2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Time and Memory-Efficient Computation of Hamilton-Jacobi Reachable Sets based on a Level Set Method Employing Adaptive Grids

Christopher Bohn, Philipp Reis, Manuel Schwartz, and Sören Hohmann

***Abstract*— Formal verification of dynamic control systems often involves reachability analysis to ensure safety and performance characteristics. Hereby, Hamilton-Jacobi-based methods are beneficial as they can be applied to non-linear, continuous systems under the influence of bounded disturbances. Furthermore, they can consider input- and state constraints. These benefits come with the computational effort of solving a Hamilton-Jacobi partial differential equation. State-of-the-art methods numerically determine a viscosity solution at the vertices of a static grid that is used to discretize the state space. This becomes particularly costly if the reachable set propagates fast and needs to be determined precisely, as this requires a grid of many vertices. This contribution proposes a method that computes the solution successively on small adaptive grids instead of on one static grid to reduce the computational effort of Hamilton-Jacobi reachability analysis. Using the proposed method, changes between grids can be performed in an outer or inner approximative manner. The performance of the proposed method is demonstrated in a numerical example computing a forward reachable set of a Dubins car model. While increasing the accuracy of the resulting set, the method proposed saves $73\,\%$ of computation time, $76\,\%$ of average memory usage, and $43\,\%$ of maximum memory usage in the presented scenario.**

***Index Terms*— Reachability Analysis, Hamilton-Jacobi, System Verification, Adaptive Discretization**

I. Introduction

The increasing complexity of dynamic systems and control systems induces challenges in the process of verifying that a system complies with its specifications [1]. As established verification approaches like simulation-based verification and other brute-force methods become challenging or even intractable for complex systems [2], the need for formal verification methods increases. Formal verification of dynamic systems often involves analyzing whether a system is capable of reaching a certain area in the state space or not [3]. This connects formal verification of dynamic systems to reachability analysis.

Reachability analysis is commonly classified into forward and backward reachability regarding time and into scenarios in which the goal is to reach or avoid a certain area in the state space [2], [4]. Therefore, it can be used to check whether a system is safe, either by ensuring that it cannot reach unsafe states, or by ensuring that it is capable of reaching safe states.

Hamilton-Jacobi (HJ) reachability methods differentiate from other methods that can be used to compute reachable sets as they are compatible with non-linear continuous systems that are subject to input and state constraints under the influence of bounded disturbances [2]. To determine the reachable set, an HJ partial differential equation (PDE) needs to be solved. Therefore, state-of-the-art methods employ level set methods to numerically determine a viscosity solution [5] of the HJ PDE at the vertices of a grid that is used to discretize the state space [6]. According to [7], level set methods appear to be convergent for an increasing resolution of the grid. For this reason, a grid of a high resolution is favorable. If the reachable set needs to be determined for a long time horizon or if the velocity of the system dynamics is fast, the size of the grid needs to be large as well. A high resolution combined with a large size of the grid results in a considerable computational effort, as the computational effort is related to the number of vertices of the grid. Hence, suboptimal trade-offs often need to be taken leading to imprecise results.

To address this challenge, this contribution proposes a method for computing HJ reachable sets in a time and memory-efficient way compared to state-of-the-art methods. For that, the method proposed successively employs multiple adaptive grids (see the green boxes in Fig. 1) to solve the corresponding HJ PDE instead of employing one static grid as in the state-of-the-art (see the gray box in Fig. 1).

Throughout this contribution, the method is presented for the scenario of checking which states can be reached by a system despite the possibly worst disturbances. Therefore, the maximum forward reachable set (FRS) of the system is determined. The method can equivalently be utilized to compute backward reachable sets and reachable tubes and to verify that a system is able to avoid certain states.

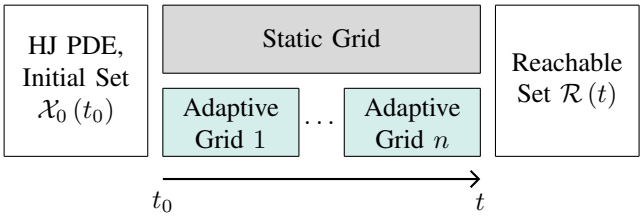


Fig. 1: Employing multiple adaptive grids instead of one static grid for computing the reachable set over time

A. Related Work

There exist methods for computing reachable sets that are customized for different classes of input constrained systems: linear, non-linear, and hybrid systems. Especially methods that exploit certain representations of sets are known to be

C. Bohn, M. Schwartz, and S. Hohmann are with the Institute of Control Systems (IRS) at the Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany. P. Reis is with the Forschungszentrum Informatik (FZI), 76131 Karlsruhe, Germany. The first two authors contributed equally. Corresponding author is Christopher Bohn, christopher.bohn@kit.edu.

computationally efficient, e.g. methods that represent sets using ellipsoids [8], zonotopes [9], or polynomial zonotopes [10]. Furthermore, methods that are based on flow-pipes [11] and methods that use the mean-value theorem [3] are prevalent. However, these methods are neither able to consider state constraints nor bounded disturbances in a worst-case manner. This separates them from HJ-based methods. Also, in contrast to the other methods, HJ-based methods allow the regarded sets to be of an arbitrary shape [2].

There are suitable numerical tools, as the level set methods [6], [12], [13], that can be used to compute viscosity solutions [5] of the HJ PDE that corresponds to the reachability problem. Furthermore, tools are available that implement these algorithms making them widely used in state-of-the-art contributions [2], [4], [14]–[17]: the Level Set Toolbox (toolboxLS) [18], that can conveniently be employed for reachability problems through the helperOC[1] Toolbox, and the Berkley Efficient API in C++ for Level Set methods (BEACLS[2]) Toolbox. However, the support of these toolboxes does not reduce the computational effort, as they compute the viscosity solution on one user-defined static grid.

As an alternative to numerical approaches, there are neural PDE solvers such as DeepReach [15] that can be employed to compute reachable sets. These methods are beneficial, as no grids need to be employed to discretize the state space. Therefore, the computational effort does not depend on the number of states of the system, but only on the shape of the reachable sets [15]. The preprocessing time needed for training the network is in the timescale of $15\,\mathrm{h}$ to $25\,\mathrm{h}$ [15]. Afterward, sets can be computed almost instantly. However, this method is not well suited for scenarios in which state constraints change unpredictably, as changing state constraints requires retraining the network. For instance, this is the case in the application of motion planning of mobile robots: to avoid collisions with other objects, constraints in the position states need to be incorporated. These state constraints change continuously and unpredictably if there are dynamic objects in the surroundings of the robot.

B. Contribution and Outline

This contribution proposes a time and memory-efficient method for computing the viscosity solution of HJ PDEs for application in reachability analysis. In contrast to state-of-the-art methods that compute the solution on one static grid, adaptive grids are employed that successively cover the relevant region of the state space. The method determines the size and the resolution of the grids based on the current volume expansion of the reachable set and the velocity of the flow field in the respective area. Changes between grids can be performed in an outer or inner approximative manner. Hence, the consistency of HJ reachable sets with almost analytical solutions demonstrated in [7] can be exploited for safety applications. The method can be combined with existing approaches that reduce computational efforts as [19], [20] to further improve computational efficiency. The method is demonstrated using the well-known example of a three-dimensional Dubins car system. The results show that using the proposed method, reachable sets are computed at higher accuracy compared to state-of-the-art approaches by saving a significant amount of time and memory.

The paper is structured as follows: section II outlines the background of HJ reachability, Section III describes the state-of-the-art method for numerically solving the corresponding HJ PDE. The proposed method is presented in Section IV and its computational efficiency is demonstrated in Section V. Section VI concludes the results of this contribution.

II. Problem Setup

A non-linear, time continuous system with the state vector $\boldsymbol{x} \in \mathbb{R}^{\mathrm{n}}$, the system input $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^{\mathrm{p}}$ and the disturbance $\boldsymbol{d} \in \mathcal{D} \subset \mathbb{R}^{\mathrm{q}}$ is considered. The dynamics of the system with respect to time t are described by

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{d}). \tag{1}$$

It is assumed that $\boldsymbol{f} : \mathbb{R}^{\mathrm{n}} \times \mathcal{U} \times \mathcal{D} \to \mathbb{R}^{\mathrm{n}}$ is uniformly continuous, bounded, and Lipschitz continuous in t for given signals $\boldsymbol{u}(\cdot)$ and $\boldsymbol{d}(\cdot)$. Furthermore, the control input signal $\boldsymbol{u}(\cdot)$ and the disturbance signal $\boldsymbol{d}(\cdot)$ are assumed to be drawn from the set of measurable functions as described in [6]:

$$\boldsymbol{u}(\cdot) \in \mathbb{U}(t) = \{\Theta : [0,t] \to \mathcal{U} : \Theta(\cdot) \text{ is measurable}\},$$
$$\boldsymbol{d}(\cdot) \in \mathbb{D}(t) = \{\Theta : [0,t] \to \mathcal{D} : \Theta(\cdot) \text{ is measurable}\}.$$

For a given initial state $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ at an initial time t_0, there exists a unique solution $\boldsymbol{\xi}_{\mathrm{f}}$ of (1) when considering a given input signal $\boldsymbol{u}(\cdot)$ and a given disturbance signal $\boldsymbol{d}(\cdot)$ that satisfies (1):

$$\boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)) : \quad [t_0, t] \to \mathbb{R}^{\mathrm{n}}, \tag{2}$$

with

$$\begin{aligned} &\dot{\boldsymbol{\xi}}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)) \\ &\quad = \boldsymbol{f}(\boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)), \boldsymbol{u}(t), \boldsymbol{d}(t)), \\ &\boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)) = \boldsymbol{x}. \end{aligned} \tag{3}$$

The goal of the reachability task is to compute the maximum FRS $\mathcal{R}^{\max}_{\mathcal{X}_0}(t)$ of (1) that can be reached by the system at time t when starting in a closed initial set of states $\mathcal{X}_0 \subset \mathbb{R}^{\mathrm{n}}$ at t_0, regardless of any disturbances that may occur.

This task corresponds to a differential game between two players, in which player I represents the input $\boldsymbol{u}$ and player II represents the disturbance $\boldsymbol{d}$. The strategy of player II is defined by a map γ restricting player II to non-anticipative strategies $\Gamma(\cdot)$:

$$\begin{aligned} \gamma \in \Gamma(t) \coloneqq \{\mathcal{Z} : \mathbb{U}(t) \to \mathbb{D}(t) \,|\, \\ \boldsymbol{u}(\tilde{t}) = \hat{\boldsymbol{u}}(\tilde{t}) \;\forall \tilde{t} \in [t_0, t] \\ \Rightarrow \mathcal{Z}[\boldsymbol{u}](\tilde{t}) = \mathcal{Z}[\hat{\boldsymbol{u}}](\tilde{t}) \;\forall \tilde{t} \in [t_0, t]\}. \end{aligned} \tag{4}$$

This restriction provides player II an informational advantage, as it cannot react differently on two inputs of player I

[1] https://www.github.com/HJReachability/helperOC
[2] https://www.github.com/HJReachability/beacls

until these two inputs are different: thus, player II knows the input of player I at each time until t, and therefore, it can adapt its own input accordingly.

In the context of the reachability task, this ensures that at each time t, the possibly worst disturbance is considered that shrinks the reachable set in the regarded case as much as possible. Consequently, potential disturbances are considered in a worst-case manner in the reachability task. The corresponding maximum FRS $\mathcal{R}_{\mathcal{X}_0}^{\max}(t)$ is defined as

$$\begin{aligned}\mathcal{R}_{\mathcal{X}_0}^{\max}(t) \coloneqq \{\boldsymbol{x} \in \mathcal{X} \,|\, \exists \boldsymbol{x}_0 \in \mathcal{X}_0, \exists \boldsymbol{u}(\cdot), \forall \boldsymbol{d}(\cdot) : \\ \boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \gamma[\boldsymbol{u}](\cdot)) = \boldsymbol{x}\}.\end{aligned} \tag{5}$$

HJ reachability methods use costs and the principle of dynamic programming to determine $\mathcal{R}_{\mathcal{X}_0}^{\max}(t)$ [6]. The cost C at time t that is associated with starting in $\boldsymbol{x}_0$ at $t_0 \leq t$ and applying $\boldsymbol{u}(\cdot)$ and $\boldsymbol{d}(\cdot)$ to a system $\boldsymbol{f}$ can be defined using a Bolza cost function. It consists of the running Lagrange cost $l : \mathbb{R}^{\mathrm{n}} \to \mathbb{R}$ and the terminal Mayer cost $m : \mathbb{R}^{\mathrm{n}} \to \mathbb{R}$:

$$\begin{aligned}&C(\boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot))) \\ &\quad = \int_{t_0}^{t} l\left(\boldsymbol{\xi}_{\mathrm{f}}(\tau; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot))\right) \mathrm{d}\tau \\ &\qquad + m\left(\boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot))\right).\end{aligned} \tag{6}$$

In reachability tasks, the effort for reaching a state is not of interest and therefore, $l = 0$ in (6). To obtain a suitable measure to be used as the terminal Mayer cost, the initial set $\mathcal{X}_0$ is defined as the zero sublevel set of a bounded and Lipschitz continuous function $g : \mathbb{R}^{\mathrm{n}} \to \mathbb{R}$ with

$$\mathcal{X}_0 = \{\boldsymbol{x} \in \mathcal{X} \,|\, g(\boldsymbol{x}) \leq 0\}. \tag{7}$$

This is always possible as g can be chosen as the signed distance function to the boundary of the initial set $\partial\mathcal{X}_0$. Using $g(\boldsymbol{x_0})$ as the terminal Mayer cost, the cost $C\left(\boldsymbol{\xi}_{\mathrm{f}}\left(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)\right)\right)$ of a trajectory that ends in state $\boldsymbol{x} = \boldsymbol{\xi}_{\mathrm{f}}(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot))$ is defined as $g\left(\boldsymbol{x}_0\right)$ of the state $\boldsymbol{x}_0$ in which the regarded trajectory $\boldsymbol{\xi}_{\mathrm{f}}\left(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)\right)$ starts. Therefore,

$$\tilde{C}\left(\boldsymbol{\xi}_{\mathrm{f}}\left(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \boldsymbol{d}(\cdot)\right)\right) = g(\boldsymbol{x}_0) \tag{8}$$

is used. The maximum FRS $\mathcal{R}_{\mathcal{X}_0}^{\max}(t)$ according to (5) is associated with player I aiming to minimize this cost with $\boldsymbol{u}\left(\cdot\right)$ and player II aiming to maximize this cost with $\boldsymbol{d}\left(\cdot\right)$. So player I tries to steer the system to the desired state $\boldsymbol{x}$ at time t, starting in the cheapest possible state $\boldsymbol{x}_0$, from which player I is able to reach $\boldsymbol{x}$, while player II tries to hinder player I from doing so. Using the non-anticipative strategy (4), the corresponding value function $\phi\left(\boldsymbol{x}, t\right)$ is defined as

$$\begin{aligned}\phi(\boldsymbol{x}, t) = \sup_{\gamma \in \Gamma(t)} \inf_{\boldsymbol{u}(\cdot) \in \mathbb{U}(t)} \\ \tilde{C}\left(\boldsymbol{\xi}_{\mathrm{f}}\left(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \gamma[\boldsymbol{u}](\cdot)\right)\right),\end{aligned} \tag{9}$$

with $\boldsymbol{\xi}_{\mathrm{f}}\left(t; t_0, \boldsymbol{x}_0, \boldsymbol{u}(\cdot), \gamma[\boldsymbol{u}](\cdot)\right) = \boldsymbol{x}$ and $t_0 \leq t$. Using dynamic programming, it can be shown that $\phi(x, t)$ is the viscosity solution of the Hamilton-Jacobi-Isaacs PDE:

$$\frac{\partial \phi(\boldsymbol{x}, t)}{\partial t} + H\left(\boldsymbol{x}, \nabla\phi\left(\boldsymbol{x}, t\right)\right) = 0 \tag{10}$$

with the initial value

$$\phi(\boldsymbol{x}, 0) = g(\boldsymbol{x}). \tag{11}$$

The associated Hamiltonian H is given by

$$\begin{aligned}H(\boldsymbol{x}, \nabla\phi\left(\boldsymbol{x}, t\right)) = \min_{\gamma \in \Gamma(t)} \max_{\boldsymbol{u} \in \mathcal{U}} \\ \nabla\phi\left(\boldsymbol{x}, t\right)^{\top} f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{d}).\end{aligned} \tag{12}$$

The value function $\phi(\boldsymbol{x}, t)$ represents the distance of $\partial\mathcal{X}_0$ to the initial state $\boldsymbol{x}_0$ from which the trajectory starts, that ends in $\boldsymbol{x}$ and is optimal w.r.t. (9). Consequently, according to (7), $\phi(\boldsymbol{x}, t) \leq 0$ if the optimal $\boldsymbol{x}_0 \in \mathcal{X}_0$ and $\phi(\boldsymbol{x}, t) > 0$, if the optimal $\boldsymbol{x}_0 \notin \mathcal{X}_0$. Finally, the maximum FRS is represented by the zero sublevel set of $\phi(\boldsymbol{x}, t)$ with $\phi(\boldsymbol{x}, t) = 0$ being the boundary of the maximum FRS $\partial\mathcal{R}_{\mathcal{X}_0}^{\max}(t)$.

III. Numerical Solution

Numerical solution approaches for solving (10) are favorable over analytical approaches, as analytical approaches depend on the characteristic of the system, and hence, are generally not guaranteed to succeed. The solution of (10) may contain shocks and rarefactions that can lead to kinks resulting in discontinuous derivatives [6]. In such a case, a classical HJ PDE solution may not exist. Therefore, viscosity solutions [5] create great practical value. Level set methods are designed especially for computing approximations of viscosity solutions of (10). The approach of these methods is to propagate boundaries in a flow field. In the presented case, $\partial\mathcal{R}_{\mathcal{X}_0}^{\max}(t)$, represented by $\phi(\boldsymbol{x}, t) = 0$, is propagated in the flow field defined by the system dynamics (1).

As the state space is defined in $\mathbb{R}^{\mathrm{n}}$, $\phi(\boldsymbol{x}, t)$ is computed on an n-dimensional uniform, equidistant, Cartesian grid $\mathbb{G}^{\mathrm{n}}$. An n-dimensional uniform, equidistant, Cartesian grid $\mathbb{G}^{\mathrm{n}}$ is defined by the minimum $x_{i,\min}$ and maximum $x_{i,\max}$ value of each dimension i, $\forall i \leq n$, and the number of vertices N_i in each dimension. Based on this, the resolution of the grid $\mathbb{G}^{\mathrm{n}}$ in dimension i is defined as

$$\Delta x_i = \frac{x_{i,\max} - x_{i,\min}}{N_i - 1}. \tag{13}$$

The following describes the methods implemented in [18] based on [6]. A benefit of level set methods is that the three parts of (10), $\nabla\phi\left(\boldsymbol{x}, t\right)$, $H\left(\boldsymbol{x}, \nabla\phi\left(\boldsymbol{x}, t\right)\right)$, and $\frac{\partial\phi(\boldsymbol{x}, t)}{\partial t}$ can be approximated separately using different techniques: the spatial derivative $\nabla\phi\left(\boldsymbol{x}, t\right)$ is determined using a weighted essentially non-oscillatory (WENO) scheme providing sufficient accuracy for computing reachable sets. The Hamiltonian $H\left(\boldsymbol{x}, \nabla\phi\left(\boldsymbol{x}, t\right)\right)$ is approximated using the Lax-Friedrichs approximation, and the time derivative $\frac{\partial\phi(\boldsymbol{x}, t)}{\partial t}$ is computed based on explicit total variation diminishing Runge-Kutta schemes. Thereby, the time step Δt is restricted by the Courant-Friedrichs-Lewy (CFL) condition [21]. This is a necessary condition for the convergence of the numerical scheme. It states that the numerical domain of dependence must include the physical domain of dependence, such that

$$\mathrm{CFL} = \frac{\Delta t}{\Delta \boldsymbol{x}} \left| U \right| \leq 1, \tag{14}$$

with the velocity U of the flow of the dynamical system.

As (10) is defined over all $\mathbb{R}^n$, there is no physical boundary. However, when numerically solving (10) on $\mathbb{G}^n$, boundary conditions are required. For periodic dimensions such as the orientation of a system, one complete period can be included. Hence, the boundary conditions can be chosen periodically. For non-periodic dimensions, boundary conditions need to be enforced. Therefore, [6] proposes Neumann boundary conditions as they do not disturb the solution globally. However, as they are physically incorrect, they cause local disturbances in the solution close to the boundary of the grid $\partial\mathbb{G}^n$. Hence, it must be ensured that the distance of the zero level $\phi(\boldsymbol{x}, t) = 0$ to the boundary of the grid $\partial\mathbb{G}^n$ is large enough.

IV. Adaptive Grids

To reduce the computational effort of the numerical solution approach described in Section III, this contribution proposes a method to compute $\phi(\boldsymbol{x}, t)$ successively on small adaptive grids, that are adjusted to the volume expansion of the reachable set and to the velocity of the flow field. Transferring the solution from one grid to another can be conducted in an outer or inner approximative manner. The computational benefit of employing adaptive grids originates from computing the solution on small grids and repeatedly transferring intermediate solutions to another grid is more efficient than computing the solution on one large static grid.

A. Construction of Adaptive Grids

The size of the adaptive grids in dimension i in the state space is determined based on the volume expansion $\mathrm{ve}(\mathcal{R}_i) = \max(\partial\mathcal{R}_i) - \min(\partial\mathcal{R}_i)$ of the reachable set $\mathcal{R}$ in dimension i at a grid change and the flow velocities

$$\alpha_i = \max_{p_i} \left| \frac{\partial H}{\partial p_i} \right|, \quad \text{with } p_i = \frac{\partial \phi(\boldsymbol{x}, t)}{\partial x_i}, \tag{15}$$

of the underlying dynamical system in the respective dimension. With the center of the reachable set $\mathrm{c}(\mathcal{R}_i)$ in dimension i and a user-defined expansion factor η, the minimum $x_{i,\min}$ and maximum $x_{i,\max}$ values of the subsequent grid in the respective dimension are computed by

$$x_{i,\max} = \mathrm{c}(\mathcal{R}_i) + \mathrm{ve}(\mathcal{R}_i) \cdot \eta \cdot (1 + \max(0, \alpha_i)), \tag{16a}$$

$$x_{i,\min} = \mathrm{c}(\mathcal{R}_i) - \mathrm{ve}(\mathcal{R}_i) \cdot \eta \cdot (1 - \min(0, \alpha_i)). \tag{16b}$$

This ensures that the area of the state space covered by the subsequent grid is reasonable for the impending computation steps. The resolution of the subsequent grid is chosen to be equal to the one of the previous grid. However, it is limited by a maximum number of vertices $N_{i,\max}$ in one dimension of the grid.

A grid change is triggered if the smallest norm $\mathrm{d}_{\min}$ between the boundary of the reachable set $\partial\mathcal{R}$ and the boundary of the grid $\partial\mathbb{G}^n$

$$\mathrm{d}_{\min}(\partial\mathcal{R}, \partial\mathbb{G}^n) := \min_{\boldsymbol{x}_s \in \partial\mathcal{R}, \boldsymbol{x}_g \in \partial\mathbb{G}^n} \|\boldsymbol{x}_s - \boldsymbol{x}_g\|, \tag{17}$$

becomes smaller than a predefined limit $d_{\min}$. Complying with the CFL condition (14) ensures, that $\partial\mathcal{R}$ does not propagate farther than the smallest resolution of the grid in one dimension $\min \Delta x_i$ in one time propagation step Δt. To ensure that there are enough vertices within this limit for determining the gradient at the boundary of the grid, a parameter $\zeta \in \mathbb{N}$ is used, such that $d_{\min} \leq \zeta \Delta x_i$. Appropriately choosing these parameters ensures that the distance to the boundary of the grid is large enough such that disturbances caused by physically incorrect boundary conditions become insignificant.

B. Grid Changes

Generally, the vertices of the subsequent grid do not overlap with the vertices of the previous grid. For transferring the value function $\phi(\boldsymbol{x}, t)$, the values at the vertices of the subsequent grid that are located inside the previous grid (illustrated by the solid black box in Fig. 2) are interpolated linearly. This applies to the vertices between L_-^i and L_+^i in Fig. 2. The values at the vertices in the area of the state space that is not covered by the previous grid are set to the value of the proximate vertex of the previous grid. These values are positive, as the distance between the reachable set and the boundary of a grid is designed to be large enough. This approach turns out to avoid oscillations in the subsequent computation steps.

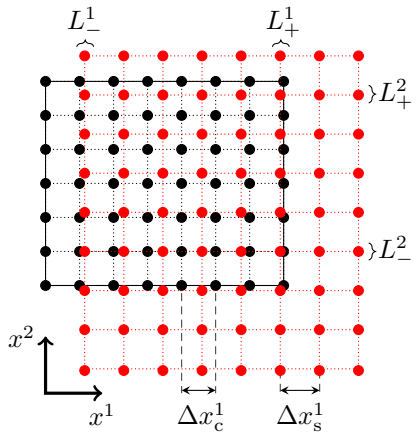


Fig. 2: At a grid change, the values at the vertices of the previous grid • need to be transferred to the vertices of the subsequent grid •

C. Inner Approximative Grid Changes

To be able to exploit the consistency of HJ reachable sets with almost analytical solutions demonstrated in [7] using the method proposed, it needs to be ensured that no non-reachable states are added at a grid change. Therefore, interpolated values of $\phi_{\mathrm{int}}(\boldsymbol{x}, t)$ at a grid change need to be larger or equal to the actual value of $\phi(\boldsymbol{x}, t)$ in the vicinity of the zero level. Although $\phi(\boldsymbol{x}, t_0)$ is a distance function by definition, the evolution of $\phi(\boldsymbol{x}, t)$ generally distorts this property [6]. To ensure $\phi_{\mathrm{int}}(\boldsymbol{x}, t) \geq \phi(\boldsymbol{x}, t)$ in the vicinity of the zero level, Proposition 1 is used:

Proposition 1: Inner Approximative Grid Change Altering the value function by the minimum value of an adjacent vertex to the zero level $\phi(\boldsymbol{x}^*, t)$

$$\tilde{\phi}(\boldsymbol{x}, t) = \phi(\boldsymbol{x}, t) + |\phi(\boldsymbol{x}^*, t)|, \tag{18}$$

and interpolating $\tilde{\phi}(\boldsymbol{x},t)$ instead of $\phi(\boldsymbol{x},t)$ ensures, that the set represented by the interpolated values $\tilde{\phi}_{\text{int}}(\boldsymbol{x},t) \leq 0$ is an inner approximation of the set represented by $\phi(\boldsymbol{x},t) \leq 0$.

Proof: Consider an arbitrary cell of a grid, that includes the zero level (see e.g. the gray cell in Fig. 3). Furthermore, consider two vertices, $\boldsymbol{x}_{\text{i}} \in \mathcal{X}_0$ and $\boldsymbol{x}_{\text{o}} \notin \mathcal{X}_0$, that are both part of this cell, and therefore, are in the vicinity of the zero level. The initial value function $\phi(\boldsymbol{x},t_0)$ is constructed as a signed distance function: its values are negative inside the initial set and positive outside the initial set. Therefore, the value function increases along a straight line from $\boldsymbol{x}_{\text{i}}$ to $\boldsymbol{x}_{\text{o}}$. This property holds for $\phi(\boldsymbol{x},t)\,\forall t \geq t_0$, as $\phi(\boldsymbol{x},t)$ is propagated in a uniformly continuous flow field defined by (1). Therefore, the minimum value of all adjacent vertices to the zero level $\phi(\boldsymbol{x}^*,t)$ is a lower bound of all linearly interpolated values of the subsequent grid in the vicinity of the zero level. Altering the value function according to (18) ensures, that in the vicinity of the zero level, all linearly interpolated values $\tilde{\phi}_{\text{int}}(\boldsymbol{x},t)$ are larger or equal to the actual value of the value function $\phi(\boldsymbol{x},t)$. Consequently, the set represented by $\tilde{\phi}_{\text{int}}(\boldsymbol{x},t)$ is an inner approximation of the set represented by $\phi(\boldsymbol{x},t)$. □

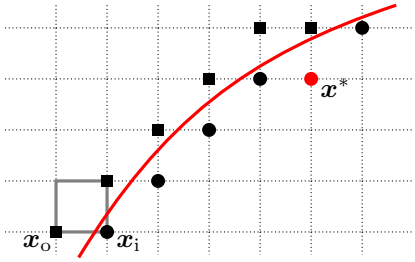


Fig. 3: Two-dimensional grid including the zero level in red, the adjacent vertices to the zero level (positive ■, negative •), the minimum value adjacent vertex •, and an example cell that contains the zero level in gray

The approach for performing an outer approximative grid change is equivalent to the inner approximative case, whereas the value function is altered by the maximum value of a vertex adjacent to the zero level $\phi(\boldsymbol{x}^\star,t)$:

$$\tilde{\phi}(\boldsymbol{x},t) = \phi(\boldsymbol{x},t) - \phi(\boldsymbol{x}^\star,t). \tag{19}$$

V. Numerical Example

To demonstrate the proposed method, the dynamics of a Dubins car model are employed. The system consists of three states: x and y represent the position of the model on a two-dimensional surface, and θ represents the heading of the model relative to the x-axis. The heading can be influenced using the system input u that represents the yaw rate $\dot{\theta}$. The parameter v is the translational velocity of the model in the direction of its heading. The dynamics are described by

$$\dot{\boldsymbol{x}} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} v \cdot \cos(\theta) \\ v \cdot \sin(\theta) \\ u \end{bmatrix} = \boldsymbol{f}(\boldsymbol{x},u). \tag{20}$$

The proposed method is utilized to compute the maximum FRS $\mathcal{R}^{\max}_{\mathcal{X}_0}(t)$ for $t = 1\,\text{s}$ starting at $t_0 = 0\,\text{s}$ with the initial set of states $\mathcal{X}_0$ being represented by a sphere around the origin with the radius $R = 0.12$:

$$g(\boldsymbol{x}) = \phi(\boldsymbol{x},0) = \sqrt{x^2+y^2+\theta^2} - R. \tag{21}$$

The solution that is used as ground truth in this contribution is computed on a static grid of $200 \times 200 \times 200$ vertices. This is the largest number of vertices in a grid used in [7]. Also, according to [7], the accuracy of resulting sets increases with an increasing resolution of the grid. Therefore, it is reasonable to compare sets, that are computed on grids of lower resolution with this ground truth. Furthermore, a static grid of $100 \times 100 \times 100$ vertices is used as benchmark, as this number of vertices is used as ground truth in [17]. The boundary values of all static grids are $\boldsymbol{x}_{\min,\text{sg}} = [-1\ -2\ -\pi]^\top$ and $\boldsymbol{x}_{\max,\text{sg}} = [2\ 2\ \pi]^\top$.

All results are computed on a 2.9 GHz Quad-Core Intel i7 processor using the *ToolboxLS* [18] with MATLAB 2020b. The accuracy of a set is measured using the Hausdorff-Distance $\mathrm{d_H}$ of the set to the ground truth solution. The parameters of the method are chosen according to Table I.

TABLE I: Parameters used in the numerical example

Parameter	Value
ζ	4
$d_{\min}$	$\zeta \Delta x_i$
η	1.2

For computing the solutions, the velocity is set to $v = 1$ and the input of the system is constraint to the interval $u \in [-0.8, 0.8]$. The initial set $\mathcal{X}_0$ represented by $\phi(\boldsymbol{x},0)$ is transferred to a grid of $30 \times 30 \times 30$ vertices with the resolution $\Delta\boldsymbol{x}_{\text{init,ag}} = [0.02\ 0.03\ 0.03]^\top$ in the case of adaptive grids and to a static grid of the respective different sizes with $\boldsymbol{x}_{\min,\text{sg}}$ and $\boldsymbol{x}_{\max,\text{sg}}$ equal to the benchmark case.

In Fig. 4, the evolution of the FRS $\mathcal{R}^{\max}_{\mathcal{X}_0}(t)$ from $t_0 = 0\,\text{s}$ until $t = 1\,\text{s}$ is illustrated. Grid boundaries are depicted by dotted lines, the initial grid is black. In this example, there are four grid changes. Two changes are triggered by the extension of an intermediate set in the x-dimension, one results in the cyan grid and the other one results in the purple grid. Another grid change is triggered by the volume expansion of an intermediate set in the θ-dimension resulting in the olive grid, and the last change is triggered by the volume expansion of an intermediate set in the y-dimension resulting in the green grid.

For comparing the adaptive grids method with the static grid method, the number of vertices used for computing the static grid solution has been set to the maximum number of vertices to be used in the case of adaptive grids. The resulting computation times, maximum memory usage, average memory usage, and the accuracy of the resulting set represented by $\mathrm{d_H}$ are depicted in Fig. 5.

In Fig. 5a, it can be seen that in the case of $N_i \leq 60$ vertices per dimension, the computation time of the adaptive grids method is higher than in the static grid case. This is due to the computational overhead of the operations related to the adaptive grids method. However, for $N_i > 60$ vertices

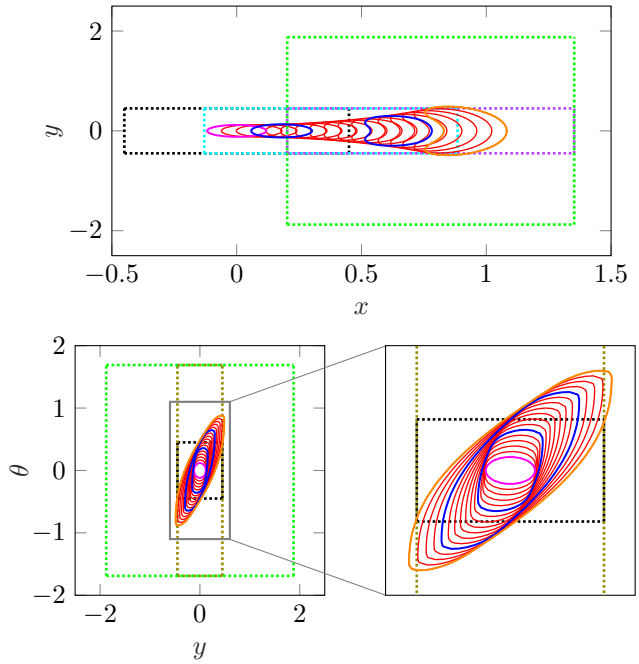


Fig. 4: The evolution of the FRS determined using adaptive grids is depicted in red, $\mathcal{X}_0$ is depicted in magenta and $\mathcal{R}(t = 1\,\mathrm{s})$ in orange, the boundaries of the adaptive grids are depicted by dashed lines, intermediate sets that triggered a grid change are depicted in royal blue, projection onto the x-y-plane (top) and onto the y-θ-plane (bottom)

per dimension, the computation time is lower in the adaptive grids case than in the static grid case.

In terms of memory usage, it can be seen in Fig. 5b, that the adaptive grids method does not exploit the maximum number of allowed vertices in all dimensions. Therefore, the maximum memory usage and the average memory usage are both smaller than in the static grid case. It can also be seen, that the memory usage in the case of adaptive grids grows significantly slower than in the static grid case.

In Fig. 5c, the accuracy of the resulting reachable set compared to the ground-truth solution is illustrated. It can be seen that the adaptive grids method achieves solutions of higher accuracy, especially in the case of only a few vertices per dimension. In the case of more vertices per dimension, the accuracy of the solution computed on adaptive grids becomes similar to the one of the static grid solution.

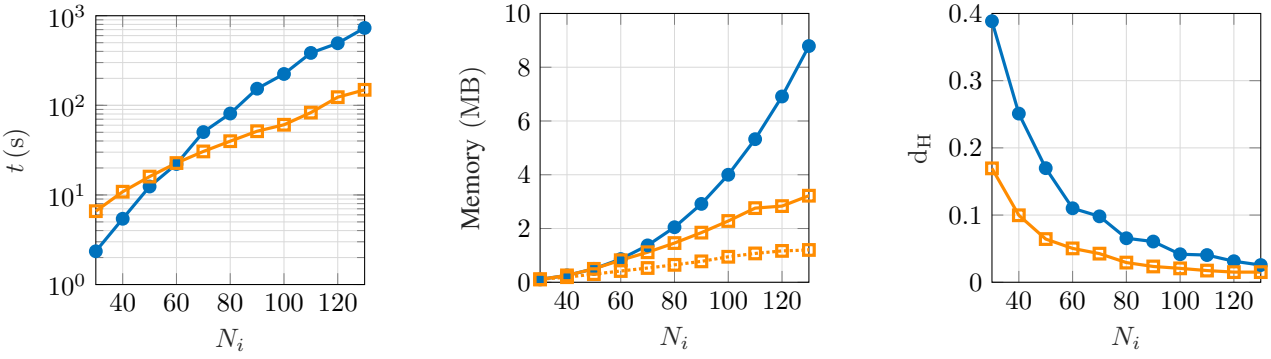
Comparing the adaptive grids method to the static grid method with $N_i = 100$ vertices per dimension, as used in [6], [17], the computation time of the adaptive grids method is 60 s compared to 223 s as of the static grid method. This is a reduction of 73 %. The maximum memory used by the adaptive grids method is 2.28 MB and the average memory used is 0.95 MB. In comparison, the static grid method constantly uses 4 MB. Consequently, the proposed method saves a maximum memory usage of 43 % and an average memory usage of 76 % in this scenario. The accuracy of the proposed method is higher than in the static grid case in this scenario, $\mathrm{d_H} = 0.0207$ (adaptive) and $\mathrm{d_H} = 0.0417$ (static).

The reachable sets computed by both methods during a limited time of 60 s and the ground truth solution are depicted in Fig. 6. It can be seen, that the reachable set computed with the adaptive grids method is closer to the ground truth solution than the static grid solution. Therefore, the proposed method results in more accurate sets in the presented scenario, especially in the regions of high curvature.

In Fig. 7, an inner approximative grid change is presented. The set depicted in royal blue is transferred to the subsequent grid in an inner approximative manner resulting in the set depicted in red.

VI. Conclusion

In contrast to state-of-the-art methods that compute reachable sets by numerically determining the viscosity solution of the corresponding Hamilton-Jacobi (HJ) partial differential equation (PDE) on a large static grid, the proposed method successively utilizes smaller adaptive grids to determine the solution only in the area of the state space that is relevant at a certain time. The adaptive grids are automatically adjusted to the volume expansion of the reachable set and the velocity of the flow field. The method ensures that the distance between the boundary of the reachable set and the boundary of the respective grid is large enough to prevent the solution from



(a) Computation time (b) Memory usage (c) Accuracy

Fig. 5: Comparison of computation time, maximum (solid) and average (dotted) memory usage, and accuracy of the resulting set for different numbers of vertices per dimension N_i in the case of one static grid and different numbers of maximum vertices per dimension N_i in the case of adaptive grids

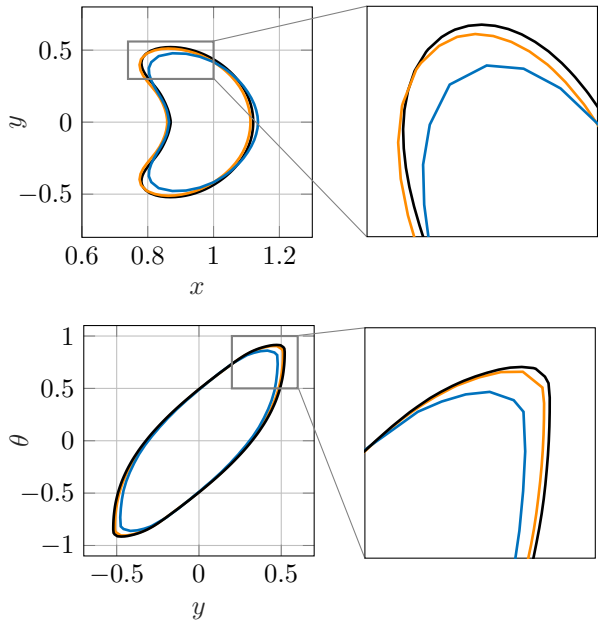


Fig. 6: Comparison of the FRS computed on a static grid in blue, on adaptive grids in orange and the ground truth solution in black, projection onto the x-y-plane (top) and onto the y-θ-plane (bottom)

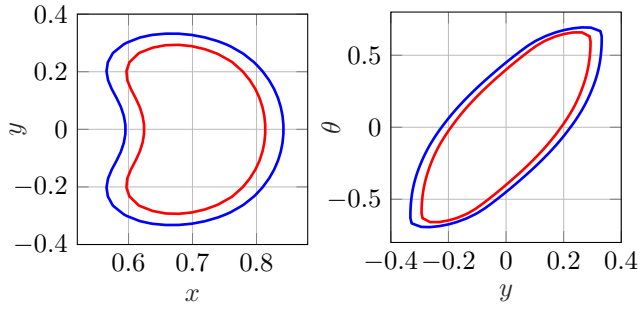


Fig. 7: Inner approximative grid change from the set depicted in royal blue to the set depicted in red, projection onto the x-y-plane (left) and onto the y-θ-plane (right)

being biased by physically incorrect boundary conditions. Changes between grids can be performed in an outer or inner approximative manner. The numerical example demonstrates the application of the method for computing the forward reachable set (FRS) of the well-known Dubins car model. In the presented scenario, the results show, that by utilizing adaptive grids, the FRS is computed 73 % faster, saving 43 % of maximum and 76 % of average memory usage while increasing the accuracy compared to computing the FRS on a static grid. Approaches that further reduce the area in the state space, in which the solution of the respective HJ PDE needs to be computed, are promising to reduce the computational effort of HJ reachability analysis further.

References

[1] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, no. 7, 2005.

[2] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Conf. Decision Control*, IEEE, 2017.

[3] E. Goubault and S. Putot, "Inner and outer reachability for the verification of control systems," in *Proc. 22nd ACM Int. Conf. on Hybrid Systems: Computation and Control*, ser. HSCC '19, Montreal, Quebec, Canada: Association for Computing Machinery, 2019.

[4] J. F. Fisac, M. Chen, C. J. Tomlin, and S. S. Sastry, "Reach-avoid problems with time-varying dynamics, targets and constraints," in *Proc. 18th Int. Conf. on Hybrid Systems: Computation and Control*, A. Girard and S. Sankaranarayanan, Eds., New York, NY, USA: ACM, 2015.

[5] M. G. Crandall and P.-L. Lions, *Viscosity solutions of hamilton-jacobi equations*, 1983.

[6] I. M. Mitchell, "Application of level set methods to control and reachability problems in continuous and hybrid systems," 2002.

[7] ——, "Games of two identical vehicles," Stanford University, SUDAAR #740, Tech. Rep., 2001.

[8] A. A. Kurzhanskiy and P. Varaiya, "Ellipsoidal techniques for reachability analysis of discrete-time linear systems," *IEEE Transactions on Automatic Control*, no. 1, 2007.

[9] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *2008 47th IEEE Conf. Decision Control*, 2008.

[10] N. Kochdumper and M. Althoff, "Computing non-convex inner-approximations of reachable sets for nonlinear continuous systems," in *2020 59th IEEE Conf. Decision Control*, 2020.

[11] X. Chen, S. Sankaranarayanan, and E. Ábrahám, "Under-approximate flowpipes for non-linear continuous systems," in *2014 Formal Methods in Computer-Aided Design*, 2014.

[12] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. National Academy of Sciences of the United States of America*, no. 4, 1996.

[13] S. Osher, *Level Set Methods and Dynamic Implicit Surfaces*, ser. Appl. Mathematical Sciences. New York, NY: Springer-Verlag New York Inc, 2003.

[14] K. Margellos and J. Lygeros, "Hamilton–jacobi formulation for reach–avoid differential games," *IEEE Transactions on Automatic Control*, no. 8, 2011.

[15] S. Bansal and C. J. Tomlin, "Deepreach: A deep learning approach to high-dimensional reachability," in *2021 IEEE Int. Conf. Robot. Autom.*, IEEE, 2021.

[16] M. Chen *et al.*, "Fastrack: A modular framework for real-time motion planning and guaranteed safe tracking," *IEEE Transactions on Automatic Control*, no. 12, 2021.

[17] A. Lin and S. Bansal, *Generating formal safety assurances for high-dimensional reachability*, 2022.

[18] I. M. Mitchell and J. A. Templeton, *A toolbox of hamilton-jacobi solvers for analysis of nondeterministic continuous and hybrid systems*, Zurich, Switzerland, 2005.

[19] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of reachable sets and tubes for a class of nonlinear systems," *IEEE Transactions on Automatic Control*, no. 11, 2018.

[20] S. L. Herbert, S. Bansal, S. Ghosh, and C. J. Tomlin, "Reachability-based safety guarantees using efficient initializations," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, IEEE, 2019.

[21] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen differenzengleichungen der mathematischen physik," *Mathematische Annalen*, 1928.