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 Zang[4,5], Kaizhe Hu[1,3], Yongyuan Liang[6], Bo An[4], Xiaoli Li[5,4], Huazhe Xu[1,3]

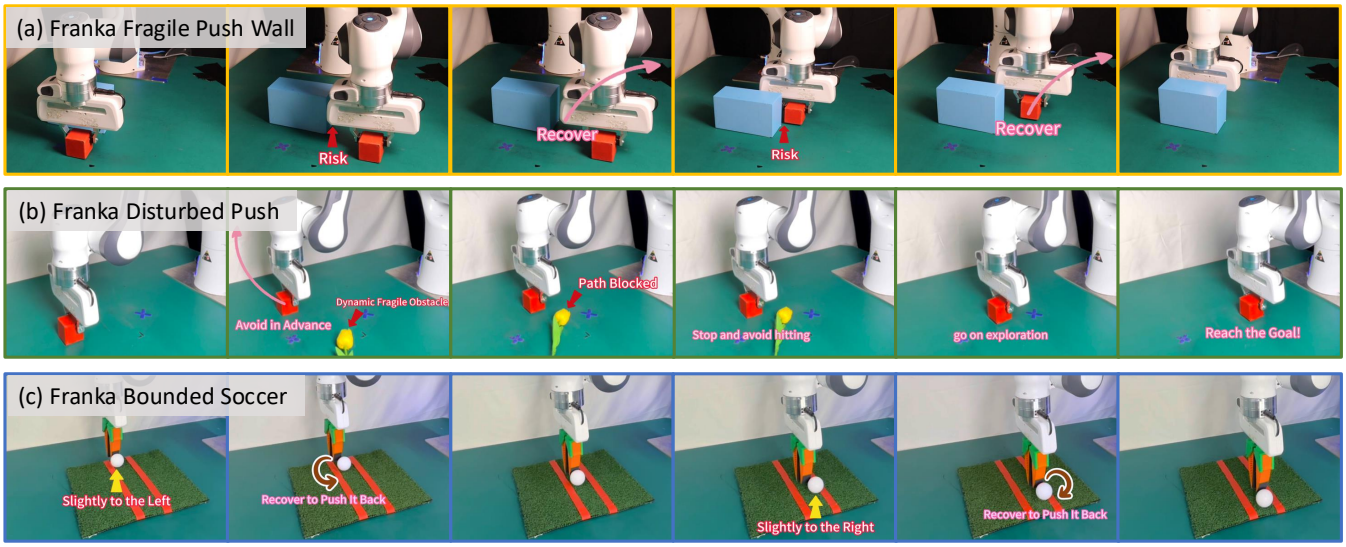

Fig. 1: FARL demonstrated on a Franka Emika Panda robot across three manipulation tasks: (a) pushing fragile objects while avoiding wall collisions, (b) pushing with dynamic obstacle avoidance, (c) soccer with boundary constraints (detailed in Section VI.C.1). FARL predicts potential failures and executes 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 (FARL), a framework for 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 FARL in significantly reducing IR Failures while improving performance and generalization during online reinforcement learning post-training. FARL reduces IR Failures by 73.1% while elevating performance by 11.3% on average during real-world RL post-training.**

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. 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 **F**ailure-**A**ware Offline-to-Online **R**einforcement **L**earning (FARL), 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. To study various failure scenarios in simulation, we introduce **FailureBench**, a benchmark built

*Equal contribution. [1]Shanghai Qi Zhi Institute, [2]SJTU, [3]THU, [4]NTU, [5]A*STAR, [6]UMD. Project Website: https://failure-aware-rl.github.io

upon existing environments [10] that incorporates common IR Failure scenarios requiring human intervention. 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 and real-world settings, while simultaneously enhancing task performance and generalizability. **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, 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

Safe RL. Safe RL has been extensively studied, mainly in learning-from-scratch scenarios [13–15]. Traditional approaches via Lagrangian relaxation [16], Lyapunov functions [17], or Control Barrier Functions [18] often enforce constraints prematurely, limiting exploration [19]. Recovery RL [20] and ABS [19] predict constraint violations and employ recovery policies to preemptively avoid unsafe states. SafeDreamer [21] integrates Lagrangian methods into model-based planning. Unlike these, FARL targets offline-to-online post-training in the real world.

Offline-to-online RL. Offline RL addresses distributional shift via conservatism [4] or regularization [5]. Uni-O4 [6] unifies offline and online learning with a PPO [22] objective, while RL-100 [23] combines iterative offline RL with online fine-tuning for diffusion-based policies. However, deploying such methods in real-world remains unsafe due to IR Failures during exploration. We extend Uni-O4 with failure-aware mechanisms for safer policy refinement.

III. Method

We formulate RL-based post-training under Constrained Markov Decision Processes (CMDPs) [24], defined by tuple $\mathcal{M}(\mathcal{S}, \mathcal{A}, P, R, \gamma, C, \gamma_{\text{risk}}, \mu)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, dynamics $P : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0, 1]$, reward $R : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$, discount factor γ, and binary constraint cost $C : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$ [20] with discount factor γ_{risk} for future constraint violations. The H-step discounted constraint violation probability is:

$$C_H^\pi = \sum_{i=t}^{t+H} \boldsymbol{\gamma}_{\text{risk}}^{i-t} \mathbb{P}\left(\boldsymbol{C}(s_i, a_i) = 1\right) \tag{1}$$

The CMDP objective is:

$$\pi^* = \underset{\pi\in\Pi}{\operatorname{argmax}} R^\pi \text{ s.t. } C_H^\pi \le \boldsymbol{\varepsilon}_{\text{safe}} \tag{2}$$

We formally introduce FARL, our offline-to-online failure-aware RL framework that enhances robotic models learned from demonstrations. The overall pipeline comprises offline and online phases, as summarized in Figure 2.

A. Offline Pre-training

We pre-train a task policy, a recovery policy, and a world model. The task policy is trained using task demonstrations; the recovery policy with recovery demonstrations that illustrate how to escape near-failure states; and the world model with both task and failure demonstrations. For the task policy pre-training, we follow Uni-O4 [6]: the policy undergoes behavior cloning, followed by fine-tuning using the objective:

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

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, clip$(\cdot)$ is a conservatism operation that constrains the ratio, hyperparameter 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 using recovery demonstrations, following the same offline pipeline as the task policy. Specifically, we keep the recovery policy fixed during the online phase due to limited failure data, which we found enhances safe exploration in real-world environments.

The world model is pre-trained to predict future failures over a limited number of near-future steps, rather than the entire episode as in Recovery RL [20]. We augment the world model with a constraint prediction head. The training objective is:

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

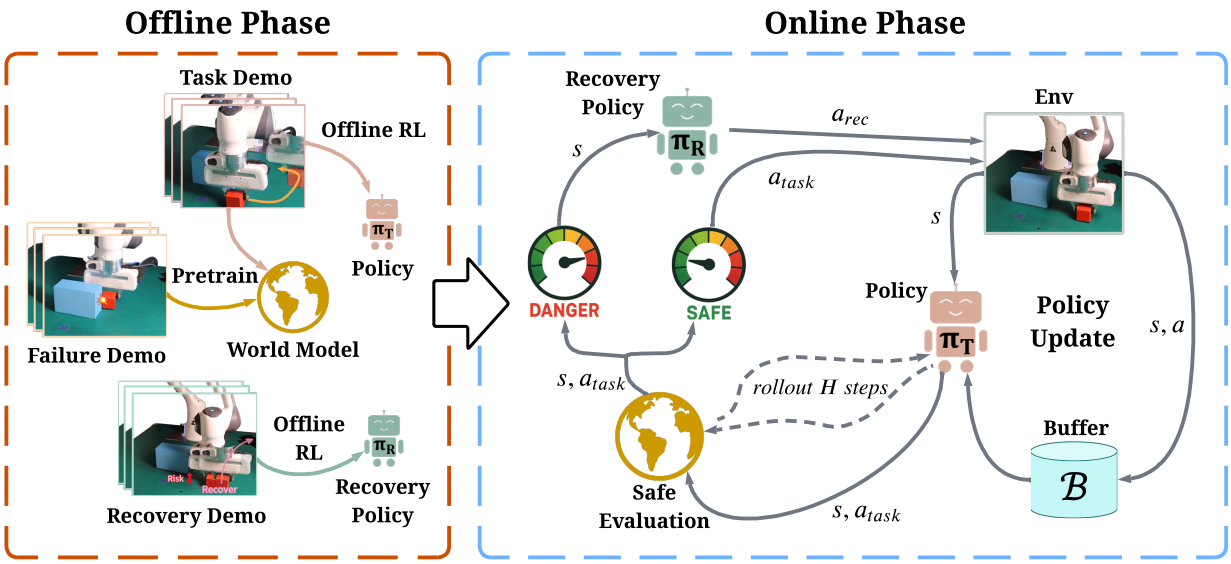

Fig. 2: The entire training pipeline consists of two main phases: 1) the offline phase, which involves pre-training the task policy, recovery policy, and world model, and 2) the online phase, during which the task policy is fine-tuned within safe exploration settings.

which can be computed step-wise as follows:

$$\mathcal{L}(\theta;\Gamma_i) = c_1 \underbrace{\mathrm{CE}(R_\theta(\mathbf{z}_i,\mathbf{a}_i),\ r_i)}_{\text{reward}} \tag{5}$$

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

$$+ c_3 \cdot \underbrace{\|d_\theta(\mathbf{z}_i,\mathbf{a}_i) - h_{\theta^-}(\mathbf{s}_{i+1})\|_2^2}_{\text{latent state consistency}} \tag{7}$$

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

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

where h_θ encodes observations s_t into latent representations $\mathbf{z}_t$, $d_\theta(\mathbf{z}_t,\mathbf{a}_t)$ predicts the next latent state, R_θ, Q_θ, C_θ predict rewards, values, and constraints, S_θ decodes latent states back to observations, and π_θ outputs actions. 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:

$$C_H^\pi = \mathbb{E}_{\boldsymbol{\pi}}\Bigg[\sum_{i=t}^{t+H} \gamma_{\text{risk}}^{i-t}\big[C_\theta(z_i,a_i)\,|\,z_i = h_\theta(s_i);\ a_i = \pi_{task}(S_\theta(z_i));\ z_{i+1} = d_\theta(z_i,a_i)\big]\Bigg] \tag{10}$$

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. We consider the safe exploration setting, in which each state-action tuple (s,a) is checked by the world model for near-future safety. We define the task transitions as $\mathcal{T}_{task}^\pi = (s_t, a_t^{\pi_{task}}, s_{t+1}, r_t)$, where $(s,a) \in \mathcal{S}\times\mathcal{A} : C_H^{\pi_{task}} \le \varepsilon_{\text{safe}}$ and the recovery transitions $\mathcal{T}_{rec}^\pi = (s_t, a_t^{\pi_{rec}}, s_{t+1}, r_t)$, where $(s,a) \in \mathcal{S}\times\mathcal{A} : C_H^{\pi_{task}} > \varepsilon_{\text{safe}}$. The safe transitions are defined as:

$$\mathcal{T}_{\text{safe}}^\pi = \begin{cases} \mathcal{T}_{\text{task}}^\pi, & C_H^{\pi_{task}} \le \varepsilon_{\text{safe}} \\ \mathcal{T}_{\text{rec}}^\pi, & C_H^{\pi_{task}} > \varepsilon_{\text{safe}} \end{cases} \tag{11}$$

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

IV. 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 actions based on H-step constraint violation:

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

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

The risk probability at state s is defined as:

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

We denote $\bar{A}^\pi_{safe}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)|a\in\mathcal{A}_{safe}(s)}[A^{\pi_{task}}(s,a)]$ and $\bar{A}^\pi_{risk}(s)$ analogously for $\mathcal{A}_{risk}(s)$.

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 from the recovery policy provide better expected advantage than risky actions from the task policy:

$$\bar{A}^{\pi_{rec}}_{safe}(s) \geq \bar{A}^{\pi_{task}}_{risk}(s) + \delta$$

where $\delta > 0$ represents the advantage gap.

C. Main Result

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

$$\begin{aligned}\Delta J_{FARL} \geq &\Delta J_{baseline} \\ &+ \mathbb{E}_{s\sim\rho_{\pi_{task}}}[p_{risk}(s)] \cdot \delta \cdot (1 - \epsilon_{rec}) - O(\epsilon_{rec})\end{aligned} \tag{12}$$

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:

$$\begin{aligned}\mathbb{E}_{FARL}[A^{\pi_{task}}(s,a)] = &(1 - p_{risk}(s))\bar{A}^{\pi_{task}}_{safe}(s) \\ &+ p_{risk}(s)\mathbb{E}_{a\sim\pi_{rec}}[A^{\pi_{task}}(s,a)]\end{aligned} \tag{13}$$

The improvement per state is:

$$\begin{aligned}&\mathbb{E}_{FARL}[A^{\pi_{task}}(s,a)] - \mathbb{E}_{\pi_{task}}[A^{\pi_{task}}(s,a)] \\ &= p_{risk}(s)\left(\mathbb{E}_{a\sim\pi_{rec}}[A^{\pi_{task}}(s,a)] - \bar{A}^{\pi_{task}}_{risk}(s)\right)\end{aligned} \tag{14}$$

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 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.

V. 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:

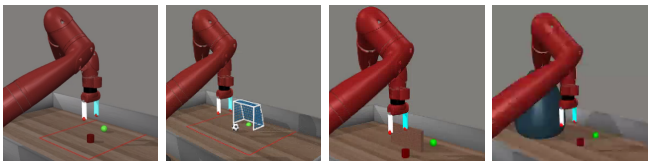

Fig. 3: Illustration of the four tasks in our FailureBench. From left to right: Bounded Push, Bounded Soccer, Fragile Push Wall, and Obstructed Push.

- **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 a 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 [25], 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. The larger improvements in highly dynamic environments (Bounded Soccer) and complex constraint scenarios (Obstructed Push) highlight our method's strength in handling challenging interaction dynamics.

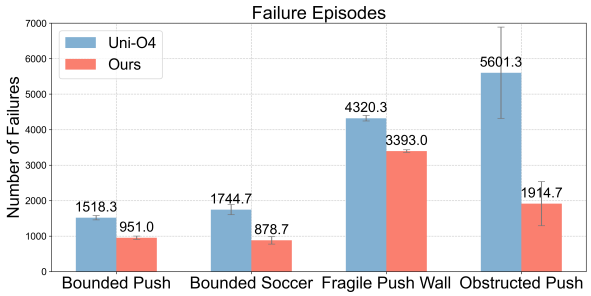

Fig. 4: Comparison of average failure episodes during fine-tuning for Uni-O4 (blue) and our method (red) in FailureBench.

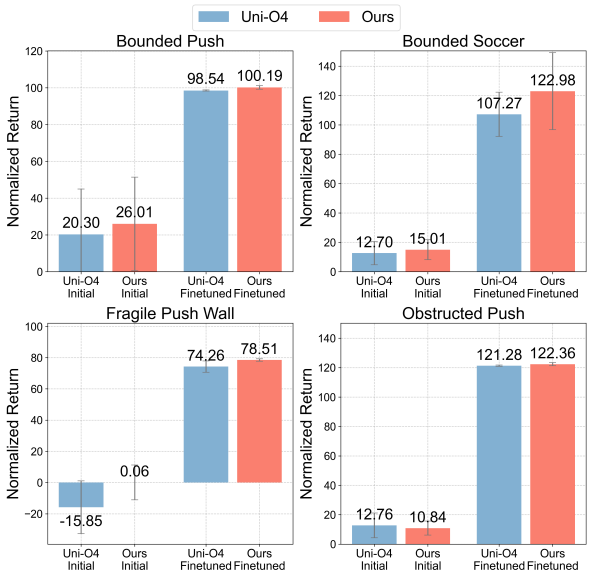

Fig. 5: Performance comparison between Uni-O4 (blue) and our method (red) in FailureBench. The bars show average return before and after fine-tuning. All returns are normalized relative to an expert script policy's performance (100).

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.

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

3) Ablation Studies: To better understand the contribution of each component in FARL, we conduct two key ablation 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 [20], which is based on a simple MLP. This approach estimates the discounted future probability of constraint violation under the current policy:

$$Q_{\text{safe}}(s_t, a_t) = \mathbb{E}_\pi \left[\sum_{t'=t}^{\infty} \gamma_{\text{safe}}^{t'-t} c_{t'} | s_t, a_t \right] \tag{15}$$

$$= c_t + (1 - c_t)\gamma_{\text{safe}} \mathbb{E}_\pi \left[Q_{\text{safe}}(s_{t+1}, a_{t+1}) | s_t, a_t \right]. \tag{16}$$

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) [26], 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. Replacing our world-model-based safety critic with the simple critic from Recovery-RL [20] leads to substantially more failures in all environments, most pronounced in Bounded Soccer with a 92% increase in failure episodes, 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 also underperforms our learned recovery policy, particularly in dynamic environments, suggesting that expert recovery demonstrations capture behaviors difficult to discover through planning alone. The return comparison in Figure 6 further confirms that FARL 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	$\mathbf{951.00 \pm 52.37}$	1250.67 ± 293.33	1112.00 ± 180.01
Bounded Soccer	$\mathbf{878.67 \pm 120.65}$	1687.00 ± 130.85	2022.67 ± 279.74
Fragile Push Wall	$\mathbf{3393.00 \pm 42.51}$	4234.00 ± 56.93	3753.00 ± 13.23
Obstructed Push	$\mathbf{1914.67 \pm 711.46}$	2175.00 ± 128.42	2229.00 ± 727.12

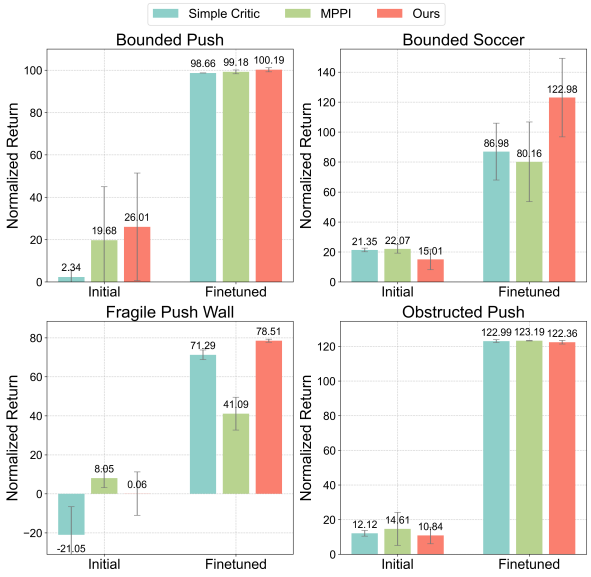

Fig. 6: Comparison of average returns before and after fine-tuning for our method and the two ablation variants. All returns are normalized relative to an expert script policy's performance (100).

TABLE III: Average return comparison for three Franka tasks.

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

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).

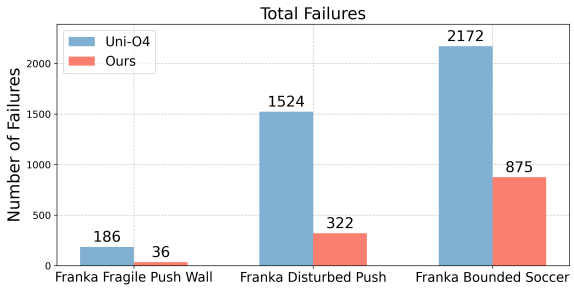

Fig. 7: Comparison of total failure state-action pairs during 50 training episodes in real-world experiments.

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 [27] 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 the recovery demonstrations, the teleoperator identifies when the robot is approaching a failure state and demonstrates corrective actions. For the failure demonstrations, we guide the robot into various failure states for safety critic pretraining.

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 before and after fine-tuning. Online RL finetuning boosts task performance and decreases variance, with particularly notable gains in Disturbed Push and Bounded Soccer. This confirms that FARL successfully leverages online RL finetuning while minimizing failures requiring human intervention.

ACKNOWLEDGMENTS

We would like to thank Zhecheng Yuan, Tianming Wei, Guangqi Jiang, and Xiyao 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 *IJCAI*, 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] Q. Liang, F. Que, and E. Modiano, "Accelerated primal-dual policy optimization for safe reinforcement learning," *ArXiv preprint*, vol. abs/1802.06480, 2018.

[17] 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.

[18] A. D. Ames *et al.*, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[19] T. He *et al.*, "Agile but safe: Learning collision-free high-speed legged locomotion," in *RSS*, 2024.

[20] B. Thananjeyan *et al.*, "Recovery rl: Safe reinforcement learning with learned recovery zones," *RAL*, vol. 6, no. 3, pp. 4915–4922, 2021.

[21] W. Huang, J. Ji, C. Xia, B. Zhang, and Y. Yang, "Safedreamer: Safe reinforcement learning with world models," in *ICLR*, 2024.

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *ArXiv preprint*, vol. abs/1707.06347, 2017.

[23] K. Lei *et al.*, "Rl-100: Performant robotic manipulation with real-world reinforcement learning," *arXiv preprint arXiv:2510.14830*, 2025.

[24] E. Altman, *Constrained Markov decision processes*. 2021.

[25] T. Ji *et al.*, "Seizing serendipity: Exploiting the value of past success in off-policy actor-critic," in *ICML*, 2024.

[26] G. Williams, A. Aldrich, and E. Theodorou, *Model predictive path integral control using covariance variable importance sampling*, 2015.

[27] C. Chi *et al.*, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *RSS*, 2024.