

SafeExplorer: An Unbiased Mixed-Policy Gradient for Reinforcement Learning with Recovery Interventions

Elham Daneshmand
McGill University & Mila

elham.daneshmand@mail.mcgill.ca

Majid Khadiv
Technical University of Munich

majid.khadiv@tum.de

Glen Berseth
Université de Montréal & Mila

glen.bersth@umontreal.ca

Hsiu-Chin Lin
McGill University & Mila

hsiu-chin.lin@mcgill.ca

Reviewed on OpenReview: <https://openreview.net/forum?id=ppOrbSsMM0>

Abstract

Training agents directly on the task they will be deployed on often leads to the best policy performance, but training on physical robots can cause damage through falls, so minimizing training falls is critical. A standard mitigation hands control to a separate *recovery policy* whenever the agent leaves a designer-specified *safe region*. The resulting mixed-policy rollouts bias the on-policy update, and importance sampling cannot remove this bias when the recovery policy is deterministic. We propose SafeExplorer, a drop-in modification of proximal policy optimization (PPO) built on an unbiased policy-gradient estimator of the mixed-policy return. The estimator uses the score function only inside the safe region and never evaluates the recovery policy’s density, so deterministic recovery policies pose no problem. On final reward, the estimator matches or outperforms truncated importance sampling, which is itself biased for that same gradient. SafeExplorer also speeds up learning near the safe-region boundary with a closed-form critic target at recovery-triggering states when dynamics and recovery are deterministic, and an imitation loss that copies recovery actions only from successful recoveries. On three simulated locomotion environments with ten seeds each, and given an already available recovery controller, SafeExplorer incurs $255\times$, $43\times$, and $21\times$ fewer training falls than standard PPO on HalfCheetah, Ant, and Unitree Go1. SafeExplorer also matches or exceeds the final reward of PPO. On Ant, where 8.7% of recovery interventions end in a fall, over $40\times$ the HalfCheetah rate, SafeExplorer reaches the success threshold on every seed and none of the five baselines reaches the threshold on any seed.

1 Introduction

Reinforcement learning (RL) policies often perform best when trained directly on the task they will be deployed on, but moving that training onto a physical robot is hard for a reason unrelated to asymptotic performance: learning requires failures, and on real hardware every failure has a price. A *fall*, a loss of balance that ends the episode, can damage the platform and cannot be undone the way a simulator reset can. Recent methods learn capable controllers by collecting hundreds of thousands of episodes in massively parallel simulation (Lee et al., 2020; Rudin et al., 2021; Agarwal et al., 2022), where these failures are free,

The authors used a large language model (Claude, Anthropic) to assist with drafting and editing the text and with writing analysis code. The authors checked all content and take full responsibility for it.

but on a real robot they are not (Ibarz et al., 2021; Smith et al., 2023b). What gates real-world training is therefore not the final reward but the number of falls spent reaching it, and the goal of SafeExplorer is to keep that number small. Falls are not the only obstacle. Sample complexity binds from-scratch real-world training, and our experiments are simulated at simulation-scale budgets. We therefore frame the contribution as an unbiased gradient estimator for PPO pipelines, from scratch or fine-tuning a sim-pretrained policy with the same optimizer, not as a hardware-practicality claim.

The common way to keep learning on real hardware is to pair the agent with a separate *recovery policy*, a controller that takes over whenever the agent leaves a designer-specified *safe region* and steers the system back toward a nominal state, upright standing in ours (Section 3). Because the safe region is set conservatively, the agent leaves it routinely, so the recovery policy fires far more often than a fall would occur, converting most would-be falls into cheap, reversible interventions. The controller is easy to assemble from off-the-shelf parts, a model predictive control (MPC) solver (Pua & Khadiv, 2024), a soft actor-critic (SAC) expert (Haarnoja et al., 2018), or a learned recovery controller (Lee et al., 2019). But it carries a hidden cost: the data the agent collects is no longer produced by the agent alone, but by a *mixed policy*, the main policy inside the safe region and the recovery outside it, so each rollout is a blend of two controllers rather than a sample from the one being improved.

This blend breaks the on-policy assumption. PPO (Schulman et al., 2017) and related methods estimate the gradient of the policy that produced the rollout, so they are valid only when the rollout distribution matches the policy being updated. The moment the recovery policy intervenes, every update is biased toward the recovery policy’s behavior. Most safe-RL work leaves this mismatch unaddressed, updating as if all data came from the main policy (Srinivasan et al., 2020; Thananjeyan et al., 2021; Yang et al., 2022), or sidesteps it by shaping the reward to discourage unsafe states (Tessler et al., 2019; Stooke et al., 2020). The standard remedy, an importance-sampling correction (Degris et al., 2012), does not solve the problem either: a deterministic recovery (an MPC solver or a greedy SAC actor) produces a single action rather than a distribution, so the importance ratio has no density in its denominator and is undefined precisely where it is needed, the same obstruction that motivates deterministic policy gradients (Silver et al., 2014). The mechanism that makes real-world training feasible is thus the one that corrupts the learning signal, and a correct gradient must factor the recovery policy out rather than reweight through it.

Contributions. We close this gap with SafeExplorer, a practical algorithm built on a theoretical foundation for safe RL with a recovery policy. We make four contributions. *First*, we prove a policy-gradient theorem that is unbiased for the gradient of the mixed-policy return for any external recovery policy that takes over on a fixed subset of states (Theorem 1). The gradient never evaluates the recovery policy’s action density, so the gradient also covers deterministic recoveries. On final reward, the gradient estimator matches or outperforms truncated importance sampling (IS), which is itself biased for this gradient (Remark 1). Jump-Start RL (Uchendu et al., 2023) and state-triggered shielded RL follow as corollaries (Corollaries 8 and 9). *Second*, we bound the gap between the mixed-policy return we optimize and the main-policy return, a gap that shrinks as the safe region grows (Theorem 4). *Third*, when dynamics and recovery are deterministic, the value of a recovery-triggering state has a closed form that gives the critic accurate targets at the safe-region boundary (Proposition 6). *Fourth*, an outcome-gated compatibility regularizer imitates recovery actions only from successful recovery segments, and without the gate the regularizer reduces to Dataset Aggregation (DAgger)-style behavioral cloning (Ross et al., 2011) (Definition 7 and Proposition 10).

On three simulated locomotion environments, SafeExplorer cuts training falls over standard PPO by $21\times$ to roughly $255\times$ (10 seeds; Section 7.1), at matching or better final reward; on ANT, where the recovery controller often fails, SafeExplorer reaches the fixed success bar on every seed while no baseline of Table 1 reaches the bar on any seed, the unmasked ablation variant (Section 8) on 2 of 10.

2 Related Work

We study an agent paired with an external recovery policy that takes over outside a safe region. The setup causes training rollouts to follow a mixed policy that biases every on-policy update.

Off-policy RL Correcting that bias looks like a standard off-policy problem, but off-policy policy-gradient and evaluation methods (Degrís et al., 2012; Gu et al., 2017; Jiang & Li, 2016) rely on the importance ratio that Section 1 showed a deterministic recovery leaves undefined. Truncated-IS schemes such as V-trace (Espeholt et al., 2018) and Retrace (Munos et al., 2016) cannot help, because the singularity sits upstream of the truncation. Our Theorem 1 factors the recovery measure out instead of reweighting, so it applies uniformly to deterministic and stochastic recoveries, MPC controllers (Chiu et al., 2022) included.

Offline RL Offline RL (Jackson et al., 2025), including model-based MOREL (Kidambi et al., 2020) and MOPO (Yu et al., 2020), incurs zero additional environment falls by construction but presupposes a dataset whose collection incurred its own failures and whose coverage bounds the achievable policy; online training with a recovery policy is preferable when no adequate dataset exists or a recovery controller is cheaper than a covering dataset, offline in the complementary case. Our conditioning assumption, an already available recovery controller, is accounted for in Appendix D.4.

Mixed-policy and intervention data If reweighting is not the route, the alternative is to ask how prior work uses the same mixed-policy data, and rollouts in which a second policy intervenes are in fact common: they appear in DAgger (Ross et al., 2011; Kelly et al., 2019), in human-in-the-loop RL (Spencer et al., 2020; Saunders et al., 2018), and in Jump-Start RL (JSRL) (Uchendu et al., 2023). These either imitate the intervening policy or discard its transitions, so none extract a bias-corrected on-policy gradient from that data. Our masked gradient (Theorem 1) recovers that gradient directly, and our compatibility regularizer (Section 4.4) reduces in the JSRL setting to a state-dependent imitation loss with an outcome-based gate (Proposition 10).

Constraints and safety filters Delegating safety to an external recovery policy is itself the minority choice: the dominant lines of safe RL (García & Fernández, 2015) build safety into the policy rather than handing it to a separate controller. Constrained Markov decision processes (CMDPs) encode safety as constraints on long-run cost (Altman, 1999), optimized via primal-dual schemes such as constrained policy optimization (CPO) (Achiam et al., 2017), reward-constrained policy optimization (Tessler et al., 2019), and PID-Lagrangian variants (Stooke et al., 2020). Because they trade cost against return, they suit soft-constraint settings where some failures are tolerable, whereas we treat falls as something to minimize rather than to budget against return. Closer to a hard guarantee, control barrier function (CBF) approaches (Ames et al., 2019) and shields (Junges et al., 2016; Alshiekh et al., 2018; Dalal et al., 2018; Srinivasan et al., 2020; Hasanbeig et al., 2020; Kang et al., 2022) project unsafe actions at execution time; this is exactly the case of our mechanism in which the intervention is a single-step projection, so our setup subsumes it as one instance. Closely related are the state-wise constrained Markov decision process family (Zhao et al., 2023) and almost-surely safe RL (Sootla et al., 2022). Section 3 places our setting in the constraint-formulation taxonomy of Wachi et al. (2024).

RL with a recovery policy The work closest to ours commits fully to that delegation: it uses an explicit recovery policy as an alternative to CMDP costs and pointwise filters, facing the same mixed-policy bias extended from the single-step projections of our shielded-RL corollary to multi-step recovery segments. Eysenbach et al. (2018) learn a reset policy jointly with the task policy and abort to it when the reset policy’s value function predicts a state cannot be returned from; the recovery controller there is learned during training rather than fixed in advance, and no correction of the mixed-rollout bias is attempted. Thananjeyan et al. (2021) train a safety critic and switch to the recovery policy when constraint risk crosses a threshold; they deliberately relabel recovery transitions with the task policy’s proposed action, coherent for their off-policy Q-learning objective (Appendix B.8) but the source of the bias we correct once carried into an on-policy update. Yang et al. (2022) apply a similar template to legged locomotion with reward shaping, and Lee et al. (2019) train a model-free quadruped recovery controller, an example of the external recovery policy our masked gradient handles directly. Closest, Wagener et al. (2021) intervene on an advantage criterion and prove the intervened process safe, but still optimize the policy on the mixed rollouts without correcting the bias we address. Reverse-curriculum methods (Florensa et al., 2017) grow the region of starting states, and teacher-scheduled interventions (Turchetta et al., 2020) induce a safety curriculum, whereas safe-region curricula like ours grow the region the agent is permitted to enter. None of these treat the resulting distributional bias on the on-policy gradient, which our masked-gradient theorem (Theorem 1) addresses without requiring the recovery policy to have a density.

RL for legged locomotion End-to-end RL has advanced rapidly for legged locomotion (Ha et al., 2025; Lee et al., 2020; Peng et al., 2018; Tan et al., 2018; Peng et al., 2020; Rudin et al., 2021; Agarwal et al., 2022; Kumar et al., 2021; Haarnoja et al., 2019; Bogdanovic et al., 2022), and fine-tuning a policy in the real world has become common practice (Smith et al., 2023b; 2024; 2023a; Liu et al., 2024). These methods secure safety through reward design and sim-to-real robustness, but none confronts the bias a recovery policy injects into the on-policy gradient. That bias comes from the mixed rollout rather than from any property of legged robots, so Theorem 1 applies to any θ -independent, state-triggered intervention rule. The platform-independence claim attaches to the estimator, not to the system as instantiated, which needs a per-environment distance to the nominal state, safe-region size and pretrained recovery policy (Section 6).

3 Problem Setting and Notation

We study RL in the *hard-safety regime* of the constraint-formulation taxonomy of Wachi et al. (2024). On a real robot a fall can break hardware, damage property, or injure a bystander, so the goal is to keep falls as rare as possible rather than to budget them against return.

The strictest formalism related to this regime is the state-wise constrained Markov decision process family (Zhao et al., 2023), which demands constraint satisfaction at every step. We relax that requirement to *violation-minimization*, the model-free goal the survey identifies: without prior knowledge of the dynamics, hard state-wise safety cannot be guaranteed during training, so the target is to incur as few constraint violations (falls) as possible.

Among the ways to enforce safety in this regime (Section 2), ours is an external *multi-step* recovery policy that takes over whenever the state leaves a designer-specified safe region and returns the agent toward a nominal configuration, in ours the upright standing pose, unlike single-step action filters such as control-barrier projections and shields. We call this the *safe-region intervention* mechanism. A consolidated notation table is in Appendix B.3.

We instantiate the mechanism with a discounted Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, P, r, \gamma, T)$ with continuous state space $\mathcal{S}$ and action space $\mathcal{A}$, transition kernel P , reward $r \in [-r_{\max}, r_{\max}]$, discount $\gamma \in (0, 1)$, and episode horizon T . The analysis uses the infinite-horizon discounted convention (absorption at failure states); experiments truncate at $T = 1000$ steps.

Safe region and curriculum. The safe region is the sublevel set $\mathcal{R}(d) := \{s : \mathcal{D}(s) \leq d\}$ of a task-space distance $\mathcal{D}$ to a nominal configuration. The radius $d \in [0, d_{\max}]$ grows over training by a curriculum schedule $d_{\text{iter}}(\cdot)$, where $d_{\max}$ is a per-environment hyperparameter (Section 5).

Policies. Let $\pi_\theta(a | s)$ be the main policy, stochastic and θ -parameterized. The recovery policy $\pi^{\text{rec}}(a | s)$ is a state-conditioned policy that may be deterministic (MPC, greedy SAC) or stochastic. The recovery policy does not depend on θ , since the recovery policy’s parameters (SAC weights or an MPC cost) stay fixed during training, and we omit these parameters from the notation. The *mixed policy* π_θ^{mix} follows π_θ inside $\mathcal{R}(d)$ and π^{rec} outside it. We give its general form and an unbiased gradient in Section 4.1.

Objectives. The trajectory distributions under π_θ alone and under π_θ^{mix} are $p_\theta(\tau) = p(s_1) \prod_t \pi_\theta(a_t | s_t) P(s_{t+1} | s_t, a_t)$ and $p_\theta^{\text{mix}}(\tau) = p(s_1) \prod_t \pi_\theta^{\text{mix}}(a_t | s_t) P(s_{t+1} | s_t, a_t)$. The *main-policy return* is

$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)}[r(\tau)], \quad r(\tau) := \sum_t \gamma^t r_t. \quad (1)$$

The return J is not directly observable during training, since rollouts are drawn from $p_\theta^{\text{mix}}(\tau)$, not $p_\theta(\tau)$. The *mixed-policy return*, the quantity training rollouts actually realize, is

$$J^{\text{mix}}(\theta) = \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}(\tau)}[r(\tau)]. \quad (2)$$

Its state-conditional form, the *mixed-policy value* $V^{\pi_\theta^{\text{mix}}}(s) := \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}(\tau)}[r(\tau) | s_1 = s]$, is what PPO’s learned critic V_θ estimates. Discounted state-visitations under π_θ and π_θ^{mix} are ν^{π_θ} and $\nu^{\pi_\theta^{\text{mix}}}$, respectively. We treat

the recovery policy as remaining active at deployment (Section 9), so J^{mix} is itself deployment-relevant, not only a training surrogate for J .

Two rates. We use two rates, distinguished by their sampling distribution. The *recovery rate* $\alpha(\theta, d) := \Pr_{s \sim \nu^{\pi_\theta^{\text{mix}}}}[s \notin \mathcal{R}(d)]$ is sampled under the *mixed* distribution that training rollouts produce, hence *observable* as the fraction of rollout steps on which the recovery policy fired. The *out-of-region rate* $\beta(\theta, d) := \Pr_{s \sim \nu^{\pi_\theta}}[s \notin \mathcal{R}(d)]$ is sampled under the *main-only* distribution never deployed during training, hence *not directly observable*, yet it is the quantity that controls the gap in Theorem 4.

Together, these objects give the setup four operational features: binary set-membership safety, a possibly-deterministic recovery policy, multi-step recovery (a genuine mixed-policy distribution), and the unmodified main-policy objective, each addressed by one method component in Section 4 (Appendix B.4).

4 Method

SafeExplorer builds on PPO with four components, each removing a failure mode of recovery-using RL: (i) a *masked policy gradient* (Section 4.1) that removes the bias from treating recovery transitions as main-policy transitions; (ii) an *objective-gap bound* (Section 4.2) relating the training (mixed-policy) return to the deployment (main-policy) return and showing a safe-region curriculum closes the gap; (iii) an *analytic recovery value* (Section 4.3) that replaces the learned critic at recovery-triggering states with a closed form under deterministic dynamics; and (iv) an *outcome-gated compatibility regularizer* (Section 4.4) that imitates recovery only after successful segments. We develop each in turn.

4.1 Unbiased policy gradient via partition masking

To make the most of training data, we want every rollout to contribute to the task gradient, even those where the recovery policy took over for long stretches. The obstacle is that treating recovery transitions as if the main policy generated them biases the on-policy update, and the IS fix breaks down whenever the recovery policy is deterministic. π_θ , a Gaussian in our PPO implementation, has a differentiable log-density, whereas a deterministic π^{rec} is a Dirac measure with no density. Because this obstacle arises whenever an external, θ -independent policy takes over on a subset of states, not only in safety, we solve it in full generality first and then specialize to safe-region intervention.

Setup. We generalize the mixed policy of Section 3. Let $\mathcal{M} \subseteq \mathcal{S}$ be a designer-specified *main-policy set* (where π_θ acts), and let $\mu(\cdot | s)$ be any state-conditioned action measure that is *independent of θ* (it may be deterministic or stochastic). The resulting *partition policy* is

$$\pi_\theta^{\text{mix}}(a | s) = \begin{cases} \pi_\theta(a | s) & s \in \mathcal{M}, \\ \mu(a | s) & s \notin \mathcal{M}, \end{cases} \quad (3)$$

and the trajectory return $J^{\text{mix}}(\theta) = \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}}[r(\tau)]$.

Assumption 1 (Regularity). **(R1)** For every (s, a) on the support of p_θ^{mix} with $s \in \mathcal{M}$, $\pi_\theta(a | s) > 0$ and $\theta \mapsto \log \pi_\theta(a | s)$ is differentiable. **(R2)** $T < \infty$, or $|r_t| \leq r_{\max}$ uniformly. **(R3)** Differentiation and integration commute in Equation (4).

For diagonal-Gaussian π_θ with bounded mean and log-std, and bounded reward, (R1) to (R3) all hold.

Theorem 1 (Unbiased partition-policy gradient). *Under Assumption 1, the gradient of J^{mix} is*

$$\nabla_\theta J^{\text{mix}}(\theta) = \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}} \left[\left(\sum_{t=1}^T \mathbb{I}[s_t \in \mathcal{M}] \nabla_\theta \log \pi_\theta(a_t | s_t) \right) r(\tau) \right]. \quad (4)$$

Here $\mathbb{I}[\cdot]$ is the indicator function, so the sum runs only over timesteps with $s_t \in \mathcal{M}$.

We prove this in Appendix A.1: because μ is θ -independent, its factor in the trajectory density vanishes under differentiation, so the masked gradient never evaluates μ 's density and holds for any μ . This is exactly the

case the intra-option policy-gradient results (Sutton et al., 1999; Bacon et al., 2017) assume away; Theorem 1 extends the factorization to the density-free external controllers those results exclude, which is why we never need the IS correction.

Remark 1 (Masking versus IS). For stochastic μ the per-step IS estimator decomposes into the masked estimator plus a term carried by the weights π_θ/μ at recovery steps; that term has a generically nonzero mean under a genuine advantage, so per-step IS is in general biased for $\nabla_\theta J^{\text{mix}}$, whereas the masked estimator is exactly unbiased (Theorem 1; decomposition in Equation (17), Appendix E.16). The comparison below is thus between estimators that do not share a mean, with no dominance claimed among unbiased estimators of a common objective; the trained comparison uses truncated per-step IS (clipped at $c = 10$), the practical alternative, itself biased for $\nabla_\theta J^{\text{mix}}$ and undefined for a deterministic recovery (Appendix E.16). The weights also degenerate, since recovery acts unlike the main policy by design: the measured IS gradient variance exceeds the masked variance by up to 13 orders of magnitude (an empirical measurement, not a theorem), and truncated IS loses 40% reward on ANT, while masking matches or outperforms it on final reward in every regime we test. Masking, not IS, is the estimator SafeExplorer optimizes (Corollary 3).

Specializations. Different choices of $(\mathcal{M}, \mu)$ specialize Theorem 1 to concrete settings.

Corollary 2 (Safe-region intervention). *When the main-policy set is the safe region, $\mathcal{M} = \mathcal{R}$, and the action measure is the recovery policy, $\mu = \pi^{\text{rec}}$, the partition policy is the mixed policy π_θ^{mix} of Section 3. Theorem 1 then gives $\nabla_\theta J^{\text{mix}}(\theta) = \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}(\tau)}[(\sum_t \mathbb{1}[s_t \in \mathcal{R}] \nabla_\theta \log \pi_\theta(a_t | s_t)) r(\tau)]$.*

The same factorization yields Jump-Start RL (Uchendu et al., 2023) and state-triggered shielded RL as corollaries (Corollaries 8 and 9, Appendix A.1). The rest of the paper develops safe-region intervention (Corollary 2).

Corollary 3 (PPO surrogate with safe-step masking). *Specializing Corollary 2 to the clipped PPO surrogate, the empirical masked-gradient estimator is*

$$\widehat{\nabla J^{\text{mix}}}(\theta) = \frac{1}{|T_{\mathcal{R}}|} \sum_{t: s_t \in \mathcal{R}} \nabla_\theta \log \pi_\theta(a_t | s_t) \hat{A}_t^{\text{full}}, \quad (5)$$

where $\hat{A}_t^{\text{full}}$ is the generalized advantage estimation (GAE) (Schulman et al., 2016) advantage over the full mixed-policy trajectory (recovery segments enter through the return exactly as in Theorem 1, while the mask restricts only the policy gradient term) and $|T_{\mathcal{R}}| = |\{t : s_t \in \mathcal{R}\}|$ is the number of timesteps inside the safe region. The clipped PPO surrogate is

$$L_{\text{safe}}^{\text{PPO}}(\theta) = -\frac{1}{|T_{\mathcal{R}}|} \sum_{t: s_t \in \mathcal{R}} \min\left(\rho_t(\theta) \hat{A}_t^{\text{full}}, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon) \hat{A}_t^{\text{full}}\right), \quad (6)$$

where $\rho_t(\theta) = \pi_\theta(a_t | s_t)/\pi_{\theta_{\text{old}}}(a_t | s_t)$ is the probability ratio of the main policy, well defined because both terms are the main policy.

Relative to the standard clipped-PPO surrogate over the same rollouts, Equation (5) changes only two things: the sums run over safe steps, and the normalizer is $|T_{\mathcal{R}}|$, the number of such steps in a rollout and hence random, which the curriculum on d keeps large in practice (Section 5, Appendix B.6). The masked score-function term is exactly unbiased for $\nabla_\theta J^{\text{mix}}$ (Theorem 1). GAE, the clipping, the multi-epoch reuse of each rollout, and thus PPO’s stability heuristics are otherwise unchanged: the mask and this normalizer are the only estimator-level departures from standard PPO, and the approximations that remain are inherited by every PPO-based method we compare against.

Because the normalizer $|T_{\mathcal{R}}|$ is random, Equation (5) is a self-normalized ratio estimator, consistent for a positive multiple of $\nabla_\theta J^{\text{mix}}$, with $O(1/B)$ bias in the batch size B , zero once recovery stops firing (Proposition 12, Appendix B.6).

4.2 Bounding the gap between mixed-policy and main-policy returns

Theorem 1’s gradient targets the *mixed-policy return* $J^{\text{mix}}(\theta)$, the quantity training rollouts realize, but the deployment-relevant quantity is the *main-policy return* $J(\theta)$, what π_θ earns alone, which training never

samples directly. Optimizing one while caring about the other is only safe if the two cannot drift far apart, so this section bounds $|J^{\text{mix}} - J|$ and shows it is exactly zero under reading (R-A) below.

The two returns differ only on the steps where π_θ acts but π_θ^{mix} would have called the recovery policy, that is, the steps where π_θ alone leaves the safe region. The gap is therefore controlled by how reliably π_θ keeps itself inside $\mathcal{R}$, which we make precise with a one-step invariance property.

Assumption 2 (Approximate π_θ -invariance of $\mathcal{R}$). There exists $\eta \geq 0$ such that, starting from any reachable $s \in \mathcal{R}$, π_θ stays in $\mathcal{R}$ at the next step with probability at least $1 - \eta$ uniformly over time.

This one-step set-invariance property holds with $\eta = 0$ exactly in the idealized limit (R-A) below and approximately in our setting (R-B).

Summed over a discounted trajectory, this one-step slack η controls the out-of-region rate $\beta(\theta, d)$ (Section 3), which inherits a bound $\beta(\theta, d) \leq \eta\gamma/(1 - \gamma) \leq \eta/(1 - \gamma)$ whenever episodes start inside the region, as ours do (Appendix A.2).

Theorem 4 (Objective-gap bound). *Under Assumption 1,*

$$|J(\theta) - J^{\text{mix}}(\theta)| \leq \frac{2r_{\max}}{(1 - \gamma)^2} \beta(\theta, d). \quad (7)$$

The proof (Appendix A.2) applies the Performance Difference Lemma (Kakade & Langford, 2002). The advantage of π_θ against π_θ^{mix} vanishes inside $\mathcal{R}$, where the two policies agree, and is bounded by $2r_{\max}/(1 - \gamma)$ outside it, so the gap scales with the main-policy out-of-region rate $\beta(\theta, d)$. The bound is non-vacuous only for $\beta(\theta, d) < 1 - \gamma$ ($\eta_* < (1 - \gamma)^2$ in Corollary 5); we read it structurally (Appendix E.15).

How far Equation (7) can be pushed as the curriculum radius d_{iter} grows to $d_{\max}$ depends on the largest safe region it reaches, $\mathcal{R}(d_{\max})$, which admits two readings.

Corollary 5 (Conditional fixed-point). *(Exact, R-A.) If $\mathcal{R}(d_{\max})$ covers the reachable state space, then at $d_{\text{iter}} = d_{\max}$, $\pi_\theta \equiv \pi_\theta^{\text{mix}}$ on the reachable support of $p_\theta(\tau)$, $\beta(\theta, d_{\max}) = 0$, and $J^{\text{mix}}(\theta) = J(\theta)$ exactly. (Approximate, R-B.) Otherwise $|J(\theta) - J^{\text{mix}}(\theta)| \leq 2r_{\max}\eta_*/(1 - \gamma)^3$, where $\eta_* = \eta(\theta, d_{\max}) \geq 0$ is the env-dependent invariance slack that (R-B) does not force to zero.*

The proof specializes Theorem 4 to the two readings of $\mathcal{R}(d_{\max})$ (Appendix A.3). Our setting is (R-B): $\mathcal{R}(d_{\max})$ is a hand-set hyperparameter strictly inside the reachable set, so the bound is approximate, the theory no longer forcing $\eta = 0$, though the curriculum can still drive it low in practice. Theorem 4 then says any reduction in $\beta(\theta, d)$ tightens the gap proportionally, which the linear schedule $d_{\text{iter}}: d_0 \rightarrow d_{\max}$ achieves, with the observable $\alpha(\theta, d)$ as its diagnostic (Appendix E.15). Theorem 4 does not bound falls, and the falls still possible under (R-B) are what the compatibility regularizer (Section 4.4) targets. Appendix A.7 sketches why $\beta(\theta, d)$ is hard to bound from *observable* quantities in continuous-action settings.

4.3 Analytic recovery value

The masked gradient fixes the policy gradient, but it leaves a second error untouched: the critic. A learned critic trained over the recovery segments bootstraps its value targets *through* them, so even with a corrected gradient the value-target error persists; empirically, the learned- V variant under-performs every other variant on HALF-CHEETAH. Replacing the learned critic over recovery segments with the analytic value derived below removes this error; Section 8 and Appendix E.5 give the per-component attribution, train-side and recovery-disabled.

When dynamics and recovery are both deterministic, the practically common case (MPC, greedy SAC), the value at a recovery-triggering state has a closed form: the recovery segment unrolls along a single fixed path, so its contribution is the return along that path rather than something the critic must learn.

Proposition 6 (Analytic recovery value). *Suppose dynamics P and recovery π^{rec} are both deterministic. Let s_t be a recovery-triggering state, $k \geq 1$ the number of recovery steps until either re-entry into $\mathcal{R}$ at $t + k$ (success) or episode termination at $t + k - 1$, the segment's last step (failure), and $G_{t,k} = \sum_{j=0}^{k-1} \gamma^j r_{t+j}$ the*

realized segment return. Then

$$V^{\pi_\theta^{\text{mix}}}(s_t) = \begin{cases} G_{t,k} + \gamma^k V^{\pi_\theta^{\text{mix}}}(s_{t+k}) & \text{success,} \\ G_{t,k} & \text{failure.} \end{cases} \quad (8)$$

Under our convention that recovery accrues zero per-step task reward and failure incurs a one-time terminal failure reward r_{term} , the segment return $G_{t,k}$ is 0 on success and $\gamma^{k-1}r_{\text{term}}$ on failure. In practice we bootstrap the re-entry value with the learned critic V_θ , writing $\gamma^k V_\theta(s_{t+k})$ into the buffer on success (Algorithm 1). This carries no Monte-Carlo variance under deterministic P and π^{rec} .

The proof is direct (Appendix A.4): under deterministic dynamics and recovery, the post-trigger trajectory is a single deterministic path of length k , so its return is the realized return.¹

We write Equation (8) into the rollout buffer at *every* step of each recovery segment (intermediate recovery rewards stored as zero), so intra-segment TD residuals vanish and λ -GAE telescopes through the segment. Lemma 11 (Appendix B.2) derives the target this induces: under deterministic dynamics it is exact up to the critic residual at re-entry, and under stochastic dynamics it becomes a one-sample plug-in (Appendix A.5).

4.4 The compatibility regularizer

The masked gradient removes the specific bias that recovery attribution introduces into the on-policy update (Theorem 1) and the analytic value supplies correct targets over recovery segments, but removing that bias alone does not make the agent self-sufficient: neither component pulls π_θ toward behaving like recovery at unsafe states. This matters precisely when recovery is unreliable (Section 7.2): an update that is unbiased for J^{mix} over repeated failed recoveries teaches the policy nothing about avoiding those states unaided. We close this gap with a lightweight imitation-from-recovery loss with an outcome-conditioned weight. As an imitation signal rather than an importance correction, it needs no recovery density and covers deterministic controllers. The closest methods, DAgger (Ross et al., 2011) and JSRL (Uchendu et al., 2023), do not gate the imitation signal by segment outcome, and self-imitation learning (Oh et al., 2018), which does, imitates the agent’s own past actions rather than an external recovery policy, the distinction we now make precise.

Definition 7 (Compatibility regularizer). Let $a_t^{\text{rec}} = \pi^{\text{rec}}(s_t)$ be the recovery action executed at unsafe step t . The trajectory-level compatibility score is

$$C_\theta(\tau) = \prod_{t:s_t \notin \mathcal{R}} \pi_\theta(a_t^{\text{rec}} | s_t), \quad \log C_\theta(\tau) = \sum_{t:s_t \notin \mathcal{R}} \log \pi_\theta(a_t^{\text{rec}} | s_t). \quad (9)$$

A high $C_\theta(\tau)$ measures exactly the agreement we want, π_θ assigning high likelihood to the recovery actions at the states where it intervened. Raising it indiscriminately is harmful: a failed recovery (segment ends in termination) is exactly what π_θ should *not* imitate. We therefore gate the imitation signal by the realized outcome of each segment. Let $\sigma_k \in \{0, 1\}$ indicate success of recovery segment k (re-entry into $\mathcal{R}$), and $\text{seg}(t)$ the segment containing step t . Reading success off the realized rollout keeps the gate applicable whether recovery is deterministic or stochastic. The *hard outcome gate* is

$$w_t^{\text{hard}} = \sigma_{\text{seg}(t)}, \quad t : s_t \notin \mathcal{R}. \quad (10)$$

Two boundary cases follow the same reading: a segment cut off by rollout truncation counts as a tentative success with a bootstrapped value, and an immediate re-exit opens a new segment. A signed soft variant based on per-step value change is ablated in Appendix B.5; the hard gate dominates empirically, so $w_t = w_t^{\text{hard}}$ throughout unless noted.

Putting the score and its gate together, the regularizer is the outcome-gated form of $-\log C_\theta(\tau)$ from Equation (9), added to the PPO loss with coefficient λ_{compat} :

$$L^{\text{compat}}(\theta) = -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_{t:s_t \notin \mathcal{R}} w_t \log \pi_\theta(a_t^{\text{rec}} | s_t), \quad L(\theta) = L_{\text{safe}}^{\text{PPO}} + c_v L^V + L^{\text{compat}}. \quad (11)$$

¹Theorems 1 and 4 hold for any bounded reward, covering this training-signal convention; the induced train-to-deploy return gap is controlled by the same $\beta(\theta, d)$ (Appendix A). Reported returns always use the unmodified task reward (Section 6).

Here N_{rec} counts the recovery-controlled minibatch steps (Appendix B.3; per-segment normalization ablated in Appendix E.11), and L^V is the standard PPO critic loss with coefficient c_v (entropy coefficient $c_e = 0$ by default; Appendix B).

L^{compat} is a supervised auxiliary term, deliberately outside the unbiased J^{mix} gradient: it trades a tunable amount of bias, scaled by λ_{compat} , for a signal the unbiased components cannot supply, pulling π_θ toward the actions of *successful* recovery segments. That trade pays off exactly where it should: the term costs a little reward on HALFCHEETAH but adds substantially on GO1 and ANT (+1036 and +2524 reward over the variant without the regularizer, Section 8).

The gate is all that separates this term from standard imitation: always on ($\sigma_k \equiv 1$), L^{compat} reduces to DAgger-style (Ross et al., 2011) cloning on the recovery-controlled states, π^{rec} as teacher (Proposition 10, Appendix A.6); the gate echoes the Q-filter of Nair et al. (2018), reading a realized segment outcome instead of a learned value estimate.

5 Algorithm

SafeExplorer assembles the three algorithmic components developed in the previous sections, the masked policy gradient (Theorem 1, Corollary 3), the analytic recovery value (Proposition 6), and the hard outcome-gated compatibility regularizer (Section 4.4), into a single PPO update (Algorithm 1); the fourth component, the objective-gap bound (Theorem 4), needs no implementation beyond the curriculum below. Because each component intervenes only where the recovery policy acts, the resulting algorithm is a small edit to a standard PPO implementation: the changes touch only the rollout’s recovery branch and the loss, everything else is unchanged. That locality keeps the method cheap: the added cost is one recovery-policy forward pass per triggered step plus segment bookkeeping (runtimes in Appendix C).

This leaves one moving part, the safe-region radius that decides when recovery is triggered, which we anneal over training with a linear curriculum (Bengio et al., 2009; Narvekar et al., 2020): $d_{\text{iter}}(u) = d_0 + \frac{u-1}{N} d_{\text{max}}$, growing the region from a tight d_0 to the per-environment d_{max} over the N updates (d_0 , d_{max} , N in Appendix C). Appendix E.9 ablates logarithmic, step, and constant schedules on ANT, our least reliable recovery policy, where the linear schedule attains the highest reward. This annealing is the practical mechanism by which the recovery rate $\alpha(\theta, d)$ falls over training (Appendix E.15), an empirical diagnostic for the gap-tightening predicted by Theorem 4.

6 Experimental Setup

Our experiments evaluate three claims. The first is the *safety-reward trade-off* of SafeExplorer against standard PPO, recovery-using baselines, and CMDP baselines, judged on both falls and reward. The second is the *unreliable-recovery regime* that most motivates the method, where the recovery policy itself often fails and the trade-off is hardest to win. The third asks *which component* drives the gains: the unbiased masked gradient (Theorem 1), the analytic recovery value (Proposition 6), or the outcome-gated regularizer (Definition 7). A single metric adjudicates all three: the number of training falls a method incurs before first reaching task success, where success is reaching a reward threshold, reported across a grid of thresholds rather than at a single one (defined under *Evaluation*). Table 1 therefore reports the underlying rewards and fall counts separately. All safety results are simulated and conditional on an already available recovery controller, with the end-to-end accounting in Appendix D.4; the instantiation additionally requires a per-environment distance $\mathcal{D}$, a per-environment d_{max} , and a separately pretrained recovery policy (Appendices C and D).

Environments. We evaluate on three continuous-control locomotion environments: HALFCHEETAH and ANT (custom MuJoCo (Todorov et al., 2012)/Gymnasium (Towers et al., 2023) variants with a configurable safe-region indicator; full specification in Appendix D), and Go1, a Unitree Go1 quadruped built on the mujoco_menagerie model (Zakka et al., 2022) with a velocity-tracking reward adapted from legged_gym (Rudin et al., 2021). The mechanism is identical across all three: the distance $\mathcal{D}$ defining $\mathcal{R}(d)$ is a (z, tilt) task-space distance to the nominal upright pose (exact formulas in Appendix D.2), and each environment

keeps its standard locomotion reward, zeroed only during recovery (Section 4.3). The environments differ in how hard they make safety: **HALFCHEETAH** (planar, strong recovery) sets the baseline, **ANT** (3D, weaker recovery) tests whether the compatibility regularizer compensates for an unreliable fall-back, and **GO1** (12-degree-of-freedom (DoF) quadruped) scales the claim to a higher-dimensional, multi-contact platform.

Recovery policies. Each environment uses a separately pre-trained SAC (Haarnoja et al., 2018) actor as π^{rec} : a stand-and-stabilize objective on its own environment, task terms disabled, no demonstrations, aggressively randomized resets (recipes in Appendix D.3). At rollout time it takes the SAC mean action, deterministic, so it exercises the no-density case the masked gradient is built for (Theorem 1). The **ANT** recovery policy is meaningfully weaker than the others, the most informative axis of variation: its strong control-magnitude penalty caps how aggressively it can correct, and the harder 3D stabilization needs those corrections (Appendix D).

The remaining choice is the radius d_{max} , the one quantity a skeptic might suspect was tuned. Unbiasedness holds for any d_{max} (Theorem 1), so no radius can manufacture statistical validity, though it still shapes results through the gap bound (Theorem 4) and the curriculum. We set each d_{max} (Appendix C) by a qualitative criterion, that the curriculum lowers the observable intervention rate (Appendix E.15), without a quantitative search. The fixed- d sweep (Table 12, Appendix E.8) shows sensitivity to a *constant* radius that the curriculum mostly recovers (Appendix E.9); no systematic sweep varies d_{max} under the curriculum, a limitation (Appendix E). Every reported reduction is therefore tied to the trigger radius it was run at, and every table whose runs use a radius other than the defaults of Appendix C prints that radius; the radius-dependence case study is in Appendix E.18.

Variants compared. SafeExplorer (Algorithm 1) is compared against three external baselines spanning the established alternatives identified in Section 2. Standard PPO is the no-recovery extreme. Recovery RL (action-relabeling port) (Thananjeyan et al., 2021) and Safe Legged (reward-shaping port) (Yang et al., 2022) are *on-policy ports* of the two published recovery-using alternatives: both run inside the same PPO loop, sharing SafeExplorer’s safe-region trigger and frozen SAC recovery policy, and each isolates its published method’s data-handling rule at recovery steps, action relabeling for Recovery RL versus relabeling plus a fixed reward penalty of 1 for Safe Legged (exact rules and deviations in Appendix B.8). The comparison set is deliberately on-policy under this shared protocol, so the data-handling rule at recovery steps is the only varying factor; a native off-policy entrant would vary optimizer, replay, and network capacity at once, which is why Recovery RL enters as a port rather than in its off-policy form. A controlled ablation isolating each component of SafeExplorer is reported in Section 8; additional soft-gate variants are in Appendix E.6.

CMDP baselines. To position SafeExplorer against the standard no-recovery alternative, we compare to two CMDP solvers, CPO (Achiam et al., 2017) and PPO-Lagrangian (Ray et al., 2019; Stooke et al., 2020), which act with a single policy and encode safety as a constraint on an indicator cost (1 per fall). We run both via OmniSafe (Ji et al., 2024) at settings matched to SafeExplorer, except the solvers’ default observation and reward normalization, which SafeExplorer does not use, favoring the baselines (Appendix E.17). Section 7.3 analyzes their constraint feasibility.

Training. We train each variant for 8M environment steps on **HALFCHEETAH** and **ANT** and 16M on **GO1** (its higher-dimensional action space needs a longer horizon); all main results use ten seeds, and each appendix cell prints its own count. Final reward is the per-episode mean of the last 10% of training, computed from the unmodified task reward, not the recovery-zeroed learning signal; falls are the cumulative count over the full training run, a fall being a per-environment unhealthy termination (Table 7), never a time-limit truncation. We call this pair the *train-side* numbers, to distinguish them from the recovery-disabled evaluation protocol, which freezes the final policy and rolls it out with the recovery branch switched off (Section 7.2, Appendix E.14). Shared PPO hyperparameters follow the CleanRL reference values (Huang et al., 2022) and are listed in Table 4.

Evaluation. Tables report mean $\pm$ sample standard deviation over seeds; plots aggregate by interquartile mean (IQM) with 95% stratified-bootstrap confidence intervals (Agarwal et al., 2021); all cells use every seed in their batch, no post-hoc exclusions. We define *falls-to-success* as the number of training falls before the agent first reaches a reward threshold. Specifically, falls-to-success is the cumulative fall count up to the first point where a trailing mean of the reward curve crosses the threshold and stays at or above it

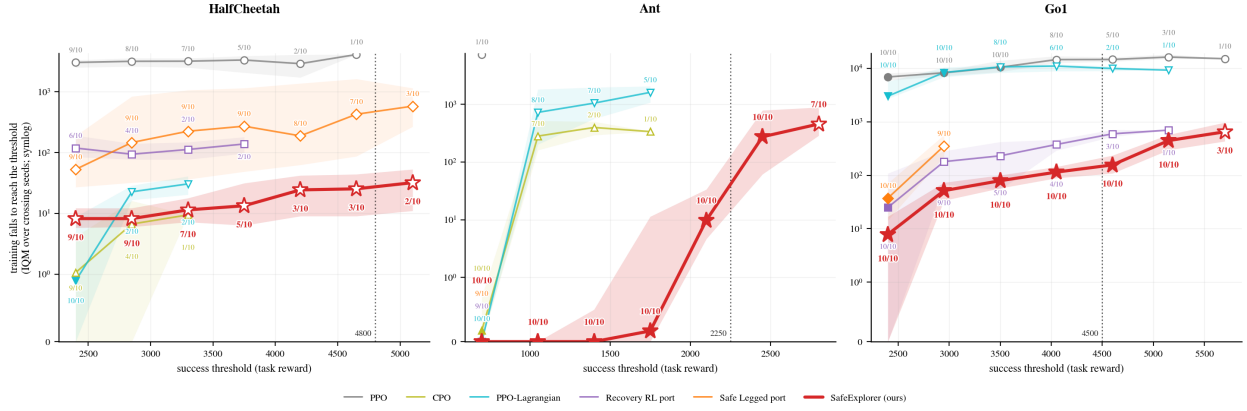


Figure 1: **Falls-to-success across the threshold range.** Cumulative training falls at the first persistent crossing of each success threshold, aggregated as an interquartile mean over the k of n seeds that reach it (the count is printed at each marker, which is hollow when $k < n$; a missing marker means no seed reaches it), with 95% percentile-bootstrap bands over seeds; symlog vertical axis. The dotted line is the fixed threshold of Section 6. Lower is cheaper, further right is more capable.

for the full window. Seeds that never reach the threshold within the training budget are not excluded but right-censored; each such seed counts as unsolved in the solve fraction k/n , the k of n seeds that reach the threshold, and enters a censored all-seeds lower bound. Over the seeds that succeed, falls-to-success is aggregated by IQM with a percentile bootstrap over seeds. We report falls-to-success at seven thresholds per environment, spread around a fixed success bar of 4800 reward on HALFCHEETAH, 2250 on ANT, and 4500 on Go1, at which the tabulated numbers are read (Figure 1). The window size, the interpolation rule, and the remaining fine print are in Appendix E.2.

7 Main Results

7.1 The safety-reward trade-off

Addressing our first claim, we compare all methods on the three wide-band environments: Figure 1 reports falls-to-success across a grid of success thresholds, with the per-method solve counts k/n , Table 1 the separate final-reward and total-training-fall figures (different quantities, since total falls span the whole run, not just the climb to success), and Figure 2 (Appendix E) the per-metric reliable interquartile-mean view. Under the fixed-bar metric of Section 6, SafeExplorer solves 10/10 on Go1 and reaches the ANT bar on all ten seeds, where no baseline of Table 1 reaches it on any seed and the unmasked ablation variant on 2 of 10; on HALFCHEETAH the bar sits above the mean reward of every method in Table 1 and solve rates are variance-driven, so Table 1 instead shows SafeExplorer at 3855 ± 958 reward against the constrained solvers’ 2768 and 2743, at fall counts in the same range (15.3 against 17 and 9). In the lifetime view (Table 1) it cuts total training falls over PPO by $21\times$ to roughly $255\times$ at matching or better reward; Appendix E.2 carries the ratios with intervals, the solve counts, and the every-seed comparisons.

Across the profile, no method reaches a higher solve rate at a lower cost than SafeExplorer at any plotted threshold on ANT or Go1, nor on HALFCHEETAH from 2850 upward. At the lowest plotted HALFCHEETAH threshold (2400, under $0.6\times$ the best mean reward in Table 1), PPO-Lagrangian reaches 10/10 at one fall and CPO 9/10 at about one, against SafeExplorer’s 9/10 at eight.

Reading the environments one at a time, the same ranking holds but the reason shifts with how hard safety is. On HALFCHEETAH, where the recovery policy is reliable, PPO is competitive on reward but pays for it in falls, while SafeExplorer matches that reward at a fraction of the falls. On Go1, the larger action space and longer horizon make PPO highly unsafe, while the Recovery RL and Safe Legged ports control falls only at substantial reward cost and SafeExplorer attains both the highest reward and the lowest fall count. ANT

Table 1: Final reward (same unmodified task reward for all methods) and total training falls, mean $\pm$ sample standard deviation over 10 seeds at full budget: the body batch for recovery-using rows, the matched-budget OmniSafe batch (cost 1 per fall, limit 0.05) for CMDP rows; recovery-using rewards are training-time mixed-policy returns (recovery-disabled evaluation in Table 17). Batch provenance and caveats in Appendix E.2.

Method	HALFCHEETAH		ANT		Go1	
	Reward $\uparrow$	Falls $\downarrow$	Reward $\uparrow$	Falls $\downarrow$	Reward $\uparrow$	Falls $\downarrow$
PPO (no recovery)	3462 $\pm$ 924	3899 $\pm$ 1573	259 $\pm$ 100	37600 $\pm$ 11202	3915 $\pm$ 603	17458 $\pm$ 1950
Recovery RL (action-relabeling port)	2209 $\pm$ 1404	1298 $\pm$ 2492	76 $\pm$ 14	51987 $\pm$ 7168	3332 $\pm$ 949	1753 $\pm$ 807
Safe Legged (reward-shaping port)	4117 $\pm$ 964	1135 $\pm$ 987	70 $\pm$ 14	55202 $\pm$ 9929	1838 $\pm$ 155	5410 $\pm$ 1228
CPO	2768 $\pm$ 343	17 $\pm$ 24	1153 $\pm$ 256	838 $\pm$ 152	364 $\pm$ 51	37364 $\pm$ 4787
PPO-Lagrangian	2743 $\pm$ 355	9 $\pm$ 16	1530 $\pm$ 368	3586 $\pm$ 2515	4018 $\pm$ 737	12859 $\pm$ 1649
SafeExplorer (ours)	3855 $\pm$ 958	15.3 $\pm$ 14.7	2698 $\pm$ 237	866 $\pm$ 718	5509 $\pm$ 155	830 $\pm$ 216

is the informative exception: the ranking still favors SafeExplorer, but it behaves differently because the recovery policy itself is unreliable, the case we isolate next in Section 7.2.

This dependence on recovery is exactly what Theorem 4 predicts: Appendix E.15 confirms on body-matched 10-seed runs that the gap $J^{\text{mix}}(\theta_k) - J(\theta_k)$ tracks the recovery rate $\alpha(\theta, d)$, closing to seed noise on HALFCHEETAH and ANT, where the policy becomes self-stable, and settling at a proportional residual on Go1, where it does not. Per-environment learning curves are deferred to Figure 4 in Appendix E.

7.2 The unreliable-recovery regime: Ant

This subsection addresses our second claim, the unreliable-recovery regime: Section 7.1 cannot show *why* SafeExplorer remains effective where the recovery policy itself fails, so we isolate the one environment where that happens, ANT, whose SAC recovery is markedly less reliable. Replaying the recovery from the states a trained policy actually drives it into, it completes the segment only 70 to 91% of the time on ANT, against 97 to 100% on HALFCHEETAH and Go1 (one representative run per environment; Table 8). On ANT, PPO and the Recovery RL and Safe Legged ports all fail (38k to 55k falls, reward ≤ 259 , 10 seeds). The mechanism is a feedback cycle, because when the recovery policy is unreliable, any method that relies on it *reactively* keeps re-entering unsafe states, recovery keeps failing, and the agent never learns to avoid them unaided. SafeExplorer breaks that cycle and reaches the fixed 2250 bar on all ten evaluated seeds, at an order of magnitude fewer falls than any recovery-using baseline; no baseline of Table 1 reaches that bar on any seed. The ablations pinpoint why: every variant that drops the hard outcome gate reaches the bar on at most 2 of 10 seeds (Section 8); the gate, not the masked gradient or the analytic value, supplies the fallback skill when recovery is unreliable.

Does the policy stand on its own? Evaluated with the recovery branch disabled, SafeExplorer on ANT retains 3124 $\pm$ 358 reward at 90 $\pm$ 72 evaluation falls, against the next-best variant’s 1215 $\pm$ 406 at 460 $\pm$ 189, PPO’s 299 $\pm$ 95 at 939 $\pm$ 23, and the ports’ 83 $\pm$ 69 and 70 $\pm$ 36. On HALFCHEETAH the leading variants’ recovery-disabled rewards are within seed noise of one another; on Go1 SafeExplorer retains the highest recovery-disabled reward in both batches while remaining below its mixed-policy reward, so it is not fully self-sufficient there. Recovery-disabled reward and falls for the remaining methods and environments are in Table 17.

7.3 Comparison to CMDP baselines

This subsection completes our first claim by comparing SafeExplorer against the CMDP alternative to a recovery policy, which casts safety as a constraint and solves the resulting CMDP rather than intervening on the trajectory. CPO and PPO-Lagrangian, the trust-region and Lagrangian solvers, appear in Table 1 and Figure 3 (Appendix E) with a cost of 1 per fall, at the matched settings of Section 6. SafeExplorer attains the highest reward of the three in every environment. On falls, SafeExplorer has the fewest on Go1; on ANT its count is within seed noise of CPO’s, so the claim there is the 2.3 $\times$ reward lead rather than a falls lead; on HALFCHEETAH all three are low, within seed noise (Table 1, Appendix E.17). The more revealing comparison is constraint feasibility: whether each solver keeps the falls it promised to.

The constrained solvers’ poor safety on the hard environments is not under-tuning. Each CMDP solver is configured with a cost limit on falls, as its formulation requires, and meets it on HALFCHEETAH where staying safe is easy, but *violates its own limit* on ANT and GO1 by up to 19 $\times$: CPO converges to an episodic cost of 0.96 on GO1 against the 0.05 budget it was set, and even incurs more GO1 falls than unconstrained PPO at a tenth of the reward (Appendix E.17). Constraint satisfaction thus holds where safety is easy and fails exactly where the safe-region intervention is needed. The difference is one of mechanism: the CMDP baselines act alone and only penalize the violation after the fact, whereas SafeExplorer acts on the safe region directly, which is why it keeps falls low where the constrained solvers cannot.

8 Ablation Analysis

This section addresses our third claim, which component drives SafeExplorer’s gains over prior recovery-using methods. We isolate the three components that change the training update with a ladder of controlled variants. All variants share the same recovery policy and curriculum and differ only in how recovery transitions are handled. The ladder starts from the biased update those prior methods use, which keeps recovery transitions as if the main policy had generated them (a learned-critic on-policy gradient), and turns on one component at a time. *Unmasked PG* is that biased baseline. The *masked, learned V* variant switches to the unbiased masked policy gradient (Theorem 1), the *masked, analytic V* variant adds the analytic recovery value (Proposition 6), and SafeExplorer adds the hard outcome gate (Equation (10)). PPO (no recovery) anchors the no-intervention end. Figure 3 (Appendix E) ranks these variants by falls-to-success and Figure 5 (Appendix E.4) gives the complementary reward-vs-falls view. Masking removes this bias at the cost of dropping every recovery step from the gradient; the analytic value and the outcome gate are what pay that cost back, so its payoff appears in combination with them rather than in isolation (Appendix E.5).

The first component, the masked policy gradient (Theorem 1), is the foundation. Its biased alternative, Unmasked PG, reaches competitive or higher reward on HALFCHEETAH and GO1, but pays 11 $\times$, 5.2 $\times$, and 3.4 $\times$ more falls than SafeExplorer on HALFCHEETAH, ANT, and GO1 (Table 10), and is ill-posed for deterministic recovery. Masking removes that bias, the prerequisite for the next two components.

Removing the bias exposes a second problem: the unbiased gradient is sparse at the safe-region boundary, so the masked, learned-V variant alone regresses to 1982 reward on HALFCHEETAH, the lowest of any variant (Table 10). The analytic recovery value (Proposition 6) supplies dense, correct targets exactly there, lifting HALFCHEETAH reward to 3870, the largest single-component gain on that environment (+1888) on the train-side 5-seed ladder; the same configuration’s 10-seed cell reads 2608 ± 1090 (Table 9), so the size of the gain depends on the seed batch. Under recovery-disabled evaluation that gain does not transfer. On HALFCHEETAH and ANT the analytic value is within seed noise both with and without the gate, and the outcome-gated imitation does most of the work there; on GO1 the attribution runs the other way, with the analytic value the largest single-component contribution at the body operating point. The components interact rather than add, and which one carries an environment changes with the environment (Appendix E.5, Table 17).

The top variant adds the hard outcome gate (SafeExplorer), which leads the ladder on safety in all three environments (on GO1 nominally, inside seed noise), and supplies the largest single-component reward gains on GO1 and ANT (+1036 and +2524; Table 10), where the analytic value alone leaves reward flat; it is the only ladder variant to reach the fixed ANT success bar on every seed, the strongest evidence for the imitation reading of $C_\theta(\tau)$ (Section 4.4).

In the λ_{compat} sweep (Table 11; HALFCHEETAH and ANT only, the default transferred to GO1 untested), HALFCHEETAH reward stays at 4770 to 5886 across $[10^{-3}, 1]$, peaking at 10^{-2} , but falls are minimized at the 10^{-3} default (33) and rise 4.3 $\times$ within a decade (142 at 10^{-2}); we choose the default on falls, not reward. Appendix E reports the rest: robustness to hyperparameters and noise (Appendices E.7 to E.9, E.11 and E.12), the soft-gate variants (Appendix E.6), compute-scaling (Appendix E.10), the full ladder (Appendix E.4), and a failure-mode catalog (Appendix E.13).

9 Discussion and Limitations

When the method helps, and when it does not. SafeExplorer’s reward advantage is smallest when PPO is already safe and recovery is reliable, and largest in the opposite regimes: weak recovery (ANT), dangerous unaided PPO (GO1). The hard outcome gate makes this work: a soft gate imitates failed recovery actions and degrades both metrics (Appendix E.6). We treat the recovery policy as a permanent fallback, not temporary scaffolding: on GO1 it stays mildly engaged ($\alpha(\theta, d) \approx 0.05$) even at 16M steps, and keeping it available is what bounds falls there (Appendix E.15); our claim is reduced training falls, not their elimination. Appendix D.4 gives the end-to-end accounting, with recovery pretraining counted in full; charged to a single training run the reductions read $21.3\times$ (HALFCHEETAH), $2.0\times$ (ANT), and $2.8\times$ (GO1), rising with checkpoint reuse. Claims are scoped to the simulated environments we run, and reducing falls in simulation is not a real-world safety guarantee (Appendix F). Where training falls cannot be tolerated at all, offline RL is the competing paradigm (Section 2). We keep the imperfect ANT recovery deliberately; an imperfect recovery is where the outcome gate earns its keep, but a bad enough one is worse than none, sub-threshold WALKER2D controllers producing roughly $3\times$ PPO’s falls while destroying task learning. Rather than retraining a weak recovery controller, we recommend screening it by its offline catch rate before any task training is spent (Appendix D.3, Appendix E.18).

Assumptions behind the guarantees. The analytic value (Proposition 6) is exact only under deterministic dynamics and recovery. Under noise it becomes a single-sample estimate, unbiased up to critic error at re-entry, favorable in practice since π^{rec} is near-deterministic; a multi-sample extension is in Appendix A.5. The objective-gap bound (Theorem 4) is controlled by the unobservable out-of-region rate $\beta(\theta, d)$, tracked via the observable $\alpha(\theta, d)$ as a diagnostic (Appendix E.15). A certified observable bound remains open (a deterministic recovery makes the natural simulation-lemma bound vacuous; routes in Appendix A.7).

Generality of the compatibility regularizer. The gate is also what makes the regularizer general: without it L^{compat} is DAgger-style imitation (Proposition 10), so outcome-gated imitation should transfer to other teacher-intervention settings, human-in-the-loop RL (Spencer et al., 2020) or JSRL (Uchendu et al., 2023).

Beyond locomotion. That transfer argument is theoretical; our evaluation covers simulated locomotion only, though the correction applies wherever a fixed recovery policy takes over at unsafe states (a scripted reset in manipulation, an emergency-stop in driving); validating these is future work. The method also presupposes a pre-trained recovery policy and a hand-designed distance $\mathcal{D}$ (Appendices C and D).

10 Conclusion

We built SafeExplorer, an on-policy RL algorithm for safety-critical control that, as a single PPO update, cuts training falls on three simulated locomotion environments by $21\times$ to roughly $255\times$ (10 seeds) while matching or beating standard PPO on reward.

These gains rest on a theoretical foundation for RL with an external recovery policy: a masked policy-gradient theorem unbiased for the gradient of the mixed-policy return J^{mix} under any recovery, an objective-gap bound with an exact fixed point, an analytic recovery value, and an outcome-gated compatibility regularizer, all instantiated for *safe-region intervention* to minimize training falls. The unbiasedness claim attaches to the Theorem 1 estimator of $\nabla_{\theta} J^{\text{mix}}$, not to the full practical update; Theorem 4 bridges to J .

Acknowledgements

This work benefited from a grant from the Fonds de recherche du Québec (<https://doi.org/10.69777/332223>) and from a Borealis AI Fellowship. We also acknowledge funding support from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canadian Institute for Advanced Research (CIFAR), and compute support from the Digital Research Alliance of Canada (through Calcul Québec and SHARCNET), Mila IDT, and NVIDIA.

References

- Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 22–31, 2017.
- Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Legged locomotion in challenging terrains using egocentric vision. In *Conference on Robot Learning*, 2022.
- Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:29304–29320, 2021.
- Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.
- Eitan Altman. *Constrained Markov Decision Processes*. Chapman & Hall/CRC, 1999.
- Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *Proceedings of the 2019 European Control Conference (ECC)*, pp. 3420–3431, 2019.
- Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1726–1734, 2017.
- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML ’09*, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL <https://doi.org/10.1145/1553374.1553380>.
- Miroslav Bogdanovic, Majid Khadiv, and Ludovic Righetti. Model-free reinforcement learning for robust locomotion using demonstrations from trajectory optimization. *Frontiers in Robotics and AI*, 9, 2022.
- Jiawei Chiu, Jean-Pierre Sleiman, Mayank Mittal, Farbod Farshidian, and Marco Hutter. A collision-free MPC for whole-body dynamic locomotion and manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4686–4693, 2022.
- William G. Cochran. *Sampling Techniques*. John Wiley & Sons, 3rd edition, 1977.
- Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *CoRR*, abs/1801.08757, 2018.
- Thomas Degris, Martha White, and Richard S. Sutton. Off-policy actor-critic. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.
- Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 1407–1416, 2018.
- Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine. Leave no trace: Learning to reset for safe and autonomous reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2018.
- Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, pp. 482–495, 2017.
- Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015.

- Shixiang Gu, Tim Lillicrap, Zoubin Ghahramani, Richard E. Turner, Bernhard Schölkopf, and Sergey Levine. Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- Sehoon Ha, Joonho Lee, Michiel van de Panne, Zhaoming Xie, Wenhao Yu, and Majid Khadiv. Learning-based legged locomotion: State of the art and future perspectives. *The International Journal of Robotics Research*, 44(8):1396–1427, 2025.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL <https://proceedings.mlr.press/v80/haarnoja18b.html>.
- Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. In *Robotics: Science and Systems XV*, 2019. doi: 10.15607/RSS.2019.XV.011.
- Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Cautious reinforcement learning with logical constraints. In Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith (eds.), *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS ’20, Auckland, New Zealand, May 9-13, 2020*, pp. 483–491. International Foundation for Autonomous Agents and Multiagent Systems, 2020. doi: 10.5555/3398761.3398821.
- Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G. M. Araújo. CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL <http://jmlr.org/papers/v23/21-1342.html>.
- Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40:698 – 721, 2021.
- Matthew Thomas Jackson, Uljad Berdica, Jarek Liesen, Shimon Whiteson, and Jakob Nicolaus Foerster. A clean slate for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.
- Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. OmniSafe: An infrastructure for accelerating safe reinforcement learning research. *Journal of Machine Learning Research*, 25(285):1–6, 2024.
- Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pp. 652–661, 2016.
- Sebastian Junges, Nils Jansen, Christian Dehnert, Ufuk Topcu, and Joost-Pieter Katoen. Safety-constrained reinforcement learning for MDPs. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Lecture Notes in Computer Science, pp. 130–146, 2016. doi: 10.1007/978-3-662-49674-9_8.
- Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*, pp. 267–274, 2002.
- Katie Kang, Paula Gradu, Jason J. Choi, Michael Janner, Claire J. Tomlin, and Sergey Levine. Lyapunov density models: Constraining distribution shift in learning-based control. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 10708–10733. PMLR, 2022.
- Michael Kearns, Yishay Mansour, and Andrew Y. Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning*, 49(2–3):193–208, 2002.

- Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-Dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083, 2019. doi: 10.1109/ICRA.2019.8793698.
- Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOREL: Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.
- Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: rapid motor adaptation for legged robots. In Dylan A. Shell, Marc Toussaint, and M. Ani Hsieh (eds.), *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021*, 2021. doi: 10.15607/RSS.2021.XVII.011.
- Joonho Lee, Jemin Hwangbo, and Marco Hutter. Robust recovery controller for a quadrupedal robot using deep reinforcement learning. *CoRR*, abs/1901.07517, 2019.
- Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, 2020. doi: 10.1126/scirobotics.abc5986.
- Minghuan Liu, Zixuan Chen, Xuxin Cheng, Yandong Ji, Ruihan Yang, and Xiaolong Wang. Visual whole-body control for legged loco-manipulation. In *Conference on Robot Learning (CoRL)*, 2024.
- Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc G. Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.
- Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299, 2018. doi: 10.1109/ICRA.2018.8463162.
- Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020. URL <http://jmlr.org/papers/v21/20-212.html>.
- Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3878–3887. PMLR, 2018.
- Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3803–3810, 2018. doi: 10.1109/ICRA.2018.8460528.
- Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. In Marc Toussaint, Antonio Bicchi, and Tucker Hermans (eds.), *Robotics: Science and Systems XVI, Virtual Event / Corvallis, Oregon, USA, July 12-16, 2020*, 2020. doi: 10.15607/RSS.2020.XVI.064.
- Doina Precup, Richard S. Sutton, and Satinder P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, pp. 759–766, 2000.
- Xun Pua and Majid Khadiv. Safe learning of locomotion skills from MPC. In *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, pp. 459–466, 2024. doi: 10.1109/Humanoids58906.2024.10769799.
- Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. Technical report, OpenAI, 2019. URL <https://cdn.openai.com/safexp-short.pdf>.
- Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 627–635, 2011.

- Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In Aleksandra Faust, David Hsu, and Gerhard Neumann (eds.), *Conference on Robot Learning, 8-11 November 2021, London, UK*, volume 164 of *Proceedings of Machine Learning Research*, pp. 91–100. PMLR, 2021.
- William Saunders, Girish Sastry, Andreas Stuhlmüller, and Owain Evans. Trial without error: Towards safe reinforcement learning via human intervention. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS ’18*, pp. 2067–2069, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.
- John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.
- Laura M. Smith, J. Chase Kew, Tianyu Li, Linda Luu, Xue Bin Peng, Sehoon Ha, Jie Tan, and Sergey Levine. Learning and adapting agile locomotion skills by transferring experience. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu (eds.), *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023a. doi: 10.15607/RSS.2023.XIX.051.
- Laura M. Smith, Ilya Kostrikov, and Sergey Levine. Demonstrating A Walk in the Park: Learning to walk in 20 minutes with model-free reinforcement learning. In *Robotics: Science and Systems XIX*, 2023b. doi: 10.15607/RSS.2023.XIX.056.
- Laura M. Smith, Yunhao Cao, and Sergey Levine. Grow your limits: Continuous improvement with Real-World RL for robotic locomotion. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10829–10836, 2024.
- Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyang Wang, David H Mguni, Jun Wang, and Haitham Ammar. Saute RL: Almost surely safe reinforcement learning using state augmentation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20423–20443. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/sootla22a.html>.
- Jonathan Spencer, Sanjiban Choudhury, Matthew Barnes, Matthew Schmitt, Mung Chiang, Peter Ramadge, and Sidd Srinivasa. Learning from interventions: Human-robot interaction as both explicit and implicit feedback. In *Proceedings of Robotics: Science and Systems (RSS)*, 2020.
- Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. Learning to be safe: Deep RL with a safety critic. *CoRR*, abs/2010.14603, 2020.
- Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID Lagrangian methods. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.
- Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.
- Jie Tan, Tingnan Zhang, Erwin Coumans, Atıl İscen, Yunfei Bai, Daniyar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In Hadas Kress-Gazit, Siddhartha S. Srinivasa, Tom Howard, and Nikolay Atanasov (eds.), *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018. doi: 10.15607/RSS.2018.XIV.010.

- Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019.
- Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minh Hwang, Joseph E. Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery RL: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6(3):4915–4922, 2021. doi: 10.1109/LRA.2021.3070252.
- Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.
- Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. Gymnasium, March 2023.
- Matteo Turchetta, Andrey Kolobov, Shital Shah, Andreas Krause, and Alekh Agarwal. Safe reinforcement learning via curriculum induction. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.
- Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, Sergey Levine, and Karol Hausman. Jump-start reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.
- Akifumi Wachi, Xun Shen, and Yanan Sui. A survey of constraint formulations in safe reinforcement learning. In Kate Larson (ed.), *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pp. 8262–8271. International Joint Conferences on Artificial Intelligence Organization, August 2024. doi: 10.24963/ijcai.2024/913. URL <https://doi.org/10.24963/ijcai.2024/913>. Survey Track.
- Nolan C. Wagener, Byron Boots, and Ching-An Cheng. Safe reinforcement learning using advantage-based intervention. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10630–10640. PMLR, 2021.
- Tsung-Yen Yang, Tingnan Zhang, Linda Luu, Sehoon Ha, Jie Tan, and Wenhao Yu. Safe reinforcement learning for legged locomotion. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2454–2461, 2022. doi: 10.1109/IROS47612.2022.9982038.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, 2020.
- Kevin Zakka, Yuval Tassa, and MuJoCo Menagerie Contributors. MuJoCo Menagerie: A collection of high-quality models for the MuJoCo physics engine, 2022. URL http://github.com/google-deeppmind/mujoco_menagerie.
- Weiye Zhao, Tairan He, Rui Chen, Tianhao Wei, and Changliu Liu. State-wise safe reinforcement learning: A survey. In Edith Elkind (ed.), *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 6814–6822. International Joint Conferences on Artificial Intelligence Organization, August 2023. doi: 10.24963/ijcai.2023/763. URL <https://doi.org/10.24963/ijcai.2023/763>. Survey Track.

A Full Proofs

A.1 Proof of Theorem 1 (unbiased gradient of $J^{\text{mix}}(\theta)$)

Proof. Starting from the definition, $J^{\text{mix}}(\theta) = \int p_{\theta}^{\text{mix}} r(\tau) d\tau$. Under Assumption 1, differentiation and integration commute: $\nabla_{\theta} J^{\text{mix}}(\theta) = \int \nabla_{\theta} p_{\theta}^{\text{mix}} r(\tau) d\tau$. Factor p_{θ}^{mix} explicitly by partition membership of the

visited states:

$$p_{\theta}^{\text{mix}} = \underbrace{p(s_1)}_{\text{no } \theta} \cdot \underbrace{\prod_{t:s_t \in \mathcal{M}} \pi_{\theta}(a_t | s_t)}_{\text{only } \theta\text{-dependent factor}} \cdot \underbrace{\prod_{t:s_t \notin \mathcal{M}} \mu(a_t | s_t)}_{\text{no } \theta} \cdot \underbrace{\prod_{t=1}^T P(s_{t+1} | s_t, a_t)}_{\text{no } \theta}, \quad (12)$$

where $\mu(\cdot | s)$ denotes the (possibly Dirac) θ -independent action measure at $s \notin \mathcal{M}$. Equation (12) is an equality of measures on trajectory space and does *not* require μ to have a Lebesgue density. Deterministic, stochastic, point-mass at a hand-coded action, and pre-trained-policy measures are all covered. Only the $\mathcal{M}$ -step factor depends on θ . By the product rule and the score-function identity applied factor-wise,

$$\nabla_{\theta} p_{\theta}^{\text{mix}} = p_{\theta}^{\text{mix}} \sum_{t:s_t \in \mathcal{M}} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t). \quad (13)$$

No gradient of μ appears, because μ carries no θ -dependence; at no point do we evaluate log of a Dirac measure. Substituting back,

$$\nabla_{\theta} J^{\text{mix}}(\theta) = \mathbb{E}_{\tau \sim p_{\theta}^{\text{mix}}} \left[\left(\sum_{t:s_t \in \mathcal{M}} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) r(\tau) \right]. \quad \square$$

Corollaries (Corollaries 2, 8 and 9). The proof never used the specific form of μ , only that μ carries no θ -dependence, so each corollary follows by simply instantiating the pair $(\mathcal{M}, \mu)$. The safe-region intervention case ($\mathcal{M} = \mathcal{R}$, $\mu = \pi^{\text{rec}}$) is the body of this paper. The Jump-Start RL case ($\mathcal{M} = \{(s, t) : t \geq h\}$ for a per-trajectory handoff step h , $\mu = \pi^{\text{teach}}$ for a fixed teacher) inherits the same factorization: the teacher is θ -independent, the handoff step is data-dependent but not θ -dependent, and the score function evaluates only on the student-controlled tail of each trajectory. The state-triggered shielded RL case ($\mathcal{M} = \{s : \Sigma \text{ does not engage at } s\}$, $\mu(a | s) = \delta(a - \sigma(s))$ for the shield’s corrective action σ) inherits the same factorization with μ a Dirac measure that the proof’s factor-level differentiation handles directly. The common payoff is the same in all three: the IS fix is ill-defined when μ has no density at the relevant states, whereas the factor-level proof never forms the ratio and so bypasses the need for a density on μ . We state the two non-safe-region specializations formally for reference.

Corollary 8 (Jump-Start RL (Uchendu et al., 2023)). *Let π^{teach} be a fixed teacher policy and h a per-trajectory handoff step. Take $\mathcal{M} = \{(s, t) : t \geq h\}$ (states reached after handoff; the timestep t is absorbed into the state, standard for finite-horizon MDPs, so $\mathcal{M}$ is a subset of the augmented state space) and $\mu = \pi^{\text{teach}}$. Then Theorem 1 gives an unbiased gradient of the student’s contribution to the mixed return, where the score function is evaluated only on the student-controlled tail of each trajectory.*

Corollary 9 (State-triggered shielded RL). *Let Σ be a state-triggered shield, whose engagement is decided by the state alone and which, at unsafe states, replaces the proposed action with a θ -independent corrective action $\sigma(s)$. Take $\mathcal{M} = \{s : \Sigma \text{ does not engage at } s\}$ and $\mu(a | s) = \delta(a - \sigma(s))$. Then Theorem 1 gives an unbiased gradient evaluated only at unshielded states, with no density required on the shield’s correction.*

Action-triggered shields, as in Alshiekh et al. (2018), decide the override from the proposed action $a \sim \pi_{\theta}(\cdot | s)$: at states where only some actions are unsafe, the executed-action distribution mixes π_{θ} restricted to safe actions with the correction weighted by the θ -dependent probability of proposing an unsafe action, so it depends on θ and falls outside Theorem 1’s hypothesis of a θ -independent μ .

Truncated importance sampling. One might hope that the singularity is only a problem for the naive IS estimator and that a more careful clipped estimator escapes it. It does not. Truncated-IS estimators such as V-trace (Espenholt et al., 2018) and Retrace (Munos et al., 2016) reach the mixed-policy gradient by clipping the importance ratio $p_{\theta}/p_{\theta}^{\text{mix}}$, but that ratio is still undefined at the deterministic-recovery steps where μ is a point mass. Clipping bounds variance downstream of the singularity rather than removing it. The factor-level proof above avoids the ratio entirely, which is exactly why it extends to the deterministic recovery policy our method uses.

A.2 Proof of Theorem 4 (objective-gap bound)

This subsection and the remainder of the appendix work in the infinite-horizon discounted convention fixed in Section 3, under which $V^{\pi_\theta^{\text{mix}}}$, $Q^{\pi_\theta^{\text{mix}}}$, $A^{\pi_\theta^{\text{mix}}}$, and the normalized discounted visitations ν^{π_θ} , $\nu^{\pi_\theta^{\text{mix}}}$ are stationary.

Proof. Theorem 1 shows we optimize J^{mix} rather than the deployment objective J , so the question is how far apart the two can be. We bound the gap by routing it through the states where the two policies actually differ. By the Performance Difference Lemma (Kakade & Langford, 2002),

$$J(\theta) - J^{\text{mix}}(\theta) = \frac{1}{1-\gamma} \mathbb{E}_{s \sim \nu^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} [A^{\pi_\theta^{\text{mix}}}(s, a)], \quad (14)$$

where $A^{\pi_\theta^{\text{mix}}}(s, a) = Q^{\pi_\theta^{\text{mix}}}(s, a) - V^{\pi_\theta^{\text{mix}}}(s)$. Split the expectation by the safe region:

$$J(\theta) - J^{\text{mix}}(\theta) = \frac{1}{1-\gamma} \left(\underbrace{\mathbb{E}_{s \in \mathcal{R}}[\cdot]}_{=0} + \mathbb{E}_{s \notin \mathcal{R}}[\cdot] \right).$$

On $s \in \mathcal{R}$, both π_θ and π_θ^{mix} sample from π_θ , so the inner expectation $\mathbb{E}_{a \sim \pi_\theta(\cdot|s)} [A^{\pi_\theta^{\text{mix}}}(s, a)] = 0$ by definition of $V^{\pi_\theta^{\text{mix}}}$. On $s \notin \mathcal{R}$, $|A^{\pi_\theta^{\text{mix}}}(s, a)| \leq |Q^{\pi_\theta^{\text{mix}}}(s, a)| + |V^{\pi_\theta^{\text{mix}}}(s)| \leq 2r_{\max}/(1-\gamma)$ since both $|Q|$ and $|V|$ are bounded by $r_{\max}/(1-\gamma)$. Therefore

$$|J(\theta) - J^{\text{mix}}(\theta)| \leq \frac{1}{1-\gamma} \cdot \frac{2r_{\max}}{1-\gamma} \cdot \Pr_{s \sim \nu^{\pi_\theta}}[s \notin \mathcal{R}] = \frac{2r_{\max}}{(1-\gamma)^2} \beta(\theta, d). \quad \square$$

Bound on the out-of-region rate $\beta(\theta, d)$ (Section 4.2). The gap above scales with $\beta(\theta, d)$, so the bound is only useful once $\beta(\theta, d)$ itself is controlled. Section 4.2 uses $\beta(\theta, d) \leq \eta\gamma/(1-\gamma) \leq \eta/(1-\gamma)$, which we now derive from the per-step invariance slack, under the additional hypothesis that episodes start inside the safe region, $\text{supp } p(s_1) \subseteq \mathcal{R}(d)$ (otherwise the initial step alone contributes $(1-\gamma)\Pr[s_1 \notin \mathcal{R}]$ to $\beta(\theta, d)$ even with $\eta = 0$). All three of our environments satisfy the hypothesis: episodes reset to an upright pose inside the region. Starting from a reachable initial state $s_0 \in \mathcal{R}$ (this derivation counts steps from zero, so s_0 here is the s_1 of the trajectory notation), Assumption 2 gives $\Pr[s_0, \dots, s_t \in \mathcal{R}] \geq (1-\eta)^t$. Since $\{s_t \in \mathcal{R}\} \supseteq \{s_0, \dots, s_t \in \mathcal{R}\}$, the marginal in-region probability dominates the all-stay probability, $\Pr[s_t \in \mathcal{R}] \geq (1-\eta)^t$, hence $\Pr[s_t \notin \mathcal{R}] \leq 1 - (1-\eta)^t$. With the discounted state-visitation $\beta(\theta, d) = (1-\gamma) \sum_{t \geq 0} \gamma^t \Pr[s_t \notin \mathcal{R}]$,

$$\beta(\theta, d) \leq (1-\gamma) \sum_{t \geq 0} \gamma^t (1 - (1-\eta)^t) = \frac{\eta\gamma}{1-\gamma(1-\eta)} \leq \frac{\eta\gamma}{1-\gamma} \leq \frac{\eta}{1-\gamma},$$

the last step using $1-\gamma(1-\eta) \geq 1-\gamma$ and $\gamma \leq 1$.

Train-to-deploy reward gap (footnote in Section 4.3). The training-signal convention of Proposition 6 (zero per-step reward during recovery, one-time terminal reward r_{term}) and the unmodified task reward differ only at states outside $\mathcal{R}$ (and through the one-time r_{term}). Bounding the per-step discrepancy by $r_{\max} + |r_{\text{term}}|$ on those steps and summing the discounted visitation under π_θ , the returns the two conventions induce under π_θ differ by at most $(r_{\max} + |r_{\text{term}}|) \beta(\theta, d)/(1-\gamma)$, so the train-to-deploy chain is controlled by the same $\beta(\theta, d)$ as Theorem 4.

A.3 Proof of Corollary 5 (conditional fixed point)

Proof. The gap bound of Theorem 4 leaves $\beta(\theta, d)$ free, so the natural endpoint of the analysis is the case that drives $\beta(\theta, d)$ to zero. The corollary splits into two readings of when that happens.

Reading (R-A): If $\mathcal{R}(d_{\max})$ covers the reachable state space, then $\Pr[s \notin \mathcal{R}(d_{\max})] = 0$ under *any* action distribution, including π_θ alone. Hence $\beta(\theta, d_{\max}) = 0$, and Equation (7) gives $J^{\text{mix}}(\theta) = J(\theta)$ exactly. Furthermore, π_θ^{mix} is identical to π_θ on the reachable support of $p_\theta(\tau)$ (recovery is never engaged).

Reading (R-B): If $\mathcal{R}(d_{\max})$ is strictly contained in the reachable state space, the theory no longer forces π_θ to remain in $\mathcal{R}$, so its invariance slack $\eta_\star := \eta(\theta, d_{\max}) \geq 0$ (Assumption 2) need not be zero. Substituting $\beta \leq \eta_\star/(1-\gamma)$ (derived above under the initial-state hypothesis $\text{supp } p(s_1) \subseteq \mathcal{R}$) into Equation (7) gives the upper bound $|J(\theta) - J^{\text{mix}}(\theta)| \leq 2r_{\max}\eta_\star/(1-\gamma)^3$, which, unlike (R-A), is not pinned to zero. Whether $\eta_\star$ (and hence the gap) is small is environment-dependent: Appendix E.15 finds $\eta_\star \rightarrow 0$ on HALFCHEETAH and ANT, where the gap closes to seed noise, and $\eta_\star > 0$ on GO1, where a proportional residual remains. $\square$

A.4 Proof of Proposition 6 (analytic recovery value)

Proof. The gradient and gap results above treat $V^{\pi_\theta^{\text{mix}}}$ as given; what remains is to compute it cheaply on the recovery segments themselves, where the trajectory is no longer under the agent’s control. By the definition of $V^{\pi_\theta^{\text{mix}}}$, $V^{\pi_\theta^{\text{mix}}}(s_t) = \mathbb{E}_{\tau \sim p_\theta^{\text{mix}}(\tau)|s_t} \left[\sum_{j=0}^{\infty} \gamma^j r_{t+j} \right]$. Under deterministic dynamics and deterministic recovery, the trajectory starting from a recovery-triggering state s_t is fully determined for the duration of the recovery segment $[t, t+k]$. The expectation collapses. Two cases:

Recovery success ($s_{t+k} \in \mathcal{R}$): the realized return over the segment is $G_{t,k} = \sum_{j=0}^{k-1} \gamma^j r_{t+j}$. From s_{t+k} the agent resumes under π_θ^{mix} with value $V^{\pi_\theta^{\text{mix}}}(s_{t+k})$. Hence $V^{\pi_\theta^{\text{mix}}}(s_t) = G_{t,k} + \gamma^k V^{\pi_\theta^{\text{mix}}}(s_{t+k})$.

Recovery failure (terminal): the realized return is $G_{t,k}$ over the k deterministic steps, the episode ends, no further reward accrues. Hence $V^{\pi_\theta^{\text{mix}}}(s_t) = G_{t,k}$.

Combining gives Equation (8). Our implementation instantiates this general statement under a chosen reward: zero per-step reward during recovery and a one-time terminal reward r_{term} accrued at the final segment step $(t+k-1)$ on failure. The terminal penalty is thus the only nonzero contribution to the segment return $G_{t,k} = \sum_{j=0}^{k-1} \gamma^j r_{t+j}$, which is therefore 0 on success and $\gamma^{k-1} r_{\text{term}}$ on failure, matching Equation (8). $\square$

A.5 Analytic value under stochastic dynamics

The closed form just derived rests on the determinism assumption, so it is worth asking what survives when that assumption is relaxed. Proposition 6 assumes deterministic P and π^{rec} . Under process noise or stochastic recovery, the realized $\gamma^k V_\theta(s_{t+k})$ is no longer equal to $V^{\pi_\theta^{\text{mix}}}(s_t)$. It becomes a single Monte-Carlo realization drawn from the distribution over post-trigger trajectories, whose variance grows with the noise of P and π^{rec} accumulated over the segment. Two natural extensions preserve the spirit of bypassing critic bootstrapping across recovery. (i) A multi-sample MC target: re-roll the recovery M times from s_t (where the simulator supports state-resets) and average $G_{t,k_m} + \gamma^{k_m} V_\theta(s_{t+k_m})$ across the M realizations. This averages over the segment stochasticity, with variance falling as $1/M$. (ii) The single-sample plug-in we already use is a one-sample MC estimate of $V^{\pi_\theta^{\text{mix}}}(s_t)$, and its single-realization noise reaches downstream GAE. Both targets bootstrap the endpoint critic, so both are unbiased for $V^{\pi_\theta^{\text{mix}}}(s_t)$ only to the extent that V_θ is unbiased at the re-entry state; any bias there carries through, and the stochastic-case guarantee is conditional on it. The bias-variance trade-off is favorable in our setting because π^{rec} is approximately deterministic in practice (MPC, greedy SAC), so the per-segment variance is small. A formal analysis under controlled stochasticity, and the regime where (i) is worth the extra simulator calls, is left to follow-up.

A.6 Proof of Proposition 10

The recovery segments that supplied the analytic value also supply state-action pairs for the compatibility regularizer, and identifying its limiting cases shows that this loss is not an ad-hoc addition but a familiar imitation objective in disguise.

Proposition 10 (Special cases of the compatibility regularizer). *Let π^{rec} be deterministic, so $a_t^{\text{rec}} = \pi^{\text{rec}}(s_t)$ at recovery-controlled states, and consider the gate-set choice $\sigma_k \equiv 1$ for all segments.*

- (a) **Behavioral cloning on the recovery dataset.** Under $\sigma_k \equiv 1$,

$$L_{\sigma \equiv 1}^{\text{compat}}(\theta) = -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_{t: s_t \notin \mathcal{R}} \log \pi_{\theta}(\pi^{\text{rec}}(s_t) | s_t), \quad (15)$$

the (scaled) negative log-likelihood of π_{θ} on the dataset $\mathcal{B} = \{(s_t, \pi^{\text{rec}}(s_t)) : s_t \notin \mathcal{R}\}$ of recovery-controlled state-action pairs collected from rollouts of $\pi_{\theta}^{\text{mix}}$. Equivalently, $L_{\sigma \equiv 1}^{\text{compat}}$ is the per-step behavioral-cloning objective applied to π^{rec} 's actions at the states $\pi_{\theta}^{\text{mix}}$ visits during recovery.

- (b) **A region-restricted DAgger aggregation step.** The dataset $\mathcal{B}$ is collected as in one iteration of DAgger (Ross et al., 2011) with the teacher queried only on the intervention set: states are visited under the mixed roll-out policy $\pi_{\theta}^{\text{mix}}$ (predominantly student-induced up to the safe-region boundary), and labels are the controller π^{rec} 's actions at the recovery-controlled states only, whereas DAgger labels every visited state. Minimizing $L_{\sigma \equiv 1}^{\text{compat}}$ as the per-iteration training step then reproduces a region-restricted analogue of DAgger's update with π^{rec} in the teacher role and π_{θ} as the student.
- (c) **Outcome gate as a designer filter.** The general gate $\sigma_k \in \{0, 1\}$ restricts $\mathcal{B}$ to recovery segments that re-entered $\mathcal{R}$, discarding the rest. JSRL-style mixed-policy data collection (Uchendu et al., 2023) produces the same mixed-policy roll-out distribution but does not include an imitation loss. L^{compat} with $\sigma_k \equiv 1$ supplies the analogous student-imitation step that the JSRL algorithm itself omits. What is new in the case $\sigma_k \neq 1$ is gating imitation of an external recovery controller by the realized success of its multi-step segment, as opposed to gating by value estimates of one's own past actions (Oh et al., 2018) or of demonstrations (Nair et al., 2018).

Proof. (a) From Equation (11) with $w_t = \sigma_{\text{seg}(t)}$, where $\text{seg}(t)$ is the index of the recovery segment containing t (Section 4.4), and the assumption $\sigma_k \equiv 1$,

$$L_{\sigma \equiv 1}^{\text{compat}}(\theta) = -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_{t: s_t \notin \mathcal{R}} 1 \cdot \log \pi_{\theta}(a_t^{\text{rec}} | s_t) = -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_{t: s_t \notin \mathcal{R}} \log \pi_{\theta}(\pi^{\text{rec}}(s_t) | s_t),$$

using $a_t^{\text{rec}} = \pi^{\text{rec}}(s_t)$ for deterministic π^{rec} . This is, up to the constant scale $\lambda_{\text{compat}}/N_{\text{rec}}$, the empirical negative log-likelihood of π_{θ} on the dataset $\mathcal{B} = \{(s_t, \pi^{\text{rec}}(s_t)) : s_t \notin \mathcal{R}\}$, the (population-level) maximum-likelihood / behavioral-cloning objective for π_{θ} on $\mathcal{B}$.

(b) The dataset $\mathcal{B}$ is collected by rolling out $\pi_{\theta}^{\text{mix}}$, retaining the state-action pairs at the unsafe-region timesteps. Under $\pi_{\theta}^{\text{mix}}$, π_{θ} controls the rollout up to the first exit from $\mathcal{R}$ (so the visited boundary state s_t has π_{θ} -induced distribution), and π^{rec} provides actions thereafter. This matches the data-collection rule of a single DAgger iteration with π^{rec} as the teacher and π_{θ} as the student, states visited under the mixed roll-out policy and labels from the teacher, except that the teacher is queried only at the recovery-controlled states rather than at every visited state as in Ross et al. (2011). Minimizing $L_{\sigma \equiv 1}^{\text{compat}}$ as the per-iteration update therefore reproduces a region-restricted analogue of the DAgger aggregation step.

- (c) For general $\sigma_k \in \{0, 1\}$, the regularizer is

$$L^{\text{compat}}(\theta) = -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_k \sigma_k \sum_{t: s_t \notin \mathcal{R}, \text{seg}(t)=k} \log \pi_{\theta}(\pi^{\text{rec}}(s_t) | s_t),$$

i.e. the behavioral-cloning objective restricted to the sub-dataset $\mathcal{B}_{\text{succ}} = \{(s_t, \pi^{\text{rec}}(s_t)) : s_t \notin \mathcal{R}, \sigma_{\text{seg}(t)} = 1\}$ collected from segments that re-entered $\mathcal{R}$. JSRL (Uchendu et al., 2023) likewise rolls out a mixed teacher-then-student policy but trains the student only by reinforcement on the student-controlled tail. It does not include a teacher-imitation step. The $\sigma_k \equiv 1$ specialization of L^{compat} is the analogous imitation loss JSRL omits. In the $\sigma_k \neq 1$ case, the new element is the gating criterion and target: imitation of an external recovery controller gated by the realized success of its multi-step segment, rather than by value estimates of the agent's own past actions (Oh et al., 2018) or of demonstrations (Nair et al., 2018). $\square$

A.7 Sketch: why an observable upper bound on β is hard in continuous actions

The bound on $\beta(\theta, d)$ above is stated in terms of the per-step invariance slack η , which is not directly measurable; a bound in terms of an observable quantity would be far more useful, and we close by explaining

why that is hard. A natural attempt is to bound β by the observable recovery rate α via a simulation-lemma argument: $\beta = \Pr_{\nu^{\pi_\theta}}[s \notin \mathcal{R}] \leq \alpha + \text{TV}(\nu^{\pi_\theta}, \nu^{\pi_\theta^{\text{mix}}})$. Here $\text{TV}(\cdot, \cdot)$ denotes total variation (TV) distance. The TV term can in turn be bounded by the expected per-step TV between π_θ and π_θ^{mix} via the standard simulation lemma (Kearns et al., 2002). But on unsafe states π_θ^{mix} collapses to a Dirac (deterministic recovery) while π_θ remains absolutely continuous. The per-step TV at unsafe states is 1 pointwise, so the simulation-lemma term contributes a full $1/(1 - \gamma) > 1$, and the bound $\beta \leq \alpha + 1/(1 - \gamma)$ exceeds 1 and is therefore vacuous (since $\beta \leq 1$ trivially). Routes that may produce a non-vacuous bound include (i) replacing the deterministic recovery with a smoothed (LSE / Gaussian-mixture) variant whose TV with π_θ is finite, (ii) Wasserstein-action-gap bounds under known dynamics Lipschitz constants, and (iii) periodic offline evaluation of π_θ alone. We sketch (i) below and leave a complete treatment to follow-up work.

Sketch via smoothed recovery. Replace $\pi^{\text{rec}}(\cdot|s) = \delta(a - \bar{a}(s))$ with $\tilde{\pi}_{\text{rec}}(\cdot|s) = \mathcal{N}(\bar{a}(s), \sigma_{\text{rec}}^2 I)$ for small σ_{rec} . Then both $\pi_\theta(\cdot|s)$ and $\tilde{\pi}_{\text{rec}}(\cdot|s)$ are absolutely continuous and, when the scales are matched ($\sigma_{\text{rec}} \approx \sigma_\theta(s)$, with σ_θ the main policy’s standard deviation), the Gaussian TV bound gives $\text{TV}(\pi_\theta, \tilde{\pi}_{\text{rec}}) \leq \text{const} \cdot \|\mu_\theta(s) - \bar{a}(s)\|/\sigma_{\text{rec}}$ at unsafe states up to a scale-mismatch term (with mismatched covariances the TV is bounded away from zero even at equal means), so the per-step TV is integrable under these stated conditions. The constant, however, grows with the action dimension, so even this smoothed bound loosens in high dimensions, the same mechanism by which the importance weights collapse on the 12-dimensional Go1 action space (Appendix E.16). Moreover, the TV to a fixed-scale π_θ approaches one as $\sigma_{\text{rec}} \rightarrow 0$, so the smoothing scale trades fidelity to the deployed deterministic recovery against tightness of the bound; the sketch therefore yields an observable upper-bound proxy for β only under the matched-scale conditions above, not a general bound.

B Algorithm Details

B.1 Full pseudocode

Algorithm 1 gives the complete SafeExplorer update. The per-environment curriculum grows the safe-region radius linearly, $d_{\text{iter}}(u) = d_0 + \frac{u-1}{N} d_{\text{max}}$ at update u of N , with $d_0 = 0.01$ on HALF-CHEETAH and ANT and $d_0 = 0.05$ on GO1, up to the per-environment d_{max} values in Table 4 (Appendix C).

B.2 The exact target induced by the whole-segment overwrite

Algorithm 1 overwrites the value at *every* step of a recovery segment, not only at the triggering state. The following lemma derives the exact λ -GAE target this implementation induces and relates it to the analytic target of Proposition 6; a start-only overwrite would not have this property for $\lambda < 1$, since intermediate learned-critic values would enter the TD residuals.

Lemma 11 (Induced GAE target of the whole-segment overwrite). *Consider a recovery segment of length k starting at t that is resolved within the rollout, by re-entry at $t+k$ (success) or episode termination at $t+k-1$ (failure), with intermediate recovery rewards stored as zero, the failure-terminal step storing r_{term} , and values overwritten as in Algorithm 1: on success $V(s_{t+j}) \leftarrow \gamma^{k-j} V_\theta(s_{t+k})$ and on failure $V(s_{t+j}) \leftarrow \gamma^{k-1-j} r_{\text{term}}$, for $j = 0, \dots, k-1$. Then, for every $\lambda \in [0, 1]$:*

- (i) *every intra-segment TD residual vanishes identically, so ordinary λ -GAE telescopes through the segment;*
- (ii) *on failure segments the induced target at the triggering state equals the Proposition 6 analytic target exactly;*
- (iii) *on success segments the induced target equals the analytic target with the endpoint critic bootstrap $V_\theta(s_{t+k})$ partially replaced by the observed post-re-entry λ -return (fully replaced at $\lambda = 1$), which reduces reliance on the endpoint critic rather than corrupting the target; at no λ does any intermediate critic estimate enter;*
- (iv) *the last safe step before the segment receives the residual $\delta = r + \gamma \cdot \gamma^k V_\theta(s_{t+k}) - V(s)$, that is, the entire recovery segment collapses into a single discounted semi-MDP transition. The one difference from a literal one-transition collapse is credit decay: the segment still counts as $k+1$ steps in the λ -return, contributing a factor $(\gamma\lambda)^{k+1}$ rather than the single λ factor of a true one-step collapse.*

Algorithm 1 SafeExplorer: PPO with safe-step masking, analytic recovery V , and hard $C_\theta(\tau)$.

Require: Main policy π_θ , recovery policy π^{rec} , value function V_θ , safe-region radius schedule d_{iter} , total updates N , rollout length T_{ro} , num envs E , compatibility coef λ_{compat} , learning rate η_{lr} , terminal reward convention r_{term} (Appendix C.2).

```

1: for update = 1, ...,  $N$  do
2:    $d \leftarrow d_{\text{iter}}(\text{update})$ 
3:   Rollout phase: for  $T_{\text{ro}}$  steps in each of  $E$  parallel envs,
4:     if  $s_t \in \mathcal{R}(d)$ :  $a_t \sim \pi_\theta(\cdot|s_t)$ , store  $(s_t, a_t, r_t, \log \pi_\theta(a_t|s_t), V_\theta(s_t))$ 
5:     if  $s_t \notin \mathcal{R}(d)$ :  $a_t = \pi^{\text{rec}}(s_t)$ , is_rec $_t \leftarrow 1$ 
6:     store  $(s_t, a_t, r_t, \log \pi_\theta(a_t|s_t), V_\theta(s_t)) \triangleright \log \pi_\theta(a_t^{\text{rec}}|s_t)$  feeds  $L^{\text{compat}}$  only; never enters the PPO
       ratio (safe-step mask).
7:   Analytic-V overwrite, whole segment (Proposition 6, Lemma 11): for each recovery segment
       starting at  $t_k^{\text{start}}$  of length  $\ell_k$  (intermediate recovery rewards stored as zero), and every offset  $j =$ 
        $0, \dots, \ell_k - 1$ ,
8:     if segment ended in re-entry at  $t_k^{\text{start}} + \ell_k$ :  $V(s_{t_k^{\text{start}}+j}) \leftarrow \gamma^{\ell_k-j} V_\theta(s_{t_k^{\text{start}}+\ell_k}) \triangleright$  the segment return
       is 0 under the zeroed-recovery-reward convention, Proposition 6
9:     if segment ended in termination:  $V(s_{t_k^{\text{start}}+j}) \leftarrow \gamma^{\ell_k-1-j} r_{\text{term}} \triangleright$  the terminal reward; recovery
       steps are zeroed, so this is the remaining failure-segment return
10:    if segment cut off by rollout truncation: treat as re-entry at the truncation boundary  $\triangleright$  tentative
       success with a bootstrapped value, Section 4.4
11:    Advantage estimation: compute  $\hat{A}^{\text{full}}$  via GAE on the overwritten value sequence; the value target
       is  $\hat{R}_t = \hat{A}_t^{\text{full}} + V(s_t)$ , with  $V(s_t)$  the stored, possibly overwritten, value.
12:    Outcome gates: for each recovery segment  $k$ , set  $\sigma_k \leftarrow \mathbb{I}[\text{segment ended in re-entry}]$ .
13:    for epoch = 1, ...,  $K$  do
14:      for minibatch  $\subset$  rollout do
15:         $T_{\mathcal{R}} \leftarrow \{t : \text{is\_rec}_t = 0\}$  (safe-step mask)
16:         $L_{\text{safe}}^{\text{PPO}} \leftarrow -\frac{1}{|T_{\mathcal{R}}|} \sum_{t \in T_{\mathcal{R}}} \min(\rho_t \hat{A}_t^{\text{full}}, \text{clip}(\rho_t) \hat{A}_t^{\text{full}})$ 
17:         $L^V \leftarrow \frac{1}{T_{\text{ro}}} \sum_t (V_\theta(s_t) - \hat{R}_t)^2 \triangleright$  unmasked: recovery-trigger targets are variance-free, exact up
       to critic error at re-entry
18:         $L^{\text{compat}} \leftarrow -\frac{\lambda_{\text{compat}}}{N_{\text{rec}}} \sum_{t \notin \mathcal{R}} \sigma_{\text{seg}(t)} \log \pi_\theta(a_t^{\text{rec}}|s_t)$ 
19:         $L \leftarrow L_{\text{safe}}^{\text{PPO}} + c_v L^V + L^{\text{compat}} - c_e H[\pi_\theta]$ 
20:         $\theta \leftarrow \theta - \eta_{\text{lr}} \nabla_\theta L$ 
21:      end for
22:    end for
23: end for

```

Proof. The overwritten values satisfy the zero-reward Bellman recursion exactly. On a success segment, interior steps have reward 0 and consecutive overwritten values $\gamma^{k-j} V_\theta(s_{t+k})$ and $\gamma^{k-j-1} V_\theta(s_{t+k})$, so $\delta_{t+j} = 0 + \gamma \cdot \gamma^{k-j-1} V_\theta(s_{t+k}) - \gamma^{k-j} V_\theta(s_{t+k}) = 0$; on a failure segment the same computation with $\gamma^{k-1-j} r_{\text{term}}$ in place of the bootstrap gives $\delta_{t+j} = 0$, and the failure-terminal step, because it stores r_{term} and has overwritten value r_{term} , gives $\delta = r_{\text{term}} - r_{\text{term}} = 0$. This proves (i). The λ -GAE advantage at any step is the $(\gamma\lambda)$ -discounted sum of downstream residuals; with all intra-segment residuals zero, the value target at the triggering state is its overwritten value plus $(\gamma\lambda)^k$ -discounted post-segment residuals. On failure there are no post-segment residuals, giving (ii). On success the surviving terms are exactly the ones that mix the endpoint bootstrap $\gamma^k V_\theta(s_{t+k})$ with the observed post-re-entry λ -return, in the standard λ -return proportions, giving (iii); at $\lambda = 1$ the bootstrap is fully replaced by the Monte-Carlo return. For (iv), substituting the overwritten trigger-state value into the residual of the preceding safe step yields $\delta = r + \gamma^{k+1} V_\theta(s_{t+k}) - V(s)$, the single discounted semi-MDP transition, and the $(\gamma\lambda)^{k+1}$ credit-decay factor follows from the segment occupying $k+1$ positions in the λ -return. $\square$

Credit decay is a trace convention. The $(\gamma\lambda)^{k+1}$ factor in (iv) is not an error in the target. At $\lambda = 1$ the induced advantage is the Monte-Carlo advantage $G_t - V(s_t)$ under either accounting, so the two coincide

exactly. For $\lambda < 1$ the segment keeps GAE’s per-timestep eligibility trace over an interval that lasts k real steps, whereas a literal collapse would apply a single λ factor to the same interval. We keep the per-timestep trace: it propagates less post-segment credit back across a long intervention, which matches the fact that the post-segment state is set by where π^{rec} lands rather than by the task action preceding the segment. The gap between the two conventions therefore grows with segment length, and is largest on ANT, whose mean segment is 159 steps (Appendix D.4).

Exclusions. The lemma is scoped to segments resolved within a rollout by re-entry or termination. Two code edge cases are excluded and stated as such: time-limit-truncated segments, which keep the environment reward at the last step and bootstrap the post-reset value, and segments spanning an environment auto-reset.

B.3 Notation

Table 2 collects the symbols used in Algorithm 1 and the design rationale below.

Table 2: Notation used in Algorithm 1 and throughout the paper.

$\pi_\theta(a s)$	main policy, stochastic, θ -parameterized
$\pi^{\text{rec}}(a s)$	recovery policy; no θ -dependence; arbitrary form
π_θ^{mix}	mixed policy (Section 3)
$\mathcal{R}, d, d_{\max}$	safe region, current boundary (referred to as the radius), per-environment safe-region maximum (tuned hyperparameter)
$\mathcal{D}(s)$	task-space distance from s to the nominal configuration; $\mathcal{R}(d) = \{s : \mathcal{D}(s) \leq d\}$ (Appendix D.2)
d_0	initial safe-region radius of the curriculum (0.01 on HALF-CHEETAH/ANT, 0.05 on GO1)
$d_{\text{iter}}(u), N$	curriculum schedule $d_0 + \frac{u-1}{N}d_{\max}$ at update u , and the total number of training updates
γ, λ	discount factor and GAE coefficient (0.99 and 0.95, Table 3)
$T_{\mathcal{R}}, \hat{A}^{\text{full}}$	set of safe (main-policy) steps in a rollout, and the GAE advantage over the full mixed-policy trajectory
$p_\theta(\tau), p_\theta^{\text{mix}}(\tau)$	trajectory distributions under π_θ alone and the mixed policy
$J(\theta), J^{\text{mix}}(\theta)$	main-policy return (unobservable during training) and mixed-policy return
$\alpha(\theta, d)$	recovery rate (observable)
$\beta(\theta, d)$	main-policy out-of-region rate (unobservable during training)
$C_\theta(\tau)$	compatibility score $\prod_{t \notin \mathcal{R}} \pi_\theta(a_t^{\text{rec}} s_t)$
$V^{\pi_\theta^{\text{mix}}}, Q^{\pi_\theta^{\text{mix}}}, A^{\pi_\theta^{\text{mix}}}$	value, Q , advantage under the mixed policy
ℓ_k	length of recovery segment k
$\text{seg}(t)$	index of the recovery segment containing step t
σ_k	outcome gate of segment k (1 iff it ends with re-entry into $\mathcal{R}$)
N_{rec}	number of recovery-controlled steps in the current minibatch (normalizer of L^{compat})
r_{term}	environment reward on the terminal step of a failed recovery segment; per-environment values in Appendix C.2

B.4 Operational features of the safe-region intervention setup

The safe-region intervention mechanism produces a training-time setup with four operational features that simultaneously place it outside CMDP and pointwise-filter approaches: (i) the safety predicate is binary set membership $s \in \mathcal{R}$ rather than a scalar cost, so there is no Lagrangian dual to optimize; (ii) π^{rec} may be deterministic (MPC, greedy SAC), in which case it has no Lebesgue density and the IS correction is ill-defined; (iii) recovery takes control for multi-step segments rather than at a single step, so training rollouts follow a genuine mixed-policy distribution rather than a one-step perturbation; (iv) the objective is the original main-policy return $J(\theta)$, not a constrained surrogate, so safety is enforced operationally rather than penalized in the reward. Each of the four method components in Section 4 responds to a specific feature in this list.

B.5 Soft outcome gate

In addition to the hard outcome gate of Section 4.4, we considered a signed, per-step preference based on whether the post-segment state was higher- or lower-value than the pre-segment state, with $k(t)$ the length of the segment containing t :

$$w_t^{\text{soft}} = \gamma^{k(t)} V_\theta(s_{t+k(t)}) - V_\theta(s_t), \quad \text{normalized to zero mean per minibatch.} \quad (16)$$

Empirically, the soft variant is unstable on HALF-CHEETAH and GO1 (the soft-gate variants in Appendix E.6) because the regularizer fits failed-recovery actions, so we drop it from the headline comparison.

B.6 Random-denominator and value-loss target choices

Two further choices in Algorithm 1 warrant explanation: the denominator of the masked policy gradient and the set of timesteps the critic regresses on. The masked PG estimator divides by $|T_{\mathcal{R}}|$, a random variable that depends on the rollout.

From Theorem 1 to Equation (5). Theorem 1 weights the masked score sum by the whole-trajectory return $r(\tau)$, while Equation (5) pairs each safe step’s score with the per-step target $\hat{A}_t^{\text{full}}$. The two standard policy-gradient reductions justify this at safe steps exactly as in PPO. At a safe step the action is drawn from $\pi_\theta(\cdot | s_t)$, so the score $\nabla_\theta \log \pi_\theta(a_t | s_t)$ has conditional mean zero given the history; multiplying it by any history-measurable quantity, the rewards collected before t or a state baseline $b(s_t)$, contributes nothing in expectation, which turns $r(\tau)$ into the reward-to-go minus a baseline. Recovery steps contribute no score term, so neither reduction touches them. Substituting GAE for the reward-to-go and clipping then modify the estimator exactly as in standard PPO, the inherited approximations named in Section 4.1.

The immediate question about Equation (5) is what a random denominator does to the estimator, since in general $\mathbb{E}[\sum_t g_t / |T_{\mathcal{R}}|] \neq \mathbb{E}[\sum_t g_t] / \mathbb{E}[|T_{\mathcal{R}}|]$. We answer it exactly. Equation (5) is a self-normalized ratio estimator whose weights are the safe-step indicators, the construction weighted importance sampling uses in off-policy evaluation (Precup et al., 2000), where an $O(1/n)$ bias is knowingly accepted in exchange for variance reduction (Cochran, 1977). Parts (i) and (ii) below are that standard behaviour specialized to indicator weights; part (iii) is specific to the safe-region setting. Equation (5) is written over a whole rollout for readability; the implementation applies the same normalization within each minibatch, so the denominator is the safe count of the minibatch rather than of the rollout. Both are the same estimator with a different sampling unit, and the proposition is stated for either.

Proposition 12 (What Equation (5) computes). *Fix θ and consider a batch of B timesteps drawn from the mixed-policy occupancy, treated as independent draws (the standard idealization in on-policy analyses; B is the rollout length or the minibatch size according to the sampling unit of Equation (5)). For timestep i put*

$$D_i = \mathbb{1}[s_i \in \mathcal{R}], \quad g_i = D_i \nabla_\theta \log \pi_\theta(a_i | s_i) \hat{A}_i^{\text{full}},$$

so that $\sum_i D_i = |T_{\mathcal{R}}|$ and Equation (5) is the ratio $\hat{g}_B = \bar{g} / \bar{D}$ of the two sample means. Let $\alpha(\theta, d) = \Pr[s_i \notin \mathcal{R}]$, assume $\alpha(\theta, d) < 1$ and $\mathbb{E}\|g_1\| < \infty$, and write $g^* = \mathbb{E}[g_1]$, which Theorem 1 identifies with a positive multiple of $\nabla_\theta J^{\text{mix}}$ when $\hat{A}^{\text{full}}$ is replaced by $r(\tau)$. Then

1. (**Consistency**) $\hat{g}_B \rightarrow g^* / (1 - \alpha(\theta, d))$ almost surely as $B \rightarrow \infty$. The limit is a positive scalar multiple of the target, so the ascent direction is unchanged and the factor only rescales the step.
2. (**Finite-batch bias**) Componentwise, for any j with $g_j^* \neq 0$,

$$\frac{\mathbb{E}[\hat{g}_{B,j}] - g_j^* / (1 - \alpha(\theta, d))}{g_j^* / (1 - \alpha(\theta, d))} = \frac{1}{B} \left(\frac{\alpha(\theta, d)}{1 - \alpha(\theta, d)} - \rho_j \text{CV}(g_{1,j}) \text{CV}(D_1) \right) + O(B^{-2}),$$

where $\rho_j = \text{Corr}(g_{1,j}, D_1)$. The bias is $O(1/B)$, and its second term is nonzero precisely when the safe count covaries with the masked gradient, which is what happens when the number of safe samples tracks trajectory return, recovery duration or gradient magnitude.

3. (**Exactness without intervention**) If $\alpha(\theta, d) = 0$ then $D_i \equiv 1$, the denominator equals B deterministically, both terms above vanish identically, and Equation (5) is the standard PPO batch mean.

Proof. (i) D_1 is Bernoulli with mean $1 - \alpha(\theta, d) > 0$ and $\mathbb{E}\|g_1\| < \infty$, so the strong law gives $\bar{g} \rightarrow g^*$ and $\bar{D} \rightarrow 1 - \alpha(\theta, d)$ almost surely; the continuous mapping theorem applied to $(x, y) \mapsto x/y$, continuous at $(g^*, 1 - \alpha(\theta, d))$ because the second coordinate is nonzero, gives the claim.

(ii) Write $u = \bar{g} - g^*$ and $v = \bar{D} - (1 - \alpha(\theta, d))$, both mean zero with $\mathbb{E}[v^2] = \text{Var}(D_1)/B$ and $\mathbb{E}[u_j v] = \text{Cov}(g_{1,j}, D_1)/B$. Expanding the ratio in v ,

$$\begin{aligned}\hat{g}_{B,j} &= \frac{g_j^* + u_j}{(1 - \alpha(\theta, d)) \left(1 + \frac{v}{1 - \alpha(\theta, d)}\right)} \\ &= \frac{g_j^*}{1 - \alpha(\theta, d)} + \frac{u_j}{1 - \alpha(\theta, d)} - \frac{g_j^* v}{(1 - \alpha(\theta, d))^2} - \frac{u_j v}{(1 - \alpha(\theta, d))^2} + \frac{g_j^* v^2}{(1 - \alpha(\theta, d))^3} + \dots\end{aligned}$$

Taking expectations kills the two linear terms and leaves

$$\mathbb{E}[\hat{g}_{B,j}] - \frac{g_j^*}{1 - \alpha(\theta, d)} = \frac{1}{B} \left(\frac{g_j^* \text{Var}(D_1)}{(1 - \alpha(\theta, d))^3} - \frac{\text{Cov}(g_{1,j}, D_1)}{(1 - \alpha(\theta, d))^2} \right) + O(B^{-2}).$$

Dividing by $g_j^*/(1 - \alpha(\theta, d))$ and using $\text{Var}(D_1) = \alpha(\theta, d)(1 - \alpha(\theta, d))$, so that $\text{CV}^2(D_1) = \alpha(\theta, d)/(1 - \alpha(\theta, d))$, and $\text{Cov}(g_{1,j}, D_1)/((1 - \alpha(\theta, d))g_j^*) = \rho_j \text{CV}(g_{1,j})\text{CV}(D_1)$ gives the stated form.

(iii) If $\alpha(\theta, d) = 0$ then $D_1 = 1$ almost surely, so $\text{Var}(D_1) = 0$ and $\text{Cov}(g_{1,j}, D_1) = 0$; the expansion in (ii) terminates and every correction term is identically zero, not merely $O(1/B)$. $\square$

Three consequences follow. First, the leading term $\alpha(\theta, d)/((1 - \alpha(\theta, d))B)$ is computable from quantities we already log; at the minibatch size of Table 4 and the whole-run mean recovery rates of Appendix D.4 it is on the order of 10^{-4} to 10^{-3} in relative terms, far below the Monte-Carlo noise of the gradient itself. Second, part (iii) is not asymptotic; on HALFCHEETAH and ANT the recovery rate reaches zero at the printed precision by the end of training, so there the ratio bias is a transient of the early curriculum rather than a property of the converged update; on GO1 it persists at the residual $\alpha(\theta, d) \approx 0.048$. Third, the bias is removable at will. Normalizing by the fixed batch size B instead of $|T_{\mathcal{R}}|$ is exactly unbiased for g^* itself, $(1 - \alpha(\theta, d))$ times the self-normalized limit, at the cost of an update rescaled by the realized safe fraction rather than its mean; we retain the self-normalized form because it keeps the gradient scale stable while $\alpha(\theta, d)$ anneals, the same bias-for-variance trade that motivates the weighted over the unweighted importance-sampling estimator. Early in training, when $\alpha(\theta, d)$ is high, $|T_{\mathcal{R}}|$ can be small: with $m = |T_{\mathcal{R}}|$ safe steps the masked mean has variance on the order of σ^2/m , so a small m inflates the per-update gradient variance. We do not apply variance-control heuristics. The curriculum schedule (Section 5) keeps $|T_{\mathcal{R}}|$ above a working threshold within the first few updates on every environment we evaluate.

The value-loss default also depends on whether the analytic recovery value (Proposition 6) is in use:

- **Analytic V on:** train the critic on *all* timesteps. Recovery-state targets under Proposition 6 carry no Monte-Carlo variance and are exact over the segment; the only residual error is the critic’s own error at the single re-entry state (failure-segment targets are exact), so unmasked is principled and gives the critic dense supervision.
- **Analytic V off:** prefer the masked value loss $L^V(\theta) = |T_{\mathcal{R}}|^{-1} \sum_{t:s_t \in \mathcal{R}} (V_\theta(s_t) - \hat{R}_t)^2$. Otherwise the critic regresses against bootstrapped values that themselves depend on the critic at unsafe states, creating a self-consistency loop.

Our experiments use the analytic-V variant by default. Section 8 ablates this choice.

B.7 Where the changes land in the rollout/update loop

The choices above touch a standard PPO loop at a few points; the list below states where each is implemented.

- **Rollout, recovery branch:** when $s_t \notin \mathcal{R}$, the recovery action is queried, the action is stored in the buffer, $\log \pi_\theta(a_t | s_t)$ is computed at the recovery action, and the step is flagged as recovery-controlled.

- **Analytic-V overwrite:** for each recovery segment, the segment endpoint is identified and the value at *every* segment step is overwritten, at offset j with $\gamma^{\ell-j}V_\theta(s_{t+\ell})$ on success or $\gamma^{\ell-1-j}r_{\text{term}}$ on failure, where ℓ is the segment length (Lemma 11). This happens in the rollout-buffer pass, before GAE.
- **GAE** (standard, no modification): operates on the overwritten value sequence.
- **Compatibility loss:** computes $\log \pi_\theta(a_t^{\text{rec}} | s_t)$ at unsafe steps and applies the chosen gate (σ_k for the hard gate, the signed advantage w_t^{soft} for the soft variant). The resulting loss is added to the PPO surrogate before backpropagation.
- **Masked PG and value loss:** the safe-step indicator masks both the policy-gradient term and (in the learned-critic variant, without the analytic value) the value loss.

B.8 Baseline ports

The two recovery-based baselines in Table 1 are ports into the same on-policy loop as SafeExplorer: both run inside the identical PPO update, share the identical safe-region trigger $s \notin \mathcal{R}(d)$ with the same curriculum on d , and query the identical frozen SAC recovery policy. They differ from SafeExplorer only in how recovery-controlled transitions enter the rollout buffer.

- **Recovery RL port** (Thananjeyan et al., 2021): at recovery steps the buffer keeps the *task policy*’s action, value, and log-probability, the action-relabeling data-handling rule of the original method; recovery-step rewards are zeroed in the learning signal, as for SafeExplorer. The original method treats the recovery as part of the environment dynamics, which is coherent for its off-policy Q-learning objective; the bias corrected in this paper arises only when that relabeling is carried into an on-policy policy-gradient update.
- **Safe Legged port** (Yang et al., 2022): the same relabeling, plus a reward of $r_t - 1$ stored at recovery-triggered steps, the recovery-penalty shaping component of the published method.

What is deliberately *not* ported: Recovery RL’s learned safety critic and off-policy training, and the model-based switching criterion of Yang et al. (2022). Replacing both with the shared distance trigger isolates each method’s data-handling rule under a matched on-policy setup with an identical trigger and identical recovery controller, which is what makes the comparison interpretable. No baseline-specific tuning enters the Table 1 port cells beyond the shared PPO configuration of Appendix C; the bounded per-baseline tuning sweep is in Appendix C.7.

B.9 Pseudocode parameters

Table 3 gives the default values that instantiate Algorithm 1 as run in our experiments.

Table 3: Default parameter values for Algorithm 1.

Symbol in Algorithm 1	Default value
N (total updates)	$\text{total_timesteps}/(\text{num_envs} \times \text{num_steps})$
E (parallel envs)	4
T_{ro} (rollout length)	2048
K (PPO epochs)	10
c_v (value coef)	0.5
c_e (entropy coef)	0
λ_{compat} (compat coef)	10^{-3}
ϵ (PPO clip)	0.2
$\gamma, \text{GAE } \lambda$	0.99, 0.95
r_{term} (terminal reward)	environment terminal-step reward (Appendix C.2)

C Hyperparameters

C.1 Training hyperparameters (shared across all variants)

Every method in Table 1 except the two CMDP baselines is trained with the single PPO configuration of Table 4, so that any difference among them reflects the method rather than per-variant tuning. CPO and PPO-Lagrangian additionally carry the constraint hyperparameters their objective requires (Appendix C.3). The compatibility regularizer λ_{compat} is the only entry active for SafeExplorer alone. All remaining hyperparameters are shared. The per-environment d_{max} values were set by a qualitative criterion, large enough that the recovery policy stops triggering under normal task operation as d approaches d_{max} ; no systematic search over d_{max} was run.

What was and was not tuned. No baseline-specific hyperparameter tuning enters Table 1 or the ablations. The ports and every ablation variant run the shared configuration above, and the CMDP baselines run OmniSafe defaults plus the constraint settings of Appendix C.3. Two facts mitigate this. The ports share the identical PPO loop, trigger, curriculum, and frozen recovery policy with SafeExplorer, and the CMDP baselines receive observation and reward normalization that SafeExplorer does not use, which favors the baselines. A bounded per-baseline tuning sweep is reported separately in Appendix C.7.

Table 4: PPO training hyperparameters for every variant in Table 1. Per-environment d_{max} values are in env-specific task-space units, not comparable across environments.

Parameter	Value
Optimizer	Adam ($\text{eps} = 10^{-5}$)
Learning rate	3×10^{-4}
Anneal LR linearly to 0	yes
Number of parallel envs	4
Rollout length per env	2048
Batch size ($\text{num_envs} \times \text{num_steps}$)	8192
Mini-batch size	batch size / 32
GAE λ	0.95
Discount factor γ	0.99
PPO ratio clipping ϵ	0.2
Update epochs per batch	10
Value loss coefficient c_v	0.5
Max gradient norm	0.5
Entropy coefficient c_e	0
Compatibility regularizer λ_{compat} (SafeExplorer only)	10^{-3} (see Appendix E.7 for sweep)
Initial safe-region radius d_0	0.01 (Go1: 0.05)
Per-env d_{max}	HALFCHEETAH: 2.0; ANT: 0.4; Go1: 0.15
Per-step recovery reward in the learning signal	0 (zeroed; Section 4.3)
Terminal-step reward r_{term}	environment terminal reward (Appendix C.2)

C.2 Terminal-step reward r_{term}

Intermediate recovery-step rewards are zeroed in the learning signal (Section 4.3), but the terminal-step reward that the analytic- V failure branch propagates as r_{term} (Algorithm 1) is the environment’s *own* reward on the terminating step, preserved in the buffer. Concretely: on HALFCHEETAH the terminal step includes an explicit -1 penalty; on Go1 the per-step reward is floored at zero and the healthy bonus vanishes on an unhealthy step, so $r_{\text{term}} = 0$ (a configurable termination-penalty flag exists but is 0 in all main-table runs); on ANT the constant healthy bonus of 1.0 is paid even on the terminal step, so its terminal reward is that bonus minus the step’s control and contact costs.

C.3 CMDP baseline hyperparameters

Both CMDP baselines run on OmniSafe 0.5.x. The constraint cost is 1 on each unhealthy termination and 0 otherwise; time-limit truncations incur no cost, matching the fall-counting convention of Appendix D. The cost limit is 0.05. Shared solver settings: steps per epoch 2048, 10 update iterations, minibatch size 64, target KL 0.02, $\gamma = \gamma_{\text{cost}} = 0.99$, GAE $\lambda = \lambda_{\text{cost}} = 0.95$, actor and critic MLPs of two hidden layers of size 64 with tanh activation, learning rate 3×10^{-4} for both with linear decay, observation normalization on, reward normalization on, cost normalization off. PPO-Lagrangian additionally uses ratio clip 0.2, Lagrange multiplier initialized at 0.001, and multiplier learning rate 0.035. CPO uses the OmniSafe defaults for its trust-region settings plus the cost limit above. No baseline-specific tuning enters Table 1 beyond these settings.

C.4 Per-environment training budget and robot-time equivalents

Table 5: Per-environment training budget, SafeExplorer wall-clock cost, robot-time equivalents computed from the per-environment control timesteps of Appendix D (0.05 s on HALFCHEETAH/ANT, 0.01 s on Go1), and the recovery-pretraining budgets with their robot-time equivalents and fall counts. Pretraining fall counts run to the deployed checkpoints (4M/2M/5M steps), with run-end counts in Appendix D.4; the HALFCHEETAH count is an early-termination count with a legacy-trainer caveat (Appendix D.3).

Environment	Training steps	Wall-clock (4 CPU cores)	Robot-time equiv.	Pretraining steps	Pretraining robot-time	Pretraining falls
HALFCHEETAH	8×10^6	~ 3.5 h	~ 111 h	5×10^6	~ 69 h	168
ANT	8×10^6	~ 5 h	~ 111 h	3×10^6	~ 42 h	17,723
Go1	16×10^6	~ 12 h	~ 44 h	10×10^6	~ 28 h	5,469

The one quantity that does vary across environments is the training budget, which we fix before any rollout begins and report in Table 5. All reported runs train on 4 CPU cores; no GPU is used. The robot-time column states what these budgets would mean on hardware. The equivalents are orders of magnitude beyond the roughly 20 minutes of real-world training reported by Smith et al. (2023b), the comparison behind the framing of Section 1. Algorithmically, the additions over standard PPO (the mask, the per-segment value overwrite, and the gated imitation term) are $O(1)$ buffer operations per step, so GPU throughput follows the underlying PPO implementation. The larger budget on Go1 reflects its 12-DoF action space and longer main-task time-to-converge; the frozen SAC recovery policy (Appendix D.3) does not enter this budget, and is accounted separately in Table 5 and Appendix D.4. Matching all environments at 16M, or repeating Go1 at 8M for a fully matched comparison, is left to follow-up work.

C.5 Network architecture

Within each run, the network that PPO optimizes is the standard CleanRL actor/critic: the actor and critic are independent MLPs with two hidden layers of size 64 and tanh activation. The actor outputs a mean vector and a learnable global $\log \sigma$ for a diagonal-Gaussian policy. Layer weights are initialized orthogonally. The critic’s output layer uses gain 1.0 and the actor’s mean output uses gain 0.01.

C.6 Curriculum schedule

The remaining quantity that changes during a run is the safe-region radius, which is not held fixed but annealed along the linear schedule of Section 5, interpolating $d_0 \rightarrow d_{\text{max}}$ over the $N = \text{total_timesteps}/(\text{num_envs} \times \text{num_steps})$ training updates. Appendix E.9 ablates the schedule shape (linear / log / step / constant) on ANT; the headline results all use the linear schedule.

C.7 Bounded per-baseline tuning sweep

We ran a bounded per-baseline tuning sweep for the port baselines on all three environments, one defining hyperparameter per baseline (the trigger radius d_{max} for the Recovery RL port, the shaping coefficient for

the Safe Legged port), three values bracketing the default, at the full body budget. Per-cell seed counts are printed in Table 6. The HALFCHEETAH and ANT cells are ten seeds each.

Table 6: Tuned-best and default port cells (train falls / train reward, body protocol, seed count per cell). SafeExplorer reference cells are the 10-seed batch of Table 1. Radius and hyperparameter values are printed per cell. The GO1 rows and every reference cell use the last-10% reward window of Table 1; the HALFCHEETAH and ANT port rows use the last-5% window this sweep was harvested at, a difference far inside seed noise on every cell measured both ways.

Env	Cell (hyperparameter, operating point)	Seeds	Falls	Reward
HALFCHEETAH	Recovery RL port, default radius ($d_{\max}=2.0$)	10	2852 ± 4883	1500
HALFCHEETAH	Recovery RL port, doubled radius ($d_{\max}=4.0$), tuned best	10	1557 ± 2905	3544
HALFCHEETAH	Safe Legged port, default coefficient ($c=1.0$)	10	943 ± 1582	3415
HALFCHEETAH	Safe Legged port, largest coefficient ($c=3.0$), tuned best	10	738 ± 932	4427
HALFCHEETAH	SafeExplorer (reference, $d_{\max}=2.0$)	10	15.3 ± 14.7	3855 ± 958
ANT	Best tuned port cell (Recovery RL, halved radius $d_{\max}=0.2$)	10	15628 ± 7280	103
ANT	SafeExplorer (reference, $d_{\max}=0.4$)	10	866 ± 718	2698 ± 237
GO1	Recovery RL port, halved radius ($d_{\max}=0.05$)	10	1566 ± 1033	3088 ± 817
GO1	Recovery RL port, default radius ($d_{\max}=0.1$)	10	2132 ± 2164	2895 ± 753
GO1	Recovery RL port, doubled radius ($d_{\max}=0.2$)	10	2963 ± 1911	3808 ± 911
GO1	Safe Legged port, $c=0.3$	9	3651 ± 1792	2056 ± 366
GO1	Safe Legged port, $c=1.0$ (default)	10	4573 ± 1792	1977 ± 335
GO1	Safe Legged port, $c=3.0$, tuned best	10	2314 ± 777	2513 ± 243
GO1	SafeExplorer (reference, $d_{\max}=0.15$)	10	830 ± 216	5509 ± 155

Tuning does improve the ports in places, but no tuned cell Pareto-dominates SafeExplorer; at ten seeds no tuned Recovery RL cell exceeds SafeExplorer’s mean reward, and that port’s best case, twice the trigger radius on HALFCHEETAH, reads 1557 ± 2905 falls at reward 3544. On HALFCHEETAH one tuned cell sits nominally above SafeExplorer’s 10-seed reward, the Safe Legged port at its largest coefficient (4427 against 3855 ± 958 , a gap inside one seed standard deviation at ten seeds per cell), while sitting at roughly $48\times$ SafeExplorer’s falls (738 ± 932 against 15.3 ± 14.7), so we claim no reward lead in either direction for that cell. On GO1, SafeExplorer leads on both axes in every one of the six tuned cells. The reward lead is interval-separated. SafeExplorer reads 5509 ± 155 against a best tuned cell of 3808 ± 911 , and across all 59 tuned runs none reaches the SafeExplorer mean and none reaches even its worst seed (5214; the highest tuned run is 5196). The falls lead is weaker and is not uniform across the two ports. Against all three Safe Legged cells it separates (3651 ± 1792 , 4573 ± 1792 and 2314 ± 777 against 830 ± 216 ; Welch $p = 0.001$, 0.0001 and 0.0001). Against Recovery RL it separates only at the doubled radius (2963 ± 1911 , $p = 0.012$) and stays unresolved at the two tighter $d_{\max}$ values (1566 ± 1033 and 2132 ± 2164 , $p = 0.16$ and 0.15), so on those two cells we report an ordering of means and not a separated difference. At three seeds this sweep put the closest Recovery RL cell at 1335 ± 761 ; at ten seeds it reads 1566 ± 1033 , and the comparison stays unresolved. On ANT no hyperparameter value makes either port functional. Halving $d_{\max}$ does not reduce the Recovery RL port’s GO1 falls by a detectable amount; against the sweep’s own default arm (2132 ± 2164 at ten seeds, per-seed range 523 to 6916) the difference is noise.

D Environment and Recovery-Policy Details

D.1 Environment summary

We evaluate on three MuJoCo locomotion environments that differ in dimensionality and in how the safe region is sensed. Table 7 lists, for each one, the degrees of freedom, the observation and action dimensions, the indicator that the recovery policy uses to decide whether the agent has left the safe region $\mathcal{R}$, and the termination predicate that defines a fall. The indicator is a function of torso or base height together with tilt to the nominal pose, the signal the recovery policy is trained to restore.

Table 7: Per-environment dimensions, the safe-region indicator the recovery policy uses, and the termination predicate that defines a fall.

Environment	DoF	Obs dim	Action dim	Safe-region indicator	Fall (termination) iff
HALFCHEETAH	6	17	6	torso height z + tilt to nominal	$z \notin [-0.5, 0.5]$ or $ \theta_{\text{pitch}} > 1.57$
ANT	8	27	8	torso height z + tilt to nominal	$z \notin (0.3, 1.5)$ or non-finite state
Go1	12	49	12	base z + base tilt (roll, pitch) to nominal	$z \notin (0.22, 0.65)$ or $ \text{roll} $ or $ \text{pitch} > 60^\circ$

Termination, horizon, and fall counting. The HALFCHEETAH termination predicate is a custom addition; the stock Gymnasium HALFCHEETAH never terminates, which is why our fall counts on HALFCHEETAH are nonzero. The terminating step also carries a -1 reward, the only other deviation from the stock HALFCHEETAH reward function; at most one unit per episode against returns in the thousands, it is immaterial to every comparison, but we state it because the HALFCHEETAH success bar is imported from the stock environment (Appendix E.2). Our ANT departs from stock in two respects of its own: the healthy range is $(0.3, 1.5)$ rather than the stock $(0.2, 1.0)$, and the control-cost weight is 0.05 rather than the stock 0.5 . The first terminates sooner on a collapse and the second penalizes large actions less, so they act in opposite directions on achievable return; we state them because Gymnasium’s registered ANT score of 6000 is likewise not validated for this variant. All three environments run under a 1000-step episode time limit. The control timestep is 0.05s on HALFCHEETAH and ANT (frame skip 5 at a 0.01s simulation step) and 0.01s on Go1 (frame skip 5 at a 0.002s simulation step), so the horizons are 50s and 10s of simulated time respectively. A *fall* is an unhealthy termination as defined in Table 7; time-limit truncations are never counted as falls, in training and evaluation alike.

D.2 Safe-region distance and curriculum

The membership test is $s \in \mathcal{R}(d)$ iff $D(s) \leq d$, with an environment-specific distance D computed from the base height and orientation:

- HALFCHEETAH: $D(s) = \|(z, \theta_{\text{pitch}})\|_2$ against the nominal $(0, 0)$, with z in meters and pitch in radians, unweighted.
- ANT: $D(s) = \|(z - 0.75, \text{tilt})\|_2$, where $\text{tilt} = \frac{\pi}{180} \arccos(\text{axis}_z) \cdot \phi / \pi$ with $\phi = 2 \arccos(q_w)$, a rescaled axis-angle deviation from upright (q_w is the scalar quaternion component and axis_z the vertical component of the rotation axis). This scaling strongly downweights orientation relative to height on ANT; it is an implementation choice that we state plainly and keep for fidelity with the trained runs.
- Go1: $D(s) = \|(z - 0.3, \text{roll}, \text{pitch})\|_2$, with z in meters and roll and pitch in radians.

Because each D mixes meters and (rescaled) radians in a single Euclidean norm, d and d_{max} carry the same mixed units and are therefore not comparable across environments, the reason Table 4 reports them as env-specific values. During training, d follows the linear curriculum $d = d_0 + \frac{u-1}{N} d_{\text{max}}$ at update u of N updates (Algorithm 1), with the per-environment (d_0, d_{max}) of Table 4.

D.3 Recovery-policy training

For each environment, the recovery policy π^{rec} is a separately pre-trained SAC actor, trained on the same MuJoCo dynamics in a recovery-flavored configuration: the forward and velocity-tracking task terms of the environment reward are zeroed, so the remaining reward favors surviving and staying near the nominal pose. The shared recipe is CleanRL SAC with twin 256-256 Q networks, a 256-256 tanh-squashed Gaussian actor, replay buffer of 10^6 transitions, $\gamma = 0.99$, batch size 256, and an auto-tuned entropy coefficient; recovery-training episodes are capped at 250 steps, and at deployment we use the actor’s deterministic tanh-mean action. Recovery training is *not* reset-free. At every episode reset, the initial state is aggressively randomized to mimic the failure modes a learning task policy will produce.

Initial-state randomization (per env reset):

- **Joint positions:** sampled uniformly over the full mechanical joint range (or a large subset). For ANT, each joint has env-specific bounds (e.g., ± 0.52 to ± 1.23 rad from nominal). For GO1, all 12 joint positions are drawn from the full actuator range.
- **Base height (and optionally x, y):** base z is randomized over a range that includes low and near-fall configurations (e.g., $z \in [0.22, 0.45]$ for GO1).
- **Base orientation:** the base quaternion is randomized to produce varied roll, pitch, and yaw. For ANT, a random quaternion is generated and constrained so the forward vector has non-negative z (“face up”). For GO1, roll and pitch are drawn in $\pm\pi/10$ and yaw in $[0, 2\pi]$, again with a face-up constraint.
- **Velocities:** initialized with small noise around zero.

Per-environment training and cost. On HALFCHEETAH the recovery policy was trained on the stock environment with the forward reward disabled, for 5M environment steps; the deployed checkpoint is the 4M-step snapshot. This policy was trained with an earlier revision of the trainer, so we report its checkpoint provenance rather than a reconstructed reward. That revision also terminated episodes early on a predicate that is not preserved in the repository; the 168 pretraining falls of Appendix D.4 count those early terminations, which is why the count is nonzero although the stock environment never terminates (Appendix D). On ANT it was trained for 3M steps (2M-step snapshot deployed; policy learning rate 10^{-4} , Q learning rate 3×10^{-4} , target-smoothing $\tau = 0.002$) on the recovery-flavored environment, whose effective reward is the healthy bonus minus control-magnitude and contact costs, so the ANT recovery is torque-limited. On GO1 it was trained for 10M steps (5M-step snapshot deployed; $\tau = 0.005$, Q learning rate 10^{-3} , policy learning rate 3×10^{-4}) with the velocity-tracking terms zeroed; the effective reward is the healthy indicator minus quadratic pose, height, and joint-deviation costs, floored at zero. Recovery pre-training therefore costs 5M (HALFCHEETAH), 3M (ANT), and 10M (GO1) environment steps. These steps are not included in the training budgets of Table 5’s first column, but are reported there alongside them, and the falls incurred during recovery pre-training are counted from the pretraining logs in Appendix D.4. The cost is nonetheless bounded: at most 62.5% of a single 8M-step HALFCHEETAH or ANT training run (5M of 8M) and 62.5% of the 16M-step GO1 budget (10M of 16M), which bounds the sense in which the recovery policy is cheap to obtain.

Per-environment recovery quality. How well this training regime works depends on the environment. Empirically, the HALFCHEETAH and GO1 recoveries succeed at stabilization with high probability across the randomized initial states. The ANT recovery policy does not, producing the unreliable-recovery regime that is the most informative axis of variation in our results (see Section 7). We attribute this to the recovery policy’s reward design rather than to undertraining: the control-magnitude penalty ($0.5\|a\|^2$ in the ANT reward) caps how much torque the policy can apply, and the harder 3D ANT stabilization needs exactly those aggressive corrections. Relaxing that penalty or adopting a stronger recovery class for ANT, and confirming the attribution with a longer-training control, is left to follow-up work. We keep the imperfect ANT recovery deliberately rather than retraining it. On hardware the recovery controller will be imperfect for reasons outside the learner’s control, actuator and torque limits among them, and the regime where recovery sometimes fails is the one in which the outcome-gated imitation term becomes visible at all (the 20.9-point survival gap in Table 8, and the gated variant is the only one to reach the ANT success bar on every seed). Retraining improves one instance; the method is about training safely with whatever recovery controller one has. The practitioner guidance we state instead is a catch-rate screen. Before any task training is spent, score a candidate recovery offline by its single-shot catch rate from doomed handover states collected at the intended trigger band, a cheap measurement that in our WALKER2D study predicts the in-loop outcome where the recovery’s own training return does not (Appendix E.18). Because the screen’s handover states are doomed by construction, its catch rate is a stricter quantity than the operational survival rates of Table 8, which average over every triggered exit, and the two percentages are not comparable.

Recovery quality on the operational distribution. The randomized-reset measurement above grades the recovery on states the designer samples, not on the states the learning policy actually drives it into (which, unlike the face-up-constrained reset distribution, can include face-down configurations), so the operational grade is, if anything, the harder of the two. To measure the latter, during a trained run we log every

first exit from $\mathcal{R}$ (the state at which recovery is triggered), then replay the SAC recovery from each logged state and record the fraction that *survive*, meaning the recovery completes the segment without an episode termination. We do this for SafeExplorer and for the masked, analytic- V ablation variant of Section 8, which shares the recovery policy and the curriculum with SafeExplorer and differs from it only by the hard outcome gate $C_\theta(\tau)$ (Equation (10)). Table 8 aggregates all logged triggers of one representative run per cell. Two effects stand out. First, SafeExplorer leaves $\mathcal{R}$ between $2.1\times$ and $4.2\times$ less often, so it calls on the recovery far less. Second, on ANT, where the recovery policy is unreliable, the exits SafeExplorer produces survive 91.3% of the time against 70.4% for the ungated variant, a gap of 20.9 points. On HALFCHEETAH and GO1, where the recovery policy is reliable, both variants survive at essentially the same rate (about 100% and 97%). The ANT gap offers an operational reading of the +2524 reward the hard gate adds on ANT in Section 8. The gate biases the main policy toward the part of state space where the weak ANT recovery policy actually succeeds. The trigger counts in Table 8 place the binomial 95% confidence interval on each survival rate within ± 0.5 points. These intervals reflect sampling over triggers within one run, not variation across training seeds, which we do not characterize here; the table is therefore a single-run illustration rather than a multi-seed estimate.

Table 8: Recovery quality on the *operational* trigger-state distribution. We replay the SAC recovery from every logged first exit from $\mathcal{R}$ of one representative trained run per cell. “Exits” counts the logged recovery triggers; “Survived” is the percentage of those exits the recovery completes without termination. The ungated variant is the masked, analytic- V variant of Section 8.

Environment	Exits (triggers)		Survived (%)	
	SafeExplorer	ungated	SafeExplorer	ungated
HALFCHEETAH	80,546	338,671	99.8	100.0
ANT	16,366	35,150	91.3	70.4
GO1	94,212	255,881	97.0	97.4

D.4 End-to-end safety accounting

All reported safety results are conditional on an already available recovery controller; this subsection accounts for that condition.

Pretraining falls. Counted from the pretraining logs up to the deployed checkpoints (4M steps on HALFCHEETAH, 2M on ANT, 5M on GO1), the counts are 168 falls on HALFCHEETAH (an early-termination count from a legacy revision of the trainer, the caveat stated in Appendix D.3), 17,723 on ANT, and 5,469 on GO1. The two WALKER2D recovery checkpoints incurred 586 and 1,027 pretraining falls (the winning WALKER2D configuration uses the second). Recovery pretraining is standard episodic RL with resets from aggressively randomized initial states, run in simulation or, on hardware, in a controlled rig, so the training falls are not eliminated end to end; they are moved into a phase where failures are cheap and controlled.

Intervention statistics. From the existing training logs and the replay machinery behind Table 8, the recovery engages on a whole-run mean of 7.8% (HALFCHEETAH), 27.6% (ANT), and 12.9% (GO1) of training steps, annealing to zero on HALFCHEETAH and ANT and to about 5% on GO1; mean recovery-segment durations are 4.9, 159, and roughly 20 steps respectively; and failed-recovery rates on the operational trigger distribution are 0.21%, 8.72%, and 2.96% per segment (the GO1 rate is an upper bound), consistent with the recovery survival rates of Table 8. Two log-coverage caveats apply. The dense GO1 logs cover the first 6.2M steps, so its whole-run engagement mean is an estimate and per-quartile durations are not available there; and per-segment length histograms are not recoverable from the existing logs on any environment, so we report exact mean durations per training phase rather than full distributions.

End-to-end ratios and the reuse curve. With the recovery pretraining falls charged in full to a single training run, the end-to-end fall reduction versus PPO on the Table 1 cells is $21.3\times$ on HALFCHEETAH, $2.0\times$ on ANT, and $2.8\times$ on GO1; counted to the ends of the executed pretraining runs rather than to the deployed checkpoints (199, 21,425, and 5,798 falls), the same charge reads $18.2\times$, $1.7\times$, and $2.6\times$. Because the frozen checkpoint is shared by every recovery-based run, the acquisition cost amortizes. As a function of

the number n of runs sharing the checkpoint, the checkpoint-basis reduction rises from its single-run charge above to $200\times$ (HALFCHEETAH), $29\times$ (ANT), and $18\times$ (GO1) at $n = 40$, the number of recovery-using runs in this paper’s own study, approaching the training-only ratios of Table 1 with further reuse.

D.5 Go1 environment specification

The GO1 observation is 49-dimensional, in order: base linear velocity scaled by 2.0 (3), base angular velocity scaled by 0.25 (3), base height z (1), roll, pitch, yaw (3), commanded velocity scaled by 2.0 (3), joint positions minus their defaults (12), joint velocities scaled by 0.05 (12), and the previous action (12); the vector is clipped to ± 100 . The action is 12 absolute joint-position targets driving MuJoCo position servos with proportional gain $k_p = 20$, joint damping 1.0, and an actuator force range of ± 23.7 N m (± 35.55 N m at the knees). The commanded velocity (v_x, v_y, ω_z) is resampled once per episode, each component uniform in $[-0.5, 0.5]$. The reward is $4.0 \exp(-\|v_{xy}^{\text{cmd}} - v_{xy}\|^2/0.25) + 2.0 \exp(-(\omega_z^{\text{cmd}} - \omega_z)^2/0.25)$, plus a healthy bonus of 1.0 and a feet-air-time term of weight 1.0, minus costs with weights: torque 2×10^{-4} , vertical velocity 2.0, xy angular velocity 0.05, action rate 0.001, joint-limit violation 0.01, joint acceleration 1.25×10^{-9} , orientation 0.4, height $4.0|z - 0.3|$, and joint deviation 0.1; the total is floored at 0.

D.6 Custom XML modifications

Supporting the recovery training and value conventions above requires three changes to the environment definitions. Our custom MuJoCo XMLs extend the standard Gymnasium MuJoCo environments with: (i) explicit joint-range attributes used by the recovery policy’s randomized-reset training (Appendix D.3), (ii) a configurable flag that zeros the training reward outside $\mathcal{R}$ (the learning-signal convention of Section 4.3; reported returns are unmodified), and (iii) for GO1, the `mujoco_menagerie` Unitree Go1 model (Zakka et al., 2022) with a custom 12-DoF position-actuator configuration and a velocity-tracking reward adapted from `legged_gym` (Rudin et al., 2021).

E Additional Results

Run-batch provenance. The sensitivity studies in this appendix come from dedicated sweep batches run independently of the Table 1 runs; each caption states its batch, operating point, and seed count. Cumulative falls are heavy-tailed across seeds, so independent batches at identical settings can differ noticeably in absolute falls while preserving the orderings the text relies on. Where a table’s operating point differs from the body protocol of Section 6, specifically the wider $d_{\text{max}} = 3.2$ on HALFCHEETAH and 0.56 on ANT used by the sensitivity sweeps, the caption says so; within-table comparisons remain matched. The sweeps at these wider d_{max} values measure the effect of the trigger radius; they are not a second answer to the body-protocol question. On GO1 the sensitivity radius is wide enough that the trigger almost never fires, so the method reduces to PPO by construction and the reduction there is $1.6 \times [1.4, 1.8]$, a property of the radius and not of the estimator.

E.1 Headline comparison: reliable interquartile-mean intervals

This appendix supplies the evidence behind every claim made in the body, ordered as a descent from the headline numbers to the design choices that produce them, the robustness checks that stress them, and the theory and baselines that justify them. We begin with the headline. Figure 2 gives the per-metric reward and total-falls intervals for the six comparison-set methods, complementing the single-bar falls-to-success view (Figure 3). It shows reliable interquartile means with 95% stratified-bootstrap confidence intervals (Agarwal et al., 2021), drawn as ranked intervals so the falls and reward orderings read directly.

E.2 Fixed-bar metric: fine print and per-seed detail

This subsection carries the fine print of the metric of Section 6 (the threshold profile itself is Figure 1, in the body), with the per-seed detail behind Section 7.1 and Table 1.

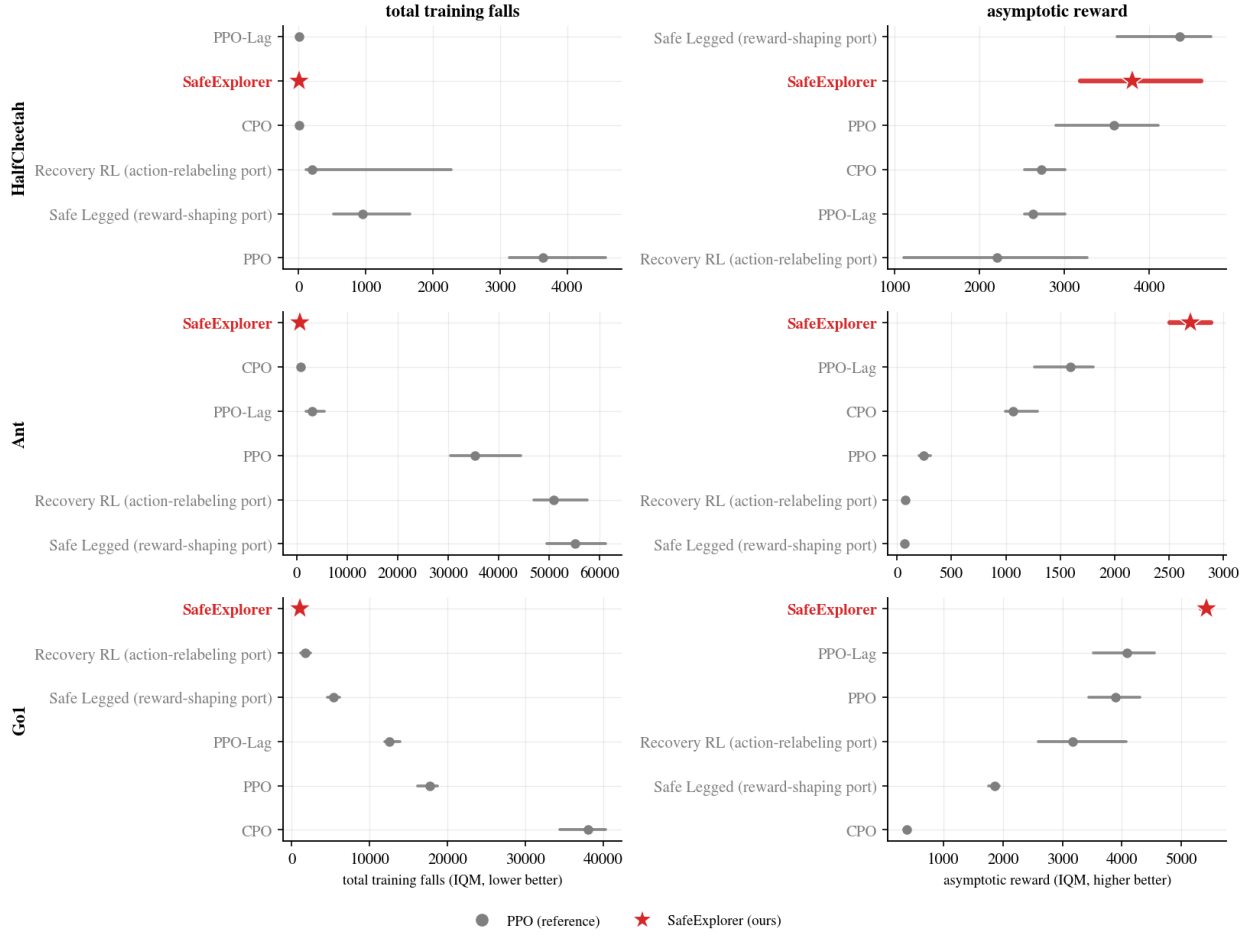


Figure 2: reliable interquartile-mean intervals of reward and total falls for the six methods in Table 1, with 95% stratified-bootstrap confidence intervals; a per-metric companion to the single-bar falls-to-success view (Figure 3).

Metric fine print. The HALFCHEETAH bar (4800) is imported from the stock Gymnasium score; the ANT and GO1 bars (2250 and 4500) are set by us. Neither our HALFCHEETAH nor our ANT is the stock Gymnasium environment (Appendix D), so the registered scores of 4800 and 6000 are transplanted rather than validated here; 6000 in particular sits above anything any method reaches, so adopting it would leave every method unsolved, and GO1 has no registered score at all. Since none of the three is externally validated for our variants, Figure 1 reports the metric at seven thresholds per environment spanning the range in which any method reaches it; the three bars anchor the numbers tabulated here. That externality carries one qualification. The value 4800 is the conventional solved score for the *stock* Gymnasium HALFCHEETAH, which never terminates; our variant adds the termination predicate of Appendix D, so episodes can end before the 1000-step limit and the same score is strictly harder to reach here. No method in Table 1 reaches it in mean, whereas PPO clears it routinely on the stock environment. We keep the bar unchanged rather than recalibrate it to this variant, because changing a threshold after observing that it is unflattering is precisely the failure a fixed bar is adopted to prevent. The consequence is that HALFCHEETAH is measured against a deliberately conservative anchor, which is why its solve counts are low for every method and why the threshold profile, not this bar, is the discriminating instrument there. On ANT and GO1, where we set the bar, the outcome is saturated (SafeExplorer at 10 of 10 seeds, every baseline censored on all 10 on ANT and on at least half of its seeds on GO1), so no bar in a wide interval changes the comparison. On HALFCHEETAH the imported bar sits above every method’s mean final reward and cannot discriminate. Falls-to-success is the cumulative training-fall count at the first logged point where a trailing mean of the

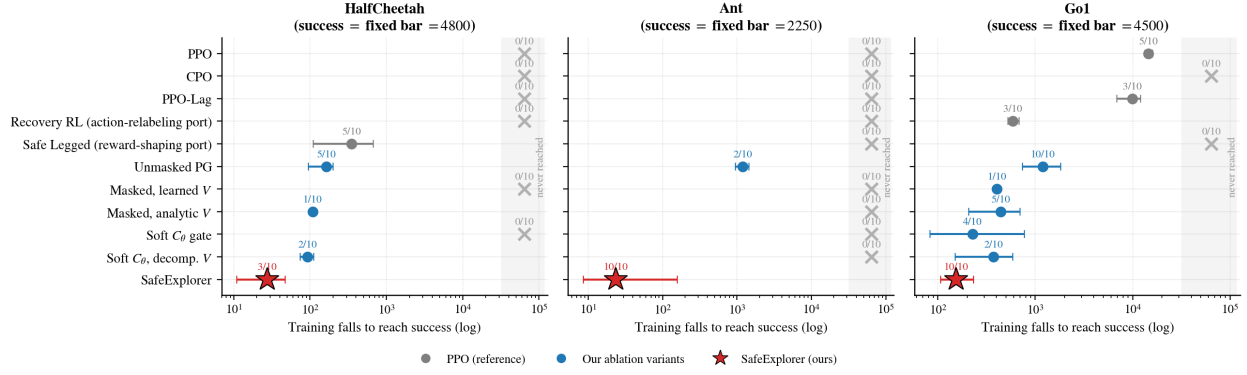


Figure 3: The single-bar view (4800 on HALFCHEETAH, 2250 on ANT, 4500 on Go1), carrying every ablation-ladder variant, which Figure 1 omits for legibility. The 10-seed batch of Table 1, every ladder variant likewise evaluated at ten seeds (the unmasked variant on its own batch, Table 10); interquartile mean over succeeding seeds with 95% bootstrap confidence intervals (Agarwal et al., 2021); labels show k/n , never-reachers censored and marked at right; log axis, fewer falls is better. Baselines gray, ladder variants (Section 8) blue, SafeExplorer the red star; on HALFCHEETAH the ranking claim is carried by the threshold profile.

reward curve, window clip($N/20, 3, 25$) logged points (about 5% of the series), reaches the bar and stays at or above it for the whole of that window; the fall count at the crossing is obtained by linear interpolation of the monotone cumulative-falls curve. Dropping the persistence requirement, so that a single crossing counts, would lower SafeExplorer’s ANT falls-to-success from 24 to 19, its Go1 figure from 154 to 120 and its HALFCHEETAH figure from 27.3 to 27.0, while raising several baselines’ solve counts (Go1 PPO from 5/10 to 10/10, the HALFCHEETAH Safe Legged port from 5/10 to 6/10). We keep the persistent rule because a single crossing of a trailing mean can be produced by one noisy evaluation. Of the two natural treatments for never-solving seeds, censoring and assignment of the full budget, we take censoring. The metric is denominated in falls rather than steps, so assigning a failed run its full-budget fall count would score a policy that never solves and never falls as the safest method present; a censored lower bound carried with the k/n solve count cannot.

Per-seed detail. On Go1 SafeExplorer solves 10/10 at 154 falls, with every baseline’s censored lower bound $10\times$ to $246\times$ higher. On ANT SafeExplorer reaches the 2250 bar on 10 of 10 seeds; each of the five baselines is censored on all 10; and the unmasked variant reaches it on 2 of 10. The remaining four ladder variants are censored on all 10 seeds, their ANT rewards far below the bar (Table 10). On HALFCHEETAH SafeExplorer reaches the Gym bar on 3 of 10 seeds, the Safe Legged port on 5 of 10, and the unmasked variant on 5 of 10; neither difference from SafeExplorer is significant (Fisher exact $p = 0.65$ for both). Among seeds that solve, SafeExplorer pays 11 to 47 falls against 88 to 200 for the unmasked variant and 80 to 698 for the Safe Legged port, a complete separation from the port with probability $1/56$ under seed exchangeability; the HALFCHEETAH threshold-sweep ranking is a ranking of censored lower bounds, which is why it is not headlined as a fewest-falls claim.

Provenance of the reduction ratios. All three ratios take both sides from the 10-seed batch of Table 1, reading HALFCHEETAH $255\times$ [148, 491], ANT $43\times$ [27, 79] and Go1 $21\times$ [18, 25] (95% bootstrap intervals), and the threshold sweep brackets each bar, its lowest point between $0.3\times$ and $0.5\times$ of the bar and its highest between $1.1\times$ and $1.3\times$, across the three environments. The every-seed dominance statement holds in all three environments. Of the bolded leads in Table 1, two are nominal with overlapping intervals, the Safe Legged port’s HALFCHEETAH reward over SafeExplorer’s and CPO’s ANT falls over SafeExplorer’s.

E.3 Per-environment learning curves

The interval plot summarizes the endpoints; the learning curves show how each method gets there. Figure 4 traces reward and cumulative falls across the full training run, the trajectory view of the headline endpoints just reported.

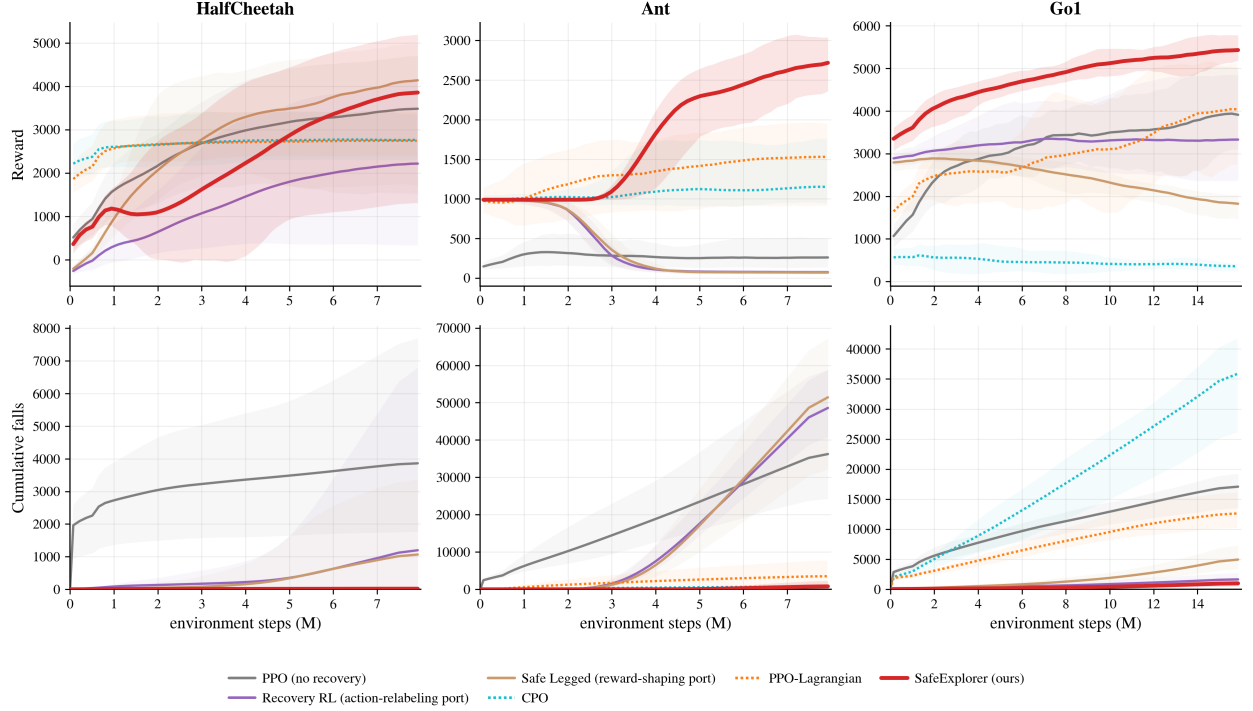


Figure 4: Per-environment learning curves: rows are episodic reward and cumulative training-time falls; columns are HALFCHEETAH, ANT, GO1. Solid = mean across seeds, shaded = [min, max] envelope; curves smoothed with a moving average for display. Recovery-rate curves are deferred to Appendix E.15.

E.4 Additional ablations

Having established what the headline shows, we now ask which design choices produce it, starting from the component ablation. Section 8 analyzes that ablation in the body. Figure 5 gives its reward-vs-falls view, the trade-off complement to the falls-to-success ranking of Figure 3; here we also report the remaining variants. The two threads opened here are picked up in turn below: the soft-gate variants (Appendix E.6) help marginally on ANT and hurt on HALFCHEETAH, and the λ_{compat} sweep is in Appendix E.7.

The GO1 ladder cells of Figure 5 are read at the matched 13.04M budget, the largest at which all ten seeds of each cell are comparable.

E.5 Crossed ablation cells: masking $\times$ analytic value $\times$ outcome gate

Table 9 reports the four crossed component cells in two blocks whose protocols differ and must not be conflated; each row carries its protocol, batch, and seed count.

Recovery-disabled block (end-of-training recovery-disabled evaluation reward, wider sensitivity d_{max} values). The gated learned-critic variant (masked + compatibility imitation with a learned critic in place of the analytic value) is within seed noise of SafeExplorer on ANT and HALFCHEETAH and trails it on GO1, the one environment where the analytic value adds on top of the gate. On HALFCHEETAH and ANT the outcome-gated imitation, not the analytic value, does most of the work, and the analytic value without

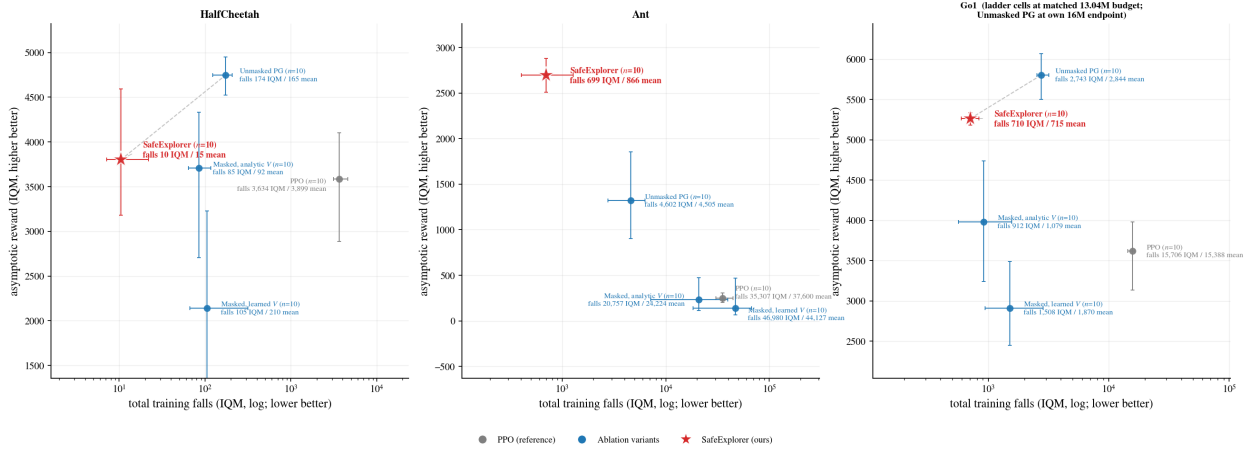


Figure 5: Ablation ladder on the reward-vs-falls plane, the 10-seed batch of Table 1, single reward pipeline (its last-10% convention). Markers are reliable interquartile means with 95% stratified-bootstrap confidence intervals (Agarwal et al., 2021); per-point seed counts are printed (the unmasked variant on its own 10 seeds, Table 10). Total training falls on a log axis, upper-left is better. SafeExplorer (red star) is Pareto-non-dominated in every environment, under both the IQM and the mean convention, including against the CMDP baselines of Table 1.

the gate does not improve on the masked-only variant’s evaluation reward on any environment at this protocol. At the body operating point, however, the GO1 gate-off contrast runs the other way (Table 17), so the analytic value’s standalone payoff is environment- and radius-dependent rather than uniformly absent. The components interact rather than add. A second independent 10-seed batch at the same protocol replicates every shared cell within one standard deviation (Appendix E.14).

Unmasked block (8M steps, body operating point, train-side). The two cells, unmasked + analytic value and unmasked + compatibility imitation, were run with a flag-gated lift of the Theorem 1 mask (the masked setting is bit-exact with the unmodified trainer, so the cells differ from their controls in the mask alone). Both collapse in expectation, and the failure is bimodal rather than uniform across seeds. On HALFCHEETAH the analytic-value cell ends with 9 of 10 seeds at negative return and its one partial recovery below every seed of the same-batch masked control; the imitation cell ends with 6 of 10 seeds negative, its best recovered seed still 2.3× below the same-batch masked control’s worst; so no unmasked seed reaches either matched control’s mean. The falls column points the same way but only as a direction; its ten-seed standard deviations exceed its means, and on HALFCHEETAH a collapsed policy barely moves and therefore barely terminates, so a low fall count there records inactivity rather than safety, which is why reward is the unambiguous axis. On GO1 the unmasked cells show the same bimodal shape, one analytic-value seed partially recovering and no imitation seed doing so, with falls catastrophic in both cells. With the same-batch masked controls, all four masked-versus-unmasked comparisons on GO1 separate, by 82× and 164× on falls and 24× and 59× on reward, with no overlap at one standard deviation on either axis. HALFCHEETAH separates on all four on means as well, whereas ANT separates on the imitation row only, its masked analytic-value control being the high-variance outlier. Against SafeExplorer’s Table 1 GO1 cell each single masked component reaches lower reward, so here too the components interact rather than add. The mechanism is the conditioning failure the IS appendix documents; recovery actions lie far in the tail of the task policy, so with the mask removed the likelihood ratios at recovery steps explode and the policy-gradient loss reaches 10⁷ against roughly 10⁻³ in the masked runs; the blowup develops over the first 10⁵ to 6 × 10⁵ steps rather than at initialization, so it is a training-dynamics failure rather than a startup one. The ladder’s Unmasked-PG variant (Table 10) is a *different* estimator; it keeps the log-probability of the action the main policy sampled while relabeling the action, so its recovery-step ratios collapse toward zero instead of exploding and it trains normally; the two cells here take the analytic-value and imitation estimator literally and delete only the Theorem 1 mask.

Table 9: Crossed component cells in two protocol blocks at 10 seeds per cell: recovery-disabled (end-of-training recovery-disabled evaluation reward, wider sensitivity $d_{\max}$, 3.2 HALF-CHEETAH / 0.56 ANT / 2.2 Go1) and train-side (reward and falls at the body operating point, the four crossed cells and their same-batch masked controls). Spreads are sample standard deviations.

Cell	Protocol	HALF-CHEETAH	ANT	Go1
Masked + gated imitation, learned V	rec-disabled eval reward, sens. $d_{\max}$	4658 ± 1084	2950 ± 623	4106 ± 901
SafeExplorer (masked + gate + analytic V)	rec-disabled eval reward, sens. $d_{\max}$	4859 ± 336	2801 ± 487	5173 ± 618
Unmasked + analytic V	train-side reward, body $d_{\max}$	-552 ± 847	382 ± 243	193 ± 428
Unmasked + analytic V	train-side falls, body $d_{\max}$	5267 ± 10914	13655 ± 24718	117340 ± 57749
Unmasked + gated imitation	train-side reward, body $d_{\max}$	-233 ± 1377	395 ± 231	86 ± 52
Unmasked + gated imitation	train-side falls, body $d_{\max}$	6621 ± 11466	24584 ± 38407	102435 ± 41216
Masked + analytic V	train-side reward, body $d_{\max}$	2608 ± 1090	460 ± 502	4641 ± 1155
Masked + analytic V	train-side falls, body $d_{\max}$	243 ± 384	30321 ± 34149	1436 ± 1136
Masked + gated imitation	train-side reward, body $d_{\max}$	4278 ± 633	2866 ± 403	5049 ± 537
Masked + gated imitation	train-side falls, body $d_{\max}$	72 ± 184	662 ± 631	626 ± 284

E.6 Soft compatibility-gate variants

The body ablation (Section 8, Figure 3) analyzes the clean ablation ladder Unmasked PG $\rightarrow$ masked learned $V \rightarrow$ analytic $V \rightarrow$ hard $C_\theta(\tau)$ (SafeExplorer); Table 10 tabulates its full numbers. The two soft-gate variants, a soft re-entry-advantage gate w_t^{soft} (Equation (16)) applied on the analytic- V model, and the same soft gate applied on a decomposed value, are negative results. They help marginally on ANT and hurt on HALF-CHEETAH, and never approach SafeExplorer’s safety. We report their full numbers here and omit them from the body ablation to keep it focused on variants that isolate a single design choice. Table 10 places them alongside the ladder variants so the off $\rightarrow$ soft $\rightarrow$ hard progression is visible; all rows are the 5-seed body batch except the SafeExplorer row (the 10-seed batch of Table 1) and the Unmasked PG cells (their own 10 seeds). The three lowest Go1 falls cells lie within seed noise of one another and span two batches, so that column orders means rather than separating them. The falls-to-success views (Figure 3, Figure 5) evaluate every ladder variant at ten seeds; this table’s reward and falls cells keep the batches stated above.

Table 10: Soft-gate ablations against the full ladder: Unmasked PG (the biased no-masking baseline of Section 8, added here so all variants appear in one table), masked learned- V , analytic- V , and the hard gate. The soft gate is $w_t^{\text{soft}} = \gamma^k V(s_{t+k}) - V(s_t)$. **Bold** marks the best value per column.

Method	HALF-CHEETAH		ANT		Go1	
	Reward	Falls	Reward	Falls	Reward	Falls
Unmasked PG	4884 ± 305	165 ± 53	1364 ± 608	4505 ± 2184	6004 ± 459	2844 ± 461
Masked, learned V	1982 ± 1058	306 ± 360	222 ± 275	50011 ± 30828	4374 ± 758	972 ± 206
Analytic V (no gate)	3870 ± 447	73 ± 19	174 ± 150	35481 ± 21851	4473 ± 628	926 ± 259
Soft $C_\theta(\tau)$ gate	2645 ± 829	1215 ± 1843	663 ± 323	11745 ± 18434	4142 ± 1471	2044 ± 2647
Soft $C_\theta(\tau)$, decomposed V	3569 ± 1074	721 ± 985	505 ± 319	8183 ± 8064	4374 ± 1191	1242 ± 472
SafeExplorer (hard $C_\theta(\tau)$)	3855 ± 958	15.3 ± 14.7	2698 ± 237	866 ± 718	5509 ± 155	830 ± 216

The mechanism behind the soft gate’s failure is given in Section 4.4: on a failed recovery segment the soft gate still emits a negative-signed imitation pressure, teaching π_θ to *unlearn* the recovery action at those states, a noisy signal that the hard gate removes by zeroing the term on failures.

E.7 λ_{compat} sensitivity (SafeExplorer)

With the gate fixed, the remaining hyperparameter of the regularizer is its weight. We sweep $\lambda_{\text{compat}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1.0\}$ on HALF-CHEETAH and ANT.

The $\lambda_{\text{compat}} = 10^{-3}$ row runs Table 1’s configuration as a separate sweep arm. The same runs also back Table 18 (HALF-CHEETAH/ANT).

Table 11: Compatibility-regularizer weight sweep on HALFCHEETAH and ANT. Reward and falls are training-time (last-10% mean reward, cumulative falls); mean $\pm$ std over all 10 seeds, dedicated sweep batch at the body operating point, run independently of the Table 1 runs. **Bold** marks the best value per column.

λ_{compat}	HALFCHEETAH reward	HALFCHEETAH falls	ANT reward	ANT falls
10^{-4}	2781 ± 1517	200 ± 220	1874 ± 965	2412 ± 2947
10^{-3}	4770 ± 458	33 ± 32	2936 ± 228	423 ± 362
10^{-2}	5886 ± 633	142 ± 121	3038 ± 457	531 ± 395
10^{-1}	5258 ± 649	112 ± 87	2136 ± 944	168 ± 262
1.0	5352 ± 617	236 ± 258	1010 ± 867	7738 ± 18532

The optimum is environment-specific. On HALFCHEETAH the reward peaks sharply at $\lambda_{\text{compat}} = 10^{-2}$ (5886). On ANT the reward is within seed noise from $\lambda_{\text{compat}} = 10^{-3}$ to 10^{-2} , with 10^{-3} the safer choice. Both environments are robust within a $\sim 10\times$ band around the optimum.

The main results use a single $\lambda_{\text{compat}} = 10^{-3}$. SafeExplorer in Table 1 and Figure 4 fixes $\lambda_{\text{compat}} = 10^{-3}$ across all three environments rather than a per-environment optimum. We make this choice on *safety* grounds. In the sweep above, $\lambda_{\text{compat}} = 10^{-3}$ attains the lowest HALFCHEETAH training-fall count (33, against 142 at 10^{-2}) and a near-lowest count on ANT. A larger $\lambda_{\text{compat}} = 10^{-2}$ would raise HALFCHEETAH reward substantially (5886 vs 4770) and is marginally better on ANT, but it does so at roughly $4.3\times$ the HALFCHEETAH falls. We therefore keep the uniform 10^{-3} default so that the headline fall reductions are not traded away for extra falls, and we note 10^{-2} as the reward-optimal setting for HALFCHEETAH for practitioners who can tolerate the higher fall count.

E.8 Fixed- d ablation: the safe-region schedule vs. a constant d

With the gate and its weight fixed, the last training-time design choice is the safe-region radius, which SafeExplorer anneals on a curriculum. A natural counterfactual is to pick a smaller d_{max} and hold d constant there. Table 12 sweeps fixed $d \in \{0.25, 0.5, 0.75, 1.0\} \cdot d_{\text{max}}$ on each of the three environments, with otherwise-identical SafeExplorer.

Table 12: Fixed- d sweep at the body operating point (SafeExplorer), dedicated sweep batch, mean $\pm$ std over all 10 seeds. d is held constant at the listed percentage of the per-env d_{max} (2.0 / 0.4 / 0.15) throughout training. Cells are **Reward / Falls**; **bold** marks the best reward and lowest falls per environment, ranging over the whole column including the curriculum reference row; at $d = d_{\text{max}}$ the HALFCHEETAH intervention rate is 0 on all ten seeds, so its 100% cell reproduces PPO without recovery, while the ANT and Go1 100% cells run with the mechanism live. The sweep’s own linear arm replicates the curriculum reference within seed noise. Bottom row: linear-curriculum reference at the same d_{max} , the 10-seed cells of Table 1.

d/d_{max}	HALFCHEETAH	ANT	Go1
25%	$3053 \pm 1622/1400 \pm 1102$	$990 \pm 2/4 \pm \mathbf{3}$	$2852 \pm 132/2010 \pm 453$
50%	$2272 \pm 524/3487 \pm 1747$	$1322 \pm 616/8233 \pm 4481$	$5409 \pm 92/\mathbf{390} \pm \mathbf{43}$
75%	$3154 \pm 1095/5038 \pm 3744$	$724 \pm 497/17854 \pm 5284$	$5558 \pm 176/813 \pm 290$
100%	$3904 \pm 814/3746 \pm 689$	$469 \pm 247/25700 \pm 9405$	$5817 \pm 127/1466 \pm 306$
Linear curr. (SafeExplorer, ref.)	$d_{\text{max}}=2.0: 3855 \pm 958/\mathbf{15.3} \pm \mathbf{14.7}$	$d_{\text{max}}=0.4: \mathbf{2698} \pm \mathbf{237}/866 \pm 718$	$d_{\text{max}}=0.15: 5509 \pm 155/830 \pm 216$

HalfCheetah. Every fixed- d cell pays $\sim 1,400$ to $5,000$ training falls and tops out near $2,300$ to $3,900$ reward, while the linear-curriculum reference reaches $3,855$ reward at only 15.3 falls (a roughly $92\times$ to $329\times$ falls advantage on the Table 1 cell). No constant d matches the curriculum’s reward-at-low-falls. The small- d cells stay safer but low-reward and the large- d cells climb in falls without gaining reward.

Ant. The smallest fixed- d (p25, $d = 0.1$) is very safe (4 falls) but useless (990 reward). Larger constant d accumulates $8,200$ to $25,700$ falls without reaching the curriculum’s reward. The linear curriculum gets 2698 reward at 866 falls (the Table 1 cell), which no constant d matches.

Go1. On Go1 the comparison is closer, and the curriculum does *not* strictly dominate. A tuned constant d is competitive. Fixed- d -p50 (5409/390) attains essentially the curriculum’s reward (5509 on the Table 1 cell) at *fewer* falls (390 against the 830 of Table 1; both the fixed- d Go1 cells and the reference are 10 seeds), and p75 matches the curriculum on both axes within seed noise (5558 ± 176 / 813 ± 290 against 5509 ± 155 / 830 ± 216), while p100 reaches a higher reward, 5817 ± 127 against 5509 ± 155 , and pays $1.8\times$ its falls (1466 against 830) to get there. The curriculum’s value on Go1 is therefore robustness to the choice of d , it reaches high reward at low falls without a per-environment d search, rather than a strict win over the best constant d .

Summary. On HALFCHEETAH and ANT no constant d matches the schedule’s high reward at comparable safety. On Go1 a tuned fixed- d (p50) is competitive on both axes, so there the schedule’s contribution is sparing the practitioner a per-environment d search rather than a strict dominance. The safe-region schedule is a supporting mechanism for the objective-gap bound (Theorem 4). The method’s correctness rests on the unbiased masked gradient (Theorem 1), not on the schedule.

E.9 Curriculum-schedule sensitivity (SafeExplorer on Ant)

Table 13: Schedule shape on ANT SafeExplorer. Reward / Falls, mean $\pm$ std over all 10 seeds; sensitivity batch at the wider radius ($d_{\max} = 0.56$) rather than the body 0.4, so within-table comparisons are matched but absolute values differ from Table 1. **Bold** marks the best value per column; Linear and Log rewards tie within seed noise and are both bolded.

Schedule	Reward	Falls
Linear	2938 ± 268	1058 ± 934
Log	3055 ± 346	1120 ± 752
Step (4 jumps)	1601 ± 464	5796 ± 2282
Constant ($d = d_{\max}$)	305 ± 85	30305 ± 6863

Granting that some schedule beats a constant d , the next question is which one. Table 13 reports three alternatives against the linear default. Linear and log are tied within seed noise on every metric. The step schedule is worse on both, with lower reward and $5.5\times$ more falls. Constant- d (no curriculum) collapses, accumulating $29\times$ the falls of the linear curriculum at a tenth of its reward. This is consistent with the fixed- d ablation above (Appendix E.8), whose constant $d=d_{\max}$ cell shows the same qualitative collapse (at the body radius rather than this table’s wider one), and with the prediction of Theorem 4 that the gap closes as β shrinks, which the curriculum drives.

E.10 Compute-scaling: $2\times$ matched-budget on HalfCheetah and Ant

With the design choices settled, we turn to whether the chosen configuration holds up under conditions the body did not exercise, starting with more compute. We re-run PPO and SafeExplorer at $2\times$ the main budget ($8M \rightarrow 16M$) on HALFCHEETAH and ANT (Table 14; wider $d_{\max}$ than the body protocol, see the caption). Re-tuning $d_{\max}$ for longer horizons or a multi-stage curriculum is left to follow-up work.

E.11 Compatibility-regularizer normalization (global vs. per-segment)

The $C_{\theta}(\tau)$ loss in Equation (11) averages across all recovery-active timesteps in the current minibatch (“global”). An alternative “per-segment” normalization averages within each contiguous safe $\rightarrow$ unsafe $\rightarrow$ safe segment, then over segments, which prevents long unsafe excursions from dominating the loss. Table 15 reports both on HALFCHEE-

Table 15: Compat-norm: global vs. per-segment. Reward / Falls, mean $\pm$ std over all 10 seeds; sensitivity batch at the wider $d_{\max}$ (3.2 HALFCHEETAH, 0.56 ANT). **Bold** marks the arm better on both metrics per environment.

Env	global (default)	per-segment
HALFCHEETAH	$4609 \pm 1062/164 \pm 211$	$4968 \pm 519/90 \pm 111$
ANT	$2938 \pm 268/1058 \pm 934$	$2525 \pm 439/2038 \pm 1395$

Table 14: Compute-scaling: SafeExplorer vs PPO at the main budget (8M) and $2\times$ (16M) on HALFCHEETAH and ANT. Reward / Falls, mean $\pm$ std over all 10 seeds. All cells are the sensitivity batch at the wider $d_{\max}$ (3.2 on HALFCHEETAH, 0.56 on ANT) rather than the body protocol; the 16M runs stretch the d -curriculum over the doubled budget. Table 1’s body-radius values (HALFCHEETAH SafeExplorer 3855 ± 958 / 15.3 ± 14.7) therefore differ from the Main column here; see the preamble of this appendix. **Bold** marks the better method per environment and budget.

	Method	Main (8M)	$2\times$ (16M)
HALFCHEETAH	PPO	$3277 \pm 1088/5505 \pm 2457$	$4129 \pm 1463/10206 \pm 7734$
	SafeExplorer	$4609 \pm 1062/164 \pm 211$	$5805 \pm 732/210 \pm 321$
ANT	PPO	$289 \pm 51/33984 \pm 8130$	$223 \pm 113/84739 \pm 31184$
	SafeExplorer	$2938 \pm 268/1058 \pm 934$	$2822 \pm 628/4227 \pm 2958$

TAH and ANT. Per-segment helps on HALFCHEETAH (higher reward, fewer falls) but hurts on ANT (lower reward, more falls). With no consistent winner across environments and the means within roughly one standard deviation, we keep `global` as the simpler default.

E.12 Action-noise robustness (HalfCheetah SafeExplorer)

Table 16: Action-noise on HALFCHEETAH SafeExplorer at $\lambda_{\text{compat}} = 10^{-3}$. Reward / Falls, mean $\pm$ std over all 10 seeds; sensitivity batch at the wider HALFCHEETAH radius ($d_{\max} = 3.2$). The $\sigma_a = 0$ arm is the same runs as the global arm of Table 15 and the Main SafeExplorer cell of Table 14.

σ_a	Reward	Falls
0.00	4609 ± 1062	164 ± 211
0.05	4212 ± 587	182 ± 267
0.10	2813 ± 981	388 ± 329

A second stress axis perturbs the actions themselves. We re-train SafeExplorer on HALFCHEETAH with Gaussian action noise of standard deviation $\sigma_a \in \{0.0, 0.05, 0.10\}$ added to every action (recovery and main) at the $\lambda_{\text{compat}} = 10^{-3}$ default (Table 16). Reward and falls stay within seed noise up to $\sigma_a = 0.05$, then degrade at $\sigma_a = 0.10$. An action-noise sweep at the HC-tuned $\lambda_{\text{compat}} = 10^{-2}$ is left to follow-up work. We also ran $\sigma_a = 0.20$ as a stress test. SafeExplorer collapses (1863 ± 542 reward, 1659 ± 1807 falls) when the injected noise is comparable in scale to the recovery’s action magnitude, and we omit it from the table as a known out-of-regime failure mode.

E.13 Failure-mode catalog

The stress tests above probe how the chosen configuration degrades; this catalog records the qualitative failure each rejected variant exhibits, so the quantitative gaps in the ablations have a concrete behavioral reading.

- **PPO** on Go1: agent falls within first 200 steps; recovery is unavailable; episode terminates.
- **Analytic- V variant** on ANT: agent learns to balance briefly but never escapes the recovery policy’s basin; α stays near 1 for the full training run.
- **Soft-gate variant** on HALFCHEETAH: regularizer fits failed-recovery actions; reward oscillates; final policy is unreliable.
- **SafeExplorer** on ANT: clean separation, α drops to near zero around update 1000, reward climbs steadily thereafter, falls plateau.

E.14 Recovery-disabled evaluation for all methods

Table 17 reports recovery-disabled evaluation for the full method set. The sharpest contrast is on ANT, where recovery is unreliable; SafeExplorer is the only method whose recovery-disabled policy works. On HALFCHEETAH most methods transfer, and SafeExplorer’s recovery-disabled falls are within noise of the best variant at near-top reward. On GO1 at the near-body $d_{\max}$ (0.1, just under the body 0.15) SafeExplorer’s recovery-disabled policy leads all eight methods on both axes; at the wider $d_{\max}$ it again leads on reward but four masked variants record fewer evaluation falls. An independent 10-seed batch at the same protocol (the crossed-cell batch of Appendix E.5) replicates every shared cell within one standard deviation. On WALKER2D the winning configuration is *not* self-sufficient; recovery-disabled evaluation reaches under half the mixed-controller reward (10 seeds; Appendix E.18).

Table 17: Recovery-disabled final evaluation (deterministic main policy, recovery branch disabled): final eval reward and cumulative eval falls, 10 seeds unless noted. HALFCHEETAH/ANT: the wider sensitivity $d_{\max}$ (3.2 / 0.56), 8M steps. GO1: 16M steps at two logged $d_{\max}$ values (0.1 and 2.2). The columns differ only in the training batch the policies come from. Coverage is every method except the Unmasked-PG variant on GO1, never run there; two GO1 cells are short of ten seeds.

Method	HALFCHEETAH		ANT	
	Eval reward	Eval falls	Eval reward	Eval falls
PPO (no recovery)	3525 ± 989	232 ± 173	299 ± 95	939 ± 23
Recovery RL (action-relabeling port)	2341 ± 1775	216 ± 318	83 ± 69	972 ± 5
Safe Legged (reward-shaping port)	4756 ± 1406	196 ± 120	70 ± 36	974 ± 1
Unmasked PG	4863 ± 296	44 ± 14	1215 ± 406	460 ± 189
Masked, learned V	3266 ± 1099	63 ± 117	404 ± 422	670 ± 279
Masked, analytic V	3421 ± 1482	65 ± 121	433 ± 498	688 ± 324
Soft $C_\theta(\tau)$ gate	2845 ± 1460	251 ± 296	877 ± 664	437 ± 291
Soft $C_\theta(\tau)$, decomposed V	4033 ± 815	162 ± 192	683 ± 780	469 ± 220
SafeExplorer	4585 ± 1320	52 ± 101	3124 ± 358	90 ± 72
Method	Go1 near-body batch		Go1 sensitivity batch	
	Eval reward	Eval falls	Eval reward	Eval falls
PPO (no recovery)	3555 ± 1187	1113 ± 83	4063 ± 934	1054 ± 103
Recovery RL (action-relabeling port)	1926 ± 1076	1593 ± 338	3788 ± 678	1074 ± 85
Safe Legged (reward-shaping port)	126 ± 41	1951 ± 1	2928 ± 1175	1365 ± 193
Masked, learned V	1275 ± 1299	1779 ± 327	4198 ± 903	845 ± 167
Masked, analytic V	3196 ± 1384	1310 ± 456	3697 ± 1076	958 ± 110
Soft $C_\theta(\tau)$ gate	3027 ± 1136	1302 ± 453	4647 ± 775	984 ± 79
Soft $C_\theta(\tau)$, decomposed V	1901 ± 1778	1555 ± 469	3959 ± 1010	911 ± 214
SafeExplorer	4821 ± 755	912 ± 69	5173 ± 587	1005 ± 71

E.15 Empirical check of Theorem 4

We now turn from the method’s empirical robustness to its theory, checking each of the paper’s two theorems against the runs, starting with the objective-gap bound. Theorem 4 bounds the deployment gap $|J^{\text{mix}}(\theta_k) - J(\theta_k)|$, between the during-training mixed-policy return and the return of π_θ deployed alone, by the main-policy out-of-region rate $\beta(\theta, d)$. The bound is a proportionality, not a vanishing guarantee. It predicts a small gap exactly when $\beta(\theta, d)$ is small, and Corollary 5 sharpens this to an exact fixed point only in the idealized $\beta(\theta, d) \rightarrow 0$ limit. We test that prediction directly. Every 4 PPO updates we run a recovery-disabled evaluation pass, averaging 5 episodes per pass, that samples from the same stochastic policy $\pi_\theta(\cdot|s) = \mathcal{N}(\mu_\theta(s), \sigma_\theta^2)$ used in training, the operative $J(\theta_k)$, and compare it to the mixed-policy rollout return $J^{\text{mix}}(\theta_k)$. Because $\beta(\theta, d)$ is an out-of-region rate under π_θ alone and is not directly observable, we report alongside it the observable recovery rate α , which Section 4.2 identifies as a diagnostic for $\beta(\theta, d)$ rather than a certified upper bound. A deterministic pass using $\mu_\theta(s)$ alone is also logged as a noise-reduced deployment diagnostic but is not the quantity the theorem concerns.

Table 18: End-of-training values for SafeExplorer at the body operating point, all 10 seeds, over the last 20% of training; The batch is shared with the $\lambda_{\text{compat}} = 10^{-3}$ arm of Table 11 on HALFCHEETAH and ANT and the body-16M family on GO1. Columns: the mixed-policy return J^{mix} , the π_θ -only stochastic eval return J , their signed difference, the recovery rate α (observable diagnostic for $\beta(\theta, d)$), cumulative training-time falls, and per-episode eval-time falls under π_θ alone (recovery disabled).

Env	J^{mix}	J	$J^{\text{mix}} - J$	α	Falls (train)	Falls (eval/ep)
HALFCHEETAH	4671 ± 461	4670 ± 474	0 ± 16	0.000	33 ± 32	0.00
ANT	2906 ± 229	2912 ± 230	-6 ± 30	0.000	423 ± 362	0.12
GO1	5384 ± 453	4800 ± 620	584 ± 222	0.048 ± 0.009	911 ± 267	0.39

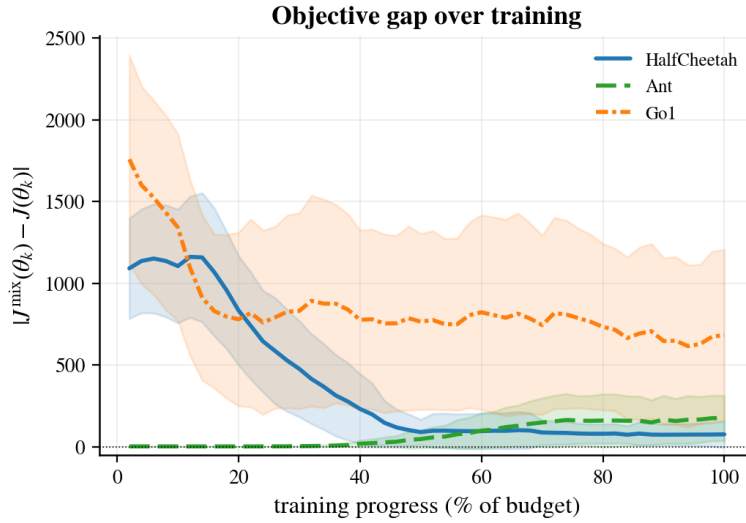


Figure 6: Objective gap over training for the SafeExplorer runs of Table 18: the curve is the seed-mean of the per-seed absolute gap $|J^{\text{mix}}(\theta_k) - J(\theta_k)|$, smoothed with a 9-point moving average for display; shaded $= \pm 1$ std. Because the curve averages magnitudes while Table 18 reports the signed across-seed mean over the final 20% of training, the curve’s end value can sit above the table’s near-zero signed gap (as on ANT). The curves are drawn from the ten seeds of Table 18.

The result follows the bound’s structure precisely, and the key observation is that *the gap tracks $\beta(\theta, d)$* . On HALFCHEETAH and ANT the recovery rate falls to essentially zero ($\alpha = 0.000$ on both at end-of-training), and the gap closes to within seed noise of zero (0 ± 16 and -6 ± 30). These are the self-stable environments. Here π_θ holds the safe region unaided, so $\pi_\theta^{\text{mix}} \equiv \pi_\theta$ along the trajectory, $\beta(\theta, d) \rightarrow 0$, and the fixed point of Corollary 5 is reached up to the invariance-slack residual $\eta_\star$ the corollary anticipates (the residuals here are statistically indistinguishable from zero; empirically it is the stochastic policy’s entropy floor that keeps $\eta_\star$ from being exactly zero). On GO1 the policy does not become fully self-stable, because the quadruped’s gait cannot stay inside the tight $\mathcal{R}(d_{\text{max}})$ tube. The recovery rate therefore settles at a small positive floor ($\alpha = 0.048 \pm 0.009$) rather than at zero, and the gap settles at a correspondingly positive residual (584 ± 222 , about 11% of J^{mix}) rather than vanishing. After the early-curriculum transient the gap collapses from its peak on HALFCHEETAH, stays within seed noise of zero on ANT in the final average despite a late upward drift in the plotted per-seed magnitude, and declines but stays positive on GO1 (Figure 6). On GO1 it falls from ~ 1750 when the safe region is small to a small positive residual at 16M env-steps. The end-of-training gap and recovery rate in Table 18 make the proportionality explicit. The gap is near zero exactly where α is, and the only environment with a residual gap is the only one with a residual α .

This two-regime behavior is a stronger test of Theorem 4 than a uniform collapse to zero would be. The bound does not assert that the gap vanishes; it asserts that the gap is governed by $\beta(\theta, d)$. HALFCHEETAH

and ANT realize the vanishing regime ($\beta(\theta, d) \rightarrow 0$), and GO1 realizes the residual regime ($\beta(\theta, d)$ bounded away from zero because Assumption 2 holds only approximately for a quadruped confined to a tight tube), with the observable α moving in lockstep with the gap in both. The sign of the gap is itself interpretable. On GO1, $J^{\text{mix}} - J > 0$ means that deploying π_θ alone forfeits the return that the recovery policy would otherwise secure, so the gap measures the policy’s residual reliance on the recovery policy, which HALFCHEETAH and ANT drive to zero and GO1 reduces to about 11%.

The absolute prefactor of Theorem 4 ($2r_{\max}/(1-\gamma)^2 \approx 2 \times 10^5$ for $r_{\max} = 10$, $\gamma = 0.99$) is loose, as is standard for performance-difference-lemma bounds, so we do not read the inequality quantitatively. Its operative content is the structural claim that any mechanism reducing $\beta(\theta, d)$, here the radius curriculum together with the compatibility regularizer, tightens the gap. Figure 6 confirms that claim across the full range of $\beta(\theta, d)$ our three environments realize.

Measuring $\beta(\theta, d)$ and the bound’s numerical magnitude. $\beta(\theta, d)$ is reported two ways, collected below. The first is a pair of certified lower bounds per environment (except on HALFCHEETAH, explained below) from the existing recovery-disabled evaluation logs, a fall route (every evaluation-time fall is an out-of-region step) and a gap route (the measured gap together with the bound itself). The second is a direct measurement $\hat{\beta}$ on the saved final checkpoints, the fraction of steps outside $\mathcal{R}(d_{\text{end}})$ over 20 recovery-disabled episodes per seed, at 10 seeds; six of the ten ANT seeds also measure exactly 0. Gaps are the absolute tail gaps over the 10-seed batch; Table 18 prints the signed end-of-training values, and we label both rather than let them read as one quantity.

Env	$\beta(\theta, d)$			Bound right-hand side		
	fall route	gap route	direct $\hat{\beta}$	at fall-route $\beta(\theta, d)$	at $\hat{\beta}$	Measured gap
HALFCHEETAH	$\geq 1.7 \times 10^{-5}$	$\geq 5.0 \times 10^{-6}$	$< 6 \times 10^{-6}$	3.4		1.0
ANT	$\geq 1.4 \times 10^{-4}$	$\geq 4.6 \times 10^{-5}$	$(2.4 \pm 3.6) \times 10^{-4}$	28	48	9.3
GO1	$\geq 3.9 \times 10^{-4}$	$\geq 2.9 \times 10^{-3}$	0.032 ± 0.034	78	6400	585

The route-based right-hand sides are lower bounds on the true right-hand side, so they can legitimately sit below the measured gap. At the directly measured $\hat{\beta}$ the ANT inequality holds with a factor-5 margin rather than vacuously, while the GO1 right-hand side sits within a factor of two of the trivial return bound $r_{\max}H = 10^4$; GO1 is beyond the $\beta(\theta, d) < 1 - \gamma$ regime where the bound is informative, which is why we read the bound structurally. On HALFCHEETAH every non-terminal state lies inside $\mathcal{R}(d_{\text{end}})$, so $\hat{\beta}$ is zero by construction. A HALFCHEETAH fall can also end inside the region, and the measured HALFCHEETAH gap is then seed noise, so neither HALFCHEETAH route-based cell is a lower bound. The bound therefore cannot be tested there, and we draw no conclusion on HALFCHEETAH.

Eval-time falls under π_θ alone (recovery disabled) corroborate the reading. They are 0.00, 0.12, and 0.39 per episode on HALFCHEETAH, ANT, and GO1, ordered exactly as the gaps and the recovery rates are. Figure 7 plots the full recovery-rate trajectories that this diagnostic summarizes.

E.16 Masked gradient versus IS under a stochastic recovery

With the objective-gap bound checked, we turn to the gradient estimator: Theorem 1 makes the masked estimator exactly unbiased at recovery steps, while per-step IS, where it applies at all, is biased in general (Remark 1); what remains empirical is how the two compare in practice. Theorem 1 holds for any θ -independent recovery; for a deterministic one, not even IS applies (Section 4.1). Here we ask what happens when the recovery *is* stochastic, so IS becomes an option. Writing $w_t = \pi_\theta(a_t | s_t)/\pi^{\text{rec}}(a_t | s_t)$ for the importance weight at a recovery step, the IS estimator of $\nabla_\theta J^{\text{mix}}(\theta)$ (Corollary 2) decomposes as

$$\hat{g}_{\text{IS}} = \underbrace{\sum_{t: s_t \in \mathcal{R}} \nabla_\theta \log \pi_\theta(a_t | s_t) A_t}_{\hat{g}_{\text{masked}}} + \sum_{t: s_t \notin \mathcal{R}} w_t \nabla_\theta \log \pi_\theta(a_t | s_t) A_t, \quad (17)$$

with A_t the PPO advantage estimate. The second sum would be mean zero by the score-function identity only if A_t were replaced by an action-independent baseline $b(s_t)$; with the actual advantage, which depends

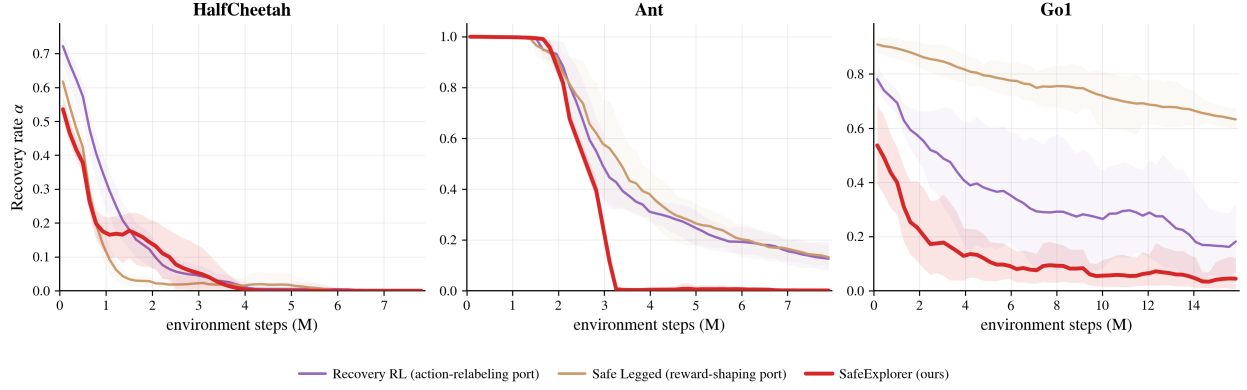


Figure 7: Recovery rate α (fraction of rollout steps on which recovery fired) over training, per environment, for the recovery-using methods (the CMDP solvers act with a single policy and never invoke recovery, so they have no α). Solid = mean across seeds, shaded = [min, max] envelope.

on a_t through the reward and the successor state, its conditional mean is the off-policy policy-gradient contribution at the unsafe states, which is generically nonzero. The per-step IS estimator is therefore biased for $\nabla_{\theta} J^{\text{mix}}$, and masking removes a bias-carrying term, not merely a noisy one. A recovery policy differs sharply from the main policy by design, so the term’s weights are heavy-tailed; truncating them (Espeholt et al., 2018; Munos et al., 2016) trades variance for further bias. The measurement below therefore compares two estimators that do not share a mean: the recorded explosion of the IS weights is evidence about the conditioning of the added term, not a like-for-like variance comparison between unbiased estimators.

Concretely, we rerun SafeExplorer on all three environments (settings of Section 6) with the recovery action *sampled* from the SAC recovery rather than taken greedily at its mean, and shadow-compute, per rollout, the weights’ effective sample size and each estimator’s gradient variance. Training itself always uses the masked gradient.

Figure 8 shows that the three regimes expose complementary failure modes. On HALFCHEETAH the strong recovery makes the weights degenerate from the first update. The effective sample size never exceeds 0.5% of the recovery steps, and the IS gradient variance exceeds the masked variance by a factor of up to 8×10^{12} . On ANT the weak recovery initially overlaps the main policy and IS stays formally usable for longer, but the variance ratio holds near 10^{13} for the first 1.2M steps. GO1 completes the spectrum. Over its 12-dimensional action space the recovery’s actions are so unlikely under the main policy that the weights die at once (effective sample size always below 3%, reaching 7×10^{-5}), so the added term vanishes before its variance can build, and the ratio never rises above ~ 40 and sits at one for the full 16M steps, over which the quadruped’s recovery rate never reaches zero (Appendix E.15). Once the main policy separates from the recovery the weights vanish and the ratio returns to one. IS proves either catastrophically noisy or identical to the masked estimator, never better in any regime we measure.

Training with the truncated-IS gradient (w_t clipped at $c = 10$) confirms the cost (Figure 9). These runs use the sampled-recovery setup of this appendix, so their absolute rewards are not comparable to Table 1’s greedy-recovery numbers; only the within-setup masked-vs-IS differences are meaningful. On HALFCHEETAH the weights die immediately and the two variants are statistically indistinguishable (reward 4675 ± 642 versus 4421 ± 647). On ANT the million-step degeneracy window is expensive: 793 ± 214 reward against 1325 ± 289 for the masked gradient, a 40% drop, and its lower fall count (2066 ± 1582 versus 4682 ± 1047) reflects the weaker policy rather than safer behavior. On GO1 the truncated-IS variant again trails the masked gradient (4884 ± 241 versus 5243 ± 219 reward, a 7% drop) at statistically indistinguishable falls (3293 ± 589 versus 3401 ± 351).

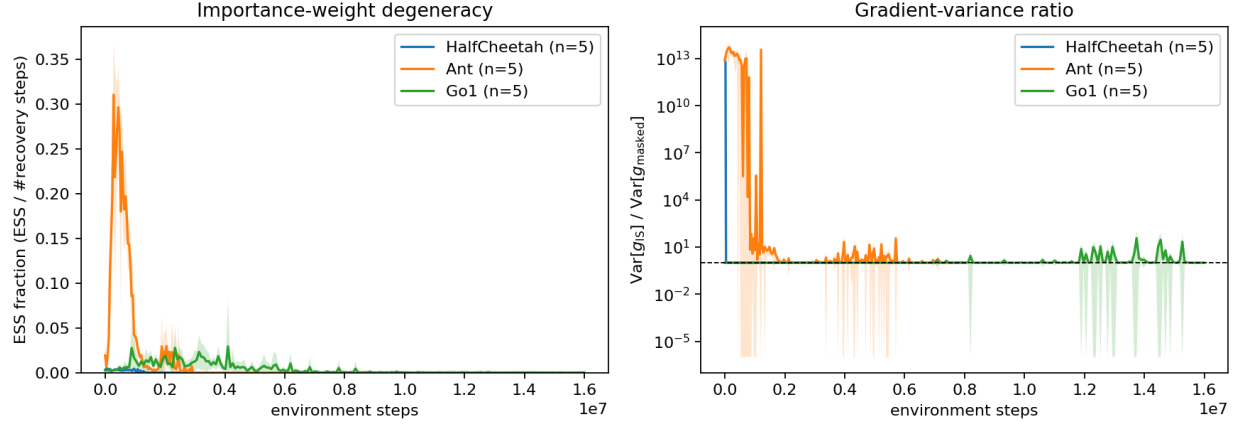


Figure 8: Masked gradient versus IS on a stochastic recovery, SafeExplorer (mean $\pm$ std). *Left*: the effective sample size of the importance weights collapses toward zero when the recovery is active, the signature of weight degeneracy. *Right*: the ratio of IS to masked gradient variance (log scale; dashed line at one). The masked estimator targets $\nabla_{\theta} J^{\text{mix}}$ (Equation (17)) without the IS term’s bias and variance pathologies. The HALF-CHEETAH and ANT curves end where the recovery stops engaging, because the diagnostics exist only on updates with recovery steps; on Go1 recovery stays mildly engaged throughout, so the curve spans the full run.

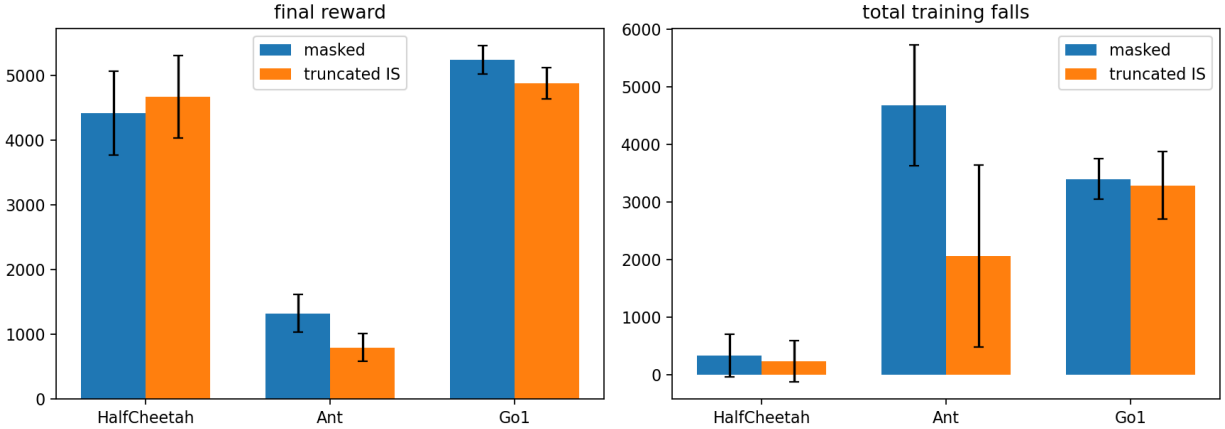


Figure 9: Training SafeExplorer with the masked gradient versus the truncated-IS gradient (mean $\pm$ std). *Left*: final reward. *Right*: total training falls.

E.17 CMDP baselines: constraint feasibility

Having checked the method’s own theory, we finally audit the baselines it is compared against, asking whether the CMDP solvers even satisfy the constraint they are given. This appendix supports the CMDP comparison of Section 7.3. For the comparison to be fair, both solvers use the OmniSafe implementation (Ji et al., 2024) and run on the identical environments, reward functions, network architecture, optimizer, and step budget as the variants in Table 1. We use the direct cost encoding, a cost of 1 on a fall (a termination), the actual safety signal the constraint should bound. The only incidental differences from SafeExplorer are observation and reward normalization, applied to the CMDP baselines and not to SafeExplorer, so they favor the baselines. Reported rewards for both solvers are the raw episode return (EpRet in OmniSafe), un-normalized and therefore on the same task-reward scale as the other methods. The normalization affects only the solvers’ internal optimization, not the reported numbers.

On this matched footing, Table 19 reports the final episodic cost each solver converges to, against the cost budget of 0.05 it was given (a per-episode fall rate), averaged over the same seeds. The pattern is clean. Both solvers reach feasibility on HALFCHEETAH, where staying safe is easy, and both *fail* to reach feasibility on ANT and Go1, where the converged cost exceeds the budget by up to $19\times$. The poor safety of the CMDP baselines is therefore not a tuning artifact. The constrained optimization does not find a feasible policy at all in the regimes where the safe-region intervention is needed.

Table 19: Converged episodic cost vs. the cost budget (≤ 0.05) for the CMDP baselines (final 5% of training). $\checkmark$ feasible, $\times$ budget violated.

Method	HALFCHEETAH	ANT	Go1
CPO	0.007 ± 0.011 $\checkmark$	0.103 ± 0.029 $\times$	0.956 ± 0.025 $\times$
PPO-Lagrangian	0.001 ± 0.001 $\checkmark$	0.326 ± 0.177 $\times$	0.377 ± 0.126 $\times$

A safe-region indicator cost (cost 1 at every step outside $\mathcal{R}(d_{\max})$) is a second natural encoding. We report it for both CMDP solvers (Table 20). Across the table, *feasibility in the region metric does not bound falls*. On HALFCHEETAH the dense and sparse encodings coincide, the constraint never binding. On ANT both solvers are feasible on the region budget yet fall at least $12\times$ (CPO) and $9\times$ (PPO-Lagrangian) SafeExplorer’s count, at far lower reward; on Go1 the dense cost raises PPO-Lagrangian’s reward, but both solvers violate their own budget and still fall at least $12\times$ (PPO-Lagrangian) and $80\times$ (CPO) SafeExplorer’s count. Every comparative claim here holds at either endpoint of the falls brackets.

Table 20: Dense safe-region CMDP baseline: cost 1 per step outside $\mathcal{R}(d_{\max})$, limit 5.0, 10 seeds per cell, body budgets and $d_{\max}$. Reward is the last-10% mean; episodic region cost is the final-5% mean against the limit (≤ 5.0); falls are conservative reconstructed brackets [lb, ub] from episode-length accounting, since these runs log no per-fall signal.

Env	Method	Reward	Episodic region cost	Feasible?	Falls bracket
HALFCHEETAH	CPO	2729 ± 303	0.0 ± 0.0	$\checkmark$ (never binds)	[4.5, 312]
HALFCHEETAH	PPO-Lagrangian	2733 ± 336	0.0 ± 0.0	$\checkmark$ (never binds)	[5.2, 316]
ANT	CPO	676 ± 118	4.2 ± 0.4	$\checkmark$	[11119, 19078]
ANT	PPO-Lagrangian	1265 ± 640	3.5 ± 1.9	$\checkmark$	[8384, 16296]
Go1	CPO	993 ± 531	5.9 ± 1.1	$\times$	[66747, 82746]
Go1	PPO-Lagrangian	4498 ± 589	5.1 ± 1.5	$\times$	[10208, 26203]

E.18 Walker2d: trigger calibration and dose-response

This subsection reports WALKER2D, a fourth environment kept out of the body results. It sits at the narrow end of the failure spectrum; falls are fast and intrinsically hard to prevent, PPO averaging one fall per roughly 210 training steps. The protocol is 8M steps per run and 5 to 10 seeds per arm with 10 on the headline comparison, with PPO retrained under identical code and separately pretrained SAC checkpoints as the recovery policies. The legacy distance is $\mathcal{D}(s) = \|(z - 1.25, \theta)\|$; the calibrated distance normalizes each component by its distance from the nominal state to the termination surface, $\mathcal{D}'(s) = \|((z - 1.25)/0.55, \theta/2.0)\|$, so both fall routes cross $\mathcal{D}' \approx 1$ exactly at termination. Table 21 holds the full trigger-calibration dose-response. Trigger tightness under the calibrated distance acts as a dial; falls are monotone in the radius, and the widest calibrated radius reproduces the transplanted failure of the legacy rows. That legacy transplant failure is geometric rather than a competence failure; 61% of the falls in the legacy arms occurred in episodes where the trigger never fired, a height collapse capping the legacy distance at about 0.55 while the annealed radius sweeps past it, even though the offline catch-rate screen clears the deployed checkpoint at 96 to 99% at every trigger distance tested. The winning cell is a moderated claim by construction, a real but single-digit reduction ($3.94\times$, 95% stratified-bootstrap CI [3.27, 4.78], 10 seeds) at a 25% reward cost, not the orders of magnitude of the wide-band regime.

Catch-rate screen and disclosures. The practitioner guidance we draw is a screening protocol rather than a retraining recommendation. Before any task training is spent, roll a candidate recovery out from doomed handover states collected at the intended trigger band and record the fraction it saves. The measurement is cheap, needs no task training, and in this study predicts the in-loop outcome where the recovery’s own training return does not. The screen itself uses 200 handover states per trigger band, snapshotted at the first threshold crossing of each excursion of an unassisted walker; left uncontrolled, 88 to 97% of them fall within about 15 to 20 steps, so the states are genuinely doomed, and the catch rate is the fraction the candidate returns to the safe region before termination. Table 22 holds the in-loop result. With the trigger fixed at the winning calibrated setting and only the recovery controller varying, the deployed 97% catcher yields the $3.94\times$ reduction, while both sub-threshold checkpoints (72% and 58%) produce roughly $3\times$ more falls than running with no recovery controller at all while destroying task learning. The effect is a cliff rather than a dial; the two sub-threshold arms are statistically indistinguishable from each other. The legacy-trigger 58% arms of Table 21 sit at only 1.1 to $1.45\times$ PPO’s falls, not $3\times$, because there the mis-calibrated trigger rarely fired; the trigger determines how often the controller is consulted, and the catch rate what happens when it is. Both recovery checkpoints’ pretraining falls are counted in Appendix D.4. Recovery-disabled evaluation of the winning configuration reaches under half the mixed-controller reward; at the tight trigger the learned policy leans on the recovery, so unlike on HALFCHEETAH and ANT it is not self-sufficient, and the deployment-relevant quantity is the mixed-controller number under the recovery-permanent J^{mix} framing of Section 3 (Appendix E.14).

Table 21: WALKER2D trigger calibration: full dose-response over both trigger families, 8M steps per run, mean $\pm$ sample standard deviation (denominator $n - 1$) over seeds (spreads are sample standard deviations; every arm carries at least 5 seeds and the calibrated arms 10). Catch rate is the offline single-shot catch rate of the arm’s recovery checkpoint from doomed handover states at the tight band; d_{max} is the annealed radius parameter (final trigger radius $d_{\text{max}} + 0.05$); Rec. frac. is the fraction of late-training steps under recovery. The caveat on the winning arm (bold) is that its recovery-disabled evaluation reward is 260 ± 110 versus the mixed-controller 636 ± 117 , so the trained policy is not self-sufficient without the recovery (Appendix E.14).

Trigger	Catcher	d_{max}	n	Training falls	Training reward	Rec.-off eval reward	Rec. frac.
none (PPO)	none	–	10	37954 $\pm$ 5182	848 $\pm$ 192	861 $\pm$ 183	0.000
legacy $\mathcal{D}$	58%	1	5	54864 $\pm$ 3977	270 $\pm$ 48	263 $\pm$ 46	0.029
legacy $\mathcal{D}$	58%	2	5	45007 $\pm$ 3479	550 $\pm$ 205	700 $\pm$ 520	0.000
legacy $\mathcal{D}$	58%	4	5	44755 $\pm$ 17222	709 $\pm$ 153	768 $\pm$ 200	0.000
legacy $\mathcal{D}$	58%	8	5	41037 $\pm$ 3226	761 $\pm$ 160	787 $\pm$ 166	0.000
legacy $\mathcal{D}$	97%	1	5	33003 $\pm$ 3183	433 $\pm$ 89	431 $\pm$ 107	0.011
legacy $\mathcal{D}$	97%	2	5	35477 $\pm$ 3785	672 $\pm$ 89	696 $\pm$ 97	0.000
legacy $\mathcal{D}$	97%	4	5	33182 $\pm$ 2500	811 $\pm$ 206	829 $\pm$ 222	0.000
legacy $\mathcal{D}$	97%	8	5	33840 $\pm$ 3581	961 $\pm$ 176	1000 $\pm$ 207	0.000
calibrated $\mathcal{D}'$	97%	0.2	10	9634 $\pm$ 2647	636 $\pm$ 117	260 $\pm$ 110	0.273
calibrated $\mathcal{D}'$	97%	0.4	10	17722 $\pm$ 2982	333 $\pm$ 158	206 $\pm$ 152	0.200
calibrated $\mathcal{D}'$	97%	0.8	10	29984 $\pm$ 8793	283 $\pm$ 170	261 $\pm$ 169	0.105

F Broader Impact

The goal of this work is to make reinforcement learning on physical robots safer to train. Achieving that goal would reduce hardware damage during training, lower the experimental cost borne by robot-learning research groups, and provide a foundation for on-robot continual learning that does not depend on extensive pre-training in simulation.

Set against that intent, the dual-use surface of the technique is narrow. It is dual-use only in the trivial sense that any improvement in robotic control technology is so, because the method does not change the capabilities a deployed policy can express, only the training-time safety profile of how that policy is reached.

Table 22: WALKER2D recovery-controller competence at the winning calibrated trigger ($\mathcal{D}'$, $d_{\max} = 0.2$): only the recovery controller varies, 8M steps per run, mean $\pm$ sample standard deviation over 10 seeds. Catch rate is the offline screen described in the text (200 doomed handover states at the tight band). A sub-threshold controller is not merely less helpful; consulted at every excursion, it accumulates roughly $3\times$ PPO’s falls at near-zero reward, and the two sub-threshold arms are statistically indistinguishable from each other (Welch $t = 0.22$).

Recovery controller (catch rate)	Training falls	Training reward	Rec.-off eval reward
97% (deployed)	9634 $\pm$ 2647	636 $\pm$ 117	260 $\pm$ 110
none (PPO)	37954 $\pm$ 5182	848 $\pm$ 192	861 $\pm$ 183
72% (second candidate checkpoint)	120080 $\pm$ 12731	29 $\pm$ 20	40 $\pm$ 17
58% (weaker pretrained checkpoint)	115608 $\pm$ 62204	30 $\pm$ 17	41 $\pm$ 21

For this reason we do not foresee specific misuse pathways introduced by this work that are not already present in the underlying PPO + SAC + MuJoCo / Go1 stack.

The experiments themselves are deliberately scoped to limit any training-time harm. We do not collect or release human-subject data and do not perform experiments that could affect bystanders during training. All experiments are in simulation (MuJoCo and the Go1 model), so the training-time falls we reduce are simulated rather than physical; carrying that reduction onto hardware is future work (Section 9). Complete safety requires hardware safety mechanisms, human supervision, emergency stops, and formal safety validation, and SafeExplorer replaces none of them. What we show is that falls can be reduced, which reduces the chance of hardware damage during training. The method still relies on a hand-designed safe region and a recovery controller that can fail.

Because the contribution is methodological, its value depends on being reproducible and affordable, and we account for both: the full algorithm, hyperparameter, and environment specifications needed to rerun the experiments are given in Appendices B to D. We report negative results alongside the positive ones, the soft-gate variants, the fixed- d concessions, and the action-noise stress failures (Appendices E.6, E.8 and E.12), so that future work can avoid repeating them. The compute budget for the component ablation (9 variants $\times$ 3 environments $\times$ 5 seeds, 4 cores each, weighted by per-environment wall-clock) is approximately 3,700 CPU-core-hours, modest relative to contemporary safe-RL benchmarks; the ten-seed extensions, sensitivity sweeps, and WALKER2D study of Appendix E add further compute.