

Failure-Aware RL: Reliable Offline-to-Online Reinforcement Learning with Self-Recovery for Real-World Manipulation

Huanyu Li^{1,2*}, Kun Lei^{1,2*}, Sheng Zang⁴, Kaizhe Hu^{1,3}, Yongyuan Liang⁶, Bo An⁴, Xiaoli Li⁵, Huazhe Xu^{1,3}

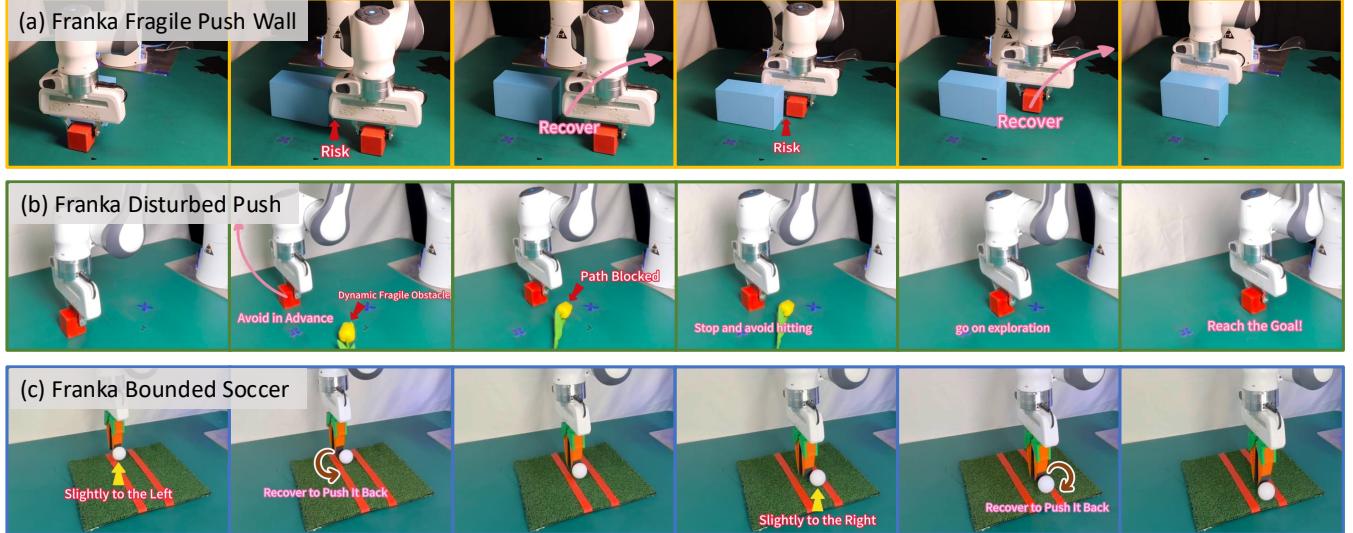


Fig. 1: FTRL demonstrated on a Franka Emika Panda robot across three manipulation tasks: (a) pushing fragile objects while avoiding wall collisions, (b) pushing with dynamic obstacle avoidance, (c) soccer with boundary constraints (detailed in Section VI.C.1). FTRL predicts potential failures and executes recovery actions, significantly reducing Intervention-requiring Failures during real-world RL while improving task performance.

Abstract—Post-training algorithms based on deep reinforcement learning can push the limits of robotic models for specific objectives, such as generalizability, accuracy, and robustness. However, Intervention-requiring Failures (IR Failures) (e.g., a robot spilling water or breaking fragile glass) during real-world exploration happen inevitably, hindering the practical deployment of such a paradigm. To tackle this, we introduce Failure-Aware Offline-to-Online Reinforcement Learning (FTRL), a new paradigm minimizing failures during real-world reinforcement learning. We create FailureBench, a benchmark that incorporates common failure scenarios requiring human intervention, and propose an algorithm that integrates a world-model-based safety critic and a recovery policy trained offline to prevent failures during online exploration. Extensive simulation and real-world experiments demonstrate the effectiveness of FTRL in significantly reducing IR Failures while improving performance and generalization during online reinforcement learning post-training. FTRL reduces IR Failures by 73.1% while elevating performance by 11.3% on average during real-world RL post-training. Videos and code are available at failure-aware-rl.github.io.

I. INTRODUCTION

Learning efficient and accurate policies has been longed for years by roboticists and AI researchers. However, pure imitation learning [1–3] often overfits the training distribution, while reinforcement learning is notoriously data-inefficient.

*Equal contribution. ¹Shanghai Qi Zhi Institute, ²Shanghai Jiao Tong University, ³IIIS, Tsinghua University, ⁴Nanyang Technological University, ⁵A*STAR Institute for Infocomm Research, ⁶University of Maryland.

Consequently, reinforcement learning (RL) based post-training becomes crucial for continuously refining specific objectives and adapting policies to the dynamic conditions of the real world. Specifically, offline-to-online RL [4–6], where the agent is first trained offline on demonstrations and then fine-tuned through online RL, leverages the advantages of both demonstration-based pre-training and RL-based post-training. However, a significant challenge in deploying RL in such settings is the occurrence of Intervention-requiring Failures (IR Failures) during the learning process. These failures stem from the intrinsic necessity for exploration in RL, which introduces randomness into the agent’s actions. While exploration is crucial for learning optimal policies, it can result in actions that lead to irreversible damage or unsafe situations, such as breaking fragile objects, knocking items out of reach, or damaging the robot arm by hitting objects. Thus, IR Failures often necessitate human intervention to resolve, as they cannot be easily addressed through heuristic methods or reset-free RL techniques [7–9].

To this end, we introduce the concept of Failure-Aware Offline-to-Online Reinforcement Learning (FTRL), where the agent refines its policy while minimizing IR Failures that would otherwise require human intervention.

Our method builds on an offline-to-online RL algorithm [6], which employs a policy gradient objective to unify online and offline RL in an on-policy manner.

In this work, we focus on failure-aware post-training

within real-world manipulation settings.

To study various failure scenarios in simulation environments, we introduce a new benchmark, **FailureBench**.

FailureBench builds upon existing simulation environments [10] by incorporating common IR Failure scenarios that frequently occur in real-world manipulation tasks. We simulate situations where failures necessitate human intervention, such as objects being pushed out of the workspace or the robot entering unsafe states. This benchmark enables us to evaluate how effectively the RL algorithms balance performance improvement and generalization while minimizing IR Failures during exploration.

Our research reveals that existing online and offline-to-online RL algorithms [6, 11, 12] frequently encounter IR Failures in this setting due to the inherent exploration-exploitation trade-off in RL. To mitigate this problem, we introduce a safety critic based on a latent world model for failure prediction, along with a recovery policy designed to prevent failures foreseen by the safety critic.

Both components are trained offline using carefully curated demonstrations and later deployed to prevent failures during the online task policy post-training process.

Our framework demonstrates significant reductions in IR Failures across various simulated settings within FailureBench, while simultaneously enhancing task performance and generalizability. We validate the effectiveness of FARL through real-world experiments conducted on a Franka Emika Panda robot, showing that our methods substantially reduce the need for human intervention during training. **Our contributions are summarized as follows:**

- We identify a major barrier to deploying RL in real-world scenarios: IR Failures caused by exploration-induced randomness. To study such potential risks in existing methods, we introduce **FailureBench**, a benchmark that repurposes existing RL environments to study IR Failure cases requiring human intervention, enabling evaluation of RL algorithms for both performance and failure minimization.
- We propose a failure-aware offline-to-online framework, including a specifically designed world model and recovery policy to minimize IR Failures while facilitating learning and adaptation with RL, theoretically justified by an “advantage correction” analysis to simultaneously enhance learning and safety.
- We conduct extensive experiments in both simulated environments and three challenging real robotic tasks susceptible to IR Failures to validate the effectiveness of our approach.

II. RELATED WORK

Previous research in safe reinforcement learning has extensively examined safety through various approaches, focusing mainly on learning-from-scratch scenarios [13–15]. However, with advances in modern robotic models that learn from demonstrations [1–3, 16–18], there is an increasing recognition of the importance of online post-training strategies.

Consequently, this work shifts its focus to a failure-aware offline-to-online reinforcement learning setting.

Safe RL. Constrained Markov Decision Processes (CMDPs) [19–21] have gained significant attention within the RL community, particularly in the context of constrained and safe real-world learning. Safe and recovery RL seeks to address two primary challenges [22]: the development of effective safety constraints [23] and the formulation of safe actions. To tackle these challenges, a substantial body of work utilizes the established online RL workflow to balance task rewards and constraints, their focus lies on optimization techniques, such as Lagrangian relaxation [24–26], Lyapunov functions [27, 28] and robustness guarantees [29–32]. However, a common issue is that these constraints are often enforced prematurely, which limits exploration and can reduce overall performance [33]. The term “recovery” in safe RL can be misleading. True recovery methods like damage adaptation [34] address post-failure adaptation, while many “recovery RL” methods actually prevent failures. Recovery RL [35] and ABS [33] predict constraint violations and employ recovery policies to avoid unsafe states before failures occur. Similarly, recent prevention methods include Control Barrier Functions [36] that ensure forward invariance of safe sets, and predictive safety filters [37] that use MPC to modify unsafe control inputs. SafeDreamer [38] integrates Lagrangian methods into the DreamerV3 planning process for model-based safe RL. While SafeDreamer focuses on traditional safe RL tasks in simulation, our FARL addresses offline-to-online post-training of pre-trained policies in the real world. Alternative approaches include hierarchical safe RL [39, 40], which utilizes structural dynamics, and methods incorporating safety certificates from control theory [41, 42].

Offline-to-online RL. Offline RL aims to address distributional shift issues that arise when a policy encounters out-of-distribution (OOD) state-action pairs. Prior methods mitigate this challenge by incorporating conservatism [4] or constraint-based regularization [5, 43, 44], thereby either discouraging the policy from exploring OOD regions or providing conservative value estimates. Once pre-trained with offline data, policies can be further improved through online fine-tuning. However, directly applying standard online off-policy RL algorithms often leads to severe performance degradation due to distributional shift during online exploration [45]. Uni-O4 [6] directly applies the PPO [46] objective to unify offline and online learning, eliminating the need for extra regularization. RL-100 [47] combines iterative offline RL with online RL to train diffusion-based policies for deployable robot learning. However, deploying such offline-to-online methods in real-world robotic systems remains unsafe and expensive due to the high risk of IR Failures during online exploration. In this work, we build upon Uni-O4 and extend it to address safety concerns in real-world manipulation tasks, enabling safer and more efficient policy refinement in real-world environments.

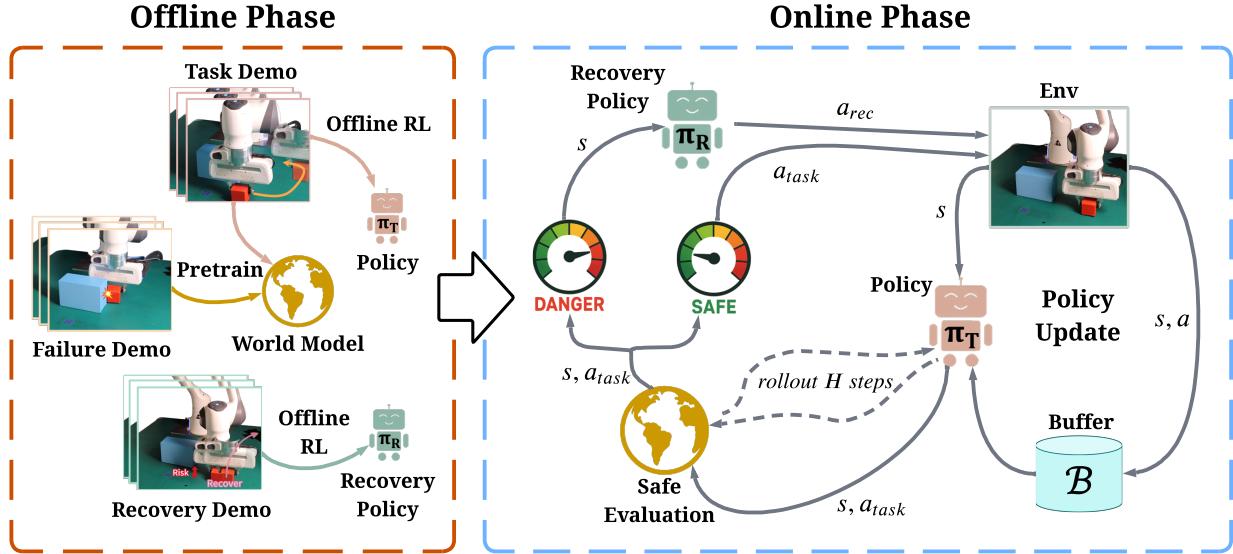


Fig. 2: The entire training pipeline consists of two main phases: 1) the offline phase, which involves pre-training the task policy, recovery policy, and world model, and 2) the online phase, during which the task policy is fine-tuned within safe exploration settings.

III. PROBLEM STATEMENT

We consider RL-based post-training under Constrained Markov Decision Processes (CMDPs) [19]. Following Thananjeyan et al. [35], we limit constraint costs to binary indicator functions that identify constraint-violating states. This can be described by tuple $\mathcal{M}(\mathcal{S}, \mathcal{A}, P(\cdot | \cdot, \cdot), R(\cdot, \cdot), \gamma, C(\cdot), \gamma_{\text{risk}}, \mu)$. Here, \mathcal{S} and \mathcal{A} are the state and action spaces. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ denotes the stochastic dynamics model which maps the given state and action to a probability distribution over the next state. $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, and γ denotes the discount factor. $C : \mathcal{S} \times \mathcal{A} \rightarrow \{0, 1\}$ is a constraint cost function, which denotes whether a state and action pair violate the designed constraint and it is associated with a threshold γ_{risk} .

Given a set of behavior policies $\pi \in \Pi$, the objective of the expected return can be defined as $R = \mathbb{E}_{\pi, \mu, P} [\sum_t \gamma^t (s_t, a_t)]$. We consider the H -steps discounted probability of constraint violation, which can be defined as

$$\begin{aligned} C_H^\pi &= \mathbb{E}_{\pi, \mu, P} \left[\sum_{i=t}^{t+H} \gamma_{\text{risk}}^{i-t} C(s_i, a_i) \right] \\ &= \sum_{i=t}^{t+H} \gamma_{\text{risk}}^{i-t} \mathbb{P}(C(s_i, a_i) = 1) \end{aligned} \quad (1)$$

in which the constraint costs are defined by the binary indicator functions.

Thus, the objective of RL under CMDPs is:

$$\pi^* = \underset{\pi \in \Pi}{\operatorname{argmax}} R^\pi \text{ s.t. } C_H^\pi \leq \epsilon_{\text{safe}} \quad (2)$$

The constraints define a set of feasible policies, i.e., $\{\pi \in \Pi : C_H^\pi \leq \epsilon_{\text{safe}}\}$. We optimize this objective over the set of feasible policies.

In this work, we propose a failure-aware offline-to-online RL algorithm designed to optimize safe policy exploration

and refinement. The core concept is inspired by recovery RL. *However, our focus is on post-training, which enhances robotic models learned from demonstrations by refining specific capabilities and providing a better policy initialization for safety, rather than relying on unnecessary random exploration from scratch.*

IV. METHOD

We formally introduce FARL, our offline-to-online failure-aware RL framework. Our overall algorithm pipeline comprises both offline and online phases, as summarized in Figure 2: 1) We first pre-train a task policy, a recovery policy, and a world model. The task policy is trained using task demonstrations that show successful task completion. The recovery policy is trained with recovery demonstrations that illustrate how to avoid or escape from near-failure states. The world model leverages both task demonstrations and failure demonstrations that capture state-action sequences leading to IR Failures. 2) Next, we fine-tune the task policy, which is optimized according to Eq. (2), and guided by the recovery policy. This policy directs the agent back to the state-action pair (s, a) where $C_H^\pi \leq \epsilon_{\text{safe}}$. During the online phase, the recovery policy and world model remain fixed to minimize IR Failures during exploration.

A. Offline Pre-training

We pre-train a task policy to serve as the initial framework for online fine-tuning. In addition, we also pre-train a recovery policy and a world model, both designed to guide the task policy's exploration and help prevent IR Failures during the online phase. For the task policy pre-training, we follow the training pipeline established in Uni-O4 [6]. Initially, the policy undergoes behavior cloning, followed by fine-tuning

using the objective:

$$J_k(\pi) = \mathbb{E}_{s \sim \rho_\pi(\cdot), a \sim \pi_k(\cdot|s)} \left[\min \left(r(\pi) A(s, a), \right. \right. \\ \left. \left. \text{clip}(r(\pi), 1 - \epsilon, 1 + \epsilon) A(s, a) \right) \right] \quad (3)$$

where ρ_π is the stationary distribution of states under policy π , $r(\pi) = \frac{\pi(a|s)}{\pi_k(a|s)}$ denotes the importance sampling ratio between the target policy π and behavior policy π_k , $\text{clip}(\cdot)$ is a conservatism operation that constrains the ratio, hyper-parameter ϵ is used to adjust the degree of conservatism, and $A(s_t, a_t)$ is the advantage function. Subsequently, we continue fine-tuning within the environment to optimize the objective in Equation 3 using GAE advantage estimation. Next, we train the recovery policy through behavior cloning, utilizing recovery demonstrations. The training process during the offline phase mirrors that of the task policy, beginning with behavior cloning and then fine-tuning via Uni-O4 in an offline context. However, we avoid fine-tuning the recovery policy during the online phase due to the limited availability of failure data. We observed that this enhances the safe exploration of the task policy in real-world environments.

We also pre-train a world model with both task and failure demonstrations, specifically for predicting future failures. The world model focuses on a limited number of near-future steps rather than the entire episode in recovery reinforcement learning [35]. Our findings indicate that planning a short distance into the future can effectively minimize IR Failures in scenarios of real-world manipulation. To achieve this, we augment the world model for failure prediction by introducing a constraint prediction head. The training objective of our world model can be defined as:

$$\mathcal{J}(\theta; \Gamma) = \sum_{i=t}^{t+H} \lambda^{i-t} \cdot \mathcal{L}(\theta; \Gamma_i), \quad (4)$$

which can be computed step-wise as follows:

$$\mathcal{L}(\theta; \Gamma_i) = c_1 \underbrace{\|R_\theta(\mathbf{z}_i, \mathbf{a}_i) - r_i\|_2^2}_{\text{reward}} \quad (5)$$

$$+ c_2 \cdot \underbrace{\text{CE}(Q_\theta(\mathbf{z}_i, \mathbf{a}_i) - (r_i + \gamma Q_{\theta^-}(\mathbf{z}_{i+1}, \pi_\theta(\mathbf{z}_{i+1}))))}_{\text{value}} \quad (6)$$

$$+ c_3 \cdot \underbrace{\text{CE}(d_\theta(\mathbf{z}_i, \mathbf{a}_i) - h_{\theta^-}(\mathbf{s}_{i+1}))}_{\text{latent state consistency}} \quad (7)$$

$$+ c_4 \underbrace{\|C_\theta(\mathbf{z}_i, \mathbf{a}_i) - c_i\|_2^2}_{\text{constraint}} \quad (8)$$

$$+ c_5 \underbrace{\|S_\theta(\mathbf{z}_i) - s_i\|_2^2}_{\text{decoder}}, \quad (9)$$

where the model consists of seven components:

Representation: $\mathbf{z}_t = h_\theta(\mathbf{s}_t)$	Policy: $\hat{\mathbf{a}}_t \sim \pi_\theta(\mathbf{z}_t)$
Latent dynamics: $\mathbf{z}_{t+1} = d_\theta(\mathbf{z}_t, \mathbf{a}_t)$	Decoder: $\hat{\mathbf{s}}_t \sim S_\theta(\mathbf{z}_t)$
Reward: $\hat{r}_t = R_\theta(\mathbf{z}_t, \mathbf{a}_t)$	Constraint: $\hat{\mathbf{c}}_t \sim C_\theta(\mathbf{z}_t, \mathbf{a}_t)$
Value: $\hat{q}_t = Q_\theta(\mathbf{z}_t, \mathbf{a}_t)$	

where s_t is the observation at time-step t , which is encoded to a representation \mathbf{z} by an encoder h_θ . Conditioned

on the representation, each head of the dynamics model predicts the representation of the next state, the reconstructed state, the constraint, a single-step reward, a state-action Q -value, and an action that maximizes the Q -function.

We incorporate value and reward signals during training to enrich latent representations while providing gradient-based regularization that prevents overfitting to constraint patterns.

After pre-training, we infer the discounted near-future constraint of π_{task} mentioned in Eq. (1) as:

$$C_H^\pi = \mathbb{E}_\pi \left[\sum_{i=t}^{t+H} \gamma^{i-t} [C_\theta(z_i, a_i) | z_i = h_\theta(s_i); \right. \\ \left. a_i = \pi_{task}(S_\theta(z_i)); z_{i+1} = d_\theta(z_i, a_i)] \right] \quad (11)$$

B. Online fine-tuning with recovery

The task policy is fine-tuned by online PPO with the policy and value network initialization from offline pre-training, following Uni-O4. However, we consider the safe exploration setting, in which we require each state-action tuple (s, a) to be safe. Let task policy π_{task} interact with the environments, each state action tuple will be checked by the world model to consider safety in the near-future steps through planning. We define the task transitions as $\mathcal{T}_{task}^\pi = (s_t, a_t^{\pi_{task}}, s_{t+1}, r_t)$, where $(s, a) \in \mathcal{S} \times \mathcal{A} : C_H^{\pi_{task}} \leq \varepsilon_{safe}$ and the recovery transitions $\mathcal{T}_{rec}^\pi = (s_t, a_t^{\pi_{rec}}, s_{t+1}, r_t)$, where $(s, a) \in \mathcal{S} \times \mathcal{A} : C_H^{\pi_{task}} \geq \varepsilon_{safe}$. In other words, if the action sampled from the task policy π_{task} under state s does not satisfy $C_H^{\pi_{task}} \leq \varepsilon_{safe}$, we would revise the task transition to the recovery transition via the action a_{rec} sampled from π_{rec} . The workflow is also described in the *safe exploration* area of the online phase in Figure 2. The safe transitions could be defined as:

$$\mathcal{T}_{safe}^\pi = \begin{cases} \mathcal{T}_{task}^\pi, & C_H^{\pi_{task}} \leq \varepsilon_{safe} \\ \mathcal{T}_{rec}^\pi, & C_H^{\pi_{task}} > \varepsilon_{safe} \end{cases} \quad (12)$$

Based on the safe transitions, we fine-tune the task policy using Objective 3 with GAE advantage estimation.

V. THEORETICAL ANALYSIS

We provide theoretical justification for FARL's superior performance through an "action correction" mechanism.

A. Preliminaries

Let $A^{\pi_{task}}(s, a)$ denote the GAE-based advantage function under policy π_{task} . We classify states based on H-step constraint violation:

$$\mathcal{A}_{rec} = \{s \in \mathcal{S} | \exists a \sim \pi_{task}(\cdot|s) : C_H^{\pi_{task}}(s, a) > \varepsilon_{safe}\}$$

We classify actions based on H-step constraint violation at the action level:

$$\mathcal{A}_{risk}(s) = \{a \in \mathcal{A} | C_H^{\pi_{task}}(s, a) > \varepsilon_{safe}\}$$

$$\mathcal{A}_{safe}(s) = \{a \in \mathcal{A} | C_H^{\pi_{task}}(s, a) \leq \varepsilon_{safe}\}$$

The risk probability at state s is defined as:

$$p_{risk}(s) = \mathbb{P}_{a \sim \pi_{task}(\cdot|s)}[a \in \mathcal{A}_{risk}(s)]$$

B. Assumptions

Assumption 1 (Non-trivial Risk Distribution): There exists a non-negligible fraction of states where the task policy samples risky actions:

$$\mathbb{E}_{s \sim \rho_{\pi_{task}}} [p_{risk}(s)] > 0$$

where $\rho_{\pi_{task}}$ is the state visitation distribution under policy π_{task} .

Assumption 2 (Probabilistic Safe Recovery): For any state s , the recovery policy provides safe actions with high probability:

$$\mathbb{P}_{a \sim \pi_{rec}(\cdot|s)} [a \in \mathcal{A}_{safe}(s)] \geq 1 - \epsilon_{rec}$$

where $\epsilon_{rec} > 0$ is the recovery failure rate.

Assumption 3 (Safe Action Advantage): For states where both safe and risky actions exist, safe actions provide better expected advantage:

$$\mathbb{E}_{a \in \mathcal{A}_{safe}(s)} [A^{\pi_{task}}(s, a)] \geq \mathbb{E}_{a \in \mathcal{A}_{risk}(s)} [A^{\pi_{task}}(s, a)] + \delta$$

where $\delta > 0$ represents the advantage gap between safe and risky actions.

C. Main Result

Theorem 1 (Action Correction Benefit): Under Assumptions 1-3, the policy improvement from FARL’s corrected transitions satisfies:

$$\begin{aligned} \Delta J_{FARL} &\geq \Delta J_{baseline} \\ &+ \mathbb{E}_{s \sim \rho_{\pi_{task}}} [p_{risk}(s)] \cdot \delta \cdot (1 - \epsilon_{rec}) - O(\epsilon_{rec}) \end{aligned} \quad (13)$$

where $p_{risk}(s) = \mathbb{P}_{a \sim \pi_{task}} [a \in \mathcal{A}_{risk}(s)]$ is the probability of sampling risky actions at state s .

Proof: [Proof Sketch] The baseline improvement is $\Delta J_{baseline} = \mathbb{E}_{s, a \sim \pi_{task}} [A^{\pi_{task}}(s, a)]$.

FARL’s action correction yields expected advantage:

$$\begin{aligned} \mathbb{E}_{FARL} [A^{\pi_{task}}(s, a)] &= (1 - p_{risk}(s)) \mathbb{E}_{a \in \mathcal{A}_{safe}(s)} [A^{\pi_{task}}(s, a)] \\ &+ p_{risk}(s) \mathbb{E}_{a \sim \pi_{rec}} [A^{\pi_{task}}(s, a)] \end{aligned} \quad (14)$$

The improvement per state is:

$$\begin{aligned} \mathbb{E}_{FARL} [A^{\pi_{task}}(s, a)] - \mathbb{E}_{\pi_{task}} [A^{\pi_{task}}(s, a)] \\ = p_{risk}(s) (\mathbb{E}_{a \sim \pi_{rec}} [A^{\pi_{task}}(s, a)] - \mathbb{E}_{a \in \mathcal{A}_{risk}(s)} [A^{\pi_{task}}(s, a)]) \end{aligned} \quad (15)$$

Under Assumptions 2-3, this difference is at least $p_{risk}(s) \cdot \delta \cdot (1 - \epsilon_{rec}) - O(\epsilon_{rec})$. Averaging over states yields the result. ■

D. Discussion

The improvement bound $\mathbb{E}_{s \sim \rho_{\pi_{task}}} [p_{risk}(s)] \cdot \delta \cdot (1 - \epsilon_{rec}) - O(\epsilon_{rec})$ shows FARL gains most when: (1) risky states are frequent ($\mathbb{E}_s [p_{risk}(s)]$ large), (2) safe actions significantly outperform risky ones (δ large), and (3) recovery demonstrations are high-quality (ϵ_{rec} small).

This explains FARL’s dual benefit: improved safety and enhanced performance by selectively replacing risky actions with recovery alternatives when necessary.

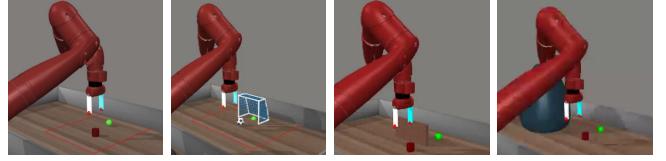


Fig. 3: Illustration of the four tasks in our FailureBench. From left to right: Bounded Push, Bounded Soccer, Fragile Push Wall, and Obstructed Push.

VI. EXPERIMENTS

A. FailureBench: Simulation Benchmark for Evaluating Failure-Aware RL

To evaluate failure-aware RL algorithms, we introduce FailureBench, a benchmark suite comprising modified versions of the MetaWorld environment [10]. FailureBench is designed to incorporate realistic failure scenarios that typically require human intervention in real-world settings. We categorize these failure scenarios into four representative tasks:

- **Sawyer Bounded Push.** The robot must push a puck to the target while keeping it within a bounded workspace. If it pushes the object beyond the boundary, it is considered an IR Failure, simulating scenarios where objects become unreachable or fall to the ground, necessitating human intervention.
- **Sawyer Bounded Soccer.** The robot must hit a ball into a goal while keeping it within a boundary. The dynamic nature of the rolling ball makes it more prone to IR Failures.
- **Sawyer Fragile Push Wall.** The robot must push an fragile object to a target position behind a wall. If the object collides with the wall, it is considered an IR Failure, simulating real-world scenarios where fragile objects would be damaged upon collision and require replacement.
- **Sawyer Obstructed Push.** The robot must navigate around a fragile vase while pushing the object to its goal location. Any collision with the vase is treated as an IR Failure.

B. Simulation Experiments

We conduct extensive experiments using our proposed FailureBench to evaluate the effectiveness of FARL. We compare FARL against the baseline Uni-O4 algorithm [6], which represents the state-of-the-art offline-to-online RL approach but without explicit failure avoidance mechanisms. Additionally, we compare our method with three state-of-the-art online safe RL algorithms, i.e., PPO-Lagrangian [13], P3O [14], and CPO [15], where each is used to fine-tune the same offline pre-trained policy.

1) *Experimental Setup:* For each environment in FailureBench, we collect three types of demonstration data for offline pre-training:

- **Task demonstrations:** 20 medium expert trajectories for each task collected using BAC [48], representing sub-optimal demonstration for manipulation tasks.

- **Recovery demonstrations:** 120 trajectories demonstrating recovery behaviors from near-failure states collected with script policy.
- **Failure demonstrations:** 20k–200k transition sequences containing failures for world model training. Specifically, we simulate online exploration by adding controlled stochastic noise during the deployment of the offline pre-trained policy. We then collect the failing trajectories, which match the distribution of failures likely to occur during online fine-tuning.

We evaluate performance using two key metrics:

- **Average Return:** The mean episodic reward improvement after 10^6 steps of online finetuning.
- **Failure Episodes:** The number of episodes containing IR Failures during 10^6 steps of online finetuning.

2) *Results and Analysis:* Figure 4 compares failure episodes between our method FARL and the baseline Uni-O4 across all four environments. Our approach consistently encounters significantly fewer failure episodes with an average reduction of **43.6%** across all tasks and up to **65.8%** in the most challenging environments (Bounded Soccer) and complex constraint scenarios (Obstructed Push) highlight our method’s strength in handling challenging interaction dynamics.

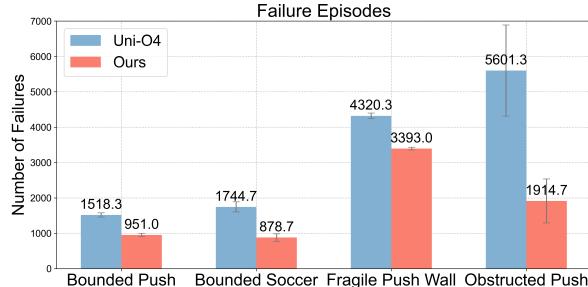


Fig. 4: Comparison of average failure episodes during fine-tuning for Uni-O4 (blue) and our method (red) in FailureBench.

Analysis of average returns (Figure 5) reveals that despite the additional constraints imposed by our failure-avoiding mechanisms, our approach achieves competitive or higher performance in terms of task rewards. This demonstrates that our method successfully balances exploration and safety, achieving effective learning without sacrificing performance.

Traditional safe RL methods significantly underperform when applied to offline-to-online scenarios, despite careful hyperparameter tuning. Table I shows that FARL outperforms these approaches by an average of over 800% across tasks, suggesting a fundamental incompatibility between standard safe RL algorithms and pre-trained policy initialization. This performance gap likely stems from their optimization objectives creating distributional shifts that prevent effective utilization of pre-trained knowledge.

3) *Ablation Studies:* To better understand the contribution of each component in FARL, we conduct two key ablation

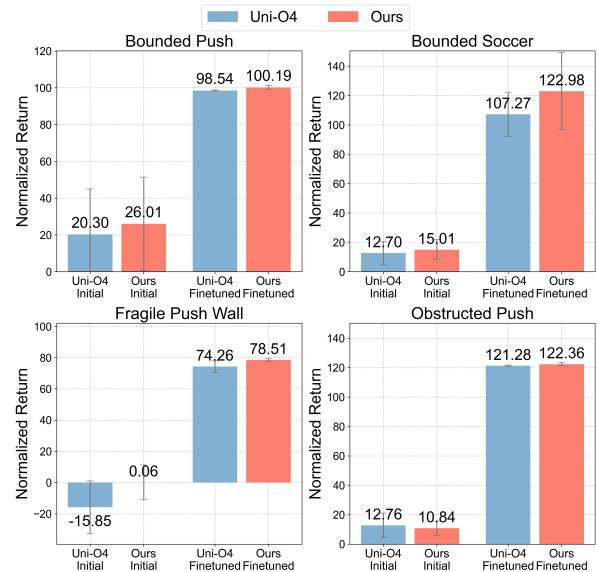


Fig. 5: Performance comparison between Uni-O4 (blue) and our method (red) in FailureBench. The bars show average return before and after fine-tuning. All returns are normalized relative to an expert script policy’s performance (100).

TABLE I: Performance comparison after fine-tuning from the same offline pre-trained policy. Values represent average returns across three seeds.

Method	Bounded Push	Bounded Soccer	Fragile Push Wall	Obstructed Push
FARL (Ours)	4593.96 ± 50.69	2276.53 ± 554.32	3220.12 ± 66.67	1227.18 ± 12.40
PPO-Lag	420.79 ± 841.07	404.80 ± 334.67	268.83 ± 695.50	543.95 ± 387.59
P3O	95.33 ± 61.51	598.55 ± 653.84	-278.62 ± 530.71	455.86 ± 348.82
CPO	156.28 ± 212.07	692.90 ± 267.52	-994.55 ± 1139.89	47.40 ± 37.26

studies focused on our failure prediction and avoidance mechanisms:

Study 1. World Model vs. Recovery-RL Safety Critic

First, we investigate whether our world-model-based safety critic with future rollout prediction is necessary by replacing it with a Q-function safety critic from Recovery-RL [35], which is based on a simple MLP. This approach estimates the discounted future probability of constraint violation under the current policy:

$$Q_{\text{safe}}(s_t, a_t) = \mathbb{E}_{\pi} \left[\sum_{t'=t}^{\infty} \gamma_{\text{safe}}^{t'-t} c_{t'} | s_t, a_t \right] \quad (16)$$

$$= c_t + (1 - c_t) \gamma_{\text{safe}} \mathbb{E}_{\pi} [Q_{\text{safe}}(s_{t+1}, a_{t+1}) | s_t, a_t]. \quad (17)$$

Study 2. Recovery Policy vs. MPPI Planning

Second, we evaluate the necessity of a separate pre-trained recovery policy by replacing it with Model Predictive Path Integral (MPPI) [49], a sampling-based model predictive control method. Our implementation plans actions by:

- Sampling multiple action trajectories from a Gaussian distribution.
- Computing rewards and constraints for each trajectory using the world model.
- Filtering trajectories based on a safety threshold ($\varepsilon_{\text{safe}}$).
- Selecting the best trajectory among safe options according to expected returns.

Result Analysis

Table II presents the failure episode results of our ablation studies. The performance across different variants reveals several interesting insights:

Replacing our world-model-based safety critic with the simple critic from Recovery-RL [35] leads to a substantial increase in failure episodes in all environments, which is the most pronounced in Bounded Soccer with 92% more failures, where complex dynamics make accurate long-term risk assessment crucial. This confirms that our approach's explicit modeling of future state-action sequences provides superior prediction of potential failures compared to a simple Q-function estimator.

MPPI planning underperforms our learned recovery policy, particularly in environments with complex dynamics. This suggests that planning-based methods that implicitly infer recovery action from the world model encounter difficulties in highly dynamic environments. Our recovery policy leverages expert demonstrations of recovery behaviors that would be difficult to discover through planning alone, providing a significant advantage in handling complex failure scenarios.

These results validate our design choices and demonstrate that our combined approach of a world-model-based safety critic and a pre-trained recovery policy provides the most robust performance across diverse failure scenarios.

The return comparison in Figure 6 further illustrates that our FARL consistently achieves superior final performance after fine-tuning.

TABLE II: Comparison of Average Failure Episodes across FailureBench environments.

Failure Episodes ↓	FARL (Ours)	Rec-RL	MPPI
Bounded Push	951.00 ± 52.37	1250.67 ± 293.33	1112.00 ± 180.01
Bounded Soccer	878.67 ± 120.65	1687.00 ± 130.85	2022.67 ± 279.74
Fragile Push Wall	3393.00 ± 42.51	4234.00 ± 56.93	3753.00 ± 13.23
Obstructed Push	1914.67 ± 711.46	2175.00 ± 128.42	2229.00 ± 727.12

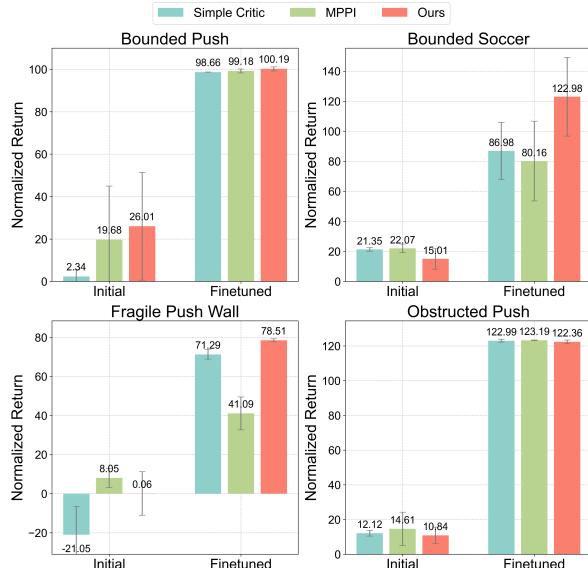


Fig. 6: Comparison of average returns before and after fine-tuning for our method and the two ablation variants. All returns are normalized relative to an expert script policy's performance (100).

TABLE III: Average return comparison for three Franka tasks.

Task	Method	Initial	Finetuned
Franka Fragile Push Wall	Uni-O4 FARL (Ours)	343.28 ± 121.23 349.95 ± 123.12	369.15 ± 8.93 363.65 ± 14.38
Franka Disturbed Push	Uni-O4 FARL (Ours)	262.62 ± 129.59 321.49 ± 149.57	308.29 ± 79.83 384.52 ± 87.45
Franka Bounded Soccer	Uni-O4 FARL (Ours)	512.79 ± 143.34 578.77 ± 133.80	615.97 ± 103.20 638.96 ± 91.59

C. Real-World Experiments

To validate the practical effectiveness of FARL, we deploy FARL on a Franka Emika Panda robot to evaluate its performance in challenging real-world scenarios. We implement three representative tasks and assess both failure reduction and performance improvement (Figure 1).

1) Experimental Setup:

- **Franka Fragile Push Wall.** The robot must push an object, which is assumed to be fragile, to a target position behind a wall. If the object collides with the wall during manipulation, it is considered damaged and requires replacement, indicating an IR Failure that necessitates human intervention.
- **Franka Disturbed Push.** The robot must push an object to a target while avoiding a dynamic obstacle (a decorative flower) that is randomly moved by a human operator, simulating unpredictable environmental changes.
- **Franka Bounded Soccer.** The robot must use a UMI gripper [50] to "kick" a ball toward a target on an uneven turf surface. If the ball rolls beyond the defined boundaries due to the irregular surface dynamics, it is considered an IR Failure.

We use an Intel RealSense D435 camera for real-time image acquisition. State estimation is performed using YOLOv8 object detection and color-based filtering to track the positions of objects in the workspace. The system operates at approximately 5Hz for both perception and control.

For each task, we conduct training sessions consisting of 50 episodes and compared FARL against the baseline algorithm. We collect 40-80 trajectories for three types of demonstration via teleoperation. Each demonstration type requires approximately 10-20 minutes of teleoperation, resulting in a total human effort of 30-60 minutes per task for collecting all three demonstration types. For the failure demonstrations, specifically, we guide the robot via teleoperation into various failure states that the robot would likely encounter during autonomous execution, providing representative data for safety critic pre-training.

2) Results and Analysis: Figure 7 presents a comparison of total failures during the 50-episode training phase for all environments. The results demonstrate a dramatic reduction in failures when using FARL compared to the baseline.

Table III shows the average return achieved by each method before and after fine-tuning, which further highlights the necessity and superiority of online RL finetuning. Online

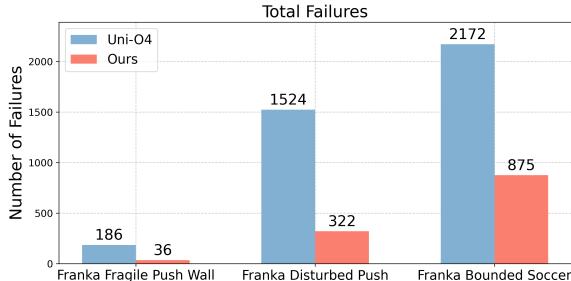


Fig. 7: Comparison of total failure state-action pairs during 50 training episodes in real-world experiments.

RL finetuning not only boosts task performance but also significantly decreases the standard deviation, indicating more robust policies. In the Disturbed Push and Bounded Soccer tasks, the finetuned policy achieves noticeably higher returns. This result signifies that FARL successfully leverages the privilege of online RL finetuning while minimizing failures requiring extensive human intervention in the real world.

VII. CONCLUSION AND FUTURE WORK

We introduce a failure-aware offline-to-online reinforcement learning framework that integrates a world-model-based safety critic and a recovery policy to reduce IR Failures during exploration significantly.

Limitations and future work. Our current work does not incorporate additional modalities such as 2D/3D vision, tactile sensing, and others, which could further enrich the perceptual capabilities of the system. In future work, we plan to extend FARL to address mobile manipulation and dual-arm robotic platforms and to integrate these other modalities.

Another promising direction is to pre-train large-scale failure world models across diverse tasks and embodiments, potentially enabling better generalization to unseen failure modes in new domains.

ACKNOWLEDGEMENTS

We would like to thank Zhecheng Yuan, Tianming Wei, Guangqi Jiang, and Xiayao Wang for their valuable discussions. This work was supported by the Tsinghua University Dushi Program.

REFERENCES

- [1] Y. Wu, G. Tucker, and O. Nachum, “Behavior regularized offline reinforcement learning,” *ArXiv preprint*, vol. abs/1911.11361, 2019.
- [2] M. Shridhar, L. Manuelli, and D. Fox, “Perceiver-actor: A multi-task transformer for robotic manipulation,” in *CoRL*, PMLR, 2023, pp. 785–799.
- [3] C. Chi et al., “Diffusion policy: Visuomotor policy learning via action diffusion,” in *RSS*, 2023.
- [4] A. Kumar, A. Zhou, G. Tucker, and S. Levine, “Conservative q-learning for offline reinforcement learning,” in *NeurIPS*, 2020.
- [5] I. Kostrikov, A. Nair, and S. Levine, “Offline reinforcement learning with implicit q-learning,” in *ICLR*, 2022.
- [6] K. Lei et al., “Uni-o4: Unifying online and offline deep reinforcement learning with multi-step on-policy optimization,” in *ICLR*, 2024.
- [7] B. Eysenbach, S. Gu, J. Ibarz, and S. Levine, “Leave no trace: Learning to reset for safe and autonomous reinforcement learning,” in *ICLR*, 2018.
- [8] A. Gupta et al., “Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention,” in *ICRA*, 2021, pp. 6664–6671.
- [9] A. Sharma et al., “Autonomous reinforcement learning: Formalism and benchmarking,” in *ICLR*, 2022.
- [10] T. Yu et al., “Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning,” vol. abs/1910.10897, 2019.
- [11] M. Nakamoto et al., “Cal-ql: Calibrated offline RL pre-training for efficient online fine-tuning,” in *NeurIPS*, 2023.
- [12] J. Li et al., “Proto: Iterative policy regularized offline-to-online reinforcement learning,” *ArXiv preprint*, vol. abs/2305.15669, 2023.
- [13] A. Ray, J. Achiam, and D. Amodei, “Benchmarking safe exploration in deep reinforcement learning,” *ArXiv preprint*, vol. abs/1910.01708, 2019.
- [14] L. Zhang et al., “Penalized proximal policy optimization for safe reinforcement learning,” in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, 2022, pp. 3744–3750.
- [15] J. Achiam, D. Held, A. Tamar, and P. Abbeel, “Constrained policy optimization,” in *ICML*, ser. Proceedings of Machine Learning Research, vol. 70, 2017, pp. 22–31.
- [16] Z. Yuan et al., “Hermes: Human-to-robot embodied learning from multi-source motion data for mobile dexterous manipulation,” *arXiv preprint arXiv:2508.20085*, 2025.
- [17] Z. He et al., “Learning visual quadrupedal loco-manipulation from demonstrations,” in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2024, pp. 9102–9109.
- [18] G. Jiang et al., “Robots pre-train robots: Manipulation-centric robotic representation from large-scale robot datasets,” *arXiv preprint arXiv:2410.22325*, 2024.
- [19] E. Altman, *Constrained Markov decision processes*. 2021.
- [20] T. Ji et al., “Ace: Off-policy actor-critic with causality-aware entropy regularization,” *arXiv preprint arXiv:2402.14528*, 2024.
- [21] G. Xu et al., “Drm: Mastering visual reinforcement learning through dormant ratio minimization,” *arXiv preprint arXiv:2310.19668*, 2023.
- [22] P. Liu, H. Bou-Ammar, J. Peters, and D. Tateo, “Safe reinforcement learning on the constraint manifold: Theory and applications,” *ArXiv preprint*, vol. abs/2404.09080, 2024.
- [23] A. Wachi, X. Shen, and Y. Sui, “A survey of constraint formulations in safe reinforcement learning,” in *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, 2024, pp. 8262–8271.
- [24] Q. Liang, F. Que, and E. Modiano, “Accelerated primal-dual policy optimization for safe reinforcement learning,” *ArXiv preprint*, vol. abs/1802.06480, 2018.
- [25] K. Srinivasan, B. Eysenbach, S. Ha, J. Tan, and C. Finn, “Learning to be Safe: Deep RL with a Safety Critic,” *ArXiv preprint*, vol. abs/2010.14603, 2020.
- [26] C. Tessler, D. J. Mankowitz, and S. Mannor, “Reward constrained policy optimization,” in *ICLR*, 2019.
- [27] Y. Chow, O. Nachum, E. A. Duéñez-Guzmán, and M. Ghavamzadeh, “A lyapunov-based approach to safe reinforcement learning,” in *NeurIPS*, 2018, pp. 8103–8112.
- [28] Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Ghavamzadeh, “Lyapunov-based safe policy optimization for continuous control,” *ArXiv preprint*, vol. abs/1901.10031, 2019.
- [29] Y. Liang, Y. Sun, R. Zheng, and F. Huang, “Efficient adversarial training without attacking: Worst-case-aware robust reinforcement learning,” in *NeurIPS*, 2022.
- [30] Z. Liu et al., “On the robustness of safe reinforcement learning under observational perturbations,” in *ICLR*, 2023.
- [31] Y. Liang et al., “Game-theoretic robust reinforcement learning handles temporally-coupled perturbations,” in *ICLR*, 2024.
- [32] X. Liu, C. Deng, Y. Sun, Y. Liang, and F. Huang, “Beyond worst-case attacks: Robust rl with adaptive defense via non-dominated policies,” *arXiv preprint arXiv:2402.12673*, 2024.
- [33] T. He et al., “Agile but safe: Learning collision-free high-speed legged locomotion,” in *RSS*, 2024.
- [34] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, “Robots that can adapt like animals,” *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.
- [35] B. Thananjeyan et al., “Recovery rl: Safe reinforcement learning with learned recovery zones,” *RAL*, vol. 6, no. 3, pp. 4915–4922, 2021.
- [36] A. D. Ames et al., “Control barrier functions: Theory and applications,” in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

- [37] K. P. Wabersich and M. N. Zeilinger, “A predictive safety filter for learning-based control of constrained nonlinear dynamical systems,” *Automatica*, vol. 129, p. 109597, 2021.
- [38] W. Huang, J. Ji, C. Xia, B. Zhang, and Y. Yang, “Safedreamer: Safe reinforcement learning with world models,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [39] G. Dalal et al., “Safe exploration in continuous action spaces,” *ArXiv preprint*, vol. abs/1801.08757, 2018.
- [40] W. Xiao, T. He, J. M. Dolan, and G. Shi, “Safe deep policy adaptation,” in *ICRA*, 2024, pp. 17286–17292.
- [41] 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 *AAAI*, 2019, pp. 3387–3395.
- [42] Y. K. Nakka et al., “Chance-Constrained Trajectory Optimization for Safe Exploration and Learning of Nonlinear Systems,” *ArXiv preprint*, vol. abs/2005.04374, 2020.
- [43] D. Brandfonbrener, W. Whitney, R. Ranganath, and J. Bruna, “Offline RL without off-policy evaluation,” in *NeurIPS*, 2021, pp. 4933–4946.
- [44] Z. Zhuang, K. Lei, J. Liu, D. Wang, and Y. Guo, “Behavior proximal policy optimization,” in *ICLR*, 2023.
- [45] Z. Yuan et al., *Rl-vigen: A reinforcement learning benchmark for visual generalization*, 2023. arXiv: [2307.10224 \[cs.AI\]](https://arxiv.org/abs/2307.10224).
- [46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” *ArXiv preprint*, vol. abs/1707.06347, 2017.
- [47] K. Lei et al., “Rl-100: Performant robotic manipulation with real-world reinforcement learning,” *arXiv preprint arXiv:2510.14830*, 2025.
- [48] T. Ji et al., “Seizing serendipity: Exploiting the value of past success in off-policy actor-critic,” in *ICML*, 2024.
- [49] G. Williams, A. Aldrich, and E. Theodorou, *Model predictive path integral control using covariance variable importance sampling*, 2015.
- [50] C. Chi et al., “Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots,” in *RSS*, 2024.