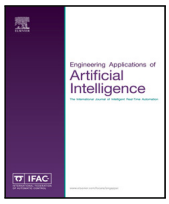
Engineering Applications of Artificial Intelligence 181 (2026) 115699

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Research paper

Robust reachability within deep reinforcement learning framework ☆

Satya Marthi *, Mayank S. Jha, Didier Theilliol, Jean-Christophe Ponsart

Universite de Lorraine, CRAN, UMR CNRS 7039, Vandoeuvre-les-Nancy, 54509, France

ARTICLE INFO

Keywords:
Reinforcement learning
Hamilton–Jacobi reachability
Deep deterministic policy gradient
Deep neural networks

ABSTRACT

We propose a novel deep reinforcement learning based approach to solve Hamilton–Jacobi reachability (HJ-R) problem for nonlinear control-affine systems with external disturbance. By recasting reachability as an optimal control problem, we solve it within a Deep Deterministic Policy Gradient (DDPG) framework: the critic learns the HJ-R value function and the actor synthesizes the optimal policy. We propose a Telescopic Incentive Reward Function (TIRF) that makes learning process efficient, promotes finite-time convergence to the target set, and reduces control oscillations near constraint boundaries. Disturbance is incorporated through agent-environment interaction, enabling robust optimal policy learning without an explicit disturbance model. The proposed approach fares well against classical grid-based dynamic programming approach and mitigates the curse of dimensionality through deep neural approximation, yielding scalability to higher-dimensional states. Numerical studies on a 2D inverted-pendulum benchmark and a 3D Dubins vehicle benchmark demonstrate target reach from diverse initial conditions, smooth control inputs relative to dynamic-programming baselines, and encouraging computational scaling on a more complex nonlinear system. These results position the proposed Robust Reachability-DDPG (RR-DDPG) framework as an efficient and robust alternative for HJ-R controller synthesis in continuous state–action spaces.

1. Introduction

The Hamilton–Jacobi reachability (HJ-R) framework has become an important research domain in the context of designing safe-controllers for autonomous systems particularly for safety-critical systems (Mitchell et al., 2005; Bansal et al., 2017). The emerging research in HJ-R domain targets providing a holistic framework to solve the class of reach-avoid problems (Margellos and Lygeros, 2011). Typically, a reach-avoid problem (also liveness-safety problem) is formulated as one where the controller tries to reach the target in presence of an external disturbance, where the disturbance often acts as an adversary and opposes the controller action (Bansal et al., 2017). Historically, various approaches have been proposed to solve the HJ-R problem starting with the use of dynamic programming (DP) based approaches (Chen and Tomlin, 2018) where the HJ-R is solved as an optimal control problem. Subsequently, the field of differential games (Bansal et al., 2017) that provides a framework to solve the HJ- reachability problem by treating them as a two-player (zero-sum) game by assigning the controller as a *reach player* and disturbance as *avoid player*, which are in conflict with one another (Bansal et al., 2017). It must be noted that the success of HJ-R framework is its capability to provide formal verification to the nonlinear systems with state and control constraints under adversarial disturbances (Chen and Tomlin, 2018). Another notable benefit of this framework is that the controller is designed for the worse-case scenario by considering all possible control inputs and trajectories. Hence, the recomputation of the safety-controller is avoided for any change in state or disturbance (Borquez et al., 2024). However, it must be noted that the optimal control policy (controller) synthesized by solving the reachability problem does not provide any performance guarantees (Borquez et al., 2024). Thus, there is a need to develop approaches that solve HJ-R problem whilst guaranteeing system performance.

Further, in the context of HJ-R problem, the traditional approaches typically solve the *backward reachable tube* (BRT), also called the *capture basin* (Mitchell et al., 2005; Aubin et al., 2011). Generally, the BRT for a dynamical system gives the set of initial states from which the system will eventually reach the target set despite the external (worst-case) disturbance (Mitchell et al., 2005; Aubin, 2001). The solution to the BRT can be obtained for any dynamical system by obtaining the viscosity solution to the Hamilton–Jacobi–Isaacs Variational Inequality (HJI-VI), which results in enforcing the reach and/or avoid sets as terminal constraints to the Hamilton–Jacobi–Bellman (HJB) partial differential equation (PDE) (Fisac et al., 2015; Bansal et al., 2017).

☆ This article is part of a Special issue entitled: 'TC 2.4 Optimal Control' published in Engineering Applications of Artificial Intelligence.

* Corresponding author.
E-mail addresses: satya.marthi@univ-lorraine.fr (S. Marthi), mayank.jha@univ-lorraine.fr (M.S. Jha), didier.theilliol@univ-lorraine.fr (D. Theilliol), jean-christophe.ponsart@univ-lorraine.fr (J.-C. Ponsart).

https://doi.org/10.1016/j.engappai.2026.115699
Received 2 November 2025; Received in revised form 29 May 2026; Accepted 12 July 2026

The HJI-VI solves a value function with minimal control effort under worse-case disturbance while minimizing the distance to the target set. The knowledge of the value function encodes the information of the set of initial states within the state-space through which a safe controller can be synthesized (Bansal et al., 2017). It is interesting to note that the safe controller computation is heavily reliant on the accuracy of the value function (Borquez et al., 2024).

However, most of the above mentioned approaches suffer with *curse of dimensionality*, notably the DP based approaches where the solutions also depend on the grid resolution (Chen and Tomlin, 2018). To alleviate problems arising from the curse of dimensionality and scale the HJ-R solutions to higher dimensional systems, recently, reinforcement learning (RL) based solutions have been proposed that utilize the power of neural networks (NN) to act as universal function approximators (So et al., 2024). The review article (Bansal et al., 2017) and the works in Ganai (2024) provide a detailed summary on the use of RL techniques to solve the HJ-R problem and propose a new methodology called *Hamilton–Jacobi reachability estimation*. However these works consider the system dynamics without external disturbance. In the works (Kokolakis et al., 2023) the HJ-R problem is cast as an Mayer optimal control problem which provides convergence guarantees does not consider external disturbance. However, Bansal and Tomlin (2021) proposed deep neural networks (DNN) to approximate the HJI-VI PDE and demonstrated the scalability of HJ-R to higher dimensional systems while also considering the external disturbance. Therein, the authors have noted that in presence of disturbance, the training time is very high (16 h for a *Air-3D*) and that the latter along with memory requirement does not scale with the dimension of the state-space, but rather depends on the signal complexity. Additionally, the work (Fisac et al., 2018) solve the safe reinforcement learning problem using HJ-R framework to synthesize safe controllers but do not treat the robustness in a formal manner.

Recent RL-control research has increasingly focused on formal safety or stability guarantees, including Lyapunov-stable neural control, safe reinforcement learning for nonlinear systems with convergence guarantees, MPC-inspired verifiable model-free control, backup-control-barrier-function-based safe exploration, and dissipativity-based neural control with stability and optimality certificates (Yang et al., 2024; Suttle et al., 2024; Lu et al., 2024; Rabiee and Safari, 2025; Wang et al., 2025). These works are well relevant from a safe-learning perspective; however, they do not directly address the finite-horizon Hamilton–Jacobi reachability problem under bounded worst-case disturbances considered in this paper. We cite them here to position the present contribution relative to the broader literature on guaranteed learning-based control.

Beyond continuous-state control, reinforcement learning has also been applied to discrete-state networked systems. Particularly, in the Boolean networks (BN) literature, the semi-tensor product (STP) of matrices provides an algebraic frameworks for synchronization, stabilization and detectability of probabilistic Boolean networks. In the recent work (Zhang et al., 2025b), STP-based analysis is combined with Q-learning to solve node-set synchronization problems of general Boolean networks. Additionally, in the work (Zhang et al., 2025a) an optimal-flip-based segmented RL is presented for detectability analysis of probabilistic Boolean networks.

Finally, Yang et al. (2025) presents a deep RL methodology that has been extended to complex decision-optimization tasks where state-split-flow deep Q-learning networks with threat-degree indices address combinatorial allocation under adversarial conditions.

Despite the above progress, four scientific gaps remain. First, classical HJ reachability solvers provide accurate value-function solutions but suffer from the curse of dimensionality. Second, several RL-based HJ formulations either neglect external disturbances or focus primarily on value-function estimation, without simultaneously learning a directly deployable feedback controller for the robust reach problem. Third, deep reachability methods that account for disturbance can incur substantial training time and memory requirements. Fourth, the finite-horizon HJ-R objective depends on the running minimum distance to the target along the trajectory, which is not directly compatible with standard actor–critic reward formulations.

To address these gaps, this paper reformulates the robust HJ-R problem within a DDPG framework (Lillicrap et al., 2015; Liu et al., 2025; Ahmed et al., 2023; Andrei et al., 2023). The key idea is to encode the finite-horizon reach objective through a telescopic reward that preserves the running-minimum structure of the HJ-R cost, while augmenting it with a dense incentive term that improves exploration and learning efficiency. Worst-case disturbance is handled through agent-environment interaction over a bounded adversarial disturbance set, and the actor/critic networks are trained jointly so that the critic approximates the reachability value function and the actor yields a directly deployable feedback controller.

The contributions of this work are as follows:

- We propose a robust DDPG-based formulation for finite-horizon HJ reachability under bounded worst-case disturbance, termed Robust Reach DDPG (RR-DDPG), in which the disturbance maximization is handled through environment interaction and the actor learns the minimizing feedback law.
- We propose a Telescopic Incentive Reward Function (TIRF) that combines a telescopic reward preserving the HJ-R objective with a dense incentive term that mitigates reward sparsity.
- We provide an simulation study using 2D inverted-pendulum and 3D Dubins models under disturbance as examples to assess the efficacy of the proposed approach and provide computational-time comparison with representative HJ solution pipelines using two established approaches (DeepReach and HelperOC).

It should be noted that unlike the works that have focused on convergence/stability properties within the framework of DDPG such as Kim et al. (2020), the focus of this work is not on establishing rigorous proofs on the convergence of value functions and policy to their optimality, but leverage the benefits of DDPG to solve the reachability problem and address existing problems. Such rigorous proofs on stability in Lyapunov sense as well as convergence to optimality is typically developed in the domain of Adaptive Dynamic Programming (ADP) /RL, which are "control-theory centric" and address these problems in the context of Hamilton–Jacobi–Bellman type value functions. We refer the readers to following works for the same: Heuristic dynamic programming based approaches for optimal tracking (Jha et al., 2023), off-policy based policy iteration leading to guaranteed stability, optimality and safety (Kanso et al., 2025; Jha and Kiumarsi, 2025; Kanso et al., 2024, 2026; Rutschke et al., 2025) as well as on-policy based policy iteration (Marthi et al., 2025).

Notations

The Euclidean norm is denoted by $\|\cdot\|$. The Frobenius norm of a matrix $A \in \mathbb{R}^{m\times n}$ is denoted by $\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called *Lipschitz continuous* on a compact set $\mathcal{X} \subset \mathbb{R}^n$ if there exists a constant $L \in \mathbb{R}^+$ (set of positive real numbers) such that $\|f(x_2) - f(x_1)\| \leq L\|x_2 - x_1\|, \forall x_2, x_1 \in \mathcal{X}$. A continuous function $\alpha(\cdot) : [0, a) \to [0, \infty)$ is a class-$\mathcal{K}$ function for $a > 0$ if it is strictly increasing and $\alpha(0) = 0$ and if $\alpha(\cdot)$ is defined over the entire real-line, then the function α belongs to an extended class $\mathcal{K}_\infty$. $\eta_{x,t}^{u,d}$ represents the state from x at a particular time t under the control signal $u(\cdot)$ and disturbance signal $d(\cdot)$. $\mathcal{T}$ is the target set which encodes the goal and $\mathcal{R}$ represents the reach set of the system.

2. Background and problem formulation

2.1. System dynamics

Consider the following continuous-time nonlinear dynamics affine in control input and disturbance

$$\dot{x} = f(x) + g_u(x)u + g_d(x)d, \quad x(t_0) = x_0 \quad t \geq t_0 \tag{1}$$

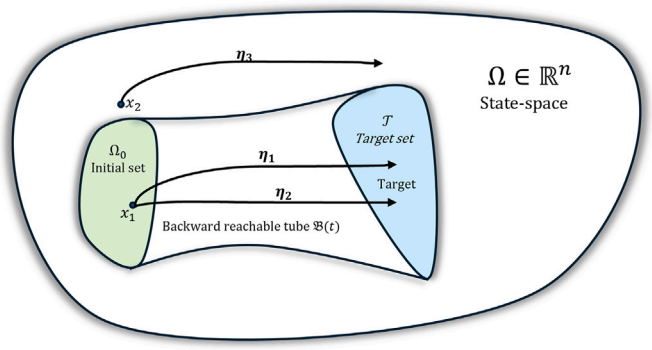


Fig. 1. BRT and the initial set that contains all the states that reach the target set under worse-case disturbance (η_1, η_2 and η_3).

where, for every $t \geq t_0$; $x(t) \in \mathcal{X} \subset \mathbb{R}^n$ is the state vector, $u(t) \in \mathcal{U} \subset \mathbb{R}^m$ is the control input vector and $d(t) \in \mathcal{D} \subset \mathbb{R}^p$ is the disturbance to the system. The disturbance considered is bounded within $\|d(t)\|_2 \in \rho$. We assume that the control input and disturbance are continuous and bounded. The drift dynamics $f(x) \in \mathbb{R}^n$, control input $g_u(x) \in \mathbb{R}^{n\times m}$ and disturbance $g_d(x) \in \mathbb{R}^{n\times p}$ are Lipschitz continuous. We consider that $g_u(x)$ and $g_d(x)$ satisfies $\|g_u(x)\|_{\mathrm{F}} \leq g_U$ and $\|g_d(x)\|_{\mathrm{F}} \leq g_D$. We also assume that the matrices $g_u(x)$ and $g_d(x)$ are invertible such that $g_u^{-1}(x)g_u(x) = \mathbb{I} \in \mathbb{R}^{m\times m}$ (Huang and Liu, 2014). The system trajectory is denoted by $\eta_{x,t}^{u,d}(t)$ corresponding to (1) for a given $u(\cdot)$, $d(\cdot)$ signals at time t.

Remark 2.1. These assumptions should be interpreted as local regularity conditions on a compact operating region rather than as global properties of the plant. In practice, boundedness and Lipschitz continuity are standard for smooth mechanical, robotic, and aerospace systems once the state and input are restricted to physically meaningful operating ranges. The control-affine disturbance model captures matched disturbances or bounded exogenous inputs entering through known channels. When these conditions do not hold, for example, with non-affine disturbance channels, singular input directions, or non-smooth dynamics; the present analysis becomes local and the disturbance maximization may need to be handled numerically rather than through the simplified boundary argument.

Further, we assume that for a given initial condition, there exists an unique and continuous trajectory $\eta_{x,t}^{u,d}$ given input u and disturbance signals are piece-wise continuous with time (Borquez et al., 2024). We also assume that there exists a set of admissible controls ($\mathcal{U}$) for (1). The safe controllers synthesized are $\mathcal{U}_s \subset \mathcal{U}$. The origin is assumed to be the equilibrium point and the system is fully *controllable* and fully *observable* and the system dynamics are completely known.

Definition 1 (*Target Set Evans, 2010*)**.** A set $\mathcal{T} \subset \mathcal{X}$ is called a user-defined target set for the sub-zero level set of a continuous differentiable, bounded function $\ell(x)$ given by:

$$\mathcal{T} := \{x(t) | \ell(x) \leq 0\}. \tag{2}$$

Definition 2 (*Reach Set Evans, 2010*)**.** A set $\mathcal{R} \subset \mathcal{X}$ is a reach set of a target set of the system (1) for a given initial condition x_0 and a time t_0 such that:

$$\mathcal{R}(t_0; t, \mathcal{T}) = \left\{ x_0 \middle| \max_{d(\cdot)\in\mathcal{D}} \min_{u(\cdot)\in\mathcal{U}} \ell(\eta(t; t_0, x_0, u, d)) \leq 0 \right\}. \tag{3}$$

Definition 3 (*Admissible Control Policy Kokolakis et al., 2023*)**.** A control policy $\pi(x) : \mathbb{R}^m \to \mathcal{U}$ is said to be admissible with respect to the target set $\mathcal{T}$ if the closed-loop trajectories for an initial state are unique, well-posed and bounded while satisfying the input constraints.

2.2. Reach problem using hamilton–Jacobi reachability

The primary objective of this work is to synthesize controllers $\pi(x) : [0, T] \mathcal{X} \to \mathcal{U}$ that ensure that the system reaches a desired set of states $\mathcal{T}$ (see definition 1) in finite-time conditions (Fisac et al., 2015). Here the set $\mathcal{T}$ could represent the goal of the system (for example, the hovering destination of a drone in fire fighting situations). The objective of the system is to reach the target set in finite time under external disturbance and control input saturation (Borquez et al., 2024).

In the traditional HJ-R framework, the reach problem is cast as an optimal control problem. The optimization is performed to minimize the distance between the current system state and the target set over the finite horizon. For the reach case (also called the liveness problem) the optimization is defined as:

$$\max_{d(\cdot)} \min_{u(\cdot)} \min_{\tau\in[t,T]} \ell(\eta_{x,t}^{u,d}), \tag{4}$$

$$\text{s.t.} \quad \dot{x} = f(x) + g_u(x)u + g_d(x)d, \tag{5}$$

where the minimum cost over time is defined by,

$$J(x, t, u(\cdot), d(\cdot)) = \min_{\tau\in[t,T]} \ell(\eta_{x,t}^{u,d}). \tag{6}$$

Remark 2.2. The cost function computes the minimal distance along the entire trajectory $\eta_{x,t}^{u,d}$ to the target set given by the sub-zero level set of the function $\ell(x)$.

The minimum distance under the optimal control and external disturbance is considered as a value function given by:

$$V(x, t) = \max_{d(\cdot)} \min_{u(\cdot)} J(x, t, u(\cdot), d(\cdot)). \tag{7}$$

The viscosity solution of the Hamilton–Jacobi–Issacs Variational inequality (HJI-VI) is the value function (7) (Borquez et al., 2024).

$$\max \left\{ \frac{\partial V(x,t)}{\partial t} + H\left(x, t, \nabla V(x,t)\right), \ell(x) - V(x,t) \right\} = 0 \quad \text{for } t \in [0, T] \text{ and } V(x, T) = \ell(x). \tag{8}$$

The value function obtained by solving (8) can be used to obtain the Backward Reachable Set (BRS) for the system (1). BRS gives the initial set of states from which despite the worse-case disturbance there exists a control input such that the system can reach the target in time $(t-T)$.

BRS is computed from the sub-zero level set of the value function solution given by Borquez et al. (2024):

$$\mathcal{B}(t) = \{x : V(x, t) \leq 0\}. \tag{9}$$

Thus, computing the BRS by taking the sub-zero level set of the value function (7), gives the set of all possible initial states subject to various input and disturbance signals to drive the system to the target. Fig. 1 presents an illustration of the sub-zero level set (Ω_0) obtained by solving (9).

The HJ reachability framework also enables the computation of optimal controllers and optimal disturbance that solve the reach problem (8). For a given state $x(t)$ the optimal control policy can be computed by:

$$\pi^\star(x, t) = \arg\min_{u\in\mathcal{U}} \max_{d\in\mathcal{D}} \nabla V(x, t) \cdot \left(f(x) + g_u(x)u + g_d(x)d\right), \tag{10}$$

and the optimal disturbance can be obtained using:

$$d^\star(x, t) = \min_{u\in\mathcal{U}} \arg\max_{d\in\mathcal{D}} \nabla V(x, t) \cdot \left(f(x) + g_u(x)u + g_d(x)d\right). \tag{11}$$

In essence, the optimal control policy steers the system towards the target whereas the optimal disturbance drives the system away from the target. The work (Borquez et al., 2024) presents a comprehensive discussion on the controller selection to solve the reach problem. This outcome follows the solution of (10), which does not consider any performance related constraints. In addition, several other studies

(Bui et al., 2021; Ganai et al., 2023) provide numerous approaches to solve both the reach problem and avoid problem and analyze the computational requirements to solve the reach-avoid problem (Chen et al., 2016). However, most of the works simplify the problem by neglecting the disturbance to the system.

In contrast, this works proposes a novel approach to address the reach problem while explicitly considering the disturbance with lower computational cost by leveraging the DDPG framework.

2.3. Deep deterministic policy gradient (DDPG)

Introduced in Lillicrap et al. (2015), DDPG is an robust off-policy reinforcement learning algorithm that also solves the optimal control problems with continuous state-space (Rego and de Araujo, 2022). The DDPG framework uses a Markov decision process (MDP). The MDP is characterized by the tuple $\{\mathscr{X}, \mathscr{A}, \mathscr{R}, \mathscr{D}, \gamma\}$ where $\mathscr{X}$ is the state-space, $\mathscr{A}$ being the action-space and $\mathscr{R}$ as reward with the assumption that the rewards are positive $r \geq 0$ and $\mathscr{D}$ is the disturbance and thus implicitly includes the transition matrix (system dynamics) with $0 \leq \gamma \leq 1$ being the discount factor.

Remark 2.3. Although the DDPG algorithm is presented for system with continuous space systems, during the implementation the system dynamics must be discretized (Lillicrap et al., 2015). For notational simplicity we denote the reward function with $r(x_k)$ at time-step k. We also consider the MDP with transition probability one (deterministic dynamics) and omit the details on discretization.

DDPG adopts the actor–critic structure to solve the control problems by using the Q-value function also called the *state–action value function* $Q(x,u)$ using the recursive Bellman equation for a finite horizon problem is given by:

$$Q^{\pi}(x_k, u_k) = \mathbb{E}_{\tau\sim\pi,\mathscr{Y}}\left[\sum_{k=0}^{T} \gamma^k r(x_k, u_k)\right]. \tag{12}$$

where $r(x,u)$ The optimal control policy can be computed by

$$u^{\star}(x) = \arg\min_{u(\cdot)} Q(x_k, u_k). \tag{13}$$

In DDPG approaches, the learning is driven by the choice of reward function which must be engineered with precision for successful completion of a task. The algorithm also employees the use of target networks and replay buffer to stabilize the learning under the nonexistence of convergence guarantees (Lillicrap et al., 2015).

2.4. Problem statement

For the system (1) with a given terminal cost condition $\phi(x)$, the value function to solve the finite horizon optimal control problem becomes:

$$V(x,t) = \min_{u(\cdot)} \max_{d(\cdot)} \int_t^T L(x(s), u(s, d(s)))\, ds + \phi(x(T)). \tag{14}$$

The HJI takes the form:

$$\frac{\partial V(x,t)}{\partial t} + \min_{u\in\mathscr{U}} \max_{d\in\mathscr{D}} \{\nabla V(x,t)^{\top}(f(x) + g_u(x)u + g_d(x)d) + L(x,u,d)\} = 0. \tag{15}$$

where $L(\cdot)$ is the running cost and the target set (see definition 1) becomes the terminal cost condition. Then, the goal is to employ the DDPG algorithm to approximate the value function (15) using the critic network and the corresponding optimal feedback policy (optimal control law) (10) using the actor network that is robust against disturbance.

To this end, we propose a novel reward signal that incorporates worse-case disturbance considered through agent-environment interactions, leading to a novel DDPG-based robust-reachability algorithm.

3. Solving the value function using DDPG

To solve the HJ-R optimal control problem using DDPG, it is necessary to define following key properties of the DDPG learning framework like, the environment, the reward function and the loss function that drives the learning procedure under finite time condition.

3.1. Designing the reward function

For the system (1), the signed distance function to the target set $\mathscr{T}$ is minimized with the terminal cost condition $\ell(x)$ which is encoded using the value function. Typically, the Q-value function computes the cumulative reward over an finite horizon. However, in our work, the optimality must be achieved over a finite time horizon. To achieve finite-time solutions to (15), in this work, the following reward function is proposed:

$$r(x_k) = J_k - J_{k+1} \tag{16}$$

where J_k is the cost (6) computed at time-step k. The reward function (16) is now introduced in value function (12) with $\gamma = 1$ to give:

$$\begin{aligned} V(x_k) &= \sum_{k=0}^{T-1} r(x_k) = \sum_{k=0}^{T-1} \left(J_k - J_{k+1}\right), \\ V(x_k) &= J_0 - J_T, \end{aligned} \tag{17}$$

where J_0 is the cost at initial condition (x_0) and J_T is the terminal, finite horizon cost function. Given the finite-time nature of the value function, the reward function (16) becomes a telescopic reward function. The telescopic nature of the reward function becomes crucial to solve the HJ-R problem using DDPG.

Lemma 1. *For the system (1), the telescoping reward function (16) is considered within the value function (12) solves the HJ-R optimal control problem.*

Proof. Consider a function $c(\cdot) : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ that computes the instantaneous distance to the target set at each time-step k which is also the instantaneous value of the cost function (6):

$$c(x_k) = \max\{0, \|x_k - x_T\| - r_T\} \geq 0 \tag{18}$$

where x_T is the center of the target set and r_T represents the size of the target set. Using the function $c(\cdot)$ the telescoping reward to compute the minimum distance is given by:

$$J_{k+1} = \min\{J_k, c(x_{k+1})\} \quad (k = 0, \ldots T-1). \tag{19}$$

Now, the undiscounted (where $\gamma = 1$) cumulative reward function over a fixed time horizon T is:

$$\sum_{k=0}^{T-1} r(x_k) = \sum_{k=0}^{T-1} \left(J_k - J_{k+1}\right) = J_0 - J_T, \tag{20}$$

here, $J_0 = c(x_0)$ and J_T enforces the terminal condition. Therefore, under the condition that J_0 does not depend on the policy $\pi(x)$, (20) gives the same result as solving (6) thus concluding the proof. □

Remark 3.1. Although the cost function includes the computation of signed-distance to target set, the function $c(x)$ is an unsigned function that is convex and Lipschitz continuous. Furthermore, in Lemma 1 the initial cost $J_0 = c(x_0)$ is independent of the policy, maximizing the cumulative return is equivalent to minimizing $J_T = \min_{\tau\in[t,T]} \ell(\eta^{u,d}_{x,t})$, which is the HJ-R cost (6).

Using the aforementioned formulations, the telescopic reward function becomes:

$$r_t(x) = \begin{cases} \mathbf{w}, & \text{if } \ell(x_{k+1}) < J_k, \\ 0, & \text{otherwise}, \end{cases} \tag{21}$$

where $\mathbf{w}$ is a positive constant (hyper-parameter). At each time step, the telescopic reward function (21) is nonzero when the system trajectory moves close to the target but necessarily zero otherwise. Thus making the reward structure sparse. Learning algorithms like DDPG with sparse reward structures require high exploration noise or sometimes lead to suboptimal learning. As such, there is a need to mitigate the problem arising from the sparsity, whilst preserving the telescopic nature of the reward function (16) we propose an *Incentive function* that produces nonzero reward at each time-step.

Incentive function (IF)

To continuously guide learning with a dense reward, an incentive term is added to the sparse telescopic reward. For goal-reaching problems, three reward-design choices are common. A sparse terminal reward is faithful to the task objective but provides weak learning signal (Dewey, 2014). A dense distance-based reward improves exploration but can dominate or distort the original finite-horizon objective. Potential-based shaping preserves policy invariance under specific constructions, but its direct use does not by itself encode the running-minimum structure of the HJ-R cost. The proposed TIRF combines the advantages of sparse and dense shaping: the telescopic term preserves the HJ-R objective by rewarding only improvements in the running minimum cost, while the incentive term provides continuous step-wise guidance towards the target set.

To define the dense incentive term, we use the signed distance to the target (the instantaneous cost function (18) is reformulated):

$$\delta(x) = \|x - x_T\| - r_T, \qquad \begin{cases} \delta(x) < 0, & \text{if } x \in \mathscr{T}, \\ \delta(x) = 0, & \text{if } x \in \partial\mathscr{T}, \\ \delta(x) > 0, & \text{if } x \notin \mathscr{T}. \end{cases} \tag{22}$$

We then define the incentive term as

$$i(x) = \exp(-\alpha\, \delta(x)) - 1, \tag{23}$$

where $\alpha > 0$ controls the steepness of the shaping near the target boundary. With this definition, $i(x) > 0$ inside the target set, $i(x) = 0$ on the boundary, and $i(x) < 0$ outside the target set, thereby encouraging actions that move the state towards $\mathscr{T}$.

Augmenting the sparse telescopic reward with the incentive term yields the proposed Telescopic Incentive Reward Function (TIRF):

$$r(x_k) := \begin{cases} \mathbf{w} + \mathbf{w_C}\, i(x_k), & \text{if } \ell(x_{k+1}) < J_k, \\ \mathbf{w_C}\, i(x_k), & \text{otherwise}, \end{cases} \tag{24}$$

where $\mathbf{w} > 0$ sets the magnitude of the sparse bonus awarded when the running minimum cost improves, and $\mathbf{w_C} > 0$ scales the dense incentive term so that it guides exploration without overwhelming the telescopic objective.

It should be noted that incorporating IF within the telescopic reward function (16) offers two advantages. First, the function returns negative rewards when the agent is outside the target and returns positive rewards within the target set, thereby encouraging the actions towards the target set. Second, the IF also allows the function can be flexibly adjusted according to the boundaries of the target set. Fig. 2 illustrates the behavior of IF for a target set given by $-5 \le \delta(x) \le 5$ (see Fig. 3).

Remark 3.2. To learn the HJ-R value function using the DDPG learning framework, the use of telescoping reward function becomes critical to preserve the Markovian property that is critical in the learning paradigm. This originates from the original HJ-R cost function that computes the minimum distance over the entire trajectory which introduces a dependency over all the previous system states. Additionally, under the proposed TIRF (24) in DDPG algorithm recovers the approximate solution to the viscosity solution to the HJI-VI (8).

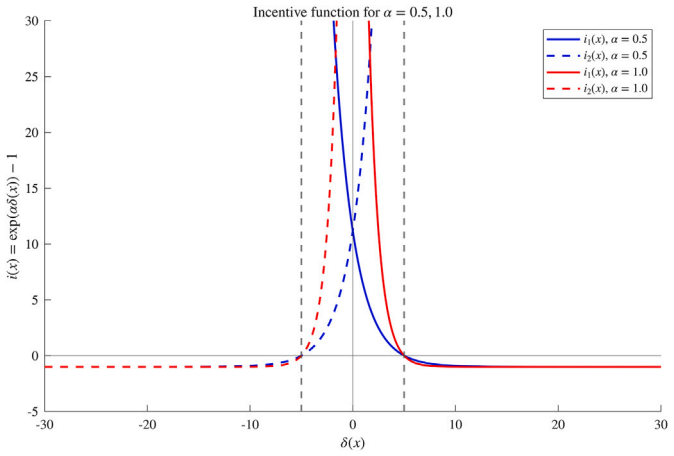


Fig. 2. Incentive function $i(x)$ for the target $-5 \le \delta(x) \le 5$.

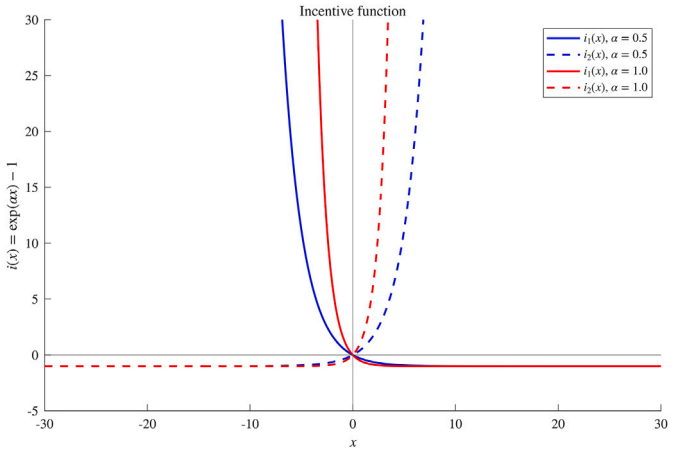


Fig. 3. Behavior of incentive function $i(x)$.

3.2. Environment

In order to leverage DDPG for learning optimal control policy, we consider the state space formulation of the system in (1) through a MDP that evolves (the transition is given by $\mathscr{A}, \mathscr{X}, \mathscr{D}$) and consider the discretized (in-time) equations in the following way:

$$x_{k+1} = f(x_k) + g_u(x_k)u_k + g_d(x_k)d_k, \quad \|d_k\| \in \rho. \tag{25}$$

Proposition 3.1. *For the system (25), using Definition 3, the functions $f(x)$, $g_u(x)$ and $g_d(x)$ are bounded and the instantaneous cost function (18) is convex in $\mathscr{D}$.*

Proof. The control input and disturbance matrices are bounded under the Frobenius norm $\|g_u(x)\|_F < g_U$ and $\|g_d(x)\|_F < g_d$. Given the existence of set of admissible control policies to the system (25), the system dynamics are bounded. The reader is advised to see Lemma-1 in Huang and Liu (2014) for detailed proof. Thus, for the system (25) for a given state and control policy (x_r, u_r) the dynamics can be written as follows:

$$x(d) = \underbrace{f(x_r) + g_u(x_r)u_r}_{\text{bounded function}(F(x,u))} + \underbrace{g_d(x_r)}_{(G(x))}\, d. \tag{26}$$

The (26) can be reduced to into a dynamics that is affine in disturbance given by:

$$x(d) = F(x,u) + G(x)\ d. \tag{27}$$

The cost function according to the new affine dynamics (27) becomes:

$$C(d) = c(x(d)) \tag{28}$$

where $c(\cdot)$ is the cost function (18). This renders the $C(d)$ a convex function on $\mathscr{D}$. The instantaneous cost function (18) computes the unsigned distance (Hinge version) which gives a convex function. Thus, the system dynamics are bounded under the stated assumptions and cost function is convex which concludes the proof. □

Lemma 2. *For the system (25), for a fixed $(x_k, u_k) \in \Omega$, a worst-case disturbance can be attained at a boundary point of the disturbance set $\mathscr{D}$.*

Proof. For a fixed $(x_k, u_k) \in \Omega$, the disturbance-affine dynamics (27) can be written as

$$x(d) = F(x_k,u_k) + G(x_k)d, \qquad d \in \mathscr{D} := \{d \in \mathbb{R}^p : \|d\|_2 \le \rho\}. \tag{29}$$

By Proposition 3.1, the instantaneous cost $c(\cdot)$ is convex. Since $x(d)$ is affine in d, the composite function

$$C(d) := c\big(F(x_k,u_k) + G(x_k)d\big) \tag{30}$$

is convex on the compact and convex set $\mathscr{D}$. A convex function over a compact convex set attains its maximum at an extreme point of that set. For the Euclidean ball $\mathscr{D}$, every extreme point lies on the boundary $\|d\|_2 = \rho$. Hence, a worst-case disturbance can be chosen on the boundary of $\mathscr{D}$. □

Furthermore, for the hinge-distance cost (18) under the Euclidean norm, one has the Lipschitz bound

$$c(F(x,u) + G(x)d) \le c(F(x,u)) + L_c\|G(x)d\|_2, \tag{31}$$

with $L_c = 1$. This relation quantifies the worst-case growth induced by the disturbance magnitude. We emphasize, however, that $L_c = 1$ is specific to the cost (18) under the Euclidean norm and is not the essential reason for the boundary-attainment argument. This argument applies directly to disturbance-affine systems with compact convex disturbance sets and convex state-cost surrogates such as (18). For systems with non-affine dependence on d, nonconvex disturbance sets, or higher-order nonlinear disturbance channels, the maximizer need not lie on the boundary; in such cases, the inner maximization over d must be solved explicitly, e.g., by numerical optimization or an adversarial disturbance policy.

Using Lemma 2, the search for the worst-case disturbance in (11) can be restricted from the full compact set $\mathscr{D}$ to its boundary, thereby reducing the computational burden of the inner maximization.

3.3. Finite-horizon computation

To solve the reach problem within a finite-horizon we employ two methodologies, one is by the use of telescopic reward (20) within the Q-value function and the other is by imposing the final value function $(V_T(x) = 0)$ by fixing the episodic time for each epoch as T.

Lemma 3. *For the system (25), using Lemmas 1 and 2 with telescoping reward and environment with disturbance for a fixed episode length T during the implementation, enforces the finite-time condition through the terminal value function condition that solves the HJ-R finite horizon optimal control reach problem.*

Proof. Using the cumulative minimum distance reward function

$$V_k^{DDPG} = \min_{\pi}\max_{\mathscr{D}} \sum_{k=0}^{T-1} r(x_k) = J_0 - J_T. \tag{32}$$

as $c(x_0)$ is fixed and equal to the episodic time during the learning procedure. Thus, the Bellman equation obtained becomes:

$$V_k^{\text{DDPG}}(x,J) = r_k + V_{k+1}^{\text{DDPG}}(x_{k+1}, J_{k+1}), \tag{33}$$

with terminal condition $V_T^{DDPG}(x,J) = 0$ which concludes the proof. □

The solution to HJ-R value function (15) is therefore equivalent to solution obtained by solving the (33). For the rest of the paper the $V_k^{\text{DDPG}}(x,t)$ is represented by $Q^{\pi}(x_k,u_k)$.

3.4. Loss function

The value function (33) along with the control policy are approximated using a deep neural networks (DNNs). The target equation is given by:

$$Q^{\pi}(x_k,u_k) = \mathbb{E}_D[r(x_k) + Q^{\pi}(x_{k+1},u_{k+1})]. \tag{34}$$

In the DDPG algorithm, the critic network updates the (34) to solve the Bellman equation and the actor network approximates $u_k = \pi(x_k)$ that maximizes $Q^{\pi}(x_k,u_k)$.

To this end, the critic loss function using the temporal difference (TD) target is given by:

$$\mathcal{L}_{\text{critic}}(\phi) = \mathbb{E}_D[(y - Q_{\phi}(x_k,u_k))^2], \tag{35}$$

with the Bellman target y is:

$$y = r(x_k) + \gamma Q_{\phi}(x_{k+1},u_{k+1}), \tag{36}$$

where, the critic network is parameterized by ϕ.

Remark 3.3. We assume that the replay buffer $\mathcal{B}$ provides sufficiently rich data to learn useful approximations of the value function and the control policy. As in standard DDPG (see Lillicrap et al. (2015)), experience replay and target networks improve the numerical robustness of training; however, they do not provide a general convergence guarantee for the nonlinear continuous-control setting considered here. Therefore, in this work, stability is interpreted empirically through training diagnostics such as episodic reward, the initial critic estimate Q_0, critic loss, and the norm of the actor policy gradient. As such, we do not provide a formal convergence guarantees for RR-DDPG in this work. See the (last parts of) Introduction section for references that provide formal guarantees for off-policy and on-policy based Policy Iteration approaches.

Similarly, the actor loss function which is parameterized by θ is updated using the deterministic policy gradient (DPG) that becomes:

$$\mathcal{L}_{\text{actor}}(\theta) = \mathbb{E}_D[Q_{\phi}(x,\pi_{\theta}(x))]. \tag{37}$$

The episodes terminate after T time steps or if the target is reached thus solving the reach problem in finite-horizon. The earlier-defined components like the environment, the reward function and the loss function are introduced within the DDPG framework for the finite-time, robust computation of optimal control policy that solves the reachability problem. Therefore, using all these formulations the following theorem can be established.

Theorem 1. *For the control and disturbance affine system (1), the HJ reachability problem, which is formulated as an optimal control problem can be solved using the reward function (24) within the DDPG framework.*

Proof. By combining the proofs of Lemmas 1, 2, and 3 with the loss functions of the actor and critic networks, the necessary conditions are established to approximately solve the HJ-R optimal control problem within the DDPG framework.However, similar to the standard DDPG algorithm (Lillicrap et al., 2015), this formulation does not guarantee algorithmic convergence. Therefore, the stability claims made in this paper are empirical and are assessed through the training diagnostics reported in the simulation section. □

Remark 3.4. The existing methodologies such as HelperOC (Chen and Tomlin, 2018) and Deepreach (Bansal and Tomlin, 2021) solve the classical HJ-R value function (7) by solving a *min–max* optimization problem where the controller minimizes the cost while the disturbance maximizes the cost. However, in the proposed framework (Theorem 1), the min–max optimization is decoupled by maximizing the effect of disturbance through agent-environment interaction (see Lemma 2). Consequently, the DDPG actor network only needs to solve a single (minimization) problem while the critic network learns the value function associated with the resulting minimization problem. The function approximations using actor and critic network and the absence of formal convergence guarantees of DDPG algorithm (Lillicrap et al., 2015), the resulting solution is an approximate, numerical solution to the HJI-VI (8).

4. Robust-reach algorithm

Following Theorem 1, we propose the robust reach DDPG (RR-DDPG) algorithm that leverages the DDPG framework to solve the HJ-R optimal control problem (8) under worse-case disturbance.

Algorithm 1 Robust Reach DDPG (RR-DDPG)

Initialize actor network $\pi(x_k|\theta^\pi)$ and critic network $Q(x_k, u_k|\theta^Q)$
Initialize target networks $\theta^{\pi'} \leftarrow \theta^\pi$, $\theta^{Q'} \leftarrow \theta^Q$
Initialize replay buffer $\mathcal{B}$
for each episode **do**
  Initialize random process $\mathcal{N}$ for action exploration
  Receive initial state x_0
  for each step k **do**
    Select action $u_k = \pi(x_k|\theta^\pi) + \mathcal{N}_k$
    Execute u_k, observe reward r_k and next state x_{k+1}
    Store transition (x_k, u_k, r_k, x_{k+1}) in $\mathcal{B}$
    Sample minibatch of N transitions from $\mathcal{B}$
    Set $y_k = r(x_k) + \gamma Q_\phi(x_{k+1}, u_{k+1})$
    Update critic by minimizing loss:

$$L = \frac{1}{N}\sum_i (y_k - Q(x_k, u_k|\theta^Q))^2$$

    Update actor using policy gradient:

$$\nabla_{\theta^\pi} J \approx \frac{1}{N}\sum_i \nabla_u Q(x_k, u_k|\theta^Q)|_{x_k, u_k=\pi(x_k)} \nabla_{\theta^\pi}\pi(x_k|\theta^\pi)$$

    Update target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \quad \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

The algorithm 1 employs the reward (24) by interacting with the environment with disturbance. The resulting control policy obtained upon successful training ensures robustness and can reach the target within finite time horizon T. Although the controller synthesized is conservative in nature, such behavior remains acceptable while navigating an environment with external disturbance.

5. Simulation results

5.1. Example-I

We study the efficacy of the proposed approach in simulation by considering an inverted pendulum system with external disturbance:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}x_2\\ -\sin x_2\end{bmatrix} + \begin{bmatrix}0\\1\end{bmatrix} u + \begin{bmatrix}0\\ \cos x_1\end{bmatrix} d, \tag{38}$$

where x_1 and x_2 are the angle (θ) and angular velocity (ω) of the pendulum respectively, u is the input torque applied and d is the external disturbance to the system (see Fig. 4).

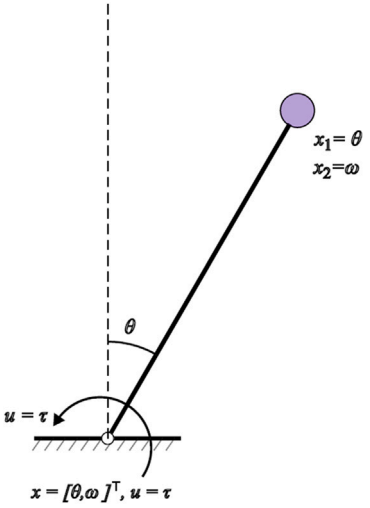


Fig. 4. Example I: Schematic of the 2D Inverted pendulum system.

The disturbance is scalar, i.e., $p = 1$, and enters the dynamics through

$$g_d(x)d = \begin{bmatrix}0\\ \cos x_1\end{bmatrix} d.$$

The admissible disturbance set is defined as

$$\mathcal{D} := \{d(t) \in \mathbb{R} : |d(t)| \le \rho\}, \qquad \rho = 1.25.$$

In the discrete-time implementation, consistent with Lemma 2, the environment applies an adversarial disturbance at each sampling instant on the boundary of the admissible set, i.e.,

$$d_k \in \{-1.25, +1.25\},$$

with the sign selected so as to oppose the controller and maximize the instantaneous cost. Hence, no fixed analytic waveform is prescribed a priori; rather, the disturbance is selected online according to the worst-case boundary rule induced by the robust reachability formulation.

The target set considered is a circular zone with the origin as the center of the target set. The target set is defined by considering the sub-level set of the following function:

$$\ell(x) = x_1^2 + x_2^2 - 0.25.$$

The TIRF (24) is considered with $\mathbf{w} = 3.5$, $\mathbf{w_C} = 0.3$ with $\alpha = 0.45$ in the incentive reward function (23). Each training episode consists of simulation over 12 s, which, under sampling period of 0.01s, translates to 1200 simulation steps. We train the system by running the DDPG over 250 episodes. The TIRF hyperparameters have distinct roles. The parameter α controls the steepness of the dense incentive term near the target boundary: larger values yield stronger local shaping, while smaller values give smoother but weaker guidance. The constant $\mathbf{w}$ sets the magnitude of the sparse telescopic bonus awarded when the running minimum cost improves, whereas $\mathbf{w_C}$ scales the dense incentive term so that it guides exploration without overwhelming the telescopic objective. In the reported simulations, $\alpha = 0.45$, $\mathbf{w} = 3.5$, and $\mathbf{w_C} = 0.3$ were selected empirically to balance boundary sensitivity, reward magnitude, and numerical stability during training.

Actor–critic implementation details. For implementation, we consider data at discrete time k, that is obtained by discretization of continuous data using standard zero order hold based sampling scheme. The DDPG agent uses an augmented Markov state,

$$\bar{x}_k = [x_{1,k},\ x_{2,k},\ J_k,\ \tau_k]^\top \in \mathbb{R}^4,$$

Table 1
DDPG training hyper parameters for 2D Inverted Pendulum.

actor learning rate	3×10^{-4}
critic learning rate	2×10^{-3}
discount factor	1.0
L2 regularization factor (actor)	3×10^{-3}
L2 regularization factor (critic)	2×10^{-3}
α (barrier-like reward function)	0.45
Replay buffer size	1×10^{5}
Mini-batch size	256
Soft target update coefficient	0.01
Exploration noise standard deviation	0.1
Exploration noise decay rate	1×10^{-3}

where x_1 and x_2 denote the pendulum angle and angular velocity, J_k is the running minimum-distance cost used in the telescopic reward construction, and $\tau_k := T - k\Delta t$ is the time-to-go. The inclusion of J_k and τ_k allows the finite-horizon reachability problem to be represented within a Markov decision process.

The actor network maps $\bar{x}_k$ to the scalar control input $u_k \in \mathbb{R}$ and is implemented as a fully connected network with architecture 4–64–1, with ReLU activation in the hidden layer and tanh activation in the output layer. The critic network receives the pair $(\bar{x}_k, u_k)$ and consists of two input branches: a state branch of architecture 4–64 with ReLU activation, and an action branch of dimension 1. After concatenation, the resulting 65-dimensional feature vector is passed through a 65-64-1 fully connected head, with ReLU activation in the hidden layer and a linear scalar output for $Q(\bar{x}_k, u_k)$. This concatenated critic structure captures the coupling between system dynamics and control input. The resulting architectures are shown in Figs. 10 and 11.

Training uses an actor learning rate of 1×10^{-4}, a critic learning rate of 1×10^{-3}, L_2 regularization factor 3×10^{-3} for both networks, and decaying Gaussian exploration noise with initial standard deviation 0.10 and decay rate 1×10^{-3}. The replay buffer capacity is set to 1×10^{5} transitions, mini-batches of size 256 are sampled for each gradient update, and the target networks are soft-updated with coefficient of 0.1.

The reward and the corresponding average reward computed over 150 episodes are presented in Fig. 5, which shows the learning profile of the controller under the reward function (24) and the hyperparameters presented in Table 1. As it can be seen, the reward (as well as the average reward) increases progressively, demonstrating an improvement in the performance of the controller and settles to a non-negative value (above zero). The settlement of the reward around a constant value demonstrates that saturation is reached on critic learning. The latter implies convergence of the actor network to corresponding optimal policy.

Training stability is assessed empirically rather than through a formal convergence proof. The per-episode reward increases as the agent learns to reach the target and saturates near its maximum after approximately 1047 episodes, at which point training is terminated. In addition to the reward and the initial critic estimate Q_0, Fig. 6 shows the critic loss norm and the actor policy loss norm during training. The critic converges faster compared to the actor (as expected). This confirms that the gradients are stable during the learning procedure. To further reinforce the stability of learning procedure, we can overlay the reward per episode to show the overall stability of both actor and critic networks thus learning the appropriate optimal policies. These quantities remain bounded and settle as learning proceeds, which is consistent with numerically stable actor–critic updates for the considered benchmark. We emphasize, however, that these illustrations do not constitute a Lyapunov proof of convergence of DDPG or a formal closed-loop stability proof of the learned policy; they provide empirical evidence of stable training for the benchmark studied here (see Remark 3.3).

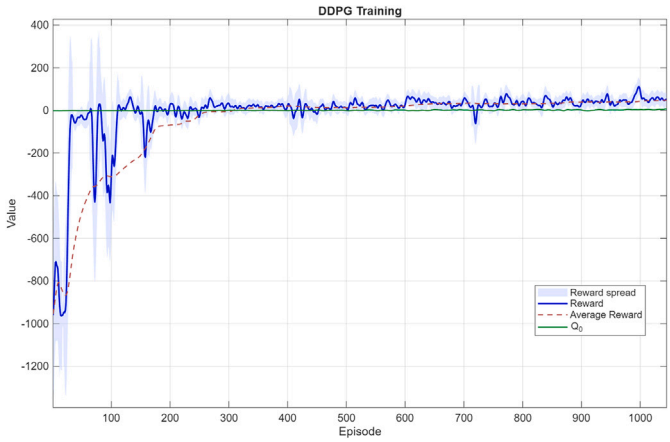


Fig. 5. Example I: Reward during DDPG training.

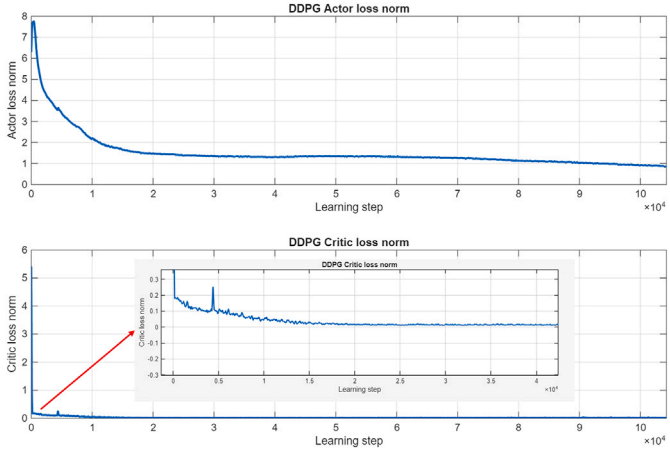


Fig. 6. Example I: Critic loss and actor loss during training.

Fig. 7 illustrates the temporal difference (TD) error norm that converges to a bounded value and remains stationary. The nonzero residual is a consequence of the critic network approximation of the value function. The stationarity of TD error norm verifies that the critic has reached a stable learning.

After training, the critic network approximates the optimal value function (15) and the actor approximates the optimal control policy (10). The actor thus serves as a near-optimal feedback controller robust to worst-case disturbance. Fig. 8 illustrates closed-loop trajectories from multiple initial conditions under the trained controller. Simulations of the system dynamics show that trajectories reach the target-set boundary within the finite horizon as intended.

Fig. 9 plots the state and control evolution under the trained controller alongside a grid-based dynamic programming baseline computed with the HelperOC toolbox (Chen and Herbert, 2019). The state trajectories align closely across the horizon, confirming consistency with the classical solution. In contrast, the DP control exhibits pronounced chattering effect, a well-known artifact of grid-based HJ/DP solvers that tend to produce bang–bang inputs in the absence of input regularization or control-effort penalties. By design, the proposed Telescopic Incentive Reward Function yields smooth, well-conditioned control while preserving reach performance. Finally, it is noted that trajectories generated by the proposed method terminate at the target-set boundary rather than at its center (the origin). This reflects the DDPG objective of boundary hitting within a finite horizon, whereas the HelperOC reference drives the state to the target center, explaining minor differences near the goal.

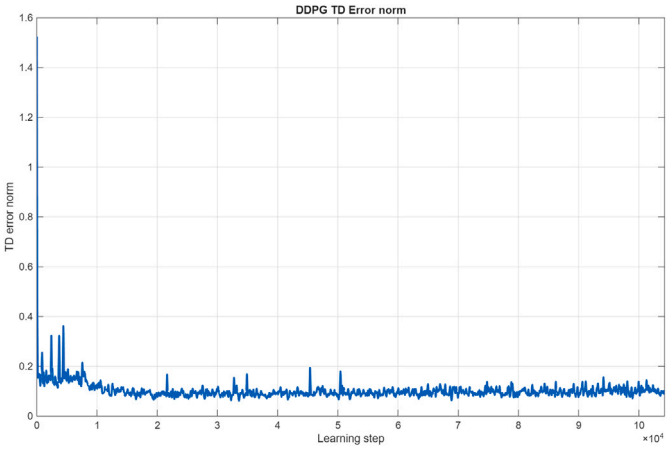


Fig. 7. Example I: Temporal Difference error.

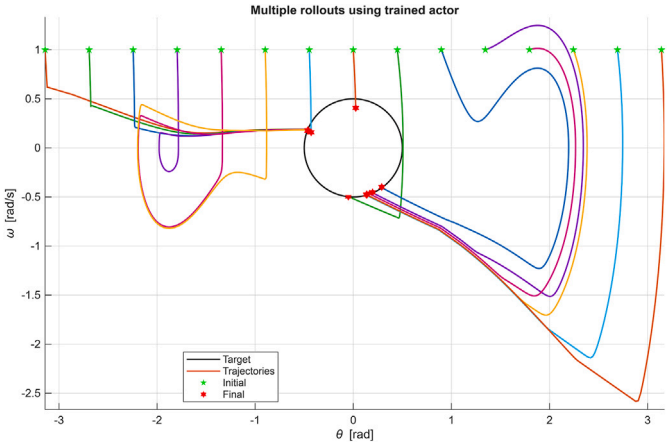


Fig. 8. Example I: System trajectory with multiple initial conditions.

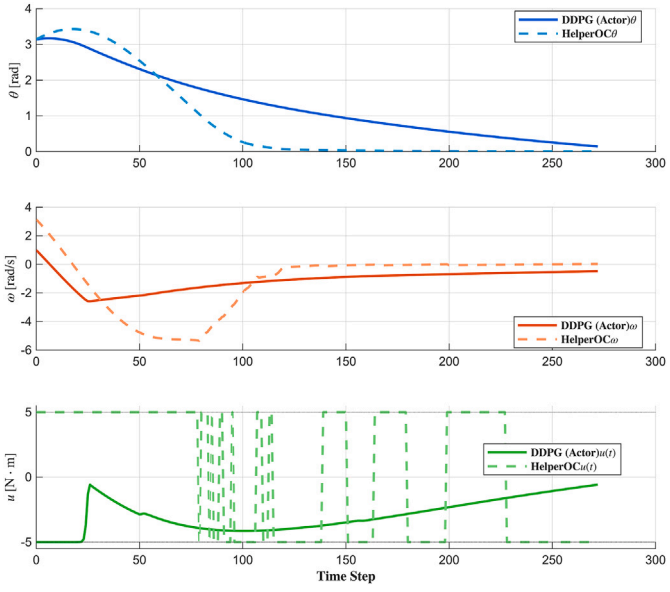


Fig. 9. Example I: System trajectory and control input using the proposed approach and the HelperOC toolbox (Chen and Herbert, 2019).

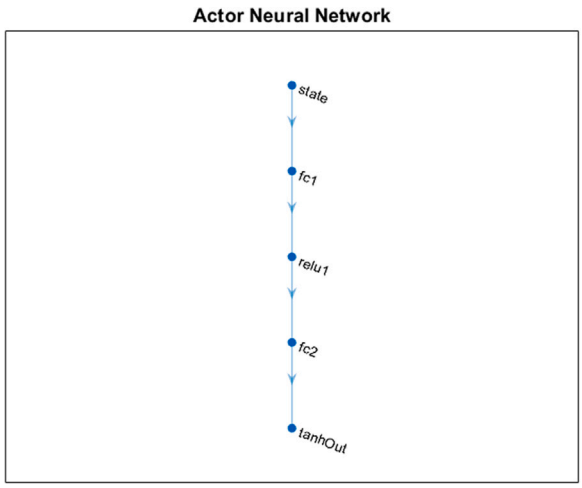


Fig. 10. Example I: Actor network architecture used in RR-DDPG.

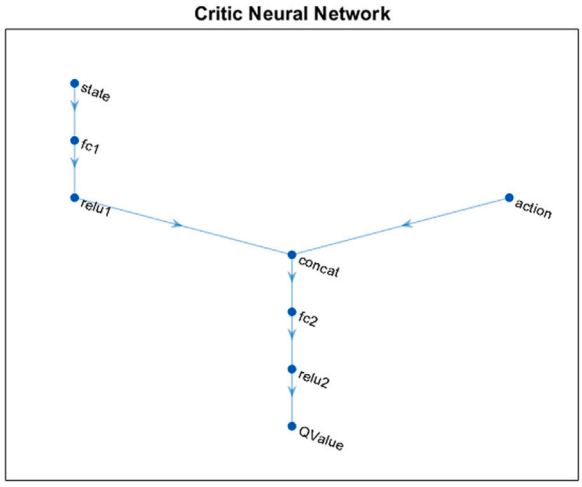


Fig. 11. Example I: Critic network architecture used in RR-DDPG.

5.2. Example-II

To further assess the scalability of RR-DDPG beyond the 2D inverted-pendulum benchmark, we consider a planar Dubins vehicle with bounded control and bounded disturbance (Chen et al., 2016):

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}v\cos(x_3)\\ v\sin(x_3)\\ 0\end{bmatrix} + \begin{bmatrix}0\\0\\u\end{bmatrix} + \begin{bmatrix}0\\0\\d\end{bmatrix} \tag{39}$$

Here, x_1 and x_2 denote the Cartesian position of the vehicle, and x_3 denotes its heading angle. The vehicle moves forward with constant translational speed v, i.e., $\sqrt{\dot{x}_1^2+\dot{x}_2^2}=v$, while the control input u and the disturbance d act on the heading dynamics. The admissible control and disturbance sets are $u \in [-u_{\max}, u_{\max}]$, and $d \in [-d_{\max}, d_{\max}]$. In the discrete-time implementation, consistent with Lemma 2, the environment applies an adversarial boundary disturbance at each sampling instant $d_k \in \{-d_{\max}, +d_{\max}\}$, with the sign chosen so as to maximize the instantaneous cost. Thus, the disturbance is not prescribed as a fixed analytic waveform, but is generated online according to the worst-case boundary rule induced by the robust reachability formulation. The parameters used in the simulations are $v = 1.0$, $u_{\max} = 1.0$ rad/s, $d_{\max} = 0.25$ rad/s, and sampling time $T_s = 0.05$ s (see Fig. 12).

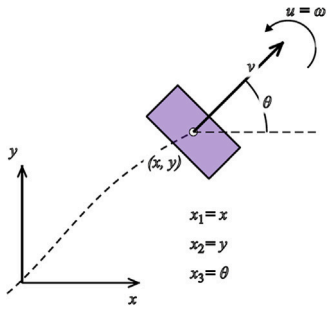


Fig. 12. Schematic of 3D Dubins car.

Table 2
DDPG training hyperparameters for 3D Dubins.

actor learning rate	1×10^{-4}
critic learning rate	1×10^{-3}
discount factor	1.0
L2 regularization factor (actor)	1×10^{-3}
L2 regularization factor (critic)	3×10^{-3}
α (barrier-like reward function)	0.45
Replay buffer size	1×10^{6}
Mini-batch size	256
Soft target update coefficient	0.01
Exploration noise standard deviation	0.10
Exploration noise decay rate	5×10^{-3}

The target set is cylindrical in the full state space and depends only on the position coordinates:

$$\ell(x) = x_1^2 + x_2^2 - 0.25. \tag{40}$$

The TIRF (24) is parameterized by $\mathbf{w} = 2.75$, $\mathbf{w_C} = 0.65$, and $\alpha = 0.45$. Each training episode spans 5 s, which corresponds to 100 simulation steps under sampling period $T_s = 0.05$ s. The DDPG algorithm is trained for 1250 episodes.

Actor–critic implementation details. The continuous-time dynamics (39) are discretized using a fourth-order Runge–Kutta scheme under zero-order-hold inputs. The augmented Markov state used by the DDPG agent is

$$\bar{x}_k = [x_{1,k},\ x_{2,k},\ \sin(x_{3,k}),\ \cos(x_{3,k}),\ J_k,\ \tau_k]^\top \in \mathbb{R}^6, \tag{41}$$

where J_k is the running minimum-distance cost and τ_k is the time-to-go. Although the physical system state is 3-dimensional, the augmented state is 6-dimensional because the finite-horizon formulation introduces J_k and τ_k, while the heading variable is represented by $\sin(x_{3,k})$ and $\cos(x_{3,k})$ rather than by $x_{3,k}$ directly in order to avoid angular discontinuities and preserve periodicity.

The actor network maps $\bar{x}_k$ to the scalar control input u_k and is implemented as a fully connected network with architecture 6-128-64-1, using ReLU activations in the hidden layers and a tanh activation in the output layer. The critic receives the pair $(\bar{x}_k, u_k)$ and consists of a state branch of architecture 6-128 with ReLU activation and an action branch of dimension 1. After concatenation, the resulting 129-dimensional feature vector is passed through a fully connected head of architecture 129-128-64-1, with ReLU activations in the hidden layers and a linear scalar output for $Q(\bar{x}_k, u_k)$. This structure captures the coupling between the augmented state and the steering action in the finite-horizon setting. The corresponding architectures are shown in Figs. 13 and 14.

Table 2 summarizes the training hyperparameters used for the 3D Dubins benchmark. Fig. 16 shows the reward and its running average during training under the proposed TIRF. The training process terminates after approximately 1250 episodes, with the running average computed over 125 episodes.

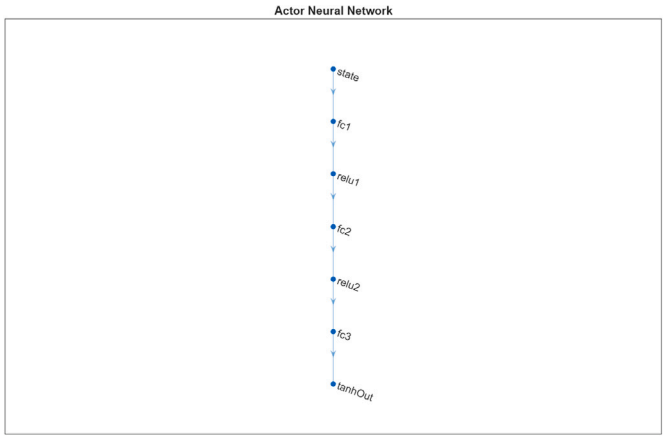


Fig. 13. Example II: Actor network architecture used in RR-DDPG.

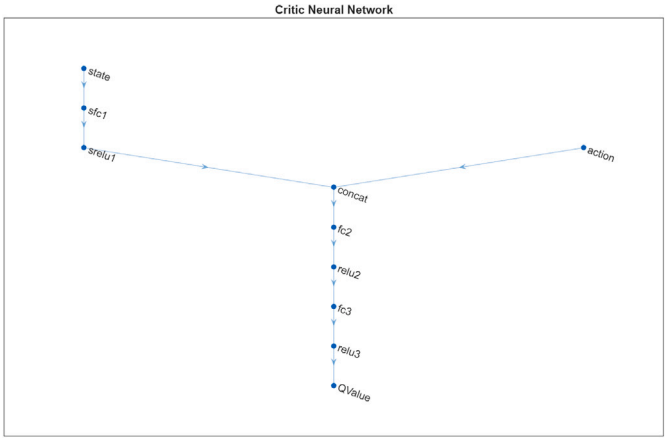


Fig. 14. Example II: Critic network architecture used in RR-DDPG.

After training, the actor network is deployed as a feedback controller. The multi-start rollouts in Fig. 15 show that trajectories initialized from diverse positions and headings reach the target-set boundary under the adversarial disturbance model, demonstrating that the trained actor is directly deployable as a controller.

To assess training stability empirically, Fig. 17 reports the actor and critic loss norms, and Fig. 18 reports the temporal-difference (TD) error norm. Together with the reward and Q_0 traces in Fig. 16, these diagnostics provide empirical evidence of stable training on the 3D benchmark. A transient degradation is observed near episode 200, where exploration noise drives the agent towards an unfavorable policy/initial-state combination; the controller recovers within approximately 50 episodes, which is also reflected in the temporary increase in actor loss. The TD-error norm remains bounded and settles to a low residual level, which is consistent with a stable critic approximation.

Compared with the 2D inverted-pendulum benchmark, the 3D Dubins example exhibits noisier loss and TD-error traces. This is expected because the Dubins vehicle is nonholonomic and the position states (x_1, x_2) are influenced only indirectly through the heading state x_3, resulting in a more intricate value function. Nevertheless, Fig. 19 shows that RR-DDPG preserves reach performance while producing a substantially smoother control input than the HelperOC baseline.

After successful training, the trained actor network is deployed as a controller and the resulting trajectory for various initial conditions are presented in Fig. 15. The trajectories presented in Fig. 15 demonstrate the performance of the trained network as a deployable controller for various initial conditions.

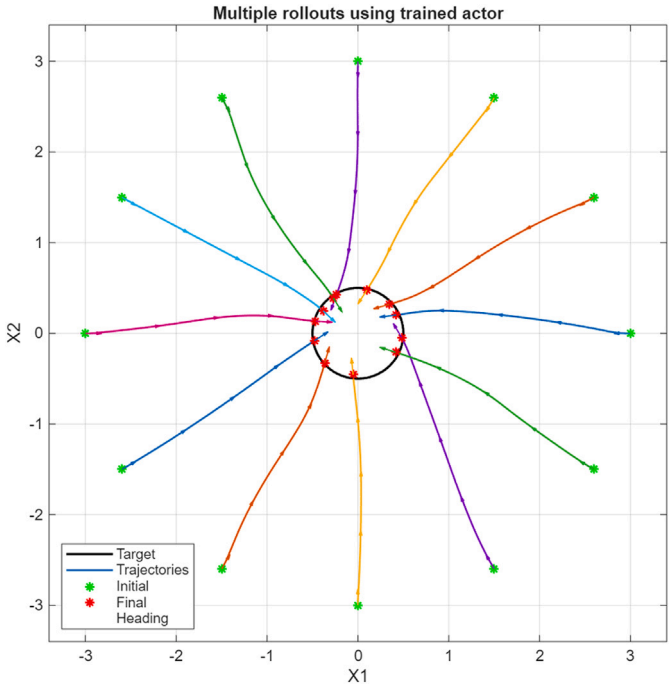


Fig. 15. Example II: System trajectory with multiple initial conditions.

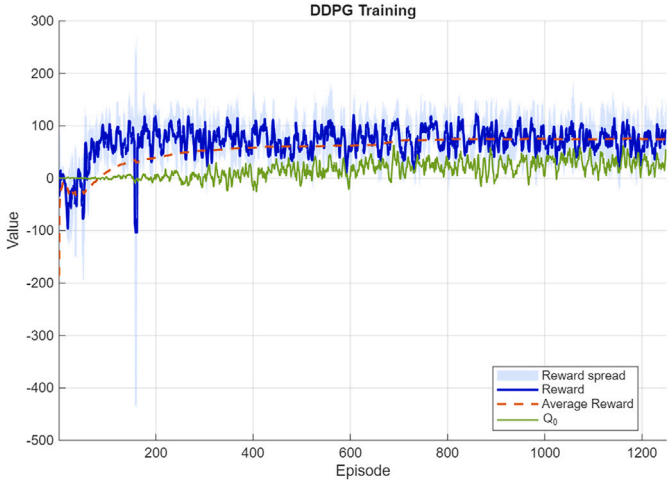


Fig. 16. Example II (3D Dubins): Reward during training.

To empirically assess the stability of the training procedure, Fig. 17 illustrates the actor and critic loss norms respectively and along with Fig. 16 verify that the training is stable. It can be noted in Fig. 16, at approximately 200 episode the reward drops suddenly due to the high exploration noise that encounters an unfavorable policy and initial condition; the agent however recovers within approximately 50 episodes. This also explains the increase in actor loss norm (at episode 200) in Fig. 17.

The TD error norm is presented in Fig. 18 which is bounded and remains stationary around a lower value confirming that the critic approximation quality is preserved, as the state dimension increases.

For the 3D system (39), although the learning is stable, the TD error, actor and critic loss norms, and Q_0 are noisy compared to the 2D inverted-pendulum benchmark. This is due to the nonholonomic, underactuated nature of the Dubins dynamics i.e., the position states

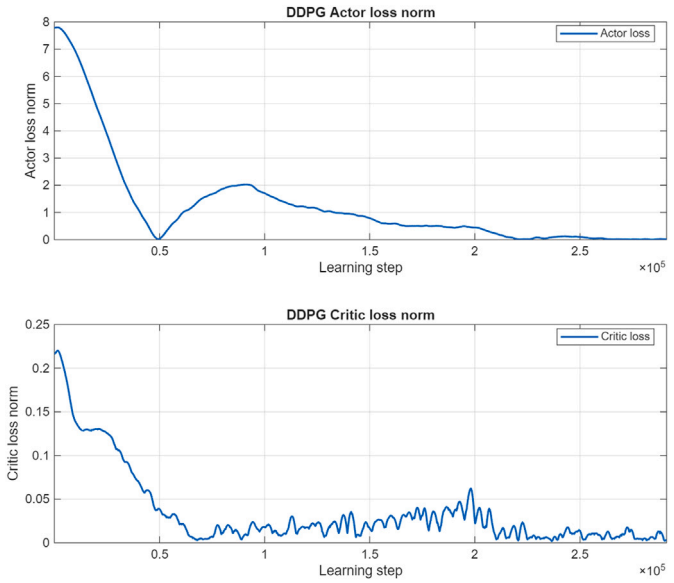


Fig. 17. Example II (3D Dubins): Actor and Critic loss norm.

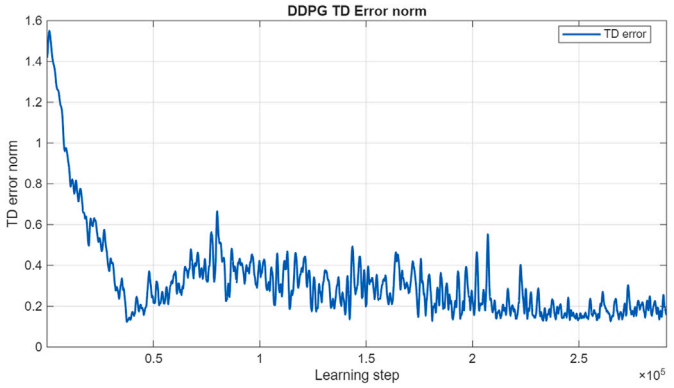


Fig. 18. Example II (3D Dubins): TD error.

(x_1, x_2) are not directly controllable and can only be influenced indirectly through the heading x_3, resulting in a more complex value function that is harder to approximate with finite-capacity networks.

The effectiveness of the proposed RR-DDPG framework is further supported by Fig. 19, where the closed-loop trajectories successfully reach the target-set boundary while producing a smooth, continuous control signal, in contrast to the oscillatory control input generated by the HelperOC baseline.

5.3. Discussion

All simulations and training runs (HelperOC, RR-DDPG, and DeepReach for the 2D benchmark) were carried out on the same workstation (Intel Core i9-13950HX, 64 GB RAM, and an NVIDIA GPU) to enable a consistent wall-clock comparison. Table 3 summarizes the computational time of three representative HJ reachability solution pipelines: the grid-based solver HelperOC, the learning-based HJ solver DeepReach, and the proposed RR-DDPG method.

For the 2D inverted-pendulum benchmark, HelperOC is the fastest method (10 min 33 s), which is expected for a low-dimensional problem where grid-based dynamic programming remains tractable. However, this advantage is largely limited to small state dimensions and deteriorates rapidly as dimensionality increases because of the curse of dimensionality. DeepReach avoids an explicit grid by learning a neural

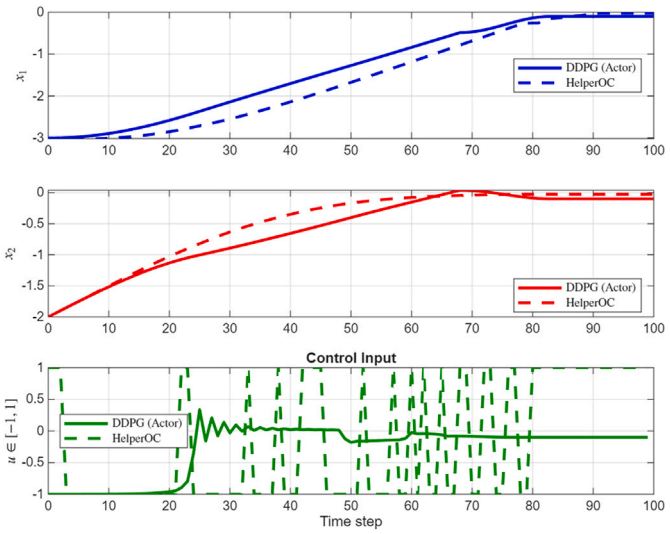


Fig. 19. Example II (3D Dubins): System trajectory and control input using proposed approach and HelperOC toolbox.

approximation of the HJI value function, but on this benchmark it required the largest computational time (12 h 12 min). RR-DDPG lies between these two extremes: on the 2D problem, its training time is higher than HelperOC but substantially lower than DeepReach, while simultaneously providing a directly deployable feedback controller through the actor network.

For the 3D Dubins benchmark, RR-DDPG required 56 min 32 s, compared with 1 h 32 min for HelperOC. A direct 3D DeepReach implementation was not carried out on our workstation. Therefore, the value reported in Table 3 is taken from Bansal and Tomlin (2021), where DeepReach was applied to the Air3D problem. Since Air3D and Dubins3D are not identical benchmarks and were not executed under the same hardware/software setting, this value is included only as an order-of-magnitude reference for a 3D neural-HJ pipeline, not as a strict one-to-one runtime comparison.

⋆ *Note on computational time.* DeepReach was implemented and timed by the authors only for the 2D inverted-pendulum benchmark. The 3D DeepReach time is taken from Bansal and Tomlin (2021) for the Air3D example and is reported only as an order-of-magnitude reference.

Remark 5.1. A direct numerical comparison of value-function accuracy across the methods in Table 3 is not entirely straightforward. HelperOC and DeepReach solve the classical coupled min–max HJ reachability problem, whereas RR-DDPG learns an approximate numerical solution associated with the environment-based disturbance treatment adopted in this work. Accordingly, Table 3 should be interpreted primarily as a comparison of computational burden, scalability trends, and controller-synthesis workflow rather than as a strict ranking of value-function approximation accuracy.

HelperOC and DeepReach are primarily value-function solvers; obtaining a feedback controller from them still requires a separate optimization step based on (10). In contrast, RR-DDPG learns the value function and the feedback policy jointly. After training, the actor network can be directly deployed for multiple initial conditions, as illustrated in Fig. 8. This makes RR-DDPG attractive when online deployability and smooth control actions are important, even if a low-dimensional grid-based solver remains competitive in raw computation time on small benchmarks.

Among learning-based baselines, DeepReach was selected because it is a representative neural HJ reachability method for continuous-state systems. We did not include robust RL methods developed for different

Table 3
Comparison of HJ reachability solution pipelines. HelperOC and RR-DDPG times were measured on the same workstation; the 3D DeepReach entry is a literature-based reference for Air3D and is included for indicative comparison only.

Method	Inverted pendulum (2D)	Dubins (3D)
HelperOC	10 min 33 s	1 h 32 min
DeepReach	12 h 12 min	~ 16 h⋆
RR-DDPG	35 min 47 s	56 min 32 s

problem classes. For instance, STP-based reinforcement learning approaches for Boolean network synchronization (Zhang et al., 2025b) and detectability synthesis of probabilistic Boolean networks (Zhang et al., 2025a) operate on discrete state spaces with tabular Q-learning and address synchronization or detectability objectives, which differ in both state-space structure and reward structure from continuous-state, finite-horizon reachability problem. Similarly, deep Q-learning variants for weapon-target assignment (Yang et al., 2025) solve combinatorial allocation problems rather than continuous-time reach problems under worst-case disturbance. Although the works share the same theme of using RL to compute the optimal control policy, they do not solve the same finite-horizon continuous-state HJ reachability problem considered here and therefore do not admit a fair benchmark on the present inverted-pendulum and Dubins systems.

6. Conclusion

The work presents a novel approach to solve the robust reach problem using deep deterministic policy gradient (DDPG) algorithm for a non-linear systems under external disturbance. We present a comprehensive approach by casting the Hamilton Jacobi -Reach (HJ-R) problem within DDPG framework learning both HJ-R value function and optimal control policy through critic and actor networks respectively. Simulations show that proposed RR DDPG solves the reach task across diverse initial conditions, with trajectories reaching the target. Compared with the classical dynamic programming based approach (in HelperOC toolbox) and DeepReach, our approach mitigates the undesirable chattering behavior and gives a smooth control input. These gains arise from the proposed telescopic incentive reward function, which drives the agent towards the target within a finite time horizon while regularizing the control effort. We also reduce computation by modeling the disturbance through agent environment interaction. Because value and policy are represented by deep neural networks, the proposed framework avoids the grid-based scaling burden of classical HJ solvers and shows encouraging extension from the 2D inverted-pendulum benchmark to the more complex 3D Dubins benchmark. Future work will address reach avoid and stay problems.

CRediT authorship contribution statement

Satya Marthi: Writing – original draft, Formal analysis. **Mayank S. Jha:** Writing – review & editing, Supervision. **Didier Theilliol:** Writing – review & editing, Supervision. **Jean-Christophe Ponsart:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported and funded by French National project ANR (Agence nationale de la recherche) under the SOS (Self-organizing, smart and safe robots) (ANR-23-IAS2-0003).

Data availability

Data will be made available on request.

References

Ahmed, M.H., AboHussien, A., El-Shafei, A., Darwish, A.M., Abdel-Gawad, A.H., 2023. Active control of flexible rotors using deep reinforcement learning with application of multi-actor-critic deep deterministic policy gradient. Eng. Appl. Artif. Intell. 124, 106593.

Andrei, G.C., Jha, M.S., Theilliol, D., 2023. Complementary reward function based learning enhancement for deep reinforcement learning. In: Recent Developments in Model-Based and Data-Driven Methods for Advanced Control and Diagnosis. Springer Nature Switzerland, pp. 237–247. http://dx.doi.org/10.1007/978-3-031-27540-1_21.

Aubin, J.P., 2001. Viability kernels and capture basins of sets under differential inclusions. SIAM J. Control Optim. 40 (3), 853–881.

Aubin, J.P., Bayen, A.M., Saint-Pierre, P., 2011. Viability Theory: New Directions, second ed. Springer.

Bansal, S., Chen, M., Herbert, S.L., Tomlin, C.J., 2017. Hamilton-Jacobi reachability: A brief overview and recent advances. In: Proc. IEEE Conf. Decision and Control.

Bansal, S., Tomlin, C.J., 2021. Deepreach: A deep learning approach to high-dimensional reachability. In: 2021 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 1817–1824.

Borquez, J., Chakraborty, K., Wang, H., Bansal, S., 2024. On safety and liveness filtering using Hamilton–Jacobi reachability analysis. IEEE Trans. Robot.

Bui, M., Lu, M., Hojabr, R., Chen, M., Shriraman, A., 2021. Real-time Hamilton–Jacobi reachability analysis of autonomous system with an fpga. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS, IEEE, pp. 1666–1673.

Chen, M., Herbert, S., 2019. HelperOC Library. https://github.com/HJReachability/helperOC.

Chen, M., Herbert, S., Tomlin, C.J., 2016. Exact and efficient Hamilton–Jacobi-based guaranteed safety analysis via system decomposition. arXiv preprint arXiv:1609.05248.

Chen, M., Tomlin, C.J., 2018. Hamilton–Jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management. Annu. Rev. Control. Robot. Auton. Syst. 1 (1), 333–358.

Dewey, D., 2014. Reinforcement learning and the reward engineering principle. In: AAAI Spring Symposia.

Evans, L.C., 2010. An Introduction to Mathematical Optimal Control Theory. In: Lecture Notes, University of California, Berkeley.

Fisac, J.F., Akametalu, A.K., Zeilinger, M.N., Kaynama, S., Gillula, J.H., Tomlin, C.J., 2018. A general safety framework for learning-based control. IEEE Trans. Autom. Control 64 (7), 2737–2752.

Fisac, J.F., Chen, M., Akametalu, A.K., Tomlin, C.J., 2015. Reach-avoid problems with time-varying dynamics, targets and constraints. In: Proc. IEEE Conf. Decision and Control.

Ganai, M., 2024. Hamilton-Jacobi Reachability Estimation in Reinforcement Learning (Master's thesis). University of California, San Diego.

Ganai, M., Gong, Z., Yu, C., Herbert, S., Gao, S., 2023. Iterative reachability estimation for safe reinforcement learning. Adv. Neural Inf. Process. Syst. 36, 69764–69797.

Huang, Y., Liu, D., 2014. Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm. Neurocomputing 125, 46–56.

Jha, M.S., Kiumarsi, B., 2025. Off-policy safe reinforcement learning for nonlinear discrete-time systems. Neurocomputing 611, 128677. http://dx.doi.org/10.1016/j.neucom.2024.128677.

Jha, M.S., Theilliol, D., Weber, P., 2023. Model-free optimal tracking over finite horizon using adaptive dynamic programming. Optim. Control. Appl. Methods 44 (6), 3114–3138. http://dx.doi.org/10.1002/oca.3028.

Kanso, S., Jha, M.S., Theilliol, D., 2024. Off-policy model-based end-to-end safe reinforcement learning. Internat. J. Robust Nonlinear Control 34 (4), 2806–2831. http://dx.doi.org/10.1002/rnc.7109.

Kanso, S., Jha, M.S., Theilliol, D., 2025. Reinforcement learning-based degradation-tolerant control design for affine nonlinear systems. Asian J. Control 1–17. http://dx.doi.org/10.1002/asjc.3780.

Kanso, S., Shekhar Jha, M., Theilliol, D., 0000. Safe Reinforcement Learning for Optimal Tracking of Continuous-Time Nonlinear Systems, Internat. J. Robust Nonlinear Control, https://doi.org/10.1002/rnc.70518, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rnc.70518, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.70518.

Kim, J.W., Park, B.J., Yoo, H., Oh, T.H., Lee, J.H., Lee, J.M., 2020. A model-based deep reinforcement learning method applied to finite-horizon optimal control of nonlinear control-affine system. J. Process Control 87, 166–178.

Kokolakis, N.M., Vamvoudakis, K.G., Haddad, W., 2023. Reachability analysis-based safety-critical control using online fixed-time reinforcement learning. In: Learning for Dynamics and Control Conference. PMLR, pp. 1257–1270.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Liu, J., Wang, Y., Sun, C., 2025. Unmanned surface vehicle autonomous racing and obstacle avoidance with robust adversarial deep reinforcement learning. Eng. Appl. Artif. Intell. 161, 112250.

Lu, Y., Li, Z., Zhou, Y., Li, N., Mo, Y., 2024. Mpc-inspired reinforcement learning for verifiable model-free control. In: 6th Annual Learning for Dynamics & Control Conference. PMLR, pp. 399–413.

Margellos, K., Lygeros, J., 2011. Hamilton–Jacobi formulation for reach–avoid differential games. IEEE Trans. Autom. Control 56 (8), 1849–1861.

Marthi, S.V.C., Jha, M.S., Kanso, S., Ponsart, J.C., Theilliol, D., 2025. On-policy safe reinforcement learning under input saturation and state constraints for nonlinear discrete-time systems. In: Proceedings of the IEEE Conference on Decision and Control. CDC.

Mitchell, I.M., Bayen, A.M., Tomlin, C.J., 2005. A time-dependent Hamilton–Jacobi formulation of reachable sets for continuous dynamic games. IEEE Trans. Autom. Control 50 (7), 947–957.

Rabiee, P., Safari, A., 2025. Safe exploration in reinforcement learning: Training backup control barrier functions with zero training-time safety violations. In: Ozay, N., Balzano, L., Panagou, D., Abate, A. (Eds.), Proceedings of the 7th Annual Learning for Dynamics & Control Conference. In: Proceedings of Machine Learning Research, vol. 283, PMLR, pp. 1326–1337, URL https://proceedings.mlr.press/v283/rabiee25a.html.

Rego, R.C., de Araujo, F.M.U., 2022. Lyapunov-based continuous-time nonlinear control using deep neural network applied to underactuated systems. Eng. Appl. Artif. Intell. 107, 104519.

Rutschke, T., Jha, M.S., Garnier, H., 2025. Neural ordinary differential equations based system identification for reinforcement learning with provable guarantees. In: Proceedings of the IEEE Conference on Decision and Control. CDC.

So, O., Ge, C., Fan, C., 2024. Solving minimum-cost reach avoid using reinforcement learning. Adv. Neural Inf. Process. Syst. 37, 30951–30984.

Suttle, W., Sharma, V.K., Kosaraju, K.C., Seetharaman, S., Liu, J., Gupta, V., Sadler, B.M., 2024. Sampling-based safe reinforcement learning for nonlinear dynamical systems. In: International Conference on Artificial Intelligence and Statistics. PMLR, pp. 4420–4428.

Wang, H., Miao, K., Madeira, D., Papachristodoulou, A., 2025. Learning neural controllers with optimality and stability guarantees using input-output dissipativity. arXiv preprint arXiv:2506.06564.

Yang, L., Dai, H., Shi, Z., Hsieh, C.-J., Tedrake, R., Zhang, H., 2024. Lyapunov-stable neural control for state and output feedback: a novel formulation. In: Proceedings of the 41st International Conference on Machine Learning. ICML '24, JMLR.org.

Yang, F., Gong, Z., Zhao, Z., Fang, C., 2025. AAV weapon target assignment via state split flow deep Q-learning network with threat degree index. IEEE Internet Things J.

Zhang, Z., Bian, C., Xia, C., Chen, Z., 2025a. Optimal-flip-based segmented reinforcement learning for detectability synthesis of probabilistic boolean networks. IEEE Trans. Neural Networks Learn. Syst.

Zhang, Z., Du, L., Li, H., Liu, Y., Chen, Z., 2025b. Node-set synchronization and calculation of general boolean networks combining STP and Q-learning. IEEE Trans. Syst. Man, Cybern.: Syst.