation	$2n$ halfspaces

The points in $\mathcal{V}$-representation are stored either in counter-clockwise order or lexicographically in ascending x-direction.

The complexity of the operations involving convex 2-dimensional polytopes is:

A. Linear Mapping $e^{At}\ \mathcal{P}_{x/y}$

$O(p)$, result keeps counter-clockwise point order of $\mathcal{P}_{x/y}$ for the particular A.

B. Minkowski Sum $\mathcal{P}_{x/y} \oplus \mathcal{P}_u$

$O(p + u)$, if both $\mathcal{P}_{x/y}$ and $\mathcal{R}_u$ are sorted in counter-clockwise order [33, Th. 3.10].

C. Intersection $\mathcal{P}_{x/y} \cap (\cap_{j=0}^{n} \overline{\mathcal{ICS}}(\overline{I}_j))$

$O(p + 2n)$, if $\cap_{j=0}^{n} \overline{\mathcal{ICS}}$ is given in $\mathcal{V}$-representation and both are ordered counter-clockwise. The conversion from $\mathcal{H}$-representation takes $O(2n \log 2n)$ [33, Corollary 3.10].

D. Convex Hull of $\cup_{q=1}^{b} \mathcal{P}_{x/y}^{(q)}$

$O(bp \log bp)$ and, if all points are sorted lexicographically, $O(bp)$ [33, Th. 1.1]. The sorting step can be improved if each $\mathcal{P}_{x/y}^{(q)}$ is ordered counter-clockwise. The points of each $\mathcal{P}_{x/y}^{(q)}$ can be distributed into two lists: one in lexicographically ascending order, and one in descending order in $O(p)$. Then, all $2n$ sorted lists are merged using a k-way merge in $O(bp \log 2b)$ [34, Problem 9.5–9].

Next, we consider the complexity of the operations involving axis-aligned rectangles and orthogonal polygons. The number of edges of polygons is denoted e. For orthogonal polygons the number of horizontal edges equals the number of vertical edges $e_v = e/2$.

E. Partition Orthogonal Polygons into Axis-Aligned Rectangles

$O(e_v \log e_v)$. We use a sweep line algorithm that cuts parts of the polygon at vertical edges. The active set of vertical edges is kept in an interval tree. It can be shown that the number b_n of newly created axis-aligned rectangles in this partition is $2b_n \geq e_v \geq b_n$. By assumption it holds that $b \geq b_n$.

F. Merge Axis-Aligned Rectangles to Orthogonal Polygons

$O(b \log b + e \log(b^2/e))$ [28].

G. Overlap Between Two Sets of Axis-Aligned Rectangles

$O(b^2)$, if all rectangles of one set overlap with all rectangles of the other.

REFERENCES

[1] D. Ferguson, T. M. Howard, and M. Likhachev, "Motion planning in urban environments," *J. Field Robot.*, vol. 25, nos. 11–12, pp. 939–960, 2008.

[2] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.

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

[4] M. McNaughton, C. Urmson, J. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 4889–4895.
[5] D. Madås *et al.*, "On path planning methods for automotive collision avoidance," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2013, pp. 931–937.
[6] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 72–89, 1986.
[7] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *Int. J. Robot. Res.*, vol. 29, no. 5, pp. 485–501, 2010.
[8] M. Likhachev, G. J. Gordon, and S. Thrun, "Ara*: Anytime a* with provable bounds on sub-optimality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 767–774.
[9] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 952–964, Feb. 2017.
[10] J. Park, S. Karumanchi, and K. Iagnemma, "Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1101–1115, Oct. 2015.
[11] I. Xausa, R. Baier, O. Bokanowski, and M. Gerdts, *Computation of Safety Regions for Driver Assistance Systems by Using a Hamilton–Jacobi Approach*, document, 2014. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01123490
[12] M. Gerdts and I. Xausa, "Avoidance trajectories using reachable sets and parametric sensitivity analysis," in *System Modelling and Optimization*. Berlin, Germany: Springer, 2011, pp. 491–500.
[13] C. Schmidt, F. Oechsle, and W. Branz, "Research on trajectory planning in emergency situations with multiple objects," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Sep. 2006, pp. 988–992.
[14] N. Kaempchen, B. Schiele, and K. Dietmayer, "Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 4, pp. 678–687, Dec. 2009.
[15] S. Söntges and M. Althoff, "Determining the nonexistence of evasive trajectories for collision avoidance systems," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 956–961.
[16] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH J.*, vol. 1, no. 1, p. 1, Dec. 2014.
[17] A. Chutinan and B. H. Krogh, "Computational techniques for hybrid system verification," *IEEE Trans. Autom. Control*, vol. 48, no. 1, pp. 64–75, Jan. 2003.
[18] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *Proc. IEEE Conf. Decision Control*, Dec. 2008, pp. 4042–4048.
[19] T. Dang and O. Maler, "Reachability analysis via face lifting," in *Hybrid Systems: Computation and Control*. Berlin, Germany: Springer, 1998, pp. 96–109.
[20] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis: Internal approximation," *Syst. Control Lett.*, vol. 41, no. 3, pp. 201–211, 2000.
[21] A. Girard and C. Le Guernic, "Zonotope/hyperplane intersection for hybrid systems reachability analysis," in *Hybrid Systems: Computation and Control*. Berlin, Germany: Springer, 2008, pp. 215–228.
[22] O. Stursberg and B. H. Krogh, "Efficient representation and computation of reachable sets for hybrid systems," in *Hybrid Systems: Computation and Control*. Berlin, Germany: Springer, 2003, pp. 482–497.
[23] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi, "Beyond HYTECH: Hybrid systems analysis using interval numerical methods," in *Hybrid Systems: Computation and Control*. Berlin, Germany: Springer, 2000, pp. 130–144.
[24] S. Kambhampati and L. Davis, "Multiresolution path planning for mobile robots," *IEEE J. Robot. Autom.*, vol. RA-2, no. 3, pp. 135–145, Sep. 1986.
[25] T. Fraichard and H. Asama, "Inevitable collision states—A step towards safer robots?" *Adv. Robot.*, vol. 18, no. 10, pp. 1001–1024, 2004.
[26] I. M. Mitchell, "Comparing forward and backward reachability as tools for safety analysis," in *Hybrid Systems: Computation and Control*. Berlin, Germany: Springer, 2007, pp. 428–443.
[27] A. Lawitzky, A. Nicklas, D. Wollherr, and M. Buss, "Determining states of inevitable collision using reachability analysis," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 4142–4147.
[28] W. Lipski and F. P. Preparata, "Finding the contour of a union of iso-oriented rectangles," *J. Algorithms*, vol. 1, no. 3, pp. 235–246, 1980.
[29] M. Althoff and S. Magdici, "Set-based prediction of traffic participants on arbitrary road networks," *IEEE Trans. Intell. Veh.*, vol. 1, no. 2, pp. 187–202, Jun. 2016.
[30] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 987–993.
[31] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.
[32] J. Johnson and K. Hauser, "Optimal acceleration-bounded trajectory planning in dynamic environments along a specified path," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2035–2041.
[33] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, *Computational Geometry*. Berlin, Germany: Springer, 2000.
[34] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.

Sebastian Söntges received the Diploma degree in electrical engineering from Technische Universität München, Germany, in 2011, where he is currently pursuing the Ph.D. degree. From 2012 to 2013, he was a Research Assistant with the Department of Electrical Engineering. In 2014, he joined the Department of Computer Science. His research interests include control theory and motion planning with applications for autonomous vehicles.

Matthias Althoff received the Diploma Engineering degree in mechanical engineering and the Ph.D. degree in electrical engineering from Technische Universität München, Germany, in 2005 and 2010, respectively. From 2010 to 2012, he was a Post-Doctoral Researcher with Carnegie Mellon University, Pittsburgh, USA, and an Assistant Professor with Technische Universität Ilmenau, from 2012 to 2013. He is currently an Assistant Professor in computer science with Technische Universität München. His research interests include formal verification of continuous and hybrid systems, reachability analysis, planning algorithms, nonlinear control, automated vehicles, and power systems.