2025 IEEE 64th Conference on Decision and Control (CDC)
December 10-12, 2025. Rio de Janeiro, Brazil

Threshold Strategy for Leaking Corner-Free Hamilton-Jacobi Reachability with Decomposed Computations

Chong He, Mugilan Mariappan, Keval Vora and Mo Chen

2025 IEEE 64th Conference on Decision and Control (CDC) | 979-8-3315-2627-6/25/$31.00 ©2025 IEEE | DOI: 10.1109/CDC57313.2025.11312331

***Abstract*— Hamilton-Jacobi (HJ) Reachability is widely used to compute value functions for states satisfying specific control objectives. However, it becomes intractable for high-dimensional problems due to the curse of dimensionality. Dimensionality reduction approaches are essential for mitigating this challenge, whereas they could introduce the "leaking corner issue", leading to inaccuracies in the results. In this paper, we define the "leaking corner issue" in terms of value functions, propose and prove a necessary condition for its occurrence. We then use these theoretical contributions to introduce a new local updating method that efficiently corrects inaccurate value functions while maintaining the computational efficiency of the dimensionality reduction approaches. We demonstrate the effectiveness of our method through numerical simulations. Although we validate our method with the self-contained subsystem decomposition (SCSD), our approach is applicable to other dimensionality reduction techniques that introduce the "leaking corners".**

I. Introduction

Nonlinear systems control technologies have received growing interest in recent years and have many important applications [1]. The applicable fields include legged robots [2], safe collaboration between human and robotic systems [3], and autonomous vehicles [4], especially when including tasks ensuring safety by avoiding a failure set (safety task) and successfully reaching a goal set (liveness task). We generally call the failure sets and goal sets the target sets in our study.

Reachability analysis is generally used to ensure the safety and liveness of nonlinear systems. Researchers have developed various methods to analyze reachability [5]–[7], among which Hamilton-Jacobi (HJ) reachability excels at handling general nonlinear dynamics. HJ reachability formulates an initial value function based on the signed distance function of a target (goal or obstacle) and updates it backward in time using optimal control. This value function provides insights into the robot's capabilities for achieving tasks. However, HJ reachability suffers from the curse of dimensionality. As the system's dimensionality increases, computational complexity grows exponentially, making high-dimensional problems computationally infeasible.

Various approaches have been proposed to address the curse of dimensionality. These include linearization of nonlinear dynamics [8], [9], using hopf-lax formula [10]–[12], value function over- or under-approximation [13]–[15], reinforcement learning techniques [16]–[19], warm-starting with specific initializations [20], and paralleling the computation [21], [22]. Additionally, methods for decoupling or reducing high-order dynamics have been explored [23], [24], including the method for composition into self-contained subsystems (SCSs) [25], [26].

High-order dynamics decoupling methods are particularly advantageous due to their inherent parallelizability and immediate reduction of dimensionality. Compared to other computational acceleration methods, they offer significant computational time reduction while preserving the key system properties. It is achieved by computing the sub-value functions in low-dimensional subspaces and reconstructing the value function in full-dimensional space. However, in certain problem formulations, the value function reconstructed in full-dimensional space will deviate from the value function directly computed from the full-dimensional system. As a result, it can fail to guarantee liveness or safety for some states. This phenomenon is known as the "leaking corner issue." [26], [27]

A previous method for detecting the "leaking corners" involves computing the admissible control set in low-dimensional subsystems and comparing them to identify affected states [28]. However, it is currently restricted to the self-contained subsystem decomposition (SCSD) method and only applies to cases with scalar control inputs.

In this paper, we provide:

- A formal definition of "leaking corners" from the perspective of value functions.
- A novel detection method capable of handling both scalar and vector control inputs without additional computation, which is also general to methods with decomposed computations.
- A local updating procedure that ensures accurate results even without complete knowledge of all "leaking corners," while maintaining computational efficiency.

II. Background

A. System Dynamics

Consider the following control-affine dynamical system:

$$\frac{dz}{dt} = \dot{z} = f(z) + g(z)^\top u, \quad t \leq 0 \tag{1}$$

where $z \in \mathcal{Z} \subseteq \mathbb{R}^n$ denotes the system state within a state space, $u \in \mathcal{U} \subset \mathbb{R}^m$ denotes the control input. We assume that the control space $\mathcal{U}$ is given by the constraint

$$c(u) = \|\alpha \odot u\|_\beta - \overline{u} \leq 0. \tag{2}$$

*This work was supported by the Canada CIFAR AI Chairs and NSERC Discovery Grants Programs.

All authors are with the school of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada {chong_he, mmariapp, keval, mochen}@sfu.ca.

where the weight $\alpha \in \mathcal{A} \subseteq \mathbb{R}^m$, the norm $\beta \geq 1$, and the constraint value $\overline{u} \geq 0$. The symbol "$\odot$" represents the Hadamard product, which takes two matrices of the same dimensions as input and produces a matrix containing the multiplication of the corresponding elements in the inputs.

We assume the control function $u(\cdot) : [t, 0] \mapsto \mathbb{U}$ is drawn from the set of measurable functions. We also assume that $f : \mathcal{Z} \mapsto \mathbb{R}^n$ and $g : \mathcal{Z} \mapsto \mathbb{R}^{n \times m}$ are such that the dynamics (1) uniformly continuous, bounded, and Lipschitz continuous in z.

B. Value Functions in Hamilton-Jacobi Reachability

Hamilton Jacobi (HJ) reachability analysis is an optimal control problem used to analyze the liveness and safety properties of dynamical systems. The cost function $\ell : \mathbb{R}^n \mapsto \mathbb{R}$ of the optimal control problem is designed such that its 0 sub-level set is the goal (or failure) set, generally called the target set: $\mathcal{T} = \{z : \ell(z) \leq 0\}$. A common choice of the cost function is the signed distance function to the set $\mathcal{T}$.

The value function is the solution of the terminal value HJB-PDE

$$D_t V(z,t) + H(z, u, D_z V(z,t)) = 0 \tag{3}$$

with Hamiltonian for liveness and safety cases

$$H_R(z,u,p) = \min_{u \in \mathcal{U}} \{p^\top f(z) + p^\top g(z)^\top u\} \tag{4a}$$

$$H_A(z,u,p) = \max_{u \in \mathcal{U}} \{p^\top f(z) + p^\top g(z)^\top u\} \tag{4b}$$

where D_t and D_z represent the derivative with respect to t and z respectively. The boundary condition is given by a final value function $V(z, 0) = \ell(z)$, and the value function $V(z,t)$ can be computed via applying dynamic programming backward in time until $t < 0$. Whenever necessary, we will add the subscript of V, and use $V_R(z,t)$ and $V_A(z,t)$ respectively to represent the value functions for liveness and safety problems.

The optimal control u^* is obtained as follows:

$$u_R^* = \arg\min_{u \in \mathcal{U}} H_R(z, u, D_z V_R(z,t)), \tag{5a}$$

$$u_A^* = \arg\max_{u \in \mathcal{U}} H_A(z, u, D_z V_A(z,t)). \tag{5b}$$

We will omit the subscript of u^* when convenient and use u^* to denote the optimal control obtained from solving Eq. (5).

C. Computational Dimensionality Reduction

Numerically, the HJB PDE (3) is solved on a discrete grid; therefore, the computation scales exponentially with state dimension. This motivates the use of dimensionality reduction methods to reduce computational costs. Reducing computational dimensionality while maintaining exact global results is challenging. The low-dimensional subsystems of *coupled* nonlinear systems are defined below.

In this paper, for simplicity and clarity of presentation, we assume that there are two subsystems, but the results presented in the paper generalize to an arbitrary number of subsystems.

Definition 1: (Subsystem) Consider the special case in which the state z can be expressed as $z = (z_1, z_2, z_c)$, with $z_1 \in \mathbb{R}^{n_1}$, $z_2 \in \mathbb{R}^{n_2}$, $z_c \in \mathbb{R}^{n_c}$, $n_1, n_2 > 0$, $n_c \geq 0$, and $n_1 + n_2 + n_c = n$. Following [26], we call z_1, z_2, z_c "state partitions" of the system.

Define $x_1 = (z_1, z_c) \in \mathcal{X}_1 \subseteq \mathbb{R}^{n_1+n_c}$ as the state of subsystem 1, and $x_2 = (z_2, z_c) \in \mathcal{X}_2 \subseteq \mathbb{R}^{n_2+n_c}$ as the state of subsystem 2.

We also express u as $u = (u_1, u_2, u_c)$, with $u_1 \in \mathbb{R}^{m_1}$, $u_2 \in \mathbb{R}^{m_2}$, $u_c \in \mathbb{R}^{m_c}$, $m_1, m_2 > 0$, $m_c \geq 0$, and $m_1 + m_2 + m_c = m$. We call u_1, u_2, and u_c "control partitions" of the system.

Let $w_1 = (u_1, u_c) \in \mathcal{W}_1 \subseteq \mathbb{R}^{m_1+m_c}$ be the control signal of subsystem 1, and $w_2 = (u_2, u_c) \in \mathcal{W}_2 \subseteq \mathbb{R}^{m_2+m_c}$ be the control signal of subsystem 2.

In the case of control-affine systems (1), we can write the dynamics of the two subsystems as

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)^\top w_1, \tag{6a}$$

$$\dot{x}_2 = f_2(x_2) + g_2(x_2)^\top w_2. \tag{6b}$$

The control constraints for each of the subsystems are as follows:

$$c_1(w_1) = \|a_1 \odot w_1\|_\beta - \overline{u} \leq 0, \tag{7a}$$

$$c_2(w_2) = \|a_2 \odot w_2\|_\beta - \overline{u} \leq 0. \tag{7b}$$

where $a_1 = (\alpha_1, \alpha_c)$ and $a_2 = (\alpha_2, \alpha_c)$.

Remark 1: We have $z_1 = x$, $z_2 = y$, $z_c = \theta$, and $u_c = (v, \omega)$ for 3D Dubins Car: $\dot{x} = v\cos(\theta), \dot{y} = v\sin(\theta), \dot{\theta} = \omega$.

In terms of the low-dimensional controls, the constraint in Eq. (2) can be rewritten as

$$c_{\text{joint}}(w_1, w_2) \leq 0, \tag{8}$$

where $c_{\text{joint}}(w_1, w_2) = c(u)$ can be expanded as follows:

$$c_{\text{joint}}(w_1, w_2) = \|[\alpha_1, \alpha_2, \alpha_c] \odot [u_1, u_2, u_c]\|_\beta - \overline{u} \tag{9}$$

$$= \sqrt[\beta]{\sum_{j=1}^{m_1} |\alpha_{1,j} u_{1,j}|^\beta + \sum_{j=1}^{m_2} |\alpha_{2,j} u_{2,j}|^\beta + \sum_{j=1}^{m_c} |\alpha_{c,j} u_{c,j}|^\beta} - \overline{u}$$

The projection operator shows how the full-dimensional state z is related to the low-dimensional state x_i.

Definition 2: (Projection operator) A projection operator $\text{proj} : \mathbb{R}^n \mapsto \mathbb{R}^{n_i+n_c}$ maps a state z in full-dimensional space $\mathbb{R}^n$ to a state x_i in low-dimensional subspace $\mathbb{R}^{n_i+n_c}$:

$$\text{proj}_i(z) := (z_i, z_c) = x_i. \tag{10}$$

The "leaking corner issue" also occurs without the dimensional reduction approaches. In thoses cases, both equations in Eq. (6) become the full-dimensional system in Eq. (1), and our method will still work.

Running example: Consider decomposing the 2D Single Integrator Model $z = (p_x, p_y)$ using the method in [26]:

$$\dot{p}_x = u_x \quad \dot{p}_y = u_y, \tag{11}$$

The state partitions of the system are p_x and p_y. The control input is $u = (u_x, u_y)$ and constrained by $\|u\|_2 \leq \overline{u}$.

There are 2 low-dimensional subsystems, and the two subsystems are

$$\text{Subsystem 1: } x_1 = p_x, \quad \text{Subsystem 2: } x_2 = p_y. \tag{12}$$

Subsystem 1 has a control input given by $w_1 = u_x$, while subsystem 2 has $w_2 = u_y$. The control constraints in low-dimensional systems are as:

$$\begin{aligned} &\text{Subsystem 1: } c_1(w_1) = \|u_x\|_2 \leq \overline{u}, \\ &\text{Subsystem 2: } c_2(w_2) = \|u_y\|_2 \leq \overline{u} \end{aligned} \tag{13}$$

to best preserve the information.

D. Value Function Reconstruction

When applying the computational acceleration method, the low-dimensional value function is computed through the subsystem dynamics as in Definition 1. Consequently, the initial value function representing the target set also needs to be modified for low-dimensional computation. The cost function associated with the low-dimensional computation is given by $\ell_i : \mathbb{R}^{n_i+n_c} \mapsto \mathbb{R}$.

The low-dimensional value function is the solution of the terminal value HJB-PDE

$$D_t\phi_i(x_i,t) + H_{\phi_i}(x_i, w_i, D_{x_i}\phi_i(x_i,t)) = 0 \tag{14}$$

with the boundary condition being given by $\phi_i(x_i,0) = \ell_i(x_i)$. The low-dimensional optimal control w_i^* is obtained as follows:

$$w_{R,i}^*(t) = \arg\min_{w_i \in \mathcal{W}_i} H_{\phi_i}(x_i, w_i, D_{x_i}\phi_i(x_i,t)), \tag{15a}$$

$$w_{A,i}^*(t) = \arg\max_{w_i \in \mathcal{W}_i} H_{\phi_i}(x_i, w_i, D_{x_i}\phi_i(x_i,t)). \tag{15b}$$

Definition 3: (Full-dimensional sub-value function) The full-dimensional sub-value function $V_i(z,t)$ is defined as the value function of the subsystem i with respect to the full-dimensional space $\mathbb{R}^n$.

$$V_i(z,t) = \phi_i(x_i,t), \text{ where } x_i = \text{proj}_i(z) \tag{16}$$

$V_i(z,t)$ could be different depending on the method.[1]

1) : The compuation is termed an **intersection** case when the full-dimensional initial value function relates to the full-dimensional initial sub-value functions as

$$V(z,0) = \max_i(V_i(z,0)). \tag{17}$$

The point-wise maximum operation corresponds to the intersection of sets, following the convention of representing sets using sublevel sets of functions [29].

2) : The compuation is termed **union** case when the full-dimensional initial value function is related to the full-dimensional initial sub-value functions as

$$V(z,0) = \min_i(V_i(z,0)). \tag{18}$$

Definition 4: (Approximated value function) Given the full-dimensional sub-value functions $V_i(z,t)$, the approximated (full-dimensional) value function $\hat{V}(z,t)$ are given, based on the **intersection** or **union** cases, as follows:

$$\textbf{Intersection} : \hat{V}(z,t) = \max_i\{V_i(z,t)\}; \tag{19a}$$

$$\textbf{Union} : \quad \hat{V}(z,t) = \min_i\{V_i(z,t)\}. \tag{19b}$$

The approximated value functions sometimes provide safety and liveness guarantees with better computational efficiency. However, they also sometimes vary from the directly computed true value function. We refer to the latter phenomenon as the "leaking corner issue" [25]–[27].

[1]For Mixed Implicit Explicit (MIE) formulation [23], the full-dimensional sub-value functions are $V_i(z,t) = \gamma_i(\phi_i(x_1,t), x_2)$, where $x_1 = \text{proj}_1(z)$ and $x_2 = \text{proj}_2(z)$. $\gamma_i : \mathbb{R} \times \mathbb{R}^{n_2+n_c} \mapsto \mathbb{R}$ is continuous.

III. LEAKING CORNERS

Definition 5: (Leaking Corners) Suppose we obtain $V(z,t)$ by solving the HJ PDE (3), $\hat{V}(z,t)$ via Eq. (19). The set of "leaking corners" $\mathcal{L}(t)$ is defined as

$$\mathcal{L}(t) = \{z : V(z,t) \neq \hat{V}(z,t)\}. \tag{20}$$

Based on Theorem 1 and 2 in [26], there are 2 cases that will have the "leaking corners" :

(1) The intersection case for the liveness problem, where we obtain $\hat{V}_R$ as follows:

$$\hat{V}_R(z,t) = \max\{V_{R,i}(z,t)\}; \tag{21}$$

(2) The union case for the safety problem:

$$\hat{V}_A(z,t) = \min\{V_{A,i}(z,t)\}. \tag{22}$$

Intuitively, the "leaking corner issue" arises due to the mismatch between the control inputs in different low-dimensional subsystems. The following subsections formalize this intuition.

A. Allowable Control and the "Leaking Corner"

The low-dimensional value functions could evolve independently. However, due to the coupled constraints on the low-dimensional controls, the optimal controls (15) may not always satisfy the original constraints (8). To address this, we introduce the concept of allowable control.

Definition 6: (Allowable Control) We define a pair of low-dimensional control signals $(\tilde{w}_1, \tilde{w}_2)$ that satisfies the coupled constraint in Eq. (8) as *allowable controls*. We also define *allowable control functions* $(\tilde{w}_1(\cdot), \tilde{w}_2(\cdot))$ as control functions that satisfy Eq. (8) for all time.

The corresponding full-dimensional sub-value function corresponding to allowable control functions $(\tilde{w}_1(\cdot), \tilde{w}_2(\cdot))$ is denoted $\tilde{V}_i(z,t)$.

For convenience, assume $\tilde{w}_1$ is the optimal control w_1^* given in Eq.(15), we will denote the other component of the pair of *allowable controls* as $\tilde{w}_2^*$ with $c_{\text{joint}}(w_1^*, \tilde{w}_2^*) \leq 0$. The corresponding value functions are denoted as $V_1(z,t)$ and $\tilde{V}_2^*(z,t)$.

Lemma 1: Suppose $\tilde{w}_1(t) = w_1^*(t) := (u_1(t), u_c(t))$ for all t and $(\tilde{w}_1(\cdot), \tilde{w}_2(\cdot))$ is a pair of allowable control functions. Then, $\tilde{w}_2(t) = \tilde{w}_2^*(t) = (u_2(t), u_c(t))$, where $u_2(t) = \mathbf{0}$ for all t.

Proof: First, note that $\|\cdot\|_\beta$ is convex and the objectives in Eq. (15) are linear in w_i. This means constraint in Eq. (7a) at the optimal control w_1^* of subsystem 1 must be tight:

$$c_1(w_1^*) = \|a_1 \odot w_1^*\|_\beta - \overline{u} = 0 \tag{23}$$

Next, since the joint control constraints in Eq. (8) are given by $0 \geq c_{\text{joint}}(w_1^*, \tilde{w}_2^*)$, we have the following:

$$0 \geq c_{\text{joint}}(w_1^*, \tilde{w}_2^*) \tag{24a}$$

$$= \sqrt[\beta]{\sum_{j=1}^{m_1} |\alpha_{1,j}u_{1,j}|^\beta + \sum_{j=1}^{m_2} |\alpha_{2,j}u_{2,j}|^\beta + \sum_{j=1}^{m_c} |\alpha_{c,j}u_{c,j}|^\beta} - \overline{u} \tag{24b}$$

$$\geq \sqrt[\beta]{\sum_{j=1}^{m_1} |\alpha_{1,j}u_{1,j}|^\beta + \sum_{j=1}^{m_c} |\alpha_{c,j}u_{c,j}|^\beta} - \overline{u} \tag{24c}$$

$$= \|a_1 \odot w_1^*\|_\beta - \overline{u} = 0 \tag{24d}$$

Thus, $c_{\text{joint}}(w_1^*, \tilde{w}_2^*) = 0$; in particular, this means that the expressions in (24b) and (24c) are equal, which implies $\sum_{j=1}^{m_2} |\alpha_{2,j} u_{2,j}|^\beta = 0$. This is equiavlent to $u_2 = \mathbf{0}$.

In addition, this must be true at every time t. Therefore, we have that $\tilde{w}_2^*(t) = (u_2(t), u_c(t))$, where $u_2(t) = \mathbf{0}$ for all t. ∎

Lemma 2: A state z is not in the leaking corner, $z \notin \mathcal{L}(t)$, if and only if there exists a pair of allowable control functions $\tilde{w}_1(\cdot)$ and $\tilde{w}_2(\cdot)$ such that they satisfy the following:

$$\text{Case in Eq. (21): } \max\{\tilde{V}_{R,1}, \tilde{V}_{R,2}\} = \hat{V}_R, \tag{25a}$$

$$\text{Case in Eq. (22): } \min\{\tilde{V}_{A,1}, \tilde{V}_{A,2}\} = \hat{V}_A. \tag{25b}$$

Proof: We will begin with proving the intersection case for liveness problem (Eq. (21)):

Theorem 2 in [28] implies, in terms of value functions, that

$$\max\{\tilde{V}_{R,1}(z,t), \tilde{V}_{R,2}(z,t)\} \geq V_R(z,t). \tag{26}$$

Additionally, from Lemma 2 in [28], we know that

$$V_R(z,t) \geq \hat{V}_R(z,t). \tag{27}$$

Combining Eqs. (26) and (27), we obtain

$$\max\{\tilde{V}_{R,1}, \tilde{V}_{R,2}\} \geq V_R \geq \hat{V}_R. \tag{28}$$

Thus, if $\max\{\tilde{V}_{R,1}(z,t), \tilde{V}_{R,2}(z,t)\} = \hat{V}_R(z,t)$, we have $V_R(z,t) = \hat{V}_R(z,t)$, which is equivalent to $z \notin \mathcal{L}(t)$.

Conversely, if $z \notin \mathcal{L}$(t), then $V_R(z,t) = \hat{V}_R(z,t)$. By Theorem 1 in [28], there exists a pair of allowable control functions ensuring $\max\{\tilde{V}_{R,1}(z,t), \tilde{V}_{R,2}(z,t)\} = V_R(z,t)$.

The proof for the case described in Eq. (22) follows the same reasoning and is omitted for brevity. ∎

IV. Correcting of Leaking Corners

In this section, we present a method for detecting and correcting the "leaking corners" using full-dimensional sub-value functions.

A. Detecting the Leaking Corners

Based on the allowable controls and their corresponding value functions in Definition 6, we can detect the leaking corners $\mathcal{L}(t)$ by value comparison.

Theorem 1: We can find the set of leaking corners $\mathcal{L}(t)$ by comparing the (full-dimensional) sub-value functions.

$$\mathcal{L}(t) = \{z : |V_1(z,t) - V_2(z,t)| < \Delta\}. \tag{29}$$

The value of Δ is

$$\Delta = \begin{cases} |\tilde{V}_1^* - V_1|, & \text{if } V_{R,2} \geq V_{R,1} \text{ or } V_{A,1} \geq V_{A,2}. \quad (30a) \\ |\tilde{V}_2^* - V_2|, & \text{if } V_{R,1} \geq V_{R,2} \text{ or } V_{A,2} \geq V_{A,1}. \quad (30b) \end{cases}$$

Proof: Without loss of generality, we will consider the liveness problem, in the case where $V_{R,1}(z,t) \geq V_{R,2}(z,t)$. The case in which $V_{R,2}(z,t) \geq V_{R,1}(z,t)$ can be proven by repeating the exact same argument. The cases involvign the safety problem can be proven by starting with $\min\{V_{A,1}(z,t), V_{A,2}(z,t)\} = \hat{V}_A(z,t)$ and $\tilde{V}_{A,1}(z,t) \leq V_{A,1}(z,t)$, and then following the exact same arguments.

In the liveness problem, as stated in (21), we have $\max\{V_{R,1}(z,t), V_{R,2}(z,t)\} = \hat{V}_R(z,t)$. In the case that $V_{R,1}(z,t) \geq V_{R,2}(z,t)$, we have $V_{R,1}(z,t) = \hat{V}_R(z,t)$.

Note that $V_{R,1}(z,t) \leq \tilde{V}_{R,1}(z,t)$ as it is solved by minimizing the Hamiltonian (4a). From Eq. (25a) in Lemma 2, there exists a pair of allowable controls to let $\max\{\tilde{V}_{R,1}(z,t), \tilde{V}_{R,2}(z,t)\} = \hat{V}_R(z,t)$ if $z \notin \mathcal{L}(t)$. Thus, $\tilde{V}_{R,1}(z,t) = V_{R,1}(z,t) = \hat{V}_R(z,t)$.

When $V_{R,1} \geq V_{R,2}$,

$$z \notin \mathcal{L}(t) \Leftrightarrow \tilde{V}_{R,1} = V_{R,1} = \hat{V}_R \wedge \max\{\tilde{V}_{R,1}, \tilde{V}_{R,2}\} = \hat{V}_R \tag{31}$$

$$(\tilde{w}_1 = w_1^*, \text{ and } \tilde{w}_2^* \text{ can be found with Lemma 1})$$

$$\Leftrightarrow \tilde{V}_{R,2}^* = \tilde{V}_{R,2} \leq \tilde{V}_{R,1} = V_{R,1} = \hat{V}_R \tag{32}$$

Note that $V_{R,2} \leq \tilde{V}_{R,2}$ as it is solved by minimizing the Hamiltonian (4a), we define

$$\Delta = |\tilde{V}_{R,2}^* - V_{R,2}| = \tilde{V}_{R,2}^* - V_{R,2}, \tag{33}$$

and let

$$|V_{R,1} - V_{R,2}| = V_{R,1} - V_{R,2} < \Delta \tag{34}$$

$$V_{R,1} - V_{R,2} < \tilde{V}_{R,2}^* - V_{R,2} \tag{35}$$

$$V_{R,1} < \tilde{V}_{R,2}^* \tag{36}$$

which contradicts Eq. (32), thus we prove that when $V_{R,1}(z,t) - V_{R,2}(z,t) < \Delta$, state $z \in \mathcal{L}(t)$. ∎

B. Local Updating Procedure

Definition 7: (Island) The "leaking corner" set $\mathcal{L}(t)$ consists of a finite union of k connected sets, referred to as islands, denoted by $\mathcal{I}_\kappa(t)$ for $\kappa \in \{1, 2, 3, ..., k\}$. Each $\mathcal{I}_\kappa(t)$ is a connected component of $\mathcal{L}(t)$.

Assume that for all $\kappa \in \{1, 2, 3, ..., k\}$, there exists at least one state $z^* \in \mathcal{I}_\kappa(t)$ such that

$$V_1(z^*, t) = V_2(z^*, t) \tag{37}$$

for both cases suffering from the "leaking corner issue," as described in Eq. (21) and Eq. (22).

Since $\hat{V}(z^*, t)$ is obtained through Eq. (19). If Eq. (25) and Eq. (37) hold, the following condition must hold:

$$\tilde{V}_1(z^*, t) = V_1(z^*, t), \text{ and } \tilde{V}_2(z^*, t) = V_2(z^*, t). \tag{38}$$

However, to satisfy the above equation, optimal control in both subsystems must be applied, which violates the control constraint given by Eq. (8). The coupled control constraint attains its worst-case violation:

$$c(w_1^*, w_2^*) = \max_{w_1 \in \mathcal{W}_1, w_2 \in \mathcal{W}_2} c(w_1, w_2) > 0. \tag{39}$$

Such a state z^* is a "leaking corner" as indicated in Lemma 2.

Let $V_d(z,t)$ represent the absolute difference between the low-dimensional value functions:

$$V_d(z,t) = |V_1(z,t) - V_2(z,t)| \geq 0. \tag{40}$$

$V_i(z,t)$ as Eq. (16) is from the viscosity solution of a continuous initial value function [30], then $V_i(z,t)$ and $V_d(z,t)$ are also continuous.

For neighboring states where $V_d(z,t) \geq 0$, there exist control options (w_1, w_2) that are different from the optimal control pair (w_1^*, w_2^*) if try to satisfy the condition given by Eq. (25). The constraint violation is reduced, though it may not be eliminated. Mathematically, this is expressed as:

Algorithm 1: Local updating procedure

Data: $\hat{V}(\cdot,\cdot), \hat{\mathcal{L}}(\cdot), Z, t_{\text{list}} = [t, t+\delta ..., 0]$
Result: $\check{V}(\cdot,\cdot)$
$s \leftarrow 0$; ▷ Backward Computation
$\check{V}(\cdot,0) \leftarrow \hat{V}(\cdot,0)$;
Frontier $\leftarrow$ nextFrontier $\leftarrow$ visited $\leftarrow \{\}$;
while $s > t$ **do**
  for $z \in Z$ **do**
    if $z \in \hat{\mathcal{L}}(s)$ **then**
      updateValue$(z, s, \delta,$ Frontier$)$
    else
      $\check{V}(z, s-\delta) \leftarrow \hat{V}(z, s-\delta)$;
    end
  end
  visited $\leftarrow \hat{\mathcal{L}}(s)$
  Frontier $\leftarrow$ Frontier $\setminus$ visited
  while *Frontier* $\neq \emptyset$ **do**
    for z *in Frontier* **do**
      updateValue$(z, s, \delta,$ nextFrontier$)$
    end
    visited $\leftarrow$ visited $\cup$ Frontier
    Frontier $\leftarrow$ nextFrontier $\setminus$ visited
    nextFrontier $\leftarrow \{\}$
  end
  $s \leftarrow s - \delta$;
end
def *updateValue*$(z, s, \delta,$ *Frontier*$)$**:**
  $\check{V}(z, s-\delta) \leftarrow$ HJ Update$(\check{V}(z,s))$ ▷ Equation 3
  if $\check{V}(z, s-\delta) \neq \hat{V}(z, s-\delta)$ **then**
    Frontier $\leftarrow$ Frontier $\cup$ neighbor(z)
  end

$$0 \leq c(w_1, w_2) < c(w_1^*, w_2^*). \tag{41}$$

As we move slightly away from the worst-case mismatch points, we eventually reach states where allowable control pairs $(\tilde{w}_1, \tilde{w}_2)$ exist such that the constraint is satisfied:

$$c(\tilde{w}_1, \tilde{w}_2) \leq 0. \tag{42}$$

These states are not "leaking corners" according to Lemma 2.

Since each connected island $\mathcal{I}_\kappa$ contains at least one worst-case scenario state z_κ, for each $\mathcal{I}_\kappa(t)$, there exists a state $z \in \mathcal{L}(t)$ from Theorem 1. The transition from extreme constraint violation to a region free from the "leaking corner issue" occurs continuously within each island.

To correct these "leaking corners", we apply Algorithm 1, which locally updates the affected regions. We first detect an approximated "leaking corners" $\hat{\mathcal{L}}(\cdot) \subseteq \mathcal{L}(\cdot)$ through Theorem 1. Using the approximated value function $\hat{V}(\cdot,\cdot)$ from Definition 4, we locally refine the value of the "leaking corners" through Algorithm 1 and denote the updated result as $\check{V}(\cdot,\cdot)$. Fig. 1 provides a visual representation of this update procedure, illustrating how the detected leaking corners are iteratively corrected.

In Algorithm 1, neighbor(z) returns a set of states that are adjacent to the state z. Every island of the "leaking corners" will be included, and every "leaking corner" is covered using our formula.

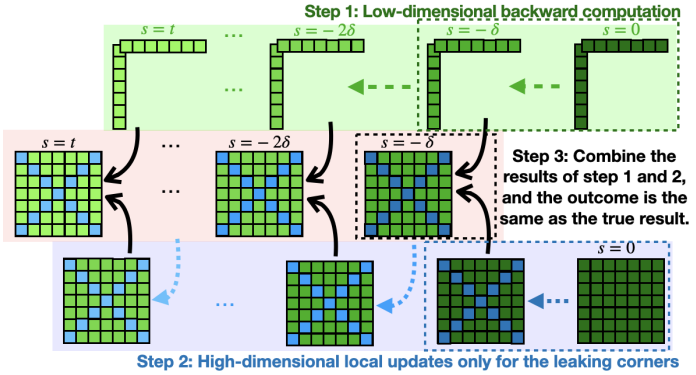

Fig. 1: The results from the low-dimensional computation are in the first row. The third row shows the local updated results in the full-dimensional space. The combined results, which equal the true results, are in the second row.

V. Numerical Examples

In this section, the 2D and 6D examples demonstrate that:
(1) Theorem 1 accurately locates the "leaking corners."
(2) The local updating process in Algorithm 1 produces results equivalent to the ground truth while maintaining computational efficiency.

The experiment was conducted on a system with 96.0 GB of memory, an AMD Ryzen 9 5950X 16-core processor (32 threads), and Ubuntu 22.04.3 LTS as the operating system. We use the highly parallelized Optimized_dp [22] for computation, while our correction step, implemented in Python, is not parallelized. We choose the self-contained subsystem decomposition (SCSD) as the dimensionality reduction method for our experiments.

We consider the two values the same if their difference is within a threshold of 1×10^{-3}.

A. Running Example: 2D Single Integrator

Consider the 2D single integrator example, which is used as a demonstration throughout the paper. Let the control constraint value be $\overline{u} = 1$m/s.

The initial value function for subsystem 1 is $\phi_1(x_1, 0) = \ell_1(x_1) = |x| - 1$, and for subsystem 2, it is $\phi_2(x_2, 0) = \ell_2(x_2) = |y| - 1$. The initial value function for the full-dimensional system is given by $V(z, 0) = \max\{\ell_1(x_1), \ell_2(x_2)\}$.

The grid consists of 101×101 points with each dimension ranging from -4 to 4.

1) Single time step case: In this case, we set $t = -0.02$s and $\delta = 0.02$s. The value function updated with our method is denoted as $\check{V}(z, t)$. For comparison, the ground truth value function $V(z, t)$ is obtained through direct computation from the full-dimensional system.

First, we identify the "leaking corners" using Theorem 1. The ground truth number of "leaking corners" is 200, and the detection using Theorem 1 with $\Delta = 0.02$ identified 201 "leaking corners". After locating the "leaking corners",

we perform the local updating process as outlined in Algorithm 1.

TABLE I: 2D Accuracy Comparison for One Step

Metric	Before	After
Number of grid points with different values from the ground truth	200	0
Average absolute difference from ground truth	1.2×10^{-4}	9.51×10^{-18}
Maximum absolute difference from ground truth	2×10^{-2}	2.22×10^{-16}

TABLE II: 2D Time Comparison for One Step

Process	Time (seconds)
Direct computation	3.3×10^{-2}
SCSD computation + HJ local update computation	$7 \times 10^{-4} + 1.3 \times 10^{-3} = 2.0 \times 10^{-3}$

Table I presents the accuracy comparison, while Table II shows the computational time comparison. The results demonstrate that our method accurately locates and corrects the "leaking corners." Additionally, compared to direct computation, our method offers better time efficiency.

2) 10 time steps case: In this case, $t = -0.2$s and $\delta = 0.02$s. The computation needs to run backward for 10 time steps to complete.

For the final time step of computation, the ground truth number of the "leaking corners" is 1344. Using Theorem 1 with $\Delta = 0.2$, we identified 1001 "leaking corners". After detecting these "leaking corners", we apply the local updating process described in Algorithm 1.

TABLE III: 2D Accuracy Comparison for 10 Steps

Metric	Before	After
Number of states with different values from the ground truth	1344	0
Average absolute difference from ground truth	2.5×10^{-3}	1×10^{-9}
Maximum absolute difference from ground truth	7.39×10^{-2}	2.44×10^{-8}

TABLE IV: 2D Time Comparison for 10 Steps

Process	Time (seconds)
Direct computation	3.36×10^{-1}
SCSD computation + HJ local update computation	$4.72 \times 10^{-3} + 1.61 \times 10^{-1} = 1.65 \times 10^{-1}$

Table III presents the accuracy comparison, while Table IV provides the computational time comparison. The results show that our method successfully identifies every island of "leaking corners." Moreover, our approach improves time efficiency compared to direct computation, even without implementing parallel computing for making the corrections.

The value comparison for the final time step is shown in Fig. 2

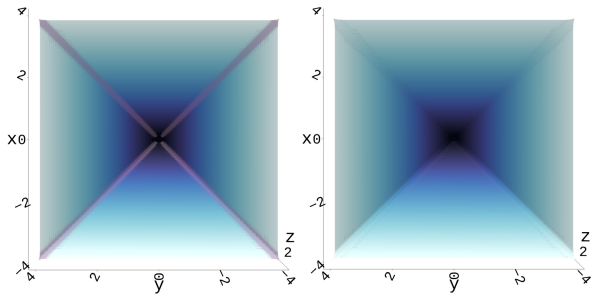

Fig. 2: The 2 figures illustrate the value functions. The left figure displays the approximated value function, with the leaking corners" $\mathcal{L}(t)$—where the values deviate from the ground truth—highlighted in gray. The right figure shows the value function after applying our correction method, where no "leaking corners" remain, demonstrating the effectiveness of our method in aligning with the ground truth.

B. Running Example: 6D Planar Quadrotor

Consider the 6D Planar Quadrotor:

$$\dot{z} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ -u_T \sin\theta \\ u_T \cos\theta - g \\ \omega \\ u_\tau \end{bmatrix} \tag{43}$$

The control signal is $u = (u_T, u_\tau)$, with the control constraint $c^1(u_T) = |u_T| \leq \overline{u}_T = 1$ m/s, and $c^2(u_\tau) = |u_\tau| \leq \overline{u}_\tau = 1$ rad/s^2. The state partitions of the system are $x, y, v_x, v_y, \theta, \omega$.

Using the SCSD method, the dynamics could be decomposed into 2 subsystems. Subsystem 1 has the dynamics:

$$\dot{x}_1 = (\dot{x}, \dot{v}_x, \dot{\theta}, \dot{\omega}), \tag{44}$$

and subsystem 2 has the dynamics:

$$\dot{x}_2 = (\dot{y}, \dot{v}_y, \dot{\theta}, \dot{\omega}). \tag{45}$$

The initial value function for subsystem 1 is $\phi_1(x_1, 0) = \ell_1(x_1) = x$, and for subsystem 2, it is $\phi_2(x_2, 0) = \ell_2(x_2) = y$. The initial value function for the full-dimensional system is given by $V(z, 0) = \min\{\ell_1(x_1), \ell_2(x_2)\}$.

The grid consists of 21^6 points. The x and y dimensions range from -1.0 to 4.0, while the remaining dimensions range from -2.0 to 2.0.

TABLE V: 6D Accuracy Comparison for One Step

Metric	Before	After
Number of grid points with different values from the ground truth	3.89×10^6	0
Average absolute difference from ground truth	6.97×10^{-4}	0.0
Maximum absolute difference from ground truth	4×10^{-2}	0.0

Table V presents the accuracy comparison for 1 time step. The computational time results are summarized in Table VI. This case was previously intractable due to the "curse of dimensionality."

TABLE VI: Computation Time and Delta Value for ts

t (s)	Δ	**Decomposition Time + Local Updating Time (seconds)**
-0.02	0.04	2.447 + 47.1078 = 49.5548
-0.06	0.1212	6.769 + 157.3528 = 164.1218
-0.1	0.204	11.8038 + 250.7859 = 262.5897

The value function slices for different Δ values at corresponding time steps are shown in Fig. 3.

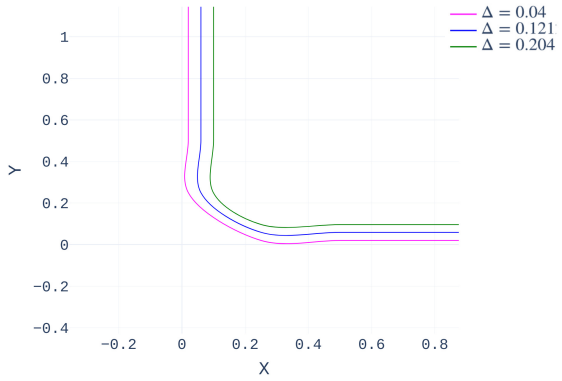

Fig. 3: The figure demonstrates the local updating results for the backward computation of 0.02 seconds with $\Delta = 0.04$, 0.06 seconds with $\Delta = 0.1212$ and 0.1 seconds with $\Delta = 0.204$. The dimensions shown are x and y. We take a slice of $v_x = -1$, $v_y = -1$, $\theta = 0$, and $\omega = 0.4$.

VI. Conclusion and Future Work

In this paper, we propose a threshold-based method to detect the leaking corners when low-dimensional control inputs are subject to certain control constraints. We also introduce a local updating method that ensures accuracy while maintaining computational efficiency. The proposed method is validated using a 2D Single Integrator system and a 6D Planar Quadrotor system with the SCSD method.

Future work includes: (1) Testing the method with other computational acceleration techniques; (2) Parallelizing the local updating process for faster computation; (3) Exploring machine learning or other techniques for new value updating methods;

References

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

[2] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. Del Prete, "Optimization-based control for dynamic legged robots," *IEEE Transactions on Robotics*, vol. 40, pp. 43–63, 2023.

[3] M. Natarajan, E. Seraj, B. Altundas, R. Paleja, S. Ye, L. Chen, R. Jensen, K. C. Chang, and M. Gombolay, "Human-robot teaming: grand challenges," *Current Robotics Reports*, vol. 4, no. 3, pp. 81–100, 2023.

[4] D. Hanover, A. Loquercio, L. Bauersfeld, A. Romero, R. Penicka, Y. Song, G. Cioffi, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing: A survey," *IEEE Transactions on Robotics*, 2024.

[5] S. Kong, S. Gao, W. Chen, and E. Clarke, "dreach: δ-reachability analysis for hybrid systems," in *Tools and Algorithms for the Construction and Analysis of Systems*, 2015.

[6] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "Reachnn: Reachability analysis of neural-network controlled systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[7] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Trans. on automatic control*, vol. 50, no. 7, 2005.

[8] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "Spaceex: Scalable verification of hybrid systems," in *Comput. Aided Verification*, 2011.

[9] M. Althoff, O. Stursberg, and M. Buss, "Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes," *Nonlinear analysis: hybrid systems*, 2010.

[10] J. Darbon and S. Osher, "Algorithms for overcoming the curse of dimensionality for certain hamilton–jacobi equations arising in control theory and elsewhere," *Research in the Mathematical Sciences*, vol. 3, no. 1, p. 19, 2016.

[11] D. Lee and C. J. Tomlin, "Efficient computation of state-constrained reachability problems using hopf–lax formulae," *IEEE Transactions on Automatic Control*, vol. 68, no. 11, pp. 6481–6495, 2023.

[12] W. Sharpless, Y. T. Chow, and S. Herbert, "Conservative linear envelopes for nonlinear, high-dimensional, hamilton-jacobi reachability," *arXiv preprint arXiv:2403.14184*, 2024.

[13] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis: internal approximation," *Syst. & Control Letters*, vol. 41, no. 3, 2000.

[14] M. Li, P. N. Mosaad, M. Fränzle, Z. She, and B. Xue, "Safe over- and under-approximation of reachable sets for autonomous dynamical systems," in *Formal Modeling and Analysis of Timed Systems*, 2018.

[15] V. Liu, C. Manzie, and P. M. Dower, "Efficient value function upper bounds for a class of constrained linear time-varying optimal control problems," in *2024 American Control Conference (ACC)*, pp. 4084–4089, IEEE, 2024.

[16] J. F. Fisac, N. F. Lugovoy, V. Rubies-Royo, S. Ghosh, and C. J. Tomlin, "Bridging hamilton-jacobi safety analysis and reinforcement learning," in *Int. Conf. on Robot. and Automat.*, IEEE, 2019.

[17] S. Bansal and C. J. Tomlin, "Deepreach: A deep learning approach to high-dimensional reachability," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1817–1824, IEEE, 2021.

[18] A. Lin and S. Bansal, "Verification of neural reachable tubes via scenario optimization and conformal prediction," in *6th Annual Learning for Dynamics & Control Conference*, pp. 719–731, PMLR, 2024.

[19] L. Ruthotto, S. J. Osher, W. Li, L. Nurbekyan, and S. W. Fung, "A machine learning framework for solving high-dimensional mean field game and mean field control problems," *Proceedings of the National Academy of Sciences*, vol. 117, no. 17, pp. 9183–9193, 2020.

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

[21] Y. T. Chow, J. Darbon, S. Osher, and W. Yin, "Algorithm for overcoming the curse of dimensionality for state-dependent hamilton-jacobi equations," *Journal of Computational Physics*, vol. 387, pp. 376–409, 2019.

[22] M. Bui, G. Giovanis, M. Chen, and A. Shriraman, "Optimizeddp: An efficient, user-friendly library for optimal control and dynamic programming," *arXiv preprint arXiv:2204.05520*, 2022.

[23] I. M. Mitchell, "Scalable calculation of reach sets and tubes for nonlinear systems with terminal integrators: a mixed implicit explicit formulation," in *Proceedings of the 14th international conference on Hybrid systems: computation and control*, pp. 103–112, 2011.

[24] P. Holmes, S. Kousik, B. Zhang, D. Raz, C. Barbalata, M. Johnson-Roberson, and R. Vasudevan, "Reachable sets for safe, real-time manipulator trajectory design," 2020.

[25] M. Chen, S. Herbert, and C. J. Tomlin, "Exact and efficient hamilton-jacobi guaranteed safety analysis via system decomposition," in *Int. Conf. on Robot. and Automat.*, IEEE, 2017.

[26] 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," *Trans. on Automatic Control*, vol. 63, no. 11, 2018.

[27] D. Lee, M. Chen, and C. J. Tomlin, "Removing leaking corners to reduce dimensionality in hamilton-jacobi reachability," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9320–9326, IEEE, 2019.

[28] C. He, Z. Gong, M. Chen, and S. Herbert, "Efficient and guaranteed hamilton-jacobi reachability via self-contained subsystem decomposition and admissible control sets," *IEEE Control Systems Letters*, 2023.

[29] I. M. Mitchell *et al.*, "A toolbox of level set methods," *UBC Department of Computer Science Technical Report TR-2007-11*, 2007.

[30] L. C. Evans, *Partial differential equations*, vol. 19. American Mathematical Society, 2022.