ROBUSTRLLIB: A UNIFIED LIBRARY AND BENCHMARK FOR ROBUST REINFORCEMENT LEARNING ALGORITHMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Robust reinforcement learning (RL) is critical for reliable real-world deployment, where policies may fail under diverse shifts and disruptions. Existing robust RL algorithms offer a rich set of mechanisms and principles, yet two major barriers limit their broader potential: limited algorithm reusability and an unclear empirical capability boundary. We introduce *RobustRLlib*, an algorithm-centric library and benchmark that addresses both challenges. It integrates 16 representative robust online, offline, and safe RL methods under a unified interface for substantially improved reusability, together with an enhanced evaluation platform spanning diverse real-world shifts and advanced humanoid and vision-language-action tasks. Systematic evaluation over 273 perturbation configurations provides an up-to-date capability map of current robust RL algorithms, revealing where they remain effective and where substantial robustness gaps persist. As an open-source, user-friendly platform, *RobustRLlib* offers a reproducible and extensible codebase for applying existing methods to new problems and developing new robust RL algorithms. The code and tutorial are available at `https://robust-rllib.site`.

1 INTRODUCTION

Despite remarkable advances in reinforcement learning (RL), its real-world deployment remains limited by robustness and safety concerns, particularly in high-stakes domains such as autonomous driving (Ding et al., 2023b), clinical trials (Liu et al., 2015), robotics (Li et al., 2021), and semiconductor manufacturing (Kozak et al., 2023). Policies trained in idealized environments can easily fail under distribution shifts or disruptions affecting various stages of the interaction, such as observations (Zhang et al., 2020; 2021; Han et al., 2022; Sun et al., 2021; Xiong et al., 2022), actions (Huang et al., 2017), or environment dynamics (Iyengar, 2005; Pinto et al., 2017), induced by diverse factors such as sim-to-real gaps (Pinto et al., 2017), model uncertainty (Bertsimas et al., 2019), noise, or adversarial attacks (Zhang et al., 2020; Klopp et al., 2017; Mahmood et al., 2018).

Robust RL therefore aims to develop policies that remain reliable under diverse types of disruptions and shifts, or their combinations in real-world scenarios (Bertsimas et al., 2019; Tversky & Kahneman, 1974; Arthur, 1991). A broad range of robust RL algorithms has shown strong performance in their specific type of disrupted application scenarios. More importantly, existing methods provide a rich set of robustness mechanisms and principles, including adversarial training (Pinto et al., 2017; Zhang et al., 2021; Rigter et al., 2022), dynamics randomization (Peng et al., 2018), robust value updates (Yang et al., 2022; Panaganti et al., 2022), and data augmentation (Yu et al., 2020b; Lu et al., 2023; Ding et al., 2024), offering substantial promise across a far broader range of applications and shift scenarios.

However, realizing this broader potential faces two fundamental barriers: *algorithm reusability* and *systematic characterization of the capability boundary*. First, existing robust RL algorithms are fragmented across separate implementations and training pipelines, lacking a unified library or interface through which practitioners can readily try and compare a broad collection of robust RL methods on a new application. Second, the empirical capability boundary of current robust RL remains unclear: for a given application and distribution shift, we lack a systematic understanding of how well exist-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ing methods perform, when they remain effective, and where they fail. This is largely because the methods are typically evaluated in distinct environments and shift scenarios, making cross-method comparison difficult and performance beyond their original settings poorly understood.

To address these two gaps, we introduce *RobustRLlib*, an algorithm-centric robust RL benchmark that brings together (1) a robust RL method library with reusable implementations to substantially lower the barrier to applying existing methods to new problems, and (2) a comprehensive evaluation pipeline across tasks and diverse real-world shift scenarios. Complementing existing robust RL benchmarks that primarily focus on developing evaluation tasks and shift scenarios (Dulac-Arnold et al., 2020; Zouitine et al., 2024; Gu et al., 2025), *RobustRLlib* focuses on making robust RL algorithms systematically accessible, comparable, and extensible, while providing a broader evaluation framework (Comparison in Appendix B.1). Our main contributions are summarized as follows:

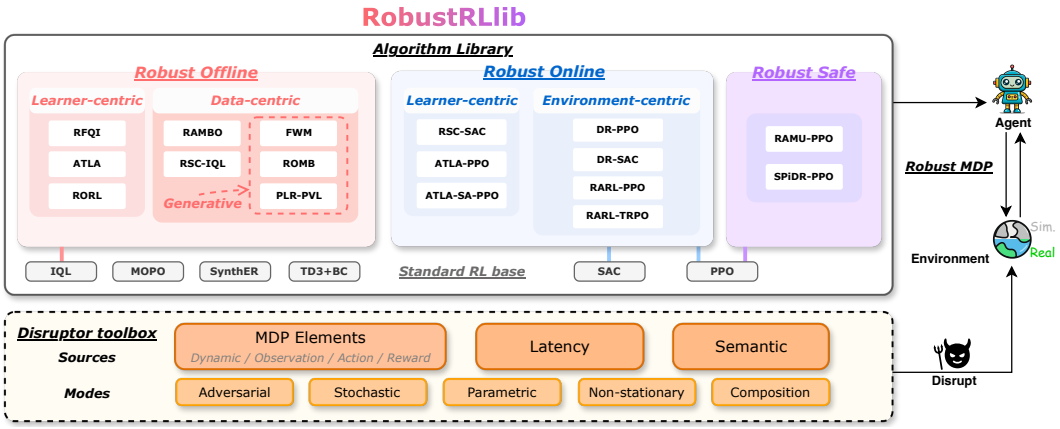

Figure 1: The overview of *RobustRLlib*.

- **A robust RL algorithm library.** We integrate sixteen representative robust online, offline, and safe RL methods, understanding and systematically organizing them by their learning mechanisms, intended shift targets, and base algorithms for applicators' easy reference. It provides a reusable foundation for applying existing methods to new applications and developing new robust RL algorithms.
- **Enhanced evaluation platform.** Our extensible disruption toolbox covers a broad range of distribution shifts, including MDP-element shifts in observations, actions, dynamics, and reward/cost, as well as control-loop latency and task-semantic shifts. It supports diverse shift modes, including stochastic, adversarial, and parametric perturbations with non-stationary schedules, enabling both isolated and compound-shift evaluation. The same disruption interface further extends to massively parallel training in Isaac Lab (Mittal et al., 2025) and advanced vision–language–action (VLA) robotic tasks (Intelligence et al., 2025; Zhang et al., 2026).
- **An up-to-date capability map of robust RL algorithms.** Evaluation over 273 perturbation configurations reveals that robust methods outperform standard references on average, but both groups deteriorate markedly as shift severity increases. A method's claimed target does not reliably predict its strongest channel, gains vary across tasks and shift sources, with online mechanisms showing larger incremental effects than offline ones. By connecting mechanisms to their measured strengths, this map provides guidance for designing robust RL methods for deployment shifts.

2 *RobustRLlib*: An Algorithm-Centric Robust RL Benchmark

RobustRLlib is organized around *algorithm attribution*: it records how a robust RL method changes learning, which shift the method claims to address, and which base algorithm realizes the method. The disruptor toolbox then instantiates the claimed robustness independently of the learner, while the evaluation platform measures both common robustness and method-specific specialization. This separation makes each result traceable to an algorithmic mechanism and an evaluation condition rather than treating robust RL as a single undifferentiated class.

Regime	Family	Method	Claimed robustness	Base algorithm	Shifted-env rollout	Adversarial network	Learned model
Robust Offline	Learner-centric	RFQI	Dynamic	FQI			
		RORL	Observation	SAC			
		ATLA	Observation	IQL		✓	
	Data-centric	RSC-IQL	Semantics	IQL			✓
		RAMBO	Dynamic	SAC		✓	✓
		Generative models					
		ROMB	Dynamic	IQL		✓	✓
		FWM	Dynamic	IQL			✓
		PLR-PVL	Dynamic	IQL			✓
Robust Online	Learner-centric	ATLA	Observation	PPO		✓	
		ATLA-SA	Observation	PPO		✓	
		RSC	Semantics	SAC			✓
	Environment-centric	RARL	Dynamic	PPO / TRPO	✓	✓	
		DR	Dynamic	SAC / PPO	✓		
Robust Safe	Robust Safe	RAMU	Dynamic	PPO			
		SPiDR	Dynamic	PPO	✓		✓

Table 1: **Structural comparison of the robust methods in the library.** Online methods are grouped as learner-centric or environment-centric, offline methods as learner-centric or data-centric; generative models are a data-centric subfamily, and robust safe methods form a separate family. *Shifted-env rollout* marks rollouts collected from a deliberately perturbed physical environment. *Adversarial network* marks a learned adversary optimized against the policy. *Learned model* marks a trained next-state predictor used by the robustness mechanism. *Claimed robustness* records the intended shift target. A checkmark denotes a present feature; a blank denotes its absence.

Markov decision processes (MDP). We consider a finite-horizon MDP $\mathcal{M}_0 = (\mathcal{S}, \mathcal{A}, T, P_0, r_0)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, T is the horizon, and P_0 and r_0 denote the nominal transition kernel and reward function. At time t, the policy selects $a_t \sim \pi_t(\cdot \mid s_t)$, receives reward $r_{0,t}(s_t, a_t)$, and the environment transitions according to $s_{t+1} \sim P_{0,t}(\cdot \mid s_t, a_t)$. Classical robust MDPs replace the nominal transition model P_0 with a shift set $\mathcal{P}$ and seek a policy with high worst-case return (Iyengar, 2005; Nilim & El Ghaoui, 2005):

$$\max_{\pi} \inf_{P \in \mathcal{P}} J(\pi; P, r_0), \qquad J(\pi; P, r_0) = \mathbb{E}_{\pi,P}\left[\sum_{t=1}^{T} r_{0,t}(s_t, a_t)\right]. \tag{1}$$

For safety-constrained tasks, we augment the MDP with a cost function $c = \{c_t\}_{t=1}^{T}$ and a budget d, requiring $\mathbb{E}_{\pi,P}[\sum_{t=1}^{T} c_t(s_t, a_t)] \leq d$ in addition to maximizing task return. This cost interface is retained when shift is introduced, allowing robustness and safety to be evaluated jointly.

2.1 Robust RL algorithm library construction

The library has three tracks: *offline* RL learns from fixed data, *online* RL learns through continued environment interaction, and *safe* RL adds cost constraints, including robust-safe methods implemented here in the online setting. Within the offline and online tracks, robustness primarily enters through two routes: *learner-centric* methods change how experience is interpreted or optimized, while *data-centric* and *environment-centric* methods change the distribution available for learning. Table 1 summarizes each method's mechanism, claimed shift source, and base algorithm.

2.1.1 Robust offline algorithms

Offline RL learns from a fixed dataset without environment interaction, making limited data coverage and optimistic value estimates for unsupported actions central challenges. Robust offline RL must also maintain performance under distribution shift. We therefore group its methods by how they address these challenges: *learner-centric* methods modify the objective or update, whereas *data-centric* methods reshape the training distribution, including through data augmentation or generative models. For comparison, IQL (Kostrikov et al., 2021) and TD3+BC (Fujimoto & Gu, 2021) learn directly from logged transitions; MOPO (Yu et al., 2020b) adds model rollouts, and SynthER (Lu et al., 2023) adds diffusion-generated transitions.

Learner-centric mechanisms. These methods make policy learning less sensitive to uncertain transitions or perturbed observations by changing the value target or policy update.

- RFQI: worst-case Bellman update. RFQI (Panaganti et al., 2022) replaces the nominal fitted-Q target with a worst-case backup over transition ambiguity.
- RORL: local policy and value smoothing. RORL (Yang et al., 2022) combines pessimistic bootstrapping with smoothing around states to stabilize predictions under observation perturbations.
- ATLA: observation adversary. The offline ATLA adaptation (Zhang et al., 2021) trains an observation adversary alongside standard base algorithm, exposing policy updates to adverse inputs.

Data-centric mechanisms. These methods expand the experience available to the base learner through counterfactual transitions or model-generated rollouts.

- RSC: causal counterfactual rewrite. The offline RSC (Ding et al., 2023a) adaptation rewrites logged transitions using inferred causal mask to weaken spurious semantic correlations.
- RAMBO: adversarial model rollouts. RAMBO (Rigter et al., 2022) branches short synthetic rollouts from dataset states and adjusts its dynamics model adversarially within a data-fit constraint.

Generative model. These data-centric methods learn conditional models of transition dynamics and use generated next states or multi-step futures in policy learning, which potentially improve robustness under dynamics shifts. SynthER is excluded from this family because it simply upsamples the logged transition for data augmentation, rather than to model dynamics for policy learning.

- ROMB: maximin model ensemble. ROMB (Chen et al., 2025) optimizes against adverse transitions represented by an ensemble of learned models.
- FWM: flow-generated multi-step targets. FWM uses flow-matching to replace diffusion world model (Ding et al., 2024) and samples future trajectories to construct multi-step critic targets.
- PLR-PVL: worst-case world model selection. PLR-PVL (Berdica et al., 2024) searches a flow-model ensemble for adverse plausible futures during policy learning.

2.1.2 Robust online algorithms

Online robust RL methods preserve continued interaction with the environment but differ in where robustness enters training. We distinguish two primary routes. *Learner-centric* methods modify the policy input and learning objective while retaining the nominal physical transition law. *Environment-centric* methods instead partially get the shifted environment access and construct training rollouts under disturbances. Whether a method updates on-policy or off-policy remains an important implementation choice, so we record it through the base algorithm. PPO (Schulman et al., 2017), TRPO (Schulman et al., 2015) and SAC (Haarnoja et al., 2018) serve as the standard base algorithms in our experiments.

Learner-centric mechanisms. These methods train the policy to make stable decisions using adversarial observations, local regularization, or causal counterfactual experience.

- ATLA: observation adversary. ATLA (Zhang et al., 2021) alternates updates between the policy and a learned adversary that perturbs its observations.
- ATLA-SA: state-adversarial regularization. ATLA-SA (Zhang et al., 2021) adds recurrent history and SA-MDP regularization to stabilize decisions under perturbed observations.
- RSC: causal counterfactual replay. RSC (Ding et al., 2023a) infers causal dependencies and rewrites experience counterfactually to reduce reliance on spurious semantic features.

Environment-centric mechanisms. These methods build robustness through training trajectories generated under varied physical environments or external disturbances.

- RARL: external force adversary. RARL (Pinto et al., 2017) perturbs the environment with bounded forces selected by an adaptive adversary. The protagonist and adversary train in a two-player game, concentrating experience on the policy's current vulnerabilities.
- DR: domain randomization. DR (Peng et al., 2018) samples physical parameters across training episodes, exposing the policy to a prescribed envelope of transition dynamics.

2.1.3 ROBUST SAFE RL ALGORITHMS

Robust safe RL seeks to preserve task performance while satisfying cost constraints under dynamics uncertainty.

- RAMU: risk-sensitive reward and cost targets. RAMU (Queeney & Benosman, 2023) applies risk measures over transition-model uncertainty to both reward and cost Bellman targets.
- SPiDR: robust safety via penalized costs. SPiDR (As et al., 2026) measures disagreement among next-state predictions from domain-randomized models and penalizes uncertain safety outcomes.

2.2 EXPANDABLE DISRUPTOR TOOLBOX

Robust RL preserves the underlying task objective of standard RL by maximizing cumulative return but extends the single nominal MDP into a family of robust MDP with different types of shifts. These shifts intervene in different components of the agent–environment interaction, such as observations, actions, transition dynamics, rewards or costs, latency, and task semantics. The resulting robust RL problem therefore seeks a policy that performs reliably across these shifted interaction conditions.

2.2.1 SHIFT SOURCES

We classify a shift by where it enters the agent–environment interaction. The first four sources act directly on components of the MDP interface; latency and semantics then capture deployment factors that are not represented cleanly by a single nominal MDP component. Given $x_t = (s_t, \tilde{a}_t)$ and $y_t = (s_t, \tilde{a}_t, s_{t+1})$, we expose the six shift sources as below:

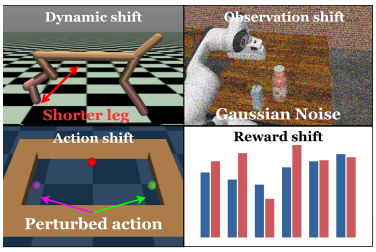

Figure 2: MDP-element shifts

Dynamic shift changes the environment response under the same state and executed action, $s_{t+1} \sim P_{\theta^p}(\cdot \mid x_t)$ where θ^p specifies physical-parameter changes, external disturbances, and latent process noise. Its operators cover physical changes including *internal dynamics* such as morphology, friction, gravity and *external disturbance* such as wind, external pushes, and latent noise. These variations represent physical mismatch and motivate dynamics randomization for sim-to-real control (Peng et al., 2018).

Observation shift changes the information available to the policy, $\tilde{o}_t \sim O_{\theta^o}(\cdot \mid s_t)$ and $a_t \sim \pi(\cdot \mid \tilde{o}_t)$, where θ^o specifies the perturbation applied to the policy observation. It represents sensor noise, bias, missing or imperfect feedback, and bounded observation attacks (Dulac-Arnold et al., 2020).

Action shift changes the command that reaches the plant, $\tilde{a}_t \sim A_{\theta^a}(\cdot \mid s_t, a_t)$, where θ^a specifies how the intended action is altered before execution. It captures execution noise, calibration error, saturation, opposition, and other actuator-side distortions (Tessler et al., 2019).

Reward/cost shift changes the feedback used for task learning or safety evaluation, $\tilde{r}_t \sim R_{\theta^r}(\cdot \mid y_t)$ and $\tilde{c}_t \sim C_{\theta^c}(\cdot \mid y_t)$, where $\theta^{rc} = (\theta^r, \theta^c)$ specifies the reward and cost perturbations. It represents noisy or biased supervision while keeping the underlying transition explicit.

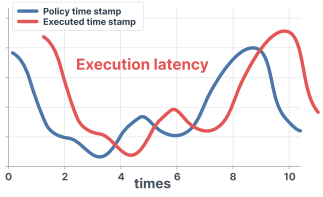

Figure 3: Latency shift.

Latency shift captures deployment-time mismatch in the timing of sensing, action execution, control, and VLA inference delay (Dulac-Arnold et al., 2020; Bouteiller et al., 2021; Tan et al., 2018). We treat latency as a composable deployment-shift source within this unified robust RL evaluation framework. Specifically, we expose observation delay, action delay, and control-period variation through $\tilde{o}_t = \mathcal{I}(\mathcal{H}_t, t - \delta_o), \tilde{a}_t = a_{t-\delta_a}, s_{t+1} \sim P_0^{(\kappa_t)}(\cdot \mid s_t, \tilde{a}_t)$, capturing delayed sensing, delayed execution, and variable control rates, respectively. Implementation details are provided in Appendix B.2.1.

Semantic shift changes task-relevant scenes or structures rather than merely adding task-agnostic noise. Such shifts can be represented as $o_t \sim O_{\theta^z}(\cdot \mid s_t), s_{t+1} \sim P_{\theta^z}(\cdot \mid x_t), (r_t, c_t) \sim (R_{\theta^z}, C_{\theta^z})(\cdot \mid y_t)$, where θ^z specifies appearance or task-semantic parameter changes.

Camera, lighting, color, blur, and distractor shifts act primarily through O_{θ^z}, whereas relocating a goal or object may also change the transition dynamics and task feedback. Such semantic shifts are increasingly important for visually grounded policies, where deployment may involve diverse task-specific scenes, including different objects, workspaces, viewpoints, layouts, or instructions (Brohan et al., 2023; Pumacay et al., 2024; Fei et al., 2025), potentially inducing semantic spurious correlations as an important deployment shift (Tobin et al., 2017; Ding et al., 2023a).

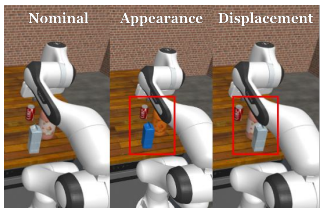

Figure 4: Semantic shift.

2.2.2 Shift modes and severity

A mode says *how* an intervention is built, and is one field of the specification tuple (channel, mode, severity, resampling rule, optional schedule, seed): severity says *how far* the intervention departs from nominal, and the last two say *how it evolves* within an episode. We retain the stochastic interface of Robust-Gymnasium (Gu et al., 2025), and extend it with several new modes as below:

- **Stochastic.** Samples signal errors or parameters from specified distributions.
- **Adversarial.** Computes bounded perturbations from the current states to counter the policy.
- **Parametric.** Applies a specified assignment, scale, offset, or translation at a severity.
- **Non-stationary.** Varies a mode's severity over time or triggers intermittent disturbances.
- **Composition.** Stacks compatible shifts into reproducible multi-shift conditions.

Together, these controls support both isolated shift sweeps and compound shift scenarios. The appendix B.2 provides the mode, operator, severity, resampling rule, and composition procedure.

2.2.3 Expandable to advanced robotics tasks

Because a shift is fully described by its specification tuple, a new backend needs only one adapter to the simulator's own primitives. We validate this on two deployment-facing platforms. For Isaac Lab (Mittal et al., 2025), the adapter realizes transition, action, and latency shifts on the vectorized GPU environment and exposes a joint-limit cost channel for every method's evaluation, so unconstrained and constrained learners are compared on one safety quantity. For a VLA policy (Intelligence et al., 2025) fine-tuned on the bimanual demonstrations (Zhang et al., 2026), the adapter injects appearance, proprioception, action, latency, and physical shifts into a closed-loop simulator.

3 Experimental Evaluation

The evaluation proceeds in three blocks on our library. §3.1 measures how much a robustness mechanism buys across the whole library and how that gain behaves as the shift grows. §3.2 isolates each shift source and composes them, to ask which shift a mechanism actually defends. §3.3 carries the same interface to deployment-scale humanoid and vision–language–action (VLA) systems.

Experimental setup. The library implements 22 algorithms: 6 standard references, 14 robust methods and 2 robust safe methods. The library-wide grid runs the first 20 algorithms across Locomotion, Manipulation, Navigation and Vehicle Control; the two robust safe methods enter on the advanced robotic tasks. Every method starts from the same nominal task specification, uses its native training procedure and prescribed budget, and is then frozen and evaluated on the same shift grid. The results use five training seeds and 20 paired evaluation episodes per condition. Exact environments, evaluation configurations, and protocols are in Appendix C.

Performance metric. For a comparable return across different tasks, we report the unclipped normalized score $\mathrm{S}(\bar{R}) = 100(\bar{R} - R_{\min})/(R_{\max} - R_{\min})$ as our evaluation metric, where $R_{\min}$ and $R_{\max}$ are fixed task-level reference returns shared by every method and condition. Appendix C.1 details the metrics.

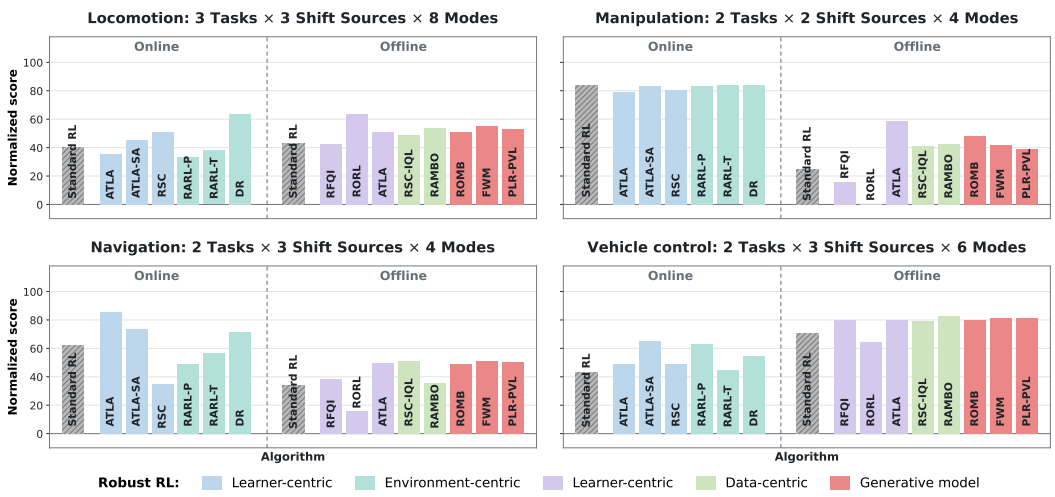

Figure 5: **The full library under controlled distribution shift.** Every coloured bar is one of the fourteen robust methods, keyed by mechanism family and online/offline regime. Scores average severities within a shift source, sources within a task, and then tasks with equal weight. The aggregation covers 20 algorithms across 220 distinct perturbation configurations.

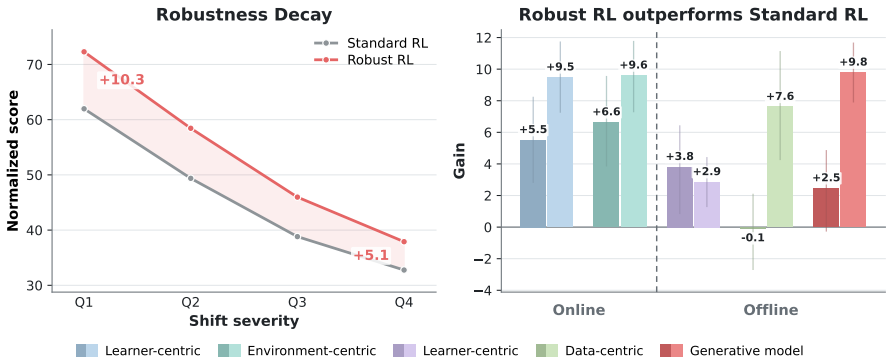

Figure 6: **Robustness across severity and mechanism families.** *Left*: pooled Standard and Robust scores across severity quartiles Q1–Q4 (Appendix C.2 gives the construction and the per-ladder assignment). *Right*: each family's gain over each member's own base algorithm (dark) and over the pooled Standard RL mean (light). The full results are in Appendix G.1.

3.1 Robustness of the Algorithm Library

Robust RL methods differ in their training assumptions and access to data and environments, making results from their original settings difficult to compare directly. We therefore evaluate their final policies on 220 representative robust RL tasks to identify their capability boundaries. Our first comparison covers the complete online and offline robust libraries including 20 algorithms over 220 perturbation configurations across four task families. Figure 5 shows each robust algorithm separately, while Figure 6 aggregates the same evaluations by severity and mechanism family to show both the robust–standard gap and each family's gain over its base algorithms.

Robust RL helps at every severity, but both degrade as shifts intensify. Across all four severity quartiles, robust methods achieve a higher pooled normalized score than standard RL. Their advantage remains positive but narrows from $+10.3$ points in Q1 to $+5.1$ in Q4, while both groups' scores decline as shifts intensify.

Dynamics-aware mechanisms are stronger under dynamics shifts. Environment-centric online methods and generative offline methods show some of the largest gains on physical shifts, consistent

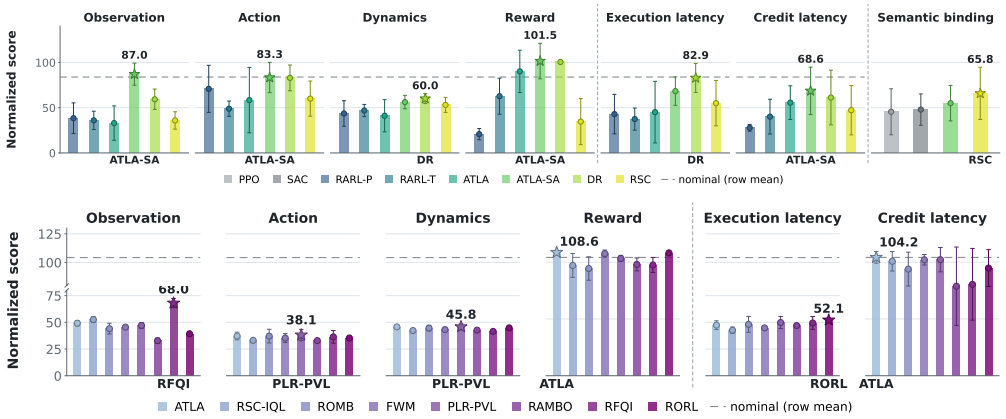

Figure 7: **Isolated shift evaluation.** The figure aggregates 14 robust and 2 standard RL methods across 44 distinct perturbation configurations. *Semantic binding* is evaluated only online, on the goal-conditioned causal door task.

with their exposure to varied dynamics or modeling of trajectories. Offline methods benefit more from the robustness of their base learners, suggesting that robustness reflects both the added robust mechanism and the strength of its base algorithm.

3.2 Isolated and Compound Deployment Shifts

We evaluate method-specific robustness through isolated channel sweeps across 44 perturbation configurations and reference-derived compound shifts. The isolated sweeps cover multiple perturbation modes. For compound tests, Hopper adapts the coupled physical, actuation, force, and delay factors of OmniH2O (He et al., 2024), while Pusher adapts the dynamics, observation, and control-rate factors of Peng et al. (2018), with goal displacement labeled separately. Appendix C.2 details the isolated tests and the compound scenarios.

Claimed channels do not fully predict empirical strengths. Figure 7 shows DR-SAC leads both dynamics and execution latency, while ATLA-SA-PPO leads observation and action; offline leaders vary across shift channels. A method's intended target therefore does not uniquely determine where it performs best. The measured ranking is also conditional on the task, mode and severity used to instantiate each shift.

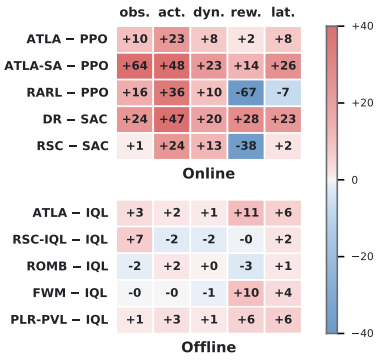

Figure 8: **Paired gain over base algorithm.** Aggregated across 5 shift sources and 44 configurations.

Online robust methods generally yield larger gains over their base learners than offline methods. Figure 8 shows that online mechanisms can produce large gains on most shift sources, whereas most offline differences remain close to their base algorithms. The smaller offline increments may partly reflect limited coverage of the tested perturbation modes in fixed datasets. Because offline RL relies on fixed data, methods often use conservative updates to limit value overestimation which potentially constrains robustness gains.

Dynamics are the common first shift, but the second varies. We test our strongest online (ATLA-SA-PPO) and offline robust (RORL) algorithms under compound shifts. Figure 9 shows that dynamics dominate both scenarios. It is the worst single source in the two tasks and what comes next depends on the task. Locomotion loses about three quarters of its score to latency, because delayed action on a balancing gait is a fall rather than a slower trajectory, while semantic shift on manipulation costs a quarter, because it moves the goal of the policy. A sim-to-real design therefore has to identify which channel is core to its task. Composition is worse than any of its parts where both algorithms drop below half of nominal, indicating poor capability for complex, multi-shift environments. The full evaluation results are in Appendix G.4.

3.3 Extension to advanced robotic tasks

We extend the benchmark from standard control to advanced robotics tasks, attaching the same disruptor interface through task-specific adapters which include a high-dimensional humanoid, a contact-rich manipulator and a VLA control task.

Robustness transfers return but not constraint satisfaction; without the second, a transfer is not deployable. We compare PPO, DR-PPO (Schwarke et al., 2025) and ATLA-PPO against the safe robust methods RAMU-PPO and SPiDR-PPO on Unitree G1 locomotion and Franka drawer manipulation in Isaac Lab (Mittal et al., 2025) (Figure 10). On G1 the two highest-scoring policies survive about a quarter of delayed episodes; on the drawer DR-PPO matches SPiDR's 99.6% success with 0% of its episodes inside the joint-limit cost budget against SPiDR's 84%. Only SPiDR-PPO keeps both metrics high on both tasks, so a higher performance is not the evidence that a method is ready for sim-to-real deployment. Full setups are in Appendix F.1.

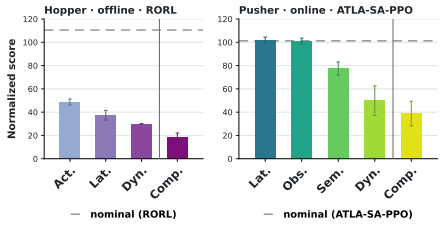

Figure 9: **Compound shifts, separated by shift source.** Each panel shows the strongest offline/online robust method (RORL and ATLA-SA-PPO), under shift sources in isolation and composition.

Sensitivity differs sharply across shift sources, and the semantic shift affects more in VLA task. We fine-tune $\pi_{0.5}$ on the *insert-mouse-battery* task (Figure 13) and drive it under eight shift axes. At the largest measured severity the policy still retains 0.95 under gravity but only 0.33 under image blur. Q1–Q4 rank severity (defined in Appendix C.2) within each axis' own ladder rather than on a scale shared across axes; blur at its largest severity moves only 13% of the image yet drops the policy to a third of nominal, while image brightness moves the same frames through the same channel at seven times that dose and costs almost nothing. Semantic shift is a core factor which affects more than physical shifts since a clear scene is the base of the action chunk.

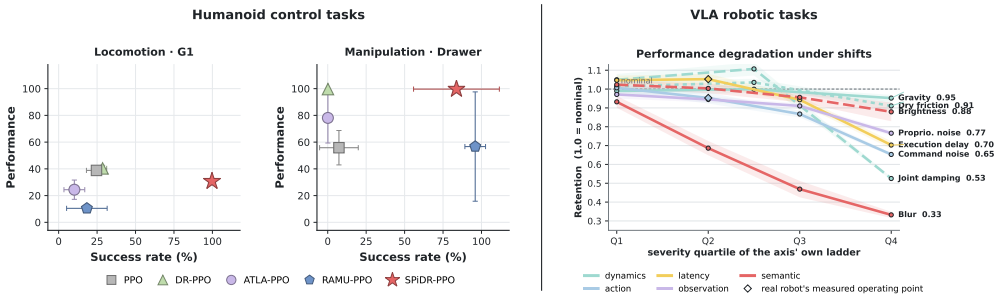

Figure 10: **Results on Isaac Lab and VLA tasks.** *Left:* one point per method, task performance against the safety success rate on G1 survival under action delay, and drawer episodes that stay inside the joint-limit cost budget. *Right:* what a VLA policy retains as each shift axis is pushed along its own severity ladder. The diamond marks each axis' operating point measured on the real platform.

4 Conclusion

In this work, we introduce *RobustRLlib*, an algorithm-centric library and evaluation platform for robust RL under deployment shifts. It integrates 16 robust methods, organized by robustness mechanism, claimed shift target, and base learner. A shared interface evaluates MDP-element shifts alongside control-loop latency and task semantics through isolated and compound tests and extends to advanced robotics tasks. Under a unified evaluation over 273 perturbation configurations, robust methods as a group outperform standard references. Shift-isolated and compound tests further reveal each mechanism's strengths and capability boundaries. Therefore, *RobustRLlib* provides a platform to support more comprehensive evaluation and guide the design of future robust RL methods.

AI USE STATEMENT

Generative AI tools were used to assist with code editing, data analyzing, and improving the readability of the manuscript. All AI-assisted outputs were reviewed and verified by the authors. Generative AI was not used to formulate research hypotheses, design the experimental methodology, or interpret experimental results. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Additional details are provided in the appendix. Appendix B.2 describes the implementation of disruption toolbox, Appendix D lists the tasks and environments, and Appendix C specifies the unified evaluation protocol and perturbation configurations. Training budgets and hyperparameters appear in Appendices F and H, respectively; full results of all the algorithms across per-task are reported in Appendix G.

REFERENCES

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

W Brian Arthur. Designing economic agents that act like human agents: A behavioral approach to bounded rationality. *The American economic review*, 81(2):353–359, 1991.

Yarden As, Chengrui Ray Qu, Benjamin Unger, Dongho Kang, Max van der Hart, Laixi Shi, Stelian Coros, Adam Wierman, and Andreas Krause. Spidr: A simple approach for zero-shot safety in sim-to-real transfer. *Advances in Neural Information Processing Systems*, 38:90882–90924, 2026.

Uljad Berdica, Kelvin Li, Michael Beukman, Alexander David Goldie, Perla Maiolino, and Jakob Nicolaus Foerster. Robust offline learning via adversarial world models. In *NeurIPS 2024 Workshop on Open-World Agents*, 2024.

Dimitris Bertsimas, Melvyn Sim, and Meilin Zhang. Adaptive distributionally robust optimization. *Management Science*, 65(2):604–618, 2019.

Mohak Bhardwaj, Tengyang Xie, Byron Boots, Nan Jiang, and Ching-An Cheng. Adversarial model for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:1245–1269, 2023.

Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *Advances in Neural Information Processing Systems*, 38:33383–33407, 2026.

Jose Blanchet, Miao Lu, Tong Zhang, and Han Zhong. Double pessimism is provably efficient for distributionally robust offline reinforcement learning: Generic algorithm and robust partial coverage. *Advances in Neural Information Processing Systems*, 36:66845–66859, 2023.

Yann Bouteiller, Simon Ramstedt, Giovanni Beltrame, Christopher J Pal, and Jonathan Binas. Reinforcement learning with random delays. In *ICLR*, 2021.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 international conference on robotics and automation (ICRA)*, pp. 8973–8979. IEEE, 2019.

Jiayu Chen, Le Xu, Aravind Venugopal, and Jeff Schneider. Policy-driven world model adaptation for robust offline model-based reinforcement learning. *arXiv preprint arXiv:2505.13709*, 2025.

Wenhao Ding, Laixi Shi, Yuejie Chi, and Ding Zhao. Seeing is not believing: Robust reinforcement learning against spurious correlation. *Advances in Neural Information Processing Systems*, 36: 66328–66363, 2023a.

Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023b.

Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning. *arXiv preprint arXiv:2402.03570*, 2024.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*, 2020.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Daniele Foffano, Alessio Russo, and Alexandre Proutiere. Adversarial diffusion for robust reinforcement learning. *Advances in Neural Information Processing Systems*, 38:68627–68658, 2026.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Vineet Goyal and Julien Grand-Clement. Robust markov decision processes: Beyond rectangularity. *Mathematics of Operations Research*, 48(1):203–226, 2023.

Shangding Gu, Laixi Shi, Muning Wen, Ming Jin, Eric Mazumdar, Yuejie Chi, Adam Wierman, and Costas Spanos. Robust gymnasium: A unified modular benchmark for robust reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 102014–102041, 2025.

Xinyang Gu, Yen-Jen Wang, and Jianyu Chen. Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer. *arXiv preprint arXiv:2404.05695*, 2024.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez, Konrad Zolna, Rishabh Agarwal, Josh S Merel, Daniel J Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *Advances in neural information processing systems*, 33:7248–7259, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Songyang Han, Sanbao Su, Sihong He, Shuo Han, Haizhao Yang, and Fei Miao. What is the solution for state adversarial multi-agent reinforcement learning? *arXiv preprint arXiv:2212.02705*, 2022.

Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

Siyi Hu, Yifan Zhong, Minquan Gao, Weixun Wang, Hao Dong, Xiaodan Liang, Zhihui Li, Xiaojun Chang, and Yaodong Yang. Marllib: A scalable and efficient multi-agent reinforcement learning library. *Journal of Machine Learning Research*, 24(315):1–23, 2023.

Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and JoÃĢo GM AraÃšjo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL `https://arxiv.org/abs/2504.16054`.

Garud N Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2): 257–280, 2005.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety gymnasium: A unified safe reinforcement learning benchmark. *Advances in Neural Information Processing Systems*, 36:18964–18993, 2023.

Olga Klopp, Karim Lounici, and Alexandre B Tsybakov. Robust matrix completion. *Probability Theory and Related Fields*, 169(1-2):523–564, 2017.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Joseph Peter Kozak, Ruizhe Zhang, Matthew Porter, Qihao Song, Jingcun Liu, Bixuan Wang, Rudy Wang, Wataru Saito, and Yuhao Zhang. Stability, reliability, and robustness of gan power devices: A review. *IEEE Transactions on Power Electronics*, 38(7):8442–8471, 2023.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Erwan Lecarpentier and Emmanuel Rachelson. Non-stationary markov decision processes, a worst-case approach using model-based reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

Haoran Li, Zicheng Zhang, Wang Luo, Congying Han, Yudong Hu, Tiande Guo, and Shichen Liao. Towards optimal adversarial robust q-learning with bellman infinity-error. *arXiv preprint arXiv:2402.02165*, 2024.

Zhongyu Li, Xuxin Cheng, Xue Bin Peng, Pieter Abbeel, Sergey Levine, Glen Berseth, and Koushil Sreenath. Reinforcement learning for robust parameterized locomotion control of bipedal robots. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2811–2817. IEEE, 2021.

Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. In *International conference on machine learning*, pp. 3053–3062. PMLR, 2018.

Yongyuan Liang, Yanchao Sun, Ruijie Zheng, and Furong Huang. Efficient adversarial training without attacking: Worst-case-aware robust reinforcement learning. *Advances in neural information processing systems*, 35:22547–22561, 2022.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Qing Liu, Yulan Li, and Katherine Odem-Davis. On robustness of noninferiority clinical trial designs against bias, variability, and nonconstancy. *Journal of biopharmaceutical statistics*, 25(1):206–225, 2015.

Zuxin Liu, Zijian Guo, Zhepeng Cen, Huan Zhang, Jie Tan, Bo Li, and Ding Zhao. On the robustness of safe reinforcement learning under observational perturbations. *arXiv preprint arXiv:2205.14691*, 2022.

Cong Lu, Philip J Ball, Jack Parker-Holder, Michael A Osborne, and Stephen J Roberts. Revisiting design choices in offline model-based reinforcement learning. *arXiv preprint arXiv:2110.04135*, 2021.

Cong Lu, Philip Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36:46323–46344, 2023.

Miao Lu, Han Zhong, Tong Zhang, and Jose Blanchet. Distributionally robust reinforcement learning with interactive data collection: Fundamental hardness and near-optimal algorithms. *Advances in Neural Information Processing Systems*, 37:12528–12580, 2024.

Jiafei Lyu, Kang Xu, Jiacheng Xu, Mengbei Yan, Jingwen Yang, Zongzhang Zhang, Chenjia Bai, Zongqing Lu, and Xiu Li. Odrl: A benchmark for off-dynamics reinforcement learning. *Advances in Neural Information Processing Systems*, 37:59859–59911, 2024.

Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy poisoning in batch reinforcement learning and control. *Advances in Neural Information Processing Systems*, 32, 2019.

A Rupam Mahmood, Dmytro Korenkevych, Gautham Vasan, William Ma, and James Bergstra. Benchmarking reinforcement learning algorithms on real-world robots. In *Conference on robot learning*, pp. 561–591. PMLR, 2018.

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Adversarially robust policy learning: Active construction of physically-plausible perturbations. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3932–3939. IEEE, 2017.

Daniel J Mankowitz, Nir Levine, Rae Jeong, Yuanyuan Shi, Jackie Kay, Abbas Abdolmaleki, Jost Tobias Springenberg, Timothy Mann, Todd Hester, and Martin Riedmiller. Robust reinforcement learning for continuous control with model misspecification. *arXiv preprint arXiv:1906.07516*, 2019.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025. URL `https://arxiv.org/abs/2511.04831`.

Adithya Mohan, Daniel Kriegl, and Torsten Schön. Road-rl: A unified library and benchmark for robust adversarial reinforcement learning. *arXiv preprint arXiv:2606.29867*, 2026.

Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

Tuomas Oikarinen, Wang Zhang, Alexandre Megretski, Luca Daniel, and Tsui-Wei Weng. Robust deep reinforcement learning through adversarial loss. *Advances in Neural Information Processing Systems*, 34:26156–26167, 2021.

Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, and Mohammad Ghavamzadeh. Robust reinforcement learning using offline data. *Advances in neural information processing systems*, 35:32211–32224, 2022.

Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, and Mohammad Ghavamzadeh. Bridging distributionally robust learning and offline rl: An approach to mitigate distribution shift and partial data coverage. *arXiv preprint arXiv:2310.18434*, 2023.

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. IEEE, 2018.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International conference on machine learning*, pp. 2817–2826. PMLR, 2017.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024.

Rong-Jun Qin, Xingyuan Zhang, Songyi Gao, Xiong-Hui Chen, Zewen Li, Weinan Zhang, and Yang Yu. Neorl: A near real-world benchmark for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:24753–24765, 2022.

James Queeney and Mouhacine Benosman. Risk-averse model uncertainty for distributionally robust safe reinforcement learning. *Advances in Neural Information Processing Systems*, 36:1659–1680, 2023.

Antonin Raffin and Freek Stulp. Generalized state-dependent exploration for deep reinforcement learning in robotics. *Arxiv*, 2020.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of machine learning research*, 22(268):1–8, 2021.

Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Clemens Schwarke, Mayank Mittal, Nikita Rudin, David Hoeller, and Marco Hutter. Rsl-rl: A learning library for robotics research. *arXiv preprint arXiv:2509.10771*, 2025.

Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv preprint arXiv:2403.10506*, 2024.

Chung-En Sun, Sicun Gao, and Tsui-Wei Weng. Breaking the barrier: Enhanced utility and robustness in smoothed drl agents. *arXiv preprint arXiv:2406.18062*, 2024.

Ke Sun, Yi Liu, Yingnan Zhao, Hengshuai Yao, Shangling Jui, and Linglong Kong. Exploring the training robustness of distributional reinforcement learning against noisy state observations. *arXiv preprint arXiv:2109.08776*, 2021.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Kai Liang Tan, Yasaman Esfandiari, Xian Yeow Lee, Soumik Sarkar, et al. Robustifying reinforcement learning agents via action space adversarial training. In *2020 American control conference (ACC)*, pp. 3959–3964. IEEE, 2020.

Takumi Tanabe, Rei Sato, Kazuto Fukuchi, Jun Sakuma, and Youhei Akimoto. Max-min off-policy actor-critic method focusing on worst-case robustness to model misspecification. *Advances in neural information processing systems*, 35:6967–6981, 2022.

Chen Tessler, Yonathan Efroni, and Shie Mannor. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pp. 6215–6224. PMLR, 2019.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty. *science*, 185(4157):1124–1131, 1974.

Yue Wang, Fei Miao, and Shaofeng Zou. Robust constrained reinforcement learning. *arXiv preprint arXiv:2209.06866*, 2022.

Jiayi Weng, Huayu Chen, Dong Yan, Kaichao You, Alexis Duburcq, Minghao Zhang, Yi Su, Hang Su, and Jun Zhu. Tianshou: A highly modularized deep reinforcement learning library. *Journal of Machine Learning Research*, 23(267):1–6, 2022.

Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

Zikang Xiong, Joe Eappen, He Zhu, and Suresh Jagannathan. Defending observation attacks in deep reinforcement learning via detection and denoising. *arXiv preprint arXiv:2206.07188*, 2022.

Huan Xu and Shie Mannor. Distributionally robust markov decision processes. *Advances in Neural Information Processing Systems*, 23, 2010.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *Advances in neural information processing systems*, 35:23851–23866, 2022.

Rui Yang, Jie Wang, Guoping Wu, and Bin Li. Uncertainty-based offline variational bayesian reinforcement learning for robustness under diverse data corruptions. *Advances in Neural Information Processing Systems*, 37:39748–39783, 2024.

Omar G. Younis, Rodrigo Perez-Vicente, John U. Balis, Will Dudley, Alex Davey, and Jordan K Terry. Minari, 2024. URL `https://doi.org/10.5281/zenodo.13767625`. Software, version 0.5.0.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020a.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in neural information processing systems*, 33:14129–14142, 2020b.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in neural information processing systems*, 33:21024–21037, 2020.

Huan Zhang, Hongge Chen, Duane S Boning, and Cho-Jui Hsieh. Robust reinforcement learning on state observations with learned optimal adversary. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=sCZbhBvqQaU`.

Shiduo Zhang, Yue Wang, Haonan Chang, Hang Zhao, Yicheng Liu, Vitor Guizilini, Andreea Bobu, Andrew Wagenmaker, Anushri Dixit, Chao Yu, Dhruv Shah, and Max Simchowitz. Post-training for robotics foundation models dataset and challenge. RSS 2026 Workshop & Challenge, 2026. URL `https://posttraining-for-robotics.github.io`.

Adil Zouitine, David Bertoin, Pierre Clavier, Matthieu Geist, and Emmanuel Rachelson. Rrls: Robust reinforcement learning suite. *arXiv preprint arXiv:2406.08406*, 2024.

A Related Work

Robust RL problems under diverse shifts. Robust RL addresses diverse shifts entering different components of the agent–environment interaction. We organize prior work into six corresponding channels. *Observation shift* corrupts the policy input through sensing errors or bounded attacks (Zhang et al., 2020; 2021; Oikarinen et al., 2021), whereas *action shift* replaces or perturbs the command executed by the plant (Tessler et al., 2019; Tan et al., 2020). *Dynamic shift* covers both ambiguity sets over transition kernels (Nilim & El Ghaoui, 2005; Iyengar, 2005; Xu & Mannor, 2010; Wiesemann et al., 2013; Goyal & Grand-Clement, 2023) and concrete deployment mismatches in physical parameters or external forces (Rajeswaran et al., 2016; Peng et al., 2018; Pinto et al., 2017; Mankowitz et al., 2019). *Reward and cost shift* changes the feedback used for task optimization or safety, motivating robust reward criteria, constraint-aware learning, and reward-poisoning analysis (Xu & Mannor, 2010; Ma et al., 2019; Queeney & Benosman, 2023). Beyond these MDP-element channels, *latency shift* changes when observations and commands arrive or how long an action is held (Bouteiller et al., 2021; Peng et al., 2018; Tan et al., 2018), while *semantic shift* changes appearance or task-relevant relationships among visual cues, objects, and goals (Tobin et al., 2017; Ding et al., 2023a). Non-stationarity is a cross-channel property that makes shift evolve over time (Lecarpentier & Rachelson, 2019; Zouitine et al., 2024), and compound settings may activate several channels simultaneously (Mandlekar et al., 2017). Because these channels intervene at different points of the interaction loop, robustness to one does not imply robustness to the others.

Robust RL algorithms. Robust RL algorithms impose robustness through two main routes: modifying the *learning objective* or modifying the *training distribution*. How these routes are realized depends on whether additional environment interaction is available. Online learning must explore effectively while collecting trajectories that cover plausible deployment conditions (Lu et al., 2024). On the objective side, learner-based methods seek a robust policy through adversarial optimization, worst-case value objectives, sensitivity regularization, or local smoothing (Zhang et al., 2020; 2021; Li et al., 2024; Liang et al., 2022; Sun et al., 2024). On the data side, environment-based methods collect a broader transition distribution through physical randomization, model ensembles, or adaptive disturbances and parameter adversaries, while counterfactual replay expands experience without changing the physical plant (Rajeswaran et al., 2016; Pinto et al., 2017; Peng et al., 2018; Mankowitz et al., 2019; Tanabe et al., 2022; Ding et al., 2023a). Robust safe RL extends these routes by requiring task performance and cost constraints to remain reliable under uncertainty (Liu et al., 2022; Wang et al., 2022; Queeney & Benosman, 2023). With fixed offline data, insufficient coverage replaces active exploration as the central difficulty. Learner-centric methods control extrapolation through robust or pessimistic Bellman updates and conservative smoothing (Panaganti et al., 2022; Blanchet et al., 2023; Yang et al., 2022; Panaganti et al., 2023). Data-centric methods instead broaden or reshape the logged distribution through counterfactual rewriting, adversarial model rollouts, synthetic transitions, or uncertainty-based reweighting (Ding et al., 2023a; Rigter et al., 2022; Bhardwaj et al., 2023; Lu et al., 2023; Yang et al., 2024). Generative models are a structured instance of this data-centric route: they explicitly model the dynamics and generate plausible multi-step futures or alternative worlds under distribution shift (Ding et al., 2024; Berdica et al., 2024; Foffano et al., 2026). Accordingly, *RobustRLlib* records the two organizing dimensions for every method the shift source it targets and whether robustness is imposed through the objective or the effective data distribution and evaluates both under a unified protocol.

Related RL benchmarks. RL benchmarks standardize different notions of progress and can be organized by the primary object they hold fixed. *Data-oriented benchmarks* fix datasets and access conditions (Fu et al., 2020; Gulcehre et al., 2020; Qin et al., 2022); *task-oriented benchmarks* fix environments and target capabilities, varying from continuous control to advanced robotics tasks (Duan et al., 2016; Yu et al., 2020a; Sferrazza et al., 2024; Ji et al., 2023; Pumacay et al., 2024); and *algorithm-oriented libraries and benchmarks* standardize implementations and training pipelines (Liang et al., 2018; Hu et al., 2023; Raffin et al., 2021; Huang et al., 2022; Weng et al., 2022). *RobustRLlib* is an algorithm-centric benchmark for robust RL. Its closest prior benchmarks are RRLS, which focuses on parameterized dynamics uncertainty in continuous control (Zouitine et al., 2024), and Robust-Gymnasium, which emphasizes modular disruption and task construction (Gu et al., 2025). In comparison, *RobustRLlib* provides a broader and systematically organized robust RL algorithm library, more deployment-oriented uncertainties including timing, semantics, and compound

shifts, and evaluation that extends from standard control to advanced high-dimensional robotic tasks. Other benchmarks address narrower settings such as off-dynamics transfer, real-world operating stressors, or adversarial robustness (Lyu et al., 2024; Dulac-Arnold et al., 2020; Mohan et al., 2026).

Sim-to-real transfer and deployment-facing robotics. Sim-to-real transfer is a central robustness problem because mismatches in the plant and control loop change the trajectories reached at deployment, while repeated correction on physical hardware is costly. Recent deployment-facing systems make this challenge more demanding: Humanoid-Gym and OmniH2O pursue transferable high-dimensional locomotion and whole-body control (Gu et al., 2024; He et al., 2024), SPiDR studies safety-preserving transfer (As et al., 2026), and VLA systems such as RT-2 extend robot policies to semantically diverse manipulation (Brohan et al., 2023). Across these settings, a common strategy is to make the training simulator approximate deployment variation. Dynamics randomization exposes policies to changes in mass, friction, contacts, and actuator properties for control and dexterous manipulation (Peng et al., 2018; Andrychowicz et al., 2020), while explicit actuator and latency modeling is critical for transferring agile locomotion (Tan et al., 2018). Adaptive transfer methods further refine the randomization distribution using real experience or infer changing dynamics online from recent interaction (Chebotar et al., 2019; Kumar et al., 2021), while real-time execution methods treat policy inference latency as part of the VLA control problem (Black et al., 2026). Such perturbations, however, are usually assembled inside task-specific pipelines, making their coverage difficult to reuse or compare and leaving substantial simulator tuning to human-in-the-loop iteration. *RobustRLlib* connects the same modular disruptor interface to humanoid, safety-constrained, and VLA pipelines, allowing physical, actuation, sensing, appearance, transition, and latency shifts to be composed within their native simulation environments. This design moves systematic deployment stress construction and diagnosis into simulation, with the goal of reducing manual iteration before hardware deployment.

B Benchmark Construction

B.1 Benchmark Comparison

Benchmark	Robust algorithms	Shifts				Task expansion
		MDP	Latency	Semantic	Compound	
RLlib (Liang et al., 2018)	0	0/4	✗	✗	✗	✓
RRLS (Zouitine et al., 2024)	4	2/4	✗	✗	❍	✓
ODRL (Lyu et al., 2024)	0	1/4	✗	✗	✗	✗
Robust-Gymnasium (Gu et al., 2025)	4	4/4	✗	✓	✓	✓
RWRL Suite (Dulac-Arnold et al., 2020)	0	3/4	✓	✗	✓	❍
RoAd-RL (Mohan et al., 2026)	0	1/4	✗	✗	✗	✓
***RobustRLlib* (ours)**	**16**	**4/4**	✓	✓	✓	✓

Table 2: **Comparison with RL libraries and robustness benchmarks.** *Robust algorithms* counts robust or robust safe RL algorithms evaluated by each benchmark; standard reference learners are excluded. A checkmark denotes explicit, evaluated support; ❍ denotes partial or scope-limited support; ✗ indicates that the capability is absent or not demonstrated in the cited release. *MDP* counts, out of four, the MDP components on which shift is evaluated: observations, actions, transitions, and reward/cost signals. *Compound* requires multiple shift sources to be active jointly. *Task expansion* indicates an extensible interface for carrying the benchmark to additional task backends.

Table 2 compares evaluated capabilities with 6 related benchmarks. Algorithm counts include robust and robust safe RL method, setting configurations evaluated in the cited release, but exclude standard learners and inference-time defenses. Shift entries require reported evaluations, whereas task expansion denotes a documented interface for adding task backends. These criteria distinguish a benchmark's demonstrated scope from its potential extensibility.

The compared systems standardize different parts of robust RL evaluation. RLlib provides general-purpose algorithm infrastructure; RRLS and ODRL compare methods primarily under dynamics-related uncertainty (Liang et al., 2018; Zouitine et al., 2024; Lyu et al., 2024). Robust-Gymnasium

offers broad disrupted environments, including semantic tasks, while the RWRL Suite evaluates practical stressors such as latency and their combination (Gu et al., 2025; Dulac-Arnold et al., 2020). RoAd-RL focuses on observation attacks and inference-time defenses (Mohan et al., 2026). Rather than claiming these capabilities individually as new, *RobustRLlib* combines a structured online, offline, and safe robust RL library with isolated and composed shift evaluations under a shared protocol. Its task-specific adapters further demonstrate how the evaluation methodology extends from standard control to advanced robotics tasks such as Isaac Lab and a VLA pipeline.

B.2 Disruptor toolbox: shift modes, operators and implementations

A perturbation is specified by (**channel**, **mode**, **severity**, **resampling rule**, **optional schedule**, **seed**). The **channel** identifies the shift source in Section 2.2, the **mode** specifies how the intervention is constructed, and **severity** sets its severity in the operator's natural units. The **resampling rule** determines whether a value is redrawn at each step, held for an episode, or fixed. The optional **schedule** controls when the intervention is active and how its severity evolves, while the **seed** makes stochastic draws reproducible. The main text summarizes five intervention families; here we detail their source-specific operators, indexed in Table 3.

B.2.1 Shift source: Latency

We treat latency as a composable deployment-shift source within this unified robust RL evaluation framework. Specifically, we expose observation delay, action delay, and control-period variation through $\tilde{o}_t = \mathcal{I}(\mathcal{H}_t, t - \delta_o), \tilde{a}_t = a_{t-\delta_a}, s_{t+1} \sim P_0^{(\kappa_t)}(\cdot \mid s_t, \tilde{a}_t)$, capturing delayed sensing, delayed execution, and variable control rates, respectively. The three operators act at different points of one decision and are implemented accordingly.

- **Observation delay.** A timestamped history $\mathcal{H}_t$ is queried at $t - \delta_o$ using linear interpolation, so sub-step delays are not rounded away (Tan et al., 2018). If the requested time precedes reset, the history returns the reset observation.
- **Action delay.** An integer buffer executes the command from δ_a control steps earlier, $\tilde{a}_t = a_{\max(t-\delta_a,0)}$, using the initial command to warm-start the buffer.
- **Control-period variation.** Following Peng et al. (2018), we set $\Delta t_t = \Delta t_0 + \xi_t$, where $\xi_t \sim \mathrm{Exp}(\lambda)$; λ is drawn per episode and ξ_t per decision. One command is held for $\kappa_t = \mathrm{round}(\Delta t_t / h_{\mathrm{sim}})$ simulator substeps, changing its duration rather than its arrival time.

B.2.2 Operator details

- **Stochastic.** Signal noise is redrawn per step, whereas **bias** persists for an episode and dynamics parameters are sampled at reset. **loguniform** samples in log space to balance reciprocal multipliers. Latency draws act on command buffers, timestamped observations, or action-hold duration.
- **Adversarial.** The observation operator searches for an input that changes the current actor's output within radius ϵ; it requires a differentiable feed-forward actor. The three action operators respectively oppose the chosen command, rotate a 2-D command, or suppress its goal-directed component. Commands are clipped to the action space before perturbation.
- **Parametric.** These operators apply a specified value rather than draw one. **set** handles parameters whose nominal value is zero, while **translate** expresses a displacement in physical units. Semantic operators change appearance or task referents; translating a non-colliding goal changes its referent without changing physical dynamics.
- **Temporal controls.** A **schedule** changes severity over time, whereas latency operators change when signals enter the control loop. **push** creates an intermittent event. For feedback **delay**, interval release sums rewards over k steps, while shifted release delivers each reward k steps later; both flush pending feedback at episode end.
- **Composition.** An ordered list specifies wrapper order explicitly. Each specification uses its own seeded random stream, making compound conditions reproducible.

Source	Operator	Intervention	Severity and refresh
Stochastic sampling			
Observation, action, reward/cost	**gauss**, **uniform**	Add componentwise signal noise	Standard deviation or sampling range; per step
Observation	**relative**	Scale noise by each feature's reference spread, $\sigma_i = c\hat{\sigma}_i$	Relative scale c; per step
Observation	**bias**	Draw and hold a sensor offset	Offset range; per episode
Dynamics	**gauss**, **uniform**, **loguniform**	Sample physical parameters or their multipliers	Parameter range; per reset
Latency	**buffer**, **interp**	Delay a buffered command or interpolate timestamped observations	Control steps or milliseconds; configured refresh
Latency	**substep**	Vary how many simulator substeps hold one policy action	Substeps; per policy decision
Adversarial selection			
Observation	**adversarial**	Search for a policy-dependent perturbation within an ℓ_∞ ball	Radius ϵ; per step
Action	**oppose**	Subtract a bounded component along the chosen action	Opposition magnitude; per step
Action	**rotate**	Change action direction while preserving magnitude; 2-D actions only	Rotation angle; per step
Action	**oppose_goal**	Remove part of the goal-directed action component	Opposition strength; per step
Parametric assignment			
Observation, action, reward/cost	**shift**	Add a specified signal offset	Signal units; fixed
Dynamics	**scale**, **set**	Multiply or assign a named model parameter	Ratio or parameter value; fixed
Dynamics, semantics	**translate**	Move a named body, object, or goal	Metric displacement; fixed
Semantics	**hue**, **tint**, **swap**, **brightness**, **blur**, **translate**	Change appearance or relocate a named goal/object	Hue angle, tint colour, named swap, brightness factor, blur radius (px), or displacement (m); fixed
Temporal controls			
Any compatible source	**schedule**	Vary an operator's severity by $g(t)$	Ramp, step, or sinusoidal profile
Dynamics	**push**	Apply an intermittent force to a named body	Force, duration, and interval; per event
Reward/cost	**delay**	Release accumulated feedback every k steps or shift it by k steps	Delay k; flush at episode end
Composition			
Compatible sources	**ordered list**	Stack specifications with independent random streams	Explicit order and seeds

Table 3: **Shift-operator index.** The first three families construct interventions; temporal controls and composition determine how interventions evolve and combine. Each row gives the compatible source, operator, intervention, and severity or refresh rule.

C Evaluation protocol

C.1 Metrics

Environment	shift axis	Unit	$Q1$	$Q2$	$Q3$	$Q4$
Online						
Hopper-v5,	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
HalfCheetah-v5,	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
Walker2d-v5	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
FetchReach-v4	Morphology	×	0.8	1.2	1.4	1.6
	Actuator gear	×	0.4	0.2	0.1	2
PointMaze_UMaze-v3	Actuator gear	×	0.8	1.2	1.4	0.6
	Dry friction	N	10	20	30	40
	Adversarial action	–	oppose (0.6)	oppose-goal (0.6)	oppose-goal (0.8)	rotate (70)
CarRacing-v2	Morphology	×	0.8, 1.2	0.6	1.4	1.6, 1.8
	Actuator gear	×	0.8, 1.2	0.6	1.4	0.4, 1.6
	Surface friction	×	0.9, 1.1	1.2	0.8	0.6, 0.7
Offline						
Hopper-v5,	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
HalfCheetah-v5,	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
Walker2d-v5	Gravity	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Morphology	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
	Actuator gear	×	0.9, 1.1	0.8, 1.2	0.7, 1.3	0.6, 1.4
AdroitHandDoor-v1	Actuator gear	×	0.8	1.2	1.4	0.6
	Latch friction	×	2	3	4	5
PointMaze_Open-v3	Actuator gear	×	0.4	0.2	0.1	0.05
	Dry friction	N	30	45	60	75
	Adversarial action	–	oppose-goal	rotate	rotate (90)	oppose
LunarLander Continuous-v3	Gravity	×	0.9, 1.1	1.15	0.85	0.75, 1.25
	Engine power	×	0.9, 1.1	0.75	1.25	0.6, 1.4
	Wind	N	5, 10	15	20	25, 30

Table 4: **The 220 evaluation configuration and its severity quartiles.** Each row is one shift axis on one environment; the four cells list every non-nominal rung evaluated on that axis, grouped by severity quartile. The nominal setting is measured separately on every axis and belongs to no quartile. Q indexes displacement from nominal rather than difficulty: for a scalar axis it is the quartile of $|\text{parameter} - \text{nominal}|$. Adversarial-action rungs are discrete attack modes with no scalar distance from nominal, so they are ordered instead by how far each mode depresses a held-out standard-RL reference pool, and are written *mode* (*parameter*).

Normalized score. To compare standard-control tasks with different native metrics, let $\bar{R}$ denote the mean raw task score over evaluation episodes. We report the unclipped normalized score

$$\mathrm{S}(\bar{R}) = 100\frac{\bar{R} - R_{\min}}{R_{\max} - R_{\min}}, \tag{2}$$

Ch.	Cell	Mode	Setting
Frozen-policy channels — five equally weighted cells each, Hopper-v5			
θ^o	`obs_rel_0.10`	stochastic, per step	Gaussian at $0.10\,\sigma_i$, σ_i the dimension's own std
θ^o	`obs_uni_0.10`	stochastic, per step	$\mathcal{U}[-0.10,\ 0.10]$
θ^o	`obs_gauss_0.10`	stochastic, per step	$\sigma = 0.10$
θ^o	`mad_0.02`	adversarial, per step	MAD gradient attack, $\epsilon = 0.02$
θ^o	`mad_0.05`	adversarial, per step	MAD gradient attack, $\epsilon = 0.05$
θ^a	`act_uni_0.10`	stochastic, per step	$\mathcal{U}[-0.10,\ 0.10]$ on the clipped action
θ^a	`act_gauss_0.10`	stochastic, per step	$\sigma = 0.10$ on the clipped action
θ^a	`act_offset_0.05`	static bias	$+0.05$ on every actuator
θ^a	`act_oppose_0.10`	adversarial, per step	$\epsilon = 0.10$ against the action just chosen
θ^a	`cmd_oppose_0.20`	adversarial, per step	$\epsilon = 0.20$, applied to the command before clipping
θ^p	`gear_static_0.8`	static parametric	actuator gear $\times\, 0.8$ for the whole episode
θ^p	`gear_ramp_0.8`	non-stationary	gear $1.0 \rightarrow 0.8$, linear over 1000 steps
θ^p	`gear_sine_p500`	non-stationary	gear cycling in $[0.8,\ 1.2]$, period 500 steps
θ^p	`push_60`	event-driven	60 N on the torso, 12 steps every 250, sign random
θ^p	`trans_t1`	stochastic kernel	state noise $\sigma_q = 10^{-3}$, $\sigma_v = 10^{-2}$ per step
θ^τ	`obs_delay_4ms`	latency, observation	delay drawn from $[2,\ 4]$ ms, interpolated
θ^τ	`obs_delay_8ms`	latency, observation	delay drawn from $[4,\ 8]$ ms
θ^τ	`obs_delay_16ms`	latency, observation	delay drawn from $[8,\ 16]$ ms
θ^τ	`act_delay_8ms`	latency, action	fixed buffer of 1 control step (8 ms)
θ^τ	`act_delay_rand1`	latency, non-stationary	buffer redrawn every step, $\{0, 1\}$ steps
Training-time channels — the shift acts during learning, the evaluation is nominal			
θ^r	Gaussian, uniform, constant	signal corruption	three arms matched at $\sigma = 0.5$ (uniform ± 0.866)
θ^τ	credit ladder	interval reward release	$k \in \{4, 16, 32, 64\}$ control steps

Table 5: **44 isolated shift sweep configuration.** The 22 fixed condition sets are instantiated separately for online and offline evaluation, yielding 44 perturbation configurations used to compute the channel scores in Section 3.2: five equally weighted cells per frozen-policy channel, and the training-time arms for reward corruption and credit latency. Equal condition counts make the channels comparable; the selected conditions do not form exhaustive severity ladders. A control step is 8 ms (`frame_skip` 4×2 ms). Both ATLA rows use three θ^o cells rather than five: a recurrent policy has no feed-forward differentiable actor, so the MAD condition is not evaluated for these methods.

where $R_{\min}$ and $R_{\max}$ are fixed task-level reference scores shared by all methods, seeds, and shift conditions. We use published D4RL references where available and otherwise use task-defined or reference-policy anchors; all values are listed in Table 7. Normalization precedes cross-task aggregation. For the standard-control experiments, we map each task's native metric to a common comparable scale.

Performance and safety. We report performance and safety separately on the advanced robotic tasks. For G1, performance is PPO-reference-normalized return over five out-of-envelope conditions, while safety is the fraction of episodes surviving the 20 s horizon under 20–80 ms action delays. For Drawer, performance is the fraction of episodes opening the drawer beyond 0.30 m over fourteen out-of-envelope conditions; safety is the fraction whose cumulative joint-limit cost remains below the per-episode budget of 20. We average episodes within each training seed, then report the mean and standard deviation across seeds.

C.2 Evaluation grids

The library-wide grid varies one shift axis at a time across online and offline tasks, grouping its non-nominal settings into four task–axis-specific severity quartiles (Table 4). The grid is fixed before any method is compared and is the same for every algorithm: 220 non-nominal conditions, 110 online and 110 offline, plus the nominal setting of every axis. The isolated-channel grid instead varies intervention modes within a shift source, allowing method-level specialization to be diagnosed

Channel	Factor	Draw	Setting
Hopper-v5 ← OmniH2O (He et al., 2024)			
Dynamics	Foot/link friction	log-uniform, per episode	$\times[0.38, 1.06]$
Dynamics	Link mass	log-uniform, per episode	$\times[0.81, 1.17]$
Dynamics	Actuator gear	log-uniform, per episode	$\times[0.84, 1.14]$
Dynamics	Torso push	onset phase per episode, sign per push	119 N, every 625 steps, 10 steps
Action	Torque noise	Gaussian, per step, pre-clip	$\sigma = 0.06$
Latency	Action delay	uniform integer, per episode	$[1, 2]$ steps
Pusher-v5s ← Peng et al. (2018) Table I			
Dynamics	Link mass	log-uniform, per episode	$\times[0.03, 32]$
Dynamics	Object mass	log-uniform, per episode	$\times[0.11, 3.8]$
Dynamics	Joint damping	log-uniform, per episode	$\times[0.02, 1.8\times10^{3}]$
Dynamics	Actuator gear	log-uniform, per episode	$\times[0.18, 5.7]$
Latency	Control period	λ per episode, $\mathrm{d}t$ per step	$\mathrm{d}t = \mathrm{d}t_0 + \mathrm{Exp}(1/\lambda)$, $\lambda \in [50, 400]$
Observation	Sensor noise	Gaussian, per step, per-feature std	$0.125\,\sigma_i$
Semantics	Goal displacement	deterministic	0.25 m diagonal

Table 6: **Reference-derived compound shifts scenario configuration.** Every factor of a profile is applied jointly in one condition; the profile is the reference system's own factor set, and this sheet lists the resulting settings at the severity the results report. A bracket is a *sampling support*, not a chosen value: the factor is redrawn from it at the granularity in *Draw* – per episode at `reset`, or per step – so a score is an average over draws, never a single perturbed model.

Task	Metric	$R_{\min}$	$R_{\max}$	Source
Hopper-v5	episodic return	-20.27	3234.3	D4RL v2 random / expert
HalfCheetah-v5	episodic return	-280.18	12135.0	D4RL v2 random / expert
Walker2d-v5	episodic return	1.63	4592.3	D4RL v2 random / expert
AdroitHandDoor-v1	episodic return	-56.51	2880.57	D4RL v2 random / expert
PointMaze_Open-v3	goals per 300 steps	0.0	3.28	no goal/ IQL nominal
LunarLanderContinuous-v3	episodic return	-213.9	136.9	random policy / IQL nominal
PointMaze_UMaze-v3	goals per 300 steps	0.23	5.5	random policy / kinematic ceiling
FetchReach-v4	neg. goal distance	-8.42	0.0	random policy / reward maximum
CarRacing-v3	episodic return	-30.44	900.0	random policy / solved threshold
Pusher-v5s	episodic return	-146.2	-23.97	random policy / PPO nominal

Table 7: **Normalization anchors** $(R_{\min}, R_{\max})$**.** The pairs entering equation 2, fixed per environment and shared by every algorithm, dataset quality, condition and severity. *Source* names the construction behind each pair: D4RL v2 where a published pair exists, and otherwise a random-policy floor with either a task-defined or a reference-learner ceiling. PointMaze is listed twice because the online and offline halves of the benchmark use different mazes.

(Table 5). Finally, the compound grid activates physical, control, sensing, and timing factors jointly in reference-derived Hopper and Pusher profiles (Table 6). Thus, the three grids measure robustness breadth, shift-specific behavior, and robustness under interacting deployment factors, respectively.

The advanced tasks bring their own grids, and both place every method on the same conditions. On Isaac Lab the conditions are positioned relative to the reference envelope (He et al., 2024) of Tables 8 and 9: G1 has 31 per seed, six inside that envelope, eighteen on its boundary, five beyond it, and flat and rough nominal. Drawer has 16, two controls and fourteen beyond the reference ranges, each moving one axis while the rest keep their training randomization.

The VLA evaluation tests eight shift axes across five shift sources (Figure 11). Each axis has its own Q1–Q4 ladder, ordered outward from its nominal setting. These quartiles indicate severity *within* an axis, not equal perturbation strength or task difficulty across axes. Hardware logs place execution delay (83 ms) and command noise (0.020 rad) at Q2, so the sweep includes both milder and more severe conditions than those measured on the platform. We report the actual input change

for perception and command shifts separately; the three dynamics shifts are specified in physical units.

Axis	Training envelop	Evaluation envelop
Contact friction	$\mathcal{U}[0.2, 1.1]$	0.1
Link mass	$\times\,\mathcal{U}[0.7, 1.3]$	$\times\,1.5$
Base CoM, x and y	± 0.1 m	$+0.15$ m along x
Torque noise	$\pm 10\%$ of the effort limit	20%
Action delay	1–3 policy steps (20–60 ms)	4 steps (80 ms)
Actuator K_p, K_d	$\times\,\mathcal{U}[0.75, 1.25]$	boundary only, 0.75 and 1.25
Velocity push	1 m/s, random direction, every 5 s	boundary only, 1 m/s

Table 8: **G1: training and evaluation configuration.**

Axis	Training envelop	Evaluation envelop
Robot friction	static and dynamic $\mathcal{U}[0.8, 1.25]$	both 0.4; both 1.8
Handle friction	static $\mathcal{U}[1.0, 1.25]$, dynamic $\mathcal{U}[1.25, 1.5]$	both 0.5; both 2.0
Cabinet mass	$\times\,\mathcal{U}[0.8, 1.2]$	$\times\,0.6$; $\times\,1.4$
Arm K_p	$\times\,\mathcal{U}[0.9, 1.1]$	$\times\,0.75$; $\times\,1.25$
Arm K_d	$\times\,\mathcal{U}[0.9, 1.1]$	$\times\,0.75$; $\times\,1.25$
Drawer damping	$\times\,\mathcal{U}[0.5, 1.5]$	$\times\,0.25$; $\times\,2.0$
Action delay	0–2 policy steps	3 steps (50 ms); 4 steps (66.7 ms)

Table 9: **Drawer: training and evaluation configuration.**

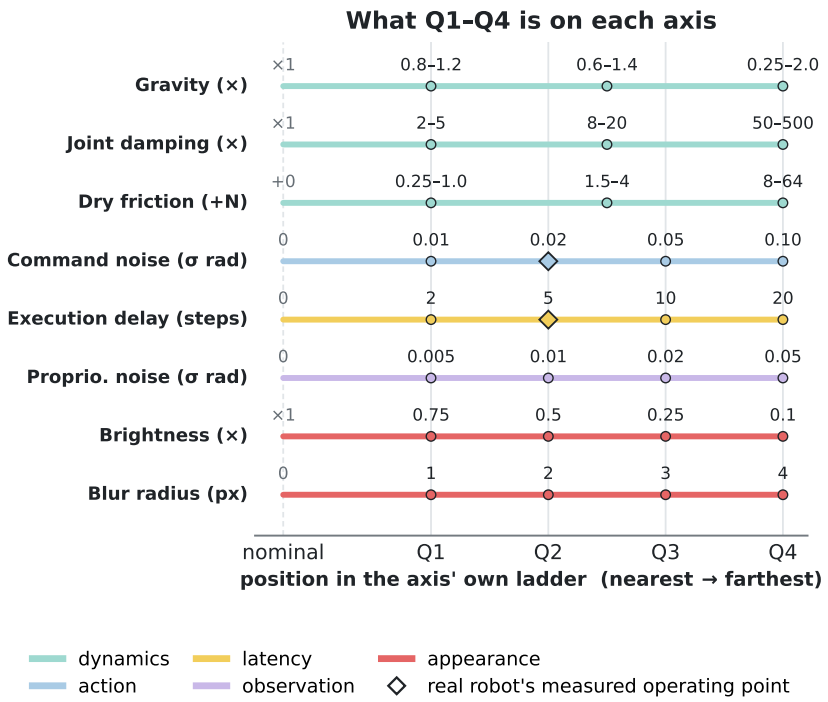

Figure 11: **VLA quartiles grid.** One row per shift axis, coloured by the channel it belongs to. The row label names the axis and the unit its rungs are written in, and the grey value on the dashed column at the left is that axis' nominal setting. Each circle is one rung, annotated with the setting it applies: a fixed value on the five input-side axes, and a range the rung samples within on the three dynamics axes. Horizontal position is the rung's place in that axis' own ladder, nearest to farthest from nominal, so Q1–Q4 index displacement rather than strength; the input-side axes have four rungs and the dynamics axes three, which is why the latter fall between the Q2 and Q3 guides. A diamond marks the operating point measured on the real platform from its demonstration logs. Both axes that carry one land at Q2, so Q1 is milder than the hardware and Q3–Q4 push past it.

D BENCHMARK TASKS AND DATASETS

In this section we present an overview of the environments the benchmark evaluates on, the control interface each of them exposes, and the offline datasets learn from. The main experiment in Section 3.1 covers locomotion, manipulation, navigation, and vehicle control; Pusher and DoorCausal provide the additional compound-profile and semantic-channel tests in Section 3.2. Table 10 gives the default interface of every task we included in our benchmark.

Offline methods learn from the fixed datasets in Table 11. We use Minari (Younis et al., 2024), including its D4RL ports (Fu et al., 2020), wherever a corresponding dataset is available; only for LunarLander do we collect our own dataset.

Advanced robotic tasks. These tasks test robust safe algorithms and show our interface extensibility meanwhile. In Isaac Lab (Mittal et al., 2025), Unitree G1 (Figure 12) locomotion measures return and survival under delay, and Franka drawer manipulation measures success and joint-limit cost under physical shifts. The VLA task (Figure 13) fine-tunes $\pi_{0.5}$ (Intelligence et al., 2025) on expert demonstrations for the bimanual *insert-mouse-battery* task from the RSS 2026 post-training challenge (Zhang et al., 2026), then evaluates perception and execution shifts through camera and robot-state interfaces.

Task	Observation	Action (rate)	Horizon	Task type
Standard control				
Hopper-v5	11-D state	3-D	1,000	Locomotion
Walker2d-v5	17-D state	6-D	1,000	Locomotion
HalfCheetah-v5	17-D state	6-D	1,000	Locomotion
AdroitHandDoor-v1	39-D state	28-D	200	Manipulation
FetchReach-v4	13-D proprioception and relative goal	4-D	50	Manipulation
Pusher-v5s	23-D state	7-D	100	Manipulation
DoorCausal	State	–	300	Manipulation
PointMaze_UMaze-v3	6-D state and desired goal	2-D	300	Navigation
PointMaze_Open-v3	6-D state and desired goal	2-D	300	Navigation
CarRacing-v3	$96 \times 96 \times 3$ pixels	3-D	1,000	Driving
LunarLanderContinuous-v3	8-D state	2-D	1,000	Driving
Advanced robotics				
Unitree Go2	48-D state	12-D (50 Hz)	1,000	Locomotion
Unitree G1	123-D state	37-D (50 Hz)	1,000	Locomotion
Franka Drawer	31-D state	8-D (60 Hz)	480	Manipulation
insert-mouse-battery	14-D joint state and three $180 \times 320 \times 3$ camera streams	14-D (60 Hz)	Demo	Bimanual manipulation

Table 10: **Benchmark tasks.** Observation and Action specify the default policy interface; control rates are shown in parentheses where reported. Numeric horizons are maximum environment steps per episode, while *Demo* follows the length of the recorded demonstration used for VLA evaluation.

Task	Dataset	Source or collection
Hopper, Walker2d, HalfCheetah	Minari `medium-v0`	Collected in Gymnasium v5 environments
AdroitHandDoor	`D4RL/door/expert-v2`	Minari port of D4RL trajectories
PointMaze_Open	`D4RL/pointmaze/open-v2`	Minari port of D4RL trajectories
LunarLanderContinuous	Collected for this benchmark	Deterministic SAC checkpoints trained for 1.5 million steps; selected medium corpus of 1,741 episodes

Table 11: **Offline datasets.** Each dataset contains approximately one million transitions.

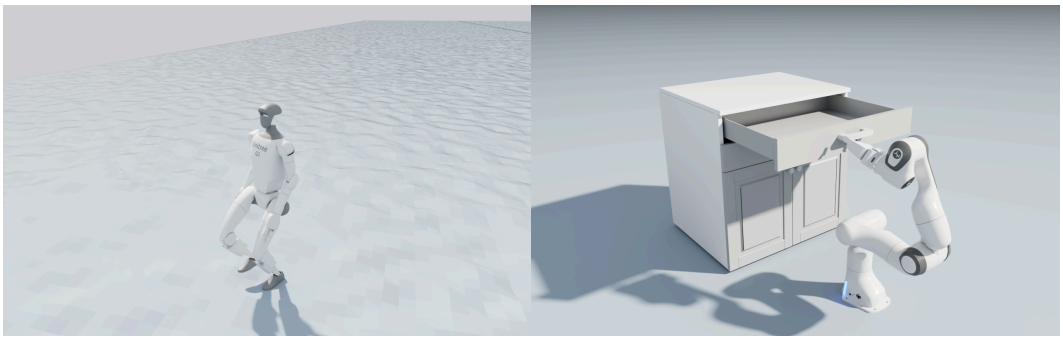

Figure 12: **Isaac Lab tasks.** *Left*: Humanoid G1 locomotion task; *right*: Franka drawer manipulation task.

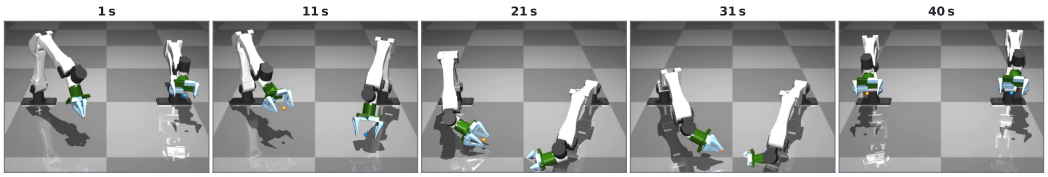

Figure 13: **VLA task.** Bimanual *insert-mouse-battery* demonstrations of the RSS 2026 post-training challenge (Zhang et al., 2026).

E ALGORITHMS

The library holds six standard references that carry no robustness mechanism, fourteen robust methods, and one safe and one safe-robust method. They are organized by the taxonomy of Section 2.1, which asks *where* robustness enters — the learner, the environment the learner collects in, the data the learner consumes, or an explicit generative model of it — and, for learner-centric methods, in which training regime.

E.1 OFFLINE ALGORITHMS

E.1.1 STANDARD REFERENCES.

IQL IQL (Kostrikov et al., 2021) avoids querying out-of-distribution actions altogether: a value function is fitted to the critic by expectile regression, the twin critics bootstrap from that value function, and the actor is extracted by advantage-weighted regression.

TD3+BC TD3+BC (Fujimoto & Gu, 2021) adds a behavior-cloning term to the TD3 actor loss and normalizes states. The term is balanced against the critic by dividing by the mean absolute Q over the batch.

MOPO MOPO (Yu et al., 2020b) learns an ensemble dynamics model, branches short rollouts from dataset states, and penalizes the rollout reward by the model's predicted uncertainty before running SAC on the mixture of real and model data.

SynthER SynthER (Lu et al., 2023) fits a diffusion model to logged transitions and trains an unmodified offline learner on a much larger sample from it. We count it as data augmentation rather than a generative mechanism: it samples individual transitions to enlarge the dataset, where FWM and PLR-PVL sample multi-step futures conditioned on the current state and turn them into critic targets.

E.1.2 LEARNER-CENTRIC MECHANISMS.

RFQI RFQI (Panaganti et al., 2022) replaces the fitted-Q target by a worst-case Bellman backup over a total-variation uncertainty set of radius ρ around the empirical transitions; the inner maximization is solved through a dual variable fitted per batch, and the actor and critic follow the release's BCQ-style architecture.

RORL RORL (Yang et al., 2022) makes an SAC learner conservative through smoothing: the critic and policy are regularized to vary little within an ϵ-ball of dataset states, and Q-values at perturbed states are penalized by the disagreement of a ten-critic ensemble.

ATLA-IQL The offline ATLA adaptation (Zhang et al., 2021) keeps ATLA's principle, a learned observation adversary alternated with the learner, but the adversary is trained by a differentiable objective on the critic by minimizing the minimum Q at the policy's action under the perturbed state instead of by RL, since no environment is available.

E.1.3 DATA-CENTRIC MECHANISMS.

RSC-IQL RSC-IQL (Ding et al., 2023a) applies RSC's counterfactual rewriting to logged data: a causal mask over state dimensions is fitted from the dataset, and transitions are rewritten by permuting or perturbing the dimensions the mask marks as non-causal.

RAMBO RAMBO (Rigter et al., 2022) augments the dataset with short rollouts from a learned dynamics ensemble and replaces the uncertainty penalty by adversarial training of the model: the ensemble is updated to lower the critic's value of the policy's transitions while a maximum-likelihood term keeps it close to the data; the policy is behavior-cloned before the first epoch.

E.1.4 GENERATIVE MODEL.

ROMB ROMB (Chen et al., 2025) adapts a learned world model against the policy under a constrained maximin objective: the model is pushed toward transitions that lower the policy's value while a supervised loss keeps it on the data, and the policy is trained on rollouts from the adapted model.

FWM FWM transfers the multi-step future modeling of Diffusion World Model (Ding et al., 2024) to a flow-matching (Lipman et al., 2022) world model: instead of a one-step bootstrap, the critic regresses an h-step return whose future is sampled from the generative model conditioned on the current state, action and return-to-go. The replacement is confined to how that future is produced. Writing x_1 for a length-T window of states and rewards and $x_0 \sim \mathcal{N}(0, I)$, the diffusion model's denoising objective becomes a regression onto the straight-line velocity of the linear path $x_t = (1-t)\,x_0 + t\,x_1$,

$$\mathcal{L} = \mathbb{E}_{t\sim\mathcal{U}[0,1],\,x_0,\,(x_1,c,g)} \big\| v_\theta(x_t, t, c, g) - (x_1 - x_0) \big\|_2^2, \tag{3}$$

with c the current state and action and g the return-to-go, dropped with probability 0.2 so one network carries both the conditional and the unconditional velocity. Sampling then replaces the reverse diffusion chain by an ODE integrated from $t = 0$ to $t = 1$ in $N = 5$ midpoint steps on the guidance-mixed velocity:

$$\tilde{v}_\theta = w\, v_\theta(\cdot \mid c, g) + (1-w)\, v_\theta(\cdot \mid \varnothing), \qquad w = 2.0. \tag{4}$$

PLR-PVL PLR-PVL (Berdica et al., 2024) trains a collection of world models consistent with the offline data, treats each as a level in the sense of unsupervised environment design, and prioritizes them during policy learning by the value loss they induce. We adapt the method by replacing their world model with the flow-matching model of equation 3: the ensemble is K flow models differing only in training seed, each proposing one h-step future for the same conditioning, and the critic target is taken per sample from the model whose rollout carries the largest positive value loss,

$$k^\star = \arg\max_k \; \frac{1}{h} \sum_{t=1}^{h} \max\big(\delta_t^{(k)},\, 0\big), \tag{5}$$

where $\delta_t^{(k)}$ is the GAE-λ residual of the current critic along the k-th rollout.

E.2 Online algorithms

E.2.1 Standard RL references.

SAC, PPO and TRPO SAC (Haarnoja et al., 2018) is the off-policy reference: twin critics with a clipped double-Q target and a stochastic actor whose objective adds an entropy bonus with automatically tuned temperature. PPO (Schulman et al., 2017) is the on-policy reference: a clipped probability-ratio surrogate on trajectories from the current policy, with generalized advantage estimation. TRPO (Schulman et al., 2015) replaces the clip by an explicit KL trust region solved with conjugate gradients; it appears only as the base of RARL-TRPO. All three are the Stable-Baselines3 implementations (Raffin et al., 2021) (TRPO from `sb3-contrib`).

E.2.2 Learner-centric mechanisms

ATLA. ATLA (Zhang et al., 2021) trains a policy against an *optimal* observation adversary in the state-adversarial MDP (Zhang et al., 2020): the adversary is itself an RL agent that outputs a bounded perturbation of the observation, is rewarded by the protagonist's negative return, and is updated in alternation with the protagonist.

ATLA-SA ATLA-SA (Zhang et al., 2021) adds the SA-PPO regularizer (Zhang et al., 2020) to ATLA: a KL penalty between the policy at a state and at the worst-case neighbor found by stochastic gradient Langevin dynamics, and a recurrent (LSTM) policy with a 100-step history so that the agent can infer a persistent perturbation from context.

RSC RSC (Ding et al., 2023a) learns a causal graph and dynamics model over the state variables from replayed experience, then generates counterfactual transitions by intervening on the variables the graph marks as non-causal for the reward. Mixing these into replay removes the spurious correlations a policy would otherwise exploit.

E.2.3 Environment-centric mechanisms.

RARL RARL (Pinto et al., 2017) casts robustness as a two-player zero-sum game: an adversary applies bounded external forces to the robot and is trained to minimize the protagonist's return, and the two are updated in alternating blocks.

DR Domain randomization (Tobin et al., 2017; Peng et al., 2018) samples simulator parameters from fixed ranges at every episode so that the policy meets a distribution of dynamics during training.

E.2.4 Robust safe methods.

RAMU RAMU (Queeney & Benosman, 2023) applies a risk measure over sampled next-state perturbations to both the reward and the cost targets; the original is defined for TD learners, and the row reported here is a PPO adaptation that applies the same transform to PPO's value targets.

SPiDR SPiDR (As et al., 2026) trains a constrained policy under domain randomization and adds a pessimistic cost penalty proportional to the disagreement of next-state predictions across sampled dynamics; it runs with the authors' solver and a joint-limit cost budget.

F Training Settings

F.1 Training protocol

We train each method with five independent seeds to its budget in Table 14. Unless otherwise noted, we freeze the final checkpoint and evaluate each policy for 20 episodes per condition using the same evaluation seed. RFQI retains its native best-checkpoint selection rule.

Our default protocol trains on the nominal task and introduces shifts only when evaluating the frozen policy. Reward corruption and delayed reward delivery are exceptions: they affect learning feedback but cannot alter the actions of a frozen policy. We therefore apply these shifts during training and

Task	Method	Env. steps	Wall-clock time (h)
	PPO	1M	0.40 ± 0.15
Walker2d-v5	PPO	2M	0.80 ± 0.29
	SAC	1M	1.60 ± 0.03

Table 12: **Walker2d training wall-clock time on H100 GPUs.** Values are mean $\pm$ sample standard deviation over three training seeds. Wall-clock time is measured from the start of training to the corresponding checkpoint save.

evaluate the resulting policies on the nominal task. These scores measure the effect of uncertain training feedback and are reported separately from post-training shift robustness.

The Isaac Lab and VLA studies are reported under their own protocol, and the departures are the following. Humanoid tasks on Isaac Lab use three training seeds and score 128 episodes per condition. They match the retained primary training interactions within each task to the budget in Table 15. The five methods share one base learner, so the comparison is between mechanisms rather than between releases. And they do not hold the shift grid out from training: PPO and RAMU-PPO train on the nominal specification, whereas DR-PPO, ATLA-PPO and SPiDR-PPO each train inside a randomization envelope. Table 8 and Table 9 show the envelopes applied in our benchmark, which is transferred from a humanoid sim-to-real pipeline (He et al., 2024). The VLA task finetunes a single $\pi_{0.5}$ model on demonstrations, and evaluated by joint-space tracking against the recorded demonstration.

F.2 Training budget alignment

This part presents our training budget and provides the evidence of our training protocol alignment. We train SAC-based algorithms for 1M environment steps and PPO-based algorithms for 2M steps, consistent with commonly used training budgets (Raffin & Stulp, 2020). This asymmetric budget accounts for PPO's lower sample efficiency: as an on-policy method, PPO generally requires more environment interactions to reach a comparable level of training maturity. Figure 14 shows that the additional PPO steps provide meaningful learning progress.

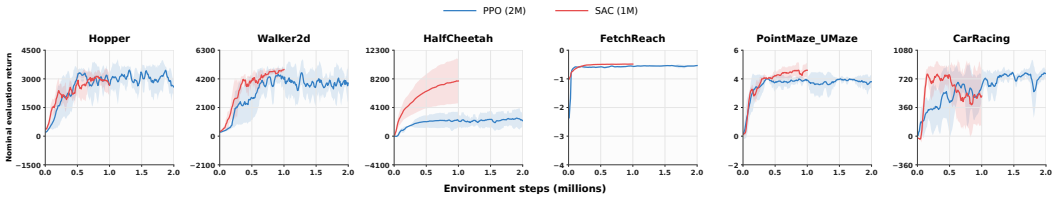

Figure 14: Online training curves

However, more environment steps do not imply a larger computational budget. On Walker2d, PPO at 2M steps requires only 0.80 ± 0.29 hours, compared with 1.60 ± 0.03 hours for SAC at 1M steps on H100 GPUs (Table 12). Thus, the 2M-step PPO budget compensates for its lower sample efficiency without giving it an advantage in wall-clock compute.

For ATLA and ATLA-SA, their official configuration remains unchanged, which is about 5M protagonist steps rather than the 2M the other PPO rows use. The asymmetry is also the one the ATLA paper itself reports under: its released configurations give vanilla PPO and SA-PPO 976 iterations of the same 2048-step rollout, i.e. the same 2M our PPO row uses, against 2441 for ATLA and ATLA-SA. Our PPO-versus-ATLA comparison therefore reproduces the budget pairing those results were established with rather than introducing a new one. The full online training budget is in Table 14.

Taken together, the online budget is set by the base algorithm. Table 13 summarizes the resulting protocol. Methods sharing the same base algorithm are trained with the same budget, while ATLA and ATLA-SA retain their official $\sim$5M-step configuration.

Budget rule	**Methods**	**Steps**
SAC-based	SAC, DR-SAC, RSC-SAC	1M
PPO/TRPO-based	PPO, RARL-PPO, RARL-TRPO	2M
Official ATLA configuration	ATLA, ATLA-SA	$\sim$5M

Table 13: **Online training budget summary.**

Method	Family	Base	Budget
Online			
SAC	Standard	—	1M steps
PPO	Standard	—	2M steps
ATLA	Learner-centric	PPO	2441×2048 steps
ATLA-SA	Learner-centric	PPO (LSTM)	2441×2048 steps
RSC-SAC	Learner-centric	SAC	1M steps
RARL-PPO / -TRPO	Env.-centric	PPO / TRPO	2M steps
DR-SAC	Env.-centric	SAC	1M steps
RAMU	Safe RL	PPO	Part-4 recipe
SPiDR	Safe-robust	PPO	Part-4 recipe
Offline			
IQL	Standard	—	1M, batch 256
TD3+BC	Standard	—	1M, batch 256
MOPO	Standard	SAC	1M, batch 256
SynthER	Standard	IQL	1M, batch 256
RFQI	Learner-centric	FQI	500k, batch 1000
RORL	Learner-centric	SAC	3M (MuJoCo, Door) / 1M
ATLA-IQL	Learner-centric	IQL	1M, batch 256
RSC-IQL	Data-centric	IQL	1M, batch 256
RAMBO	Data-centric	SAC	2M (MuJoCo, Door) / 1M
ROMB	Generative model	IQL	1M, batch 256
FWM	Generative model	IQL	1M, batch 256
PLR-PVL	Generative model	IQL	1M, batch 256

Table 14: **Training budget.** Budgets are environment steps for online methods and policy/critic gradient updates for offline methods. Training budget alignment explanation is in section F.2.

For the offline algorithms' training budgets, Table 14 shows nine of the twelve offline columns share the same 1M-update budget. RAMBO and RORL (Rigter et al., 2022; Yang et al., 2022) follow their official budgets on the tasks their releases cover and keep the 1M budget elsewhere. Thus, our MuJoCo comparisons with 1M-update references preserve native budgets. RFQI (Panaganti et al., 2022) is the only exception: it follows its official budget and reports its own best checkpoint, because forcing it onto the common protocol would break its mechanism.

Task	PPO, DR-PPO, RAMU-PPO	ATLA-PPO	SPiDR-PPO	B
G1	$1500 \times 4096 \times 24$	$750 \times 2 \times 4096 \times 24$	$900 \times 512 \times 320$	147,456,000
Drawer	$3200 \times 512 \times 96$	$1600 \times 2 \times 512 \times 96$	$3200 \times 512 \times 96$	157,286,400

Table 15: **Training budgets for Isaac Lab tasks.** Iterations are chosen so that the retained primary training interactions B match within each task.

G Additional results and analysis

G.1 What the mechanism-gain columns contain

Figure 6 reports two gains for each mechanism family. The **darker column** is the *base-relative gain*: each method with a matched backbone is compared with that backbone under the same conditions, and the differences are averaged within the family. Standalone recipes without a matched reference learner, including RFQI, RORL, and RAMBO, are excluded; the number of included methods is reported as n in Table 16. The **lighter column** is the *standard-pool gain*: the full family's mean minus the mean of all six Standard RL references. It therefore reflects both the robustness mechanism and the strength of its underlying learner.

Brackets denote 95% seed-level stratified bootstrap intervals from 10,000 replicates. Each training seed contributes its mean over 20 evaluation episodes, and seeds are resampled within each (method, task) stratum, with the same draw reused across all conditions of that task. Methods and backbones are separate runs and are resampled independently. This preserves dependence across conditions within a seed and avoids treating evaluation episodes as independent training samples.

Because the two gains use different references and may include different methods, their difference is not an exact decomposition of mechanism and base learner effects. Both online families retain positive base-relative gains ($+5.5$ and $+6.6$), whereas offline gains are smaller. The generative family offers the cleanest comparison: all three methods use IQL, yielding a $+9.8$ gain over the standard pool but $+2.5$ over IQL. Its aggregate advantage thus reflects both the added mechanism and a strong base algorithm, while fixed-data coverage may limit the incremental benefit available offline.

Mechanism family	Paired gain	Pooled gain	n paired
Learner-centric (Online)	$+5.5$ $[+2.8, +8.3]$	$+9.5$ $[+7.2, +11.8]$	3
Environment-centric	$+6.6$ $[+3.8, +9.6]$	$+9.6$ $[+7.3, +11.8]$	2
Learner-centric (Offline)	$+3.8$ $[+0.8, +6.4]$	$+2.9$ $[+1.3, +4.4]$	1
Data-centric	-0.1 $[-2.7, +2.1]$	$+7.6$ $[+4.2, +11.1]$	1
Generative model	$+2.5$ $[-0.3, +4.9]$	$+9.8$ $[+7.9, +11.7]$	3

Table 16: **The two columns of the mechanism panel, with intervals.** Point estimates and 95% seed-level stratified bootstrap intervals from 10,000 replicates. *Paired* subtracts each member's own base optimizer; *pooled* subtracts the mean of all six Standard RL references. n is the number of members that have a matched base learner and therefore enter the paired column.

G.2 Task Capability: On-policy vs Off-policy

On-policy methods update a policy using trajectories collected by that policy or a recent version of it, as in PPO (Schulman et al., 2017). Off-policy methods can also learn from data generated by earlier or different policies, often through experience replay, as in SAC (Haarnoja et al., 2018). This distinction concerns which data can be used for an update.

Figure 15 shows that the evaluated off-policy methods generally achieve higher scores on the MuJoCo locomotion tasks, while on-policy methods are stronger on PointMaze-UMaze and CarRace. In locomotion, frequent reward feedback allows replay-based learners to reuse informative transitions across many updates, which may contribute to the off-policy methods' advantage. PointMaze requires discovering goal-reaching trajectories, while CarRacing couples control with visual input; updates from recent on-policy rollouts may be more useful in these settings.

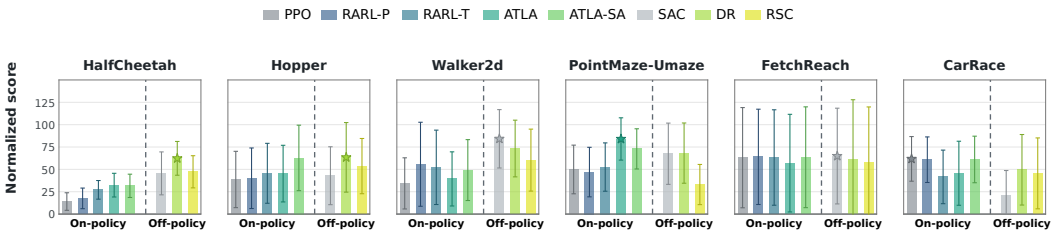

Figure 15: **On-policy and off-policy performance aggregated by task.**

G.3 Safety evaluation on Isaac Lab

Figure 16 indicates that across the Isaac Lab tasks, SPiDR-PPO most consistently balances performance and safety: it maintains high success while satisfying the cost budget on Drawer, and leads on

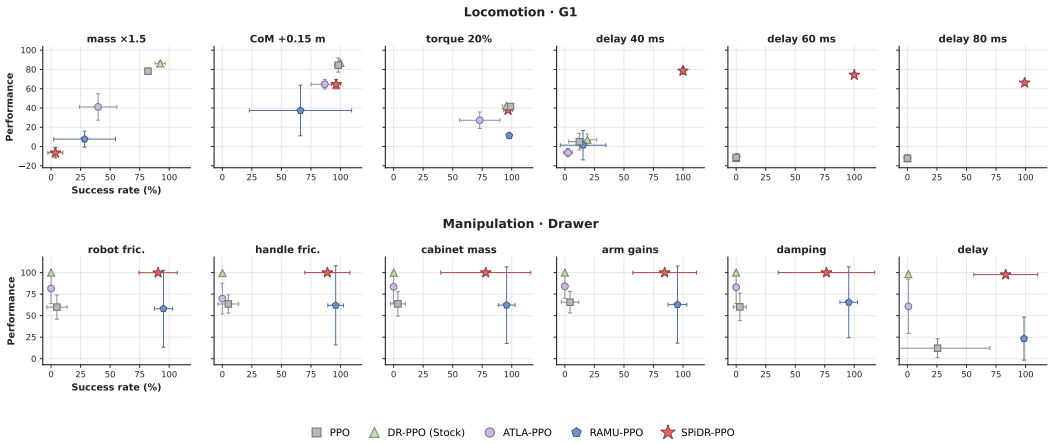

Figure 16: **Isaac Lab results.** Each panel repeats Figure 10 on one axis: safety success rate against performance, one point per method, bars the s.d. over three training seeds.

both return and survival under G1 action delays. These results show why high return alone is insufficient for sim-to-real evaluation. Robust-safe methods must be assessed for constraint satisfaction alongside task performance.

G.4 Supplementary results

Figures 17-18 show our complete comparison on 220 different configurations across 4 task families in Section 3.1. Figures 19 and 20 show our full shift-isolated evaluation on 44 distinct perturbation configurations. Figure 21 further shows full results on 9 perturbation configurations including compound-shifts scenario.



Figure 17: **Supplementary online results of Section 3.1**

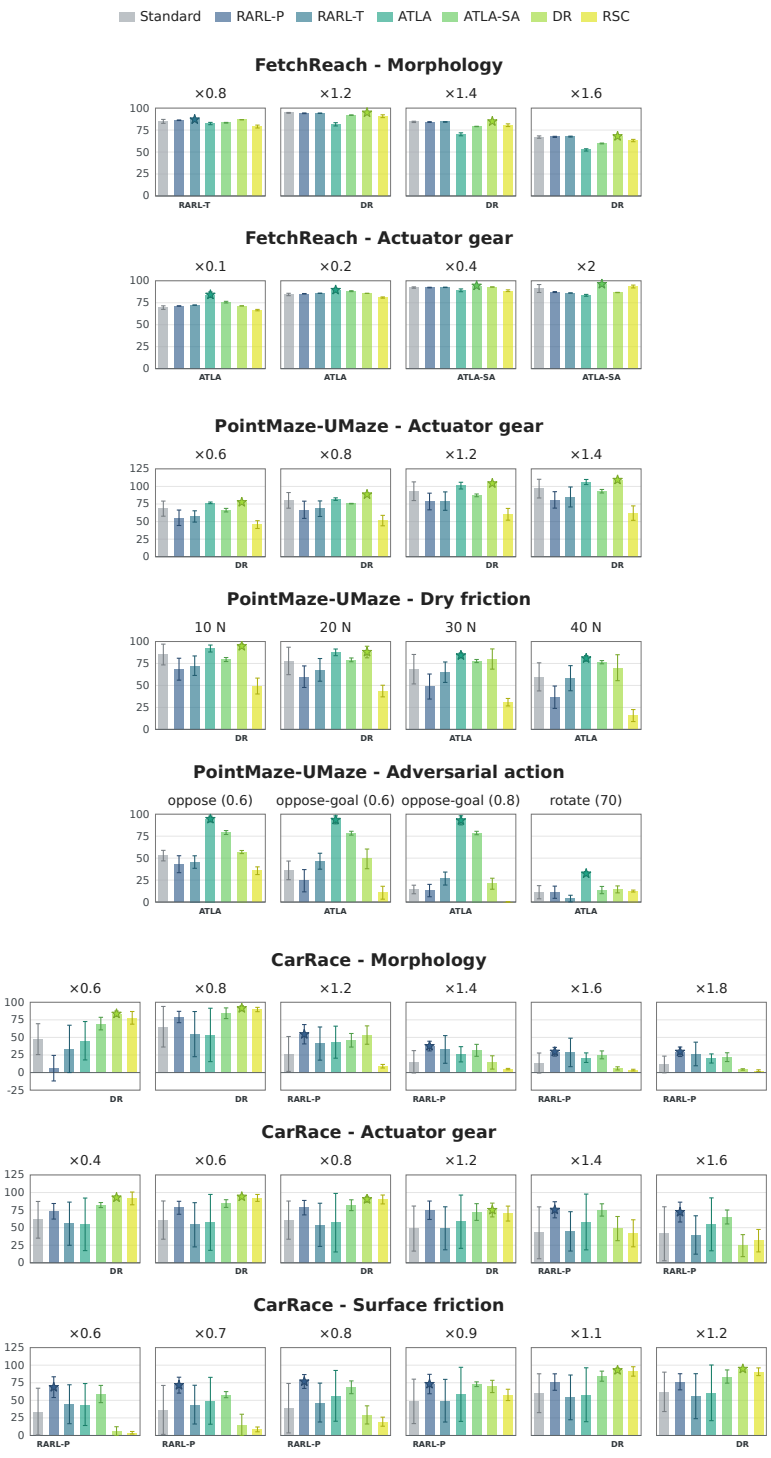

Figure 17: **Supplementary online results of Section 3.1 (continued)**



Figure 18: **Supplementary offline results of Section 3.1**



Figure 18: **Supplementary offline results of Section 3.1 (continued)**



Figure 19: **Supplementary online results of Section 3.2.**



Figure 20: **Supplementary offline results of Section 3.2.**

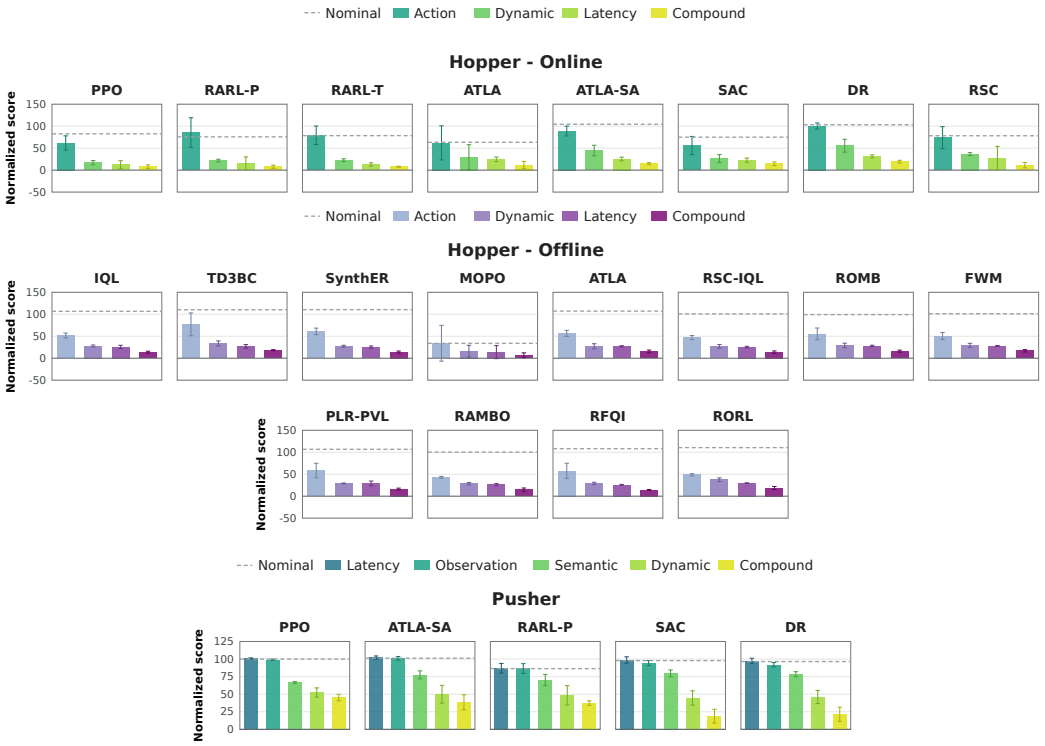

Figure 21: **Supplementary results of Section 3.2.**

H HYPERPARAMETER SETTINGS

H.1 ONLINE ALGORITHM HYPERPARAMETERS

The tables below report algorithm hyperparameters only. **HC**, **W2D**, and **H** denote **HalfCheetah**, **Walker2d**, and **Hopper**, respectively. A combined **HC/W2D/H** column is used only when all listed values agree across the three tasks. Dashes indicate that a parameter does not apply.

Parameter	**H/HC/W2D**	**PointMaze**	**FetchReach**	**CarRacing**
Learning rate	3×10^{-4}			
Hidden layers	2×64	2×64	2×64	NatureCNN
Discount factor γ	0.99			
GAE parameter λ	0.95			
PPO clipping parameter	0.2			
Value-loss coefficient	0.5			
Entropy coefficient	0			
Rollout length	2048			
Minibatch size	64			
Optimization epochs	10			
Maximum gradient norm	0.5			

Table 17: **PPO hyperparameters.**

Parameter	**H/HC/W2D**	**PointMaze**	**FetchReach**	**CarRacing**
Learning rate	3×10^{-4}			
Hidden layers (actor / critic)	2×256	2×256	2×256	NatureCNN $+2 \times 256$
Target smoothing τ	0.005			
Replay-buffer size	10^6			
Learning starts	100			
Train frequency / gradient steps	$1/1$			
Discount factor γ	0.99			
Target entropy	automatic	automatic	automatic	automatic (-3)
Minibatch size	256			

Table 18: **SAC hyperparameters.**

Parameter	**Hopper**	**Walker2d**	**HalfCheetah**	**PointMaze**	**FetchReach**	**CarRacing**
Adversary target	foot	both feet	torso and feet	particle	Cartesian action	hull
Adversary magnitude	5 N	5 N	5 N	10 N	0.1	5
Update ratio	$1/1$	$1/1$	$1/1$	$2/1$	$1/1$	$2/1$
Adversary warm-up	0	0	0	5×10^4	0	4×10^4
Zero-force probability p_0	0	0	0	0.25	0	0.25
Protagonist/adversary lr	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	2.5×10^{-4}
Entropy coefficient	0	0	0	0	0.01	0.01
Discount factor γ	0.99					
GAE parameter λ	0.95					
PPO clipping parameter	0.2					

Table 19: **RARL-PPO hyperparameters.** The update ratio denotes protagonist-to-adversary update blocks.

Parameter	Hopper	Walker2d	HalfCheetah	PointMaze	FetchReach	CarRacing
Adversary target	foot	both feet	torso and feet	particle	Cartesian action	hull
Adversary magnitude	5 N	5 N	5 N	10 N	0.1	2
Update ratio	$1/1$	$1/1$	$1/1$	$2/1$	$1/1$	$5/1$
Adversary warm-up	0	0	0	5×10^4	0	3×10^5
Zero-force probability p_0	0	0	0	0.25	0	0.5
Protagonist learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-4}
Adversary learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	5×10^{-5}
Discount factor γ	0.99					
GAE parameter λ	0.95					
Target KL divergence	0.01					
CG steps / damping	$15/0.1$					
Critic updates / policy update	10					

Table 20: **RARL-TRPO hyperparameters.** The update ratio denotes protagonist-to-adversary update blocks; CG denotes conjugate gradient.

Parameter	H/HC/W2D	PointMaze	FetchReach	CarRacing
Randomized parameters	g/morph./gear	friction/gear	morph./gear	friction/morph./gear
Randomization range	$[0.9, 1.1]$	$[0, 7.5]/[0.9, 1.1]$	$[0.9, 1.1]$	$[0.9, 1.1]$
Learning rate	3×10^{-4}	3×10^{-4}	10^{-3}	10^{-4}
Learning-rate schedule	constant	constant	constant	linear to 3×10^{-5}
Discount factor γ	0.99	0.99	0.95	0.99
Target smoothing τ	0.005			
Entropy coefficient	automatic tuning			
Target entropy	automatic	automatic	automatic	-2 (fixed)

Table 21: **DR-SAC hyperparameters.** Physical parameters are sampled independently at reset and remain fixed within an episode. Morph. represents Morphology

Parameter	H/HC/W2D	PointMaze	FetchReach	CarRacing
Actor learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}	10^{-4}
Critic learning rate	10^{-3}	3×10^{-4}	10^{-3}	10^{-4}
Learning-rate schedule	constant	constant	constant	linear to 3×10^{-5}
Causal/RSC warm-up	10^5	3×10^5	10^5	4×10^5
Replay augmentation / RSC mix	0.25	0.10	0.25	0.05
Selected fraction	–	0.25	–	0.25
Reference size	128	–	128	–
Dynamics learning rate	–	10^{-4}	–	10^{-4}
Discount factor γ	0.99			

Table 22: **RSC-SAC hyperparameters.**

Parameter	Hopper	Walker2d	HalfCheetah	PointMaze	FetchReach	CarRacing
Observation radius ϵ	0.075	0.05	0.15	0.075	0.075	$8/255$
Policy learning rate	3×10^{-4}	4×10^{-4}	10^{-4}	3×10^{-4}	3×10^{-4}	10^{-4}
Value learning rate	2.5×10^{-4}	3×10^{-4}	10^{-3}	2.5×10^{-4}	2.5×10^{-4}	10^{-4}
Adversary policy lr	3×10^{-3}	10^{-3}	3×10^{-3}	3×10^{-3}	3×10^{-3}	10^{-4}
Adversary value lr	3×10^{-5}	10^{-4}	3×10^{-4}	3×10^{-5}	3×10^{-5}	–
Adversary updates per block	1	1	1	1	1	4
Initial policy std. dev.	1.0	1.0	0.5	1.0	1.0	–
Discount factor γ	0.99					
GAE parameter λ	0.95					
PPO clipping parameter	0.2					

Table 23: **ATLA-PPO hyperparameters.**

Parameter	Hopper	Walker2d	HalfCheetah	PointMaze	FetchReach	CarRacing
Observation radius ϵ	0.075	0.05	0.15	0.075	0.075	$8/255$
SA regularization weight	0.3	0.3	0.03	0.3	0.3	0.1
Policy learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-4}
Value learning rate	10^{-2}	3×10^{-2}	3×10^{-3}	10^{-2}	10^{-2}	10^{-4}
Adversary policy lr	10^{-2}	3×10^{-4}	10^{-2}	10^{-2}	10^{-2}	10^{-4}
Adversary value lr	10^{-2}	10^{-2}	3×10^{-4}	10^{-2}	10^{-2}	–
Adversary updates per block	1	1	1	1	1	4
Initial policy std. dev.	1.0	1.0	0.5	1.0	1.0	–
History length	100	100	100	100	100	–
Discount factor γ	0.99					
GAE parameter λ	0.95					
PPO clipping parameter	0.2					
State-adversarial inner steps	2					

Table 24: **ATLA-SA-PPO hyperparameters.** State-task ϵ values act on normalized observations; CarRacing uses 8 raw pixel levels, equivalent to $8/255$ after image normalization.

H.2 Offline algorithm hyperparameters

The tables below give the algorithm hyperparameters behind the reported offline results. Dashes indicate that a parameter does not apply.

Parameter	Hopper	Walker2d	HalfCheetah	Door	PointMaze	LunarLander
AWR temperature β	3.0	3.0	3.0	0.5	3.0	3.0
Policy dropout	—	—	—	0.1	—	—
Expectile τ	0.7					
Advantage clip	100					
Learning rate	3×10^{-4}					
Hidden layers	2×256					
Discount factor γ	0.99					
Target EMA rate	0.005					
Batch size	256					

Table 25: **IQL hyperparameters.**

Parameter	Value	Parameter	Value
BC weight α	2.5	Target policy noise	0.2
Noise clip	0.5	Policy delay	2
Learning rate	3×10^{-4}	Hidden layers	2×256
Discount factor γ	0.99	Target EMA rate	0.005
State normalization	on	Batch size	256

Table 26: **TD3+BC hyperparameters.**

Parameter	Hopper	Walker2d	HalfCheetah	Door	PointMaze	LunarLander
Rollout horizon h	42	3	5	1	5	5
Penalty coeff. λ	37.28	1.0	1.0	5.0	1.0	1.0
Penalty type	ens.-std	ens.-std	max-aleat.	max-aleat.	max-aleat.	max-aleat.
Real data ratio	0.05	0.5	0.05	0.5	0.05	0.05
Dynamics ensemble	7 networks, 5 elites					
Rollouts per refresh	5×10^4					
Target entropy	-3					
Learning rate	3×10^{-4}					
Discount factor γ	0.99					
Target EMA rate	0.005					
Batch size	256					

Table 27: **MOPO hyperparameters.** The penalty type names the ensemble statistic the reward is discounted by: `max-aleat.` is the maximum per-member aleatoric standard deviation of the original paper, `ens.-std` the standard deviation across members, which (Lu et al., 2021) introduce and find to work better than the original penalty at much longer rollout horizons. The Hopper column takes its three values directly.

Parameter	Value	Parameter	Value
Denoiser	residual MLP	Width	2048
Layers	6	Time embedding	128
Fourier features	random	Sampling steps	128
σ_{data}	1.0	S_{churn}	80
Learning rate	3×10^{-4}	Schedule	cosine
Batch size	1024	Terminals modeled	yes

Table 28: **SynthER hyperparameters.** The consumer is the IQL agent of Table 25, trained on the generated dataset with normalization statistics taken from the real one.

Parameter	Hopper	Walker2d	HalfCheetah	Door	PointMaze	LunarLander
adversary weight λ_{adv}	3×10^{-4}	0	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Horizon h	5	5	5	1	1	1
BC anchor weight	1.0	—	—	1.0	1.0	1.0
Target entropy	-3	-6	-6	-90	-2	-2
Real data ratio	0.5					
Dynamics ensemble	7 networks, 5 elites					
Rollouts per refresh	5×10^4					
Critic / actor lr	3×10^{-4} / 1×10^{-4}					
Adversary lr	3×10^{-4}					
Discount factor γ	0.99					
Target EMA rate	5×10^{-3}					
Batch size	256					

Table 29: **RAMBO hyperparameters.** The BC anchor is a behaviour-cloning term kept on the actor throughout training.

Parameter	Hopper	Walker2d	HalfCheetah	Door	PointMaze	LunarLander
Uncertainty radius ρ	0.5	0.3	0.3	0.5	0.3	0.3
Actor lr	1×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Critic lr	6×10^{-4}	8×10^{-4}	8×10^{-4}	8×10^{-4}	8×10^{-4}	8×10^{-4}
Perturbation limit Φ	0.1					
Convex weight λ	0.75					
Dual tolerance / max steps	10^{-3} / 10^4					
Target / policy samples	10 / 100					
Discount factor γ	0.99					
Target EMA rate	0.005					
Batch size	1000					

Table 30: **RFQI hyperparameters.**

Parameter	Hopper	Walker2d	HalfCheetah	Door	PointMaze	LunarLander
Smoothing samples	20	20	10	20	20	20
Policy smooth ϵ	0.005	0.01	0.001	0.01	0.005	0.005
Policy smooth weight	0.1	1.0	0.1	1.0	0.1	0.1
Q smooth ϵ	0.005	0.01	0.001	0.01	0.005	0.005
Q OOD ϵ	0.01	0.01	0	0.01	0.01	0.01
Q OOD weight	0.5	0.1	0	0.1	0.5	0.5
Uncert. weight, init / min	2.0 / 0.1	1.0 / 0.1	0 / 0	1.0 / 0.1	—	—
Uncertainty decay	10^{-6}	5×10^{-7}	0	5×10^{-7}	—	—
BC anchor weight, from update 2×10^4	1.0					
Critic ensemble	10 critics					
Q smooth weight / τ	10^{-4} / 0.2					
Hidden layers	3×256					
Policy / critic lr	3×10^{-4}					
Discount factor γ	0.99					
Target EMA rate	0.005					
Batch size	256					

Table 31: **RORL hyperparameters.**

Parameter	Value	Parameter	Value
Adversary radius ϵ	0.025	ϵ ramp length	1.5×10^4 updates
Adversarial actor loss	0.1	Adversary lr	3×10^{-4}
L_2 regulariser	0.02	EMA	0.995

Table 32: **ATLA-IQL hyperparameters.**

Parameter	Value	Parameter	Value
Rewrite mode	causal permutation	Rewritten fraction of batch	0.25
Mask selection threshold	0.7	Perturbation probability scale	0.3
Mask sparsity weight	0.5		

Table 33: **RSC-IQL hyperparameters.**

Parameter	Value	Parameter	Value
Dynamics ensemble	7 networks, 5 elites	Rollout horizon	5
GAE parameter λ	0.95	Adversary clip range	0.2
Supervised anchor weight	3000	Model share of critic target	0.5
Stackelberg coefficient	0		

Table 34: **ROMB hyperparameters.**

Parameter	Value	Parameter	Value
Horizon h	5	Guidance weight	2.0
ODE sampling steps / solver	5 / midpoint	RTG conditioning	on
Rollout window T	12	Backbone	U-Net, 128 features
Learning rate	10^{-4}	Conditioning dropout	0.2
Batch size	256		

Table 35: **FWM hyperparameters.**

Parameter	Value	Parameter	Value
Flow-model ensemble	3	PVL decay λ	0.95
PVL discount $\gamma\lambda$	0.9405	Ensemble diversity	training seeds
PVL horizon	$h = 5$	Advantage clipping	positive part

Table 36: **PLR-PVL hyperparameters.**

H.3 ROBUST SAFE ALGORITHM HYPERPARAMETERS

Parameter	Value	Parameter	Value
G1			
Actor and critic	256-128-128, ELU	Observation normalization	off
Learning rate	10^{-3}, adaptive	KL target	0.01
Discount γ	0.99	GAE λ	0.95
Entropy coefficient	0.008	Clip ϵ	0.2
Drawer			
Actor and critic	256-128-64, ELU	Observation normalization	off
Learning rate	5×10^{-4}, fixed	KL target	0.02
Discount γ	0.99	GAE λ	0.95
Entropy coefficient	0.001	Clip ϵ	0.2

Table 37: **PPO hyperparameters**

Parameter	Value	Parameter	Value
G1			
Observation corruption	on	Push, every 10–15 s	$v_x, v_y \sim \mathcal{U}[-0.5, 0.5]$ m/s
Drawer			
Robot friction	$\mathcal{U}[0.8, 1.25]$	Handle friction	static $\mathcal{U}[1.0, 1.25]$, dynamic $\mathcal{U}[1.25, 1.5]$
Cabinet mass	$\times\,\mathcal{U}[0.8, 1.2]$	Arm K_p, K_d	$\times\,\mathcal{U}[0.9, 1.1]$
Drawer damping	$\times\,\mathcal{U}[0.5, 1.5]$	Action delay	0–2 policy steps

Table 38: **DR-PPO hyperparameters.**

Parameter	Value	Parameter	Value
Adversary network	128-128, ELU	Adversary lr	10^{-3}
Adversary clip	0.2	Adversary entropy coefficient	0
Perturbation	$\epsilon \tanh(\cdot)$, $\epsilon = 0.05$	Phases per outer iteration	$1 + 1$

Table 39: **ATLA-PPO hyperparameters.**

Parameter	Value	Parameter	Value
Synthetic next states	5	Uncertainty ε	0.1
Wang distortion η	0.75	Cost discount	0.9
Episode cost budget	20	Initial multiplier	0.01
Dual learning rate	0.01	Cost EMA β	0.9
KL guard, mean / max	0.03 / 0.3	Step halvings	8

Table 40: **RAMU-PPO hyperparameters.**

Parameter	Value	Parameter	Value
Auxiliary members per env	8	Uncertainty penalty	$0.075 \times \mathrm{Var}(s')$
Variance clip	10^3	Cost discount	0.9
Cost GAE λ	0.95	Episode cost budget	20
Initial multiplier	0.01	Dual learning rate	0.01

Table 41: **SPiDR-PPO hyperparameters.**