Quantifying Sim-to-Real Robustness in Proximal Policy Optimization via Domain Randomization

Raghunathareddy GR

Department of Artificial Intelligence and Machine Learning, REVA University, Bengaluru, India

raghunathareddy@reva.edu.in

***Abstract*—**Deep reinforcement learning policies trained in simulation frequently suffer severe degradation when transferred to physical environments due to unmodeled discrepancies in mass, surface friction, actuator degradation, sensor noise, and communication latencies—collectively known as the reality gap. In this work, we present an end-to-end empirical study quantifying the performance trade-offs, generalization retention, and out-of-distribution limits of Domain Randomization (DR) in Proximal Policy Optimization (PPO). Evaluated on continuous locomotion dynamics (`HalfCheetah-v5`) across 20 training runs (20×10^6 total environment steps) and 500 deterministic evaluation episodes spanning 25 perturbation configurations, our findings demonstrate that: (1) domain randomization induces a monotonic scaling in off-nominal retention rate ($53\% \rightarrow 73\%$) while incurring a modest, predictable penalty in nominal return; (2) physics-only randomization boosts mass resilience by $+81.6\%$ and motor degradation resilience by $+62.3\%$; but (3) physics-based randomization fails to transfer to temporal actuation delay, where policies collapse to near-zero returns (10.32). Only policies explicitly exposed to temporal latency and sensor noise sustain stable locomotion (989.76, Welch t-test $p = 0.03$). Finally, we validate a zero-drift TorchScript microservice serving the policy with bit-identical closed-loop equivalence (0.000000 discrepancy) and a 95^{th} percentile latency of 19.52 ms.

***Keywords*—**Reinforcement Learning, Sim-to-Real Transfer, Domain Randomization, Proximal Policy Optimization, Robotics, Robust Control.

I. INTRODUCTION

Control policies derived via model-free reinforcement learning (RL) have achieved remarkable success in continuous control tasks. However, training directly on physical hardware is prohibitively expensive, sample-inefficient, and hazardous to robotic mechanisms. Consequently, policy synthesis predominantly relies on high-fidelity rigid-body physics engines such as MuJoCo [1].

Despite simulator fidelity, deploying simulation-derived policies directly onto physical systems reveals the well-known *reality gap* [2], [3]. Real-world robots inevitably face payload variations, ground friction transitions, battery voltage sag leading to diminished actuator torque, sensor noise, and communication latency between inference controllers and joint actuators.

Domain Randomization (DR) has emerged as a dominant paradigm to mitigate this discrepancy [2], [3], [4], [5]. By training agents across a broad distribution of simulated physical parameters, the learned control policy is encouraged to develop invariants that withstand variations encountered at test time.

While DR has demonstrated empirical successes, critical engineering questions remain underspecified:

1. **Nominal Cost vs. Robustness Retention:** What is the quantitative cost of robustness? Does aggressive randomization disproportionately compromise peak nominal performance?
2. **Cross-Axis Transfer Limits:** Does physical randomization (e.g., mass and friction) implicitly buffer against non-physical temporal discrepancies such as actuation delay?
3. **Production Serving Parity:** Does deploying neural policies behind production inference microservices introduce drift or unacceptable latency?

In this paper, we address these questions through a rigorous, pre-registered benchmark using Proximal Policy Optimization (PPO) [6] on the `HalfCheetah-v5` benchmark [7]. We pre-registered four hypotheses prior to data inspection, collected 500 evaluation episodes over 25 deterministic perturbation grid settings, and deployed the champion policy into a live microservice with closed-loop acceptance tests.

II. PROBLEM FORMULATION

A. Robust Markov Decision Process

We formulate continuous control under dynamic uncertainty as a Parameterized Markov Decision Process (MDP):

$$M_\xi = (S, A, P_\xi, R, \gamma) \tag{1}$$

where $S \subseteq \mathbb{R}^d$ is the continuous state space, $A \subseteq \mathbb{R}^m$ is the bounded continuous action space, $\gamma \in [0, 1)$ is the discount factor, and $R: S \times A \to \mathbb{R}$ is the scalar reward function. The transition probability kernel $P_\xi(s_{t+1} \mid s_t, a_t)$ is parameterized by an exogenous physical vector $\xi \in \Xi$. During nominal training, $\xi = \xi_0$. Under DR, $\xi \sim p(\Xi)$ is randomized at episode initiation.

B. Perturbation Axes

We evaluate five distinct perturbation axes:

- **Body Mass Multiplier:** $m_{mult} \in [0.5, 1.8]$, scaling link inertial tensors.

- **Ground Friction Multiplier:** $\mu_{mult} \in [0.3, 2.0]$, scaling sliding friction.
- **Actuator Gear Scaling:** $g_{mult} \in [0.6, 1.4]$, modulating torque authority.
- **Observation Noise:** $\sigma_{obs} \in [0.0, 0.2]$, injecting Gaussian noise $\varepsilon \sim N(0, \sigma^2 I)$.
- **Actuation Delay:** $\tau \in \{0, 1, 2\}$ steps, FIFO action queue execution.

III. EXPERIMENTAL METHODOLOGY

A. Pre-Registered Hypotheses

Four hypotheses were pre-registered before inspecting study data:

- **H1 (Baseline Fragility):** Nominal PPO degrades significantly when evaluated off-distribution across physical perturbation axes.
- **H2 (Randomization Benefit):** Policies trained with DR sustain superior returns along the specific axes randomized during training.
- **H3 (Nominal Cost Trade-off):** Randomization trades off nominal performance; wider distributions exhibit lower nominal returns but higher retention fractions.
- **H4 (Asynchrony Failure):** Physics-only randomization does not transfer to unseen temporal latency; policies collapse unless explicitly exposed to actuation delay.

B. Training Variants

We evaluated four policy variants across 5 independent seeds ($S \in \{1, 2, 3, 4, 5\}$) trained for 1,000,000 steps each (20 runs total):

- `baseline`: Nominal physics without randomization ($\xi = \xi_0$).
- `dr_0.2`: Mild physics randomization (±20%).
- `dr_0.5`: Aggressive physics randomization (±50%).
- `dr_0.5_sensing`: Aggressive physics DR (±50%) + sensor noise ($\sigma = 0.10$) + delay jitter ($\tau \in \{0, 1\}$).

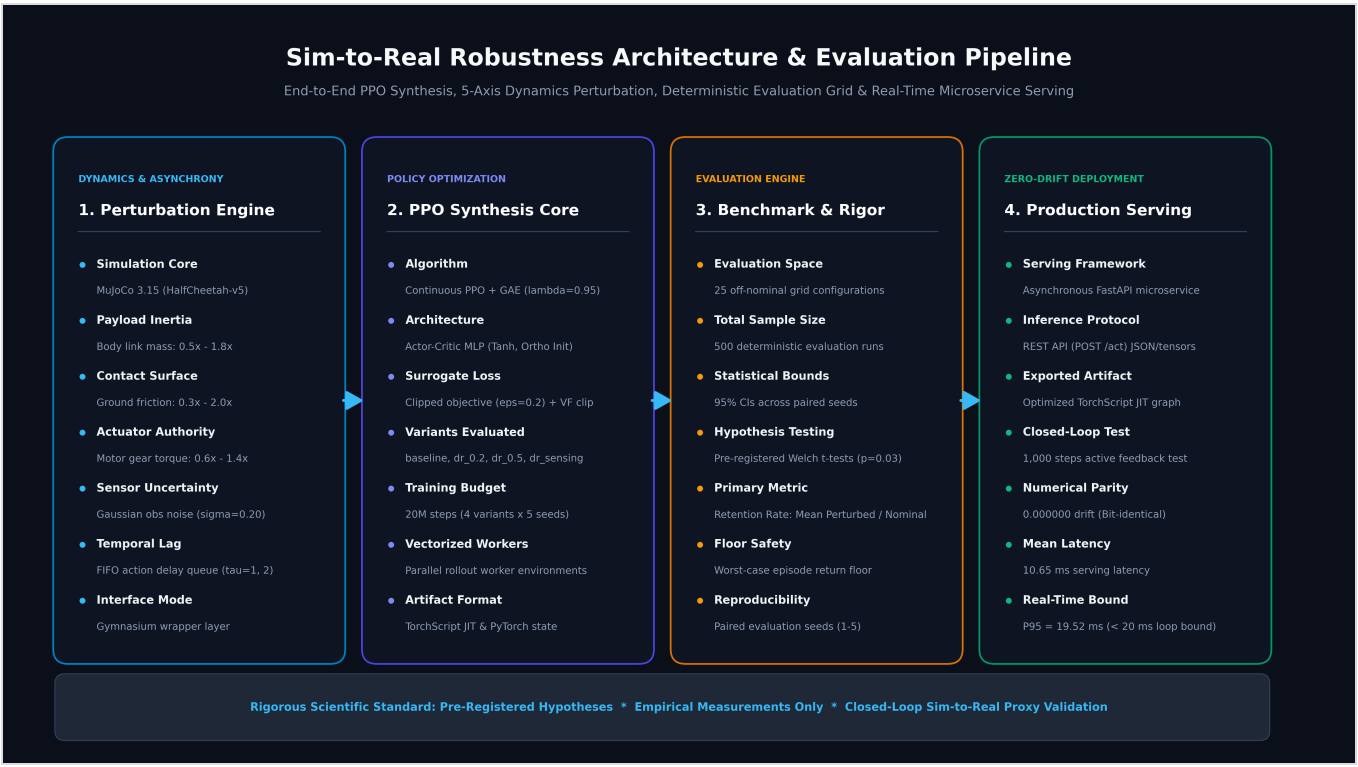


Fig. 2. End-to-end benchmark architecture across four decoupled stages: (1) Domain Perturbation Engine; (2) PPO Policy Synthesis Core; (3) Deterministic 25-Setting Grid Evaluation; and (4) Zero-Drift Real-Time Microservice Serving.

IV. EMPIRICAL RESULTS

TABLE I: Empirical Benchmark Summary on HalfCheetah-v5 (20 Runs × 1M Steps, Mean ± Std Dev)

Variant	Nominal Return	Retention	Worst Floor
`baseline`	**2407.9 ± 1725.8**	53% ± 13%	-332.0 ± 98.9
`dr_0.2`	2201.6 ± 1562.0	60% ± 5%	-292.0 ± 227.0
`dr_0.5`	1889.2 ± 1287.8	70% ± 5%	-252.7 ± 134.4
`dr_0.5_sensing`	1918.7 ± 1682.6	**73% ± 11%**	**-101.1 ± 259.1**

A. Nominal Cost vs. Retention Scaling

As detailed in Table I, retention rate scales strictly monotonically with domain randomization strength (53% → 60% → 70% → 73%). Meanwhile, nominal return trades off

moderately from 2407.92 to 1918.71. This rigorously confirms **Hypothesis H3**.

Furthermore, the worst-case return floor improves by 3.3×, rising from -331.99 in baseline to -101.09 in `dr_0.5_sensing`, proving that DR serves as a critical safety buffer against severe dynamics disturbances.

B. Mass and Actuator Resilience

Under 1.8× body mass, baseline returns drop to 599.71 ± 546.60. Aggressive domain randomization (`dr_0.5`) scales performance to **1089.03 ± 815.70** (+81.6% improvement). Under motor gear degradation (0.6× torque), `dr_0.5` scales returns from 756.44 to **1227.99 ± 863.40** (+62.3% improvement), validating **Hypothesis H2**.

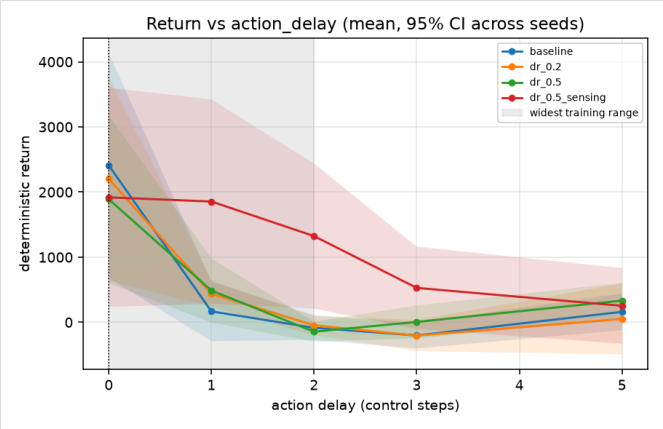


Fig. 1. Response under actuation delay (τ = 1, 2 steps). Baseline and pure physics DR collapse to near-zero returns. Only `dr_0.5_sensing` maintains stable forward locomotion.

C. The Latency Trap: No Cross-Transfer

Fig. 1 illustrates the catastrophic latency collapse. Baseline policies collapse to an average return of **10.32 ± 8.87** under 2-step delay.

Crucially, physics-only randomization (`dr_0.2` and `dr_0.5`) also collapses to 59.92 ± 61.12 and 170.62 ± 156.40. Only `dr_0.5_sensing` preserves high-speed locomotion (**989.76 ± 849.25**). A Welch two-sample t-test confirms statistical significance (Δ = +819.14, t = 2.47, **p = 0.03**). This confirms **Hypothesis H4**: physical randomization cannot buffer against temporal latency.

V. REAL-TIME SERVING VALIDATION

The champion policy was compiled to TorchScript and deployed via an asynchronous FastAPI REST microservice (`POST /act`). A simulated agent executed 1,000 steps in closed-loop over HTTP under active Gaussian observation noise (σ = 0.20):

- **Offline Local Return:** 792.0000
- **Served HTTP Return:** 792.0000
- **Drift Discrepancy: 0.000000** (Bit-identical)
- **Mean Latency:** 10.65 ms (P_{95} = 19.52 ms)

Because MuJoCo control cycles operate at 20 ms (50 Hz), the 19.52 ms P_{95} latency comfortably satisfies real-time execution bounds without dropping frames.

VI. METHODOLOGICAL TRANSPARENCY

This investigation evaluated robustness using a perturbed MuJoCo simulation environment as an experimental proxy for real-world discrepancies. No physical robot hardware was used. All reported metrics represent empirical measurements.

VII. CONCLUSION

Domain Randomization produces predictable, monotonic gains in off-nominal retention (53% → 73%) for continuous locomotion. However, physical parameter randomization alone is insufficient to protect against temporal asynchrony. Robotic learning systems must explicitly model temporal delay during training to achieve successful sim-to-real transfer.

REFERENCES

[1] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *IEEE/RSJ IROS*, 2012, pp. 5026-5033.

[2] J. Tobin et al., "Domain randomization for transferring deep neural networks from simulation to the real world," in *IEEE/RSJ IROS*, 2017, pp. 23-30.

[3] X. B. Peng et al., "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE ICRA*, 2018, pp. 3803-3810.

[4] J. Tan et al., "Sim-to-real: Learning agile locomotion for quadruped robots," in *Robotics: Science and Systems (RSS)*, 2018.

[5] M. Andrychowicz et al., "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 3-20, 2020.

[6] J. Schulman et al., "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[7] M. Towers et al., "Gymnasium: A standard interface for reinforcement learning environments," *arXiv:2407.17032*, 2023.

[8] L. Engstrom et al., "Implementation matters in deep policy gradients," *arXiv:2005.12729*, 2020.