State-Aware Perturbation Optimization for Robust Deep Reinforcement Learning

Zongyuan Zhang, Tianyang Duan, Zheng Lin, Dong Huang, Zihan Fang, Zekai Sun, Ling Xiong, *Member, IEEE,* Hongbin Liang, *Member, IEEE,* Heming Cui, *Member, IEEE,* and Yong Cui, *Member, IEEE*

***Abstract*—Recently, deep reinforcement learning (DRL) has emerged as a promising approach for robotic control. However, the deployment of DRL in real-world robots is hindered by its sensitivity to environmental perturbations. While existing white-box adversarial attacks rely on local gradient information and apply uniform perturbations across all states to evaluate DRL robustness, they fail to account for temporal dynamics and state-specific vulnerabilities. To combat the above challenge, we first conduct a theoretical analysis of white-box attacks in DRL by establishing the Adversarial Victim Dynamics Markov Decision Process (AVD-MDP), to derive the necessary and sufficient conditions for a successful attack. Based on this, we propose the Selective State-Aware Reinforcement adversarial attack (STAR), to optimize perturbation stealthiness and state visitation dispersion. STAR first employs a soft mask-based state-targeting mechanism to minimize redundant perturbations, enhancing stealthiness and attack effectiveness. Then, it incorporates an information-theoretic optimization objective to maximize mutual information between perturbations, environmental states, and victim actions, ensuring a dispersed state-visitation distribution that steers the victim agent into vulnerable states for maximum return reduction. Extensive experiments demonstrate that STAR outperforms state-of-the-art benchmarks.**

***Index Terms*—Markov Decision Process, Deep Reinforcement Learning, Adversarial Attack, Robotic Manipulation**

I. Introduction

Robotic systems have become increasingly prevalent in mobile and distributed applications, ranging from autonomous navigation [1], [2], intelligent transportation [3], [4], and industrial manufacturing [5], [6]. While traditional robotic control methods have achieved considerable success in structured environments with predefined tasks, they struggle to adapt to dynamic scenarios, handle uncertainties, or learn from experience. Deep reinforcement learning (DRL) has emerged as a promising alternative for robotic control [7]. Unlike traditional approaches that rely on manually designed rules, DRL enables agents to acquire optimal behaviors through trial-and-error interactions with their environment. By learning a policy that maps environmental states to actions while maximizing long-term rewards, DRL is particularly effective for complex robotic tasks involving delayed feedback and temporal dependencies.

The robustness of DRL policies is crucial for their successful deployment in real-world robotic applications, as DRL agents can be highly sensitive to environmental perturbations [8]. Minor variations in the input state - whether arising from sensor noise, environmental changes, or adversarial attacks - can significantly impact an agent's decision-making process and potentially trigger catastrophic failures. White-box adversarial attacks serve as an effective tool for evaluating RL robustness by exposing potential vulnerabilities in the

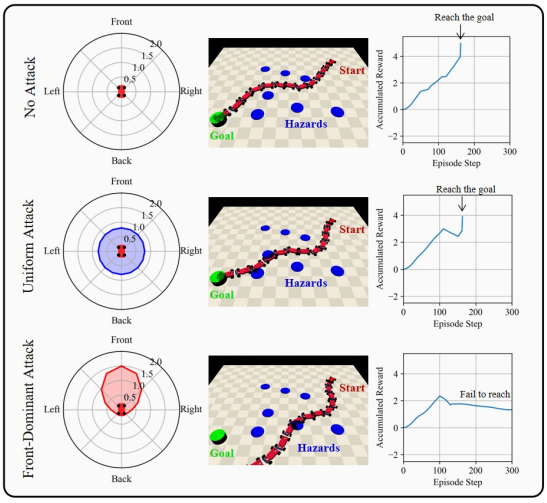


Fig. 1: Performance comparison of three attack strategies in a four-wheeled robot navigation task: No Attack (top), Uniform Attack (middle), and Front-Dominant Attack (bottom). Each row presents the attack magnitude distribution (left), navigation trajectory (middle), and accumulated reward (right). The Front-Dominant Attack exhibits the highest efficacy in disrupting navigation by concentrating perturbations in the frontal direction.

Z. Zhang, T. Duan, D. Huang, Z. Sun, and H. Cui are with the Department of Computer Science, The University of Hong Kong, Hong Kong SAR, China (e-mail: zyzhang2@cs.hku.hk; tyduan@cs.hku.hk; dhuang@cs.hku.hk; zksun@cs.hku.hk; heming@cs.hku.hk).

Z. Lin is with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong SAR, China (e-mail: linzheng@eee.hku.hk).

Z. Fang is with the Department of Computer Science, City University of Hong Kong, Hong Kong SAR, China (e-mail: zihanfang3-c@my.cityu.edu.hk).

L. Xiong is with the School of Computer and Software Engineering, Xihua University, Chengdu 610039, China. (e-mail: lingdonghua99@163.com)

H. Liang is with the National United Engineering Laboratory of Integrated and Intelligent Transportation, and the National Engineering Laboratory of Integrated Transportation Big Data Application Technology, Southwest Jiaotong University, Chengdu 611756, China (e-mail: hbliang@swjtu.edu.cn).

Y. Cui is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: cuiyong@tsinghua.edu.cn).

The work is supported in part by National Key R&D Program of China (2022ZD0160201), HK RGC RIF (R7030-22), HK RGC GRF (ref No.: 17208223 & 17204424), a Huawei flagship research grant in 2023, SuperNetAI, and the HKU-CAS Joint Laboratory for Intelligent System Software.

(Corresponding author: Tianyang Duan; Zheng Lin)

learned policies. With full access to the model architecture and parameters, white-box attacks can systematically analyze policy networks to generate perturbations [9]. Through adversarial training with perturbed samples generated by such attacks, DRL agents can learn to maintain robust performance under state perturbations, enhancing reliability in real-world deployments.

However, existing white-box attack methods face significant challenges in targeting deep reinforcement learning (DRL) agents, as they primarily rely on local gradient information to generate perturbations [10]–[12]. Adapted from supervised learning attack paradigms, these methods assume temporal independence and focus on instantaneous state-action mappings, neglecting the temporal dynamics inherent in Markov Decision Processes (MDPs). As a result, they fail to generate perturbations that can effectively disrupt the agent's cumulative rewards over extended time horizons. Furthermore, these methods apply perturbations indiscriminately across all states without identifying key features that critically impact performance. This is particularly problematic in high-dimensional state spaces, where only a subset of variables—such as specific joint angles in robotic manipulation—are crucial for policy execution. Though some works attempt to weight perturbations based on policy network gradient magnitudes [13]–[15], aiming to maximize behavioral deviation from the original policy. Nevertheless, in DRL, this does not directly align with the core attack objective of minimizing cumulative rewards, as agents can adapt by selecting alternative actions to maintain comparable long-term performance.

To investigate the limitations of state-agnostic attacks, we conduct experiments using a four-wheeled robot navigation task in the Safety Gymnasium environment [16]. As shown in Figure 1, the robot's objective is to navigate from the start position to the goal while avoiding hazards. We evaluate three attack strategies: (i) *No Attack*, where the robot operates without interference; (ii) *Uniform Attack*, where perturbations are applied uniformly across all state dimensions; and (iii) *Front-Dominant Attack*, where perturbations are concentrated in the robot's forward movement direction. For each strategy, we analyzed the perturbation distribution across state dimensions, the resulting trajectory, and the accumulated reward over time. Under *No Attack*, the robot follows an optimal trajectory, successfully avoiding hazards and reaching the goal, with the accumulated reward increasing steadily. In contrast, *Uniform Attack* introduces perturbations across all state dimensions, causing minor trajectory deviations. While the robot occasionally approaches hazards, it still reaches the goal, with a slight dip in the accumulated reward curve before near-optimal performance is regained. Finally, *Front-Dominant Attack* applies perturbations predominantly in the frontal direction, severely disrupting the robot's navigation. This results in significant trajectory deviation, with the robot encountering hazards and failing to reach the goal, leading to a substantial reduction in the accumulated reward. These experiments demonstrate that, when using the same perturbation budget as the *Uniform Attack*, the *Front-Dominant Attack* is more effective by targeting specific state dimensions, thereby revealing the vulnerability of DRL agents to state-specific perturbations.

To address the above issue, the paper aims to address a fundamental research question: *How can we develop a state-aware adversarial attack framework that identifies and exploits vulnerable states in DRL policies while accounting for long-term reward impact?* To this end, we first propose the Adversarial Victim Dynamics Markov Decision Process (AVD-MDP) to formalize white-box adversarial attacks in DRL, enabling a systematic analysis of adversarial-victim interactions under perturbation constraints. Based on this, we derive the necessary and sufficient condition for successful attacks and identify two key properties essential for effective adversarial strategies: stealthiness of perturbations and dispersion in state visitation. Guided by these insights, we then propose **S**elective S**T**ate-**A**ware **R**einforcement adversarial attack (STAR), a white-box attack algorithm that jointly optimizes these properties. STAR employs a soft mask-based state-targeting mechanism to minimize perturbations on redundant state dimensions, enhancing stealthiness while maintaining attack efficacy. Additionally, it incorporates an information-theoretic objective to maximize mutual information between adversarial perturbations, environmental states, and victim actions, promoting a dispersed state-visitation pattern. By strategically inducing the victim agent to visit vulnerable states, STAR maximizes return reduction under a fixed perturbation budget. The key contribution of the paper can be summarized as follows:

- We propose a novel adversarial attack framework for DRL that systematically accounts for temporal dependencies and state-specific vulnerabilities, addressing key limitations in existing white-box attack methods.
- We theoretically analyze the interaction between adversarial and victim policies by establishing the Adversarial Victim Dynamics Markov Decision Process (AVD-MDP), to derive the necessary and sufficient conditions for a successful attack.
- Based on the derived interaction, we propose the Selective State-Aware Reinforcement adversarial attack (STAR), a principled white-box attack algorithm that optimizes both stealthiness and distribution via a soft mask-based state-targeting mechanism and an information-theoretic optimization objective to maximize attack effectiveness under a certain perturbation budget.
- We empirically evaluate STAR with extensive experiments. The results demonstrate that STAR outperforms state-of-the-art frameworks in evaluating the robustness of DRL.

The rest of the paper is organized as follows: Section II discusses related work and technical limitations. Section III elaborates on the system model. Section IV presents a theoretical analysis of the AVD-MDP process. Section V presents the system design of STAR. Section VI details the experimental setup, followed by performance evaluation in Section VII. Finally, conclusions are presented in Section VIII.

II. Related Work

Adversarial attacks expose the vulnerabilities of deep neural networks (DNNs) by introducing carefully crafted perturba-

tions into input data, leading to incorrect predictions during inference. These perturbations, though imperceptible to humans, can significantly alter DNN outputs [17]. White-box attacks constitute a critical category of adversarial attacks, where the adversary has full access to the model, including its architecture, parameters, and gradients. Fast Gradient Sign Method (FGSM) [18] is a seminal white-box attack that efficiently generates adversarial perturbations by leveraging the gradient of the loss function with respect to the input, addressing the computational inefficiencies of earlier approaches. Projected Gradient Descent (PGD) [19] enhances attack effectiveness through iterative gradient-based updates, projecting perturbed inputs back into the constrained space after each step. Wong et al. [20] improved attack efficiency by introducing random initialization points in FGSM-based attacks. Schwinn et al. [21] increased attack diversity by injecting noise into the output while mitigating gradient obfuscation caused by low-confidence predictions. Beyond standard white-box attacks, various techniques have been proposed to improve adversarial transferability [22], [23]. These include random input transformations [24], translation-invariant perturbation aggregation [25], and substituting momentum-based gradient updates with Nesterov accelerated gradients [26].

Adversarial attacks in deep reinforcement learning (DRL) have been widely studied, revealing critical vulnerabilities in agent policies [27], [28]. Huang et al. [29] demonstrated that policy-based reinforcement learning (RL) agents are highly susceptible to adversarial perturbations on state observations, showing that FGSM attacks can significantly degrade performance in Atari 2600 games. Pattanaik et al. [30] introduced adversarial examples by computing gradients of the critic network with respect to states and integrating them into the training of Deep Double Q-Network (DDQN) and Deep Deterministic Policy Gradient (DDPG), enhancing robustness. Lin et al. [31] proposed strategically timed attacks that selectively perturb key decision-making states, achieving high attack success rates with minimal perturbations. Recent work has shifted towards theoretical modeling of adversarial attacks within the Markov Decision Process (MDP) framework. Weng et al. [32] introduced a systematic evaluation framework for RL robustness in continuous control, defining two primary threat models: observation manipulations and action manipulations. Zhang et al. [33] proposed SA-MDP, which provides a theoretical foundation for modeling state adversarial attacks within MDPs. Oikarinen et al. [13] developed RADIAL-RL, a general framework for training RL agents to enhance resilience against adversarial attacks, and introduced Greedy Worst-Case Reward as a new evaluation metric for agent robustness.

III. System Model

A. Deep Reinforcement Learning

DRL problems are formalized as Markov Decision Processes (MDPs), which provide a theoretical framework for modeling sequential decision-making in unknown environments [34]. An MDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, R, \mathcal{P}, \gamma \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ assigns a scalar reward to each state-action pair, while the transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the probability of transitioning between states given an action. The discount factor $\gamma \in [0, 1)$ governs the weight of future rewards. A stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ defines a probability distribution over actions given a state, where $a \sim \pi(\cdot \mid s)$ denotes sampling an action a in state s. At each time step t, the agent selects an action $a_t \sim \pi(\cdot \mid s_t)$ based on the current state s_t. The environment returns a reward $R(s_t, a_t)$ and transitions to the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$. The agent's objective is to learn a policy π that maximizes the expected discounted return:

$$J(\pi) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{a\sim\pi, s\sim\mathcal{P}} \left[R(s_t, a_t) \right]. \tag{1}$$

To find the optimal policy, value-based reinforcement learning methods are widely used to derive optimal policies. These methods are advantageous due to their ability to efficiently estimate long-term rewards through iterative value estimation, enabling effective decision-making in complex environments. The state value function is defined as $V^{\pi}(s_t) = \mathbb{E}_{a\sim\pi, s\sim\mathcal{P}} \left[\sum_{k=0}^{\infty} \gamma^k R(s_{t+k}, a_{t+k}) \right]$ which represents the expected discounted return starting from state s_t under policy π. Similarly, the action-value function (Q-function) that quantifies the expected discounted return for selecting action a_t in state s_t under policy π is given by $Q^{\pi}(s_t, a_t) = R(s_t, a_t) + \mathbb{E}_{s_{t+1}\sim\mathcal{P}} \left[V^{\pi}(s_{t+1}) \right]$. Policy updates rely on the value function or Q-function. For instance, in Deep Q-Networks (DQN) [35], the policy follows a greedy selection, i.e., $\pi = \arg\max_a Q^{\pi}(s, a)$, and the Q-function is updated as

$$Q_{\phi}^{\pi}(s_t, a_t) \leftarrow Q_{\phi}^{\pi}(s_t, a_t) + l \left[R(s_t, a_t) + \gamma Q_{\phi}^{\pi}(s_{t+1}, a_{t+1}) - Q_{\phi}^{\pi}(s_t, a_t) \right], \tag{2}$$

where l is the learning rate, and Q_{ϕ}^{π} is the Q-function parameterized by a deep neural network with parameters ϕ.

A fundamental aspect of policy optimization is the theoretical bound on the difference in expected returns between two policies, which provides critical insights into policy improvement guarantees and stability. Let $d^{\pi}(s) = (1-\gamma) \sum_{k=0}^{\infty} \gamma^k P(s_k = s \mid \pi)$ denote the discounted future state distribution of all possible trajectories starting from state s under π. Constrained Policy Optimization (CPO) [36] establishes the relationship between the performance difference of two policies and their state-action visitation dynamics under a reward function. Specifically, CPO provides the following bounds for the performance difference:

$$D_{\pi,f}^{-}(\pi') \leq J(\pi') - J(\pi) \leq D_{\pi,f}^{+}(\pi'), \tag{3}$$

where $D_{\pi,f}^{\pm}(\pi') = \frac{L_{\pi,f}(\pi')}{1-\gamma} \pm \frac{2\gamma\xi_f^{\pi'}}{(1-\gamma)^2} \mathbb{E}_{s\sim d^{\pi}} \left[\mathrm{D}_{\mathrm{TV}}(\pi \parallel \pi')[s] \right]$. For any function $f : \mathcal{S} \to \mathcal{R}$ and policies π, π', $L_{\pi,f}(\pi')$, and $\xi_f^{\pi'}$ are given by the following definitions:

$$L_{\pi,f}(\pi') = \mathbb{E}_{s\sim d^{\pi}, a\sim\pi, s'\sim\mathcal{P}} \left[\left(\frac{\pi'(a \mid s)}{\pi(a \mid s)} - 1 \right) \delta_f(s, a, s') \right], \tag{4}$$

$$\xi_f^{\pi'} = \max_s \left| \mathbb{E}_{a\sim\pi', s'\sim\mathcal{P}} \left[\delta_f(s, a, s') \right] \right|, \tag{5}$$

where $\delta_f(s, a, s') = R(s, a, s') + \gamma f(s') - f(s)$.

B. White-box Adversarial Attack

White-box adversarial attacks are common methods for evaluating the vulnerability of deep neural networks (DNNs) by leveraging gradient-based perturbations to manipulate input samples [37]. To be Specific, given an input sample x with corresponding ground truth label y, a DNN is defined as a function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces, respectively. The objective of an adversarial attack is to introduce a minimal perturbation η to x, such that the model produces an incorrect prediction for the perturbed input x^*. This can be formulated as a constrained optimization problem:

$$\arg\max_{x^*} J\left(x^*, y\right), \quad \text{s.t. } \left\|\eta\right\|_p \leq \epsilon, \tag{6}$$

where $x^* = x + \eta$ represents the adversarial example, η denotes the adversarial perturbation, $J\left(\cdot,\cdot\right)$ is the loss function, typically mean squared error or cross-entropy. The constraint $\left\|\eta\right\|_p \leq \epsilon$ ensures that the perturbation magnitude remains within a predefined threshold $\epsilon \in (0, \infty)$, where $\left\|\cdot\right\|_p$ denotes the L_p norm. The choice of $\left\|\cdot\right\|_p$ depends on the application: the L_∞ norm is commonly used for image-based tasks (e.g., Arcade Learning Environment [38]), whereas the L_2 norm is often preferred for physical-state-based environments (e.g., Safety Gymnasium environment [16]).

To solve the constrained optimization problem in Eq. 6, FGSM [18] approximates the loss function linearly and generates adversarial examples using a single-step update:

$$x^* = x + \epsilon \cdot \operatorname{sign}\left(\nabla_x J\left(x, y\right)\right), \tag{7}$$

where $\operatorname{sign}\left(\cdot\right)$ represents the sign function. The perturbation in Eq. 7 satisfies the L_∞ norm constraint. For the L_2 norm-setting, ϵ is adjusted as $\epsilon / \left\|\nabla_x J\left(x, y\right)\right\|_2$ to ensure the perturbation magnitude remains within the constraint. The PGD [19] attack extends FGSM by applying iterative gradient updates with a step size δ while projecting the perturbed input back into the feasible ϵ-ball:

$$x^*_{n+1} = x^*_n + \delta \cdot \operatorname{sign}\left(\nabla_x J\left(x^*_n, y\right)\right), \tag{8}$$

where $n = 0, 1, \ldots, N$, and N denotes the number of iterations, The initial perturbation is set as $x^*_0 = x + \varepsilon \cdot \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$, where ε is a scaling factor, and $\mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$ represents a multivariate standard normal distribution. To enforce the constraint in Eq. 6, the step size is typically set to $\delta = \epsilon / N$.

IV. Theoretical Analysis of Attack Success Conditions

A. Adversarial Victim-Dynamics Markov Decision Process

We formalize the execution of DRL agents under adversarial attacks by establishing an Adversarial Victim Dynamics Markov Decision Process (AVD-MDP). The AVD-MDP is defined as a tuple:

$$\left\langle \mathcal{S}, \mathcal{A}, \mathcal{B}, \mu, R, \mathcal{P}, \gamma \right\rangle, \tag{9}$$

where $\mathcal{B}$ denotes the space of allowable adversarial perturbations, and $\mu\left(\cdot \mid s\right)$ represents the policy of the victim agent (i.e., the agent under attack). The remaining elements follow the standard MDP formulation introduced in Section III-A. Let $\nu\left(\cdot \mid s, \mu\right)$ denote the adversarial policy (i.e., adversarial agent's policy), representing the attacker's strategy under a white-box setting, where the attacker has full access to the victim's policy μ and generates adversarial perturbations based on both the current state s and μ. Notably, $\nu\left(\cdot \mid s, \mu\right)$ is not necessarily a policy in the conventional DRL sense but can be derived by solving the constrained optimization problem in Eq. 6, using methods such as FGSM or PGD. At each time step t, the adversarial policy samples a perturbation $\eta_t \sim \nu\left(\cdot \mid s_t, \mu\right)$ which is then applied to the victim agent's observation. The victim agent selects an action according to its policy

$$a_t \sim \mu\left(\cdot \mid s_t + \eta_t\right), \tag{10}$$

where $s_t + \eta_t$ is the perturbed state.The single attack process is independent of the environment dynamics, forming a one-step sequential decision process. To formalize adversarial attacks in AVD-MDP, we introduce the following definition:

Definition 1. (Single-Step Adversarial Attack) *Let $\mu\left(\cdot \mid s\right)$ denote the victim agent's policy and $\nu\left(\cdot \mid s, \mu\right)$ represents the adversarial policy. We define a single-step attack $\mu \oplus \nu\left(\cdot \mid s\right)$ from the adversarial agent to the victim agent under state s as:*

$$\mu \oplus \nu\left(\cdot \mid s\right) = \mu\left(\cdot \mid s + \eta\right), \eta \sim \nu\left(\cdot \mid s, \mu\right). \tag{11}$$

In AVD-MDP, adversarial attacks occur after the victim agent has fully converged in the environment, achieving optimal or near-optimal returns in attack-free rollouts. The goal of adversarial agents is to degrade the victim agent's performance by minimizing its expected return:

$$J\left(\nu\right) = \min_{\nu} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{a \sim \mu \oplus \nu, s \sim \mathcal{P}}\left[R\left(s_t, a_t\right)\right], \tag{12}$$

where $R\left(s_t, a_t\right)$ represents the reward function and γ is the discount factor. The adversarial attack modifies the victim agent's state-action trajectory, leading to a distinction between attack-free and attacked rollouts:

$$\underbrace{s_0 \xrightarrow{\mu} a_0 \xrightarrow{\mathcal{P}} s_1 \xrightarrow{\mu} \ldots \xrightarrow{\mathcal{P}} s_t \xrightarrow{\mu} a_t \xrightarrow{\mathcal{P}} \ldots}_{\text{attack-free rollout}}$$

$$\underbrace{s_0 \xrightarrow{\nu} \eta_0 \xrightarrow{\mu} a_0 \xrightarrow{\mathcal{P}} s_1 \xrightarrow{\nu} \ldots \xrightarrow{\mathcal{P}} s_t \xrightarrow{\nu} \eta_t \xrightarrow{\mu} a_t \xrightarrow{\mathcal{P}} \ldots}_{\text{attacked rollout}}$$

Since the victim agent follows a fixed policy μ throughout the attacked rollout, it can be regarded as part of the environment's dynamics from the adversarial agent's perspective. However, unlike typical environmental transitions, this component is fully observable and known to the adversarial agent. We define the adversarial agent's value function and Q-function as the victim agent's expected return under the attacked rollout distribution:

Definition 2. (Value and Q-functions of the Adversarial Agent) *In AVD-MDP, the adversarial agent's value function $V^{\mu \oplus \nu}$ and Q-function $Q^{\mu \oplus \nu}$ with victim agent's policy μ and adversarial policy ν are defined as:*

$$V^{\mu \oplus \nu}\left(s_t\right) = \mathbb{E}_{a \sim \mu \oplus \nu, s \sim \mathcal{P}}\left[\sum_{k=0}^{\infty} \gamma^k R\left(s_{t+k}, a_{t+k}\right)\right], \tag{13}$$

$$Q^{\mu \oplus \nu}\left(s_t, a_t\right) = R\left(s_t, a_t\right) + \mathbb{E}_{s_{t+1} \sim \mathcal{P}}\left[V^{\mu \oplus \nu}\left(s_{t+1}\right)\right], \tag{14}$$

where $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$, $\eta_t \in \mathcal{B}$ are state, action, and adversarial perturbation, respectively.

AVD-MDP reformulates the adversarial attack problem as a single-agent MDP, enabling the adversary to systematically optimize perturbations over extended horizons. Unlike existing works [32], [33], which primarily target individual actions or states, AVD-MDP incorporates the victim agent's policy into the environment's dynamics and directly minimizes the victim agent's expected return. This modeling approach closely mirrors real-world safety testing scenarios, as the expected return serves as a direct measure of the victim agent's performance in its designated task.

B. Analysis of Attack Success Conditions

In real-world scenarios, a successful attack undermines the victim agent's ability to select optimal actions, resulting in measurable performance degradation within the task environment. This effect is captured by a reduction in the victim agent's expected return when subjected to adversarial perturbations. For example, in autonomous mobile robots, a successful attack may induce collisions with obstacles or navigation failures. Formally, adversarial perturbations manifest as a decrease in the victim agent's value function under attack for an arbitrary state:

$$V^{\mu}(s) - V^{\mu\oplus\nu}(s) \geq 0, \text{ for } \forall s \in \mathcal{S}, \tag{15}$$

To formalize the requirements for a successful attack, we present the following theorem, which establishes a necessary condition for attack success.

Theorem 1. (Necessary Condition for Attack Success) *Given a victim agent's policy $\mu(\cdot \mid s)$, and an adversarial policy $\nu(\cdot \mid s, \mu)$, a necessary condition for a successful attack is given by:*

$$\delta(\mu,\nu) \leq \frac{2\gamma R_{\max}}{1-\gamma} \mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s], \tag{16}$$

where $\delta(\mu,\nu) = \sum_s d^{\mu\oplus\nu}(s) \sum_a [\mu\oplus\nu(a \mid s) - \mu(a \mid s)] \cdot R(s,a)$ denotes the expected reward difference, $R_{\max} = \max_{s,a\sim\mu} |R(s,a)|$ denotes the maximum absolute value of the reward received by the victim agent during attack-free rollouts, $\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s]$ represents the total variation distance, and γ is the discount factor.

Proof. We analyze the discrepancy in expected returns between the victim agent's policy before and after the attack. Our derivation follows the part of prior work on bounding value function differences but is specifically adapted to adversarial attack settings. We begin by considering an upper bound on the value function difference between the victim's original and attacked policies. Building upon the theoretical results on policy performance differences outlined in Eq. 3, we derive the following bounds:

$$\begin{aligned}
&V^{\mu}(s) - V^{\mu\oplus\nu}(s) \\
&\leq \underbrace{\frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\mu\oplus\nu},a\sim\mu\oplus\nu}\left[\left(\frac{\mu(a \mid s)}{\mu\oplus\nu(a \mid s)} - 1\right) R(s,a)\right]}_{\text{first term of r.h.s.}} \\
&+ \underbrace{\frac{2\gamma}{(1-\gamma)^2}\max_{s,a\sim\mu}|R(s,a)|\,\mathbb{E}_{s\sim d^{\mu\oplus\nu}}\left[\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s]\right]}_{\text{second term of r.h.s.}},
\end{aligned}$$

where the first term measures the direct impact of the policy shift on expected rewards, while the second term quantifies the uncertainty introduced by policy divergence. We assume that the reward function is independent of the next state, i.e., $R: \mathcal{S}\times\mathcal{A} \to \mathcal{R}$, and assume function $f: \mathcal{S} \to \{0\}$ to simplify the derivation in Eq. 3. Under these assumptions, we derive the first term of the right-hand side as

$$\begin{aligned}
&\frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\mu\oplus\nu},a\sim\mu\oplus\nu}\left[\left(\frac{\mu(a \mid s)}{\mu\oplus\nu(a \mid s)} - 1\right) R(s,a)\right] \\
&= \frac{1}{1-\gamma}\sum_s d^{\mu\oplus\nu}(s)\,\mathbb{E}_{a\sim\mu\oplus\nu}\left[\left(\frac{\mu(a \mid s)}{\mu\oplus\nu(a \mid s)} - 1\right) R(s,a)\right] \\
&= \frac{1}{1-\gamma}\sum_s d^{\mu\oplus\nu}(s)\sum_a [\mu(a \mid s) - \mu\oplus\nu(a \mid s)]\, R(s,a) \\
&= -\frac{1}{1-\gamma}\delta(\mu,\nu)
\end{aligned}$$

We then bound the second term of the right-hand side as

$$\begin{aligned}
&\frac{2\gamma}{(1-\gamma)^2}\max_{s,a\sim\mu}|R(s,a)|\,\mathbb{E}_{s\sim d^{\mu\oplus\nu}}\left[\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s]\right] \\
&= \frac{2\gamma}{1-\gamma}\max_{s,a\sim\mu}|R(s,a)|\sum_{k=0}^{\infty}\gamma^k P(s_k = s \mid \mu\oplus\nu)\,\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s] \\
&\leq \frac{2\gamma}{(1-\gamma)^2}\max_{s,a\sim\mu}|R(s,a)|\,\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s].
\end{aligned}$$

Since the victim policy $\mu(\cdot \mid s)$ is fixed, $R_{\max}$ remains a constant. We can derive an upper bound on the change in value function under pre- and post-attack:

$$\begin{aligned}
&V^{\mu}(s) - V^{\mu\oplus\nu}(s) \\
&\leq \frac{1}{1-\gamma}\left[\frac{2\gamma R_{\max}}{1-\gamma}\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s] - \delta(\mu,\nu)\right].
\end{aligned}$$

In AVD-MDP, adversarial attacks aim to reduce the expected return of the victim agent, satisfying $V^{\mu}(s) - V^{\mu\oplus\nu}(s) \geq 0$. Thus, we can obtain a necessary condition for a successful attack:

$$\frac{1}{1-\gamma}\left[\frac{2\gamma R_{\max}}{1-\gamma}\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s] - \delta(\mu,\nu)\right] \geq 0.$$

Then, we rearrange terms, yields

$$\delta(\mu,\nu) \leq \frac{2\gamma R_{\max}}{1-\gamma}\mathrm{D_{TV}}(\mu \parallel \mu\oplus\nu)[s].$$

This concludes the proof. □

Theorem 1 establishes a necessary condition for adversarial attack success by relating the expected change in the

reward distribution, quantified by $\delta(\mu, \nu)$, to the total variation distance $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$. Specifically, $\delta(\mu, \nu)$ quantifies the expected change in the reward distribution resulting from adversarial modifications to the victim's action distribution, averaged over the state distribution during the attacked rollout. **Theorem 1** reveals that the expected reward difference $\delta(\mu, \nu)$ is upper-bounded by the total variation distance between the original victim's policies and the perturbed ones. Notably, $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$ quantifies the divergence between the victim agent's original and attacked policy distributions at each state, which in turn measures the degree of policy manipulation by the adversarial agent. This theorem formalizes the trade-off between attack effectiveness and the stealthiness of perturbations: only attacks that induce sufficiently small shifts in the policy, those that result in limited total variation divergence, can remain practically relevant by ensuring the perturbations are subtle and difficult to detect in realistic scenarios. These results validate a key insight: perturbations with higher stealthiness—quantified by smaller $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$—are of greater practical significance. When the perturbation constraint ϵ in Eq. 6 is relaxed, $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$ increases, thus loosening the necessary conditions for a successful attack and increasing the attack's success rate. However, such attacks are less meaningful in real-world scenarios, as larger perturbations are more detectable and less realistic.

To further refine our analysis, we explore a sufficient condition for a successful attack and establish a necessary and sufficient condition based on **Theorem 1**.

Theorem 2. (Necessary and Sufficient Condition for Attack Success) *Given a victim agent's policy $\mu(\cdot \mid s)$, and an adversarial agent's policy $\nu(\cdot \mid s, \mu)$, let $\delta(\mu, \nu) = \sum_s d^{\mu \oplus \nu}(s) \sum_a [\mu \oplus \nu(a \mid s) - \mu(a \mid s)] R(s, a)$, a necessary and sufficient condition to successfully attack the victim agent is given by*

$$\delta(\mu, \nu) \leq -\frac{2\gamma R_{\max}}{1-\gamma} \max_s P(s \mid \mu \oplus \nu) \, \mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s] \tag{17}$$

where $\max_s P(s \mid \mu \oplus \nu)$ is the upper bound of the state-visiting probability during the attack.

Proof. The proof proceeds similarly to Theorem 1, utilizing the lower bounds on expected returns for two arbitrary policies as established in Eq. 3. We have:

$$\begin{aligned} & V^{\mu}(s) - V^{\mu \oplus \nu}(s) \\ & \geq \underbrace{\frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\mu \oplus \nu}, a \sim \mu \oplus \nu} \left[\left(\frac{\mu(a \mid s)}{\mu \oplus \nu(a \mid s)} - 1 \right) R(s, a) \right]}_{\text{first term of r.h.s.}} \\ & \underbrace{- \frac{2\gamma}{(1-\gamma)^2} \max_{s, a \sim \mu} |R(s, a)| \, \mathbb{E}_{s \sim d^{\mu \oplus \nu}} \left[\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s] \right]}_{\text{second term of r.h.s.}} \end{aligned}$$

Assuming the reward function $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ is independent of the next state, we derive from the first right-hand terms:

$$\begin{aligned} & \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\mu \oplus \nu}, a \sim \mu \oplus \nu} \left[\left(\frac{\mu(a \mid s)}{\mu \oplus \nu(a \mid s)} - 1 \right) R(s, a) \right] \\ & = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\mu \oplus \nu}, a \sim \mu \oplus \nu} \left[\left(\frac{\mu(a \mid s)}{\mu \oplus \nu(a \mid s)} - 1 \right) R(s, a) \right] \\ & = \frac{1}{1-\gamma} \sum_s d^{\mu \oplus \nu}(s) \sum_a [\mu(a \mid s) - \mu \oplus \nu(a \mid s)] R(s, a) \\ & = -\frac{1}{1-\gamma} \delta(\mu, \nu). \end{aligned}$$

We then derive a lower bound of the second right-hand term:

$$\begin{aligned} & - \frac{2\gamma}{(1-\gamma)^2} \max_{s, a \sim \mu} |R(s, a)| \, \mathbb{E}_{s \sim d^{\mu \oplus \nu}} \left[\mathrm{D_{TV}}(\mu \oplus \nu \parallel \mu)[s] \right] \\ & = -\frac{2\gamma}{1-\gamma^{s, a \sim \mu}} \max |R(s, a)| \sum_{k=0}^{\infty} \gamma^k P(s_k = s \,|\, \mu \oplus \nu) \, \mathrm{D_{TV}}(\mu \| \mu \oplus \nu)[s] \\ & \geq -\frac{2\gamma}{(1-\gamma)^2} \max_{s, a \sim \mu} |R(s, a)| \max_s P(s \mid \mu \oplus \nu) \, \mathrm{D_{TV}}(\mu \| \mu \oplus \nu)[s]. \end{aligned}$$

By combining both terms, we can derive a lower bound on the change in value function under pre- and post-attack:

$$\begin{aligned} & V^{\mu}(s) - V^{\mu \oplus \nu}(s) \\ & \geq -\frac{1}{1-\gamma} [\delta(\mu, \nu) + \frac{2\gamma R_{\max}}{1-\gamma} \max_s P(s \mid \mu \oplus \nu) \cdot \\ & \mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]]. \end{aligned}$$

Note that when the lower bound is non-negative, the adversarial policy executes a successful attack. Thus, we can derive a sufficient condition for a successful attack:

$$\delta(\mu, \nu) \leq -\frac{2\gamma R_{\max}}{1-\gamma} \max_s P(s \mid \mu \oplus \nu) \, \mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$$

where $R_{\max} = \max_{s, a \sim \mu} |R(s, a)|$. When Eq. 17 is satisfied, Eq. 16 also holds. Therefore, it constitutes a necessary and sufficient condition for the success of the attack.

□

Theorem 2 builds upon **Theorem 1** by establishing a necessary and sufficient condition for attack success. This result formalizes the requirement for attack effectiveness: the expected reward difference $\delta(\mu, \nu)$ must not exceed a threshold jointly determined by the maximum state visitation probability $\max_s P(s|\mu \oplus \nu)$ and the total variation distance $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$. **Theorem 2** highlights that successful perturbation strategies must induce dispersion in the state visitation distribution of the victim agent. Specifically, under a fixed perturbation constraint ϵ, the total variation distance $\mathrm{D_{TV}}(\mu \parallel \mu \oplus \nu)[s]$ remains constant. Thus, to further increase attack effectiveness, one must minimize $\max_s P(s|\mu \oplus \nu)$. Reducing this maximum state-visiting probability forces the victim agent's state distribution to become more dispersed, thereby raising the upper bound of $\delta(\mu, \nu)$. This dispersion prevents the victim agent from focusing on specific states, undermining its ability to maintain an optimal policy. By exploiting vulnerabilities across a broader range of states, the adversarial policy increases the attack success rate.

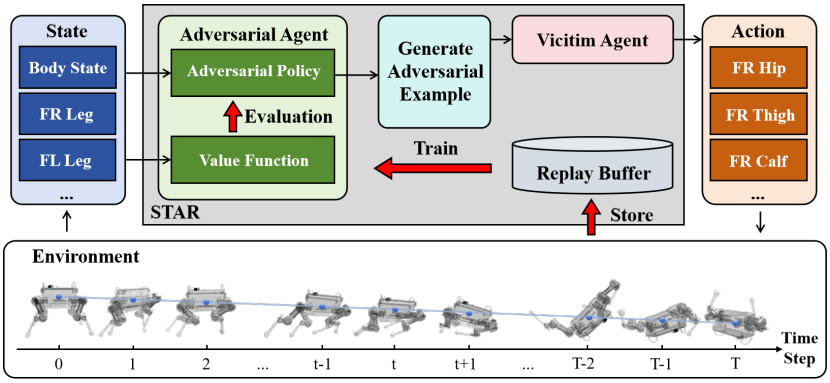


Fig. 2: The workflow of STAR.

V. The STAR Framework

To develop a practical attack algorithm, we analyze the necessary and sufficient condition for successful attacks in Section IV and derive the following insights:

- **Stealthiness of Perturbations**: The adversarial policy must carefully balance between reducing the victim's expected return and maintaining subtle, constrained perturbations. The expected reward difference of the victim policy is upper-bounded by the total variation distance between the original and perturbed policy distributions. This implies a trade-off between attack effectiveness and stealthiness: perturbations that result in smaller total variation differences in victim policy are less detectable and therefore more practical.
- **Dispersion in State Visitation**: The adversarial policy can increase the attack success rate by dispersing the victim agent's state visitation distribution. To achieve this, the adversarial policy should strategically redistribute visitation probabilities, preventing the victim agent from concentrating its behavior on specific states. This dispersed approach not only makes it harder for the victim agent to maintain an optimal policy but also ensures more effective utilization of the perturbation budget under the same constraints.

To enforce stealthiness and dispersion, we propose a DRL-based white-box attack method called **S**elective **ST**ate-**A**ware **R**einforcement adversarial attack (STAR). STAR employs a soft mask-based state-targeting mechanism that adaptively allocates the perturbation budget to critical state dimensions, thereby maximizing attack effectiveness while maintaining stealthiness by minimizing unnecessary perturbations on redundant dimensions. Furthermore, it optimizes an information-theoretic objective that maximizes mutual information between adversarial perturbations, environmental states, and victim actions to ensure a dispersed state-visitation distribution for the victim agent under attack. By integrating these strategies, STAR effectively induces the victim agent into vulnerable states while concentrating perturbations on critical dimensions, maintaining a stealthy attack through carefully constrained perturbations. As illustrated in Figure 2, STAR first extracts state information from the environment (e.g., robot body state and front right/left leg states). The adversarial agent, consisting of an adversarial policy and a value function, evaluates and generates adversarial examples targeting the victim agent. These perturbations are then injected into the victim agent, systematically disrupting its decision-making process and consequently inducing sub-optimal action selections. STAR employs a reinforcement learning paradigm with an experience replay mechanism, enabling continuous optimization of the adversarial policy and value function during training.

A. Optimization Objective

The stealthiness of perturbations necessitates gradient-based attack methods, where sign gradients remain an optimal choice. This is because sign gradients maintain the maximum allowable perturbation magnitude in the direction of the steepest descent, ensuring efficient exploitation of the perturbation budget. As a result, STAR applies perturbations in the direction of the sign gradient to achieve an optimal balance between attack effectiveness and stealthiness. Moreover, motivated by the experimental results in Section I, we propose a soft mask mechanism to further enhance attack efficacy. Our preliminary study on a four-wheeled robot navigation task demonstrates that focusing perturbations on critical state dimensions significantly increases attack impact, even under the same perturbation budget. These results reveal that agents are substantially more vulnerable to targeted, state-aware adversarial interference in comparison with state-agnostic perturbations. To exploit this vulnerability, the soft mask mechanism adaptively allocates the perturbation budget based on each state dimension's importance, thereby maximizing attack effectiveness. Formally, STAR incorporates a soft mask function defined as $M_{\text{soft}}(s) = \beta M(s) + (1-\beta)(1-M(s))$ where $M : \mathcal{S} \to \{0,1\}$ is a binary mask identifying the critical state dimensions, and $\beta \in (0,1)$ is an interpolation factor controlling the trade-off between critical and redundant dimensions. The adversarial example generated by the adversarial policy is then formulated as:

$$\nu(\cdot \mid s, \mu) = \epsilon \cdot M_{\text{soft}}(s) \cdot \text{sign}\left(\nabla_s J\left(s', a\right)\right), \tag{18}$$

where $s' = s + \varepsilon \cdot \mathcal{N}(\mathbf{0}, \mathbf{I})$, $a \sim \mu(\cdot \mid s)$, $\varepsilon \in (0,1]$ is the scaling factor, and $\mathcal{N}(\mathbf{0}, \mathbf{I})$ represents a multivariate standard normal distribution that introduces small perturbations to prevent gradient vanishing. Other notations retain their definitions in Eq. 7 and Eq. 8. As shown in Figure 3,

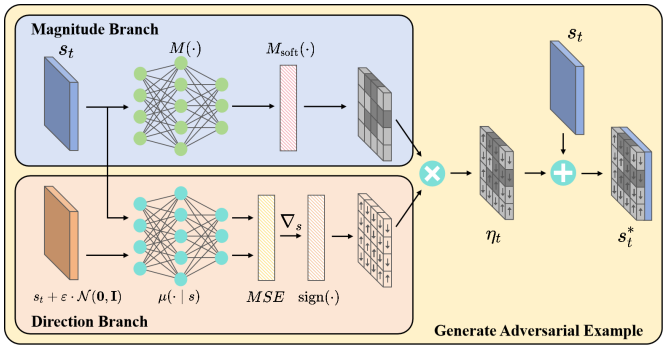


Fig. 3: The Adversarial Example Generation Framework of STAR.

adversarial example generation follows a two-branch structure: the magnitude branch and the direction branch. The magnitude branch employs a mask function and an interpolation mechanism to generate a soft mask, dynamically adjusting the perturbation magnitudes while prioritizing critical state dimensions. The direction branch calculates the perturbation direction using Gaussian noise and sign gradients, ensuring effective perturbations along the steepest descent direction. Finally, the outputs from both branches are combined to generate adversarial perturbations, which are applied to the original state.

The dispersion in state visitation indicates that the optimization objective of STAR necessitates an additional term in Eq. 12, aiming to strategically diversify the state-visitation distribution of the victim agent. This is formalized using information-theoretic concepts. Let $I(\eta; s)$ represent the mutual information between adversarial perturbations η and states s, which quantifies the shared information between perturbations and environmental states. We define $I(\eta; a \mid s)$ as the conditional mutual information between adversarial perturbations η and the victim agent's actions a in the state s. By maximizing $I(\eta; s)$, we ensure that perturbations are closely aligned with current states that have the most significant impact on the victim agent's policy. Simultaneously, maximizing $I(\eta; a \mid s)$ guarantees that the perturbations effectively influence the victim agent's actions based on the current states, thereby disrupting its ability to follow an optimal policy. The combined maximization of total mutual information $I(\eta; s) + I(\eta; a \mid s)$ forces the victim agent to deviate from its attack-free state-visitation patterns, leading to a more dispersed distribution and hindering its capacity to maintain optimal performance.

To further enhance the optimization objective of STAR, we minimize the conditional entropy $\mathcal{H}(a|s)$, which reduces the uncertainty in the victim agent's action selection a for the given state s. This strengthens dependencies between states and actions, thereby constraining the victim agent's policy to a more focused region within the action space. In summary, STAR maximizes the total mutual information, which could

Algorithm 1 STAR Training Procedure

Input: Victim agent's policy $\mu(\cdot \mid s)$, batch size N, entropy weight coefficient α, discount factor γ.

Random initialization: Mask function $M(\cdot)$ with θ^M and value function $V^{\mu \oplus \nu}(\cdot)$ with θ^V, and replay buffer $\mathcal{D}$.
for each episode **do**
  Initialize state s_0 and $T \leftarrow 0$
  $\eta_0 \sim p(s_0)$
  $a_0 \sim \mu(\cdot \mid s_0 + \eta_0)$
  for time step t **do**
    Execute a_t, compute reward $r_t = R(s_t, a_t)$, and store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$.
    if s_{t+1} is terminal **then**
      $T \leftarrow t + 1$
      Calculate and store return $\widehat{R}_t$ in $\mathcal{D}$:
      $$\widehat{R}_t = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k} + \gamma^{T-t} V^{\mu \oplus \nu}(s_T)$$
      Calculate and store advantage $\widehat{A}_t$ in $\mathcal{D}$:
      $$\widehat{A}_t = \widehat{R}_t - V^{\mu \oplus \nu}(s_t)$$
    end if
    $a_t \sim \mu \oplus \nu(\cdot \mid s_t)$
  end for
  for each epoch **do**
    Sample a batch of $\left(s_i, a_i, r_i, s_{i+1}, \widehat{R}_i, \widehat{A}_i\right)$ from $\mathcal{D}$
    Update the value function $V^{\mu \oplus \nu}$ by minimizing the loss:
    $$\mathcal{L}\left(V^{\mu \oplus \nu}\right) = \frac{1}{N} \sum_{i=0}^{N} \left[\hat{R}_i - V^{\mu \oplus \nu}(s_i)\right]^2$$
    Update the mask function $M(\cdot)$ by minimizing the objective:
    $$J(\nu) = \frac{1}{N} \sum_{i=0}^{N} \left[\hat{A}_i + \alpha \log\left(\mu(a_i \mid s_i)\, \nu(\eta_i \mid s_i, \mu)\right)\right]$$
  end for
end for

be expressed as:

$$\begin{aligned} & I(\eta; s) + I(\eta; a \mid s) - \mathcal{H}(a; s) \\ &= \mathcal{H}(\eta) - \mathcal{H}(\eta \mid s) + \mathcal{H}(\eta \mid s) - \mathcal{H}(\eta \mid s, a) - \mathcal{H}(a \mid s) \\ &= \mathcal{H}(\eta) - \mathcal{H}(\eta \mid s, a) - \mathcal{H}(a \mid s) \end{aligned} \tag{19}$$

where $\mathcal{H}(\cdot)$ represents the Shannon entropy, $\mathcal{H}(\eta)$ indicates the necessity to maximize the entropy of the adversarial policy's prior distribution. In practice, we adopt a uniform distribution as the prior distribution of the adversarial policy. The terms $-\mathcal{H}(\eta \mid s, a)$ and $-\mathcal{H}(a \mid s)$ imply that, given state s, both the victim agent's and the adversarial agent's policy distributions should exhibit low uncertainty. Consequently, maximizing the objective function presented in Eq. 19 is equivalent to minimize $\mathcal{H}(\eta \mid s, a) + \mathcal{H}(a \mid s)$. Combining the previously stated objective of the adversarial agent from Eq. 12, we derive the objective function of STAR as follows:

$$\begin{aligned} J(\nu) = \min_{\nu} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{a \sim \mu \oplus \nu, s \sim \mathcal{P}} \big[& R(s_t, a_t) \\ & + \alpha \left[\mathcal{H}(\nu(\cdot \mid s_t)) + \mathcal{H}(\mu \oplus \nu(\cdot \mid s_t))\right]\big] \end{aligned} \tag{20}$$

where $\alpha \in (0, 1]$ serves as an entropy weight coefficient that balances two optimization objectives: minimizing the victim agent's return via term $R(s_t, a_t)$ and concentrating attacks through dual entropy regularization terms $\mathcal{H}(\nu(\cdot \mid s_t)) + \mathcal{H}(\mu \oplus \nu(\cdot \mid s_t))$.

B. Training Algorithm

We employ an on-policy reinforcement learning framework integrated with Generalized Advantage Estimation (GAE) [39] to train the STAR adversarial agent. The on-policy framework

is particularly sensitive to subtle variations in the victim agent's behavioral patterns, ensuring that policy updates are consistently derived from the most recent interactions between the attacker and the victim. This approach mitigates the distribution shift issues commonly associated with off-policy methods, which rely on outdated experiences. By leveraging on-distribution samples from current policy trajectories, the on-policy framework facilitates real-time adaptation of perturbation strategies while maintaining update stability. Additionally, STAR utilizes GAE to enhance the efficiency and stability of policy gradient algorithms by providing a more flexible and accurate approximation of the advantage function. The advantage function $\widehat{A}_t$ quantifies the relative improvement of an action compared to the expected return under the current policy. However, accurate estimation of $\widehat{A}_t$ becomes challenging in environments with sparse or noisy rewards. GAE addresses this challenge by introducing a tunable hyperparameter λ, which balances short-term rewards and long-term returns while optimizing the bias-variance trade-off. Specifically, the advantage function $\widehat{A}_t$ at time step t is computed as:

$$\widehat{A}_t = \sum_{k=0}^{T-t-1} (\gamma\lambda)^k \delta_{t+k}, \tag{21}$$

where $\delta_t = r_t + \gamma V^{\mu\oplus\nu}(s_{t+1}) - V^{\mu\oplus\nu}(s_t)$. By setting $\lambda = 0.95$, GAE strikes a balance between bias and variance by incorporating both rewards and value function estimates to provide a stable approximation of the advantage function. This setup is particularly suitable for tasks with long-term dependencies, such as adversarial attacks in robotic manipulation, as it effectively captures the cumulative impact of future rewards without requiring explicit weighting schemes or truncation of temporal difference errors. To quantify state-value expectations under policy $\mu \oplus \nu$, STAR employs a parameterized value function $V^{\mu\oplus\nu}$ that is optimized through minimization of the following mean squared error loss:

$$\mathcal{L}\left(V^{\mu\oplus\nu}\right) = \mathbb{E}_{s_i,\hat{R}_i\sim\mathcal{D}}\left[\hat{R}_i - V^{\mu\oplus\nu}(s_i)\right]^2, \tag{22}$$

where $\widehat{R}_i = \sum_{k=0}^{T-i-1} \gamma^k r_{i+k} + \gamma^{T-i} V^{\mu\oplus\nu}(s_T)$ denotes the expected return and $\mathcal{D}$ represents the replay buffer of on-policy trajectories sampled from the current policy $\mu \oplus \nu$. The mask function $M(s)$ is parameterized by a neural network θ^M. It is optimized by minimizing the objective function in Eq. 20, which incorporates both the advantage and entropy regularization terms:

$$J(\nu) = \mathbb{E}_{s_i,\hat{A}_i\sim\mathcal{D}}\left[\hat{A}_i + \alpha \log\left(\mu(a_i \mid s_i)\,\nu(\eta_i \mid s_i, \mu)\right)\right]. \tag{23}$$

The STAR training algorithm is detailed in Algorithm 1. The training process begins by initializing the mask function $M(\cdot)$ with network parameters θ^M, the value function $V^{\mu\oplus\nu}(\cdot)$ with network parameters θ^V and a replay buffer $\mathcal{D}$ to store transitions (line 1). For each episode, the initial state s_0 is set, and the episode length counter T is initialized to zero (line 3). The adversarial perturbation is first sampled from a uniform distribution $p(s_0)$ to maximize the entropy of the adversarial policy's prior distribution (lines 4-5). During each time step of the episode, the selected action a_t is executed in the environment, yielding a reward r_t and transitioning to the next state s_{t+1}. The transition tuple (s_t, a_t, r_t, s_{t+1}) is stored in the replay buffer $\mathcal{D}$ for subsequent training (line 7). If the episode is not terminal, the adversarial examples are generated based on the current state s_t and the victim agent's policy, after which the victim agent selects an action a_t according to the adversarial example to continue the interaction in the environment (line 13). If the environment reaches a terminal state s_{t+1}, the episode length T is updated. Returns $\widehat{R}_t$ and advantages $\widehat{A}_t$ for all time steps t in the episode are computed and stored in the replay buffer $\mathcal{D}$ (lines 9-11). Training occurs after the collection of trajectories. For each training epoch, a batch of N transitions, returns, and advantages is sampled from the replay buffer $\mathcal{D}$ (line 16). The value function $V^{\mu\oplus\nu}$ is updated by minimizing the mean squared error between the predicted value and the stored returns $\widehat{R}i$ (line 17). The mask function $M(\cdot)$, defined in the adversarial policy in Eq. 18, is optimized by minimizing the objective $J(\nu)$ (line 18).

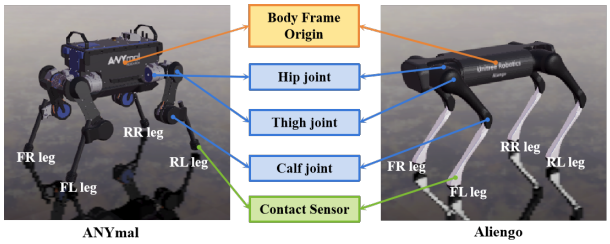


Fig. 4: Aliengo and ANYmal, quadruped robots in the Raisim platform [40].

TABLE I: State and Action Space Definition

State Space (34 dimensions)	
Body Height (m)	0
Body Orientation (rad)	1:3
Joint Angles (rad)	3:15
Body Linear Velocity (m/s)	15:18
Body Angular Velocity (rad/s)	18:21
Joint Velocities (rad/s)	21:33
Action Space (12 dimensions)	
Front Right Leg (Hip, Thigh, Calf) (rad)	0:3
Front Left Leg (Hip, Thigh, Calf) (rad)	3:6
Rear Right Leg (Hip, Thigh, Calf) (rad)	6:9
Rear Left Leg (Hip, Thigh, Calf) (rad)	9:12

VI. Implementation

Emulation Setup. We evaluate quadrupedal locomotion control and adversarial robustness using the RaiSim platform [40], conducting experiments on two commercial quadruped robots: Aliengo and ANYmal, as shown in Figure 4. Both robots feature a floating base with 6 degrees of freedom (3 for position and 3 for orientation) and 12 actuated joints, with each leg consisting of 3 joints (hip, thigh, and calf). The robots are controlled by a Proportional-derivative controller with feedforward torque, where the proportional gain $P = 100.0$ controls the stiffness of position tracking and the derivative gain $D = 1.0$ provides damping to reduce oscillations. Each leg's end-effector has a contact sensor to

detect ground interaction. A fall is defined as any non-foot component making contact with the ground.

In each episode, the robot is initialized 0.5 meters above the ground in a standardized stance. The episode runs for a maximum of 4 seconds (400 control steps) or terminates early if a fall occurs. The controller operates at 100 Hz with a control interval of 10 ms, while the physics simulation runs at 400 Hz with a timestep of 2.5 ms. All experiments are conducted on a server with 4 NVIDIA GeForce RTX 3090 (24GB) GPUs.

Victim and Adversarial Agents. The victim agent is trained using Proximal Policy Optimization (PPO) [41], an on-policy reinforcement learning algorithm that combines trust region optimization with clipped surrogate objectives. It employs a dual-network architecture consisting of an actor network and a critic network. The learning rate is adaptively modulated based on the Kullback-Leibler (KL) divergence between consecutive policy iterations. The network architecture and hyperparameters are summarized in Table II. The victim agent's reward function balances energy efficiency and performance through torque penalties and forward velocity incentives:

$$R_{\text{vic}}(\tau, v_x) = \zeta \sum_{i=1}^{12} \tau_i^2 + \kappa \min(v_x, 4.0), \tag{24}$$

where τ_i represents the torque of the i-th joint, v_x is the forward velocity, $\zeta = -4 \times 10^{-5}$ is the energy efficiency coefficient penalizing the sum of squared torques, and $\kappa = 0.3$ is the forward velocity coefficient encouraging locomotion with an upper bound of 4.0 m/s. Training spans 1×10^5 timesteps for each task, with learning trajectories shown in Figure 5.

The adversarial agent, trained by the STAR framework, consists of an adversarial policy and value function. The masking function within the adversarial policy is parameterized by neural networks, as detailed in Table II. The adversarial reward is defined as the negative of the victim agent's reward to minimize artificial effort introduced by reward shaping:

$$R_{\text{adv}} = -R_{\text{vic}}. \tag{25}$$

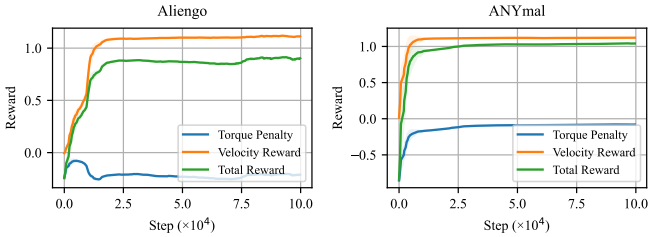
The adversarial agent's training spans 2×10^3 steps, with the trajectory shown in Figure 6.

Performance metrics. We use the following three metrics to compare the performance of STAR and state-of-the-art attack methods.

- **Reward**: The average per-step reward obtained by the victim agent. In reinforcement learning, the reward reflects the feedback from the environment based on the agent's actions. A higher reward indicates that the agent is making decisions that lead to more favorable outcomes as defined by the task objectives.
- **Forward Velocity**: The mean translational velocity of the robot in its forward direction (m/s). This metric measures how efficiently the robot moves forward. A higher forward velocity often indicates smoother and more effective movement.
- **Fall Rate (%)**: A *fall* is defined as any non-foot part of the robot contacting the ground and will result in the end of the episode. The fall rate is calculated as the ratio of *fall* events to the total time steps. This metric evaluates the stability of the agent. A lower fall rate means the robot is better at maintaining its balance.

TABLE II: Hyperparameters

Network Architecture	
Victim Policy Network	2 FC layers, 128 neurons each
Victim Value Network	2 FC layers, 128 neurons each
Adversarial Policy Network	3 FC layers, 64 neurons each
Adversarial Value Network	3 FC layers, 64 neurons each
Victim Agent Hyperparameters	
Clipping parameter	0.2
Discount factor	0.998
Value loss coefficient	0.5
Initial learning rate	5×10^{-4}
Maximum gradient norm	0.5
Batch size	64
Adaptive learning rate KL target	0.01
Adversarial Agent Hyperparameters	
Learning rate	3×10^{-4}
Temperature	1.0
Discount factor	0.998
Entropy weight coefficient	2×10^{-4}
Interpolation factor	0.75
Scaling factor	1×10^{-4}



(a) (b)

Fig. 5: Reward over training steps for two scenarios, showing two reward components (torque penalty and velocity reward) and the total reward. (a) Aliengo locomotion. (b) ANYmal locomotion.

Baselines. To evaluate the performance of STAR, we compare it with the following alternatives. For a fair comparison, given a perturbation budget ϵ, we perform iterative attacks using $N = 10, 20$ steps with step size $\alpha = \epsilon/N$.

- **Random Attack**: A simple baseline that applies uniform random noise as perturbations, serving as a naive benchmark for comparison.
- **FGSM** [18]: The Fast Gradient Sign Method (FGSM) is a single-step attack that perturbs the input in the direction of the gradient's sign, aiming to maximize the loss with minimal computational overhead.
- **DI^2-FGSM** [24]: An extension of FGSM that incorporates input transformations, such as resizing and padding, before computing gradients. This approach introduces input diversity, which has been shown to improve transferability by mitigating overfitting to specific input representations.

TABLE III: Aliengo Locomotion results, reported as mean ± SD over 20 episodes, with best in bold and suboptimal underlined. The ↑ indicates higher value is better, while the ↓ indicates lower value is better.

Method	Reward ↓		Forward Velocity ↓		Fall Rate (%) ↑	
	Raw	Drop (%)	Raw	Drop (%)	Raw	Rise (%)
No Attack	0.941 ± 0.010	0.000	3.777 ± 0.011	0.000	0.464 ± 0.018	0.000
Random	0.927 ± 0.015	1.488	3.754 ± 0.029	0.609	0.491 ± 0.041	0.027
FGSM	0.819 ± 0.033	12.965	3.596 ± 0.059	4.792	0.779 ± 0.094	0.315
DI^2-FGSM	0.674 ± 0.084	28.374	3.404 ± 0.122	9.876	1.225 ± 0.255	0.761
MI-FGSM	0.787 ± 0.053	16.366	3.557 ± 0.088	5.825	0.880 ± 0.130	0.416
NI-FGSM	0.791 ± 0.043	15.940	3.567 ± 0.064	5.560	0.893 ± 0.150	0.429
R+FGSM	0.689 ± 0.053	26.780	3.427 ± 0.096	9.267	1.175 ± 0.163	0.711
PGD	0.701 ± 0.075	25.505	3.425 ± 0.126	9.320	1.172 ± 0.166	0.708
TPGD	0.721 ± 0.063	23.379	3.411 ± 0.151	9.690	1.062 ± 0.120	0.598
EOTPGD	0.685 ± 0.081	27.205	3.407 ± 0.138	9.796	1.132 ± 0.164	0.668
MAD	0.694 ± 0.078	26.249	3.401 ± 0.110	9.954	1.128 ± 0.115	0.664
STAR (Ours)	**0.622 ± 0.079**	**33.900**	**3.268 ± 0.166**	**13.476**	**1.420 ± 0.252**	**0.956**

- **MI-FGSM** [23]: A variant of iterative FGSM that integrates a momentum term to stabilize gradient updates and reduce the likelihood of being trapped in local optima, resulting in more effective adversarial examples.
- **NI-FGSM** [26]: An extension of MI-FGSM that incorporates Nesterov accelerated gradient to refine the update direction by approximating future gradient information.
- **R+FGSM** [42]: A variation of FGSM that introduces a small random perturbation to the input before applying the gradient-based modification. This is intended to reduce sensitivity to local gradient structures, potentially improving robustness in certain scenarios.
- **PGD** [19]: Projected Gradient Descent (PGD) is an iterative extension of FGSM that applies multiple gradient ascent steps, each followed by a projection onto the allowed perturbation space.
- **TPGD** [43]: A variant of PGD in which the cross-entropy loss is replaced with KL divergence loss for adversarial example generation. This alternative loss design is intended to improve attack performance against models trained with label smoothing or other robustness-promoting techniques.
- **EOTPGD** [44]: An extension of PGD that incorporates the Expectation over Transformation (EOT) framework, where random transformations (e.g., rotations, translations) or model variations are applied during each attack iteration. This method aims to generate adversarial examples that maintain effectiveness across a distribution of input variations.
- **MAD** [33]: A RL-specific gradient-based adversarial attack method that maximizes the KL divergence between the victim policy distributions of the original state and the perturbed state, forcing significant changes in the victim agent's behavior.

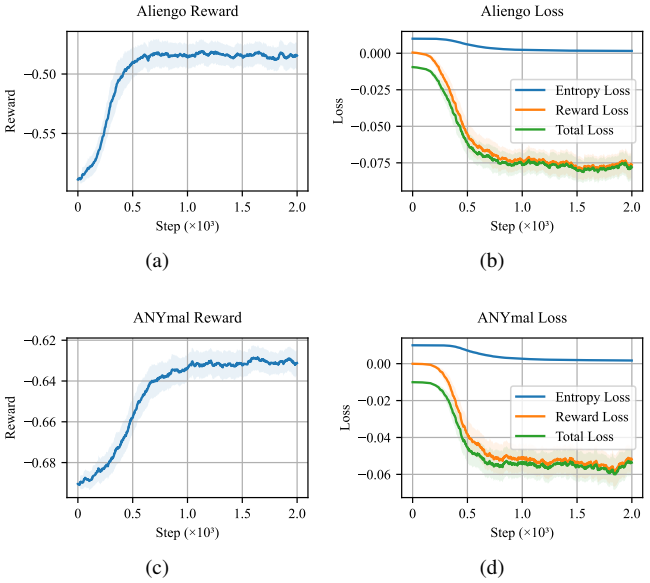


Fig. 6: Training trajectories of the adversarial agent. The left column shows reward trends, and the right column shows losses: entropy loss (blue), reward loss (orange), and total loss (green). (a) Aliengo reward. (b) Aliengo loss. (c) ANYmal reward. (d) ANYmal loss.

VII. Evaluation

A. Comparative Experiments

Tables III and IV present the comparative experimental results of robotic locomotion between the two systems, with each configuration evaluated through 20 independent trials. For iterative methods, we tested the iteration number N = 10, 20 and reported the better results. For each performance metric, we report both the raw values and the relative changes (rise/drop) with respect to the non-attacked baseline, expressed as mean ± standard deviation. Note that the relative changes for reward and forward velocity are calculated multiplicatively, while for fall rate, being a ratio metric itself, we use additive comparison. The optimal method is highlighted in bold, with the suboptimal method denoted by underlining.

Reward. For reward degradation, STAR consistently outperforms all baseline attack methods on both Aliengo and ANYmal locomotion tasks. For Aliengo, STAR induces a 33.900% reduction in reward, exceeding the second-best method, DI²-FGSM, which achieves a 28.374% decrease. Similarly, for ANYmal, STAR achieves a 45.377% reduction, compared to 38.151% for DI²-FGSM. The superior effectiveness of STAR is attributed to its soft mask mechanism, which selectively

TABLE IV: ANYmal Locomotion results, reported as mean ± SD over 20 episodes, with best in bold and suboptimal underlined. The ↑ indicates higher value is better, while the ↓ indicates lower value is better.

Method	Reward ↓		Forward Velocity ↓		Fall Rate (%) ↑	
	Raw	Drop (%)	Raw	Drop (%)	Raw	Rise (%)
No Attack	1.141 ± 0.000	0.000	3.973 ± 0.000	0.000	0.025 ± 0.000	0.000
Random	1.128 ± 0.010	1.382	3.958 ± 0.021	0.397	0.029 ± 0.009	0.004
FGSM	0.947 ± 0.039	20.616	3.607 ± 0.091	9.690	0.330 ± 0.067	0.305
DI^2-FGSM	<u>0.782 ± 0.041</u>	<u>38.151</u>	<u>3.233 ± 0.113</u>	<u>19.592</u>	<u>0.620 ± 0.094</u>	<u>0.595</u>
MI-FGSM	0.875 ± 0.042	28.268	3.447 ± 0.105	13.926	0.451 ± 0.081	0.426
NI-FGSM	0.870 ± 0.039	28.799	3.444 ± 0.091	14.006	0.458 ± 0.077	0.433
R+FGSM	0.794 ± 0.055	36.876	3.262 ± 0.142	18.824	0.594 ± 0.115	0.569
PGD	0.804 ± 0.037	35.813	3.281 ± 0.099	18.321	0.595 ± 0.088	0.570
TPGD	0.826 ± 0.040	33.475	3.343 ± 0.105	16.680	0.532 ± 0.084	0.507
EOTPGD	0.791 ± 0.041	37.194	3.257 ± 0.110	18.957	0.589 ± 0.082	0.564
MAD	0.778 ± 0.036	31.815	3.214 ± 0.113	19.104	0.603 ± 0.105	0.578
STAR (Ours)	**0.714 ± 0.061**	**45.377**	**3.053 ± 0.168**	**24.358**	**0.746 ± 0.129**	**0.721**

perturbs critical state dimensions, and its information-theoretic objective, which encourages dispersed state visitation. These mechanisms directly align with the theoretical goal of maximizing expected return degradation, demonstrating STAR's enhanced capability to compromise the performance of victim agents.

Forward Velocity. For forward velocity reduction, STAR consistently achieves the most substantial impact among all methods. For Aliengo, STAR induces a 13.476% decrease in velocity, outperforming EOTPGD (9.796%) and DI²-FGSM (9.876%). For ANYmal, STAR achieves a reduction of 24.358%, significantly outperforming the suboptimal DI²-FGSM (19.592%). These results demonstrate STAR's effectiveness in destabilizing robotic motion. STAR causes frequent and abrupt decelerations, as the victim policy is persistently perturbed away from efficient behavior patterns. This effect is attributable to STAR's dispersion mechanism, which forces the victim agents' state visitation distribution to cover more vulnerable states. Consequently, the victim agent is impeded from consistently exploiting high-reward and high-velocity trajectories.

Fall Rate. STAR also achieves the largest increase in fall rate, demonstrating its superior effectiveness in inducing robotic failures. For Aliengo, STAR elevates the fall rate to 0.956%, outperforming DI²-FGSM (0.761%) and R+FGSM (0.711%).For ANYmal, STAR increases the fall rate to 0.721%, exceeding DI²-FGSM (0.595%) and R+FGSM (0.569%). Since the fall rate directly reflects catastrophic locomotion failures, these results further demonstrate STAR's enhanced disruptive capability compared to the baseline methods. This is because STAR amplifies the uncertainty in the victim policy specifically within state dimensions that are crucial for maintaining balance. Consequently, victim agents subjected to STAR attacks are unable to recover from minor disturbances, leading to frequent and sudden failures.

B. Behavior Analysis

Figure 7 compares the locomotion of two quadrupedal robots under normal conditions and STAR-induced adversarial perturbations, with motion sequences sampled at 0.1-second intervals over 2 seconds to capture gait stability and disruption. Aliengo (first row) maintains a stable gait without interference, while STAR perturbations (second row) cause deviations at around 1.5 seconds, leading to instability and balance loss by the end of the sequence. Similarly, ANYmal (third row) moves smoothly under normal conditions, whereas STAR (fourth row) triggers earlier destabilization at around 1.2 seconds and collapses by 1.5 seconds. This visual evidence intuitively demonstrates the effectiveness of STAR in inducing instability and disrupting gait coordination.

Figure 8 visualizes the perturbation distribution across observation dimensions using temporal heatmaps. We compare STAR with the optimal baseline DI^2-FGSM. The observation space is divided into five groups, separated by dashed lines in the figure: (i) Body State, including body height, orientation, linear velocities and angular velocities; and four leg groups corresponding to (ii) Front-Right (FR), (iii) Front-Left (FL), (iv) Rear-Right (RR), and (v) Rear-Left (RL) legs, where each leg contains hip, thigh, and calf joint angles and velocities. It can be observed that STAR shows clear perturbation patterns, mainly targeting front leg states, since they directly control stability and balance during forward movement. Notably, these perturbation patterns demonstrate temporal periodicity, which may correlate with the cyclic nature of individual steps in quadrupedal locomotion. This shows STAR can identify both key state variables and important time windows during locomotion. In contrast, DI^2-FGSM applies a more uniform perturbation distribution without selectively targeting key state components. Its perturbation budget is spread across less impactful dimensions.

Figure 9 shows t-SNE visualizations of state distributions under different methods. To verify the effectiveness of the mutual information objective in maximizing state dispersion, we compare the state representations of STAR against baselines. Given the high dimensionality of the state vector (33 dimensions in total), we separated it into two key components for better visualization: the body component (dimensions 0:3 in Table I, including height and orientation) and the leg component (dimensions 3:33 in Table I, including joint angles and velocities). We then applied t-SNE dimensionality reduction to each component separately to project them into 2D space. The visualization clearly demonstrates that STAR achieves broader coverage of the state space compared to other methods. For both body and leg components, STAR's representations (red

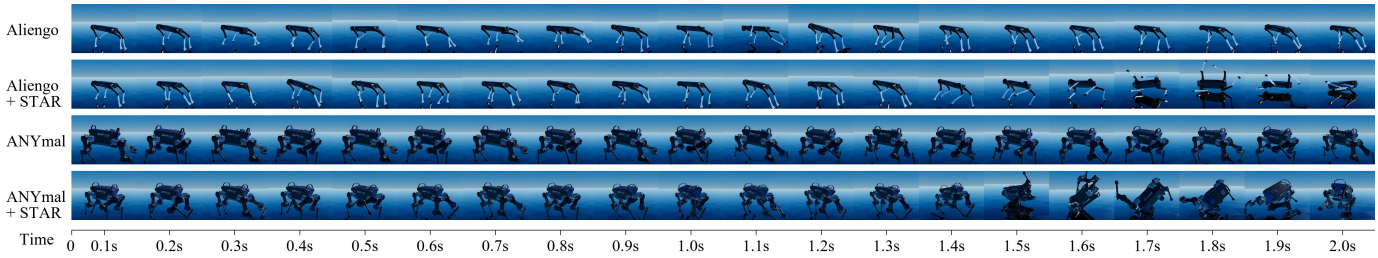


Fig. 7: Comparative motion sequences of quadrupedal robots (Aliengo and ANYmal) under normal and adversarial conditions over a 2-second duration, sampled at 0.1 second intervals.

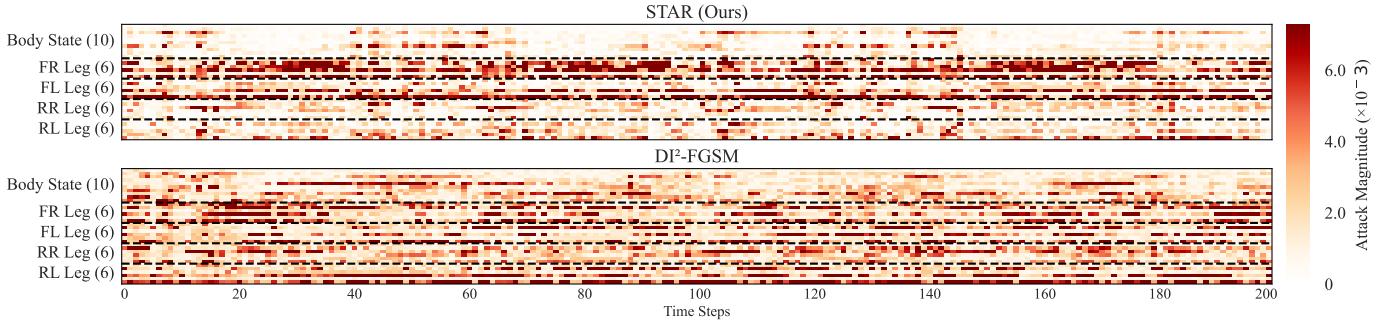
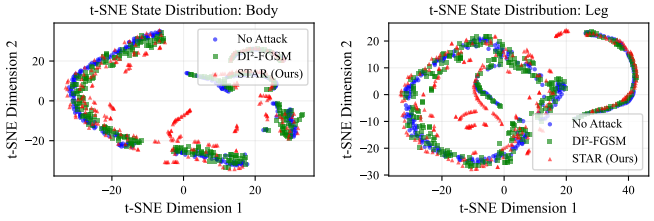
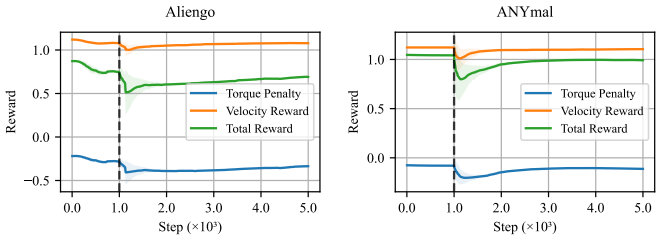


Fig. 8: Temporal visualization of perturbation magnitude comparing STAR (Ours) and DI^2-FGSM over 200 time steps. The y-axis shows Body State (10) comprising body height, orientation, linear and angular velocities; and four legs (FR, FL, RR, RL) each with 6 dimensions for joint angles and velocities. Darker color indicates higher perturbation magnitude.



(a) (b)

Fig. 9: t-SNE Visualization of State Distributions. (a) Body state distribution. (b) Leg state distribution. Different colors represent No Attack (blue), DI²-FGSM (green), and STAR (red).



(a) (b)

Fig. 10: Reward over training steps during adversarial defense, showing two reward components (torque penalty and velocity reward) and the total reward. The performance initially drops due to adversarial attacks but gradually recovers as the victim agent develops robustness. (a) Aliengo. (b) ANYmal.

points) spread into regions that other methods fail to explore. This is particularly noticeable in the body state distribution, where STAR forms distinct clusters at the boundaries. These results indicate that the mutual information objective of STAR successfully induces the victim agent to visit different states, enhancing the perturbation capability of STAR.

C. Post-Defense Robustness

Figure 10 shows the victim agent's training trajectory under STAR adversarial defense, highlighting STAR's potential for enhancing model robustness. The training runs for 4×10^3 steps at a learning rate of 3×10^{-4}, with the victim agent operating under STAR attacks. The adversarial perturbations are constrained within $\epsilon = 0.025$ for ANYmal and $\varepsilon = 0.1$ for Aliengo. When the STAR attack is introduced at 1×10^3 steps, the reward initially drops but recovers as the victim agent adapts to the adversarial attack through policy optimization.

Figure 11 presents box plots comparing the robustness of original and STAR defended victim agents against all baseline attack methods. Each box illustrates the reward distribution over 20 evaluation episodes. For Aliengo, the STAR defended agent consistently achieves median rewards above 0.83 across all attacks, with narrow interquartile ranges indicating stable performance. In contrast, the original agent exhibits significant vulnerability, with median rewards ranging from 0.65 to 0.80 and notably higher variance, especially under DI²-FGSM and EOTPGD attacks. A similar robustness improvement is observed for ANYmal: the STAR-defended agent maintains

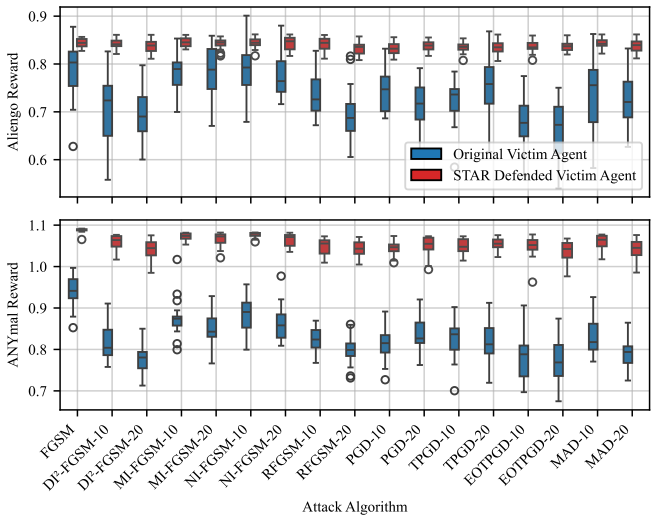


Fig. 11: Performance comparison between original and STAR defended victim agents under different adversarial attacks. For iterative attack methods, the notation algorithm-N denotes N iteration steps.

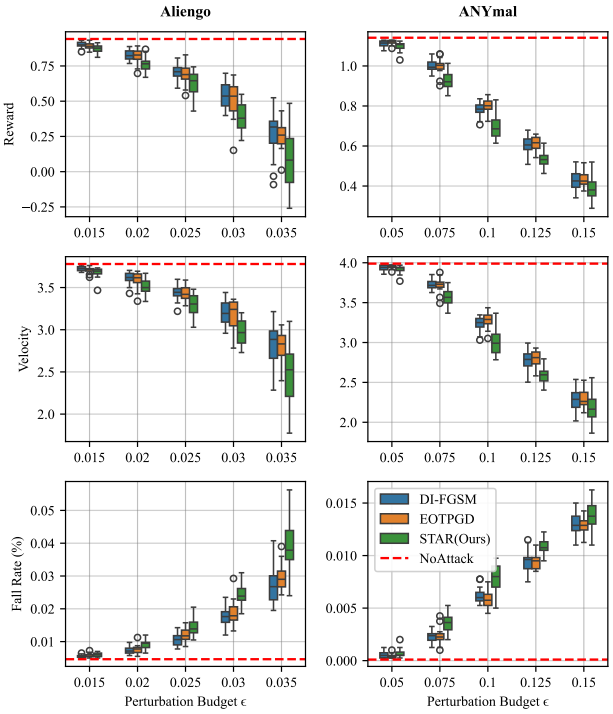


Fig. 12: Performance comparison of different attack methods under varying perturbation budgets ϵ.

median rewards above 1.0 under all attacks, whereas the original agent, except under the single-step FGSM attack, experiences substantial reward degradation (median below 0.95). This improvement is attributed to the more diverse and targeted perturbations generated by STAR, which are not captured by baseline methods. Overall, these results demonstrate that STAR effectively produces adversarial samples that enhance defense robustness. Adversarial training with STAR-generated perturbations enables the defended agent to maintain near-nominal performance even against strong iterative attacks (e.g., DI²-FGSM-20).

D. Ablation Study

Figure 12 shows the impact of perturbation budget ϵ on the attack performance. The perturbation budget ϵ serves as a crucial hyperparameter that controls the magnitude of adversarial perturbations. A larger ϵ allows for more substantial modifications to the input observations, potentially leading to more effective attacks, while a smaller ϵ ensures better imperceptibility. We compare STAR against two suboptimal baselines DI²-FGSM and EOTPGD under different ϵ values across both tasks. As the perturbation budget ϵ increases, all attack methods show enhanced effectiveness, indicated by decreasing rewards and velocities, along with rising failure rates. Our proposed STAR method consistently outperforms baseline approaches under various perturbation budgets, particularly in higher ϵ regions, demonstrating advantages over other baseline methods in terms of reward reduction, velocity decrease, and increased fall rates. This is particularly pronounced in the Aliengo locomotion task under higher perturbation budgets (when $\epsilon = 0.035$). With increased perturbation allowance, our mask-based mechanism more effectively targets identified vulnerable states (e.g., critical leg configurations), resulting in improved attack performance.

Table V presents our ablation study that examines the contributions of STAR's key components: the mutual information (MI) optimization term and the soft mask mechanism. The full STAR framework consistently outperforms all variants across evaluation metrics. Removing either the soft mask or the MI optimization term leads to substantial performance degradation, highlighting the critical impact of both components on attack effectiveness. Specifically, omitting the soft mask mechanism results in a more pronounced reduction not only in task reward (9.65% for Aliengo, 8.54% for ANYmal) but also in forward velocity (3.55% for Aliengo, 4.95% for ANYmal), compared to removing the MI optimization term (task reward drops by 5.14% and 4.34%, forward velocity drops by 1.93% and 2.78%, respectively). These findings indicate that both the soft mask mechanism and the MI optimization term significantly and complementarily improve the effectiveness of STAR attack by adaptively concentrating perturbations on critical state dimensions and promoting a dispersed state-visitation distribution. Removing both mechanisms (STAR w/o MI & Soft Mask) results in the largest performance drop in all metrics: 14.47% less reward drop for Aliengo and 14.98% for ANYmal, fewer forward velocity drop of 5.63% and 8.61%, and a substantial decrease in fall rate. This highlights a synergistic effect, as the combination of MI optimization and soft mask achieves stronger attack effectiveness than either component alone.

VIII. Conclusion

In this paper, we address the challenge of state-aware adversarial attacks in DRL for robotic control systems. Through the proposed AVD-MDP framework, we provide theoretical foundations for analyzing adversarial attacks while consid-

TABLE V: Comparison of attack effectiveness between STAR method, its variants on Aliengo and ANYmal robotic platforms

Method	Aliengo			ANYmal		
	Reward ↓ Raw (Drop %)	Forward Velocity ↓ Raw (Drop %)	Fall Rate ↑ Raw (Diff)	Reward ↓ Raw (Drop %)	Forward Velocity ↓ Raw (Drop %)	Fall Rate ↑ Raw (Diff)
STAR (Full)	**0.622 ± 0.079** **(-)**	**3.268 ± 0.166** **(-)**	**1.420 ± 0.252** **(-)**	**0.714 ± 0.061** **(-)**	**3.053 ± 0.168** **(-)**	**0.746 ± 0.129** **(-)**
STAR w/o MI	0.654 ± 0.072 (5.14%)	3.331 ± 0.153 (1.93%)	1.322 ± 0.198 (0.098)	0.745 ± 0.058 (4.34%)	3.138 ± 0.142 (2.78%)	0.683 ± 0.115 (0.063)
STAR w/o Soft Mask	0.682 ± 0.078 (9.65%)	3.384 ± 0.142 (3.55%)	1.248 ± 0.179 (0.172)	0.775 ± 0.049 (8.54%)	3.204 ± 0.125 (4.95%)	0.632 ± 0.101 (0.114)
STAR w/o MI & Soft Mask	0.712 ± 0.068 (14.47%)	3.452 ± 0.113 (5.63%)	1.124 ± 0.154 (0.296)	0.821 ± 0.033 (14.98%)	3.316 ± 0.087 (8.61%)	0.567 ± 0.079 (0.179)

ering temporal dynamics and long-term rewards. The STAR algorithm implements these insights by combining soft mask-based state targeting with information-theoretic optimization, enabling efficient generation of state-selective perturbations. Experimental results on robotic locomotion tasks demonstrate that our approach achieves improved attack effectiveness under fixed perturbation budgets compared to conventional gradient-based methods, while the resulting adversarial training enhances the robustness of DRL policies in robotic control applications.

Future work will focus on deploying the STAR algorithm on real-world robotic platforms to validate its practical applicability under various environmental settings, such as partially observable conditions. Additionally, we aim to extend the framework to a broader range of edge intelligence devices, including aerial and manipulator robots, to evaluate its generalizability across diverse applications. These efforts will further bridge the gap between theoretical advancements and real-world deployment, advancing robust deep reinforcement learning for edge intelligence systems.

REFERENCES

[1] H. Wei, B. Lou, Z. Zhang, B. Liang, F.-Y. Wang, and C. Lv, "Autonomous navigation for evtol: Review and future perspectives," *IEEE Trans. Intell. Veh.*, 2024.

[2] J. Chen, J. Cao, Z. Cheng, and S. Jiang, "Towards efficient distributed collision avoidance for heterogeneous mobile robots," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 3605–3619, 2024.

[3] S. T. Atik, A. S. Chavan, D. Grosu, and M. Brocanelli, "A maintenance-aware approach for sustainable autonomous mobile robot fleet management," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7394–7407, 2024.

[4] R. A. Khalil, Z. Safelnasr, N. Yemane, M. Kedir, A. Shafiqurrahman, and N. Saeed, "Advanced learning technologies for intelligent transportation systems: Prospects and challenges," *IEEE Open J. Veh. Technol.*, 2024.

[5] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, "A survey on deep reinforcement learning algorithms for robotic manipulation," *Sensors*, vol. 23, no. 7, p. 3762, 2023.

[6] S. Mohanti, D. Roy, M. Eisen, D. Cavalcanti, and K. Chowdhury, "L-norm: Learning and network orchestration at the edge for robot connectivity and mobility in factory floor environments," *IEEE Trans. Mobile Comput.*, vol. 23, no. 4, pp. 2898–2914, 2024.

[7] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," in *Conf. Robot Learn.*, 2021, pp. 432–448.

[8] L. Shi and Y. Chi, "Distributionally robust model-based offline reinforcement learning with near-optimal sample complexity," *J. Mach. Learn. Res.*, vol. 25, no. 200, pp. 1–91, 2024.

[9] L. Schott, J. Delas, H. Hajri, E. Gherbi, R. Yaich, N. Boulahia-Cuppens, F. Cuppens, and S. Lamprier, "Robust deep reinforcement learning through adversarial attacks and training: A survey," *arXiv preprint arXiv:2403.00420*, 2024.

[10] R. Duan, Y. Chen, D. Niu, Y. Yang, A. K. Qin, and Y. He, "Advdrop: Adversarial attack to dnns by dropping information," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 7506–7515.

[11] J. Chen, H. Chen, K. Chen, Y. Zhang, Z. Zou, and Z. Shi, "Diffusion models for imperceptible and transferable adversarial attack," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.

[12] Z. Chen, B. Li, S. Wu, K. Jiang, S. Ding, and W. Zhang, "Content-based unrestricted adversarial attack," *Adv. Neural Inf. Process. Syst.*, vol. 36, 2024.

[13] T. Oikarinen, W. Zhang, A. Megretski, L. Daniel, and T.-W. Weng, "Robust Deep Reinforcement Learning through Adversarial Loss," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 26 156–26 167, 2021.

[14] T. Hickling, N. Aouf, and P. Spencer, "Robust adversarial attacks detection based on explainable deep reinforcement learning for uav guidance and planning," *IEEE Trans. Intell. Veh.*, 2023.

[15] A. Haydari, M. Zhang, and C.-N. Chuah, "Adversarial attacks and defense in deep reinforcement learning (drl)-based traffic signal controllers," *IEEE Open J. Intell. Transp. Syst.*, vol. 2, pp. 402–416, 2021.

[16] J. Ji, B. Zhang, J. Zhou, X. Pan, W. Huang, R. Sun, Y. Geng, Y. Zhong, J. Dai, and Y. Yang, "Safety gymnasium: A unified safe reinforcement learning benchmark," *Adv. Neural Inf. Process. Syst.*, vol. 36, 2023.

[17] N. Akhtar, A. Mian, N. Kardan, and M. Shah, "Advances in Adversarial Attacks and Defenses in Computer Vision: A Survey," *IEEE Access*, vol. 9, pp. 155 161–155 196, 2021.

[18] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *arXiv preprint arXiv:1412.6572*, 2014.

[19] A. Madry, "Towards Deep Learning Models Resistant to Adversarial Attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[20] E. Wong, L. Rice, and J. Z. Kolter, "Fast Is Better Than Free: Revisiting Adversarial Training," *arXiv preprint arXiv:2001.03994*, 2020.

[21] L. Schwinn, R. Raab, A. Nguyen, D. Zanca, and B. Eskofier, "Exploring Misclassifications of Robust Neural Networks To Enhance Adversarial Attacks," *Appl. Intell.*, vol. 53, no. 17, pp. 19 843–19 859, 2023.

[22] X. Wang, X. He, J. Wang, and K. He, "Admix: Enhancing the Transferability of Adversarial Attacks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16 158–16 167.

[23] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting Adversarial Attacks with Momentum," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9185–9193.

[24] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving Transferability of Adversarial Examples with Input Diversity," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2730–2739.

[25] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading Defenses to Transferable Adversarial Examples by Translation-invariant Attacks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4312–4321.

[26] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks," *arXiv preprint arXiv:1908.06281*, 2019.

[27] I. Ilahi, M. Usama, J. Qadir, M. U. Janjua, A. Al-Fuqaha, D. T. Hoang, and D. Niyato, "Challenges and Countermeasures for Adversarial Attacks on Deep Reinforcement Learning," *IEEE Trans. Artif. Intell.*, vol. 3, no. 2, pp. 90–109, 2021.

[28] T. Duan, Z. Zhang, Z. Lin, Y. Gao, L. Xiong, Y. Cui, H. Liang, X. Chen, H. Cui, and D. Huang, "Rethinking adversarial attacks in reinforcement learning from policy distribution perspective," *arXiv preprint arXiv:2501.03562*, 2025.

[29] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial Attacks on Neural Network Policies," *arXiv preprint arXiv:1702.02284*, 2017.

[30] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, "Robust Deep Reinforcement Learning with Adversarial Attacks," *arXiv preprint arXiv:1712.03632*, 2017.

[31] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, "Tactics of Adversarial Attack on Deep Reinforcement Learning Agents," *arXiv preprint arXiv:1703.06748*, 2017.

[32] T.-W. Weng, K. D. Dvijotham, J. Uesato, K. Xiao, S. Gowal, R. Stanforth, and P. Kohli, "Toward Evaluating Robustness of Deep Reinforcement Learning with Continuous Control," in *Int. Conf. Learn. Represent.*, 2019.

[33] H. Zhang, H. Chen, C. Xiao, B. Li, M. Liu, D. Boning, and C.-J. Hsieh, "Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 21 024–21 037, 2020.

[34] R. S. Sutton, "Reinforcement learning: An introduction," *Bradford Book*, 2018.

[35] V. Mnih, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[36] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained Policy Optimization," in *Int. Conf. Mach. Learn.*, 2017, pp. 22–31.

[37] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," *arXiv preprint arXiv:1810.00069*, 2018.

[38] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *J. Artif. Intell. Res.*, vol. 47, pp. 253–279, 2013.

[39] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional Continuous Control Using Generalized Advantage Estimation," *arXiv preprint arXiv:1506.02438*, 2015.

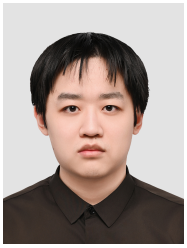
[40] J. Hwangbo, J. Lee, and M. Hutter, "Per-contact iteration method for solving contact dynamics," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 895–902, 2018. [Online]. Available: www.raisim.com

[41] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[42] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble Adversarial Training: Attacks and Defenses," *arXiv preprint arXiv:1705.07204*, 2017.

[43] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically Principled Trade-off Between Robustness and Accuracy," in *Int. Conf. Mach. Learn.*, 2019, pp. 7472–7482.

[44] X. Liu, Y. Li, C. Wu, and C.-J. Hsieh, "Adv-bnn: Improved Adversarial Defense through Robust Bayesian Neural Network," *arXiv preprint arXiv:1810.01279*, 2018.



Zongyuan Zhang received the B.Sc. degree in Information and Computing Science, and the B.Eng. degree in Computer Science and Technology from Beijing University of Technology in 2021. He is currently pursuing the Ph.D. degree with the Department of Computer Science, the University of Hong Kong. His research interests include reinforcement learning, edge intelligence, and robotics.

Tianyang Duan received the B.Eng. degree in Software Engineering from Beijing University of Technology, and University College Dublin in 2021. She is currently pursuing the Ph.D. degree with the Department of Computer Science, the University of Hong Kong. Her research interests include reinforcement learning, edge intelligence, and robotics.

Zheng Lin received the B.Eng. degree in Electronic Information from Fuzhou University in 2020, and the M.Eng. degree in Electrical and Computer Engineering from Fudan University in 2023. He is currently pursuing his Ph.D. degree with the Department of Electrical and Electronic Engineering, the University of Hong Kong, Hong Kong, China. He was awarded Fudan Outstanding Graduate and Shanghai Outstanding Graduate in 2023. He serves as a Young Professional in IEEE Vehicular Technology Society Ad Hoc Committee. His research interests include wireless networking, edge intelligence, and distributed machine learning.

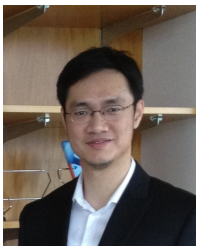
Dong Huang received his Ph.D. in Computer Science from the University of Hong Kong in 2025 and his B.Eng. in Material Processing and Control Engineering from Huazhong University of Science and Technology. He is currently a Research Fellow (Postdoctoral Researcher) with the Institute of Data Science, School of Computing, at the National University of Singapore. His current research interests are broadly in code intelligence + X (such as performance, responsibility, and security).

Zihan Fang received her master's degree from Fudan University in 2023. She is currently pursuing her Ph.D. degree at the City University of Hong Kong. Her research interests include vehicle networks and distributed machine learning.

Zekai Sun is currently working toward a PhD degree in the Systems and Networks Laboratory at the University of Hong Kong under the guidance of Prof. Heming Cui. He received Bachelor's Degree at University of Science and Technology of China (USTC) in 2020. His primary research areas include distributed systems, edge computing and systems for machine learning.

Ling Xiong (Member, IEEE) received the M.S. and Ph.D. degrees in information security from the School of Information Science and Technology, Southwest Jiaotong University. She is currently an Associate Professor with the School of Computer and Software Engineering, Xihua University. She is also a Postdoctoral researcher with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. Her research focuses on security and privacy in Internet of Things and blockchain.

Hongbin Liang (Member, IEEE) received the B.Sc. degree in Communication Engineering from Beijing University of Post and Telecommunication in 1995, and the M.Sc. and Ph.D. degrees in electrical engineering from Southwest Jiaotong University in 2001 and 2012, respectively. He is currently a Full Professor with the School of Transportation and Logistics, Southwest Jiaotong University. From 2001 to 2009, he was a Software Engineer with the Motorola Research and Development Center of China, where he focused on system requirement analysis and the Third-Generation Partnership Project protocol analysis. From 2009 to 2011, he was a visiting Ph.D. student with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His current research interests focus on resource allocation, quality of service, security and efficiency in cloud computing, vehicular networking, and wireless sensor networks.



Heming Cui (Member, IEEE) is an Associate Professor in the Department of Computer Science, the University of Hong Kong. His research interests include operating systems, programming languages, distributed systems, and cloud computing, with a particular focus on building software infrastructures and tools to improve reliability and security of real-world software.

Yong Cui (Member, IEEE) received the B.E. and Ph.D. degrees in Computer Science and Engineering from Tsinghua University in 1999 and 2004, respectively. He is currently a Full Professor with the Computer Science Department, Tsinghua University. His research interests include mobile cloud computing and network architecture. He served or serves on the Editorial Boards for IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON CLOUD COMPUTING, and the IEEE INTERNET COMPUTING.