2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Safe Deep Policy Adaptation

Wenli Xiao*, Tairan He*, John Dolan, and Guanya Shi

***Abstract*— A critical goal of autonomy and artificial intelligence is enabling autonomous robots to rapidly adapt in dynamic and uncertain environments. Classic adaptive control and safe control provide stability and safety guarantees but are limited to specific system classes. In contrast, policy adaptation based on reinforcement learning (RL) offers versatility and generalizability but presents safety and robustness challenges. We propose SafeDPA, a novel RL and control framework that simultaneously tackles the problems of policy adaptation and safe reinforcement learning. SafeDPA jointly learns adaptive policy and dynamics models in simulation, predicts environment configurations, and fine-tunes dynamics models with few-shot real-world data. A safety filter based on the Control Barrier Function (CBF) on top of the RL policy is introduced to ensure safety during real-world deployment. We provide theoretical safety guarantees of SafeDPA and show the robustness of SafeDPA against learning errors and extra perturbations. Comprehensive experiments on (1) classic control problems (Inverted Pendulum), (2) simulation benchmarks (Safety Gym), and (3) a real-world agile robotics platform (RC Car) demonstrate great superiority of SafeDPA in both safety and task performance, over state-of-the-art baselines. Particularly, SafeDPA demonstrates notable generalizability, achieving a 300% increase in safety rate compared to the baselines, under unseen disturbances in real-world experiments.**

I. INTRODUCTION

Adaptation is a crucial aspect of autonomous systems, particularly when confronted with dynamic and uncertain environments [1–7]. In these systems, robots must continually adjust their behavior and decision-making strategies to effectively respond to changing conditions. Adaptation in the context of control systems, often referred to as *adaptive control* [1], is a fundamental concept in control theory. Classic adaptive control approaches the adaptation problem by continually adjusting the control system parameters to maintain or improve its performance in the presence of changing dynamics, uncertainties, or disturbances [8–10]. However, the applicability of classic adaptive control methods is often constrained by their dependence on analytical representations of system dynamics and specific system classes (e.g., systems with linear parametric uncertainties). These limitations restrict their utility in more complex systems. In the field of *reinforcement learning* (RL) [11], policy adaptation operates in a more general setting, often without explicit system models, but it shares the core concept of adaptation [12]. Successful policy adaptation [5, 7, 13–16] empowers autonomous systems to excel in a wide range of tasks and environments, making them more versatile and capable of handling real-world challenges effectively.

Though RL policy adaptation has achieved remarkable progress in autonomous systems, safety remains a critical hurdle that limits its application in real-world control tasks, as any system must ensure that its actions do not lead to harmful consequences for itself or its surroundings. Existing end-to-end safe RL methods [17–19] are hampered in real-world deployment due to the lack of safety guarantees. Hierarchical safe RL methods that provide theoretical safety guarantees [20–22] are susceptible to breaking when confronted with modest changes in dynamics or disturbances. To the best of our knowledge, there is no prior work that jointly tackles policy adaptation and safe reinforcement learning with safety guarantees.

In this paper, we propose SafeDPA, a general RL and control framework that enables rapid policy adaptation in dynamic environments with safety guarantees. SafeDPA (as illustrated in Figure 1) first learns a policy and a control-affine dynamics model, both conditioned on environment configurations, in simulation. Subsequently, SafeDPA learns an adaption module that predicts the environment configuration using historical data in simulation. Finally, in real-world deployments, SafeDPA first fine-tunes the pre-trained dynamics and adaption module, using scant few-shot data. SafeDPA then deploys the adaptive policy with a safety filter, based on the Control Barrier Function (CBF), ensuring safety is upheld. Our contributions are:

- To the best of our knowledge, SafeDPA is the first framework that jointly tackles policy adaptation and safe reinforcement learning.
- Under mild assumptions, we provide theoretical safety guarantees on SafeDPA. Further, we show the robustness of SafeDPA against learning errors and extra perturbations, which also motivates and guides the fine-tuning phase of SafeDPA.
- We conduct comprehensive experiments to evaluate SafeDPA and state-of-the-art baseline methods in (1) a classic control problem *(Inverted Pendulum)*, (2) safe RL simulation benchmarks *(Safety Gym)*, and (3) a real-world agile robotics platform *(RC Car)*. Particularly, SafeDPA demonstrates notable generalizability, achieving a 300% increase in safety rate compared to the baselines, under unseen disturbances in real-world experiments, as illustrated in Figure 2 and Table I.

* These authors contributed equally to this work.
The authors are with the Robotics Institute, Carnegie Mellon University, USA. {wxiao2, tairanh, jdolan, guanyas}@andrew.cmu.edu.
Paper website: https://sites.google.com/view/safe-deep-policy-adaptation

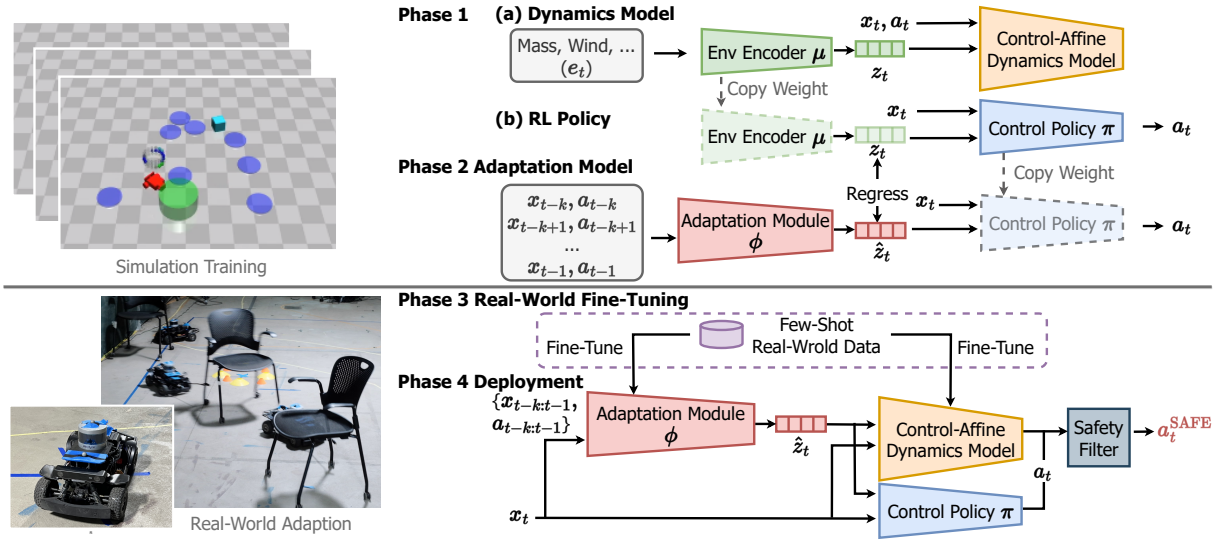

Fig. 1. Overview of the four phases of SafeDPA. In **Phase 1 (a)**, the environment encoder μ_{θ_μ} and dynamics model $f_{\theta_f}, g_{\theta_g}$ are jointly trained with offline dataset collected by a random policy in simulation. In **Phase 1 (b)**, we make the parameters of environments encoder μ_{θ_μ} frozen, and the base policy is trained in simulation using model-free RL. In **Phase 2**, we train the adaption module $\phi_\theta(x_{t-k:t-1}, a_{t-k:t-1})$ to fit environments encoder μ_{θ_μ} with the history of state and actions with on-policy data. In **Phase 3**, we fine-tune our learned dynamics model trained in simulation with few-shot real-world data. In **Phase 4**, we leverage the learned adaptive dynamics to construct a CBF-based safety filter on top of the adaptive RL policy to ensure safety during real-world deployment.

II. Related Work

A. Safe Reinforcement Learning

Safe RL methods can be divided into two categories [23]: 1) *end-to-end methods* and 2) *hierarchical methods*. One representative of *end-to-end methods* is Lagrangian methods that solve a primal-dual optimization problem to satisfy safety constraints [17, 18]. Another branch of *end-to-end methods* is direct policy optimization. Constrained policy optimization (CPO) [19] extends a standard trust-region RL algorithm to CMDP, leveraging a novel bound that relates the expected cost of two policies to their average divergence. *Hierarchical methods* solve the constrained learning problem by adding a safety filter upon the RL policy network, projecting unsafe actions to safe actions [20, 24–27]. Safety Layer [24] solves a QP problem of cost functions with learned dynamics to safeguard unsafe actions, but it does not provide adaptation ability and safety guarantees. ISSA [20] proposes an efficient sampling algorithm for general control barrier functions to get safe actions and guarantees zero violation during agent learning, but it requires a perfect dynamics model for a one-step prediction. Our method can be categorized in the *hierarchical methods* but with fewer assumptions on safety certificate functions and dynamics.

B. Safe Control

The safe control community has built a rich body of literature studying how to persistently satisfy a hard safety constraint. The most widely used methods are *energy-based methods* including potential field methods [28], control barrier functions [29], safe set algorithms [30], and sliding mode algorithms [31]. The method most related to our work is control barrier functions. Control barrier functions give a sufficient approach to guarantee forward invariance of the safe set under the respective control law, and can be naturally used as a safety filter upon RL policy networks.

C. Adaptive Control and Policy Adaptation

Classic adaptive control algorithms often investigate the problem of stability guarantees (e.g., Lyapunov stability) and tracking error convergence (e.g., asymptotic convergence) [1, 32]. Adaptive control via online system identification is critical to deploying robotics in the real world [2]. However, inferring all the exact physics parameters can be unnecessary and difficult [7]. Recently, there has been increased interest in studying online policy adaptation of reinforcement learning for robotics [7, 13–16]. These methods train RL controllers in simulation with environmental parameters (e.g., terrains and friction parameters) and then deploy the policy on real-world robots using adaptation techniques. There are also many Meta-RL works [33–35] focusing on improving the task-wise adaptation ability. Among these policy adaptation works, the most relevant work, Rapid Motor adaptation (RMA) [7], proposes a teacher-student learning framework consisting of two components: a base policy and an adaptation module. The combination of these components and the student-teacher training scheme enables the legged robots to adapt to unseen terrains. Compared to RMA, our approach has safety guarantees and better bridges the sim-to-real gap (as shown in Section VI) by fine-tuning with few-shot real-world data. To the best of our knowledge, our work is the first to investigate the policy adaptation of safe RL in dynamics.

III. Problem Statement

In this work, we consider a system with control-affine dynamics, which could generally represent any rigid-body robotic system [36], as:

$$x_{t+1} = f(x_t, e_t) + g(x_t, e_t)a_t, \tag{1}$$

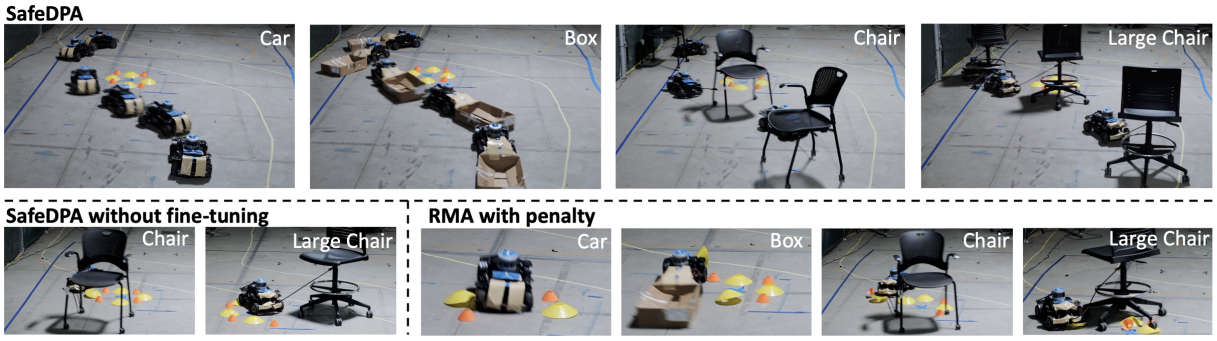

Fig. 2. Comparison of SafeDPA, SafeDPA without fine-tuning, and RMA with penalty on RC Car platforms. We showcase the successful trajectory of SafeDPA in four tasks, alongside instances of an unsafe event (i.e., collision) for SafeDPA without fine-tuning and RMA with penalty. SafeDPA safely achieves the goal, although in training or fine-tuning it never sees the box or chairs. This highlights the exceptional generalizability and adaptability of SafeDPA. We present video demonstrations in https://sites.google.com/view/safe-deep-policy-adaptation.

with time step t, state $x_t \in \mathbb{R}^n$, action $a_t \in \mathbb{R}^m$, environment configuration $e_t \in \mathbb{R}^k$, unactuated dynamics $f : \mathbb{R}^m \times \mathbb{R}^k \to \mathbb{R}^m$, and actuated dynamics $g : \mathbb{R}^m \times \mathbb{R}^k \to \mathbb{R}^{n\times m}$. The environment configuration e_t (unknown in real-world deployment) represents physical properties that may change in the system, e.g., friction, payload, disturbance, etc. Different types of configuration e_t may exert distinct influences on the system dynamics. For instance, in the case of RC Car (as shown in Figure 2), dragging a box or a heavy chair can affect both f and g. However, certain disturbances only impact f, such as an additional wind force. These environment configurations e_t play a crucial role in adaptive control. Changes in wind or dragging can cause drones to crash [37] or force cars to collide with obstacles [21], resulting in system breakdown, especially for safety-critical systems. On the other hand, e_t also has huge impacts on control policy performance. Therefore, our objective is to design a framework that accounts for varying environment configurations, ensuring tasks are executed with a high probability of safety guarantee while achieving high task rewards. We formulate this problem as: $\max_\pi \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(x_t)\right]$ s.t. $x_t \in \mathcal{C}$. The safe set $\mathcal{C}$ defines a region in the state space that the robot should always stay in. We apply the standard RL [7, 38, 39] setting for solving that, say, π is the control policy, $r \in \mathbb{R}$ denotes reward, and γ is the discounted factor.

For the above problem, it is nontrivial to model the dynamics of the robot system conditioned on environment configurations e_t, because though it is easy to access the environment configuration e_t in simulation, it is difficult to obtain it during real-world deployments. Moreover, the sim-to-real gap degrading the task performance might be acceptable in certain applications, but this gap might lead to unacceptable catastrophes in safety-critical scenarios. This makes fine-tuning the adaptive dynamics model using few-shot real-world data necessary. In the following section, we will introduce how we tackle these challenges.

IV. Safe Deep Policy Adaptation

In this section, we elaborate on our proposed SafeDPA, as outlined in Figure 1. SafeDPA comprises four phases: Training dynamics model and policy in simulation, training adaptation module in simulation, fine-tuning with few-shot real-world data, and deployment with safe filter, each of which is elaborated upon in the subsequent subsections.

A. Phase 1: Dynamics Model Learning in Simulation

Our first step is to train a neural control-affine dynamics model f_{θ_f} and g_{θ_g} to approximate system dynamics f and g in simulation. We apply an environment encoder μ_θ to extract latent representation z_t of environment configuration e_t, then pass z_t to dynamics model f_{θ_f} and g_{θ_g}. We implement μ_{θ_μ}, f_{θ_f}, and g_{θ_g} as multi-layer perceptrons (MLP) and train these models jointly through end-to-end supervised learning. The training loss is defined by:

$$\begin{aligned}\mathcal{L}^{\text{DYN}}_{\theta_f,\theta_g,\theta_\mu} &= \frac{1}{|\mathcal{D}|}\sum_t \|x_{t+1} - \hat{x}_{t+1}\|_2^2 \\ &= \frac{1}{|\mathcal{D}|}\sum_t \|x_{t+1} - f_{\theta_f}(x_t, \mu_\theta(e_t)) - g_{\theta_g}(x_t, \mu_{\theta_\mu}(e_t))a_t\|_2^2,\end{aligned} \tag{2}$$

where $\mathcal{D}$ is a dataset of transitions (x_t, e_t, a_t, x_{t+1}). We synthesize dataset $\mathcal{D}$ by rolling out a random walking agent in the simulation. We randomize the environment configuration e_t at every step to collect diverse data.

With the learned encoder μ_{θ_μ}, we can train an adaptive RL[1] policy $\pi_{\theta_\pi}(x_t, \mu_\theta(e_t))$ by integrating model-free RL algorithms (e.g. PPO [38]) with the parameter-frozen environment encoder μ_{θ_μ}. RL will learn the policy that maximizes expected return $\mathbb{E}_{\tau\sim\pi}\left[\Sigma_{t=0}^{\infty}\gamma^t r(x_t)\right]$ under different environment latent z_t. In our early empirical experiments, we find that the training scheme in RMA [7] (i.e., jointly training the environment encoder μ_{θ_μ} with the RL agent) does not apply for dynamics learning since the latent variable z_t learned from RL loss does not provide sufficient context for accurate dynamics prediction. Therefore, our approach largely improves the dynamics prediction by pertaining the environment encoder only with $\mathcal{L}^{\text{DYN}}_{\theta_f,\theta_g,\theta_\mu}$ to extract a more meaningful representation for dynamics learning.

B. Phase 2: Train Adaptation Module in Simulation

As mentioned in Section III, the environment configurations e_t are often inaccessible in the real world. Inspired

[1]While we utilize an RL agent as the control policy, our proposed SafeDPA is controller-agnostic. The RL controller can be substituted with other methods, such as Model Predictive Control (MPC).

by RMA [7] and UP-OSI [40], we learn an adaptation module $\phi_\theta(x_{t-k:t-1}, a_{t-k:t-1})$ that continuously predicts the environment latent $\hat{z}_t$ for dynamics model and control policy, where $\{x_{t-k:t-1}, a_{t-k:t-1}\}$ is a history of state-action pairs with length k. We predict latent z_t rather than environment configurations e_t, since it is more generalizable and compact to consider the impact of e_t on dynamics than its actual value [7]. We model the adaptation module using a 1-D CNN to capture spatiotemporal correlation and train it via an iterative procedure in simulation. For each iteration, we roll out an RL agent in simulation with random episodic e_t to collect dataset $\mathcal{D}^{\text{ADAPT}}$ with tuples $((x_{t-k:t-1}, a_{t-k:t-1}), e_{t-1})$. We then train ϕ_{θ_ϕ} via supervised learning with $\mathcal{D}^{\text{ADAPT}}$ to minimize loss $\mathcal{L}^{\text{ADAPT}}_{\theta_\phi} = \frac{1}{|\mathcal{D}^{\text{ADAPT}}|}\sum_t ||z_t - \hat{z}_t||_2^2$, where $z_t = \mu_{\theta_\mu}(e_t)$ and $\hat{z}_t = \phi_{\theta_\phi}(x_{t-k:t-1}, a_{t-k:t-1})$.

C. Phase 3: Real-World Data Fine-Tuning

Since the safety performance of SafeDPA heavily relies on an accurate adaption module and dynamics model (as discussed in Section V), the safety can be largely affected by the sim-to-real gap. Leveraging the adaptive structure of the dynamics model in SafeDPA, we are able to align simulation with the real world by fine-tuning the dynamics model and adaptation module with few-shot (compared with the training set) real-world data. We propose to fine-tune models via end-to-end supervised learning, which minimize $\mathcal{L}^{\text{TUNE}}_{\theta_f,\theta_g,\theta_\phi} = \frac{1}{|D_{\text{real}}|}\sum_t ||\hat{x}_{t+1} - x_{t+1}||_2^2$, where $\hat{x}_{t+1} = f_{\theta_f}(x_t, \hat{z}_t) + g_{\theta_g}(x_t, \hat{z}_t)a_t$, $\hat{z}_t = \phi_{\theta_\phi}(x_{t-1:t-k}, a_{t-1:t-k})$. Since the dynamics model is independent of control policy, the dataset D_{real} can be collected from various sources, such as teleoperation, random walks, etc. The fine-tuned neural networks $f_{\theta_f}, g_{\theta_g}, \phi_{\theta_\phi}$ have shown empirical success and necessity in real-world applications, as evidenced in Section VI-C. Remarkably, SafeDPA displays strong generalizability to adapt to new disturbances like **Chairs** and **Box** (refer to Figure 2), even though it has never seen these objects in both simulation and fine-tuning.

D. Phase 4: Safe Filter and Real-World Deployment

To achieve safe real-world deployment, we combine the adaptation module and dynamics model and leverage the control barrier function (CBF) [29] to construct a safe filter, which can guarantee the robot always stays in the safe set $\mathcal{C}$ (i.e., forward invariance). We first construct a CBF $h : \mathbb{R}^n \to \mathbb{R}$, which satisfies:

$$\mathcal{C}^h := \{x \in \mathbb{R}^n : h(x) \geq 0\} \subseteq \mathcal{C}, \forall x \in \partial\mathcal{C}^h, \frac{\partial h}{\partial x} \neq 0, \quad (3)$$

$$\forall x \in \partial\mathcal{C}^h, h(x_{k+1}) \geq (1-\eta)h(x_k), 0 < \eta \leq 1, \quad (4)$$

where $\mathcal{C}^h$ is a subset of the safe set $\mathcal{C}$, and η is a scalar representing the conservativeness of CBF. If Equation (4) is satisfied for all $x_t \in \mathcal{C}^h$, then $\mathcal{C}^h$ and the safe set $\mathcal{C}$ are rendered forward invariant [41, 42]. We adopt affine barrier functions as [21]: $h = p^T x + q$, $(p \in \mathbb{R}^n, q \in \mathbb{R})$, for the simplicity of computation. We then formulate an optimization problem (CBF-QP) satisfying Equation (4) that can be solved at each time step:

$$\begin{aligned} a_t^{\text{SAFE}} = \underset{a_t}{\text{argmin}} \quad & \|a_t - \pi_{\theta_\pi}(x_t, \hat{z}_t)\|_2 \\ \text{s.t.} \quad & p^T f_{\theta_f}(x_t, \hat{z}_t) + p^T g_{\theta_g}(x_t, \hat{z}_t)\, a_t + p^T q \\ & \geq (1-\eta)h(x_t) - \epsilon \\ & a_t \in \mathcal{A}, \end{aligned}$$

where $\mathcal{A}$ is feasible action set, $\hat{z}_t = \phi_{\theta_\phi}(x_{t-k:t-1}, a_{t-k:t-1})$, and ϵ is a variable of robust margin for safety which will be quantified in the following theory part (Section V). This QP seeks to modify RL policy to meet safety constraints with minimal interference. SafeDPA takes the output of safety filter a_t^{SAFE} to execute in the real world. The theoretical analysis on the safety of a_t^{SAFE} is presented in Section V.

V. Theoretical Analysis

In this section, we begin to present the safety guarantee of SafeDPA by introducing an assumption on the dynamics prediction error.

Assumption 1 (Bounded Prediction Error): $\forall x_t, a_t, e_t$, $\|f(x_t, e_t) - f_{\theta_f}(x_t, e_t)\|_1 < \epsilon_f$, $\|g(x_t, e_t) - g_{\theta_g}(x_t, e_t)\|_1 < \epsilon_g$, $\|\hat{z}_t - z_t\|_1 = ||\phi_{\theta_\phi}(x_{t-k:t-1}, a_{t-k:t-1}) - \mu(e_t)|| < \epsilon_z$. We further introduce another assumption regarding the Lipschitz continuity of the (neural) dynamics. These assumptions are necessary because they provide critical quantifications into the properties of dynamics and neural networks. Otherwise, we would lack the tools to rigorously analyze the reliability of these networks [43].

Assumption 2 (Lipschitz Continuity): The dynamics $f(\cdot)$, $g(\cdot)$ in Equation (1) are L_f, L_g Lipschitz continuous with respect to the 1-norm. The learned dynamics $f_{\theta_f}(\cdot)$, $g_{\theta_g}(\cdot)$ in Equation (2) are $L_{f_{\theta_f}}$, $L_{g_{\theta_g}}$ Lipschitz continuous with respect to the 1-norm.

Theorem 1 (Safe Control): Under Assumption 1 and 2, then solving the safety condition $p^T f(x_t) + p^T g(x_t)\, a_t + p^T q \geq (1-\eta)h(x_t) - \epsilon$ in Section IV-D will guarantee the forward invariance of the safe set $\mathcal{C}$ (i.e., $x_{t+n} \in \mathcal{C}, \forall n = 1, 2, 3\cdots$) if $\epsilon > \epsilon_f p^T\mathbf{1} + \epsilon_g \|a\|_1^{max} p^T\mathbf{1} + \epsilon_z(L_f + L_{f_{\theta_f}}) + \epsilon_z\|a\|_1^{max}(L_g + L_{g_{\theta_g}})$ where we denote $\|a\|_1^{max} = \max_{\forall a_t \in \mathcal{A}} \|a_t\|_1$.

Proof: We consider the worst-case prediction of the $h(\hat{x}_{t+1})$, which is $\min_{x_t, a_t} h(\hat{x}_{t+1})$ where $\hat{x}_{t+1} = f_{\theta_f}(x_t, \hat{z}_t) + g_{\theta_g}(x_t, \hat{z}_t)a_t$. The prediction error of $h(\hat{x}_{t+1})$ comes from the prediction error of $\hat{x}_{t+1}$, which can be further decoupled into two parts: 1) the approximation error of f_{θ_f} and g_{θ_g}; 2) the approximation error of ϕ_{θ_ϕ}. As for 1) the approximation error of f_{θ_f} and g_{θ_g}, supposing we have perfect regression of z_t (i.e., $\hat{z}_t = z_t$), denote the maximum error of $h(\hat{x}_{t+1})$ under this situation as $\Delta_1 = h(x_{t+1}) - h(f_{\theta_f}(x_t, \hat{z}_t) + g_{\theta_g}(x_t, \hat{z}_t)a_t) < \epsilon_f p^T\mathbf{1} + \epsilon_g\|a\|_1^{max} p^T\mathbf{1}$. As for 2) the approximation error of ϕ_{θ_ϕ}, denote the maximum error of $h(\hat{x}_{t+1})$ under this situation as $\Delta_2 = \|\hat{z}_t - z_t\|_1^{max}(L_f + L_{f_{\theta_f}}) + \|\hat{z}_t - z_t\|_1^{max}\|a\|_1^{max}(L_g + L_{g_{\theta_g}}) < \epsilon_z(L_f + L_{f_{\theta_f}}) + \epsilon_z\|a\|_1^{max}(L_g + L_{g_{\theta_g}})$. Combining Δ_1 and Δ_2, by setting $\epsilon > \Delta_1 + \Delta_2$ and solving $h(\hat{x}_{t+1}) \geq (1-\eta)h(x_t) - \epsilon$, we have $h(x_{t+1}) \geq (1-\eta)h(x_t)$. The forward

invariance of the safe set $\mathcal{C}$ is a natural proof following [44]. ■

Note that the value of ϵ depends on two types of error: 1) dynamics learning error (i.e., ϵ_f and ϵ_g); and 2) adaptation regression error (i.e., ϵ_z). As these two errors decrease (e.g., with better prediction models or fine-tuning with real-world data), ϵ will decrease as well. In practice, one can statistically estimate ϵ using fine-tuning data, but quantifying exact error bounds is a challenging machine learning problem [37, 43].

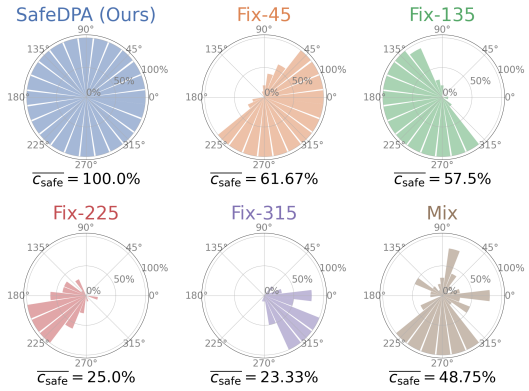

Fig. 3. The area of the colored region represents the safety rate. In the inverted pendulum task, SafeDPA consistently achieves the highest safety rate with zero violations across all directions, surpassing Fix-α and Mix where these baselines only maintain high safety rates in specific directions.

VI. EXPERIMENTS

In this section, we demonstrate extensive experimental results in Inverted Pendulum, Safety Gym [45], and real-world RC Car, to address the following questions:

- **Q1**: Can SafeDPA guarantee safety in diverse tasks and various environment configurations?
- **Q2**: Can SafeDPA strike the perfect balance between safety and task performance? How does SafeDPA compare with state-of-the-art safe RL methods?
- **Q3**: How well does SafeDPA generalize to unseen scenarios in the real-world? How does the fine-tuning phase in SafeDPA boost performance?

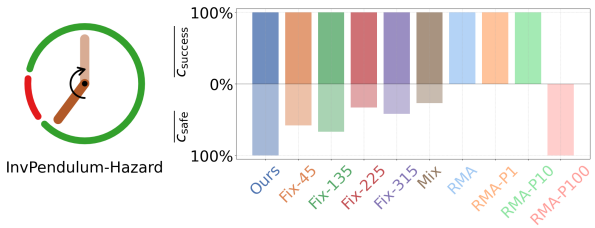

Fig. 4. On the left, we show the InvPendulum-hazard environment. On the right, we demonstrate the success rate and safety rate of SafeDPA and baselines where SafeDPA is the only algorithm that achieves both 100% for success rate and safety rate.

A. Inverted Pendulum

We adopt an inverted pendulum simulation augmented with external wind perturbations from [46]. Wind (with fixed magnitude, and variable angle from 0 to 360 degrees) is introduced as the variable environment configuration e_t. In this setting, we aim to evaluate the adaptive safety of SafeDPA, and its dual capability in safety and task performance.

*1) Address **Q1**:* First, we consider the *pendulum balancing* task, where the pendulum aims to sustain upright (angle $\theta = 0$). Training the RL agent with the reward function $r = -0.1\theta^2 - 0.1\dot{\theta}^2 - 0.0001u^2$, where u is the torque, inherently aligns it to remain upright, fulfilling the safety criteria $|\theta| < 45°$. We compare SafeDPA with two baselines: Fix-α and Mix. For Fix-α, dynamics models are trained in an environment with wind consistently directed as per α, which can be one of $\{45, 135, 225, 315\}$. Conversely, the Mix model is trained in various wind conditions without any adaptation (which can be viewed as a domain randomization method). To conduct a fair comparison of safety, we employ a control policy generating random actions for our adaptive safe module and other baseline methods. We let SafeDPA, Fix-α, and Mix share the same safety filter structure, and test them across environments with wind directions spanning from 0° to 360°. The result is demonstrated in Figure 3, in which the size of the colored area denotes the safety rate in different directions. Note that Fix-α and Mix can only guarantee safety under specific wind directions, while SafeDPA achieves zero violation in all wind directions, outperforming baselines significantly.

*2) Address **Q2**:* To evaluate both task performance and safety simultaneously, we introduce the *InvPendulum-Hazard* task, as shown in Figure 4. In this setup, the RL objective is to maintain the pendulum in an upright position. Distinct from this goal, there is a safety constraint to avoid a hazard (represented by the red region shown in the pendulum diagram, as shown in Figure 4). Notably, the pendulum's starting position is just beneath this hazard. To maintain an upright position without entering the hazard, the control policy should direct the pendulum to rotate counter-clockwise, through a longer route instead of the shorter clockwise path. In addition to Fix-α and Mix, our comparison includes RMA [7] and RMA-Pβ. The RMA-Pβ is an invariant of RMA that is trained with an extra reward penalty quantified by β, for any safety violation during training.

As shown in Figure 4, RMA consistently lacks safety guarantees, and the RMA-Pβ methods are either too aggressive or too conservative. Meanwhile, both Fix-α and Mix exhibit compromised safety across diverse e_t environments. In contrast, SafeDPA consistently achieves 100% success rate with 100% safety rate, demonstrating its capability to ensure safety while achieving high performance across varying environment configurations.

TABLE I

THE SUCCESS RATE AND SAFETY RATE OF SAFEDPA, SAFEDPA WITHOUT FINE-TUNING AND RMA WITH PENALTY ON RC CAR.

Task	Success Rate			Safety Rate		
	Box	Chair	Car	Box	Chair	Car
SafeDPA	1.00	0.80	1.00	**0.90**	**1.00**	**0.75**
SafeDPA w/o Tuning	0.80	1.00	1.00	0.70	0.45	0.34
RMA-P	1.00	1.00	1.00	0.50	0.25	0.25

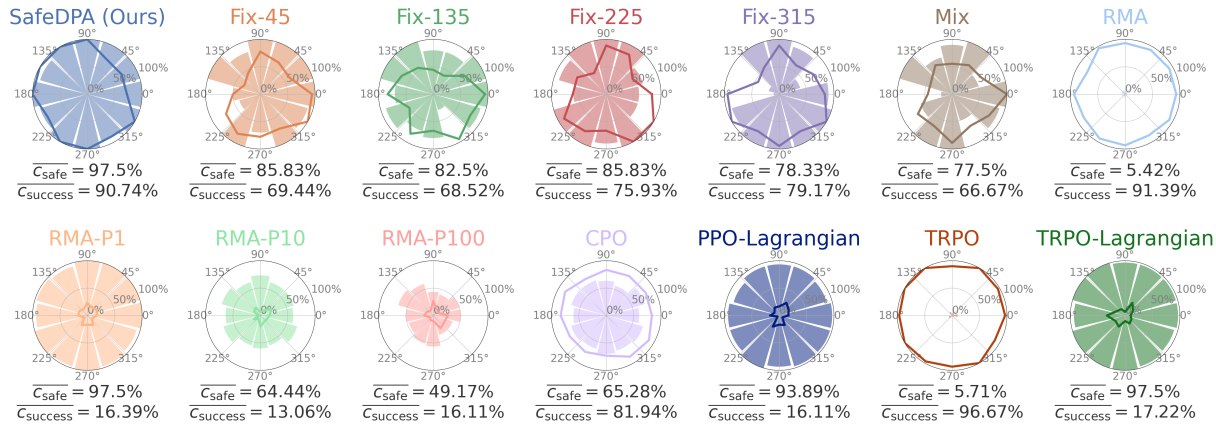

Fig. 5. The radius of the circles indicates task performance (success rate), while the size of the colored area represents the safety rate. SafeDPA stands out with the highest safety rate 97.5%. Though PPO and TRPO have marginally higher success rates, they frequently violate safety constraints.

B. Safety Gym Simulation Experiments

To answer **Q2**, we implement SafeDPA on the Safety Gym [45] benchmark. The *point* robot in Safety Gym has higher state dimensions $x_t \in \mathbb{R}^{17}$, compared to pendulum's $x_t \in \mathbb{R}^2$. We introduce an external force $F_t \in \mathbb{R}^2$ (on the 2D plane) within MuJoCo [47], and define F_t as environment configuration. The control task is to navigate the *point* robot to a target and avoid hazards. For this setting, we also incorporate standard RL methods (PPO [38] and TRPO [39]) and end-to-end safe RL methods including CPO [19], PPO-Lagrangian, and TRPO-Lagrangian [18] as benchmarks. Similar to experiments in Section VI-A, we evaluate the algorithms across various environments with the direction of F_t ranging from 0 to 360 degrees. The results are demonstrated in Figure 5. Remarkably, our proposed SafeDPA is the sole algorithm that maintains both high performance and safety across all F_t values. In contrast, other baseline algorithms tend to excel in either performance or safety, but not in both aspects simultaneously.

C. Real-World Deployment

After successfully addressing **Q1** and **Q2** in simulations, we take a step further into the complexities of real-world experiments. In this section, we delve into evaluations of SafeDPA on an RC Car, aiming to tackle **Q3**.

1) Experimental Setup: We equip a 1/10 scale racing car with a Jetson TX2 to execute an on-board control policy. This setup ensures our controller runs efficiently at a frequency of 20Hz. To achieve precise control, we employ the Vicon motion capture system, which continuously provides information about the car's location. With this real-time data, the control objective for the car is to navigate towards a set goal, avoiding any collisions with a centrally positioned obstacle.

2) Training in Simulation: We start by constructing a simulator with a modified kinematic bicycle model, in which the velocity is modeled by a PID controller. We measure the car's physical properties to align the simulation. Within this simulation, we also simulate a random drag ($F_t \in \mathbb{R}^2$), which serves as an environment configuration e_t. The car's control commands are target velocity and steering. The state of the car consists of position, yaw, and real velocity. We train RMA with a reward penalty for collision (denoted as RMA-P), and SafeDPA without fine-tuning, as baselines. Both SafeDPA and SafeDPA without fine-tuning utilize 2×10^7 state-action pairs in model learning.

3) Real-world Fine-Tuning and Deployment: We use hard-coded commands to make the car move in circles on the unobstructed ground without the drag disturbance, collecting data merely amounting to 0.1% of simulated data. We use this dataset to fine-tune the adaption module and dynamics model of SafeDPA.

4) Deployment and Adaptation: The adaptation module is deployed synchronously, given that computations could be completed within the $0.05s$ window. For real-world adaptation demonstrations, we set the obstacle in the same location as in the simulation, and test four distinct environment configurations: 1) **Car**: Standard RC car without dragging; 2) **Box**: Car attaching a paperboard box; 3) **Chair**: Car attaching a movable chair; 4) **Large Chair**: Car attaching a heavier chair. It is worth noting that configurations **Box, (Large) Chair** are totally unseen before deployment. These configurations test the model's zero-shot adaptation capabilities since these specific configurations were neither modeled in the simulation nor used during the fine-tuning phase.

5) Results: The results are summarized in Table I and Figure 2. Even though SafeDPA has never seen chairs or boxes in simulation and fine-tuning, it still achieves a high safety rate. SafeDPA 3× outperforms RMA-P on average in terms of safety. Even though RMA-P has good performance, it frequently violates safety constraints, which can be catastrophic in safety-critical systems. We could clearly answer **Q3** by the fact that fine-tuning with real-world data boosts the SafeDPA's generalizability and performance.

VII. Conclusion and Future Works

In this paper, we propose SafeDPA, a novel framework that tackles the problem of policy adaptation while ensuring safety. In the future, there are two interesting research directions. One is building closed-loop adaptation methods that directly adapt the policy network based on its performance and safety. The other is to generalize SafeDPA to vision-based control systems and non-control-affine systems.

REFERENCES

[1] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.

[2] J.-J. E. Slotine and W. Li, "On the adaptive control of robot manipulators," *The international journal of robotics research*, vol. 6, no. 3, pp. 49–59, 1987.

[3] E. Lavretsky and K. A. Wise, "Robust adaptive control," in *Robust and adaptive control: With aerospace applications*, Springer, 2012, pp. 317–353.

[4] M. O'Connell *et al.*, "Neural-fly enables rapid learning for agile flight in strong winds," *Science Robotics*, vol. 7, no. 66, eabm6597, 2022.

[5] K. Huang, R. Rana, G. Shi, A. Spitzer, and B. Boots, "Datt: Deep adaptive trajectory tracking for quadrotor control," in *Conference on Robot Learning (CoRL)*, 2023.

[6] D. Hanover, P. Foehn, S. Sun, E. Kaufmann, and D. Scaramuzza, "Performance, precision, and payloads: Adaptive nonlinear mpc for quadrotors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 690–697, 2021.

[7] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *arXiv preprint arXiv:2107.04034*, 2021.

[8] P. Parks, "Liapunov redesign of model reference adaptive control systems," *IEEE Transactions on Automatic Control*, vol. 11, no. 3, pp. 362–367, 1966.

[9] P. K. Khosla and T. Kanade, "Parameter identification of robot dynamics," in *1985 24th IEEE conference on decision and control*, IEEE, 1985, pp. 1754–1760.

[10] P. A. Ioannou and J. Sun, *Robust adaptive control*. PTR Prentice-Hall Upper Saddle River, NJ, 1996, vol. 1.

[11] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[12] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: A survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*, IEEE, 2020, pp. 737–744.

[13] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, eabc5986, 2020.

[14] J. Tan *et al.*, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.

[15] J. Hwangbo *et al.*, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, eaau5872, 2019.

[16] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *arXiv preprint arXiv:2004.00784*, 2020.

[17] Q. Liang, F. Que, and E. Modiano, "Accelerated primal-dual policy optimization for safe reinforcement learning," *arXiv preprint arXiv:1802.06480*, 2018.

[18] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," *arXiv preprint arXiv:1805.11074*, 2018.

[19] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*, PMLR, 2017, pp. 22–31.

[20] W. Zhao, T. He, and C. Liu, "Model-free safe control for zero-violation reinforcement learning," in *5th Annual Conference on Robot Learning*, 2021.

[21] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 3387–3395.

[22] T. He, C. Zhang, W. Xiao, G. He, C. Liu, and G. Shi, "Agile but safe: Learning collision-free high-speed legged locomotion," *arXiv preprint arXiv:2401.17583*, 2024.

[23] W. Zhao, T. He, R. Chen, T. Wei, and C. Liu, "State-wise safe reinforcement learning: A survey," *arXiv preprint arXiv:2302.03122*, 2023.

[24] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," *arXiv preprint arXiv:1801.08757*, 2018.

[25] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[26] W. Xiao, Y. Lyu, and J. Dolan, "Model-based dynamic shielding for safe and efficient multi-agent reinforcement learning," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2023, pp. 1587–1596.

[27] W. Zhao, T. He, and C. Liu, "Probabilistic safeguard for reinforcement learning using safety index guided gaussian process models," in *Learning for Dynamics and Control Conference*, PMLR, 2023, pp. 783–796.

[28] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles*, Springer, 1986, pp. 396–404.

[29] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*, IEEE, 2014, pp. 6271–6278.

[30] C. Liu and M. Tomizuka, "Control in a safe set: Addressing safety in human-robot interactions," in *ASME 2014 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers Digital Collection, 2014.

[31] L. Gracia, F. Garelli, and A. Sala, "Reactive sliding-mode algorithm for collision avoidance in robotic systems," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2391–2399, 2013.

[32] J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*. Prentice hall Englewood Cliffs, NJ, 1991, vol. 199.

[33] M. Luo *et al.*, "Mesa: Offline meta-rl for safe adaptation and fault tolerance," *arXiv preprint arXiv:2112.03575*, 2021.

[34] A. Nagabandi *et al.*, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," *arXiv preprint arXiv:1803.11347*, 2018.

[35] T. He *et al.*, "Reinforcement learning with automated auxiliary loss search," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1820–1834, 2022.

[36] M. Enqvist, "Linear models of nonlinear systems," 2005.

[37] A. Liu, G. Shi, S.-J. Chung, A. Anandkumar, and Y. Yue, "Robust regression for safe exploration in control," in *Learning for Dynamics and Control*, PMLR, 2020, pp. 608–619.

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

[39] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, PMLR, 2015, pp. 1889–1897.

[40] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," *arXiv preprint arXiv:1702.02453*, 2017.

[41] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation.," in *Robotics: Science and Systems*, Cambridge, MA, USA, vol. 13, 2017, pp. 1–10.

[42] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[43] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017.

[44] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*, IEEE, 2019, pp. 3420–3431.

[45] A. Ray, J. Achiam, and D. Amodei, "Benchmarking Safe Exploration in Deep Reinforcement Learning," 2019.

[46] G. Shi, K. Azizzadenesheli, M. O'Connell, S.-J. Chung, and Y. Yue, "Meta-adaptive nonlinear control: Theory and algorithms," *Advances in Neural Information Processing Systems*, vol. 34, pp. 10 013–10 025, 2021.

[47] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2012, pp. 5026–5033.