Reachability Analysis for Black-Box Dynamical Systems

Vamsi Krishna Chilakamarri*[1], Zeyuan Feng*[2], and Somil Bansal[2]

***Abstract*— Hamilton-Jacobi (HJ) reachability analysis is a powerful framework for ensuring safety and performance in autonomous systems. However, existing methods typically rely on a white-box dynamics model of the system, limiting their applicability in many practical robotics scenarios where only a black-box model of the system is available. In this work, we propose a novel reachability method to compute reachable sets and safe controllers for black-box dynamical systems. Our approach efficiently approximates the Hamiltonian function using samples from the black-box dynamics. This Hamiltonian is then used to solve the HJ Partial Differential Equation (PDE), providing the reachable set of the system. The proposed method can be applied to general nonlinear systems and can be seamlessly integrated with existing reachability toolboxes for white-box systems to extend their use to black-box systems. Through simulation studies on a black-box slip-wheel car and a quadruped robot, we demonstrate the effectiveness of our approach in accurately obtaining the reachable sets for black-box dynamical systems.**

I. Introduction

As robots operate in increasingly complex environments, ensuring safe interactions with their surroundings is crucial. A widely used approach to design safe controllers for robotic systems is reachability analysis [1], which involves computing the Backward Reachable Tube (BRT) of the system. Intuitively, the BRT represents the set of all initial states from which the system will inevitably enter a failure set, such as obstacles for a navigation robot, despite the best control effort. Therefore, the complement of the BRT defines the safe states for the system.

Several methods have been developed to compute BRTs for dynamical systems, including approaches that approximate BRTs using Zonotopes [2], [3], Sum-of-Squares programming [4], [5], and parallelotopes [6]. For a comprehensive survey on BRT computation methods, we refer interested readers to [1], [7]. Methods that compute BRTs accurately up to numerical precision include level set methods [8]–[11]. Level set methods are rooted in Hamilton-Jacobi (HJ) Reachability analysis, where BRT computation is formulated as an optimal control problem. This amounts to solving a partial differential equation, called the HJ-PDE. Correspondingly, techniques have been developed to solve this PDE numerically over a grid [12], [13] or through learning-based methods [7], [14], [15].

However, most of these methods rely on an analytical (white-box) dynamics model of the system to compute the BRT. Unfortunately, as robotic systems grow increasingly complex, this requirement becomes prohibitive. For instance, recent advances in simulation technologies [16], [17] allow for the simulation of complex dynamical systems, but the underlying dynamics are often available only as black-box models. This limitation renders current reachability methods ineffective for the safety analysis of such systems.

Several methods have been developed to compute BRTs and design safe controllers for black-box dynamical systems. These methods can be broadly classified into model-based and model-free approaches. Model-based methods first obtain or learn a white-box model of the system, possibly with uncertainty bounds, and then apply traditional reachability techniques [18]–[22]. While promising, the accuracy of the computed BRT heavily depends on the accuracy of the learned dynamics model, which can undermine the reliability of the resultant safety assurances. Other approaches leverage side information about the black-box dynamics to compute reachable sets, such as assuming linear time-invariant dynamics [23], utilizing Lipschitz constants of the dynamics and state monotonicity [24], [25], or employing polynomial templates to construct reachable sets [26]–[28].

In contrast, model-free methods do not explicitly construct a white-box model of the system. These include sampling-based techniques [29], such as scenario optimization [30]–[32] and adversarial sampling methods [33]. Another line of model-free approaches uses Reinforcement Learning (RL) to estimate HJ reachability solutions, thereby naturally handling unknown system dynamics [34]–[36]. While promising for black-box reachability problems, RL methods typically lack safety assurances, especially when function approximations are used, and the accuracy of the BRT is sensitive to the choice of discount factor. Additionally, these methods are not compatible with existing reachability techniques, such as level-set toolboxes [12], [13].

In this work, we extend level set methods to compute reachable sets for black-box dynamics models. Our *key idea* is to construct a zeroth-order approximation of the Hamiltonian function in the HJ-PDE using samples from the black-box dynamics. This approximation, which can be efficiently computed with modern computational tools, is then used to solve the HJ-PDE and approximate the BRT for the black-box system. We also provide an algorithm to further improve the efficiency of the approximation if the underlying black-box system is control-affine.

Our method can be readily integrated with existing reachability toolboxes – such as grid-based level-set toolboxes [12], [13] and learning-based methods [7] – to synthesize BRTs and optimal safe policies for black-box dynamical systems. We demonstrate the implementation of our approach in multiple reachability computation tools, including a bicycle robot system using the level-set toolbox [13], and a slip-wheel car system and a high-dimensional quadruped system using DeepReach [7], a learning-based reachability framework. In all cases, we compare our approach with various model-

* Denotes equal contribution. [1]Author is with Indian Institute of Technology, Madras, India. [2]Authors are with ECE at the University of Southern California, LA, USA. Corresponding author: {zeyuanfe@usc.edu}

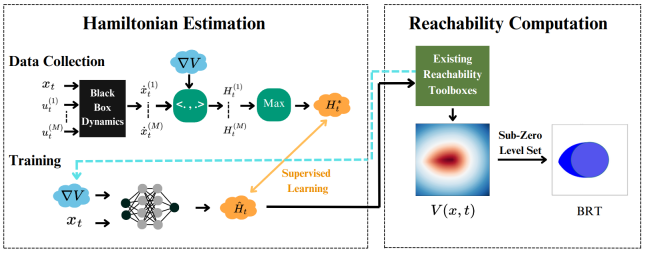


Fig. 1: Caption

based and model-free methods and show its effectiveness in providing a more accurate approximation of the BRT.

II. Problem Setup

Consider a black-box dynamical system with *unknown*, possibly nonlinear, time-invariant dynamics: $\dot{x} = f(x,u)$, where $x \in \mathbb{R}^n$ is the state, and $u \in \mathcal{U} \subset \mathbb{R}^{n_u}$ is the control input. Although f is unknown, we assume that we can query the system to obtain the state transition $(x(t), u(t), x(t+\Delta))$ (e.g., through a simulator), where Δ is the time step. We are also given a failure set $\mathcal{L} \subset \mathbb{R}^n$ that represents undesirable states for the system (e.g., obstacles for a mobile robot), and thus, needs to be avoided.

Our main goal in this work is to compute the backward reachable tube (BRT) of the system, $\mathcal{B}$, defined as the set of all initial states from which the system will inevitably enter $\mathcal{L}$ within the time horizon $[0,T]$, regardless of the control strategy. Mathematically,

$$\mathcal{B} = \left\{ x : \forall u(\cdot) \in \mathcal{U}, \exists \tau \in [0,T], \xi_{x,0}^{u(\cdot)}(\tau) \in \mathcal{L} \right\}, \quad (1)$$

where $\xi_{x,t}^{u(\cdot)}(\tau)$ denote the system state achieved at time τ by applying the policy $u(\cdot)$, starting from an initial state x at time t. As such, the BRT contains the states that are unsafe for the system and should be avoided. Correspondingly, our second goal is to synthesize a safe control policy $u^*(\cdot)$ for the system that prevents the system from entering $\mathcal{B}$.

Running Example: A Bicycle Robot. We now introduce a running example to explain the key ideas. The robot is modeled as a 5-dimensional system with state $x = [p_x, p_y, v, \phi, \delta]$, where p_x, p_y is the planar position, v is velocity, ϕ is the heading, and δ is the steering angle. The robot control is given by $u = [a, \omega]$, where $a \in [-2, 2]$ m/s^2 is the linear acceleration and $\omega \in [-2, 2]$ rad/s is the angular velocity. We use $L = 1$ in our case study. The robot dynamics are control affine and are modeled as:

$$\dot{p}_x = v\cos\phi,\ \dot{p}_y = v\sin\phi,\ \dot{v} = a,\ \dot{\phi} = \frac{v}{L}\tan\delta,\ \dot{\delta} = \omega$$

However, to test our methods, we assume that the dynamics are not known explicitly and are presented as a black box to us. Specifically, given a state x and control input u, the black-box dynamics simulates the system for a timestep $\Delta = 0.001$s, and outputs the resultant state $x(t+\Delta)$. The failure set $\mathcal{L}$ is given by a circle of radius 2.5m centered at the origin. Our goal is to compute the BRT of the system corresponding to this failure set while only relying on the black-box dynamics.

III. Background: Hamilton-Jacobi Reachability

HJ Reachability analysis is a powerful tool for synthesizing BRTs for dynamical systems [8]. Typically, users start with specifying a target function $l : \mathbb{R}^n \to \mathbb{R}$, with its sub-zero level set being the failure set, i.e., $\mathcal{L} = \{x : l(x) \le 0\}$. Given l, the BRT computation is formulated as an optimal control problem with the cost function:

$$J(x,t,u(\cdot)) = \min_{\tau \in [t,T]} l\left(\xi_{x,t}^{u(\cdot)}(\tau)\right). \quad (2)$$

Intuitively, $J \le 0$ implies that the system entered the failure set during the time horizon $[t,T]$ under $u(\cdot)$. Thus, to capture safety violations under an optimal policy, we can compute the corresponding value function:

$$V(x,t) = \sup_{u(\cdot)\in\mathcal{U}_{[t,T]}} \min_{\tau\in[t,T]} l\left(\xi_{x,t}^{u(\cdot)}(\tau)\right). \quad (3)$$

The value function can be obtained by solving the Hamilton-Jacobi-Bellman Variational Inequality (HJB-VI):

$$\begin{aligned} \min\{D_t V(x,t) + H(x,t), l(x) - V(x,t)\} = 0, \\ V(x,T) = l(x), \end{aligned} \quad (4)$$

where $H(x,t)$ is called the Hamiltonian and encodes the role of system dynamics:

$$H(x,t) = \max_{u\in\mathcal{U}} \langle \nabla V(x,t)\ ,\ f(x,u)\rangle. \quad (5)$$

Here, D_t and ∇ denote the time and spatial derivatives of the value function. We refer the interested readers to [12], [37] for a more detailed exposition of reachability analysis.

Once the value function is computed, the BRT is given by the sub-zero level set of the value function:

$$\mathcal{B} = \{x : V(x,0) \le 0\}. \quad (6)$$

Along with the BRT, the value function provides an optimal safety controller to keep the system outside the BRT:

$$u^*(x,t) = \arg\max_u \langle \nabla V(x,t), f(x,u)\rangle. \quad (7)$$

A. An Overview of Methods to Solve the HJB-VI

Traditionally, numerical methods are employed to solve the HJB-VI over a grid representation of the state space [12], [13], wherein the time and spatial derivatives in (4) are approximated numerically over the grid. These methods rely on an analytical expression of f to compute the Hamiltonian, i.e., to solve the optimization problem in (5). This is the source of the key challenge in applying these methods to black-box dynamical systems.

While the grid-based methods offer accurate solutions for low-dimensional problems, they suffer from the curse of dimensionality. Consequently, learning-based methods have been developed to solve HJB-VI for high-dimensional cases. Here, we present an overview of one such method, DeepReach [7], which we will use in our case studies. DeepReach uses a self-supervised learning scheme to estimate the solution of HJB-VI. In particular, the value function is approximated as $V_\theta(x,t) = l(x) + (T-t)\cdot O_\theta(x,t)$, where $O_\theta(x,t)$ is the output of a sinusoidal neural network (NN)

and θ represents trainable parameters. The loss function to train the NN is given by the HJB-VI residual error:

$$h_1\left(x_i, t_i; \theta\right) = \| \min\left\{D_t V_\theta\left(x_i, t_i\right) + H\left(x_i, t_i\right), \\ l\left(x_i\right) - V_\theta\left(x_i, t_i\right)\right\} \| \tag{8}$$

Once again, the loss function in (8) requires computing the Hamiltonian, which in turn, relies on a white-box dynamics model. In this work, we aim to alleviate this limitation.

IV. Solving HJB-VI for Black-Box Dynamics

This section presents an approach to compute BRTs for general black-box dynamical systems and then discuss a tailored approach under a control-affine assumption.

A. A General Approach to Compute BRTs for Black-Box Dynamical Systems

Our approach consists of computing an approximation of the Hamiltonian function using samples of system dynamics. The approximated Hamiltonian is then used in the HJB-VI in (4) to compute the value function and corresponding BRT.

Hamiltonian Estimation. We propose to train a NN, denoted as H_ν, to predict the Hamiltonian given a state x and the spatial gradient of the value function, $\nabla V(x,t)$. Using a NN allows for a quick inference of the Hamiltonian at the grid points (for numerical methods) or at the data samples (for learning-based methods) during the value function computation, though other function approximations can also be used.

To train the Hamiltonian estimator, we collect a dataset $\mathcal{D}_H$ as follows: we randomly sample a batch of (normalized) spatial gradient vectors and system states, $\left\{[x\ \nabla \bar{V}(x,t)]\right\}$. Next, for each state-gradient pair, we randomly sample a set of control inputs and query the corresponding next states of the system through black-box model in a parallel manner. Finally, we compute the Hamiltonian corresponding to each control sample using (5) and compute the empirical maximum, $\bar{H}$. The detailed dataset collection procedure is described in Algorithm 1. Once the dataset is collected, the Hamiltonian estimator is trained via supervised learning to optimize:

$$h_H\left(x_i, \nabla \bar{V}(x,t)_i; \nu\right) := \|H_\nu(x_i, \nabla \bar{V}(x,t)_i) - \bar{H}\|. \tag{9}$$

After being trained, the parameters ν are frozen when H_ν is used to predict the Hamiltonian during the value function computation. The Hamiltonian prediction is given by

$$\hat{H}\left(x, \nabla V(x,t); \nu\right) = \|\nabla V(x,t)\| H_\nu\left(x, \frac{\nabla V(x,t)}{\|\nabla V(x,t)\|}\right), \tag{10}$$

where we account for the normalization of $\nabla V(x,t)$. We next solve the HJB-VI in (4) as usual with H replaced by $\hat{H}$ to obtain an approximation $\hat{V}(x,t)$ of the value function.

Remark 1: Note that the actual distribution of $\nabla V(x,t)$ during the value function calculation can be different from the uniform distribution that is used to collect the data to train H_ν. This can cause inaccuracies in the predicted Hamiltonian, especially when the underlying dynamics are high-dimensional or stiff. For these reasons, we compute the value function once and augment the dataset $\mathcal{D}_H$ with data points $\left(\left[x\ \nabla \bar{V}(x,t)\right], \bar{H}\right)$, where the spatial derivative is sampled around the computed $\nabla \hat{V}(x,t)$. With the augmented dataset, we repeat the procedure to train H_ν and recompute the value function.

Obtaining the Optimal Safe Policy. The optimal safety control is obtained by training a neural network controller u_ψ using the dataset $\mathcal{D}_H$. The network takes x and $\nabla V(x,t)$ as inputs and predicts the optimal safety control. To train the controller network, we optimize the Mean Absolute Error loss between the normalized optimal control labels $\hat{u}^*(x)$ and the predicted control

$$h_c(x_i, t_i; \psi) = \|u_\psi(x_i, \nabla V(x_i, t_i)) - \hat{u}^*(x_i)\|. \tag{11}$$

Algorithm 1: Data Collection for Ham Estimation

Input: A pre-collected set of states $\mathcal{D}_X$, e.g., states sampled uniformly over a state space;
Output: $\mathcal{D}_H$;
Parameters: simulator time step Δ, number of samples, number of control samples;
Initialize $\mathcal{D}_H \leftarrow \emptyset$;
foreach *sample* **do**
  Sample $x \overset{\text{iid}}{\sim} \text{Uniform}(D_X)$;
  Sample $\nabla V(x,t) \overset{\text{iid}}{\sim} \text{Uniform}(\mathcal{C})$,
   $\mathcal{C} = \{x \in \mathbb{R}^n \mid |x_i| \leq 1, \forall i = 1, 2, ..., n\}$;
  $\nabla \bar{V}(x,t) \leftarrow \frac{\nabla V(x,t)}{\|\nabla V(x,t)\|}$;
  $\bar{H} \leftarrow -\infty$;
  foreach *control sample* **do**
    $u \overset{\text{iid}}{\sim} \text{Uniform}(\mathcal{U})$;
    Take action u from x, observe x_{next};
    if $\bar{H} < \langle \nabla V(x,t)\ ,\ \frac{x_{next}-x}{\Delta} \rangle$ **then**
      $\bar{H} \leftarrow \langle \nabla V(x,t)\ ,\ \frac{x_{next}-x}{\Delta} \rangle, \quad u^* \leftarrow u$;
  end
  $\mathcal{D}_H \leftarrow \mathcal{D}_H \cup \left\{\left(\left[x\ \nabla \bar{V}(x,t)\right], \bar{H}, u^*\right)\right\}$
end

Safety Assurances for the Obtained BRT. To reason about safety assurances under the proposed framework, we leverage the formal verification method proposed in [38] to obtain a probabilistic safe set from an approximate value function and corresponding optimal policy, $(\hat{V}, u_\psi)$. The overall idea is to provide a high-confidence bound δ on the value function error using conformal prediction. This results in a correction of the value function by δ. The corrected value function is then used to compute the BRT and the safe set.

Specifically, the method requires users to specify a confidence parameter $\beta \in (0,1)$ and a safety violation parameter $\epsilon \in (0,1)$. It then computes an error bound δ using conformal prediction to ensure that with at least $1-\beta$ probability:

$$\mathbb{P}_{x \in \mathcal{S}}\left(\min_{\tau \in [0,T]} l\left(\xi^{u_\psi}_{x,0}(\tau)\right) \leq 0\right) \leq \epsilon, \tag{12}$$

where $\mathcal{S} = \left\{x : x \in \mathcal{X}, \hat{V}(x,0) > \delta\right\}$. In other words, the probability of a state within the super-δ level set of $\hat{V}$ being actually unsafe during the rollouts under u_ψ is at most ϵ. The complement of $\mathcal{S}$, denoted as $\mathcal{B}_\epsilon$, thus represents a high-confidence estimate of the BRT.

B. Solving HJB-VI for Control-Affine Black-Box Systems

We now propose a variant of our approach to compute the value function under the assumption that the underlying black-box system is control-affine with box-constrained control inputs. Specifically, we now consider systems with dynamics $f(x,u) := f_1(x) + f_2(x)u$, with the control input $u := [u_1, \ldots, u_{n_u}] \in [\alpha_i - \beta_i, \alpha_i + \beta_i]^{n_u}$. This formalism aligns well with many real-world robotic systems, which are often control-affine in nature.

Our key observation is that for control-affine systems, the Hamiltonian in (5) is optimized at one of the extremal control inputs [8]. This is because the objective function in (5) is linear with respect to control variables $u_1, u_2, \ldots, u_{n_u}$ for control-affine systems, subjected to an n_u-dimensional hypercube centered at $(\alpha_1, \alpha_2, \ldots, \alpha_{n_u})$ and side length β_i in the ith dimension. Consequently, the optimization of the Hamiltonian reduces to a linear program (LP), whose solution lies at one of the corners of the hypercube. Thus, the Hamiltonian can be computed as:

$$H(x,t) = \max_{u \in \{\alpha_i - \beta_i, \alpha_i + \beta_i\}^{n_u}} \langle \nabla V(x,t)\ ,\ f(x,u) \rangle \tag{13}$$

The above equation suggests a tractable mechanism for evaluating the Hamiltonian for black-box control-affine systems. Specifically, by restricting the evaluation to a finite set of control inputs corresponding to the vertices of the hypercube, the Hamiltonian can be accurately obtained by using the control samples at these vertices. Thus, the Hamiltonian can be estimated as:

$$\hat{H}\left(x, \nabla V(x,t)\right) = \max_{u \in \{\alpha_i - \beta_i, \alpha_i + \beta_i\}^{n_u}} \langle \nabla V(x,t)\ ,\ \frac{x^u_{next} - x}{\Delta} \rangle \tag{14}$$

where x^u_{next} is the next state under control input u. Unlike the method proposed in Sec. IV-A, the proposed variant bypasses any need for data collection or network training for estimating the Hamiltonian, thereby avoiding any value function errors emanating from incorrect Hamiltonian estimation for control-affine systems.

Remark 2: If Δ is sufficiently small, the Hamiltonian estimation error tends to zero, and the BRT computed using the proposed scheme converges to the ground truth BRT. This follows immediately since as $\Delta \to 0$, the dynamics flow estimation becomes more accurate.

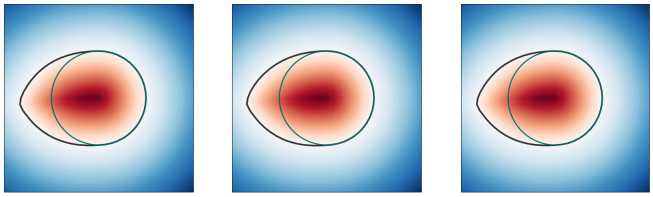
Remark 3: Even when the black-box system is not control-affine, the proposed variant generates a provably conservative BRT for the system because it always under-approximates the true Hamiltonian.

Remark 4: A key limitation of the proposed variant is that it is not scalable to high-dimensional input spaces, since it requires sampling 2^{n_u} controls in each iteration. To address this issue, we can individually determine the value of optimal control in each input dimension. Specifically, we sample a random nominal control input $\boldsymbol{u}_{nom} = (u_1^{nom}, u_2^{nom}, ..., u_{n_u}^{nom})$, and n_u extra controls given by $\boldsymbol{u}_i = (u_1^{nom}, ..., \alpha_i + \beta_i, ..., u_{n_u}^{nom})$, respectively. The optimal control in the i^{th} dimension can be obtained by comparing the Hamiltonian at the nominal control input and $\boldsymbol{u}_i$:

$$u_i^*(x) = \begin{cases} \alpha_i + \beta_i, & \text{if } \langle \nabla V(x,t)\ ,\ \frac{x^{\boldsymbol{u}_i}_{next} - x^{\boldsymbol{u}_{nom}}_{next}}{\Delta} \rangle \geq 0 \\ \alpha_i - \beta_i, & \text{otherwise} \end{cases} \tag{15}$$

The above method reduces the number of control samples from 2^{n_u} to $(n_u + 1)$, thereby extending the scalability to high-dimensional control spaces.

Obtaining Safe Policy. To obtain the safe policy, we can follow the same procedure as outlined in Sec. IV-A. However, since the optimal control is extremal in this case, one can also use a classifier network that predicts one of the two classes $\{\alpha_i + \beta_i, \alpha_i - \beta_i\}$ for the i^{th} dimension.



(a) Ground Truth (b) Ham-NN (c) Ham-CA

Fig. 2: Bicycle Robot: A slice of the value function for different methods at $t = 2$s. The zero level set (BRT) and the failure set are shown in black and green respectively.

C. Running Example

For the general approach (***referred to as Ham-NN here on*** for Hamiltonian via Neural Network), we approximate H_ν with a NN with 2 hidden layers and 64 neurons in each layer with ReLU activation. For training, we collected a dataset of 500K samples. For each state-gradient datapoint, we sample 10K controls to find the optimal control and Hamiltonian for training. Once collected, the NN was trained on this dataset using the loss function (9). Dataset collection took ≈ 5 mins, and the network training too 20 mins on a single NVIDIA GeForce RTX 3090 GPU worker. Once trained, the predicted Hamiltonian from the network was used to solve the HJB-VI and compute the value function using the Python Level Set Toolbox [13] over a grid of size [31, 31, 21, 51, 11].

Since the dynamics are control-affine in this case, we also compute the value function using the variant proposed in Sec. IV-B. For this variant (***referred to as Ham-CA here on*** for Hamiltonian via Control Affine dynamics), we sample the four corners of the 2D control hypercube and query the next state to estimate the Hamiltonian as per (14). Finally, for comparison purposes, we also compute the ground truth BRT using the actual dynamics. We evaluate the Mean Squared Error (MSE) of the Value function and the False Positive rate (FP%) for both approaches against the ground truth.

The BRT computation took 45 minutes for the Ham-NN and ground truth methods and around 90 minutes for the Ham-CA method. This can be explained by the requirement of computing the Hamiltonian at four different control inputs for the Ham-CA method, resulting in an overall higher computation time. The BRTs obtained using different methods are shown in Fig. 2. As evident from the figure, both of the proposed variants are able to obtain a high-quality approximation of the value function, without relying on the

analytical dynamics of the system. This is further confirmed by a very low MSE of 4.45×10^{-4} for Ham-NN and 1×10^{-12} for Ham-CA respectively. A significantly smaller MSE of Ham-CA can be attributed to the lack of any learning errors that might be present in the Ham-NN method, along with the fact that the dynamics are indeed control-affine in this case, so we expect a recovery of the ground-truth BRT from Ham-CA for small Δ. This is further aligned with the 0 FP (%) for the HAM-CA method (compared to a small but non-zero FP rate of 4.1×10^{-4} for the Ham-NN method).

V. Experiments

In this section, we conduct a comparative study on two reachability problems: an avoid problem for a 6D slip-wheel vehicle and an avoid problem for a quadruped robot.

Baselines. We compare the proposed methods (Ham-NN and Ham-CA) against model-based and model-free baselines. Given the high dimensionality of these case studies, we use DeepReach [7] to compute the value function and the BRT. This will also help illustrate how our method can be combined with different reachability toolboxes.

For model-based comparisons, we use the method from [22] that learns an ensemble NN model of the system dynamics along with state-dependent disturbance bound, and then use these dynamics to compute a robust BRT via DeepReach. **We call this method MB**.

For model-free comparisons, we use a time-dependent fitted value iteration RL method inspired by [34]. We also compare against a variant of this approach that additionally uses a discount factor γ during training. **We call these variants FVI and D-FVI, respectively**. DeepReach-based computations are done using the loss function in (8), whereas the RL-based baselines minimize the Bellman error:

$$\begin{aligned}\mathbf{h}_{\text{FVI}}(x_i,t_i) &= \|V(x_i,t_i) - \\ &\qquad \min\left\{l(x_i), \max_u V(x^u_{next}, t_i+\Delta)\right\}\|, \\ \mathbf{h}_{\text{D-FVI}}(x_i,t_i) &= \|V(x_i,t_i) - [(1-\gamma)l(x_i) + \\ &\qquad \gamma \min\left\{l(x_i), \max_u V(x^u_{next}, t_i+\Delta)\right\}]\|.\end{aligned} \tag{16}$$

Evaluation Metric. Our key evaluation metric is the volume of the verified BRT $\mathcal{B}_\epsilon$, μ_ϵ. A smaller μ_ϵ indicates a bigger safe set and hence a better performance. To quantify this, we sample N distinct states from the state space and count the number of states that fall within $\mathcal{B}_\epsilon$, denoted as n_ϵ. The percentage volume of the BRT is given as $\mu_\epsilon = 100 \times \frac{n_\epsilon}{N}$. We choose a large value of $N = 3 \times 10^6$ to attain samples from a significant portion of the state space. We use the verification method in [38] for all baselines, with $\beta = 10^{-10}$ and $\epsilon = 10^{-2}, 10^{-3}$ respectively. This corresponds to obtaining a safe set with 99% and 99.9% confidence levels, respectively.

A. Slip-Wheel Car System

In this example, we demonstrate how our methods perform against a black-box non-control-affine system. The dynamics, adapted from [39], represents a simplified 6D single-track vehicle model with the state $x = [p_x, p_y, \phi, U_x, U_u, r]^T$, where (p_x, p_y) are the Cartesian coordinates, ϕ is the yaw angle, U_x, U_y are body frame velocities, and r is the yaw rate. The control is represented by $u = [\delta, F_x]^T$, where δ is the steering angle, and F_x is the longitudinal tyre force. We further extend the dynamics to account for vehicle sliding when F_x applied exceeds the tire's friction cone. Specifically, when $F_x^2 + F_y^2 > \mu F_z$, a sliding friction is used to compute the tyre forces. Hence, simultaneously applying full brake and steering input will cause sliding, making a bang-bang, extremal controller unfavorable. The failure set is given by $\mathcal{L} := \left\{x : \sqrt{p_x^2 + p_y^2} \leq 2.5\right\}$. The time horizon T is 1.5 s.

For all baselines, we use a 3-layer sinusoidal NN with 256 hidden neurons per layer and 100k training iterations. The Adam optimizer with a learning rate of 2×10^{-5} is employed for training. For Ham-NN, both H_ν and u_ψ are 2-layer NNs with 128 neurons per layer. A dataset containing 1 million data points is collected to train these models, where 10^4 possible controls are sampled for each point to generate Hamiltonian and optimal control label. The data collection procedure took 20 minutes and training H_ν took another 15 minutes. For FVI and D-FVI baselines, the optimal control is estimated using a 6×6 grid of possible controls $(\delta, F_x) \in [-\pi/10, \pi/10] \times [-18794, 5600]$. We set the time step $\Delta = 0.002s$ and apply a discount factor annealing scheme where γ starts at 0.99 and gradually increases to 0.999 during training.

	Ham-NN	Ham-CA	FVI	D-FVI	MB
$\mu^{car}_{\epsilon=10^{-2}}$	**3.48%**	3.91%	5.03%	3.78%	5.56%
$\mu^{car}_{\epsilon=10^{-3}}$	**4.33%**	10.30%	7.43%	8.50%	5.56%
Training time	1.5h total	**0.5**h	7.8h	9.5h	2.0h
$\mu^{quadruped}_{\epsilon=10^{-2}}$	**6.87%**	16.96%	17.95%	18.04%	11.19%
$\mu^{quadruped}_{\epsilon=10^{-3}}$	**8.36%**	34.01%	34.29%	30.82%	17.50%
Training time	2.5h total	**2h**	7h	7h	3h

TABLE I: The BRT volume (μ) and training time for the slip-wheel car (top) and the quadruped (bottom) case studies.

The verified volume of the BRT under different methods, as well as total training time, is shown in Table I. Ham-NN consistently outperforms other methods in achieving a lower BRT volume (a higher safe set volume). Ham-CA requires the least training time as it bypasses the need for training a H_ν. In addition, it synthesizes a conservative BRT because the Hamiltonian estimated via hypercube sampling is always an underapproximation of the actual one. In spite of the conservatism, the volume of verified BRT still expands as the required safety level increases (i.e., ϵ decrease), which is illustrated by the growing unsafe set contours in Fig. 3b. This suggests that Ham-CA is more sensitive to learning errors. Another variant giving a conservative BRT is the model-based method since it captures the uncertainty in the dynamics model. This uncertainty, while allows it to safeguard against potential modeling errors, can lead to overly conservative behavior. This is also evident from the same BRT volume of MB for the two ϵ levels – while MB is over-conservative when the safety criterion is loose, it yields the second-smallest unsafe set when $\epsilon = 10^{-3}$ since the conservatism makes it more resilient to learning errors. On the other hand, the D-FVI variant achieves a similar unsafe volume compared to the Ham-NN at $\epsilon = 10^{-2}$,

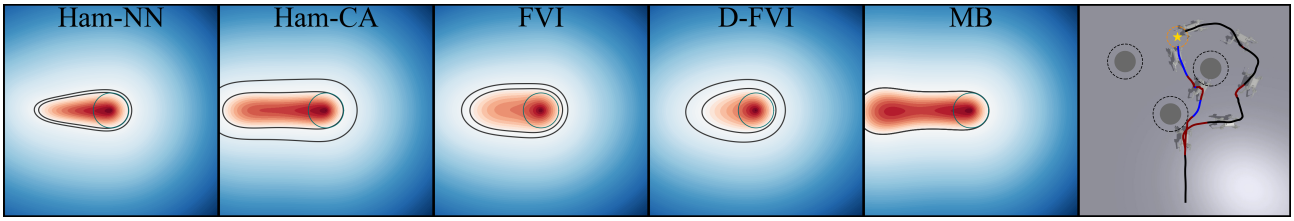


Fig. 3: The first five figures illustrate value function slices (corresponding to initial states $(p_x, p_y, 0, 12, 0, 0)$) for the slip-wheel car system. The failure set and sub-δ level sets (BRTs), with $\epsilon = 10^{-2}$ and 10^{-3}, are also shown. The last figure shows the safety-filtered trajectories for the quadruped example. The obstacle contours are shown in black dash lines and the star represents goal position. Maroon line segments indicate the activation of safety controller. Due to a less conservative BRT, the filtered trajectory (blue) with Ham-NN is able to identify a safe passage between the obstacles and lead to a more optimal performance compared to the MB baseline (black), which takes a detour.

as it samples a grid of controls to estimate the optimal one. However, this advantage comes at the cost of a much longer training time. Moreover, as we increase the required safety assurance level, the verified BRT volume increases significantly, indicating higher learning errors in D-FVI. In contrast, the BRT volume increases only modestly for Ham-NN, highlighting the robustness of the proposed approach.

B. Quadruped System

We take the last example to demonstrate how the proposed method can be applied to complex, high-dimensional systems. We use Isaac Gym [17] to simulate the robot and the underlying dynamics are unavailable. We leverage a pre-trained Reinforcement Learning (RL) policy from [40] for low-level control. The RL policy takes a high-level twist commands $u_h = [v_x^c, \omega^c]$ along with the robot's state x_{full} as inputs and outputs the desired joint angles to track u_h.

To simplify the problem, the state x_{full} is divided into low-frequency, important states x_{low} and high-frequency, less critical ones x_{high}. Here, $x_{low} = (p_x, p_y, \phi, v_x, v_y, \phi)$ includes robot's COM position, yaw angle, body frame velocities, and yaw rate, while $x_{high} = \left[\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{g}_p, \boldsymbol{c}, u_h^{last}\right]$ include the joint positions and velocities, projected gravity, foot contacts, and the previous high-level command. Since the RL controller follows specific gaits to track the twist command, we train an autoencoder to compress the x_{high} into a 2-dimensional latent representation $x_{latent} = [z_1, z_2]$. The low-frequency state and the latent state are then concatenated to create a condensed state representation $x_c = [x_{low}, x_{latent}]$ for the quadruped. This results in a new 8-dimensional black-box dynamics for DeepReach to learn on, with x_c as the state and u_h as the control. The black-box dynamics can thus be thought of as the composition of RL policy (to obtain the low-level control), the Isaac simulator (to obtain the next x_{full}), and the encoder (to obtain the next x_c). The failure set is defined as $\left\{x_c : \sqrt{p_x^2 + p_y^2} \leq 0.5\right\}$, which corresponds to an obstacle of radius 0.5.

All baselines use a 3-layer sinusoidal NN with 512 hidden neurons per layer, while other hyperparameters remain the same as in Section V-A. We collect a dataset of 1 million data points to train H_ν and u_ψ, which are 2-layer MLPs with 128 neurons per layer. Due to the relatively slower simulation queries in this case, we only sample 6 possible u_h, including the vertices of the control hypercube, to determine the optimal control.

The obtained results are presented in Table I. Similar to the slip-wheel system, Ham-NN outperforms all other baselines across both ϵ values. Given the higher complexity of the quadruped system, we see an even bigger gap between the performance of Ham-NN and other methods. Notably, both Ham-NN and MB exhibit better robustness to changes in ϵ compared to other variants. The MB variant is more robust due to its conservatism. For Ham-NN, we attribute this to the smoothening effect of the Hamiltonian estimator H_ν – the NN effectively smoothens out the sudden changes in the Hamiltonian in the parts of the state space where the dynamics are stiff, resulting in a simplified reachability problem for DeepReach and lower learning errors. Conversely, the Ham-CA variant is more sensitive to sudden changes in Hamiltonian and, therefore, performs worse than Ham-NN.

Lastly, we demonstrate the utility of the verified BRT by using it as a safety filter for a simple PD controller designed to reach a goal position in a cluttered environment (see Fig. 3f). We apply a least-restrictive safety filter where the safety control is activated if the value function predicted at the current state falls below a specified threshold; otherwise, the nominal controller is applied [41]. This threshold is set to the δ value corresponding to $\epsilon = 10^{-3}$. A comparison of the trajectories resulting from the learned solution using the proposed method and the model-based approach is shown in Figure 3f. We do not show other methods because their BRT is too conservative and takes the robot out of the operating domain as a result. The filtered trajectory with our method successfully identifies a gap to traverse through, while the model-based method is overly conservative and detours to reach the goal, resulting in sub-optimal performance.

VI. Discussion and Future Work

We propose a framework to compute BRTs and safety controllers for black-box dynamical systems. Our approach can be used with existing reachability toolboxes to solve the HJB-VI for any general black-box system. We further propose a time-efficient variant (Ham-CA), which gives accurate solutions for control-affine systems and conservative BRT approximations for general dynamics. Through various case studies, we demonstrate the effectiveness of the proposed approach in recovering safe regions for black-box systems.

While the current work can be seamlessly integrated with learning-based level-set methods, its capability to solve HJB-VI is inherently constrained by the former. Therefore, developing more sophisticated and robust learning techniques would facilitate the proposed method to encompass higher-dimensional and more intricate reachability problems for black-box systems. In addition, the input size of Ham-NN, being double the system dimension, poses substantial challenges when dealing with extremely high-dimensional problems. Mitigating this limitation is another crucial direction for future research.

REFERENCES

[1] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2242–2253.

[2] M. Althoff, O. Stursberg, and M. Buss, "Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes," *Nonlinear analysis: hybrid systems*, vol. 4, no. 2, pp. 233–249, 2010.

[3] B. Schürmann and M. Althoff, "Guaranteeing constraints of disturbed nonlinear systems using set-based optimal control in generator space," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11 515–11 522, 2017.

[4] A. Majumdar, R. Vasudevan, M. Tobenkin, and R. Tedrake, *Convex optimization of nonlinear feedback controllers via occupation measures*, 2014, vol. 33, no. 9, pp. 1209–1230.

[5] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

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

[7] 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.

[8] 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, 2005.

[9] J. Lygeros, "On reachability and minimum cost optimal control," *Automatica*, vol. 40, no. 6, pp. 917–927, 2004.

[10] J. F. Fisac, M. Chen, C. J. Tomlin, and S. S. Sastry, "Reach-avoid problems with time-varying dynamics, targets and constraints," in *Proceedings of the 18th international conference on hybrid systems: computation and control*, 2015, pp. 11–20.

[11] W. Liao, T. Liang, P. Xiong, C. Wang, A. Song, and P. X. Liu, "An improved level set method for reachability problems in differential games," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024.

[12] I. Mitchell, "A toolbox of level set methods," *http://www. cs. ubc. ca/mitchell/ToolboxLS/toolboxLS.pdf*, 2004.

[13] E. Schmerling, "hj_reachability: Hamilton-Jacobi reachability analysis in JAX," *https://github.com/StanfordASL/hj$_r$eachability*, 2021.

[14] J. Darbon, G. P. Langlois, and T. Meng, "Overcoming the curse of dimensionality for some hamilton–jacobi partial differential equations via neural network architectures," *Research in the Mathematical Sciences*, vol. 7, pp. 1–50, 2020.

[15] K. Niarchos and J. Lygeros, "A neural approximation to continuous time reachability computations," in *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006, pp. 6313–6318.

[16] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive sampling: Real-time behaviour synthesis with mujoco," *arXiv preprint arXiv:2212.00541*, 2022.

[17] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[18] M. Fliess, C. Join, and H. Sira-Ramirez, "Complex continuous nonlinear systems: their black box identification and their control," *IFAC Proceedings Volumes*, vol. 39, no. 1, pp. 416–421, 2006.

[19] S. Herbert, J. J. Choi, S. Sanjeev, M. Gibson, K. Sreenath, and C. J. Tomlin, "Scalable learning of safety guarantees for autonomous systems using hamilton-jacobi reachability," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5914–5920.

[20] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2018.

[21] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with gaussian processes," in *53rd IEEE conference on decision and control*. IEEE, 2014, pp. 1424–1431.

[22] H. Wang, J. Borquez, and S. Bansal, "Providing safety assurances for systems with unknown dynamics," *IEEE Control Systems Letters*, 2024.

[23] X. Chen and E. Hazan, "Black-box control for linear dynamical systems," in *Proceedings of Thirty Fourth Conference on Learning Theory*, ser. Proceedings of Machine Learning Research, M. Belkin and S. Kpotufe, Eds., vol. 134. PMLR, 15–19 Aug 2021, pp. 1114–1143.

[24] F. Djeumou, A. P. Vinod, E. Goubault, S. Putot, and U. Topcu, "On-the-fly control of unknown smooth systems from limited data," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3656–3663.

[25] F. Djeumou, A. Zutshi, and U. Topcu, "On-the-fly, data-driven reachability analysis and control of unknown systems: an f-16 aircraft case study," in *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, 2021, pp. 1–2.

[26] J. Kapinski, J. V. Deshmukh, S. Sankaranarayanan, and N. Aréchiga, "Simulation-guided lyapunov analysis for hybrid dynamical systems," in *Proceedings of the 17th international conference on Hybrid systems: computation and control*, 2014, pp. 133–142.

[27] U. Topcu, A. Packard, and P. Seiler, "Local stability analysis using simulations and sum-of-squares programming," *Automatica*, vol. 44, no. 10, pp. 2669–2675, 2008.

[28] H. Ravanbakhsh and S. Sankaranarayanan, "Learning control lyapunov functions from counterexamples and demonstrations," *Autonomous Robots*, vol. 43, no. 2, pp. 275–307, 2019.

[29] L. Liebenwein, C. Baykal, I. Gilitschenski, S. Karaman, and D. Rus, "Sampling-based approximation algorithms for reachability analysis with provable guarantees," in *RSS*, 2018.

[30] T. Sutter, A. Kamoutsi, P. M. Esfahani, and J. Lygeros, "Data-driven approximate dynamic programming: A linear programming approach," in *IEEE Conference on Decision and Control*, 2017, pp. 5174–5179.

[31] A. Devonport and M. Arcak, "Estimating reachable sets with scenario optimization," 2020.

[32] S. Ghosh, S. Bansal, A. Sangiovanni-Vincentelli, S. A. Seshia, and C. Tomlin, "A new simulation metric to determine safe environments and controllers for systems with unknown dynamics," in *International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 185–196.

[33] T. Lew and M. Pavone, "Sampling-based reachability analysis: A random set theory approach with adversarial sampling," *arXiv preprint arXiv:2008.10180*, 2020.

[34] J. F. Fisac, N. F. Lugovoy, V. Rubies-Royo, S. Ghosh, and C. J. Tomlin, "Bridging hamilton-jacobi safety analysis and reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8550–8556.

[35] K.-C. Hsu, V. Rubies-Royo, C. J. Tomlin, and J. F. Fisac, "Safety and liveness guarantees through reach-avoid reinforcement learning," *arXiv preprint arXiv:2112.12288*, 2021.

[36] K.-C. Hsu, D. P. Nguyen, and J. F. Fisac, "Isaacs: Iterative soft adversarial actor-critic for safety," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 90–103.

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

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

[39] K. Leung, E. Schmerling, M. Zhang, M. Chen, J. Talbot, J. C. Gerdes, and M. Pavone, "On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions," *The International Journal of Robotics Research*, vol. 39, no. 10-11, pp. 1326–1345, 2020.

[40] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid

locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.
[41] J. Borquez, K. Chakraborty, H. Wang, and S. Bansal, "On safety and liveness filtering using hamilton-jacobi reachability analysis," *IEEE Transactions on Robotics*, 2024.