Sampling-Based Optimal Trajectory Generation for Autonomous Vehicles Using Reachable Sets

Gerald Würsching and Matthias Althoff

***Abstract*— Motion planners for autonomous vehicles must obtain feasible trajectories in real-time regardless of the complexity of traffic conditions. Planning approaches that discretize the search space may perform sufficiently in general driving situations, however, they inherently struggle in critical situations with small solution spaces. To address this problem, we prune the search space of a sampling-based motion planner using reachable sets, i.e., sets of states that the ego vehicle can reach without collision. By only creating samples within the collision-free reachable sets, we can drastically reduce the number of required samples and thus the computation time of the planner to find a feasible trajectory, especially in critical situations. The benefits of our novel concept are demonstrated using scenarios from the CommonRoad benchmark suite.**

I. Introduction

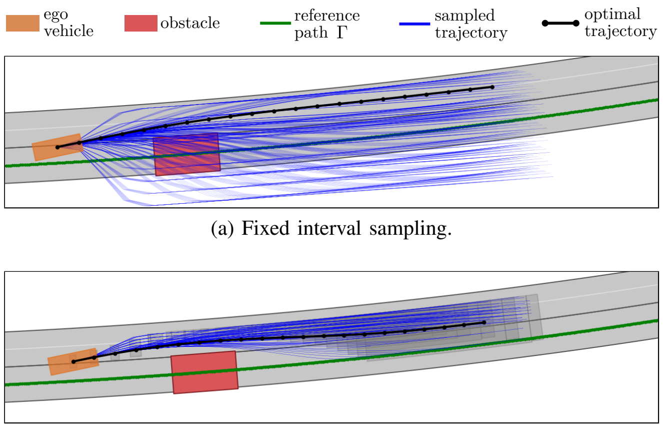
Motion planners are a crucial component within the software system of self-driving vehicles. The development of motion planning algorithms is subject to a wide range of requirements, including safety, robustness, and strict real-time capabilities. Discretization-based planning approaches often fail to meet those requirements when the criticality of the driving situation suddenly increases. Due to the discretization, the planner may not be able to swiftly detect narrow passages and generate feasible motions in safety-critical situations. Set-based reachability analysis combined with optimization-based planning techniques has shown promising results in terms of handling such complex maneuvers [1]. Particularly, the ability of reachable sets to efficiently detect narrow solution spaces in arbitrary traffic scenarios was demonstrated. In this work, we exploit reachability analysis to improve the performance of a sampling-based motion planner by constraining the sampling space using the bounds of the reachable sets (see Fig. 1).

A. Related Work

Discretization-Based Motion Planning: Various motion-planning techniques have been explored in the past; in-depth overviews are provided in [2], [3]. In this work, we focus on discretization-based approaches. *Rapidly exploring random trees* [4], [5] generate trajectories by randomly sampling and connecting states toward a goal area. *State lattices* [6]–[9] generate trajectory sets by connecting an initial state with several intermediate goal states that are typically chosen as vertices in a fixed grid. In [8], jerk-optimal trajectories are computed by sampling

This work was supported by the German Research Foundation (DFG) grant AL 1185/3-2.

The authors are with the Department of Computer Science, Technical University of Munich, 85748 Garching, Germany. gerald.wuersching@tum.de, althoff@in.tum.de



(a) Fixed interval sampling.

(b) Adaptive sampling using reachable sets.

Fig. 1. Example scenario showing the ego vehicle trying to evade a static obstacle while following a given reference route. (a) A naive sampling approach, which uses fixed intervals for each sampled state leads to a large number of non-drivable trajectories. (b) Our approach uses reachable sets (grey rectangles) to identify the narrow passageway and adapts the intervals for each sampled state according to the bounds of the reachable sets.

the terminal states of the trajectories within fixed intervals around a reference path (e.g., the lane centerline) and using quintic polynomials to generate a uniformly latticed set of trajectories (see Fig. 1a). A major drawback of lattice planners is their limited set of motions due to the constraint that partial motions must end in the vertices of the fixed grid [10]. Hence, exhaustive sampling of the state space may be required to retain reactive properties [3], which leads to an increased computational time of the planner. Several previous works have addressed this issue: Gu et al. [11], [12] propose a two-stage planning scheme, which first generates a coarse trajectory via a desired action sequence [11] or via optimization [12], followed by obtaining a fine local trajectory by sampling around the coarse trajectory. However, both approaches are limited to handling static obstacles and the optimization routine potentially leads to jittering in the reference path [13]. Similarly, [14]–[17] optimize a reference path for a subsequent sampling-based planner. Yet, these approaches mainly focus on adapting the reference path to unstructured road geometry or the presence of static obstacles, i.e., they do not consider the dynamic environment. The authors of [18] construct adaptive search spaces for sampling the trajectories based on the reachable space of the vehicle and the shape of the road, but without considering obstacles. In [19], the lattice is adapted to human driving behavior by learning sampling intervals from recorded lane change trajectories. However, this approach is tailored to a specific maneuver and also does not consider any obstacles.

Reachable Sets for Trajectory Planning: Several applications of set-based reachability analysis for motion planning have been proposed in previous works. In [20], an online verification method is presented which checks whether a trajectory is collision-free by considering all possible motions of dynamic obstacles. In [21], [22], reachability analysis is exploited to compute the drivable area of an autonomous vehicle and determine the non-existence of actions. An application to a path planning problem is shown in [23], where the exploration process of a rapidly exploring random tree is guided via reachable sets. Most recent works focus on combining reachability analysis with convex optimization techniques for planning intended motions [1] as well as fail-safe trajectories [24], [25]. Particularly, these approaches use reachable sets to obtain convex solution spaces and collision avoidance constraints for the optimization problem. Thereby, the application of set-based techniques has yielded promising results regarding the performance and computation time of the trajectory planner, especially in complex scenarios with constricted solution spaces.

B. Contributions

Inspired by the works of [1], [24], we present a concept for improving the performance of a sampling-based motion planner using set-based reachability analysis. In contrast to existing work, our approach is not tailored to a specific maneuver or limited to static obstacles. Furthermore, we are able to alleviate the dependence of the sampling-based planner on the quality of a reference path. Our concept determines the collision-free drivable area using reachability analysis and derives the intervals for sampling in the position and velocity domain from the bounds of the reachable sets. Thus, our planner is able to adapt its sampling technique to the dynamically changing environment, which increases the efficiency and performance of the planner as demonstrated by our numerical experiments. In particular, we significantly improve the computation time of the planner by reducing the number of required samples in scenarios with small solution spaces, which are inherently challenging for discretization-based approaches. We evaluate our concept using scenarios from the CommonRoad [26] benchmark platform and demonstrate the benefits of our concept compared to a standard approach that uses fixed sampling intervals.

The remainder of this paper is organized as follows: In Sec. II, we introduce the necessary preliminaries for our work. Sec. III describes our proposed concept for adapting the sampling intervals for a motion planner using reachable sets. We present the results of our experiments in Sec. IV and provide concluding remarks in Sec. V.

II. Preliminaries

In this section, we present some notations and definitions regarding reachability analysis and motion planning, which are necessary for our work.

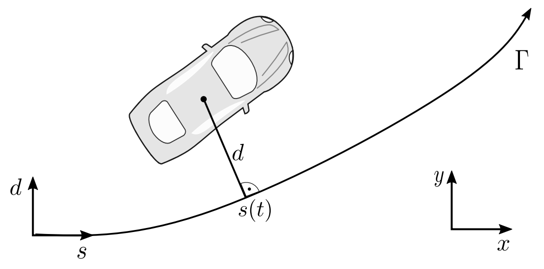


Fig. 2. Curvilinear coordinate system aligned with a reference path Γ.

A. Notations

Let us introduce the index $k \in \mathbb{N}_0$ as a discrete time step corresponding to a continuous time $t_k = k\Delta t$, with a fixed time increment $\Delta t \in \mathbb{R}^+$. We use k_0 and k_f to refer to the initial and final time step, respectively. Let us further introduce the set of admissible states $\mathcal{X}_k \subset \mathbb{R}^{n_x}$ and the set of admissible inputs $\mathcal{U}_k \subset \mathbb{R}^{n_u}$ for any dynamical system $x_{k+1} = f(x_k, u_k)$, where $x_k \in \mathcal{X}$ represents the state and $u_k \in \mathcal{U}$ the input. We adhere to the notations $\underline{\square}$ and $\overline{\square}$ to refer to the lower and upper bounds of the possible values of a variable $\square$, respectively.

B. Definitions

Definition 1 (Curvilinear Coordinate System [27]). *We use a curvilinear coordinate system that is aligned with a reference path $\Gamma : \mathbb{R} \to \mathbb{R}^2$, which is typically given by a route planner (see Fig. 2). Any position $(x, y)^T$ in the global Cartesian frame will be described by the arc length s along Γ and the orthogonal deviation d to $\Gamma(s)$.*

We denote the occupancy of the ego vehicle by $\mathcal{Q}(x_k) \subset \mathbb{R}^2$ and the occupancy of all surrounding obstacles by $\mathcal{O}_k \subset \mathbb{R}^2$. We can then define the set of forbidden states as $\mathcal{F}_k = \{x_k \in \mathcal{X}_k | \mathcal{Q}(x_k) \cap \mathcal{O}_k \neq \emptyset\}$.

Definition 2 (Reachable Set). *Let us specify the initial reachable set as $\mathcal{R}_{k_0} = \mathcal{X}_{k_0}$, with $\mathcal{X}_{k_0}$ being the set of possible initial states. Then, the reachable set $\mathcal{R}_{k+1}$ is defined as the set of states that can be reached from the previous reachable set $\mathcal{R}_k$ without intersecting with $\mathcal{F}_{k+1}$, i.e.,*

$$\begin{aligned}\mathcal{R}_{k+1} = \{x_{k+1} \in \mathcal{X}_{k+1} \mid \exists x_k \in \mathcal{R}_k, \exists u_k \in \mathcal{U}_k : \\ x_{k+1} = f(x_k, u_k) \wedge x_{k+1} \notin \mathcal{F}_{k+1}\}.\end{aligned}$$

In this work, $\mathcal{R}_k$ refers to an accurate approximation of the exact reachable set, since the computation of the exact reachable set is often too expensive for complex models [28]. For the final verification of the solution, however, an over-approximation is used [20].

Definition 3 (Projection Operator). *We define the projection operator $\text{proj}_{\square} : \mathcal{X} \to \mathbb{R}^m$, which maps every state $x \in \mathcal{X}$ to the elements specified by $\square$.*

Definition 4 (Drivable Area). *The drivable area $\mathcal{D}_k \subset \mathbb{R}^2$ contains all collision-free positions which the ego-vehicle can* *reach and is defined as the projection of the reachable set onto the position domain, i.e.,* $\mathcal{D}_k = \text{proj}_{(s,d)}(\mathcal{R}_k)$.

Since the drivable area can be disconnected due to obstacles in the environment, we introduce connected sets $\mathcal{C}_k^n \subseteq \mathcal{D}_k,\ n \in \mathbb{N}_0$, which contain the connected components of the drivable area at each time step k.

Definition 5 (Driving Corridor [1])**.** *A driving corridor* $\mathcal{C}(\cdot) = (\mathcal{C}_{k_0}, \ldots, \mathcal{C}_{k_f})$ *is a temporal sequence of connected sets* $\mathcal{C}_k \subseteq \mathcal{D}_k$.

Multiple driving corridors may exist, which correspond to different homotopy classes, i.e., sets of trajectories that can be deformed into each other without intersecting with obstacles [29]. In other words, multiple possible maneuvers of the ego vehicle might exist, e.g., evading or braking in front of an obstacle [1].

III. Sampling-Based Planning with Reachable Sets

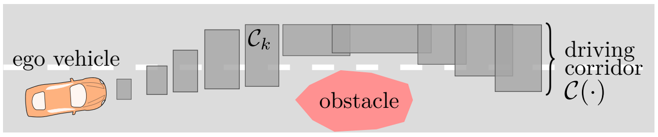
In this section, we present our motion-planning approach. We begin with a general overview of our method. Afterwards, we elaborate on the computation of reachable sets and the identification of driving corridors. Finally, we describe how we adapt the intervals for each sampled state for our trajectory planner from reachable sets.

A. Overview

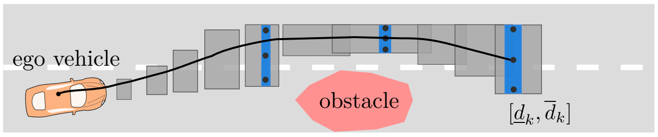
We describe the general idea of our concept using the scenario shown in Fig. 3, which depicts the ego vehicle navigating around an obstacle. Our planning approach uses a reference trajectory as input to specify a desired path and velocity which the ego vehicle should follow. Our algorithm then plans a trajectory for the planning horizon t_{k_f} using a fixed replanning time of Δt_r. The procedure for one planning cycle is described below.

We begin by computing the reachable sets for all time steps $k \in \{k_0, \ldots, k_f\}$ within the planning horizon to obtain the drivable area for the planning cycle (see Fig. 3a and Sec. III-B). Within the drivable area, we select a driving corridor (see Def. 5). We derive sampling intervals from the reachable sets corresponding to the selected driving corridor in order to constrain the sampled terminal states of our trajectories (see Fig. 3b and Sec. III-C). We determine the sampling interval for the longitudinal motion by projecting the reachable sets onto the longitudinal velocity domain v_s, since our planner aims to follow a desired velocity specified by the reference trajectory. The sampling interval for the lateral motion is obtained from the projection of the drivable area onto the lateral position domain d, as the planner should minimize deviations from the reference path.

Since we only sample the terminal points of the trajectories within the collision-free drivable area, the resulting trajectory set must be checked for kinematic feasibility and collisions [30]. The values for the kinematic constraints are taken from [31] for a medium-sized passenger vehicle. Finally, we obtain the optimal trajectory using a specified cost function.



(a) Computation of a driving corridor $\mathcal{C}(\cdot)$ as a sequence of connected sets $\mathcal{C}_k$. Each grey rectangle represents a connected set $\mathcal{C}_k$ within the collision-free drivable area $\mathcal{D}_k$ at different time steps k.



(b) Extraction of sampling intervals (shown here for the lateral position d): Each blue rectangle represents a sampling interval $[\underline{d}_k, \overline{d}_k]$ at different terminal times. The terminal points of the trajectories (black dots) are only sampled within the drivable area. The black line represents the resulting optimal trajectory.

Fig. 3. Our concept consists of two steps: (a) Determining the collision-free drivable area using reachability analysis. (b) Obtaining sampling intervals from reachable sets and sampling within them.

B. Reachability Analysis and Driving Corridors

The following algorithms for computing reachable sets and driving corridors are mainly based on [1], [21].

Reachability Analysis: For efficiency, we simplify the vehicle dynamics for the reachable set computation using a point-mass model in the curvilinear frame [1], with the vehicle center as the reference point. The state vector $x = (s, v_s, d, v_d)^T$ contains the longitudinal and lateral positions $(s, d)^T$ and velocities $(v_s, v_d)^T$. The input vector $u = (a_s, a_d)^T$ is composed of the longitudinal and lateral accelerations, a_s and a_d, respectively. The system dynamics is

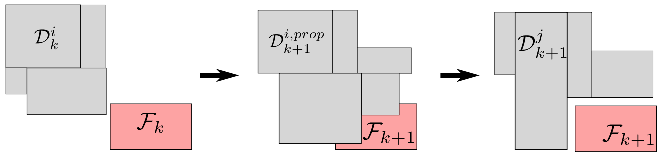
$$x_{k+1} = \begin{pmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{pmatrix} x_k + \begin{pmatrix} \frac{1}{2}\Delta t^2 & 0 \\ \Delta t & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ 0 & \Delta t \end{pmatrix} u_k, \quad (1)$$

with bounded values for the velocities and accelerations:

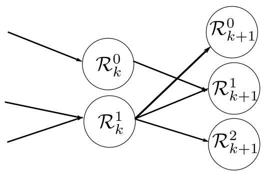
$$\underline{v_s} \le v_{s,k} \le \overline{v_s}, \quad \underline{v_d} \le v_{d,k} \le \overline{v_d},$$
$$\underline{a_s} \le a_{s,k} \le \overline{a_s}, \quad \underline{a_d} \le a_{d,k} \le \overline{a_d}.$$

We choose the bounds conservatively in order to account for kinematic limitations of the real vehicle (e.g., maximum drivable curvature) and effects resulting from the transformation of the vehicle dynamics to the curvilinear frame, which are not modeled in (1). We use this approximation for computational efficiency, since the drivability of the sampled trajectories is checked separately in any case, as mentioned above.

The reachable set at time step k is over-approximated as a union of base sets $\mathcal{R}_k^i,\ i \in \mathbb{N}_0$, where every base set denotes pairs of reachable positions and velocities in the curvilinear frame. We begin with the base set $\mathcal{R}_{k_0}^0$ and successively carry out the following procedure (see Fig. 4a): First, we propagate the previous base sets $\mathcal{R}_k^i$ forward in time using (1). The projection of the propagated base sets $\mathcal{R}_{k+1}^{i,prop}$ onto the position domain yields the propagated drivable areas



(a) Steps to compute the drivable area: The previous sets $\mathcal{D}_k^i$ are propagated to $\mathcal{D}_{k+1}^{i,prop}$ according to the system dynamics. The set of forbidden states $\mathcal{F}_{k+1}$ is excluded and we obtain the repartitioned sets $\mathcal{D}_{k+1}^j$.



(b) Example of a reachability graph $\mathcal{G}_\mathcal{R}$.

Fig. 4. Overview of reachability analysis based on [1], [21].

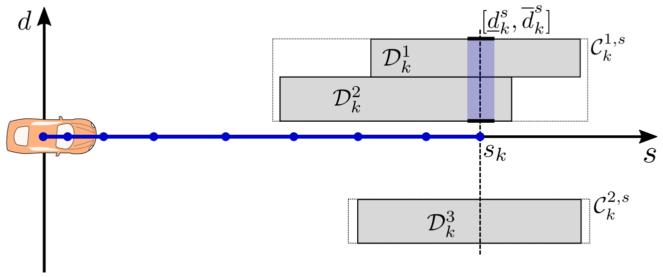


Fig. 5. For a sampled longitudinal trajectory (blue), we determine the drivable areas $\mathcal{D}_k^i$ which overlap with the end position s_k. In this example, we identify two connected sets within the union of $\mathcal{D}_k^i$ and select $\mathcal{C}_k^{1,s}$, since it is closer to the reference path. The obtained sampling interval $[\underline{d}_k^s\ ,\ \overline{d}_k^s]$ for the lateral position d is highlighted in blue.

$\mathcal{D}_{k+1}^{i,prop}$. In order to compute the collision-free drivable area, we determine the set of forbidden states $\mathcal{F}_{k+1}$ by considering static occupancies and the predicted occupancies of dynamic obstacles at time step $k+1$. Then, $\mathcal{F}_{k+1}$ is excluded from the propagated drivable areas, which results in the collision-free drivable areas $\mathcal{D}_{k+1}^j$. Afterwards, the corresponding base sets $\mathcal{R}_{k+1}^j$ are determined by computing the reachable velocities for the drivable areas $\mathcal{D}_{k+1}^j$. Thus, we have obtained all collision-free configurations which the ego-vehicle can reach at time step $k+1$. The spatiotemporal information of the reachability analysis is stored in a directed graph $\mathcal{G}_\mathcal{R}$ (see Fig. 4b). Therein, a node is constructed for each individual base set $\mathcal{R}_{k+1}^j$. An edge in $\mathcal{G}_\mathcal{R}$ implies that the corresponding base sets are reachable over one time step (e.g., $\mathcal{R}_{k+1}^2$ is reachable from $\mathcal{R}_k^1$ in Fig. 4b).

Driving Corridors: We aim to select a driving corridor (see Def. 5) from the drivable area using the approach proposed in [1]. We begin by identifying the connected sets $\mathcal{C}_{k_f}^n,\ n \in \mathbb{N}_0$ within the union of drivable areas $\mathcal{D}_{k_f}^i$ at the final time step k_f of the planning horizon and determine the driving corridor by traversing the reachability graph $\mathcal{G}_\mathcal{R}$ backwards in time. Using this approach we are able to eliminate those sets that do not have a descendant in $\mathcal{G}_\mathcal{R}$ from the driving corridor, i.e., we exclude sets which would collide in future time steps within the planning horizon. In $\mathcal{G}_\mathcal{R}$, we first determine the parent sets $\mathcal{D}_{k-1}^j$ for all $\mathcal{D}_k^i \in \mathcal{C}_k^n$. Next, we identify the connected sets $\mathcal{C}_{k-1}^m,\ m \in \mathbb{N}_0$ within the union of parent sets. Thus, we obtain all connected parent sets $\mathcal{C}_{k-1}^m$ from which the connected set $\mathcal{C}_k^n$ can be reached. This procedure is repeated backwards in time for all time steps $k \in \{k_0, \dots, k_f\}$ of the planning cycle. Since multiple driving corridors may exist, we select the driving corridor with the largest cumulative drivable area for planning, as proposed in [32].

C. Obtaining Sampling Intervals from Reachable Sets

After computing a driving corridor $\mathcal{C}(\cdot)$, we aim to obtain sampling intervals for our motion planner. The terminal times of the end states are uniformly distributed within a predefined interval, i.e., $t_{\text{terminal}} \in [t_{\text{start}}\ ,\ t_{k_f}]$. Subsequently, the procedure for computing the longitudinal and lateral sampling intervals for a given terminal time is described.

Longitudinal Velocity Intervals: Let k be the discrete time step corresponding to the selected terminal time. We then extract the connected set $\mathcal{C}_k$ from the driving corridor $\mathcal{C}(\cdot)$. The connected set contains several drivable areas $\mathcal{D}_k^i \in \mathcal{C}_k,\ i \in \mathbb{N}_0$, for which we determine the corresponding reachable sets $\mathcal{R}_k^i$. We then compute the lower and upper bounds of the longitudinal velocity, $\underline{v}_{s,k}^i$ and $\overline{v}_{s,k}^i$, respectively, for each base set using

$$\underline{v}_{s,k}^i = \inf\left\{\operatorname{proj}_{v_s}(\mathcal{R}_k^i)\right\}, \quad \overline{v}_{s,k}^i = \sup\left\{\operatorname{proj}_{v_s}(\mathcal{R}_k^i)\right\}.$$

Consequently, the sampling interval $\left[\underline{v}_{s,k}\ ,\ \overline{v}_{s,k}\right]$ for the longitudinal velocity is obtained by selecting the minimum/maximum values of the individual bounds of the base sets, i.e.,

$$\underline{v}_{s,k} = \min_i(\underline{v}_{s,k}^i), \quad \overline{v}_{s,k} = \max_i(\overline{v}_{s,k}^i).$$

Lateral Position Intervals: We compute a sampling interval for the lateral position d_k of the terminal state for each sampled longitudinal trajectory. Fig. 5 shows the procedure for determining this interval for a given longitudinal trajectory. First, we determine all sets $\mathcal{D}_k^i$ that overlap with the longitudinal end position s_k, i.e., $s_k \in \operatorname{proj}_s(\mathcal{D}_k^i)$ must hold. Afterwards, we obtain the connected components $\mathcal{C}_k^n$ within those sets. Multiple connected sets may exist when there are several homotopy classes, e.g., multiple passing sides around an obstacle. Since our motion planner aims to minimize the lateral deviation from the desired reference path, we select the connected set $\mathcal{C}_k^n$ with the closest lateral bound to the reference path. Finally, we obtain the sampling interval $\left[\underline{d}_k^s\ ,\ \overline{d}_k^s\right]$ for the lateral position by

$$\underline{d}_k^s = \inf\left\{\operatorname{proj}_d(\mathcal{C}_k^{n,s})\right\}, \quad \overline{d}_k^s = \sup\left\{\operatorname{proj}_d(\mathcal{C}_k^{n,s})\right\}.$$

Thereby, the superscript s indicates that the corresponding variable $\square^s$ is determined for a specific longitudinal position.

D. Sampling within the Intervals

In this work, we employ a simple approach by uniformly distributing the samples within the intervals. The number of samples is determined by the step size between the samples for each interval. We start with a default step size for the intervals of the terminal times t_{terminal}, the longitudinal velocity v_s, and the lateral position d. If the planner cannot determine a feasible trajectory, we successively bisect the step size for each interval, i.e., we double the number of samples per interval until a feasible trajectory is found. To bisect the intervals, we set a maximum number of iterations; consequently, the number of sampled trajectories is limited by a maximum number N_{max}. If no feasible trajectory can be obtained using the maximum number of samples, our planning algorithm aborts.

IV. Numerical Experiments

We evaluate our approach using the sampling-based planner of [8]. This planner computes jerk-optimal trajectories using quintic polynomials to connect the sampled terminal states with the initial state. The optimal trajectory is selected according to a cost function, which penalizes high acceleration values for comfort as well as deviations from the desired velocity and the reference path. We compare the performance of our concept with a basic implementation using fixed sampling intervals, which we subsequently refer to as the "standard" approach.

We first provide implementation details in Sec. IV-A. Afterwards, we evaluate our approach using an evasive scenario (Sec. IV-B) and a highway scenario (Sec. IV-C) from the CommonRoad [26] benchmark platform.

A. Implementation Details

For our experiments, we set the planning horizon to $t_{k_f} = 2\,\text{s}$, i.e., $k_f = 20$ with a time increment $\Delta t = 0.1\,\text{s}$. We set the replanning time to $\Delta t_r = 0.3\,\text{s}$, i.e., a new planning cycle begins at every third time step.

The sampling intervals for the standard approach are listed in Tab. I. Therein, the velocity v_{desired} is the scenario-specific desired velocity and a_{max} is the maximum deceleration from the kinematic constraints of the vehicle. The distribution of the samples within the intervals and the number of samples for the standard approach are determined according to the procedure described in Sec. III-D. The maximum number of sampled trajectories is set to $N_{\text{max}} = 2754$.

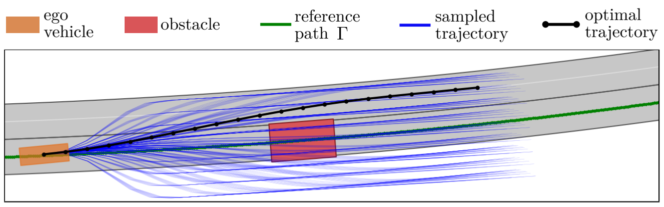
The algorithm for computing the reachable sets is implemented in C++ and the algorithms for extracting the sampling intervals and planning are implemented in Python. All experiments are performed on a single thread of a 1.80 GHz Intel Core$^{\text{TM}}$ i7-10510U CPU with 32 GB of memory. The full videos of our experiments are attached to this paper as supplementary material.

B. Evasive Scenario with a Static Obstacle

We first compare both approaches using the CommonRoad scenario *ZAM_Over-1_1:2018b*, in which the ego vehicle must evade a static obstacle. Additionally, we manually design a more critical version of the scenario by enlarging the static obstacle and reducing its distance to the initial position of the ego vehicle by approximately $10\,\text{m}$. The conditions

TABLE I
Fixed sampling intervals for the standard approach.

terminal state	**interval**
t_{terminal} in [s]	$[0.4\ ,\ t_{k_f}]$
d in [m]	$[-4.5\ ,\ 4.5]$
v_s in [m/s]	$[\max(0, v_{\text{desired}} - 0.125\, t_{k_f}\, a_{\text{max}})\ ,\ v_{\text{desired}} + 2]$



(a) Standard sampling approach with fixed intervals ($N = 540$).

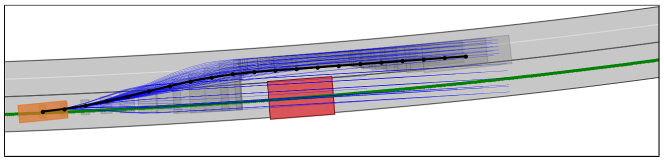
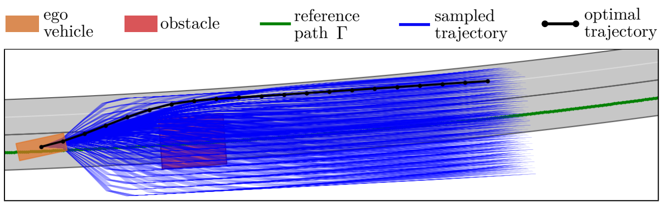

(b) Our sampling approach with reachable sets ($N = 109$).

Fig. 6. Evasive maneuver in the original scenario at $t_3 = 0.3\,\text{s}$.

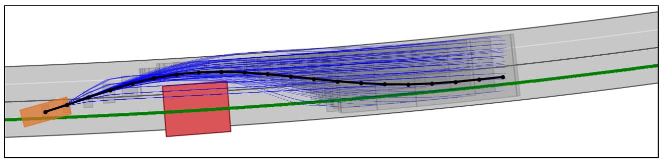
are subsequently referred to as the "original" and "critical" scenario.

Fig. 6 depicts the results of the second planning cycle at $t_3 = 0.3\,\text{s}$ for the original scenario. We can see that the standard sampling approach (Fig. 6a) needs to sample a larger number N of trajectories compared to our approach to find a feasible evasive maneuver. Since the standard approach generates the lateral terminal state of the trajectories in the predefined interval (Tab. I), the planner must increase the sampling density in order to detect the narrow passageway around the obstacle. Our approach on the other hand efficiently adapts sampling intervals using the drivable areas from the reachability analysis, resulting in generated trajectories that are mostly constrained to the narrow passageway. Since reachable sets are particularly well suited to detect small and convoluted solution spaces [1], [24], we are able to mitigate the drawbacks of our sampling-based planner in such driving situations.

The critical scenario is particularly challenging, because the obstacle is larger and closer to the ego vehicle. Hence, the solution space for feasible trajectories is further reduced. Fig. 7 illustrates the planning results at $t_3 = 0.3\,\text{s}$. The standard approach using fixed intervals was observed to result in an exhaustive sampling with $N = 2754$ generated trajectories (Fig. 7a). In contrast, our reachability-based sampling approach (Fig. 7b) is able to generate a valid evasive maneuver in this planning cycle without increasing the number of sampled trajectories compared to the original scenario. Thus, the potential of our approach to improve the performance of sampling-based planners becomes particularly apparent for complex maneuvers.



(a) Standard sampling approach with fixed intervals ($N = 2754$).



(b) Our sampling approach with reachable sets ($N = 105$).

Fig. 7. Evasive maneuver in the critical scenario at $t_3 = 0.3\,\mathrm{s}$.

TABLE II
PERFORMANCE COMPARISON AT TIME STEPS $k \in \{0, 3, 6\}$.

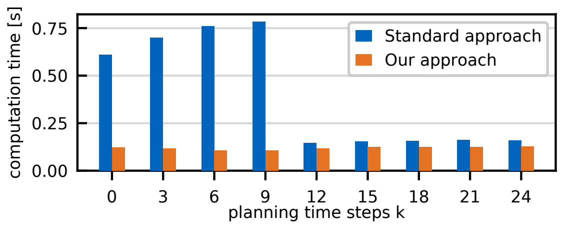
		standard approach			**our approach**		
scenario	k	N	N_{disc}	T_{C} [s]	N	N_{disc}	T_{C} [s]
original	0	540	384	0.609	112	65	0.122
	3	540	387	0.699	109	59	0.117
	6	540	393	0.760	100	44	0.108
critical	0	2754	2430	4.531	504	367	0.445
	3	2754	2479	4.265	105	74	0.084
	6	540	370	0.605	100	46	0.094

Tab. II shows the number of sampled trajectories N and discarded trajectories N_{disc} as well as the computation time T_{C} for the first three planning cycles (i.e., $k \in \{0, 3, 6\}$), where the scenario is most challenging. Thereby, T_{C} includes both the planning time and computation time for the reachability analysis. The evaluation shows that our approach significantly reduces the computation time due to the decreased number of necessary samples, which minimizes the number of trajectory evaluations and expensive collision checks the planner must execute.

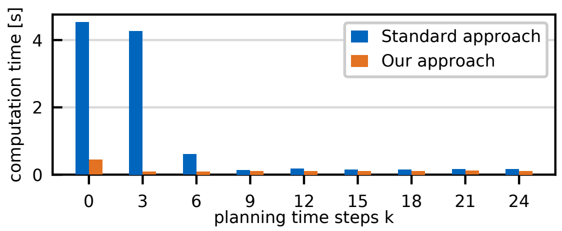
We compare the computation time T_{C} for all planning time steps for the original and critical scenarios in Fig. 8a and Fig. 8b, respectively. We notice that our concept does not significantly worsen the computation time of the planner, i.e., even after the vehicle has completed the evasive maneuver, the performance of both approaches is comparable. Although additional operations are required to obtain the reachable sets and adapt the sampling intervals in each planning cycle, the computational overhead to perform these operations is mostly compensated by the reduced planning time due to enhanced sampling. Overall, our concept tends to improve the robustness of the planner, especially when the complexity of the traffic situation increases.

C. Merging Scenario on a Highway

To evaluate our proposed concept in the presence of other traffic participants, we consider a highway lane change maneuver using the CommonRoad scenario *USA_US101-6_3_T-1:2018b*, which is obtained from real traffic data in the NGSIM dataset. We predict the motions of 29 other vehicles in the scenario using their most likely trajectories, however, any alternative prediction method could be applied.



(a) Original scenario.

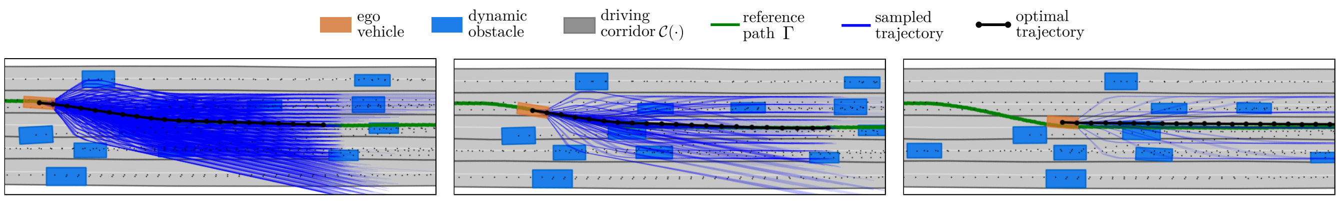


(b) Critical scenario.

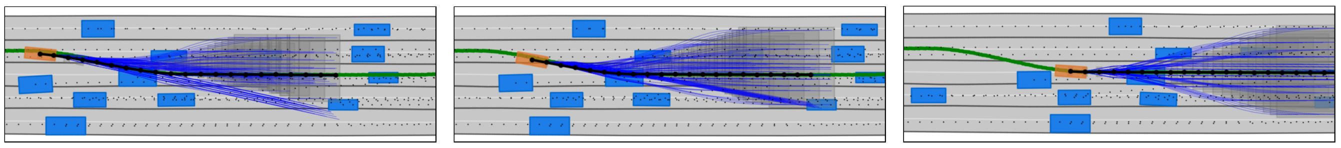
Fig. 8. Evasive maneuver: comparison of computation times over the planning cycles for the original scenario (a) and the critical scenario (b) (averaged over 10 runs).

Fig. 9 shows the considered scenario for selected time steps $k \in \{12, 15, 21\}$. The dynamic obstacles are shown at their positions at the respective time step k. The ego vehicle attempts to perform a single lane change into the middle highway lane, while another vehicle simultaneously switches to the same lane. We create a reference path that forces the ego vehicle to merge into a tight gap between two vehicles. This maneuver is particularly challenging, since the planner must detect the gap and additionally determine a feasible velocity to avoid a collision with the preceding vehicle. To guide the lane change maneuver, we generate the reference path Γ with a simple route planner, which does not consider any obstacles. Fig. 9a shows that the planner with a standard sampling approach struggles to detect the narrow gap at the beginning of the lane-change maneuver ($k = 12$). Since the solution space in the position and velocity domain is exceedingly small, the planner requires a large number of generated trajectories ($N = 2754$) to determine a feasible merging trajectory. Our reachability-based planner (Fig. 9b), however, is able to find a feasible motion at time step $k = 12$ with only $N = 120$ sampled trajectories. A similar observation can be made at the following time step $k = 15$. Both approaches can accomplish the maneuver with the ego vehicle merging into the desired lane at time step $k = 21$.

The computation time T_{C} for all planning time steps for both approaches is compared in Fig. 10. It can be observed that the computation time of the standard planner drastically increases when the scenario becomes more critical, i.e., at time steps $k \in \{12, 15\}$. The benefits of our proposed approach are particularly apparent in those time steps, because we can significantly minimize the computation time using our more efficient sampling technique.



(a) Standard sampling approach with fixed intervals. $k = 12$ (left), $k = 15$ (middle), $k = 21$ (right)



(b) Our sampling approach with reachable sets. $k = 12$ (left), $k = 15$ (middle), $k = 21$ (right)

Fig. 9. Merging maneuver in the highway scenario at selected time steps for the standard sampling approach (a) and our proposed approach (b).

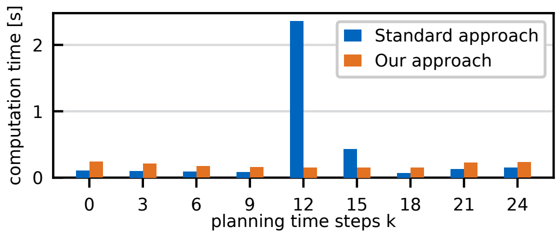


Fig. 10. Merging maneuver: comparison of computation times over the planning cycles for the highway scenario (averaged over 10 runs).

V. Conclusions

This paper proposes a concept to combine reachable sets with a sampling-based motion planner. We use reachability analysis to determine the collision-free drivable area of the ego vehicle and derive the sampling intervals from the reachable sets. As a result, our planner can adapt its sampling intervals to the dynamically changing environment in arbitrary traffic scenarios. Our experiments demonstrated that the presented approach outperforms a standard implementation, that samples in fixed intervals in terms of computation time and sampling efficiency. The benefits of our concept are particularly noticeable in complex scenarios with limited solution spaces, which are often difficult for discretization-based planners to handle. Future work will focus on further enhancing the sampling procedure of our approach, for example, by adapting the sampled terminal times of the trajectories and by incorporating different sampling strategies within the reachable sets.

Acknowledgment

The authors gratefully acknowledge financial support by the German Research Foundation (DFG) grant AL 1185/3-2.

References

[1] S. Manzinger, C. Pek, and M. Althoff, “Using reachable sets for trajectory planning of automated vehicles,” *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 232–248, 2020.

[2] 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.

[3] 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.

[4] S. M. LaValle and J. J. Kuffner Jr, “Randomized kinodynamic planning,” *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[5] S. Karaman and E. Frazzoli, “Sampling-based algorithms for optimal motion planning,” *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

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

[7] M. McNaughton, C. Urmson, J. M. Dolan, and J. W. Lee, “Motion planning for autonomous driving with a conformal spatiotemporal lattice,” *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 4889–4895, 2011.

[8] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, “Optimal trajectory generation for dynamic street scenarios in a Frenét frame,” *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 987–993, 2010.

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

[10] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, “A review of motion planning for highway autonomous driving,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 1826–1848, 2019.

[11] T. Gu and J. M. Dolan, “On-road motion planning for autonomous vehicles,” in *International Conference on Intelligent Robotics and Applications*, 2012, pp. 588–597.

[12] T. Gu, J. Snider, J. M. Dolan, and J. W. Lee, “Focused trajectory planning for autonomous on-road driving,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2013, pp. 547–552.

[13] T. Gu, J. M. Dolan, and J. W. Lee, “On-road trajectory planning for general autonomous driving with enhanced tunability,” *Intelligent Autonomous Systems 13. Advances in Intelligent Systems and Computing*, vol. 302, pp. 247–261, 2016.

[14] X. Li, Z. Sun, A. Kurt, and Q. Zhu, “A sampling-based local trajectory planner for autonomous driving along a reference path,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2014, pp. 376–381.

[15] Y. Zhang, H. Chen, S. L. Waslander, J. Gong, G. Xiong, T. Yang, and K. Liu, “Hybrid trajectory planning for autonomous driving in highly constrained environments,” *IEEE Access*, vol. 6, pp. 32 800–32 819, 2018.

[16] S. Wang, P. Zhao, Y. Zheng, B. Yu, W. Huang, H. Zhu, and H. Liang, “Reference path correction for autonomous ground vehicles driving over rough terrain,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2019, pp. 2405–2410.

[17] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu, “Real-time trajectory planning for autonomous urban driving : Framework , algorithms ,

and verifications," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 2, pp. 740–753, 2015.

[18] T. M. Howard, C. J. Green, A. Kelly, and D. Freguson, "State space sampling of feasible motions for high-performance mobile robot navigation in complex environments," *Journal of Field Robotics*, vol. 25, no. 6-7, pp. 325–345, 2008.

[19] W. Yao, H. Zhao, F. Davoine, and H. Zha, "Learning lane change trajectories from on-road driving data," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2012, pp. 885–890.

[20] 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.

[21] 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, 2017.

[22] ——, "Determining the nonexistence of evasive trajectories for collision avoidance systems," in *Proc. of the IEEE Conference on Intelligent Transportation Systems*, 2015, pp. 956–961.

[23] A. Shkolnik, M. Walter, and R. Tedrake, "Reachability-guided sampling for planning under differential constraints," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2009, pp. 2859–2865.

[24] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nature Machine Intelligence*, vol. 2, no. 9, pp. 518–528, 2020.

[25] C. Pek and M. Althoff, "Fail-safe motion planning for online verification of autonomous vehicles using convex optimization," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 798–814, 2020.

[26] 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.

[27] 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.

[28] A. Platzer and E. M. Clarke, "Formal verification of curved flight collision avoidance maneuvers: A case study," in *International Symposium on Formal Methods*, 2009, pp. 547–562.

[29] S. Bhattacharya and R. Ghrist, "Path homotopy invariants and their application to optimal trajectory planning," *Annals of Mathematics and Aritificial Intelligence*, vol. 84, no. 3, pp. 139–160, 2018.

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

[31] M. Althoff and G. Würsching, "CommonRoad : Vehicle Models." [Online]. Available: https://commonroad.in.tum.de/model-cost-functions

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