Robotic Control in Adversarial and Sparse Reward Environments: A Robust Goal-Conditioned Reinforcement Learning Approach

Xiangkun He, *Member, IEEE*, and Chen Lv, *Senior Member, IEEE*

***Abstract*—With deep neural networks based function approximators, reinforcement learning holds the promise of learning complex end-to-end robotic controllers that can map high-dimensional sensory information directly to control policies. However, a common challenge, especially for robotics, is sample-efficient learning from sparse rewards, in which an agent is required to find a long sequence of "correct" actions to achieve a desired outcome. Unfortunately, inevitable perturbations on observations may make this task trickier to solve. Here, this paper advances a novel robust goal-conditioned reinforcement learning approach for end-to-end robotic control in adversarial and sparse reward environments. Specifically, a mixed adversarial attack scheme is presented to generate diverse adversarial perturbations on observations by combining white-box and black-box attacks. Meanwhile, a hindsight experience replay technique considering observation perturbations is developed to turn a failed experience into a successful one and generate the policy trajectories perturbed by the mixed adversarial attacks. Additionally, a robust goal-conditioned actor-critic method is proposed to learn goal-conditioned policies and keep the variations of the perturbed policy trajectories within bounds. Finally, the proposed method is evaluated on three tasks with adversarial attacks and sparse reward settings. The results indicate that our scheme can ensure robotic control performance and policy robustness on the adversarial and sparse reward tasks.**

***Impact Statement*—In recent years, reinforcement learning has been an impressive component of modern artificial intelligence and is still under vigorous development. Nonetheless, compared to supervised learning, which has been widely applied in a variety of domains, reinforcement learning has not been broadly accepted and deployed in real-world problems. One key factor is an agent's trustworthiness, where its policy robustness is essential. Additionally, designing the reward function requires both domain-specific knowledge and reinforcement learning expertise, which limits the applicability of reinforcement learning. Unfortunately, on some real-world tasks, on account of reward function design complexities and inevitable perception errors, the agents have to learn under sparse rewards and observation uncertainties. However, so far there are few studies to cope with the challenge. Our approach contributes to the foundation for the realization of trustworthy and efficient artificial intelligence, potentially bringing reinforcement learning closer to a wide range of real-world applications in robotics and beyond.**

***Index Terms*—Reinforcement learning, adversarial machine learning, sparse reward, end-to-end control, robotics.**

I. Introduction

WITH the development of emerging technologies such as artificial intelligence (AI) and 5th-generation mobile communication technology (5G), data-driven robotic control has become a research hotspot, which can lead to a dramatic breakthrough in the next generation of robot industry [1]–[3]. Conventional robotic control approaches employ a hierarchical control architecture that mainly consists of sensing, planning and control modules [4]–[6]. In addition, these methods require accurate kinematics or dynamics models that are tricky to acquire, especially in complex robotic control [7]–[9].

With deep neural networks (DNNs) as function approximators, reinforcement learning (RL) methods have demonstrated their worth in a series of challenging tasks, from games to robotic control [10]–[13]. High-quality end-to-end robot controllers can be implemented through RL methods without prior hierarchical framework, kinematics and dynamics models [14]. The RL-based robot control approach was proposed through a mixture of actor-critic experts (MACE) in [15]. The residual RL for robot control is designed by combining a learnable parametrized model with a conventional feedback controller in [16]. The multi-agent advantage actor-critic (A3C) algorithm was developed for snake robot control in [17]. The RL method with Lyapunov stability theory was presented to guarantee closed-loop stability of the robot controller in [18]. The prediction-guided RL algorithm was developed for multi-objective continuous robot control tasks in [19].

While existing RL-based robot control methods have achieved many compelling results [20]–[22], they mostly rely on carefully-crafted reward functions that require both domain-specific knowledge and RL expertise. The reward design engineering restricts the real-world applicability of RL. Furthermore, since it is very tricky to design the reward function for some tasks, such as Go, the agents have to learn in sparse reward environments. Hence, a common challenge in RL, especially for robotics, is sample-efficient learning from sparse rewards, in which an agent has to find a long sequence of "correct" actions to achieve a desired outcome. Many studies have made efforts to solve this tricky issue [23]–[25]. One popular way of tackling the sparse reward task is the hindsight experience replay (HER) technique [26] that seeks to cope with this problem via converting failed experiences to successful ones through relabeling the goals.

The above studies generally assume that the agents' sensing and perception systems are free of uncertainties. Nonetheless, this assumption can barely hold in real-world situations. The observations of robots include inevitable perturbations

X. He and C. Lv are with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore 639798. (e-mail: xiangkun.he@ntu.edu.sg, lyuchen@ntu.edu.sg) *(Corresponding author: Chen Lv)*

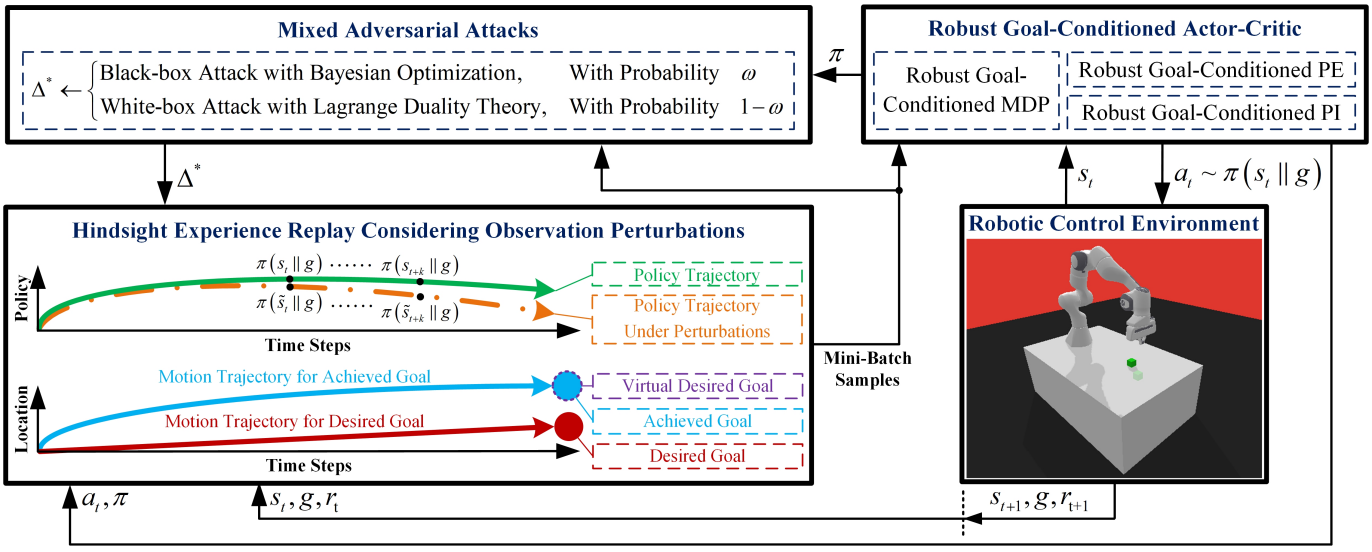

Fig. 1. Illustration of the proposed RGCRL framework for robotic control in adversarial and sparse reward environments.

that naturally arise from unpredictable stochastic noises or sensing errors. If a robot's control policies are not robust, its performance may be degraded by observation uncertainties, especially in sparse reward environments. In other words, the perturbations on observations make agents more difficult to discover a trajectory of "correct" actions to obtain a positive sparse reward signal.

In recent years, some researchers have tried to improve the robustness of policies based on RL in robotic control [27]–[29]. For example, Pinto et al. [30] presented the robust adversarial RL algorithm for robotic control by modeling uncertainty as an adversarial agent. Tessler et al. [31] proposed the robotic control approach with action robust RL through structuring probabilistic action robust Markov decision process (MDP) and noisy action robust MDP. Pattanaik et al. [32] introduced the state adversarial robust RL scheme to improve the robotic control policy's robustness via the gradient based adversarial attack technique. However, few works simultaneously try to cope with observational uncertainties and sparse rewards in robotic control tasks. Consequently, there is still room for advancement and refinement.

In this paper, we advance a novel robust goal-conditioned reinforcement learning (RGCRL) approach for end-to-end robotic control in adversarial and sparse reward environments. The RGCRL scheme is based on two key insights. First, the robust policies facilitate the agent to make a long sequence of "correct" decisions to attain a positive sparse reward signal under uncertain perturbations. Second, the method of combining white-box attack [33] and black-box attack [34]–[36] is able to generate diverse adversarial samples. Specifically, this paper's main contributions are summed up as follows:

- A mixed adversarial attack scheme is proposed by combining white-box and black-box attacks, which aims to maximize the average variation distance on attacked policies while generating diverse adversarial samples on observations.
- An HER technique considering observation perturbations is developed to turn a failed experience into a successful one and produce the policy trajectories perturbed by the mixed adversarial attacks.
- A robust goal-conditioned actor-critic (RGCAC) method is presented to optimize goal-conditioned policies and keep the variations of the perturbed policy trajectories within bounds, in sparse reward environments.

The proposed approach's framework is illustrated in Fig. 1. The module with respect to the mixed adversarial attacks aims to generate diverse adversarial samples on observations. Its input contains the state s, goal g and the goal-conditioned policy $\pi(s\|g)$. $\|$ denotes concatenation. The output is the optimal adversarial attack Δ^*.

The HER module considering observation perturbations is to provide the hindsight experiences and the policy trajectories under the mixed adversarial attacks. Its input includes the state s_t, goal g, action a_t, reward r_t, next state s_{t+1}, goal-conditioned policy π and optimal adversarial attack Δ^*. t and k represents the time steps. $\tilde{s}$ denotes the state perturbed by the mixed adversarial attacks. The output contains samples for policy optimization and adversarial attacks.

The module associated with the RGCAC algorithm attempts to optimize the end-to-end robust control policies against observation perturbations. Its input includes the hindsight experiences and the perturbed policy trajectories. The output is the robotic control policy.

In addition, the module with regard to the robotic control environment is adopted to produce the transition data. The input is the goal-conditioned policy $\pi(s\|g)$ based action a. Its output includes the state s, goal g and reward r.

Three testing cases with adversarial and sparse reward settings are executed to benchmark the performance of our robotic control method in the Franka Emika Panda robot environment [37]. The results indicate that the RGCRL scheme can learn robust control policies against observation uncertainties while improving the robot performance.

The remainder of this paper is arranged as follows. The pre-

liminaries and the methodology of our method are introduced in Sections II and III respectively. Section IV analyzes the testing results of the proposed algorithm on the different robot tasks. This work's conclusions are made in Section V.

II. Preliminaries

A. Markov Decision Process

An MDP is a mathematical formalism for modelling the sequential decision making of an agent, and it is also a straightforward paradigm for the problem of learning from interaction to attain a goal. A standard MDP is able to be expressed via the 5-tuple $(\mathcal{S}, \mathcal{A}, r, p, \gamma)$ based on the state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, transition probability function $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ and discount factor $\gamma \in (0, 1)$.

B. Reinforcement Learning

The MDP is the theoretical foundation of RL, and the RL agent aims to learn a policy that is capable of maximizing the expected return. In the RL problem, the expected return can then be denoted by:

$$J(\pi) = \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\Big], \tag{1}$$

where π indicates the the agent's policy, t is the time step, T is the last timestep, state $s_t \in \mathcal{S}$, and action $a_t \in \mathcal{A}$.

Therefore, the central optimization problem of RL can be represented with:

$$\pi^* = \arg\max_{\pi} J(\pi), \tag{2}$$

where π^* is the optimal policy.

III. Methodology

A. Mixed Adversarial Attacks

In this section, in order to generate diverse adversarial samples, the mixed adversarial attack method is introduced by combining black-box and white-box attacks. The mixed adversarial attack scheme can be described as:

$$\Delta^* \leftarrow \begin{cases} \text{black-box attack}, & \text{with probability } \omega \\ \text{white-box attack}, & \text{with probability } 1-\omega \end{cases} \tag{3}$$

where ω represents a probability.

1) Black-box Attack with Bayesian Optimization: We implement the Bayesian optimization based black-box attack technique that has a high query efficiency for approximating adversarial samples. The black-box attack methods do not require information on the objective function architecture or parameters and only observe the input-output correspondences by querying the model. The black-box attack method, however, typically requires a large number of attempts to find effective adversarial samples.

The Bayesian optimization method, a black-box optimization approach founded on the Bayes theorem, is particularly well suited for dealing with issues involving expensive queries. This technique works by building an objective function's surrogate model that can be efficiently searched via an acquisition function before candidate samples are selected for assessing the real objective function.

The Bayesian optimization based black-box attack method is described by Algorithm 1. The upper confidence bound (UCB) method is chosen to design the acquisition function in our scheme. We adopt the Gaussian process to build the surrogate model. Moreover, in this paper, the objective function $f(\cdot)$ of the black-box attack scheme is defined as:

$$f(\Delta, s, g, \pi) = \left\| \pi(s \| g) - \pi(\tilde{s} \| g) \right\|_2, \tag{4}$$

where the observation uncertainty is represented as $\Delta = [\Delta^m \;\; \Delta^a]$, the perturbed state is denoted as $\tilde{s} = \Delta^m s + \Delta^a$, Δ^m indicates the multiplicative uncertainty, and Δ^a is the additive uncertainty.

Algorithm 1 Bayesian optimization based black-box attacks

1: **for** $i = 1, 2, ..., I$ **do**
2: Probe a new adversarial sample $\Delta_i = [\Delta_i^m \;\; \Delta_i^a]$ via maximizing the UCB based acquisition function over the Gaussian process based surrogate model: $\Delta_i = \arg\max_{\Delta} \text{UCB}(\Delta | \mathcal{M}_{1:i-1})$.
3: Compute the objective function $f(\Delta_i, s, g, \pi)$.
4: Augment data in memory M: $\mathcal{M}_{1:i} = \mathcal{M}_{1:i-1} \cup [\Delta_i, f(\Delta_i, s, g, \pi)]$.
5: Update the Gaussian process based surrogate model.
6: **end for**

2) White-box Attack with Lagrange Duality Theory: The white-box attack method is transformed into a constrained optimization problem that can be solved by Lagrange duality theory. The white-box attack technique requires complete knowledge of the objective function to find successful adversarial samples.

In this paper, the white-box attack scheme is formulated as:

$$\begin{aligned} &\max_{\Delta} f(\Delta, s, g, \pi), \\ &\text{s.t.} \quad \left\| \Delta - \bar{\Delta} \right\|_1 \le \delta, \end{aligned} \tag{5}$$

where the perturbation reference value can be denoted as $\bar{\Delta} = [\bar{\Delta}^m \;\; \bar{\Delta}^a]$, $\bar{\Delta}^m$ and $\bar{\Delta}^a$ represent the multiplicative and additive uncertainties' reference values, and δ^m and δ^a are the the multiplicative and additive uncertainties' bounds, respectively.

Therefore, the Lagrangian of the above constrained optimization task can be obtained:

$$L(\Delta, \alpha) = f(\Delta, s, g, \pi) + \alpha \left(\left\| \Delta - \bar{\Delta} \right\|_1 - \delta \right), \tag{6}$$

where α is dual variable.

With Lagrange duality theory, the white-box attack can be formulated as:

$$\max_{\Delta} \min_{\alpha \ge 0} L(\Delta, \alpha). \tag{7}$$

Hence, the adversarial perturbation Δ can be updated in J steps by:

$$\Delta_{j+1} = \Delta_j + \eta \nabla_{\Delta} L(\Delta, \alpha), \tag{8}$$

where η is the learning rate of optimizing Δ. Moreover, $j = 1, ..., J$.

The dual variable α is updated in J steps through:

$$\alpha_{j+1} = \alpha_j - \xi \nabla_\alpha L(\Delta, \alpha), \tag{9}$$

where ξ is the learning rate of approximating α.

B. HER Considering Observation Perturbations

On sparse reward tasks, the reward relies on whether the policy trajectory enables the agent to reach the desired goal g or not, and only the successful policy trajectory can trigger a positive reward. In most cases, the successful policy trajectories collected by the agent are usually insufficient for training.

The HER technique considering observation perturbations aims to provide the hindsight experiences and the perturbed policy trajectories.

To generate the hindsight experiences for addressing the sparse reward problem, the HER scheme converts failed experiences into successful ones via relabeling the goals. Specifically, the HER method converts the achieved goals g' based on the states in failed experiences to the desired goals g in the training data. Here, the desired goal g denotes the real target that the agent attempts to attain. Moreover, an achieved goal g' represents a state that the agent has reached. When g is displaced via a g', the corresponding failed experiences are assigned desirable rewards, which is able to facilitate the agent to learn policies in sparse reward environments. The policy trajectory based on the hindsight experience in an episode can be expressed as:

$$\tau = \{\pi(s_0 \| g'), \cdots, \pi(s_{T-1} \| g')\}. \tag{10}$$

Additionally, we construct the perturbed policy trajectories via the sampled hindsight experiences and the mixed adversarial attacks. Hence, the perturbed policy trajectory in an episode is able to be represented as:

$$\tilde{\tau} = \{\pi(\tilde{s}_0 \| g'), \cdots, \pi(\tilde{s}_{T-1} \| g')\}. \tag{11}$$

C. Robust Goal-Conditioned Actor-Critic

This section introduces the proposed RGCAC algorithm that enables the robotic agent to learn the robust goal-conditioned control policy.

1) Robust Goal-Conditioned Markov Decision Process: A robust goal-conditioned MDP (RGCMDP) is proposed to model agent behaviors under uncertainties and sparse rewards, which can be defined as follows.

Definition 1: An RGCMDP can be represented through a 7-tuple $(\mathcal{S}, \mathcal{G}, \mathcal{A}, r, p, \Delta, \gamma)$. $\mathcal{G}$ indicates the goal space. Δ represents the observational uncertainty.

To handle the worst-case situation, RGCMDP aims to solve the optimal policies under optimal adversarial attacks on observations. The agent tries to learn goal-conditioned policies and keep the variations of the perturbed policy trajectories within bounds. The optimization task can be formulated as:

$$\begin{aligned} &\pi^* \in \arg\max_\pi \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(s_t, a_t, g)\Big], \\ &\text{s.t.} \quad \mathbb{E}[f(\Delta^*, s, g, \pi)] \le \beta, \end{aligned} \tag{12}$$

where π represents a goal-conditioned policy $\pi(s\|g)$, π^* denotes an optimal goal-conditioned policy $\pi^*(s\|g)$, and β is an upper bound.

2) Robust Goal-Conditioned Policy Evaluation: The action-value function $Q^\pi(s_t, a_t, g)$ at time step t is able to be computed under a given goal g and a fixed agent policy iteratively via a Bellman backup operator $\mathcal{T}$:

$$\mathcal{T}Q^\pi(s_t, a_t, g) \equiv r(s_t, a_t, g) + \gamma \mathbb{E}[Q^\pi(s_{t+1}, a_{t+1}, g)]. \tag{13}$$

The RGCAC algorithm leverages two action-value functions with parameters ϕ^z, $z \in \{1, 2\}$ to speed up training. The action-value function parameters are able to be learned by minimizing the following the critic network's loss function:

$$J_c(\phi^z) = \mathop{\mathbb{E}}_{T_b \sim \mathcal{M}}\Big[\|y - Q^\pi(s_t, a_t, g; \phi^z)\|_2^2\Big], \tag{14}$$

where T_b denotes a minibatch trajectory sampled from the HER memory $\mathcal{M}$, and y represents the action-value function's target value.

To alleviate the overfitting problem, a smoothing regularization scheme [38] is adopted by adding a small amount of stochastic noises to the selected action with a deterministic policy. Moreover, with the state s_t, goal g and deterministic policy π, the agent's action with the random noises can be designed as:

$$\tilde{\pi}(s_t \| g; \theta) = \pi(s_t \| g; \theta) + c\nu, \tag{15}$$

$$\nu \sim \mathcal{N}(0, 1), \tag{16}$$

where θ denotes the actor network's parameters, c is a weight coefficient, and $\mathcal{N}(\cdot)$ represents Gaussian distribution.

To mitigate the action-value function's overestimation issue, the minimum estimation between two target action-value functions is utilized to optimize the critic network's parameters. Hence, based on Eq. (13), y is able to be written as:

$$y = r(s_t, a_t, g) + \gamma \min_{z \in \{1,2\}} \hat{Q}^\pi\big(s_{t+1}, \tilde{\pi}(s_t \| g; \theta), g; \bar{\phi}^z\big). \tag{17}$$

where $\hat{Q}^\pi(\cdot)$ represents the target action-value function, $\bar{\phi}^z$ denotes the target action-value function's network parameters, and $z \in \{1, 2\}$.

To stabilize the model training, the the target action-value function's network parameters are able to be renewed through polyak averaging:

$$\bar{\phi}^z \leftarrow \mu \bar{\phi}^z + (1 - \mu)\phi^z, \tag{18}$$

where μ denotes a interpolation factor between 0 and 1.

3) Robust Goal-Conditioned Policy Improvement: The policy improvement aims to optimize and update the agent's policies. Our robotic control method attempts to maximize the agent's expected return and keep the changes of the policy trajectories perturbed by the mixed adversarial attacks within certain ranges, in sparse reward environments.

The Lagrangian of the constrained optimization problem (12) is able to be written as:

$$L(\pi, \lambda) = \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(s_t, a_t, g) + \lambda\big(\beta - f(\Delta^*, s_t, g, \pi)\big)\Big], \tag{19}$$

where $\lambda \geq 0$, and λ represents the dual variable.

With Eq. (19) and Lagrangian duality [39], the the constrained optimization problem's Lagrange dual function is able to be represented as:

$$\begin{aligned}\bar{L}(\lambda) &= \max_{\pi} L(\pi, \lambda) \\ &= \max_{\pi} \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(s_t, a_t, g) + \lambda\big(\beta - f(\Delta^*, s_t, g, \pi)\big)\Big].\end{aligned} \tag{20}$$

In addition, the Lagrange dual problem concerning the problem (12) is able to be expressed as:

$$\begin{aligned}\min_{\lambda} \bar{L}(\lambda) &= \min_{\lambda} \max_{\pi} L(\pi, \lambda) \\ &= \min_{\lambda} \max_{\pi} \mathbb{E}\Big[\sum_{t=0}^{T} \gamma^t r(s_t, a_t, g) + \lambda\big(\beta - f(\Delta^*, s_t, g, \pi)\big)\Big].\end{aligned} \tag{21}$$

Under the given state s and goal g, with Eq. (21), the optimal goal-conditioned policies $\pi^*(s\|g)$ and the optimal dual variable λ^* are able to be found iteratively. Fix a dual variable λ, and then optimize the goal-conditioned policies $\pi(s\|g)$ via maximizing Eq. (19). Furthermore, with the optimal goal-conditioned policies $\pi^*(s\|g)$, the optimal dual variable λ^* can be attained via minimizing Eq. (20). As a result, the following expressions are able to be obtained:

$$\pi^*(s\|g) = \arg\max_{\pi} L\big(\pi(s\|g), \lambda\big), \tag{22}$$

$$\lambda^* = \arg\min_{\lambda} L\big(\pi^*(s\|g), \lambda\big). \tag{23}$$

A double action-value function trick is employed to mitigate the expected return's estimation error problem. At time step t, the average action-value function with the goal g is able to be represented as:

$$\begin{aligned}\tilde{Q}^{\pi}(s_t, a_t, g) &= \frac{1}{2}\big[Q^{\pi}(s_t, a_t, g; \phi^1) + Q^{\pi}(s_t, a_t, g; \phi^2)\big] \\ &= \frac{1}{2}\big[Q^{\pi}\big(s_t, \tilde{\pi}(s_t\|g; \theta), g; \phi^1\big) \\ &\quad + Q^{\pi}\big(s_t, \tilde{\pi}(s_t\|g; \theta), g; \phi^2\big)\big].\end{aligned} \tag{24}$$

Hence, with Eq. (22) and Eq. (24), the optimal policy of the agent is able to be learned via maximizing the following actor network's objective function:

$$J_a(\theta) = \mathop{\mathbb{E}}_{T_b \sim \mathcal{M}}\Big[\tilde{Q}^{\pi}(s_t, a_t, g) - \lambda f(\Delta^*, s_t, g, \pi)\Big]. \tag{25}$$

In addition, with Eq. (23), the dual variable λ is able to be learned through minimizing the following objective function:

$$J_d(\lambda) = \mathop{\mathbb{E}}_{T_b \sim \mathcal{M}}\big[\lambda\big(\beta - f(\Delta^*, s_t, g, \pi)\big)\big]. \tag{26}$$

D. Algorithm Implementation

Algorithm 2 introduces our RGCRL approach in detail. Moreover, d_t represents if s_{t+1} is terminal.

Algorithm 2 Robust Goal-Conditioned RL

Initialize θ,λ, ϕ^1, ϕ^2, $\bar{\phi}^1 \leftarrow \phi^1$ and $\bar{\phi}^2 \leftarrow \phi^2$.
Set an empty HER memory $\mathcal{M}$.
for epoch step $n = 1, 2, \dots N$ **do**
  for episode step $e = 1, 2, \dots E$ **do**
    Sample an initial state s_0 and a desired goal g.
    for time step $t = 1, 2, \dots T$ **do**
      Determine an action with the policy: $a_t \sim \pi_\theta(s_t\|g)$.
      Execute a_t in the environment and receive a transition: $s_{t+1}, r_t, d_t \sim p(s_{t+1}|s_t, a_t)$.
    end for
    Save the transition trajectory in the HER memory $\mathcal{M}$.
    for learning step $l = 1, 2, \dots L$ **do**
      Sample a batch of hindsight experiences from $\mathcal{M}$.
      Generate the optimal adversarial attacks on observations:

$$\Delta^* \leftarrow \begin{cases}\text{black-box attack}, & \text{with probability } \omega; \\ \text{white-box attack}, & \text{with probability } 1-\omega.\end{cases}$$

      Construct a batch of the perturbed policy trajectories.
      Optimize the parameters of the critic network via Eq. (14): $\phi^1 \leftarrow \nabla_{\phi^1} J_c(\phi^1)$, $\phi^2 \leftarrow \nabla_{\phi^2} J_c(\phi^2)$.
      Optimize the parameters of the actor network via Eq. (25): $\theta \leftarrow \nabla_\theta J_a(\theta)$.
      Optimize the dual variables via Eq. (26): $\lambda \leftarrow \nabla\lambda J_d(\lambda)$.
      Update the parameters of the target action-value function network via Eq. (18): $\bar{\phi}^1 \leftarrow \mu\phi^1 + (1-\mu)\bar{\phi}^1$, $\bar{\phi}^2 \leftarrow \mu\phi^2 + (1-\mu)\bar{\phi}^2$.
    end for
  end for
end for

The actor and the critic networks are implemented by 2 fully connected hidden layers whose layer sizes are $\{256, 256\}$. Additionally, ReLU is employed as the activation function in the hidden layers. The main hyperparameters of the proposed RGCRL algorithm are provided in Table I.

TABLE I
THE MAIN HYPERPARAMETERS OF THE RGCRL AND BASELINE ALGORITHMS.

Parameters	TD3-HER	SAC-HER	RDDPG-HER	RGCRL
Batch size	256	256	256	256
Buffer size	10^6	10^6	10^6	10^6
Upper bound β	N/A	N/A	N/A	0.1
Discount factor γ	0.98	0.98	0.98	0.98
Weight coefficient c	0.2	N/A	0.2	0.2
Attack probability ω	N/A	N/A	N/A	0.1
Interpolation factor μ	0.005	0.005	N/A	0.005
Actor's learning rate l_a	0.001	0.001	0.001	0.001
Critic's learning rate l_c	0.001	0.001	0.001	0.001
Attacker's learning rate η	N/A	N/A	0.01	0.01
Dual variable's learning rate l_λ	N/A	N/A	N/A	0.001

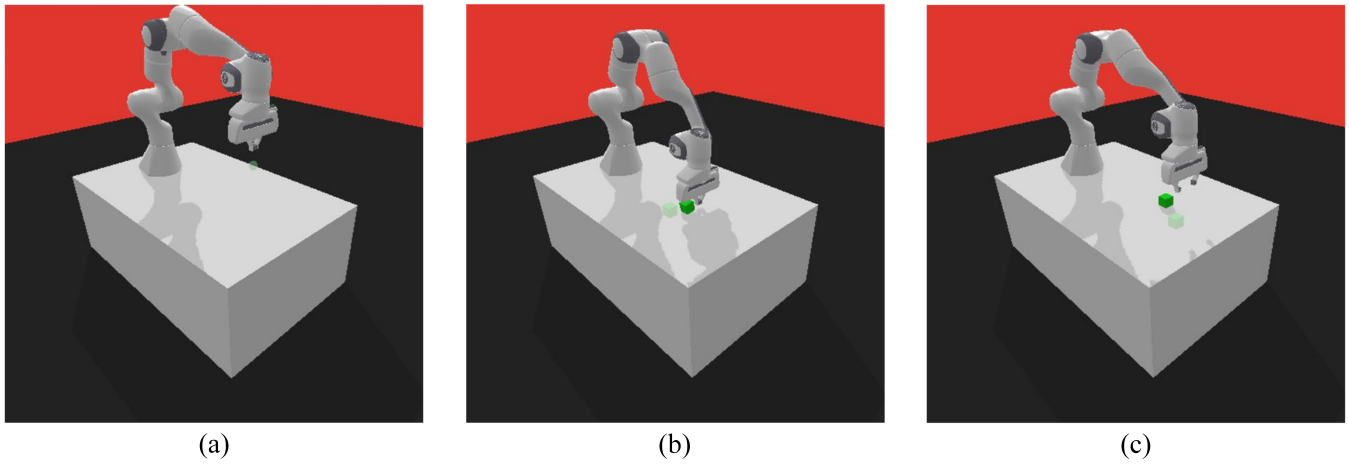

Fig. 2. Franka Emika Panda robot control environment. (a) Reach task; (b) Push task; (c) Pick-and-Place task. The target positions are green shaded.

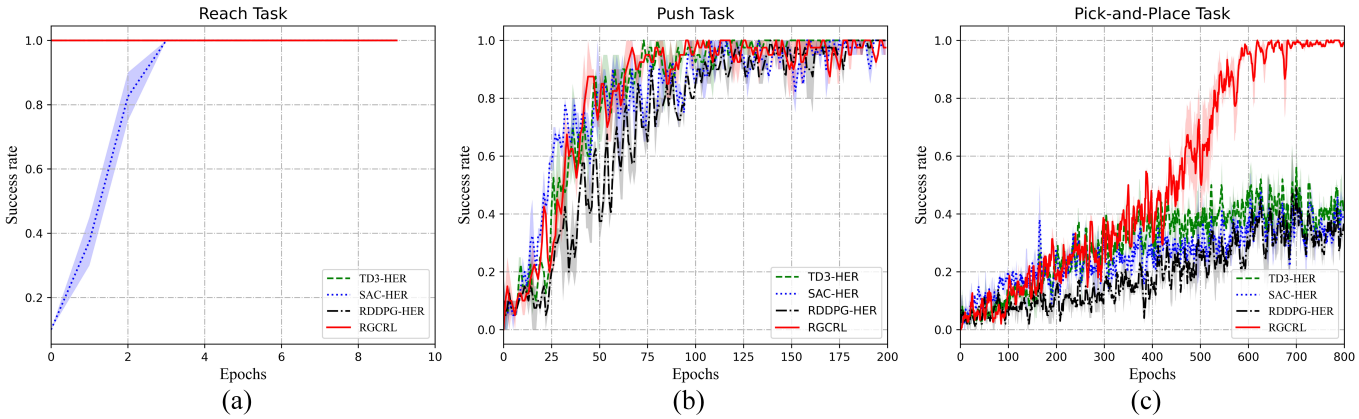

Fig. 3. Training curves obtained via the TD3-HER, SAC-HER, RDDPG-HER and RGCRL approaches on the three robotic control tasks with sparse rewards.

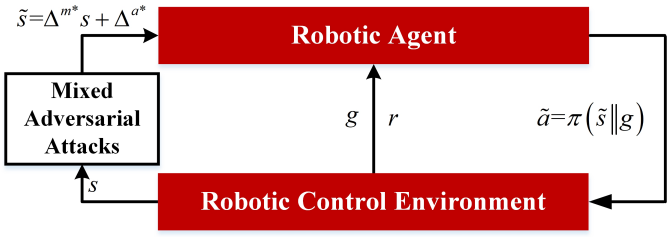

Fig. 4. Illustration of model testing scheme. In model testing, the robotic agent observes the state $\tilde{s}$ perturbed by the mixed adversarial attacks rather than the state s.

IV. Evaluation and Result

A. Experimental Environment

To benchmark our RGCRL method, we leverage the Franka Emika Panda robot environment [37] consisting of the Franka Emika Panda robotic arm model, the PyBullet physics engine [40] and OpenAI Gym [41]. As shown in Fig.2, the three robotic control tasks are illustrated, which include reach, push and pick-and-place. The green shaded spaces in Fig.2 denote the target positions. The green cubes represent the objects on push and pick-and-place tasks.

Specifically, on the reach task, the robotic arm has to control its end-effector at a specified position. The agent's input (i.e., state and goal, 9 dimensions) consists of the position and speed of its end-effector (6 dimensions), and the target position (3 dimensions). The output (i.e., action) is the end-effector control command (3 dimensions, one for movement on x, y and z axes).

On the push task, the robotic arm tries to push an object to a target position on the table surface. The robotic agent's input (21 dimensions) consists of its end-effector's position and speed (6 dimensions), the object's position, orientation, linear and rotational speed (12 dimensions), and the target position (3 dimensions). The robotic agent's output is the end-effector control command (3 dimensions, one for movement on x, y and z axes).

Moreover, on the pick-and-place task, the robotic arm attempts to pick up and place a cube at a target position above the table. The robotic agent's input (22 dimensions) consists of its end-effector's position and speed (6 dimensions), the object's position, orientation, linear and rotational speed (12

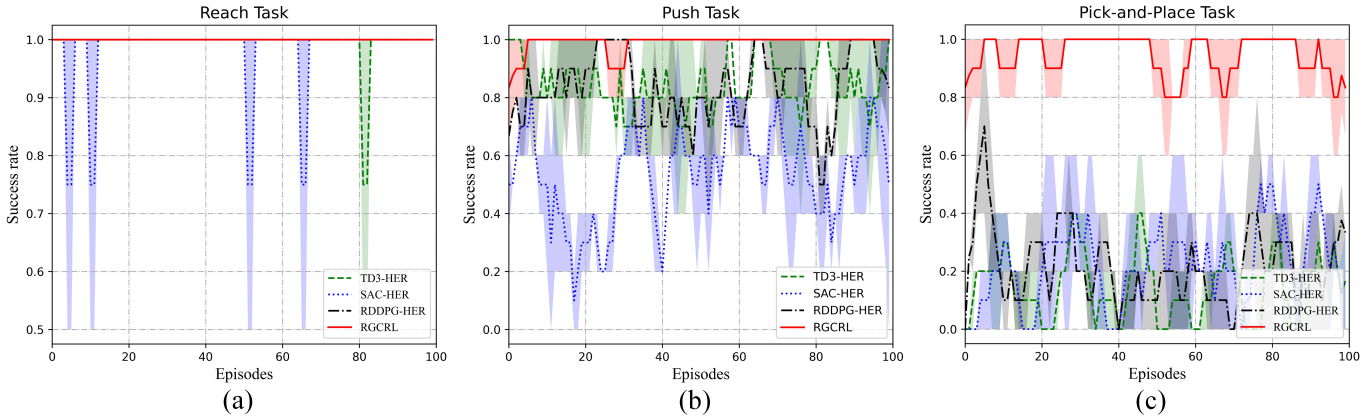

Fig. 5. Success rate of different robotic agents under the mixed adversarial attacks on observations. (a) Reach task; (b) Push task; (c) Pick-and-Place task.

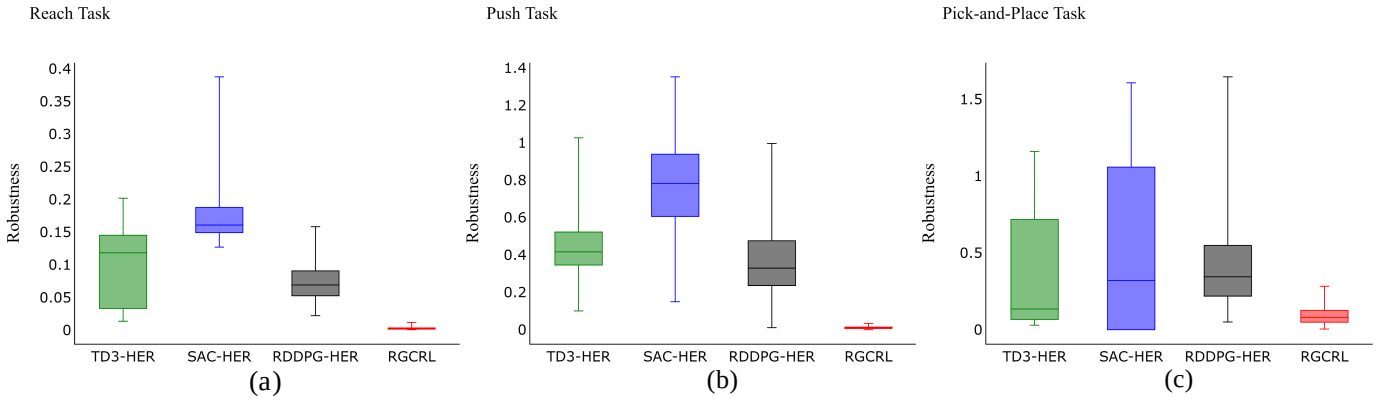

Fig. 6. Robustness of different robotic agents on the three robotic control tasks under the mixed adversarial attacks on observations.

dimensions), the target position (3 dimensions), and the opening of the end-effector (1 dimension, the distance between the fingers). The robotic agent's output (4 dimensions) is the end-effector control command (3 dimensions, one for movement on x, y and z axes), the fingers control command (1 dimension, one for movement of the fingers).

The sparse reward setting is leveraged for all three robotic control tasks, i.e., a reward of 0 is received when the object to move is at the target location and -1 otherwise.

B. Performance Evaluation

1) Baseline: To evaluate the proposed scheme, we try to implement comparisons with state-of-the-art off-policy RL algorithms, including twin delayed deep deterministic policy gradient (TD3) [38] and soft actor-critic (SAC) [42]. Additionally, since robust deep deterministic policy gradient (RDDPG) [32] is one of the state-of-the-art robust RL algorithms, it is also employed to test the proposed solution. However, it is intractable for traditional RL methods to attain desired performance in sparse reward environments. Hence, by combining HER [26] with TD3, SAC and RDDPG algorithms, three baselines are implemented to benchmark the proposed RGCRL technique, which are indicated as TD3-HER, SAC-HER and RDDPG-HER, respectively. The main hyperparameters of the baseline algorithms are given in Table I.

2) Model Training Performance: We perform 5 different runs of each method with different random seeds. On the reach, push and pick-and-place tasks, the robotic agents are trained for 10, 200 and 800 epochs respectively. 1 epoch includes 50 episodes. Moreover, 1 episode contains 50 time steps.

Fig. 3 shows the performance of each agent during training across the three robotic control tasks. The solid curve indicates the mean, and the shaded region represents the standard deviation. The results indicate that, overall, the proposed RGCRL scheme performs comparably to the baselines on the reach and push tasks, and surpasses them on the pick-and-place task with a large margin, both in the matter of the final performance and learning speed. For instance, on the reach and push tasks, the final success rates of the TD3-HER, SAC-HER, RDDPG-HER and RGCRL agents are 100%. Additionally, on the pick-and-place task, the final success rates of the TD3-HER, SAC-HER, RDDPG-HER and RGCRL agents are about 50%, 40%, 40% and 100%, respectively.

3) Model Testing Performance: As depicted in Fig. 4, in contrast to the training phase of the policy model, in model testing, the robotic agent receives the state $\tilde{s}_t$ perturbed by the mixed adversarial attacks instead of the state s_t.

TABLE II
EVALUATION OF DIFFERENT ROBOTIC AGENTS ON THE THREE ROBOTIC CONTROL TASKS UNDER THE MIXED ADVERSARIAL ATTACKS ON OBSERVATIONS.

Task	Metric	TD3-HER	SAC-HER	RDDPG-HER	RGCRL
Reach	Success Rate	0.995 ± 0.071	0.980 ± 0.140	$\mathbf{1.000 \pm 0.000}$	$\mathbf{1.000 \pm 0.000}$
	Robustness	0.095 ± 0.058	0.189 ± 0.065	0.075 ± 0.029	$\mathbf{0.003 \pm 0.002}$
Push	Success Rate	0.845 ± 0.362	0.535 ± 0.499	0.835 ± 0.371	$\mathbf{0.990 \pm 0.100}$
	Robustness	0.445 ± 0.132	0.750 ± 0.241	0.368 ± 0.167	$\mathbf{0.010 \pm 0.005}$
Pick-and-Place	Success Rate	0.145 ± 0.352	0.220 ± 0.414	0.215 ± 0.411	$\mathbf{0.945 \pm 0.228}$
	Robustness	0.365 ± 0.343	0.526 ± 0.546	0.425 ± 0.275	$\mathbf{0.092 \pm 0.054}$

We assess the final neural network models trained via each algorithm under 5 different random seeds. Each model is tested for 100 episodes on the three tasks. Here, like the model training, 1 episode includes 50 time steps. Eq. (4) is adopted to measure the policy robustness against the adversaria attacks on observations. This means the agent's policy with a smaller value of Eq. (4) shows stronger robustness against observation perturbations.

Fig. 5 and Fig. 6 show the performance and robustness of the TD3-HER, SAC-HER, RDDPG-HER and RGCRL agents during testing across the three robotic control tasks under the mixed adversarial attacks on observations. Obviously, the RGCRL agent outperforms the TD3-HER, SAC-HER and RDDPG-HER agents in terms of success rate and policy robustness, on the three tasks.

Qualitatively, we report the average metrics of 100 episodes in Table II for each agent on the three robotic control tasks under the mixed adversarial attacks on observations. In each row of Table II, the bolded number represents the best result. For example, on the reach task, in comparison with the TD3-HER, SAC-HER and RDDPG-HER agents, the RGCRL agent gains approximately 0.503%, 2.041%, and 0.000% improvements, respectively, regarding success rate. Compared with the TD3-HER, SAC-HER and RDDPG-HER agents, the policy robustness of the RGCRL agent is enhanced by approximately 96.842%, 98.413%, and 96.000%, respectively.

On the push task, in contrast to the TD3-HER, SAC-HER and RDDPG-HER agents, the RGCRL agent approximately enhances the success rate by 17.160%, 85.047%, and 18.563%, respectively. In comparison with the TD3-HER, SAC-HER and RDDPG-HER agents, the policy robustness of the RGCRL agent is improved by approximately 97.753%, 98.667%, and 97.283%, respectively.

On the pick-and-place task, compared with the TD3-HER, SAC-HER and RDDPG-HER agents, the RGCRL agent gains approximately 0.503%, 2.041%, and 0.000% improvements, respectively, in terms of success rate. Compared with the TD3-HER, SAC-HER and RDDPG-HER agents, the RGCRL agent's policy robustness is improved by approximately 551.724%, 329.546%, and 339.535%, respectively.

Taken overall, the proposed RGCRL approach surpasses the baseline schemes by a large margin, both in terms of success rate and policy robustness, especially on the complicated tasks. Furthermore, our method performs consistently across all three tasks. This means that the RGCRL algorithm can also provide more stable performance than the baselines.

V. CONCLUSION

This paper proposes a novel RGCRL scheme for end-to-end robotic control in adversarial and sparse reward environments.. Firstly, a mixed adversarial attack method is advanced to generate diverse adversarial perturbations on observations via combining white-box and black-box attacks. Secondly, a HER technique considering observation perturbations is developed to turn a failed experience into a successful one and generate the policy trajectories perturbed by the mixed adversarial attacks. Thirdly, an RGCAC method is introduced to learn goal-conditioned policies and keep the variations of the perturbed policy trajectories within bounds.

Evaluation of the policy models is carried out on the three robotic control tasks with adversarial attacks and sparse reward settings. The results demonstrate that the RGCRL approach enables the agent to learn efficiently from sparse rewards. Additionally, compared to the three baselines, the RGCRL agent has superior performance concerning the success rate and policy robustness under the mixed adversarial attacks.

While we have demonstrated the potential of the proposed RGCRL technique, some limitations remain. Consequently, future work involves evaluating our approach in more scenarios. In addition, the proposed algorithm will be applied to end-to-end robotic control tasks in the real world.

REFERENCES

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.
[2] S. Lyu and C. C. Cheah, "Data-driven learning for robot control with unknown jacobian," *Automatica*, vol. 120, p. 109120, 2020.
[3] 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 Transactions on Artificial Intelligence*, vol. 3, no. 2, pp. 90–109, 2021.
[4] D. Huang, C. Yang, Y. Pan, and L. Cheng, "Composite learning enhanced neural control for robot manipulator with output error constraints," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 209–218, 2019.
[5] Y. Yang, L. Juntao, and P. Lingling, "Multi-robot path planning based on a deep reinforcement learning dqn algorithm," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 177–183, 2020.
[6] J. Wang, J. Liu, W. Chen, W. Chi, and M. Q.-H. Meng, "Robot path planning via neural-network-driven prediction," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 3, pp. 451–460, 2021.
[7] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning." in *Robotics: Science and Systems*, 2019.
[8] S. Chen, Y. Cao, Y. Kang, P. Li, and B. Sun, "Deep feature representation based imitation learning for autonomous helicopter aerobatics," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 5, pp. 437–446, 2021.

[9] X. He, Y. Liu, C. Lv, X. Ji, and Y. Liu, "Emergency steering control of autonomous vehicle for collision avoidance and stabilisation," *Vehicle system dynamics*, vol. 57, no. 8, pp. 1163–1187, 2019.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] Q. Zhang, Y. Kang, Y.-B. Zhao, P. Li, and S. You, "Traded control of human-machine systems for sequential decision-making based on reinforcement learning," *IEEE Transactions on Artificial Intelligence*, 2021.

[12] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 124–134, 2018.

[13] X. He, H. Yang, Z. Hu, and C. Lv, "Robust lane change decision making for autonomous vehicles: An observation adversarial reinforcement learning approach," *IEEE Transactions on Intelligent Vehicles*, 2022.

[14] M. Neunert, A. Abdolmaleki, M. Wulfmeier, T. Lampe, T. Springenberg, R. Hafner, F. Romano, J. Buchli, N. Heess, and M. Riedmiller, "Continuous-discrete reinforcement learning for hybrid control in robotics," in *Conference on Robot Learning*. PMLR, 2020, pp. 735–751.

[15] X. B. Peng, G. Berseth, and M. Van de Panne, "Terrain-adaptive locomotion skills using deep reinforcement learning," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–12, 2016.

[16] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[17] G. Sartoretti, W. Paivine, Y. Shi, Y. Wu, and H. Choset, "Distributed learning of decentralized control policies for articulated mobile robots," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1109–1122, 2019.

[18] M. Han, L. Zhang, J. Wang, and W. Pan, "Actor-critic reinforcement learning for control with stability guarantee," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6217–6224, 2020.

[19] J. Xu, Y. Tian, P. Ma, D. Rus, S. Sueda, and W. Matusik, "Prediction-guided multi-objective reinforcement learning for continuous robot control," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 607–10 616.

[20] Y. Cui, T. Matsubara, and K. Sugimoto, "Pneumatic artificial muscle-driven robot control using local update reinforcement learning," *Advanced Robotics*, vol. 31, no. 8, pp. 397–412, 2017.

[21] Y. S. Shao, C. Chen, S. Kousik, and R. Vasudevan, "Reachability-based trajectory safeguard (rts): A safe and fast reinforcement learning safety layer for continuous control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3663–3670, 2021.

[22] Q. Fang, X. Xu, X. Wang, and Y. Zeng, "Target-driven visual navigation in indoor scenes using reinforcement learning and imitation learning," *CAAI Transactions on Intelligence Technology*, vol. 7, no. 2, pp. 167–176, 2022.

[23] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. Wiele, V. Mnih, N. Heess, and J. T. Springenberg, "Learning by playing solving sparse reward tasks from scratch," in *International conference on machine learning*. PMLR, 2018, pp. 4344–4353.

[24] R. Zhao, X. Sun, and V. Tresp, "Maximum entropy-regularized multi-goal reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7553–7562.

[25] C. Packer, P. Abbeel, and J. E. Gonzalez, "Hindsight task relabelling: Experience replay for sparse reward meta-rl," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2466–2477, 2021.

[26] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[27] A. Mandlekar, Y. Zhu, A. Garg, L. Fei-Fei, and S. Savarese, "Adversarially robust policy learning: Active construction of physically-plausible perturbations," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3932–3939.

[28] H. Xiong and X. Diao, "Safety robustness of reinforcement learning policies: A view from robust control," *Neurocomputing*, vol. 422, pp. 12–21, 2021.

[29] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2811–2817.

[30] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2817–2826.

[31] C. Tessler, Y. Efroni, and S. Mannor, "Action robust reinforcement learning and applications in continuous control," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6215–6224.

[32] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, "Robust deep reinforcement learning with adversarial attacks," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 2040–2042.

[33] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 31–36.

[34] N. Pitropakis, E. Panaousis, T. Giannetsos, E. Anastasiadis, and G. Loukas, "A taxonomy and survey of attacks against machine learning," *Computer Science Review*, vol. 34, p. 100199, 2019.

[35] Y. Li, L. Li, L. Wang, T. Zhang, and B. Gong, "Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3866–3876.

[36] Y. Tashiro, Y. Song, and S. Ermon, "Diversity can be transferred: Output diversification for white-and black-box attacks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[37] Q. Gallouédec, N. Cazin, E. Dellandréa, and L. Chen, "panda-gym: Open-source goal-conditioned environments for robotic learning," in *4th Robot Learning Workshop: Self-Supervised and Lifelong Learning at NeurIPS*, 2021.

[38] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[39] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[40] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[41] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[42] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

Xiangkun He (Member, IEEE) received his PhD degree in 2019 from the School of Vehicle and Mobility, Tsinghua University, Beijing, China. During 2019–2021, he was a Senior Researcher at Noah's Ark Lab, Huawei Technologies, China. He is currently a Research Fellow at the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore. He is the author or co-author of more than 40 peer-reviewed publications. His research interests include autonomous vehicles, reinforcement learning, robust machine learning, decision and control. He received many awards and honors, selectively including the Tsinghua University Outstanding Doctoral Thesis Award in 2019, Best Paper Finalist Award at 2020 IEEE International Conference on Mechatronics and Automation, 1st Class Outstanding Paper Award of China Journal of Highway and Transport in 2021, Huawei Major Technological Breakthrough Award in 2021, Huawei 2012 Lab Star Award in 2021, Huawei Hisilicon Chip Star Award in 2021, Best Paper Runner-Up Award at 2022 6th CAA International Conference on Vehicular Control and Intelligence, Runner-Up at Intelligent Algorithm Final of 2022 Alibaba Global "Future Vehicle" Intelligent Scene Innovation Challenge, and Wiley Outstanding Open Science Author. He is also a Reviewer for more than 30 journals and conferences, including IEEE TITS, IEEE TIV, IEEE TAI, IEEE TVT, IEEE TII, IEEE TIE, IEEE TTE, IEEE TCYB, IEEE/ASME TMECH, IEEE RAL, VSD, MSSP and CoRL.

Chen Lv (Senior Member, IEEE) is a Nanyang Assistant Professor at School of Mechanical and Aerospace Engineering, and the Cluster Director in Future Mobility Solutions, Nanyang Technological University, Singapore. He received his PhD degree at Department of Automotive Engineering, Tsinghua University, China in Jan 2016. He was a joint PhD researcher at UC Berkeley, USA during 2014-2015, and worked as a Research Fellow at Cranfield University, UK during 2016-2018. He joined NTU and founded the Automated Driving and Human-Machine System (AutoMan) Research Lab since June 2018. His research focuses on intelligent vehicles, automated driving, and human-machine systems, where he has contributed 2 books, over 100 papers, and obtained 12 granted patents. He serves as Associate Editor for IEEE T-ITS, IEEE TVT, and IEEE T-IV. He received many awards and honors, selectively including the Highly Commended Paper Award of IMechE UK in 2012, Japan NSK Outstanding Mechanical Engineering Paper Award in 2014, Tsinghua University Outstanding Doctoral Thesis Award in 2016, IEEE IV Best Workshop/Special Session Paper Award in 2018, Automotive Innovation Best Paper Award in 2020, the winner of Waymo Open Dataset Challenges at CVPR 2021, Machines Young Investigator Award in 2022, and Best Paper Runner Up Award at 2022 6th CAA International Conference on Vehicular Control and Intelligence.