arXiv:2404.13879v4 [cs.LG] 27 Jan 2026

Lipschitz-Regularized Critics Lead to Policy Robustness Against Transition Dynamics Uncertainty

Xulin Chen[1], Ruipeng Liu[1], Zhenyu Gan[2], Garrett Katz[1]

***Abstract*— Uncertainties in transition dynamics pose a critical challenge in reinforcement learning (RL), often resulting in performance degradation of trained policies when deployed on hardware. Many robust RL approaches follow two strategies: enforcing smoothness in actor or actor-critic modules with Lipschitz regularization, or learning robust Bellman operators. However, the first strategy does not investigate the impact of critic-only Lipschitz regularization on policy robustness, while the second lacks comprehensive validation in real-world scenarios. Building on this gap and prior work, we propose PPO-PGDLC, an algorithm based on Proximal Policy Optimization (PPO) that integrates Projected Gradient Descent (PGD) with a Lipschitz-regularized critic (LC). The PGD component calculates the adversarial state within an uncertainty set to approximate the robust Bellman operator, and the Lipschitz-regularized critic further improves the smoothness of learned policies. Experimental results on two classic control tasks and one real-world robotic locomotion task demonstrates that, compared to several baseline algorithms, PPO-PGDLC achieves better performance and predicts smoother actions under environmental perturbations.**

I. INTRODUCTION

While Reinforcement Learning (RL) has achieved remarkable success in robotic control tasks, its real-world performance on hardware is often hindered by the simulation-to-real (sim-to-real) gap. One source of this gap is the mismatch between simulated and real-world transition dynamics, due to system identification errors and idealized physics modeling. As a result, policies encounter out-of-distribution transition dynamics during hardware deployment, leading to performance degradation. Domain randomization [1], [2] alleviates this problem by training policies across a range of randomized physical parameters (e.g., mass, friction, and damping) in simulation. However, policies trained by domain randomization may fail to generalize beyond their predefined parameter distributions. This limitation is critical for real-world tasks, where environmental perturbations arise from many sources beyond the predefined domains.

To generally enhance policy robustness against uncertainties, recent research has turned to robust reinforcement learning (robust RL). One branch of this field addresses transition dynamics uncertainty by optimizing for the worst-case performance within a predefined uncertainty set [3], [4], [5]. Compared to standard RL, this approach produces conservative but reliable control policies that maintain performance under dynamic mismatches, often at the cost of nominal efficiency. However, directly solving this problem is often intractable for real-world problems due to complex dynamics. Kuang *et al.* [6] proposed a framework called State-Conservative Markov Decision Process (SC-MDP), which converts the problem into a minimization of the value function over a local region of the state space. Their accompanying solution, called Gradient Based Regularizer (GBR), uses a first-order Taylor expansion to quickly approximate the minimum. Although this approach offers a more tractable design for updating robust RL policies, it still suffers from several limitations. First, the performance of GBR suffers from unregularized gradients in the value network when the local region is relatively large. Furthermore, the efficacy of SC-MDP has not been validated on more complicated tasks such as real-world robotic control problems.

Lipschitz regularization is an effective strategy for enforcing neural network smoothness, with proven success in fields such as computer vision [7] and RL. In RL, it has been applied to smooth policy outputs against state input perturbations [8], [9] or to stabilize the entire actor-

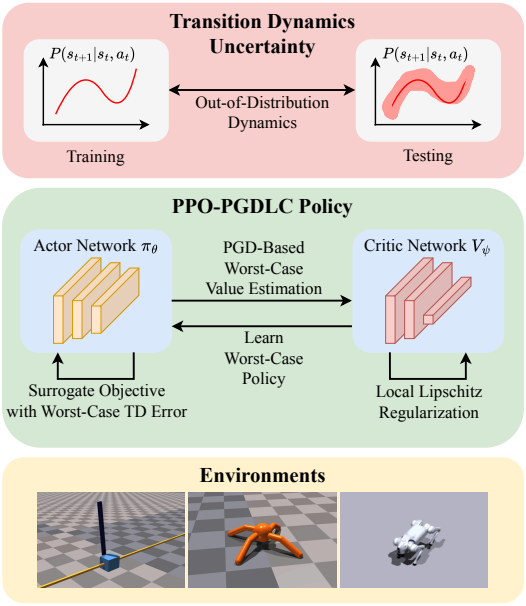


Fig. 1: Overview of the PPO-PGDLC framework. PGD estimates the worst-case value within a bounded uncertainty set, while the LC enforces local smoothness in the critic to stabilize updates. Their integration within PPO enhances policy robustness to transition dynamics uncertainty and improves sim-to-real reliability.

[1]Xulin Chen, Ruipeng Liu and Garrett Katz are with the Department of Electrical Engineering & Computer Science, Syracuse University, Syracuse, NY 13210, USA. {xchen168, rliu02, gkatz01}@syr.edu

[2]Zhenyu Gan is with the Department of Mechanical & Aerospace Engineering, Syracuse University, Syracuse, NY 13210, USA. zgan02@syr.edu

critic module [10]. However, the independent effect of a Lipschitz-regularized critic remains underexplored in these previous works. Besides, within the SC-MDP framework, we hypothesize that learning a Lipschitz-regularized critic enforces local smoothness in the value function, therefore reducing the pessimism of the approximated worst-case value function and leading to more robust policy performance.

Motivated by SC-MDP and Lipschitz regularization, this work addresses two questions:

- How sensitive is robust RL policy stability to the method for approximating the value function's local minimum?
- Does learning Lipschitz-regularized critic networks enhance policy robustness against transition dynamics uncertainty?

In this work, we propose PPO-PGDLC, a framework shown in Fig. 1 that combines PGD-based worst-case value estimation with a LC in the PPO [11] framework. The PGD component approximates the robust Bellman operator by explicitly searching for adversarial states within a bounded uncertainty set, while the Lipschitz regularization constrains the critic's gradients to prevent overreaction to local perturbations and to stabilize policy updates. Together, these mechanisms produce a principled and computationally tractable formulation of robust RL that connects the theoretical guarantees of SC-MDP with practical actor–critic learning. Extensive experiments on classic control benchmarks and real quadrupedal locomotion demonstrate that PPO-PGDLC improves both robustness to transition perturbations and action smoothness relative to existing robust-RL baselines, offering new insight into how critic smoothness contributes to reliable sim-to-real transfer.

II. RELATED WORK

A. Robust Reinforcement Learning

In contrast to traditional RL, which typically assumes a static and well-defined environment, robust RL explicitly accounts for disturbances, uncertainties, and even adversarial attacks. These may arise in the state space [12], [13], action space [14], [15], reward function [16], or transition dynamics [6], [17], [18]. When perturbations are properly constrained, techniques such as adversarial training, robust optimization, and game-theoretic approaches can be employed to mitigate their effects [19]. Robust RL is relevant both to the reality gap and to unforeseen circumstances that occur during hardware deployment. Such circumstances include unintentional accidents in safety-critical domains such as power grids and autonomous vehicles [20], and intentional deceptions by malicious actors [13], [14], [21], [22].

B. Lipschitz Continuity and Regularization

Modern deep learning systems exhibit sensitivity to small, strategically chosen input perturbations, which can substantially change their outputs [23], [24]. This vulnerability is similarly significant in RL [8], [25], [10], as modern RL methods use deep function approximation for policies and value functions. A deep model's Lipschitz constant quantifies its smoothness, and Lipschitz regularization enhances model stability by bounding its Lipschitz constant [26], [27], [28]. Lipschitz regularization can be implemented globally, constraining model behavior across the entire input space, or locally, focusing on regions near training data points [27], [29]. However, Lipschitz regularization presents a fundamental trade-off: excessive regularization causes undue smoothness that impairs performance on the primary task [26].

III. PRELIMINARIES

A. Lipschitz Continuity and Regularization

A function $f(\mathbf{x}) : \mathcal{X} \to \mathcal{Y}$ is considered to be L-Lipschitz continuous (L-Lipschitz for short) if there is a constant $L \geq 0$ such that for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, the following inequality holds

$$d_{\mathcal{Y}}(f(\mathbf{x}_1), f(\mathbf{x}_2)) \leq L \cdot d_{\mathcal{X}}(\mathbf{x}_1, \mathbf{x}_2), \tag{1}$$

where $d_{(\cdot)}$ is a distance metric defined on the given space. A lower Lipschitz constant indicates that the function is smoother. The Lipschitz condition also implies that, for all $\mathbf{x} \in \mathcal{X}$, the norm of the gradient (if it exists) is upper bounded by L, meaning $||\nabla_{\mathbf{x}} f(\mathbf{x})|| \leq L$. When f is characterized by a neural network, this property can be used to enforce local smoothness by augmenting the original loss function $\mathcal{L}^{\text{ori}}$ with a penalty term on the gradient norm, i.e.:

$$\mathcal{L}^{\text{new}}(f) = \mathcal{L}^{\text{ori}}(f) + \lambda \cdot ||\nabla_{\mathbf{x}} f(\mathbf{x})||. \tag{2}$$

B. Robust Markov Decision Process

A Robust Markov Decision Process (RMDP) [30] extends the standard MDP by allowing the transition kernel P to vary within an uncertainty set $\mathcal{P}$ ($P \in \mathcal{P}$). An RMDP is formally described by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, where $\mathcal{S} \subseteq \mathbb{R}^m$ and $\mathcal{A} \subseteq \mathbb{R}^n$ are the state and action spaces, $\mathcal{P}$ represents the set of possible transition dynamics, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discount factor for future rewards. Given a distribution $\mu(s_0)$ for the initial state s_0, the objective is to find a policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ which maximizes the infimum expected return across all transition dynamics in $\mathcal{P}$

$$\mathcal{J}_{\mathcal{P}}(\pi) \triangleq \inf_{\hat{P} \in \mathcal{P}} \mathbb{E}_{\substack{s_0 \sim \mu(s_0) \\ a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim \hat{P}(\cdot|s_t, a_t)}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right], \tag{3}$$

where $\Delta_{(\cdot)}$ denotes the set of probability distributions over the given space. For a fixed policy π, the state-value $V^{\pi}_{\mathcal{P}}(s)$ and action-value $Q^{\pi}_{\mathcal{P}}(s, a)$ are defined with respect to the worst-case transition dynamics

$$V^{\pi}_{\mathcal{P}}(s) = \inf_{\hat{P} \in \mathcal{P}} \mathbb{E}_{\pi, \hat{P}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s \right], \tag{4}$$

$$Q^{\pi}_{\mathcal{P}}(s, a) = \inf_{\hat{P} \in \mathcal{P}} \mathbb{E}_{\pi, \hat{P}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, a_0 = a \right], \tag{5}$$

and $Q^{\pi}_{\mathcal{P}}(s, a)$ is the fixed point of the following robust Bellman operator

$$\mathcal{T}^{\pi}_{\mathcal{P}} Q(s, a) \triangleq R(s, a) + \gamma \inf_{\hat{P} \in \mathcal{P}} \mathbb{E}_{\substack{\hat{s} \sim \hat{P}(\cdot|s,a) \\ \hat{a} \sim \pi(\cdot|\hat{s})}} \left[Q(\hat{s}, \hat{a}) \right]. \tag{6}$$

To make the problem well-posed, it is necessary to impose appropriate constraints on the uncertainty set $\mathcal{P}$. A widely used approach [4], [6] is defining a p^{th}-order Wasserstein ϵ-ball $\mathcal{P}_{\epsilon,p}$ centered at a given nominal dynamics $P_0(\cdot|s,a)$. In summary, RMDPs enhance policy robustness at the expense of average-case performance.

IV. METHODOLOGY

A. Problem Formulation: Worst-Case Value Estimation

Directly solving the robust Bellman operator in (6) is often infeasible for real-world problems, due to complicated dynamics and the difficulty of accurately modeling uncertainties such as variations in weights, friction, or terrain. SC-MDP [6] converts the problem into disturbances within the state space, yielding the modified robust Bellman operator:

$$\mathcal{T}^{\pi}_{\mathcal{P}} Q(s,a) \triangleq R(s,a) + \gamma \inf_{\hat{s} \in B_{\epsilon,p}(s')} \mathbb{E}_{\hat{a}\sim\pi(\cdot|\hat{s})} \left[Q(\hat{s},\hat{a}) \right], \quad (7)$$

where the state transition (s,a,s') is performed under the nominal dynamics P_0, and $B_{\epsilon,p}(s') \triangleq \{\hat{s} \in \mathcal{S} : d_{\mathcal{S}}(\hat{s} - s')^p \leq \epsilon\}$ is an ϵ-ball induced by a distance metric $d_{\mathcal{S}}$ on the state space $\mathcal{S}$. This formulation is equivalent to the original robust Bellman operator in (6) if the dynamics in the uncertainty set $P_{\epsilon,p}$ are deterministic and bounded by Wasserstein distance. Although the equivalence may not hold for stochastic dynamics [6], (7) still simplifies the update rule for robust RL policies by directly selecting the most adversarial state transition and eliminating the need for modeling transition dynamics uncertainty. Even if the magnitude of real perturbations exceeds ϵ, policies obtained via learning (7) exhibit more conservative behavior than standard RL policies.

Actor-critic algorithms in standard RL approximate either the state-value $V(s)$ or the action-value $Q(s,a)$. To adapt these algorithms for robust RL, critic networks must approximate the robust Bellman operator. Our work adapts the on-policy algorithm PPO [11] to the robust RL setting by approximating the robust Bellman operator in (7). Choosing the ∞-norm as the distance metric for $B_{\epsilon,p}(s)$, we formalize the problem as Worst-Case Value Estimation (WCVE). Specifically, given a state-value function $V(s)$, a solution $h(s,\epsilon,V)$ of WCVE is a state within a L_∞-norm ball $B_{\epsilon,\infty}(s)$ around the input state s, that corresponds to the infimum state-value:

$$h(s,\epsilon,V) = \arg_{\hat{s}\in B_{\epsilon,\infty}(s)} \inf V(\hat{s}). \quad (8)$$

B. PGD and Critic Lipschitz Regularization

Gradient Based Regularizer [6] does not solve $h(s,\epsilon,V)$ in (8) exactly, but provides a fast approximation for the infimum state-value: $\inf_{\hat{s}\in B_{\epsilon,\infty}(s)} V(\hat{s}) \approx V(s) - \epsilon||\nabla_s V(s)||_1$. However, the learned critic network is typically nonconvex, so the approximation is not always reliable. We choose to solve WCVE using PGD [31], an iterative optimization algorithm that searches for solutions within the constraint set. In each iteration, PGD updates the state towards the direction that decreases the state-value, then projects the state back into the L_∞-norm ball. Algorithm 1 presents the pseudocode for solving WCVE with PGD.

PGD does not mitigate the degradation of policy performance when the gradients of V's deep function approximator are large and inaccurate. To address this limitation, a Lipschitz-regularized critic is considered, which is justified if the true RMDP value functions are Lipschitz. Prior work [25], [32] has shown that when the reward function, policy, and transition dynamics satisfy Lipschitz continuity, the resulting state-value and state–action value functions are also Lipschitz continuous.

In many continuous control tasks, the reward function can be designed to be Lipschitz w.r.t. the state-action space (e.g. linear functions, or differentiable functions on compact domains). However, transition dynamics are often not globally Lipschitz because of impulse events that result in discontinuous state changes. This issue can be mitigated by either designing compliant models for impulse events [33] or considering local Lipschitz continuity in dynamical systems [34]. We choose the second idea and apply local Lipschitz regularization to the critic network, as in (2).

Algorithm 1 PGD solver $h(s,\epsilon,V)$ for WCVE

Input: State-value function V, input state s, number of PGD steps N, PGD step size α, constraint set $B_{\epsilon,\infty}(s)$.
Output: State with the minimum state-value in $B_{\epsilon,\infty}(s)$.

```
s^0 ← s
for i = 0 ··· N − 1 do
    s^{i+1} ← s^i − α · sign(∇_s V(s^i))
    s^{i+1} = Projection(s^{i+1}, B_{ε,∞}(s))
end for
Return h(s, ε, V) = s^N.
```

C. Practical Implementation

This section introduces the implementation of PPO-PGDLC, which incorporates the PGD solver and a Lipschitz-regularized critic into PPO. Recall that PPO employs truncated generalized advantage estimation [35] to compute variance-reduced advantage estimators. Given a trajectory $\tau = (s_0, a_0, s_1, \cdots, s_T)$, the advantage estimator A_t of time index $t \in [0,T)$ is defined as the exponentially weighted sum of temporal difference (TD) errors

$$\begin{aligned} A_t &= \delta_t + (\gamma\xi)\delta_{t+1} + \cdots + (\gamma\xi)^{T-t+1}\delta_{T-1}, \\ \text{with } \delta_t &= R(s_t,a_t) + \gamma V_\psi(s_{t+1}) - V_\psi(s_t), \end{aligned} \quad (9)$$

where $\gamma, \xi \in (0,1)$ are the discount factor for the future reward and generalized advantage estimation, respectively, and δ_t is the TD error at time t computed using the critic network V_ψ. In PPO-PGDLC, we introduce a modified advantage estimator $\hat{A}^h_t$, which calculates the worst-case TD errors $\hat{\delta}^h_t$ based on the PGD solver h in Algorithm 1:

$$\begin{aligned} \hat{A}^h_t &= \hat{\delta}^h_t + (\gamma\xi)\hat{\delta}^h_{t+1} + \cdots + (\gamma\xi)^{T-t+1}\hat{\delta}^h_{T-1}, \\ \text{with } \hat{\delta}^h_t &= R(s_t,a_t) + \gamma V_\psi(h(s_{t+1},\epsilon,V_\psi)) - V_\psi(s_t). \end{aligned} \quad (10)$$

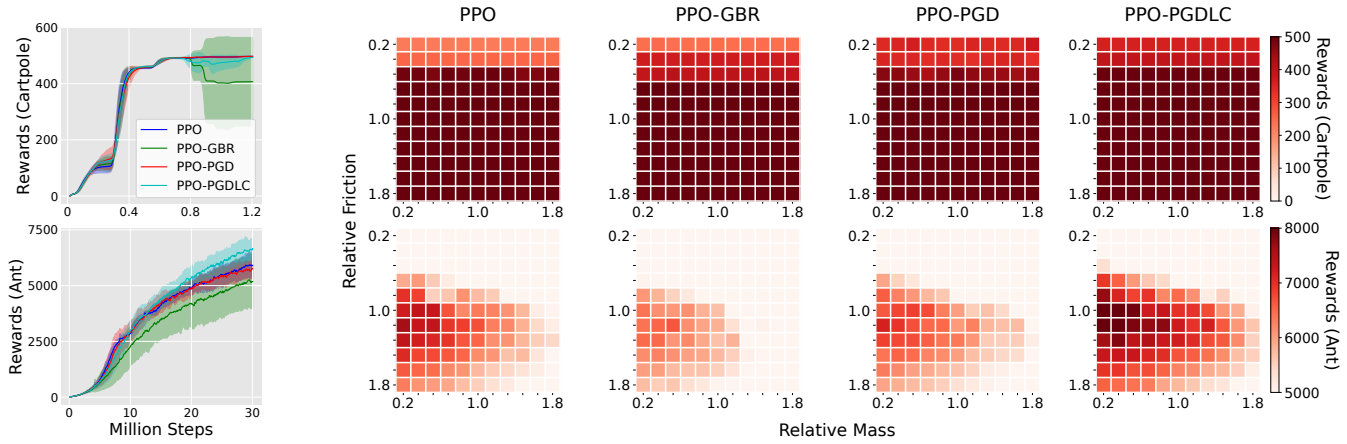


Fig. 2: The training curves (left) and heatmaps (right) for Cartpole (top row) and Ant (bottom row), where $\epsilon = 0.003$ for PPO-GBR, PPO-PGD and PPO-PGDLC. When drawing the heatmaps, we collect totally 80 episodes with 4 trained policies and average their episode rewards for each environment. Redder color indicates higher reward. Overall, PPO-PGDLC maintains higher returns and greater robustness across mass–friction perturbations compared with all baselines, highlighting the benefit of integrating PGD-based value estimation with a Lipschitz-regularized critic.

The loss function $\mathcal{L}(\theta)$ for the actor network π_θ is formulated by incorporating $\hat{A}_t^h$ into the PPO surrogate objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\Big[\min\Big(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t^h, \text{clip}\big(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\eta, 1+\eta\big)\hat{A}_t^h\Big)\Big], \tag{11}$$

where $\eta \in (0,1)$ is the constant clipping ratio. The critic loss $\mathcal{L}(\psi)$ consists of two terms: a residual error $\mathcal{L}^{\text{pred}}(\psi)$ of fitting rewards-to-go $\bar{R}_t$, and a local Lipschitz regularization term $\mathcal{L}^{\text{Lips}}(\psi)$ weighted by a hyperparameter λ. The parameter λ controls the strength of smoothness enforcement on the critic: a smaller value yields weaker regularization and allows higher gradient variability, whereas a larger value imposes stronger smoothness that can suppress gradient noise but may introduce bias in value estimation.

$$\mathcal{L}(\psi) = \mathcal{L}^{\text{pred}}(\psi) + \lambda \cdot \mathcal{L}^{\text{Lips}}(\psi), \tag{12}$$

$$\mathcal{L}^{\text{pred}}(\psi) = \mathbb{E}_{s_t\sim\mathcal{D}}\left[\frac{1}{2}\left(V_\psi(s_t) - \bar{R}_t\right)^2\right], \tag{13}$$

$$\mathcal{L}^{\text{Lips}}(\psi) = \mathbb{E}_{s_t\sim\mathcal{D}}\left[||\nabla_s V_\psi(s_t)||_1^2\right]. \tag{14}$$

Here the n-step rewards-to-go is computed as $\bar{R}_t = \gamma^n V_\psi(s_{t+n}) + \sum_{i=0}^{n-1}\gamma^t R(s_{t+i}, a_{t+i})$, and $\mathcal{L}^{\text{Lips}}(\psi)$ penalizes the 1-norm of critic gradient as in (2). The complete pseudocode of PPO-PGDLC is presented in Algorithm 2.

V. EXPERIMENTS

Experiments are conducted on three control problems: two classic benchmarks, Cartpole and Ant, in Isaac Gym [36], and a quadrupedal robot locomotion task using the Unitree Go2, which was trained in simulation and deployed on real hardware [37]. The experiments primarily aim to compare policy robustness under varying transition dynamics and to evaluate policy smoothness and performance.

Algorithm 2 PPO-PGDLC

Input: Initial actor and critic parameters θ, ψ, rollout buffer $\mathcal{D}$, PGD solver h, learning rate β_θ, β_ψ.
Output: Update θ and ψ.
for every iteration **do**
  Collect a set of trajectories $\{\tau_1, \cdots \tau_n\}$ running π_θ and add them to $\mathcal{D}$.
  For all transitions in $\mathcal{D}$, compute rewards-to-go $\bar{R}_t$ and advantage estimator $\hat{A}_t^h$ with V_ψ and h.
  for every gradient step **do**
    $\theta \leftarrow \theta - \beta_\theta \nabla_\theta \mathcal{L}(\theta)$
    $\psi \leftarrow \psi - \beta_\psi \nabla_\psi \mathcal{L}(\psi)$
  end for
end for

A. Metric for Quantifying Policy Robustness

We quantify the robustness of learned policies against transition dynamics uncertainty. We generate an $M \times M$ grid of perturbed environments $\mathcal{P} = \{P_{i,j} | 1 \le i,j \le M\}$ by varying two physical parameters, such as body mass and surface friction. We then design a metric called ρ-robustness. This metric captures the worst-case performance within a specific "attack radius" ρ on the perturbation grid measured by the L_∞-norm distance from the nominal environment at the grid center. Given a policy π and $\rho \in \{0, 1, \cdots, \lfloor \frac{M}{2} \rfloor\}$, π's ρ-robustness is the minimum of the average episode reward over all environments at radius ρ:

$$\rho\text{-robustness} = \min_{(i,j)\in\mathcal{G}_\rho} \mathbb{E}_{\tau\sim\pi, P_{i,j}}\left[\sum_{t=1}^{T} R(s_t, a_t)\right], \tag{15}$$

where $\mathcal{G}_\rho = \{(i,j) : ||(i,j) - (\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)||_\infty = \rho\}$ is the set of cells with L_∞-norm distance ρ to the grid center. Although environments at the same ρ may differ in specific parameter values, this metric enables consistent comparison of worst-

TABLE I: ρ-robustness metrics for all methods on Cartpole (top) and Ant (bottom); higher values indicate better robustness. Each entry reports the average episode reward over perturbed environments at radius ρ for different uncertainty bounds ϵ. Overall, PPO-PGDLC achieves the highest ρ-robustness across most tested radii and ϵ values, demonstrating stronger resilience to transition-dynamics perturbations compared with all baseline algorithms.

Method	**PPO**	**PPO-GBR**					**PPO-PGD**					**PPO-PGDLC (Ours)**				
ρ \ ϵ	–	0.001	0.003	0.005	0.007	0.01	0.001	0.003	0.005	0.007	0.01	0.001	0.003	0.005	0.007	0.01
0	498.46	485.16	**499.11**	498.90	497.64	417.98	454.38	498.17	498.86	497.91	495.90	497.85	498.11	487.68	498.37	497.03
1	498.53	485.93	**499.03**	491.70	496.90	368.73	398.28	498.03	496.04	497.36	493.61	497.67	497.95	479.79	498.02	494.21
2	498.35	458.64	**498.35**	484.74	491.13	362.82	372.98	497.91	496.20	495.98	491.81	496.75	497.85	473.41	492.85	485.27
3	469.98	312.67	389.71	484.37	368.36	367.31	366.20	425.62	461.67	428.17	412.21	401.61	**494.07**	476.45	486.14	391.46
4	252.30	190.31	366.12	306.90	235.82	355.25	347.56	332.35	352.66	190.94	363.45	198.69	390.34	344.58	**417.69**	249.88
5	223.29	179.05	240.70	306.05	247.43	212.51	161.07	333.17	318.14	196.87	320.19	214.75	**357.11**	142.79	313.83	250.34

Method	**PPO**	**PPO-GBR**					**PPO-PGD**					**PPO-PGDLC (Ours)**				
ρ \ ϵ	–	0.001	0.003	0.005	0.007	0.01	0.001	0.003	0.005	0.007	0.01	0.001	0.003	0.005	0.007	0.01
0	6339.91	5553.56	5700.36	6112.06	1666.71	174.35	6346.54	5997.84	5696.34	468.64	180.53	**7010.88**	6985.20	5822.36	6941.87	3076.43
1	6220.70	5800.68	5200.72	5655.35	1438.12	171.33	5667.91	5754.71	5373.74	417.02	165.94	6646.53	**6885.73**	5387.60	6548.76	2694.21
2	5536.80	5263.08	4760.70	5052.55	1438.12	165.99	5366.59	5399.01	4828.06	352.18	149.04	5996.85	**6262.51**	5118.59	5725.31	2541.31
3	4993.21	4722.58	4235.75	4430.63	1472.58	163.42	4824.69	4714.49	4261.62	294.23	143.33	5210.44	**5539.27**	4545.94	4965.86	2460.67
4	4421.72	4227.00	3796.47	3914.00	1439.15	166.53	4334.81	4211.22	3920.84	275.26	145.93	4519.73	**4794.98**	4081.62	4365.67	2239.75
5	4004.55	3804.88	3419.03	3544.69	1396.46	175.12	3904.32	3763.67	3497.29	265.72	152.83	4040.66	**4223.23**	3667.65	3898.59	2123.52

case performance under constrained dynamics uncertainty.

B. Experiments on Classic Control Tasks

1) Baseline Algorithms: We compare PPO-PGDLC to three baseline algorithms: the standard RL baseline PPO, and two robust RL variants, PPO-GBR and PPO-PGD. PPO-GBR employs the first-order state-value infimum approximation described in Section IV-B. Both PPO-GBR and PPO-PGD set the regularization weight ($\lambda = 0$) so that no Lipschitz regularization is applied to the critic. PPO-PGD serves as an ablated version of PPO-PGDLC that retains the PGD-based worst-case value estimation but omits the Lipschitz-regularized critic.

2) Training Details: Both actor and critic networks are multilayer perceptrons with hidden sizes $[256, 256]$ and learning rates of 3×10^{-4}. For robust RL algorithms, the uncertainty-set radius is varied as $\epsilon \in \{0.001, 0.003, 0.005, 0.007, 0.01\}$. To solve WCVE, PPO-PGD and PPO-PGDLC employ 10 PGD steps with a step size of $\epsilon/10$. In PPO-PGDLC, the weight λ of local regularization term $\mathcal{L}^{\text{Lips}}(\psi)$ is set to 0.001 for both Cartpole and Ant. Policies are trained in the nominal environment for 1.2×10^6 transitions on Cartpole and 3×10^7 transitions on Ant. Each configuration is repeated 4 times with different random seeds, and the evaluation results are averaged.

3) Performance and robustness results: Fig. 2 shows the training rewards for all four algorithms. On Cartpole, all policies achieve comparable performance, while on Ant, PPO-PGDLC converges to a higher final reward than the other algorithms. Next, we compare policy robustness against dynamics perturbations. We rescale the mass and static friction of all rigid bodies with factors within $[0.2, 1.8]$, creating $11 \times 11 = 121$ environments. These two parameters are chosen because they directly influence the robot's inertial and contact properties. Heatmaps in Fig. 2 visualize the performance across all perturbed environments. On Cartpole, when mass scale is in $\{0.2, 0.36, 0.52\}$, PPO-PGD and PPO-PGDLC achieve higher rewards than PPO-GBR and PPO, and all algorithms show comparable rewards for other mass scales. On Ant, PPO-PGDLC achieves higher rewards when mass scale is in $[0.2, 1.0]$ and friction scale is in $[1.0, 1.8]$, compared to the baselines. The heatmaps of PPO-PGD and PPO-PGDLC demonstrate the benefit of learning a Lipschitz-regularized critic for the WCVE problem. Table I presents the value of ρ-robustness for all algorithms and ϵ values. We summarize our findings as follows. Firstly, PPO-PGDLC achieves the highest ρ-robustness value in 3 out of 6 radii on Cartpole, and across all 6 radii on Ant. Compared to PPO, PPO-PGDLC achieves 65.55% ($\epsilon = 0.007, \rho = 4$) and 59.93% ($\epsilon = 0.003, \rho = 5$) higher ρ-robustness value on Cartpole, and 9.87% higher ρ-robustness value on average for all radii values on Ant. Additionally, compared to PPO-GBR and PPO-PGD, PPO-PGDLC improves policy performance for all radii when $\epsilon \leq 0.005$, with overall significant improvements when $\epsilon \geq 0.007$. Finally, comparing all four algorithms reveals a trade-off between robustness and performance. Training on overly large uncertainty sets (e.g. $\epsilon \in \{0.007, 0.01\}$) still impairs policy performance, which is consistent with the conclusion in [6], but learning a Lipschitz-regularized critic helps mitigate this effect.

C. Experiments on the Unitree Go2 Robot

We also test PPO-PGDLC on a real-world quadrupedal locomotion task, in both simulation and hardware.

1) Simulation Setup: Our simulation environment is based on [38], which adapts *Walk-These-Ways* [39] to learn quadrupedal gaits with a Unitree Go2 robot. An observation vector $o_t \in \mathbb{R}^{70}$ includes torso orientation ($\mathbb{R}^3$), current joint positions ($\mathbb{R}^{12}$), current joint velocities ($\mathbb{R}^{12}$), actions from the last two time steps ($\mathbb{R}^{24}$), periodic clock input for four feet ($\mathbb{R}^4$), and behavior-specific commands ($\mathbb{R}^{15}$). An action vector $a_t \in \mathbb{R}^{12}$ represents the desired joint positions, which

TABLE II: Estimated LLC for the actor and critic networks during trotting with commands $[v_x^{\text{cmd}}, v_y^{\text{cmd}}, \omega_{\text{yaw}}^{\text{cmd}}] = [0.5, 0, 0]$. Parenthetical values show the relative decrease compared to the LLC of PPO actor/critic. Given a fixed disturbance bound ϵ, larger regularization weights λ enforce stronger smoothing and lead to substantially lower LLCs, indicating reduced sensitivity to state perturbations.

Method	λ	Actor LLC	Critic LLC
PPO	–	56.728	5.303
PPO-PGDLC ($\epsilon = 0.001$)	10^{-2}	41.071 (27.6%)	0.179 (96.6%)
	10^{-3}	47.712 (15.9%)	0.416 (92.2%)
	10^{-4}	49.520 (12.7%)	1.425 (73.1%)
	5×10^{-5}	52.761 (7.0%)	2.242 (57.7%)

are converted to joint torques via PD control with gains $k_p = 25$ and $k_d = 0.6$. We use the reward functions from [39] to learn two gaits (trotting and bounding) while enforcing gait stability and smoothness, with the following ranges for command velocities $[v_x^{\text{cmd}}, v_y^{\text{cmd}}, \omega_{\text{yaw}}^{\text{cmd}}]$: $v_x^{\text{cmd}} \in [-1, 1]$m/s for forward velocity, $v_y^{\text{cmd}} \in [-0.6, 0.6]$m/s for lateral velocity, and $\omega_{\text{yaw}}^{\text{cmd}} \in [-1, 1]$rad/s for yaw rate. Both simulation and hardware experiments run at a control frequency of 50Hz.

2) Policy Setup: We compare PPO-PGDLC against PPO, but we modify the PGD solver of PPO-PGDLC due to the following reason. Recall that the observation space defined in Section V-C.1 includes velocity and gait commands. Since these commands are chosen by the human operator and not subject to noise in the environment, it is not realistic for them to be adversarially perturbed in Algorithm 1. Similar reasoning applies to the parts of the state vector containing previous actions and clock inputs. Therefore, we mask the corresponding dimensions of the state vector so that they are not altered by the PGD solver. For PPO-PGDLC policies, we select $\epsilon = 0.001$ and perform a grid search for the hyperparameter $\lambda \in \{5 \times 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$. At each time step t, the policy receives the 5 most recent observations $\{o_{t-4}, \cdots, o_t\}$ to predict an action a_t. Both actor and critic networks are multi-layer perceptrons with hidden sizes $[512, 256, 128]$ and ReLU activations. Each policy is trained for 1.97×10^8 transitions in total. To isolate the impact of PGDLC on robustness, domain randomization is *not* applied during policy training.

3) Additional Evaluation Metrics: To evaluate policy smoothness and performance, we additionally choose three metrics: Action Smoothness (AS), Second-order Fluctuation Ratio (SFR) [9], and Velocity Tracking Error (VTE)

$$\text{AS} = \mathbb{E}_{\tau\sim\pi}\left[\frac{1}{T}\sum_{t=1}^{T}||a_t - a_{t-1}||_1\right], \tag{16}$$

$$\text{SFR} = \mathbb{E}_{\tau\sim\pi}\left[\frac{1}{T}\sum_{t=1}^{T}||a_t - 2a_{t-1} + a_{t-2}||_1\right], \tag{17}$$

$$\text{VTE} = \mathbb{E}_{\tau\sim\pi}\left[\frac{1}{T}\sum_{t=1}^{T}||v_{xy,t} - v_{xy,t}^{\text{cmd}}||_2\right], \tag{18}$$

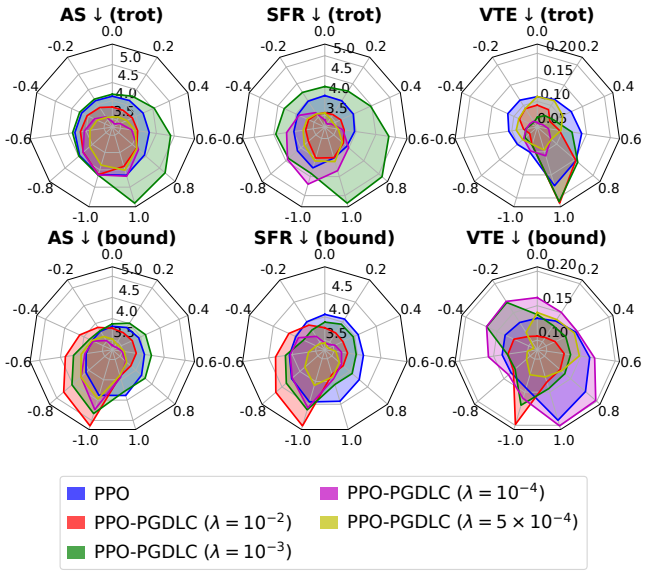


Fig. 3: Radar charts visualizing AS, SFR and VTE across different command velocities. Each vertex represents a specific command velocity, with the distance from the center indicating the metric value. For all metrics, a smaller value denotes better. PPO-PGDLC with $\lambda = 5 \times 10^{-5}$ achieves the lowest AS, SFR, and VTE across both trotting and bounding gaits, demonstrating smoother control and more accurate velocity tracking compared with the baseline policies.

where AS quantifies abrupt changes in consecutive actions to access action smoothness, SFR evaluates rapidity of action oscillations to capture higher-order smoothness, and VTE measures the velocity tracking on the ground plane.

4) Estimated Local Lipschitz Constant: To validate the effectiveness of local Lipschitz regularization in PPO-PGDLC, we estimate the local Lipschitz constant (LLC) of trained policies using *auto_LiRPA* [40]. This library is based on automatic linear relaxation to compute the bounds of network output within bounded input perturbations. For each policy, we collect 250 state transitions with trotting at commands $[v_x^{\text{cmd}}, v_y^{\text{cmd}}, \omega_{\text{yaw}}^{\text{cmd}}] = [0.5, 0, 0]$, then calculate the LLC of the actor and critic networks with L_∞-norm perturbation bound 0.001. The results are shown in Table II, and we observe that PPO-PGDLC achieves 57.7% $\sim$ 96.6% lower critic LLC and 7.0% $\sim$ 27.6% lower actor LLC than PPO, as λ increases. Lower actor LLC means PPO-PGDLC predicts smoother actions than PPO under state input perturbations, but it does not imply robustness against dynamics perturbation (see Section V-C.6).

5) Policy Smoothness and Performance: We also evaluate policy smoothness and performance for different command velocities and gaits on the nominal dynamics. For each policy, we collect 100 episodes for bounding and trotting gait with commands $v_x^{\text{cmd}} \in [-1, 1]$ m/s, $v_y^{\text{cmd}} = 0$ m/s and $\omega_{\text{yaw}}^{\text{cmd}} = 0$ rad/s. The radar charts in Fig. 3 present the results for AS, SFR and VTE. PPO-PGDLC with $\lambda = 5 \times 10^{-5}$ overall outperforms the other policies across the three metrics

TABLE III: ρ-robustness for Go2 policies (bounding and trotting with commands $[v_x^{\text{cmd}}, v_y^{\text{cmd}}, \omega_{\text{yaw}}^{\text{cmd}}] = [0.5, 0, 0]$. Each value represents the average episode reward over perturbed environments defined by variations in body mass and ground-contact friction; higher values indicate stronger robustness. Overall, PPO-PGDLC surpasses PPO across most radii, while the optimal regularization weight λ differs by gait, with $\lambda = 10^{-3}$ performing best for bounding and $\lambda = 5 \times 10^{-5}$ for trotting, indicating that the most effective level of critic smoothing depends on the task dynamics.

Method	λ	**Bounding**				**Trotting**			
		$\rho = 0$	$\rho = 1$	$\rho = 2$	$\rho = 3$	$\rho = 0$	$\rho = 1$	$\rho = 2$	$\rho = 3$
PPO	–	11.286	11.172	10.516	9.041	15.207	14.839	13.958	**12.377**
PPO-PGDLC ($\epsilon = 0.001$)	10^{-2}	13.467	12.373	10.827	**9.093**	15.660	15.045	13.671	11.740
	10^{-3}	**14.194**	**13.610**	**11.563**	3.078	12.598	12.180	11.671	10.782
	10^{-4}	9.971	8.557	7.399	6.413	15.218	14.637	13.420	11.961
	5×10^{-5}	13.452	13.085	11.308	6.771	**15.817**	**15.620**	**14.352**	11.153

TABLE IV: Comparison of PPO and PPO-PGDLC ($\lambda = 5 \times 10^{-5}$) across different payloads and commands. Each entry reports AS, SFR, and VTE at commanded forward velocities $v_x^{\text{cmd}} \in [0.6, 1.0]$ m/s with $v_y^{\text{cmd}} = 0$ m/s and $\omega_{\text{yaw}}^{\text{cmd}} = 0$ rad/s. PPO-PGDLC exhibits consistently smoother actions (lower AS in all tests and lower SFR in 5 of 6 cases) and comparable or improved velocity tracking (lower VTE in half of the tests), demonstrating better transfer stability under payload variation without additional fine-tuning.

Weight	v_x^{cmd}(m/s)	**AS** ↓		**SFR** ↓		**VTE** ↓	
		PPO	**PPO-PGDLC**	**PPO**	**PPO-PGDLC**	**PPO**	**PPO-PGDLC**
+2kg	0.6	4.501	**4.306**	**3.725**	3.767	**0.242**	0.258
	0.8	4.757	**4.488**	3.942	**3.741**	0.354	**0.306**
	1.0	5.068	**4.685**	4.252	**3.812**	0.411	**0.396**
+4kg	0.6	4.841	**4.498**	3.742	**3.739**	**0.432**	0.473
	0.8	5.067	**4.686**	4.023	**3.855**	**0.517**	0.584
	1.0	5.424	**4.913**	4.371	**4.087**	0.668	**0.526**

and both gaits, indicating superior policy smoothness and velocity tracking. However, $\lambda = 5 \times 10^{-5}$ is not necessarily optimal from a policy robustness perspective.

6) Policy Robustness against Dynamics Perturbation: We next analyze policy robustness against dynamics perturbation via the ρ-robustness metric. A total of $7 \times 7 = 49$ environments are created by rescaling body friction within the set $\{0.4, 0.6, 0.8, 1.0, 1.5, 2.0, 2.5\}$ and adding mass in $\{-4.5, -3.0, -1.5, 0.0, 1.5, 3.0, 4.5\}$kg to the torso. For each environment, we collect 25 episodes for both bounding and trotting gait under the command velocities $[v_x^{\text{cmd}}, v_y^{\text{cmd}}, \omega_{\text{yaw}}^{\text{cmd}}] = [0.5, 0, 0]$. Table III shows that while PPO-PGDLC generally outperforms PPO (7 out of 8 radii), the best-performing λ differs by gait: $\lambda = 10^{-3}$ for bounding versus $\lambda = 5 \times 10^{-5}$ for trotting. This demonstrates that the optimal regularization strength is task-specific.

7) Zero-Shot Transfer to Hardware: Finally, we compare PPO-PGDLC ($\lambda = 5 \times 10^{-5}$) against PPO in real-world experiments without fine-tuning. The Unitree Go2 robot, carrying additional payloads of 2 kg and 4 kg, is commanded to trot at different forward velocities. We calculate the value of AS, SFR, and VTE for command velocities $v_x^{\text{cmd}} \in [0.6, 1.0]$ m/s, $v_y^{\text{cmd}} = 0$ m/s and $\omega_{\text{yaw}}^{\text{cmd}} = 0$ rad/s, where the real forward velocity of robot is estimated by a Kalman filter state estimator with low pass filter. Results in Table IV show that PPO-PGDLC achieves consistently smoother actions (lower AS in all tests and lower SFR in five of six cases) and comparable or improved velocity tracking (lower VTE in three of six cases), demonstrating better transfer stability under varying payloads. Fig. 4 shows the deployment of PPO-PGDLC policy on hardware.

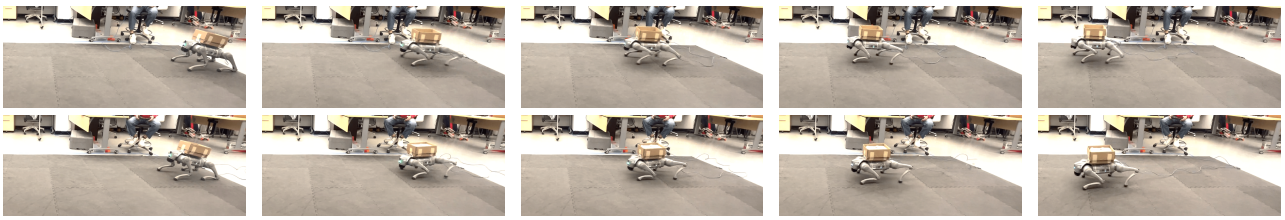


Fig. 4: PPO-PGDLC policy on Go2 hardware, with a 2kg (top) and 4kg (bottom) payload and the command $v_x^{\text{cmd}} = 1$m/s.

VI. CONCLUSIONS

In this paper, we introduce PPO-PGDLC, which leverages a Lipschitz regularized critic to improve policy smoothness and robustness under scenarios where the transition dynamics may be perturbed. The empirical results demonstrate the effectiveness of PPO-PGDLC for a range of simulated and real-world tasks. In future work we will explore more general solutions for handling various transition dynamics and perturbations in the real world.

REFERENCES

[1] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[2] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[3] G. N. Iyengar, "Robust dynamic programming," *Mathematics of Operations Research*, vol. 30, no. 2, pp. 257–280, 2005.

[4] M. A. Abdullah, H. Ren, H. B. Ammar, V. Milenkovic, R. Luo, M. Zhang, and J. Wang, "Wasserstein robust reinforcement learning," *arXiv preprint arXiv:1907.13196*, 2019.

[5] Y. Wang and S. Zou, "Online robust reinforcement learning with model uncertainty," *Advances in Neural Information Processing Systems*, vol. 34, pp. 7193–7206, 2021.

[6] Y. Kuang, M. Lu, J. Wang, Q. Zhou, B. Li, and H. Li, "Learning robust policy against disturbance in transition dynamics via state-conservative policy optimization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 7247–7254.

[7] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[8] X. Song, J. Duan, W. Wang, S. E. Li, C. Chen, B. Cheng, B. Zhang, J. Wei, and X. S. Wang, "Lipsnet: a smooth and robust neural network with adaptive lipschitz constant for high accuracy optimal control," in *International Conference on Machine Learning*. PMLR, 2023, pp. 32 253–32 272.

[9] Y. Zhang, B. Nie, and Y. Gao, "Robust locomotion policy with adaptive lipschitz constraint for legged robots," *IEEE Robotics and Automation Letters*, 2024.

[10] T. Kobayashi, "L2c2: Locally lipschitz continuous constraint towards stable and smooth reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4032–4039.

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

[12] 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," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 024–21 037, 2020.

[13] H. Zhang, H. Chen, D. S. Boning, and C.-J. Hsieh, "Robust reinforcement learning on state observations with learned optimal adversary," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=sCZbhBvqQaU

[14] 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.

[15] K. L. Tan, Y. Esfandiari, X. Y. Lee, S. Sarkar *et al.*, "Robustifying reinforcement learning agents via action space adversarial training," in *2020 American control conference (ACC)*. IEEE, 2020, pp. 3959–3964.

[16] J. Wang, Y. Liu, and B. Li, "Reinforcement learning with perturbed rewards," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 6202–6209.

[17] Y. Wang and S. Zou, "Policy gradient method for robust reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 23 484–23 526.

[18] Y. Zhang, J. Wang, and J. Boedecker, "Robust reinforcement learning in continuous control tasks with uncertainty set regularization," in *Conference on Robot Learning*. PMLR, 2023, pp. 1400–1424.

[19] 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.

[20] 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.

[21] S. H. Lim, H. Xu, and S. Mannor, "Reinforcement learning in robust markov decision processes," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[22] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell, "Adversarial policies: Attacking deep reinforcement learning," in *International Conference on Learning Representations*, 2019.

[23] A. Virmaux and K. Scaman, "Lipschitz regularity of deep neural networks: analysis and efficient estimation," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[24] V. Krishnan, A. Makdah, A. AlRahman, and F. Pasqualetti, "Lipschitz bounds and provably robust training by laplacian smoothing," *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 924–10 935, 2020.

[25] M. Pirotta, M. Restelli, and L. Bascetta, "Policy gradient in lipschitz markov decision processes," *Machine Learning*, vol. 100, pp. 255–283, 2015.

[26] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. R. Salakhutdinov, and K. Chaudhuri, "A closer look at accuracy vs. robustness," *Advances in neural information processing systems*, vol. 33, pp. 8588–8601, 2020.

[27] M. Jordan and A. G. Dimakis, "Exactly computing the local lipschitz constant of relu networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7344–7353, 2020.

[28] H.-T. D. Liu, F. Williams, A. Jacobson, S. Fidler, and O. Litany, "Learning smooth neural functions via lipschitz regularization," in *ACM SIGGRAPH 2022 Conference Proceedings*, 2022, pp. 1–13.

[29] Z. Shi, Y. Wang, H. Zhang, J. Z. Kolter, and C.-J. Hsieh, "Efficiently computing local lipschitz constants of neural networks via bound propagation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 2350–2364, 2022.

[30] W. Wiesemann, D. Kuhn, and B. Rustem, "Robust markov decision processes," *Mathematics of Operations Research*, vol. 38, no. 1, pp. 153–183, 2013.

[31] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[32] E. Rachelson and M. G. Lagoudakis, "On the locality of action domination in sequential decision making." in *ISAIM*. Citeseer, 2010.

[33] A. M. Castro, A. Qu, N. Kuppuswamy, A. Alspach, and M. Sherman, "A transition-aware method for the simulation of compliant contact with regularized friction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1859–1866, 2020.

[34] M. Khajenejad, F. Shoaib, and S. Z. Yong, "Interval observer synthesis for locally lipschitz nonlinear dynamical systems via mixed-monotone decompositions," in *2022 American Control Conference (ACC)*. IEEE, 2022, pp. 2970–2975.

[35] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[36] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[37] Unitree, "Robotics go2," https://www.unitree.com/go2, 2025.

[38] https://github.com/Teddy-Liao/walk-these-ways-go2.

[39] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," *Conference on Robot Learning*, 2022.

[40] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh, "Automatic perturbation analysis for scalable certified robustness and beyond," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1129–1141, 2020.