2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Exact and Efficient Hamilton-Jacobi Guaranteed Safety Analysis via System Decomposition

Mo Chen, Sylvia Herbert, Claire J. Tomlin

***Abstract*— Hamilton-Jacobi (HJ) reachability is a method that provides rigorous analyses of the safety properties of dynamical systems. These guarantees can be provided by the computation of a backward reachable set (BRS), which represents the set of states from which the system may be driven into violating safety properties despite the system's best effort to remain safe. Unfortunately, the complexity of the BRS computation scales exponentially with the number of state dimensions. Although numerous approximation techniques are able to tractably provide conservative estimates of the BRS, they often require restrictive assumptions about system dynamics without providing an exact solution. In this paper we propose a general method for decomposing dynamical systems. Even when the resulting subsystems are coupled, relatively high-dimensional BRSs that were previously intractable or expensive to compute can now be quickly and exactly computed in lower-dimensional subspaces. As a result, the curse of dimensionality is alleviated to a large degree without sacrificing optimality. We demonstrate our theoretical results through a 3D Dubins Car model and a 6D Acrobatic Quadrotor model.**

I. Introduction

As the presence of safety-critical systems in everyday life has grown, so has the importance for the verification of these systems. Given the number and density of autonomous systems expected in civilian space, higher-fidelity models are needed to more accurately characterize these systems so that safety can be guaranteed. Thus, tractable verification tools that are not overly conservative are urgently needed.

Optimal control and differential game theory are powerful tools for the verification of nonlinear systems due to their flexibility with respect to system dynamics, treatment of unknown disturbances, and guaranteed optimality [1]–[4]. Reachability analysis is core to these methods; here, the goal is to compute the backward reachable set (BRS), defined as the set of states from which the system can be driven into some unsafe set despite using the optimal control to avoid the unsafe set. Hamilton-Jacobi (HJ) reachability has been successfully used to guarantee safety for low-dimensional systems [2], [5]–[7]. HJ reachability theory is also very convenient to use due to the many numerical tools available to obtain optimal solutions [8]–[10].

Despite these advantages, HJ reachability-based methods involve solving a partial differential equation (PDE) or variational inequality on a grid representing a numerical discretization of the state space. As a result, the computation complexity scales exponentially with the system dimension. Application of current formulations of HJ reachability is limited to systems with five dimensions or fewer.

For the analysis of high-dimensional systems, a number of approximation techniques exist. These techniques usually place strong assumptions on system dynamics, such as requiring a polynomial form [11], [12], a linear form [13], [14], or a Hamiltonian that is only dependent on the control variable [15]. Other methods that are less restrictive in terms of system dynamics include [16], which works with projections, and [17], which involves treating system states as disturbances. In all of the methods mentioned so far, varying degrees of approximation or conservatism is introduced. Under some special scenarios such as those outlined in [18] or [19], a small dimensionality reduction may be possible when obtaining exact optimal solutions.

The previous methods either trade off between optimality and computation complexity or provide only a small dimensionality reduction. In contrast, this paper presents the *self-contained subsystem (SCS)* formulation for computing *exact, optimal solutions* of systems with dynamics while drastically reducing dimensionality. Motivated by the need to provide safety guarantees, we compute BRSs in lower-dimensional subspaces of the full system state space, and then combine these low-dimensional BRSs to exactly construct the full-dimensional BRS. The full-dimensional BRS can be exactly constructed through back projections of the lower-dimensional BRSs *even with coupling between the different subsystems*. Furthermore, the theory we present in this paper is compatible with any other method such as [17] and [18]. When different methods are combined together, even more substantial dimensionality reduction can be achieved.

This paper will be presented as follows:

- First, Sections II and III introduce the relevant HJ reachability theory, and all the required definitions.
- Next, Section IV presents the SCS formulation, our main theoretical result. We describe how BRSs in lower-dimensional subspaces can be combined to construct the full-dimensional BRS exactly.
- Finally, Section V presents two numerical examples: a 3D Dubins Car example to validate our theory and a 6D Acrobatic Quadrotor example that was previously intractable using standard methods.

II. Background

There are several HJ formulations that can compute BRSs exactly when the system dimensionality is low. Although these methods have been successfully used for lower-dimensional systems, they become intractable when the system dimension is greater than approximately five. In this

This work has been supported in part by NSF under CPS:ActionWebs (CNS-931843), by ONR under the HUNT (N0014-08-0696) and SMARTS (N00014-09-1-1051) MURIs and by grant N00014-12-1-0609, by AFOSR under the CHASE MURI (FA9550-10-1-0567). The research of M. Chen has received funding from the "NSERC PGS-D" Program.

All authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley. {mochen72, sylvia.herbert, tomlin}@berkeley.edu

section, we give a brief overview to provide a starting point on which we build the new proposed theory.

A. Full System Dynamics

Definition 1: **Full system**. Let z be the state of the system under consideration. We call this system the "full system," or just "system" for short. The evolution of the state of the full system satisfies the ordinary differential equation (ODE)

$$\frac{dz}{ds} = \dot{z} = f(z,u), s \in [t,0], t \leq 0 \\ z \in \mathcal{Z}, u \in \mathcal{U} \tag{1}$$

For clarity, we assume that the state space $\mathcal{Z}$ is $\mathbb{R}^n$, but our theory also applies to systems with periodic state dimensions such as angles. The control is denoted by u, with the control function $u(\cdot)$ being drawn from the set of measurable functions.

The system dynamics $f : \mathcal{Z} \times \mathcal{U} \to \mathcal{Z}$ is assumed to be uniformly continuous, bounded, and Lipschitz continuous in z for fixed u. With this assumption, given $u(\cdot) \in \mathbb{U}$, there exists a unique trajectory solving (1) [20], [21].

We will denote solutions, or trajectories of (1) starting from some state z at time t under control $u(\cdot)$ as $\zeta(s; z, t, u(\cdot))$. The system trajectory satisfies an initial condition and the ODE (1) almost everywhere:

$$\frac{d}{ds}\zeta(s; z, t, u(\cdot)) = f(\zeta(s; z, t, u(\cdot)), u(s)) \\ \zeta(t; z, t, u(\cdot)) = z \tag{2}$$

B. Backward Reachable Set

In this paper, we consider a common definition of the BRS relevant for guaranteeing safety. Intuitively, the BRS represents the set of states z from which the system can be driven into an unsafe set $\mathcal{L}$ at a particular time. For our definition of BRS, we stipulate that the system be driven to $\mathcal{L}$ for all control functions $u(\cdot)$. In this case, the unsafe set can often be interpreted as a set of states to be avoided (such as an obstacle), and the BRS represents the set of states that leads to the system entering the unsafe set despite all possible control functions. We now formally define the BRS.

Definition 2: **Backward reachable set**. We denote the BRS $\mathcal{V}(t)$, and define it as follows:

$$\mathcal{V}(t) = \{z \in \mathcal{Z} : \forall u(\cdot) \in \mathbb{U}, \zeta(0; z, t, u(\cdot)) \in \mathcal{L}\} \tag{3}$$

C. The Full Formulation for Computing the BRS

There are various HJ formulations such as [1], [2], [4], and [22] that cast the reachability problem as an optimal control problem and directly compute the BRS in the full state space of the system. Although these methods are not scalable beyond relatively low-dimensional systems, they form the foundation on which we will build our theory. We now briefly summarize the necessary details related to the HJ PDEs.

Let the unsafe set $\mathcal{L} \subseteq \mathcal{Z}$ be represented by the implicit surface function $l(z)$ such that $\mathcal{L} = \{z \in \mathcal{Z} : l(z) \leq 0\}$. Examples of implicit surface functions are shown as colored surfaces in Fig. 1, with the boundary of the corresponding sets they represent shown in black.

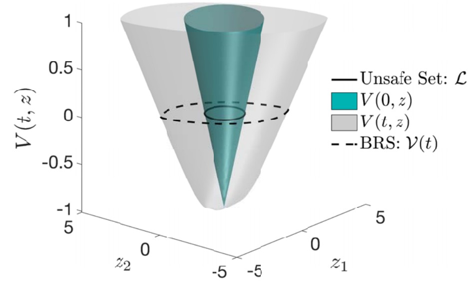


Fig. 1: A 2D example illustrating HJ reachability. The boundary of the unsafe set $\mathcal{L}$ is shown as the solid black line. The blue surface represents the implicit surface function $l(z)$ of $\mathcal{L}$, which by (6) is equivalent to $V(0,z)$. The light gray surface shows $V(t,z)$. The corresponding BRS $\mathcal{V}(t)$ is the zero sub-level set of this function; the boundary of $\mathcal{V}(t)$ is seen here as the dashed black line.

Consider the optimal control problem given by

$$V(t,z) = \max_{u(\cdot) \in \mathbb{U}} l(\zeta(0; z, t, u(\cdot))) \\ \text{subject to (2)} \tag{4}$$

with the optimal control being given by

$$u^*(\cdot) = \arg\max_{u(\cdot) \in \mathbb{U}} l(\zeta(0; z, t, u(\cdot))) \tag{5}$$

It is well-known that the value function $V(t,z)$ is the implicit surface function representing $\mathcal{V}(t)$: $\mathcal{V}(t) = \{z \in \mathcal{Z} : V(t,z) \leq 0\}$. The value function $V(t,z)$ is the viscosity solution [23], [24] of the HJ PDE

$$D_s V(s,z) + H(z, \nabla V(s,z)) = 0, \quad s \in [t,0] \\ V(0,z) = l(z) \tag{6}$$

The Hamiltonian in (6) is given by

$$H(z,p) = \max_{u \in \mathcal{U}} p \cdot f(z,u) \tag{7}$$

Fig. 1 shows an illustration of HJ reachability. $l(z)$, the implicit surface function representing $\mathcal{L}$, and the value function $V(t,z)$, the implicit surface function representing the BRS $\mathcal{V}(t)$, are shown as the blue and light gray surfaces respectively. The unsafe set $\mathcal{L}$ and the BRS $\mathcal{V}(t)$ are the zero sub-level sets of these two surface functions; the boundaries of $\mathcal{L}$ and $\mathcal{V}(t)$ are shown in black. Once the value function V is computed, the optimal control (5) can be obtained by

$$u^*(s) = \arg\max_{u \in \mathcal{U}} \nabla V(s,z) \cdot f(z,u) \tag{8}$$

We state the following algorithm for clarity and convenience:

Algorithm 1: **Full formulation**. Given an unsafe set $\mathcal{L}$ and dynamics (1), the full formulation for computing the BRS is given by the following algorithm:

1) Define the implicit surface function $l(z)$.
2) Solve (6) with Hamiltonian (7) to obtain $V(t,z)$, the implicit surface function representing $\mathcal{V}(t)$.

III. Problem Formulation

In this paper, we seek to obtain the BRS in Definition 2 via computations in a lower-dimensional subspace under the assumption that the system (1) can be decomposed into SCSs. Such a decomposition can be commonly found, since many systems involve components that are loosely coupled. In particular, in the dynamics of many vehicles, the evolution of the position variables is often weakly coupled though other variables such as heading.

A. Definitions

*1) **Subsystem Dynamics**:* Let the system $z \in \mathcal{Z} = \mathbb{R}^n$ be partitioned as follows:

$$\begin{aligned} & z = (y_1, y_2, y_3), \quad y_1 \in \mathbb{R}^{n_1}, y_2 \in \mathbb{R}^{n_2}, y_3 \in \mathbb{R}^{n_3} \\ & n_1, n_2 > 0, n_3 \geq 0 \end{aligned} \tag{9}$$

Note that n_3 could be zero, and $n_1+n_2+n_3 = n$. We call the variables y_i the "state partitions", or just "partitions".

Define the SCS states $x_1 \in \mathcal{X}_1 = \mathbb{R}^{n_1+n_3}, x_2 \in \mathcal{X}_2 = \mathbb{R}^{n_2+n_3}$ as follows:

$$x_1 = (y_1, y_3) \qquad x_2 = (y_2, y_3) \tag{10}$$

It is important to note that x_1 and x_2 in general have overlapping states in the partition y_3. Note that our theory is applicable to any finite number of subsystems defined in the analogous way; however, for clarity and without loss of generality, we will assume that there are two subsystems.

For convenience, we have assumed that $\mathcal{X}_1 = \mathbb{R}^{n_1+n_3}, \mathcal{X}_2 = \mathbb{R}^{n_2+n_3}$, but as previously mentioned, our theory also applies to systems with periodic state dimensions.

Definition 3: **Self-contained subsystem**. We call each of the systems with states x_i evolving according to (11) a "self-contained subsystem" (SCS), or just "subsystem" for short.

$$\begin{aligned} & \frac{dx_1}{ds} = \dot{x}_1 = g_1(x_1, u) = g_1(y_1, y_3, u), \quad s \in [t, 0] \\ & \frac{dx_2}{ds} = \dot{x}_2 = g_2(x_2, u) = g_1(y_2, y_3, u) \end{aligned} \tag{11}$$

Intuitively (11) means that the evolution of states in each subsystem depend only on the states in that subsystem: for example, the evolution of x_1 depends only on the states in x_1. However, the two subsystems are coupled through the state partition y_3 and control u. For example, consider the dynamics of a Dubins Car:

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \tag{12}$$

with state $z = (p_x, p_y, \theta)$, control $u = \omega$, and state partitions $y_1 = p_x, y_2 = p_y, y_3 = \theta$. The subsystems x_i have dynamics

$$\begin{aligned} & \dot{x_1} = \begin{bmatrix} \dot{y_1} \\ \dot{y_3} \end{bmatrix} = \begin{bmatrix} \dot{p_x} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ \omega \end{bmatrix} \\ & \dot{x_2} = \begin{bmatrix} \dot{y_2} \\ \dot{y_3} \end{bmatrix} = \begin{bmatrix} \dot{p_y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\sin\theta \\ \omega \end{bmatrix}, \qquad u = \omega \end{aligned} \tag{13}$$

where the overlapping state is $\theta = y_3$. For another example of a system decomposed into two self-contained subsystems, see Section V-B.

Although there may be common states in x_1 and x_2, the evolution of each subsystem does not depend on the other explicitly. Hence, each subsystem is self-contained.

*2) **Projection Operators**:* Define the projection of a state z onto a subsystem state space $\mathcal{X}_i$ as

$$\text{proj}_{\mathcal{X}_i}(z) = x_i, i = 1, 2 \tag{14}$$

For convenience, we will define the projection operator applied on sets $\mathcal{S} \subseteq \mathcal{Z}$:

$$\text{proj}_{\mathcal{X}_i}(\mathcal{S}) = \{x_i \in \mathcal{X}_i : \exists z \in \mathcal{S}, \text{proj}_{\mathcal{X}_i}(z) = x_i\} \tag{15}$$

We aim to relate the BRSs of the subsystems to the BRS of the full system, so we define the back projection operator:

$$\text{proj}^{-1}(x_i) = \{z \in \mathcal{Z} : \text{proj}_{\mathcal{X}_i}(z) = x_i\} \tag{16}$$

We will also apply the back projection operator on sets. In this case, we abuse notation and define the back projection operator on some set $\mathcal{S}_i \subseteq \mathcal{X}_i$ as

$$\text{proj}^{-1}(\mathcal{S}_i) = \{z \in \mathcal{Z} : \exists x_i \in \mathcal{S}_i, \text{proj}_{\mathcal{X}_i}(z) = x_i\} \tag{17}$$

Fig. 2 illustrates the definitions involving projections.

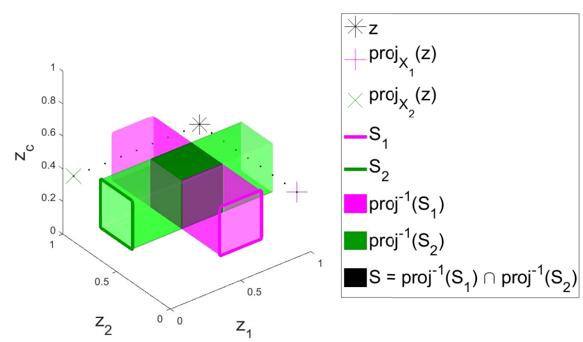


Fig. 2: This figure shows the back-projection of sets in the z_1-z_c plane S_1 and the z_2-z_c plane (S_2) to the 3D space to form the intersection shown as the black cube (S). The figure also shows projection of a point z onto the lower-dimensional subspaces in the z_1-z_c and z_2-z_c planes.

*3) **Subsystem Trajectories**:* Since each subsystem in (11) is self-contained, we can denote the subsystem trajectories $\xi_i(s; x_i, t, u(\cdot))$. The subsystem trajectories satisfy the subsystem dynamics and initial condition:

$$\begin{aligned} & \frac{d}{ds}\xi(s; x_i, t, u(\cdot)) = g_i(\xi(s; x_i, t, u(\cdot)), u(s)) \\ & \xi_i(t; x_i, t, u(\cdot)) = x_i \end{aligned} \tag{18}$$

The full system trajectory and subsystem trajectories are simply related to each other via the projection operator:

$$\text{proj}_{\mathcal{X}_i}(\zeta(s; z, t, u(\cdot)) = \xi_i(s; x_i, t, u(\cdot)) \tag{19}$$

where $x_i = \text{proj}_{\mathcal{X}_i}(z)$.

B. Goals of This Paper

We assume that the full system unsafe set $\mathcal{L}$ can be written in terms of the subsystem unsafe sets $\mathcal{L}_{x_1} \in \mathcal{X}_1, \mathcal{L}_{x_2} \in \mathcal{X}_2$ in the way depicted in Fig. 2:

$$\mathcal{L} = \text{proj}^{-1}(\mathcal{L}_{x_1}) \cap \text{proj}^{-1}(\mathcal{L}_{x_2}) \tag{20}$$

In practice, this is not a strong assumption since many obstacles can be accurately modeled as rectangular prisms in position space. In fact, the unsafe set described by (20) can even be arbitrarily shaped in the overlapping states. In addition, such an assumption is reasonable since the full-dimensional unsafe set should at least be representable in some way in the lower-dimensional spaces. However, in the worst case, taking $\mathcal{L}_{x_i} = \text{proj}_{\mathcal{X}_i}(\mathcal{L})$ leads to a conservative approximation of the BRS. Also note that with the definition in (20), we have that $\text{proj}_{\mathcal{X}_i}(\mathcal{L}) = \mathcal{L}_{\mathcal{X}_i}$.

Next, we define the subsystem BRSs $\mathcal{V}_{x_1}, \mathcal{V}_{x_2}$ the same way as in (3), but with the subsystems in (11) and subsystem unsafe sets $\mathcal{L}_{x_1}, \mathcal{L}_{x_2}$, respectively:

$$\mathcal{V}_{x_i}(t) = \{x_i : \forall u(\cdot) \in \mathbb{U}, \xi_i(0; x_i, t, u(\cdot)) \in \mathcal{L}_{x_i}\} \tag{21}$$

Given a system in the form of (11) with unsafe set that can be represented by (20), our goal is to first compute the subsystem BRSs $\mathcal{V}_{x_1}(t), \mathcal{V}_{x_2}(t)$, and then construct the full system BRS $\mathcal{V}(t)$ exactly. This process dramatically reduces computation complexity by decomposing the higher-dimensional system into two lower-dimensional subsystems. Specifically, we will show

$$\mathcal{V}(t) = \text{proj}^{-1}(\mathcal{V}_{x_1}(t)) \cap \text{proj}^{-1}(\mathcal{V}_{x_2}(t)) \tag{22}$$

It is important to note that if the subsystem states x_1, x_2 have no overlapping states (and are therefore decoupled), the above statement is relatively intuitive and easy to show; however, the subsystems have overlapping states in the partition y_3 and overlapping controls u. Our main result in this paper proves that despite this coupling, (22) still holds.

IV. Self-Contained Subsystems

With the background and definitions established, we now show the main result in a theorem, which relates lower-dimensional BRSs to the full-dimensional BRS we would like to compute. The consequence of the theorem is that for systems of the form (11), one can obtain the *exact* full-dimensional BRS by first computing the lower-dimensional BRSs $\mathcal{V}_{\mathcal{X}_i}$, and then constructing the full-dimensional BRS $\mathcal{V}(t)$ via (22). We first prove a lemma involving a key property of the projection operator.

Lemma 1: Let $\bar{z} \in \mathcal{Z}, \bar{x}_i = \text{proj}_{\mathcal{X}_i}(\bar{z}), \mathcal{S}_i \subseteq \mathcal{X}_i$ for some subsystem i. Then $\bar{x}_i \in \mathcal{S}_i \Leftrightarrow \bar{z} \in \text{proj}^{-1}(\mathcal{S}_i)$.

Proof: Forward direction: Suppose $\bar{x}_i \in \mathcal{S}_i$, then trivially $\exists x_i \in \mathcal{S}_i, \text{proj}_{\mathcal{X}_i}(\bar{z}) = x_i$ (the x_i that "exists" is just $\bar{x}_i$ itself). By the definition of back projection in (17), we have $\bar{z} \in \text{proj}^{-1}(\mathcal{S}_i)$.

Backward direction: Suppose $\bar{z} \in \text{proj}^{-1}(\mathcal{S}_i)$, then by the definition of back projection in (17), we have $\exists x_i \in \mathcal{S}_i, \text{proj}_{\mathcal{X}_i}(\bar{z}) = x_i$.

Let such an $x_i \in \mathcal{S}_i$ be denoted $\hat{x}_i$, and suppose $\bar{x}_i \notin \mathcal{S}_i$. Then, we must have $\hat{x}_i \neq \bar{x}_i$, which is a contradiction, since $\bar{x}_i = \text{proj}_{\mathcal{X}_i}(\bar{z}) = \hat{x}_i$. ■

Corollary 1: If $\mathcal{S} = \text{proj}^{-1}(\mathcal{S}_1) \cap \text{proj}^{-1}(\mathcal{S}_2)$, then

$$\bar{z} \in \mathcal{S} \Leftrightarrow \forall i, \bar{x}_i \in \mathcal{S}_i \tag{23}$$

We now prove our main result.

Theorem 1: **System decomposition for computing the BRS**. Suppose that the full system in (1) can be decomposed into the form of (11), then

$$\begin{aligned} &\mathcal{L} = \text{proj}^{-1}(\mathcal{L}_{x_1}) \cap \text{proj}^{-1}(\mathcal{L}_{x_2}) \\ &\quad \Rightarrow \mathcal{V}(t) = \text{proj}^{-1}(\mathcal{V}_{x_1}(t)) \cap \text{proj}^{-1}(\mathcal{V}_{x_2}(t)) \end{aligned} \tag{24}$$

Proof: We will prove Theorem 1 by proving the following equivalent statement:

$$\bar{z} \in \mathcal{V}(t) \Leftrightarrow \bar{z} \in \text{proj}^{-1}(\mathcal{V}_{x_1}(t)) \cap \text{proj}^{-1}(\mathcal{V}_{x_2}(t)) \tag{25}$$

By the definition of BRS in (3), we have

$$\bar{z} \in \mathcal{V}(t) \Leftrightarrow \forall u(\cdot) \in \mathbb{U}, \zeta(0; \bar{z}, t, u(\cdot)) \in \mathcal{L} \tag{26}$$

Consider the property (19), and let

$$\begin{aligned} \bar{x}_i &= \text{proj}_{\mathcal{X}_i}(\bar{z}) \\ \xi_i(0; \bar{x}_i, t, u(\cdot)) &= \text{proj}_{\mathcal{X}_i}(\zeta(0; \bar{z}, t, u(\cdot))) \end{aligned} \tag{27}$$

Noting that $\mathcal{L} = \text{proj}^{-1}(\mathcal{L}_{x_1}) \cap \text{proj}^{-1}(\mathcal{L}_{x_2})$ and using Corollary 1, we have that (26) is equivalent to

$$\forall i, \forall u(\cdot), \xi_i(0; \bar{x}_i, t, u(\cdot)) \in \mathcal{L}_{x_i} \tag{28}$$

which, by the definition of the subsystem BRS (21), is in turn equivalent to $\forall i, \bar{x}_i \in \mathcal{V}_{x_i}(t)$. By Lemma 1, this is equivalent to $\forall i, \bar{z} \in \text{proj}^{-1}(\mathcal{V}_{x_i}(t))$. ■

We now summarize our main theoretical result and its consequences with the following algorithm:

Algorithm 2: **SCS formulation**. Given an unsafe set $\mathcal{L}$ that can be decomposed as $\mathcal{L} = \text{proj}^{-1}(\mathcal{L}_{x_1}) \cap \text{proj}^{-1}(\mathcal{L}_{x_2})$ and SCSs with dynamics in the form (11), the full-dimensional BRS can be computed as follows:

1) Define the implicit surface functions representing the subsystem unsafe sets $\mathcal{L}_{x_1}, \mathcal{L}_{x_2}$.
2) Repeat for $i = 1, 2$: For ith SCS, compute its BRS by solving (6) in the space of $\mathcal{X}_i$.
3) Construct the full-dimensional BRS as follows: $\mathcal{V}(t) = \text{proj}^{-1}(\mathcal{V}_{x_1}(t)) \cap \text{proj}^{-1}(\mathcal{V}_{x_2}(t))$. By Theorem 1, the full-dimensional BRS is exactly constructed.

V. Numerical Examples

We now present two numerical examples to illustrate our method. For each example, we present a common dynamical system that can be decomposed into the form of (11). The first example, the 3D Dubins Car, illustrates that our decomposition method produces the exact full-dimensional BRS at a substantially lower computation cost. The second example, the 6D Acrobatic Quadrotor, demonstrates that our

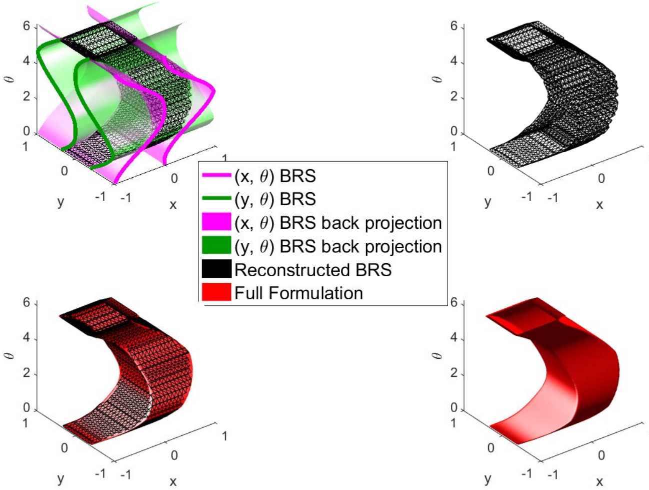


Fig. 3: Comparison of the Dubins Car BRS $\mathcal{V}(t), t = 0.5$ computed using the full and SCS formulations.

technique enables the exact computation of a BRS that was previously intractable to compute with the full formulation.

A. Dubins Car

The Dubins Car is a well-known system whose dynamics are given by (12). This system is only 3D, and its BRS can be tractably computed in the full-dimensional space, so we use it to compare the full formulation with the SCS formulation.

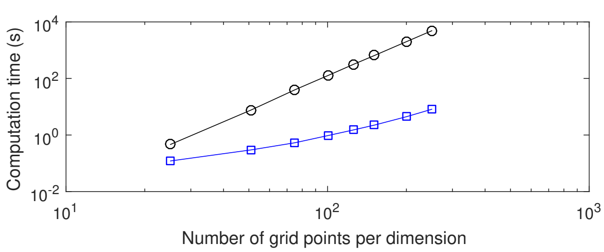


Fig. 4: Computation time of the full and SCS formulations in log scale for the Dubins Car example.

As previously mentioned, the Dubins Car dynamics can be decomposed according to (13). For this example, we computed the BRS from the unsafe set $\mathcal{L} = \{(p_x, p_y, \theta) : |p_x|, |p_y| \leq 0.5\}$. Such an unsafe set can be used to model an obstacle that the vehicle must avoid. The interpretation of the BRS $\mathcal{V}(t)$ is the set of states from which a collision with the obstacle may occur after a duration of t.

From $\mathcal{L}$, we computed the BRS $\mathcal{V}(t)$ of time horizon $t = 0.5$. The resulting full formulation BRS is shown in Fig. 3 as the red surface which appears in the two bottom subplots.

To compute the BRS using the SCS formulation, we write $\mathcal{L} = \text{proj}^{-1}(\mathcal{L}_{x_1}) \cap \text{proj}^{-1}(\mathcal{L}_{x_2})$, with

$$\begin{aligned} \mathcal{L}_{x_1} &= \{(p_x, \theta) : |p_x| \leq 0.5\} \\ \mathcal{L}_{x_2} &= \{(p_y, \theta) : |p_y| \leq 0.5\} \end{aligned} \tag{29}$$

From these lower-dimensional unsafe sets, we computed the lower-dimensional BRSs $\mathcal{V}_{\mathcal{X}_1}(t)$ and $\mathcal{V}_{\mathcal{X}_2}(t)$, and then constructed the full-dimensional BRS $\mathcal{V}(t)$ using Theorem 1: $\mathcal{V}(t) = \text{proj}^{-1}(\mathcal{V}_{x_1}(t)) \cap \text{proj}^{-1}(\mathcal{V}_{x_2}(t))$. The subsystem BRSs and their back projections are shown in magenta and green in the top left subplot of Fig. 3. The constructed BRS is shown in three subplots of Fig. 3 as the black mesh.

In the bottom left subplot of Fig. 3, we superimpose the BRS computed using the two methods. The results are indistinguishable. However, the SCS formulation allows the computation to be done significantly faster. An additional benefit of the SCS formulation is a slightly more accurate numerical solution due to reduced numerical dissipation.

Fig. 4 shows that the direct computation of the BRS in 3D becomes very time-consuming as the number of grid points per dimension is increased, while the computation using the SCS formulation hardly takes any time in comparison. Directly computing the BRS with 251 grid points per dimension using the full formulation took approximately 80 minutes, while computing the BRS using the SCS formulation only took approximately 30 seconds! Computations were timed on a desktop computer with an Intel Core i7-2600K processor and 16GB of random-access memory.

B. The 6D Acrobatic Quadrotor

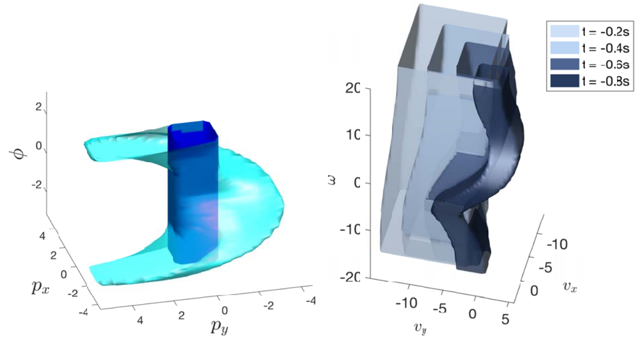
This example illustrates the ability of the SCS formulation to produce BRSs for high-dimensional systems that would be otherwise intractable to analyze by current HJ-based methods. Using the new SCS formulation we can accurately compute a BRS for the full 6D system, whose state is $z = (p_x, v_x, p_y, v_y, \phi, \omega)$; the dynamics are given by [25]:

$$\begin{bmatrix} \dot{p}_x \\ \dot{v}_x \\ \dot{p}_y \\ \dot{v}_y \\ \dot{\phi} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v_x \\ -\frac{1}{m} C_D^v v_x - \frac{T_1}{m} \sin\phi - \frac{T_2}{m} \sin\phi \\ v_y \\ -\frac{1}{m}(mg + C_D^v v_y) + \frac{T_1}{m}\cos\phi + \frac{T_2}{m}\cos\phi \\ \omega \\ -\frac{1}{I_{yy}} C_D^\phi \omega - \frac{l}{I_{yy}} T_1 + \frac{l}{I_{yy}} T_2 \end{bmatrix} \tag{30}$$

where p_x, p_y, and ϕ represent the quadrotor's horizontal, vertical, and rotational positions, respectively. Their derivatives represent the velocity with respect to each corresponding positional state. The inputs T_1 and T_2 represent the thrust exerted on either end of the quadrotor, and the constant system parameters are m for mass, C_D^v for translational drag, C_D^ϕ for rotational drag, g for acceleration due to gravity, l for the length from the quadrotor's center to an edge, and I_{yy} for moment of inertia.

The state partitions of this system are $y_1 = (p_x, v_x), y_2 = (p_y, v_y), y_3 = (\phi, \omega)$. We decompose the system into the following subsystems: $x_1 = (y_1, y_3) = (p_x, v_x, \phi, \omega)$, $x_2 = (y_2, y_3) = (p_y, v_y, \phi, \omega)$.

For this example we will compute the BRS that describes the set of initial conditions from which the system may enter the unsafe set after a given time period t despite best possible control. We define the unsafe set as a square



(a) 3D slice of the constructed 6D BRS at $v_x = v_y = 1$ m/s, $\omega = 0$ rad/s. The unsafe set is in dark blue, with the BRS in light blue.

(b) 3D slices of the constructed 6D BRS at $p_x, p_y = 1.5$ m, $\phi = 1.5$ rad at different points in time. The sets become darker as t becomes more negative.

Fig. 5: Slices of the BRS for the 6D Quadrotor model.

of length 1m centered at $(p_x, p_y) = (0, 0)$ described by $\mathcal{L} = \{(p_x, v_x, p_y, v_y, \phi, \omega) : |p_x|, |p_y| \leq 1\}$. This can be interpreted as a positional box centered at the origin that must be avoided for all angles and velocities. From the unsafe set, we define $l(z)$ such that $l(z) \leq 0 \Leftrightarrow x \in \mathcal{L}$. This unsafe set must be decomposed to provide a suitable unsafe set for each subsystem. This is done by letting $\mathcal{L}_{x_i}$, $i = 1, 2$ be

$$\begin{aligned} \mathcal{L}_{x_1} &= \{(p_x, v_x, \phi, \omega) : |p_x| \leq 1\} \\ \mathcal{L}_{x_2} &= \{(p_y, v_y, \phi, \omega) : |p_y| \leq 1\} \end{aligned} \tag{31}$$

The BRS of each 4D subsystem is computed and then combined into the 6D BRS using the SCS formulation. To visually depict the 6D BRS, 3D slices of the BRS along the positional and velocity axes were computed. Fig. 5a shows a 3D slice in (p_x, p_y, ϕ) space at $v_x = v_y = 1$ m/s, $\omega = 0$ rad/s. The dark blue set represents the unsafe set $\mathcal{L}$, with the BRS in light blue.

In Fig. 5b, 3D slices in (v_x, v_y, ω) space are visualized at $p_x, p_y = 1.5$ m, $\phi = 1.5$ rad. These colored sets represent the BRS at different points in time. As t becomes more negative, the sets become smaller because given a longer time horizon, it becomes easier to avoid the obstacle, reducing the number of states from which collision is inevitable.

VI. Conclusion

The SCS formulation that we proposed for computing BRSs significantly reduces computation burden, and makes many previously intractable computations possible. At the same time, the computation savings do *not* come at the cost of optimality: the full-dimensional BRS can be computed exactly in lower-dimensional subspaces, without relying on approximate set representations through simple shapes such as polytopes. The construction of the full-dimensional BRS from lower-dimensional BRSs is exact even when the subsystem dynamics are coupled.

References

[1] E. N. Barron, "Differential Games with Maximum Cost," *Nonlinear analysis: Theory, methods & applications*, pp. 971–989, 1990.

[2] 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*, vol. 50, no. 7, pp. 947–957, July 2005.

[3] K. Margellos and J. Lygeros, "Hamilton-Jacobi Formulation for Reach-Avoid Differential Games," *IEEE Transactions on Automatic Control*, vol. 56, no. 8, Aug 2011.

[4] O. Bokanowski and H. Zidani, "Minimal time problems with moving targets and obstacles," *{IFAC} Proceedings Volumes*, vol. 44, no. 1, pp. 2589 – 2593, 2011.

[5] J. Ding, J. Sprinkle, S. S. Sastry, and C. J. Tomlin, "Reachability calculations for automated aerial refueling," in *IEEE Conference on Decision and Control*, Cancun, Mexico, 2008.

[6] E. M. Vaisbord and V. I. Zhukovskii, *Introduction to Multi-player Differential Games and Their Applications*. Routledge, 1988.

[7] I. Mitchell, "Application of level set methods to control and reachability problems in continuous and hybrid systems," Ph.D. dissertation, Stanford University, 2002.

[8] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996.

[9] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, 2002.

[10] I. Mitchell, *A Toolbox of Level Set Methods*, 2009. [Online]. Available: http://people.cs.ubc.ca/ mitchell/ToolboxLS/index.html

[11] A. Majumdar and R. Tedrake, *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*. Springer Berlin Heidelberg, 2013, ch. Robust Online Motion Planning with Regions of Finite Time Invariance, pp. 543–558.

[12] T. Dreossi, T. Dang, and C. Piazza, "Parallelotope bundles for polynomial reachability," in *19th International Conference on Hybrid Systems: Computation and Controls*, 2016.

[13] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis: internal approximation," *Systems & control letters*, vol. 41, no. 3, pp. 201–211, 2000.

[14] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont, "Lagrangian methods for approximating the viability kernel in high-dimensional systems," *Automatica*, July 2013.

[15] J. Darbon and S. Osher, "Algorithms for overcoming the curse of dimensionality for certain hamilton-jacobi equations arising in control theory and elsewhere," *arXiv preprint arXiv:1605.01799*, 2016.

[16] I. M. Mitchell and C. J. Tomlin, "Overapproximating reachable sets by hamilton-jacobi projections," *Journal of Scientific Computing*, vol. 19, no. 1-3, pp. 323–346, 2003.

[17] M. Chen*, S. Herbert*, and C. J. Tomlin, "Fast reachable set approximations via state decoupling disturbances," in *55th IEEE Conference on Decision and Control*, 2016.

[18] 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*, 2011, pp. 103–112.

[19] J. F. Fisac, M. Chen, C. J. Tomlin, and S. S. Shankar, "Reach-avoid problems with time-varying dynamics, targets and constraints," in *18th International Conference on Hybrid Systems: Computation and Controls*, 2015.

[20] L. C. Evans and P. E. Souganidis, "Differential games and representation formulas for solutions of Hamilton-Jacobi-Isaacs equations," *Indiana Univ. Math. J.*, vol. 33, no. 5, pp. 773–797, 1984.

[21] E. A. Coddington and N. Levinson, *Theory of ordinary differential equations*. Tata McGraw-Hill Education, 1955.

[22] P. Varaiya, "On the existence of solutions to a differential game," *SIAM Journal on Control*, vol. 5, no. 1, pp. 153–162, 1967.

[23] M. G. Crandall and P.-L. Lions, "Viscosity solutions of Hamilton-Jacobi equations," *Transactions of the American Mathematical Society*, vol. 277, no. 1, pp. 1–42, 1983.

[24] M. G. Crandall, L. C. Evans, and P. L. Lions, "Some properties of viscosity solutions of hamilton-jacobi equations," *Transactions of the American Mathematical Society*, vol. 282, no. 2, p. 487, April 1984.

[25] J. H. Gillula, G. M. Hoffmann, H. Huang, M. P. Vitus, and C. J. Tomlin, "Applications of hybrid reachability analysis to robotic aerial vehicles," *International Journal of Robotics Research*, vol. 30, no. 3, pp. 335–354, March 2011.