PINNs Meet Umbrella Reinforcement Learning: Improved Convergence and Training Consistency

Anonymous authors
Paper under double-blind review

Abstract

Hard reinforcement learning problems combine sparse or delayed rewards, state traps, and the absence of a terminal state. Umbrella Reinforcement Learning (URL) addresses them with a continuous ensemble of agents described by a discounted state occupancy density and a steady-state value function, which enter the policy update through an entropy-augmented reward and a modified advantage. The original URL implementation trains these two quantities with stochastic semi-gradient updates that do not directly enforce their governing equations. We introduce PINN-URL, a physics-informed training scheme that learns both of them by minimizing their squared strong-form residuals, using sinusoidal representation networks (SIRENs) and exact sums over discrete actions, while retaining the URL policy update. On Multi-Valley Mountain Car and StandUp, PINN-URL converges faster and to a higher final return, with substantially lower run-to-run variability. Tuned PPO, including long-horizon, reward-shaped, and locally imitation-assisted variants, does not reach comparable return at the target discretization within its interaction budget. In representative runs, PINN-URL's learned value function and occupancy density agree more closely with policy-dependent simulation references. Ablations show that the joint state–action entropy is essential on Multi-Valley Mountain Car, and that neither the network architecture nor the learning configuration alone is advantageous on both benchmarks.

1 Introduction

Many reinforcement learning (RL) problems combine three features: sparse or delayed rewards, state traps, and the absence of a terminal state. Following Nuzhin & Brilliantov (2024; 2025), we refer to such problems as *hard problems*. Each feature makes rewarding behavior harder to discover or retain with trajectory-based methods, such as policy-gradient and actor-critic algorithms (Schulman et al., 2017). With sparse or delayed rewards, only a few sampled trajectories reach the small rewarded region, and even for these, credit must be assigned to actions taken long before (Sutton & Barto, 2018). State traps, regions that can be left only through coordinated action sequences, keep most trajectories in low-reward parts of the state space. Even when an escape is found, the policy must learn to reproduce the entire sequence, since small deviations can return the agent to the trap (Ecoffet et al., 2021). Finally, without a terminal state, the problem cannot be reduced to reaching a goal within a finite trajectory: optimal behavior is a sustained dynamical regime that balances reward frequency and magnitude against the agent's control capabilities, and trajectories must be long enough to reveal whether this regime is maintained. These features also compound one another: traps make rewards rarer, and the absence of termination lengthens the trajectories over which credit must be assigned.

Given the state-rate field of the environment, Umbrella Reinforcement Learning (URL) evolves a continuous ensemble of agents that covers the state space instead of learning from sampled trajectories, an idea motivated by umbrella sampling in statistical physics (Torrie & Valleau, 1977; Nuzhin & Brilliantov, 2025). URL describes the ensemble by a normalized discounted state occupancy density and a steady-state value function, which satisfy stationary transport and value equations coupled through the effective Umbrella reward. The density enters this reward through a joint state–action

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.

entropy term that favors coverage of the state space until rewarded states are found, while the value and its state gradient enter the modified advantage used in the policy-gradient update. The accuracy of both fields therefore strongly affects policy training.

The original URL implementation learns these fields with semi-gradient updates, as in semi-gradient temporal-difference learning (Sutton & Barto, 2018): they drop part of the gradient of the squared equation residuals to reduce computational cost (Nuzhin & Brilliantov, 2025), at the price of no longer following the gradient of any objective, which can make training unstable (Baird, 1995). We introduce Physics-Informed Neural Network-based Umbrella Reinforcement Learning (PINN-URL), which retains the URL policy update but minimizes the squared strong-form residuals of both equations with their full gradients, as in physics-informed neural networks (PINNs) (Raissi et al., 2019). The fields are represented by sinusoidal representation networks (SIRENs), whose smooth derivatives suit derivative-based residual losses (Sitzmann et al., 2020), and policy averages are computed exactly over the discrete actions. Because the dynamics are deterministic and the action sums are exact, the residuals are evaluated without the double-sampling bias that residual-gradient methods face with sampled transitions (Baird, 1995).

Exploration methods address sparse rewards through intrinsic bonuses or archive-based exploration (Burda et al., 2018; Ecoffet et al., 2021), and maximum-entropy RL regularizes the policy entropy (Haarnoja et al., 2018); URL instead maximizes the joint state–action entropy of the ensemble, which also rewards coverage of the state space (Hazan et al., 2019). Physics-informed neural methods have been applied to PDE-based policy iteration, model-based RL, and nonlinear optimal control (Meng et al., 2024; Ramesh & Ravindran, 2023; Wang & Wu, 2024), and state-density methods to reachable-state characterization and density-constrained RL (Meng et al., 2022; Qin et al., 2021). PINN-URL differs in applying residual learning to the coupled steady-state equations of URL, in which the learned density feeds back into the reward.

Our contributions are as follows:

(i) We introduce PINN-URL, which learns the URL occupancy density and value function by minimizing the strong-form residuals of their governing equations with full gradients, while retaining the URL policy update.

(ii) We compare PINN-URL with the original URL implementation on two hard problems, MVMC and StandUp, and show that PINN-URL converges faster, reaches a higher final return, and is markedly more consistent across runs.

(iii) Evaluating both methods by the residuals of the same field equations, we show that PINN-URL lowers these residuals by about two orders of magnitude and that its learned fields agree more closely with simulation references.

(iv) By exchanging the value and density architectures and learning configurations between the two methods and ablating the entropy terms, we show that the gains require the complete PINN-URL configuration and that the joint state–action entropy is essential on MVMC.

(v) We show that the studied problems remain hard for a trajectory-based method: even under relaxed constraints neither systematic tuning of PPO nor modifications of its training setup, including reward shaping, longer horizons, extended training, and local imitation near the goal, yield return comparable to URL within the tested budgets. Only coarser time steps yield substantial return, but such a discretization no longer represents the original task.

2 Background and Problem Setting

2.1 Umbrella Reinforcement Learning

Consider a continuous-time system with state s, discrete action $a \in \mathcal{A}$, and reward rate $r(s, a)$. Actions are sampled from a policy $\pi_\theta(a \mid s)$, and the state follows deterministic dynamics $\dot{s} = v(s, a)$ with vector field v (Nuzhin & Brilliantov, 2025). URL considers a continuous ensemble of agents with state density $p_t(s)$ and initial density $p_0(s)$. With the policy-averaged vector field $\bar{v}_\pi(s) = \mathbb{E}_{a\sim\pi_\theta}[v(s, a)]$, the ensemble obeys the continuity equation $\partial_t p_t + \nabla_s \cdot (p_t \bar{v}_\pi) = 0$ and is described by the normalized discounted state occupancy density

$$p(s) = |\log \gamma| \int_0^\infty \gamma^t p_t(s)\, dt, \tag{1}$$

where $\gamma \in (0,1)$ is the discount factor and $|\log\gamma|$ normalizes p. We refer to p simply as the density below.

URL maximizes the discounted return augmented with the joint state–action entropy of the ensemble,

$$\mathcal{J}_{\mathrm{URL}}(\pi_\theta) = J(\pi_\theta) + \alpha\, H[p\cdot\pi_\theta], \tag{2}$$

where $\alpha > 0$ is the entropy weight and

$$J(\pi_\theta) = \int_0^\infty \gamma^t\, \mathbb{E}_{s\sim p_t,\, a\sim\pi_\theta}[r(s,a)]\, dt, \qquad H[p\cdot\pi_\theta] = -\,\mathbb{E}_{s\sim p,\, a\sim\pi_\theta}\big[\log\big(p(s)\,\pi_\theta(a\mid s)\big)\big]\,.$$

By equation 1, $J(\pi_\theta) = |\log\gamma|^{-1}\,\mathbb{E}_{s\sim p,\, a\sim\pi_\theta}[r]$, so the objective can be written as $\mathcal{J}_{\mathrm{URL}} = |\log\gamma|^{-1}\,\mathbb{E}_{s\sim p,\, a\sim\pi_\theta}[r_U]$ with the effective Umbrella reward

$$r_U(s,a) = r(s,a) - \widetilde{\alpha}\,[\log p(s) + \log\pi_\theta(a\mid s)]\,, \qquad \widetilde{\alpha} = \alpha|\log\gamma|. \tag{3}$$

Unlike maximum-entropy RL, which regularizes the policy entropy alone (Haarnoja et al., 2018), this reward also contains the state-entropy term $-\widetilde{\alpha}\log p$, which favors coverage of the state space.

Let $V(s)$ denote the steady-state value, that is, the expected discounted Umbrella return from state s. The Umbrella reward enters the modified advantage

$$A_U(s,a) = r_U(s,a) + v(s,a)\cdot\nabla_s V(s) - |\log\gamma|\, V(s), \tag{4}$$

which is the continuous-time temporal-difference error of the Umbrella reward (Doya, 2000). For a fixed policy, the density satisfies the stationary transport equation $\nabla_s\cdot(p\,\bar{v}_\pi) + |\log\gamma|\,(p - p_0) = 0$. Using the identity $\nabla_s\cdot(p\,\bar{v}_\pi) = p\,\mathbb{E}_{a\sim\pi_\theta}\big[\nabla_s\cdot v + v\cdot\nabla_s\log(\pi_\theta p)\big]$, its left-hand side can be written per action as

$$D_U(s,a) = p(s)\big[\nabla_s\cdot v(s,a) + v(s,a)\cdot\nabla_s\log\big(\pi_\theta(a\mid s)\,p(s)\big)\big] + |\log\gamma|\,\big(p(s) - p_0(s)\big), \tag{5}$$

whose negative, evaluated at one sampled action per state, gives the growth-rate estimate used in the original implementation (Nuzhin & Brilliantov, 2025). Evaluating D_U requires, besides the vector field v, its divergence $\nabla_s\cdot v$; both characterize the environment and are inputs to the algorithm. The value and density are then determined by

$$\mathbb{E}_{a\sim\pi_\theta}[A_U(s,a)] = 0, \qquad \mathbb{E}_{a\sim\pi_\theta}[D_U(s,a)] = 0, \tag{6}$$

which are coupled through the Umbrella reward, in which the density appears. Differentiating equation 2 gives the policy gradient (Nuzhin & Brilliantov, 2025)

$$\nabla_\theta\mathcal{J}_{\mathrm{URL}} = \frac{1}{|\log\gamma|}\,\mathbb{E}_{s\sim p,\, a\sim\pi_\theta}[\nabla_\theta\log\pi_\theta(a\mid s)\,A_U(s,a)]\,. \tag{7}$$

Although H depends on the policy also through the logarithm of $p\,\pi_\theta$, differentiating the logarithm contributes nothing because $p\,\pi_\theta$ is normalized, so the entropy enters the gradient only through the effective reward. In continuous time, the action-value function collapses to V, and the advantage is defined as a rate, which yields equation 4.

2.2 Original URL Implementation

The original URL implementation, which we refer to as Original URL, represents the policy, value, and density by separate networks π_θ, V_ϕ, and p_η (Nuzhin & Brilliantov, 2025). At each iteration, it samples a batch $\mathcal{B}$ of states $s\sim\rho$ from a domain-covering distribution $\rho(s)$, in practice uniform, with one action $a\sim\pi_\theta(\cdot\mid s)$ per state. For the value and density, any covering distribution is suitable, since the field equations hold at every state. For the policy, ρ replaces the p-weighted state average in equation 7; for tabular policies in standard discounted problems, such reweighting does not decrease the value in any state for sufficiently small steps and preserves the stationary points of the objective (Degris et al., 2012). Denoting by $\hat{\mathbb{E}}_{\mathcal{B}}$ the empirical average over this batch and evaluating A_U and D_U with π_θ, V_ϕ, and p_η, the three networks are updated as

$$\theta \leftarrow \theta + \beta_\pi\,\hat{\mathbb{E}}_{\mathcal{B}}[\nabla_\theta\log\pi_\theta(a\mid s)\,A_U(s,a)]\,, \tag{8}$$

$$\phi \leftarrow \phi + \beta_V\,\hat{\mathbb{E}}_{\mathcal{B}}[\nabla_\phi V_\phi(s)\,A_U(s,a)]\,, \tag{9}$$

$$\eta \leftarrow \eta - \beta_p\,\hat{\mathbb{E}}_{\mathcal{B}}[\nabla_\eta\log p_\eta(s)\,D_U(s,a)]\,. \tag{10}$$

In value and density updates, $A_U(s,a)$ and $D_U(s,a)$ estimate single-action residuals of equation 6,

$$R_V(s) = \mathbb{E}_{a\sim\pi_\theta}[A_U(s,a)]\,, \qquad R_p(s) = \mathbb{E}_{a\sim\pi_\theta}[D_U(s,a)]\,, \tag{11}$$

which vanish at the exact solution. To reduce computational cost, these updates follow only part of the gradient of the squared residuals (Nuzhin & Brilliantov, 2025). However, such semi-gradient updates do not minimize any objective and are not guaranteed to converge (Baird, 1995).

2.3 Benchmark Problems

We evaluate Original URL and PINN-URL on Multi-Valley Mountain Car (MVMC) and StandUp, the two hard problems introduced with URL (Nuzhin & Brilliantov, 2025). In both, the reward is sparse, and entering the rewarded region does not terminate the dynamics, so a successful policy must sustain rewarding behavior rather than merely reach the rewarded region.

In MVMC, the state is $s = (x, \dot{x})$, where x and $\dot{x}$ are the car position and velocity, and the agent chooses between two discrete acceleration actions. The car starts in one of two outer valleys, which act as traps that can be left only by building up oscillations. It is rewarded only in a narrow interval around the top of the central hill.

StandUp models an overdamped two-link arm whose state consists of two joint angles. The agent selects one of four discrete torque combinations and is rewarded only near the upright configuration. The first link rises only under the torque combination that folds the second link, and the second link straightens only while the first descends, so the upright configuration is reached only through a coordinated sequence of folding and unfolding.

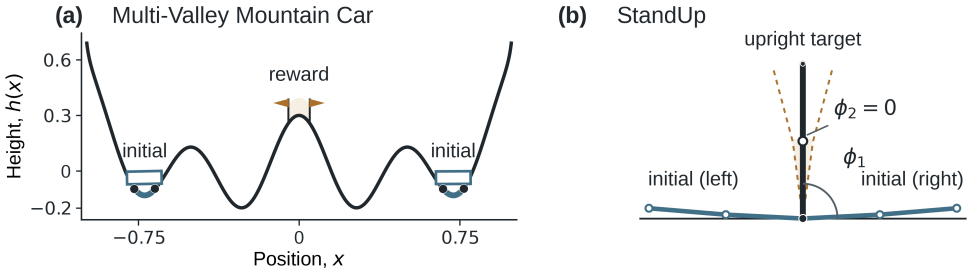


Figure 1: Benchmark geometry of Multi-Valley Mountain Car (left) and StandUp (right), showing the rewarded regions and state-space structure.

The two benchmarks differ in the domain and the dynamics. In MVMC, the domain is rectangular and the vector field is divergence-free. In StandUp, the domain is constrained by the geometry of the arm and the floor plane, and the vector field has nonzero divergence, which contributes to the density residual, equation 5. Detailed dynamics, reward regions, initial distributions, and boundary representations are given in Appendix A.

3 PINN-URL

Physics-informed neural networks (PINNs) learn the solution of a differential equation by making the equation itself the training objective: a network is fitted so that the residual of the equation vanishes at sampled collocation points, with the required derivatives computed by automatic differentiation (Raissi et al., 2019). The URL value and density are precisely such solutions, defined by the stationary value and transport equations in equation 6. PINN-URL therefore learns both fields by directly minimizing the squared residuals of these equations, so that the training loss itself measures how well the fields satisfy them, while keeping the URL policy update unchanged. Algorithm 2 in Appendix B summarizes the training procedure.

At each iteration, PINN-URL samples a batch $\mathcal{B}$ of admissible collocation states $s \sim \rho$ and evaluates the field residuals on it, with the expectations over actions computed exactly as sums over $\mathcal{A}$. The value and density losses are

$$\mathcal{L}_V = \hat{\mathbb{E}}_{\mathcal{B}}\big[R_V^2\big] + \lambda_{\text{extra}}\,\mathcal{L}_V^{\text{goal}}, \qquad \mathcal{L}_p = \hat{\mathbb{E}}_{\mathcal{B}}\big[R_p^2\big]\,, \tag{12}$$

where λ_{extra} is a weight and $\mathcal{L}_V^{\text{goal}}$ is a weighted mean of R_V^2 over an additional batch of states sampled from the rewarded region, the set of admissible states with nonzero reward. This batch reduces the value error in the narrow rewarded region, where it was found to be concentrated in preliminary runs; its sampling details and weights are given in Appendix B.

The value and density networks are trained by gradient descent on $\mathcal{L}_V$ and $\mathcal{L}_p$, each only with respect to its own parameters and with the other networks held fixed. The policy is updated by equation 8 on the same batch, with one sampled action per state.

The value and density are represented by sinusoidal representation networks (SIRENs), whose smooth derivatives suit derivative-based residual losses (Sitzmann et al., 2020). As in Original URL, both networks take fixed reflecting input representations that satisfy the boundary conditions by construction, so that no boundary terms enter the losses. Such hard constraints are common in physics-informed neural networks (Dong & Ni, 2021).

4 Experimental Setup

4.1 Compared Methods

Both URL methods are trained over 20 independent runs per method, each with 1.2×10^6 training iterations, and use $\gamma = 0.95$ and $\widetilde{\alpha} = 0.01$. Each training iteration performs one gradient step on each of the policy, value, and density networks, using a batch of sampled states and the known dynamics rather than environment rollouts. All settings are listed in Appendix B.

Original URL. Original URL uses 10,000 uniformly sampled states per iteration, discarding inadmissible states in StandUp, and the optimization settings selected in the original work for the same benchmarks (Nuzhin & Brilliantov, 2025).

PINN-URL. PINN-URL uses 10,000 admissible collocation states and 2,500 additional states from the rewarded region per iteration. The value and density networks are trained with AdamW, and all networks use cosine learning-rate schedules and network-specific gradient clipping.

PPO. As a trajectory-based baseline, we train PPO (Schulman et al., 2017) on the same tasks, with the same time step $\Delta t = 0.1$, sparse reward, and horizons, over 20 independent runs per environment with 2,048,000 environment interactions. Its hyperparameters are selected by a systematic search (Appendix G).

Relaxed-constraint baselines. Beyond this setting, we give PPO additional freedom that URL does not receive: coarser time steps, longer horizons, and extended training and, in MVMC, potential-based reward shaping (Ng et al., 1999) and a local imitation loss near the goal based on a physics-based controller whose actions are never executed in the environment. All variants are evaluated on the original sparse task reward; implementation details are given in Appendices G.3–G.4.

4.2 Evaluation Metrics

Policies are evaluated with time step $\Delta t = 0.1$ at initialization, after the first training iteration, and every 10,000 iterations thereafter. Each evaluation consists of 250 episodes with deterministic (greedy) actions from a fixed bank of 250 initial states, shared across methods, runs, and checkpoints, with horizon $H = 1000$ for MVMC and $H = 2000$ for StandUp. Returns are computed from the task reward without the entropy terms of the Umbrella reward; the precise return definition and discretization are given in Appendix C.

For each run, the normalized area under the curve (AUC) is the trapezoidal integral of the mean evaluation return over training, divided by the number of training iterations (Machado et al., 2018). It measures the mean return over training and therefore reflects both how quickly a method reaches high return and how well it maintains it.

To assess how well the learned fields satisfy their equations, we evaluate the mean-squared residuals R_V and R_p in equation 11 for both URL methods, with action expectations computed exactly, on a fixed bank of 10,000 admissible states per environment shared across methods, runs,

and checkpoints. The precise residual metric is defined in Appendix C. For Original URL these are evaluation-only metrics, whereas PINN-URL minimizes the same residuals during training. We also compare the learned fields with policy-dependent simulation references: discounted state-visitation histograms for the density and Monte Carlo rollouts for the value (Appendix E.1).

Return curves report the mean and population standard deviation across runs, and endpoint and AUC summaries report the mean and sample standard deviation, with medians where informative. Because the residuals are strongly right-skewed across runs, they are summarized by the median and interquartile range, with all runs retained. All statistics are computed across training runs: the 250 evaluation episodes are averaged into a single estimate per run and checkpoint.

4.3 Ablations

We conduct two ablations, each with ten independent runs per condition, as separate comparisons.

The swap study exchanges the value and density architectures between the two methods while keeping each learning configuration intact. This gives two combinations: SIREN networks trained with the semi-gradient updates and optimization settings of Original URL, and the ELU networks of Original URL trained with the residual losses, sampling, and optimization settings of PINN-URL. Both combinations use the same policy architecture; their training settings are given in Appendix F.

The entropy ablation varies the entropy terms of the Umbrella reward in both methods: the full reward, a policy-only entropy variant, $r - \widetilde{\alpha} \log \pi_\theta$, which excludes the state-entropy term, and a no-entropy variant with $\widetilde{\alpha} = 0$. Further details are given in Appendix F.

5 Results

5.1 Learning Performance

On both benchmarks, PINN-URL attains a higher final return than Original URL, with markedly smaller across-run dispersion: the standard deviation of the final return falls from 0.053 to 0.002 in MVMC and from 0.411 to 0.007 in StandUp. The two methods also differ in how early they approach their final return. At the iteration at which Original URL first exceeds 30% of its final return, PINN-URL has already passed 90% of its own final return on both benchmarks, (99.3% in MVMC and 99.2% in StandUp). Both effects are reflected in the normalized AUC, which is higher for PINN-URL on both benchmarks; the only exception in Table 1 is the StandUp median, which reflects the broad spread of the Original URL runs. Figure 2 shows the learning curves of Original URL and PINN-URL alongside the final returns of the PPO-based baselines. Table 1 gives the endpoint and cumulative summaries for Original URL and PINN-URL.

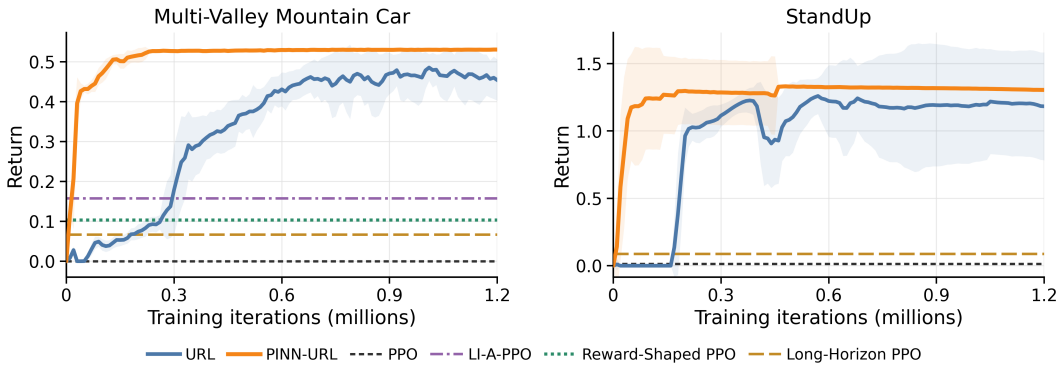


Figure 2: Return comparison for MVMC (left) and StandUp (right). URL and PINN-URL curves show the mean across 20 runs, with shaded regions indicating one standard deviation. Horizontal lines and shaded bands show the final mean $\pm$ standard deviation of PPO-based methods at their respective training budgets. LI-A-PPO and Reward-Shaped PPO are included only for MVMC.

Table 1: Performance at 1.2×10^6 training iterations. Results are reported as mean $\pm$ standard deviation across 20 runs. Best values per environment in bold.

Environment	Method	Final return	Final median	Normalized AUC
MVMC	Original URL	0.454 ± 0.053	0.457	0.329 ± 0.023
MVMC	PINN-URL	$\mathbf{0.531 \pm 0.002}$	**0.531**	$\mathbf{0.512 \pm 0.003}$
StandUp	Original URL	1.183 ± 0.411	**1.352**	0.982 ± 0.187
StandUp	PINN-URL	$\mathbf{1.304 \pm 0.007}$	1.305	$\mathbf{1.274 \pm 0.093}$

5.2 Satisfaction of the Field Equations

We next examine how closely each method satisfies the equations that define the two fields, using residuals that are a training objective for PINN-URL and evaluation-only metrics for Original URL. Figure 3 shows their median and interquartile range throughout training in MVMC. PINN-URL starts from the larger median residual for both equations, but by 10,000 training iterations both are lower and remain lower at every subsequent checkpoint. At the final evaluation, the median density-residual MSE falls from 1.292 for Original URL to 1.074×10^{-2} for PINN-URL, and the median value-residual MSE from 5.571×10^{-2} to 3.447×10^{-4}. StandUp shows the same pattern (Appendix D); across both benchmarks and both equations, the median residuals are lower by factors between 63 and 318.

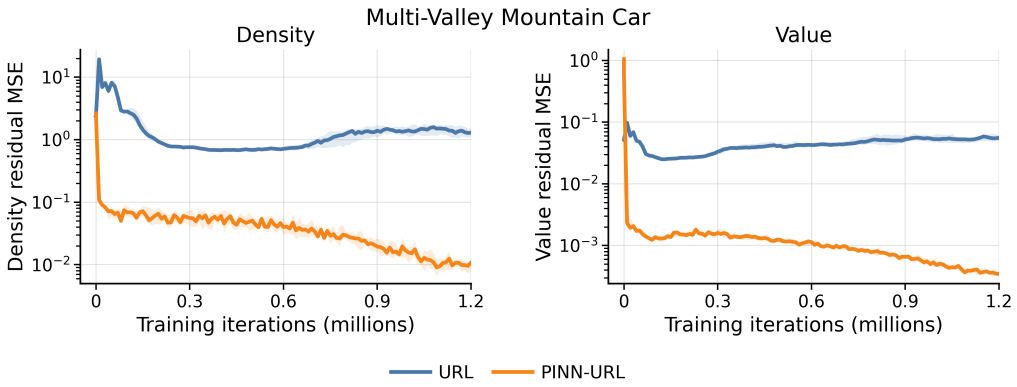


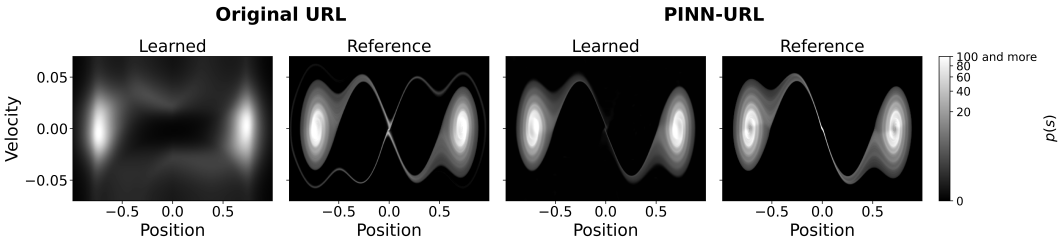
Figure 3: Mean squared residuals of the density (left) and value (right) equations in MVMC, evaluated on the shared bank of 10,000 states. Lines show the median across 20 runs; shaded regions denote the interquartile range.

The residuals of Original URL are also far less consistent across runs, which is why we summarize them by the median and interquartile range. In StandUp, its density residual is heavy-tailed: in a few runs the residual grows by many orders of magnitude at later iterations, while the median stays near 10^{-2}. Such divergence is possible for semi-gradient updates, which optimize no objective and come with no convergence guarantee (Section 2.2).

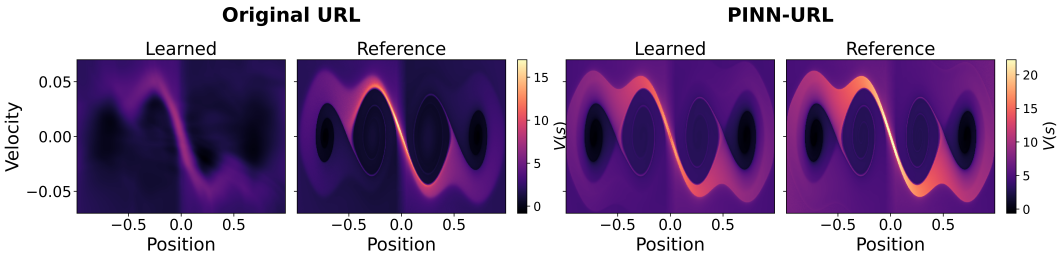
5.3 Comparison with Simulation References

The residuals of Section 5.2 are the objective that PINN-URL minimizes, so we qualitatively assess the learned fields against references that are independent of this objective: a discounted state-visitation histogram for the density and Monte Carlo rollout estimates for the value, both obtained by simulating the corresponding learned policy (Appendix E.1). Figure 4 shows the final MVMC fields of a fixed seed alongside them.

The two densities differ in how much of the reference structure they recover. Original URL captures the two dominant regions but is more diffuse than its reference and omits several narrow ridges of the state space, whereas PINN-URL recovers these regions together with the trajectory structure connecting them. The value fields differ in the same direction: the Original URL field is smoothed



(a) Density



(b) Value

Figure 4: Final MVMC density and value fields for Original URL and PINN-URL. For each method, the learned field is shown alongside its corresponding simulation reference: a discounted state-visitation histogram for density and a Monte Carlo estimate for value.

over the structure of its reference, whereas PINN-URL resolves the principal high-value ridge, the adjacent low-value regions, and the transition between them. These are the quantities through which the fields act on policy learning, since the density enters the Umbrella reward and the state gradient of the value enters the modified advantage. The spatial comparison thus agrees with the residuals of Section 5.2. StandUp fields are shown in Appendix E.

5.4 Ablation Study

Entropy terms. Both entropy terms are needed, but what they provide differs between the benchmarks. In MVMC, they are what makes the problem solvable at all: without entropy, both methods end at zero return in every run, and removing the state-entropy term alone lowers the return of both (Appendix F). In StandUp, the rewarded region is reached under every configuration, and the terms instead govern consistency. There the final return of PINN-URL varies two orders of magnitude less across seeds with the full reward than with either reduced variant, whereas the three Original URL configurations are indistinguishable within their own dispersion.

Swapped architectures. Swapping the value and density architectures between the methods, with the full reward retained, shows that neither the architecture nor the learning configuration accounts for the improvement on its own. In MVMC, the learning configuration dominates: the ELU networks of Original URL trained as in PINN-URL come close to PINN-URL itself, whereas the SIREN networks trained as in Original URL reach an order of magnitude less. In StandUp, the ordering reverses, and the ELU networks trained as in PINN-URL deteriorate after an early peak (Table 2). Since each learning configuration is transferred as a whole, the two components cannot be separated further here; what the swaps do show is that the improvement of PINN-URL follows from the complete configuration, and that either component alone can fail on one of the two benchmarks. Return curves and the full ablation settings are given in Appendix F.

5.5 Relaxed-Constraint Baselines

Under the benchmark setting with the original reward, dynamics, time step, and episode horizon, PPO does not learn either problem: after 2.048×10^6 environment interactions its final return is 0.000 ± 0.000 in MVMC and 0.014 ± 0.046 in StandUp. The outcome does not follow from the chosen hyperparameters. In a screen that varied learning rate, entropy regularization, network activation and width, rollout length, and clipping, none of the 24 MVMC runs encountered a positive

Table 2: Ablation performance at 1.2×10^6 training iterations. Results are reported as mean $\pm$ standard deviation across 10 seeds. Best values per environment and column in bold.

		Entropy terms			
Environment	Method	Full	Policy only	No entropy	Swapped arch.
MVMC	Original URL	0.460 ± 0.066	0.410 ± 0.045	0.000 ± 0.000	0.032 ± 0.034
MVMC	PINN-URL	$\mathbf{0.529 \pm 0.003}$	$\mathbf{0.503 \pm 0.013}$	0.000 ± 0.000	$\mathbf{0.524 \pm 0.013}$
StandUp	Original URL	1.057 ± 0.557	$\mathbf{1.124 \pm 0.332}$	$\mathbf{1.063 \pm 0.255}$	$\mathbf{1.253 \pm 0.094}$
StandUp	PINN-URL	$\mathbf{1.313 \pm 0.005}$	0.644 ± 0.679	0.996 ± 0.578	0.854 ± 0.247

reward during training, and StandUp yielded only partial learning signals (Appendix G.1). We therefore examine several PPO variants under relaxed constraints.

Raising the interaction budget to 6.016×10^6 lets PPO develop sustained return in MVMC, of order 10^{-1}, and extending the episode horizon to $10{,}000$ steps reaches a comparable level more gradually and with large variability. Potential-based shaping, and the local imitation loss built on it, reach that level within the first few hundred thousand interactions and maintain it, so denser reward and domain knowledge remove the exploration bottleneck without raising the level that PPO attains. All four variants remain about five times below the 0.531 of PINN-URL, and the goal is reached in only 10–15% of MVMC evaluation episodes.

Coarser time steps are the only relaxation that lifts PPO above this level, and only beyond $\Delta t = 0.3$, below which the mean final return stays at zero. A coarser time step increases the integration error, which limits how faithfully the simulated dynamics represent the original system, and forces the agent to apply each action for longer, reducing the set of achievable behaviors. Across the relaxations that preserve the task, PPO therefore learns the problems only partially and stays well below both URL methods. Figure 2 shows the final returns of PPO and Long-Horizon PPO on both benchmarks, together with Reward-Shaped PPO and LI-A-PPO on MVMC. Return trajectories, the time-step sweep, and the configurations of all variants are given in Appendix G.

6 Discussion and Conclusion

Learning the fields from their governing equations makes PINN-URL satisfy the value and density equations far more closely than the original URL implementation and, at the same time, learn faster, to a higher return, and with far smaller variability across runs on both benchmarks. This correspondence follows the structure of URL, in which the density enters the Umbrella reward and the value enters the modified advantage, so that the accuracy of both fields acts directly on policy learning. The ablations show, however, that the improvement requires both the residual objective and the SIREN representation: neither of the two performs well on both benchmarks on its own.

The two implementations also differ in what their field updates optimize. The semi-gradient updates of Original URL follow no objective and are not guaranteed to converge, which is visible in the StandUp runs whose density residual grows by orders of magnitude during training. Minimizing the squared residuals replaces these updates by gradient descent on them, which matches with the far smaller variability of PINN-URL. Here this is possible without the double-sampling bias of residual-gradient methods, since the dynamics are deterministic and the sums over actions are evaluated exactly.

These conditions also delimit the scope of the study. It is limited to two low-dimensional benchmarks with deterministic dynamics and full observability, in which the vector field and its divergence are known.

Extending the method to higher dimensions will require replacing uniform collocation with adaptive sampling of the states at which the residuals are enforced. Within the problems studied here, this approach yields a faster-converging and far more consistent realization of Umbrella Reinforcement Learning on hard problems.

REPRODUCIBILITY STATEMENT

The experimental setup and evaluation protocol are described in Section 4, and the implementation details, hyperparameters, and fixed seeds in Appendix B. The source code is available at `https://anonymous.4open.science/r/PINN-URL-ABRLIA/`.

AI USE STATEMENT

In this work, the research idea, hypotheses, methodology, and experimental design were developed by the authors without generative AI assistance. We used generative AI tools to implement and debug parts of the training and evaluation code, develop plotting and scientific-illustration scripts, including code for visualizing the benchmark environments, and discuss possible interpretations of experimental results. The authors independently analyzed the experimental data and figures, critically assessed AI-assisted suggestions, and determined the conclusions reported in the paper.

We additionally used generative AI tools for literature search and discovery, feedback on manuscript structure and experimental presentation, and drafting and editing parts of the manuscript to improve its clarity and readability.

We did not use generative AI tools to develop the underlying research concept, propose or refine hypotheses, design or provide feedback on the methodology or experiments, formulate mathematical claims, assist with mathematical derivations or proofs, generate synthetic datasets, clean or reformat datasets, or perform qualitative or thematic data analysis. Translation was not applicable to this work.

All AI-assisted text and code were reviewed by the authors. Every reported experimental result was recomputed from the retained experimental artifacts, and all AI-assisted code was verified against its intended behavior. The authors conducted the experiments and take responsibility for the final content of this work, including its text, claims, results, code, and other artifacts produced with the aid of generative AI.

REFERENCES

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine learning proceedings 1995*, pp. 30–37. Elsevier, 1995.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018. URL `https://arxiv.org/abs/1810.12894`.

Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

Suchuan Dong and Naxian Ni. A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks. *Journal of Computational Physics*, 435: 110242, 2021.

Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1): 219–245, 2000.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, 2021. doi: 10.1038/s41586-020-03157-9.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, 2018. URL `https://proceedings.mlr.press/v80/haarnoja18b.html`.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International conference on machine learning*, pp. 2681–2691. PMLR, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Yiming Meng, Ruikun Zhou, Amartya Mukherjee, Maxwell Fitzsimmons, Christopher Song, and Jun Liu. Physics-informed neural network policy iteration: Algorithms, convergence, and verification. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35378–35403, 2024. URL https://proceedings.mlr.press/v235/meng24b.html.

Yue Meng, Dawei Sun, Zeng Qiu, Md Tawhid Bin Waez, and Chuchu Fan. Learning density distribution of reachable states for autonomous systems. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 124–136, 2022. URL https://proceedings.mlr.press/v164/meng22a.html.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 278–287, 1999. URL https://people.eecs.berkeley.edu/~russell/papers/icml99-shaping.pdf.

Egor E Nuzhin and Nikolay V Brilliantov. Umbrella reinforcement learning-a tool for hard problems. In *2024 4th International Conference on Robotics, Automation and Artificial Intelligence (RAAI)*, pp. 104–111. IEEE, 2024.

Egor E. Nuzhin and Nikolay V. Brilliantov. Umbrella reinforcement learning - computationally efficient tool for hard non-linear problems. *Communications in Nonlinear Science and Numerical Simulation*, 143:108583, 2025. doi: 10.1016/j.cnsns.2024.108583.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Zengyi Qin, Yuxiao Chen, and Chuchu Fan. Density constrained reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8682–8692, 2021. URL https://proceedings.mlr.press/v139/qin21a.html.

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. URL https://www.jmlr.org/papers/v22/20-1364.html.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Adithya Ramesh and Balaraman Ravindran. Physics-informed model-based reinforcement learning. In *Proceedings of the 5th Annual Learning for Dynamics and Control Conference*, volume 211 of *Proceedings of Machine Learning Research*, pp. 26–37, 2023. URL https://proceedings.mlr.press/v211/ramesh23a.html.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/53c04118df112c13a8c34b38343b9c10-Abstract.html.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Glenn M. Torrie and John P. Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187–199, 1977. doi: 10.1016/0021-9991(77)90121-8.

Yujia Wang and Zhe Wu. Physics-informed reinforcement learning for optimal control of nonlinear systems. *AIChE Journal*, 70(10):e18542, 2024. doi: 10.1002/aic.18542.

A Benchmark Definitions

The two benchmarks follow the Multi-Valley Mountain Car (MVMC) and StandUp problems introduced with Original URL (Nuzhin & Brilliantov, 2025). Training uses their continuous-time state-rate fields, while numerical rollouts use $\Delta t = 0.1$. Neither environment terminates upon entering the rewarded region.

Following the original implementation (Nuzhin & Brilliantov, 2025), both methods impose reflecting boundary conditions on the value and density by construction, through fixed input representations of the corresponding networks. The representations defined below therefore enforce the boundary treatment without introducing boundary terms into the training objectives. Such hard constraints are common in physics-informed learning, where boundary conditions are built into the network rather than penalized in the loss; in particular, input transformations have been used to impose periodic boundary conditions exactly (Dong & Ni, 2021).

Multi-Valley Mountain Car. Let

$$s = (x, u), \qquad u = \dot{x},$$

with

$$x \in [-0.99, 0.99], \qquad u \in [-0.07, 0.07], \qquad a \in \{0, 1\}.$$

The terrain is

$$h(x) = \frac{1}{10}\left[\cos(2\pi x) + 2\cos(4\pi x) - \log(1 - x^2)\right],$$

and the state-rate field is

$$v(s, a) = \begin{pmatrix} u \\ (2a-1)f - g\,h'(x) \end{pmatrix}, \qquad f = 0.001, \quad g = 0.0025.$$

Because the first component depends only on u and the second only on x and a,

$$\nabla_s \cdot v(s, a) = 0.$$

The reward is

$$r(s, a) = \mathbf{1}\{|x| \le 0.05\}.$$

With

$$B_+ = (0.67, 0.77) \times (-0.01, 0.01), \qquad B_- = (-0.77, -0.67) \times (-0.01, 0.01),$$

the normalized source density is

$$p_0(s) = 250\,\mathbf{1}\{s \in B_+ \cup B_-\}.$$

The value and density branches use the fixed reflecting representation

$$\mathcal{R}_{\mathrm{MVMC}}(x,u) = \big(x,\, c(u)^2,\, c(u)d(x)\big)\,,$$

where

$$c(u) = \cos\!\left(\pi\frac{u+0.07}{0.14}\right), \qquad d(x) = \min\{|x+0.99|,\ |x-0.99|\}.$$

The policy receives the raw state. During numerical rollouts, velocity is updated and clipped to $[-0.07, 0.07]$, after which position is updated using the new velocity and clipped to $[-0.99, 0.99]$.

StandUp. Let

$$s = (\phi_1, \phi_2),$$

with admissible domain

$$0 < \phi_1 \le \pi, \qquad -\pi < \phi_2 \le \pi, \qquad 0 < \phi_1 + \frac{\phi_2}{2} \le \pi.$$

The four actions correspond to torque pairs

$$m_1(a) = \left(2\left\lfloor\frac{a}{2}\right\rfloor - 1\right)\tau, \qquad m_2(a) = (2(a \bmod 2) - 1)\,\tau,$$

so that

$$(m_1, m_2) \in \{(-\tau,-\tau),(-\tau,\tau),(\tau,-\tau),(\tau,\tau)\}.$$

The evaluated environment uses

$$g = 0.025, \qquad \tau = 2g = 0.05,$$

and

$$v(s,a) = \begin{pmatrix} m_1(a) - m_2(a) - g\cos\phi_1 \\ m_2(a) - g\cos(\phi_1+\phi_2) \end{pmatrix}.$$

Its divergence is

$$\nabla_s \cdot\, v(s,a) = g\left[\sin\phi_1 + \sin(\phi_1+\phi_2)\right].$$

With

$$\delta = \frac{\pi}{24},$$

the reward is

$$r(s,a) = \mathbf{1}\left\{\left|\phi_1 - \frac{\pi}{2}\right| \le \delta,\ |\phi_2| \le \delta\right\}.$$

Define the two initial regions

$$Q_R = \{0 < \phi_1 < \delta,\ 0 \le \phi_2 < \delta\},$$

$$Q_L = \{\pi - \delta < \phi_1 \le \pi,\ -\delta \le \phi_2 < 0\}.$$

The source density is

$$p_0(s) = \frac{1}{2\delta^2}\mathbf{1}\{s \in Q_R \cup Q_L\}.$$

The value and density branches use

$$\mathcal{R}_{\mathrm{SU}}(\phi_1,\phi_2) = (\sin\vartheta_1,\ \sin\vartheta_2)\,,$$

where

$$\vartheta_1 = \phi_1 - \frac{\pi}{2}, \qquad q(\vartheta_1) = 2\left(1 - \frac{|\vartheta_1|}{\pi}\right), \qquad \vartheta_2 = \frac{\phi_2 + \vartheta_1}{q(\vartheta_1)}.$$

The policy again receives the raw state. Uniform training proposals outside the admissible domain are rejected as specified in Appendix B.3. Numerical rollouts clip ϕ_1 to $[0,\pi]$ and then clip ϕ_2 to

$$\left[\max(-2\phi_1, -\pi),\ \min(2(\pi-\phi_1), \pi)\right].$$

Algorithm 1 Original URL training (evaluated implementation)

Require: Policy π_θ, value approximation V_ϕ, density approximation p_η, discount factor γ, effective entropy coefficient $\widetilde{\alpha}$, and domain-covering state distribution ρ.

1: **for** $k = 0, 1, \ldots, K-1$ **do**

2: Sample domain states from ρ, retaining admissible states where required; denote the retained batch by $\mathcal{B}_k = \{s_i\}_{i=1}^{B_k}$.

3: Sample one action $a_i \sim \pi_\theta(\cdot \mid s_i)$ for each $s_i \in \mathcal{B}_k$.

4: Evaluate $p_0(s_i)$, $r(s_i, a_i)$, $v(s_i, a_i)$, $\nabla_s \cdot v(s_i, a_i)$, $V_\phi(s_i)$, $p_\eta(s_i)$, and the required state derivatives by automatic differentiation.

5: Compute the sampled modified advantage $A_i = A_U(s_i, a_i)$ and the density-update coefficient

$$D_i = \log\gamma\,[p_\eta(s_i) - p_0(s_i)] - p_\eta(s_i)\left[\left(\nabla_s \log p_\eta(s_i) + \nabla_s \log \pi_\theta(a_i \mid s_i)\right)\cdot v(s_i, a_i) + \nabla_s \cdot v(s_i, a_i)\right].$$

6: Treat A_i and D_i as fixed targets and form

$$\mathcal{L}_\pi^{\mathrm{URL}} = -\frac{1}{B_k}\sum_{i=1}^{B_k} \log \pi_\theta(a_i \mid s_i)\, A_i,$$

$$\mathcal{L}_V^{\mathrm{URL}} = -\frac{1}{B_k}\sum_{i=1}^{B_k} V_\phi(s_i)\, A_i, \qquad \mathcal{L}_p^{\mathrm{URL}} = -\frac{1}{B_k}\sum_{i=1}^{B_k} \log p_\eta(s_i)\, D_i.$$

7: Form $\mathcal{L}^{\mathrm{URL}} = \mathcal{L}_\pi^{\mathrm{URL}} + \mathcal{L}_V^{\mathrm{URL}} + \mathcal{L}_p^{\mathrm{URL}}$ and update θ, ϕ, and η by gradient descent on this surrogate objective.

8: **end for**

B Implementation and Training Details

B.1 Training Procedures

Algorithms 1 and 2 state the update rules used in the core comparison. Both retain the URL effective Umbrella reward and sampled score-function policy update described in Section 2. Original URL trains its value and density approximations through semi-gradient updates, whereas PINN-URL minimizes squared strong-form residuals of the corresponding governing equations. Optimizer choice, learning-rate schedules, gradient clipping, and other numerical settings are implementation details and are therefore reported in Appendix B.3, rather than built into the algorithmic definitions.

B.2 Network Architectures and Representations

The policy, value, and density branches have disjoint trainable parameters. The policy architecture is unchanged between Original URL and PINN-URL. PINN-URL changes the value and density representations to sinusoidal representation networks (SIRENs) (Sitzmann et al., 2020) and uses a different positive-density parametrization (Table 3).

Table 3: Neural architectures in the core Original URL and PINN-URL experiments. The policy output dimension is $|\mathcal{A}| = 2$ for MVMC and 4 for StandUp.

Branch	Original URL	PINN-URL
Policy	Raw state; two width-128 Tanh hidden layers; linear $\lvert\mathcal{A}\rvert$-logit output	Raw state; two width-128 Tanh hidden layers; linear $\lvert\mathcal{A}\rvert$-logit output
Value	Reflecting representation; two width-128 ELU hidden layers; linear scalar output	Reflecting representation; six width-128 SIREN hidden layers; linear scalar output
Density	Reflecting representation; two width-128 ELU hidden layers; linear log-density output ℓ_η, with $p_\eta = \exp(\ell_\eta)$	Reflecting representation; six width-128 SIREN hidden layers; linear output z_η, with $p_\eta = \mathrm{Softplus}(z_\eta) + 10^{-6}$

Algorithm 2 PINN-URL training

Require: Policy π_θ, value approximation V_ϕ, density approximation p_η, discount factor γ, effective entropy coefficient $\widetilde{\alpha}$, admissible-domain collocation sampler ρ, and loss weights $\lambda_\pi, \lambda_V, \lambda_p, \lambda_{\text{extra}}$.

1: **for** $k = 0, 1, \ldots, K-1$ **do**

2: Sample a shared admissible collocation batch $\mathcal{B} = \{s_i\}_{i=1}^{B} \sim \rho$ and an environment-specific value-focused batch $\mathcal{B}_V$.

3: Sample one action $a_i \sim \pi_\theta(\cdot \mid s_i)$ for each $s_i \in \mathcal{B}$, evaluate $A_i = A_U(s_i, a_i)$, and form

$$\mathcal{L}_\pi = -\frac{1}{B} \sum_{i=1}^{B} \log \pi_\theta(a_i \mid s_i)\, A_i,$$

treating A_i as fixed during the policy update.

4: Evaluate V_ϕ, p_η, and their required state derivatives by automatic differentiation.

5: For every $a \in \mathcal{A}$, evaluate the policy, reward, state rate, and environment quantities required by R_V and R_p, and compute the finite-action sums exactly.

6: During field learning, treat the policy quantities as fixed; additionally treat p_η as fixed when evaluating R_V.

7: Form

$$\mathcal{L}_p = \frac{1}{B} \sum_{s \in \mathcal{B}} R_p(s)^2,$$

$$\mathcal{L}_V = \frac{1}{B} \sum_{s \in \mathcal{B}} R_V(s)^2 + \lambda_{\text{extra}} \frac{\sum_{s \in \mathcal{B}_V} w(s) R_V(s)^2}{\sum_{s \in \mathcal{B}_V} w(s)}.$$

8: Form the total objective

$$\mathcal{L} = \lambda_\pi \mathcal{L}_\pi + \lambda_p \mathcal{L}_p + \lambda_V \mathcal{L}_V.$$

9: Update θ, ϕ, and η by gradient descent on $\mathcal{L}$.

10: **end for**

For PINN-URL, the first SIREN layer uses frequency $\omega_0 = 30$ and all subsequent sine layers use $\omega_{\text{hid}} = 4$. The first sine layer uses the implemented uniform initialization on $[-1/d_{\text{in}}, 1/d_{\text{in}}]$; subsequent sine layers use the standard SIREN-scaled bound $\sqrt{6/d_{\text{in}}}/\omega_{\text{hid}}$. The final value and density linear layers use Xavier-uniform weights and zero biases. The policy branch uses the standard PyTorch linear-layer initialization (Paszke et al., 2019). The fixed reflecting representations are those defined in Appendix A and contain no trainable parameters.

B.3 Training, Sampling, and Optimization Settings

Each core run contains 1.2×10^6 completed training iterations. Both methods use

$$\gamma = 0.95, \qquad \widetilde{\alpha} = 0.01.$$

Table 4 gives the branch-specific optimizer settings. Adam (Kingma & Ba, 2015) is used for the Original URL branches and the PINN-URL policy branch, while the PINN-URL value and density branches use AdamW (Loshchilov & Hutter, 2019).

Original URL draws 10,000 uniform domain proposals per iteration, discarding inadmissible StandUp states without replenishment. PINN-URL instead uses 10,000 admissible collocation states, obtained by rejection sampling in StandUp, and 2,500 additional value-focused states per iteration. All PINN-URL loss weights are set to one: $\lambda_\pi = \lambda_V = \lambda_p = \lambda_{\text{extra}} = 1$.

For MVMC, value-focused states are sampled uniformly from

$$[-0.05, 0.05] \times [-0.07, 0.07].$$

Their normalized residual weight is

$$w_{\text{MVMC}}(s) = 1 + \mathbf{1}\{|x| \le 0.05\} + 50\,\mathbf{1}\{|x| \le 0.05,\ |u| \le 0.07\} + \mathbf{1}\left\{\big||x| - 0.05\big| \le 0.015\right\}.$$

For StandUp, value-focused states are sampled from the admissible part of the reward rectangle

$$\phi_1 \in \left[\frac{\pi}{2} - \delta,\ \frac{\pi}{2} + \delta\right], \qquad \phi_2 \in [-\delta, \delta],$$

and receive constant weight 50. Because the extra residual is normalized by $\sum w$, this constant cancels in the StandUp weighted mean.

Table 4: Optimizer settings for the core Original URL and PINN-URL runs. LR is the initial learning rate, $\mathrm{LR}_{\min}$ the cosine-schedule minimum, WD the weight decay, and Clip the gradient-norm threshold. A dash denotes an inactive scheduler minimum or clipping threshold.

Method / environment	Branch	Optimizer	LR	$\mathrm{LR}_{\min}$	WD	Clip
Original URL / MVMC	Policy	Adam	1×10^{-5}	–	5×10^{-6}	–
	Value	Adam	1×10^{-5}	–	1×10^{-4}	–
	Density	Adam	1×10^{-5}	–	5×10^{-4}	–
Original URL / StandUp	Policy	Adam	1×10^{-6}	–	5×10^{-5}	–
	Value	Adam	1×10^{-6}	–	1×10^{-5}	–
	Density	Adam	1×10^{-7}	–	5×10^{-4}	–
PINN-URL / MVMC	Policy	Adam	1×10^{-4}	1×10^{-5}	5×10^{-6}	–
	Value	AdamW	3×10^{-4}	5×10^{-6}	1×10^{-5}	5.0
	Density	AdamW	2.5×10^{-4}	5×10^{-6}	1×10^{-4}	5.0
PINN-URL / StandUp	Policy	Adam	3×10^{-5}	3×10^{-6}	1×10^{-5}	0.5
	Value	AdamW	5×10^{-4}	5×10^{-5}	1×10^{-5}	1.0
	Density	AdamW	2.5×10^{-4}	5×10^{-6}	5×10^{-4}	1.0

The PINN-URL learning rates follow cosine annealing (Loshchilov & Hutter, 2017) over the full 1.2×10^6-iteration training horizon. Gradient clipping is applied separately to the policy, value, and density parameter branches using the thresholds in Table 4. Original URL uses fixed learning rates and no gradient clipping.

Value-focused collocation. The additional PINN-URL collocation batch contributes only to the value residual objective; it is not used in the policy loss or density residual and does not add policy rollouts or environment interactions. The batch targets the narrow rewarded region, where preliminary runs showed a localized value approximation error. Return improvements were already present before this sampling refinement was introduced, so the observation is not used as a controlled ablation of the source of the return gain.

C Evaluation Protocol

The core evaluation uses 20 independent training runs per method, a common 250-state evaluation bank for return estimation, and a fixed 10,000-state diagnostic bank per environment for the value and density residuals. Table 5 summarizes the protocol.

Table 5: Core Original URL–PINN-URL evaluation protocol.

Setting	MVMC	StandUp
Independent training runs	20	20
Evaluated checkpoints	$0,\ 1,\ 10{,}000, \ldots, 1{,}200{,}000$ (122 checkpoints)	
Initial-state evaluation bank	250 fixed states	250 fixed states
Bank reuse	Both methods, all training runs, and every evaluated checkpoint	
Evaluation action	Deterministic argmax	Deterministic argmax
Episode horizon	1000	2000
Return-curve aggregation	Arithmetic mean / population SD across training runs	
Endpoint and AUC dispersion	Sample SD across training runs	
Residual diagnostic bank	10,000 fixed states	10,000 fixed admissible states
Primary residual aggregation	Median / interquartile range across all 20 runs	

The discounted return of episode e is the discretization of the continuous-time return $\int_0^{H\Delta t} \gamma^t r \, dt$ with step Δt, computed from the task reward without the entropy terms of the Umbrella reward:

$$G_e = \Delta t \sum_{t=1}^{H} \gamma^{(t-1)\Delta t} \, r_t. \tag{13}$$

Residual diagnostics use the same fixed bank of $N_r = 10{,}000$ admissible states per environment for both URL methods. For $\ell \in \{V, p\}$, the reported residual metric is

$$\mathrm{MSE}_\ell = \frac{1}{N_r} \sum_{i=1}^{N_r} R_\ell(s_i)^2. \tag{14}$$

The action expectations entering R_V and R_p are evaluated exactly.

The 250 evaluation episodes estimate one trained policy at one checkpoint and are not treated as independent training replicates. Original URL and PINN-URL runs with the same training-seed index are paired when paired run-level comparisons are reported. AUC uses the actual training-iteration coordinates, including the distinct points at iterations 0, 1, and 10,000; it is not computed by treating checkpoints as uniformly spaced. As stated in Section 4.2, this quantity summarizes performance on the common training-iteration coordinate and is not interpreted as an environment-sample-efficiency or formal convergence-rate measure. For the residual study, the same fixed environment-specific bank is reused for both methods, every training run, and every checkpoint. The additional PINN-URL value-focused states are not part of this diagnostic bank. Thus, Original URL and PINN-URL are evaluated with the same R_V and R_p operators on the same states. For Original URL these are post hoc equation-consistency diagnostics rather than training losses. All runs are retained.

D Additional Residual Diagnostics

The main paper reports the MVMC value- and density-residual trajectories using the median and interquartile range (IQR) across the 20 independent training runs. Figure 5 gives the corresponding StandUp trajectories using the same fixed 10,000-state environment-specific diagnostic bank described in Appendix C, and all 20 runs are retained.

D.1 StandUp Residual Trajectories

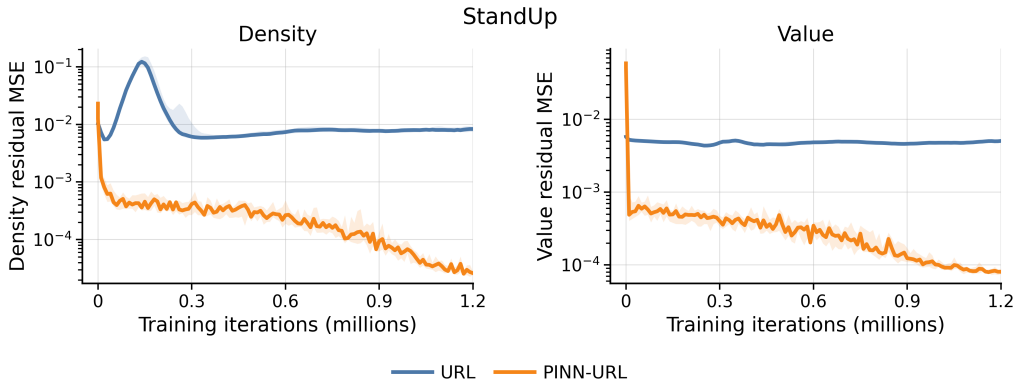


Figure 5: StandUp common strong-form residual diagnostics. Lines show medians and shaded bands show interquartile ranges across 20 independent training runs. Both panels use logarithmic vertical axes and the same fixed 10,000-state diagnostic bank for Original URL and PINN-URL. Original URL residuals are evaluation diagnostics only; PINN-URL minimizes the corresponding value and density residuals during training. All runs are retained.

At initialization, PINN-URL has the larger median residual for both StandUp equations. By the first saved checkpoint after training begins, however, its median residual is lower than Original

URL for both quantities and remains lower through the end of training. The Original URL density residual also shows a pronounced transient increase early in training before returning to the 10^{-2} regime, whereas the PINN-URL median decreases progressively into the 10^{-5} range. The Original URL value residual remains close to 5×10^{-3} over most of training, while the PINN-URL value residual continues to decrease by more than an order of magnitude beyond its initial rapid drop. At the final evaluation, the median StandUp density-residual MSE decreases from 8.308×10^{-3} for Original URL to 2.612×10^{-5} for PINN-URL, and the median value-residual MSE decreases from 5.049×10^{-3} to 8.061×10^{-5}, corresponding to reductions of approximately $318\times$ and $63\times$, respectively. PINN-URL has the lower final residual in all 20 paired runs for both equations.

E Learned Fields and Simulation References

E.1 Reference-Field Generation

The representative spatial comparisons use policy-dependent numerical references rather than analytical ground truth. Simulation-based density references are generated by rolling out the policy used for the corresponding reference construction, while Monte Carlo value references use the learned policy and neural density supplied to that calculation. Original URL and PINN-URL therefore have method-specific numerical references. The main-paper field comparison is MVMC and uses a 401×401 grid on

$$x \in [-0.99, 0.99], \qquad u \in [-0.07, 0.07].$$

Simulation-based reference density. The density reference is a discounted empirical state-visitation histogram under the stochastic categorical policy π_{ref}. It uses $N_\rho = 40{,}000$ trajectories of $T_\rho = 1000$ steps from the MVMC initial-state distribution. With

$$q = \gamma^{\Delta t} = 0.95^{0.1},$$

the unnormalized occupancy of grid cell B_{ij} is

$$M_{ij} = \sum_{e=1}^{N_\rho} \sum_{t=0}^{T_\rho - 1} q^t \, \mathbf{1}\{s_t^{(e)} \in B_{ij}\},$$

where the current state is accumulated before each transition. No kernel density estimation or smoothing is applied. If ΔA denotes the grid cell area, the histogram is normalized as

$$p_{\mathrm{ref}}(B_{ij}) = \frac{M_{ij}}{\Delta A \sum_{k,\ell} M_{k\ell}},$$

so that its numerical integral is one. The generator uses fixed histogram seed 123. The learned neural density is not renormalized before plotting, so density panels are interpreted primarily through spatial support and organization rather than exact amplitude agreement.

Monte Carlo reference value. For every grid state s, the value reference averages $N_V = 64$ stochastic-policy rollouts of $T_V = 1000$ steps. The running Umbrella reward uses the learned neural density and policy supplied to the value-reference construction, rather than the simulation histogram:

$$r_U^{\mathrm{ref}}(s_t, a_t) = r(s_t, a_t) - \widetilde{\alpha} \left[\log p_{\mathrm{net}}(s_t) + \log \pi_{\mathrm{ref}}(a_t \mid s_t) \right].$$

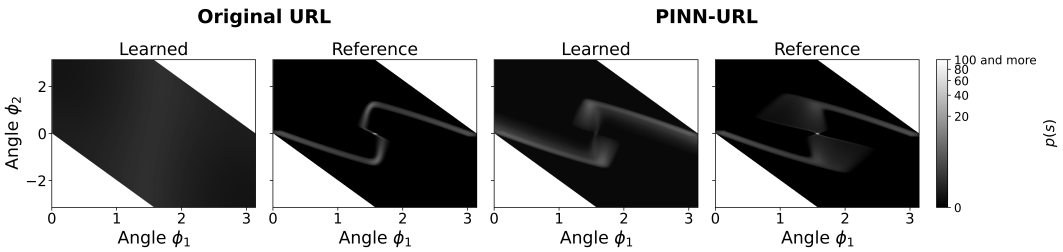
The Monte Carlo estimator is

$$V_{\mathrm{ref}}(s) = \frac{1}{N_V} \sum_{m=1}^{N_V} \Delta t \sum_{t=0}^{T_V - 1} q^t r_U^{\mathrm{ref}} \left(s_t^{(m)}, a_t^{(m)} \right), \qquad a_t^{(m)} \sim \pi_{\mathrm{ref}}(\cdot \mid s_t^{(m)}).$$

Density and action probabilities entering logarithms are floored at 10^{-12}. The value generator uses seed 456 and processes grid states in batches of 512; batching does not alter the estimator. Goal entry does not terminate the rollout.

Visualization. Density fields use a gray colormap with $\texttt{PowerNorm}(\gamma = 0.3)$ and display range $[0, 100]$; values above 100 are clipped only for visualization and the upper colorbar tick is labeled "100 and more." Value fields use the `magma` colormap with a common linear scale within each learned/reference comparison. Figures are rendered at 600 DPI. These fields provide qualitative spatial evidence; the quantitative across-run equation-satisfaction comparison is given by the common residual diagnostics.

E.2 StandUp Density Fields

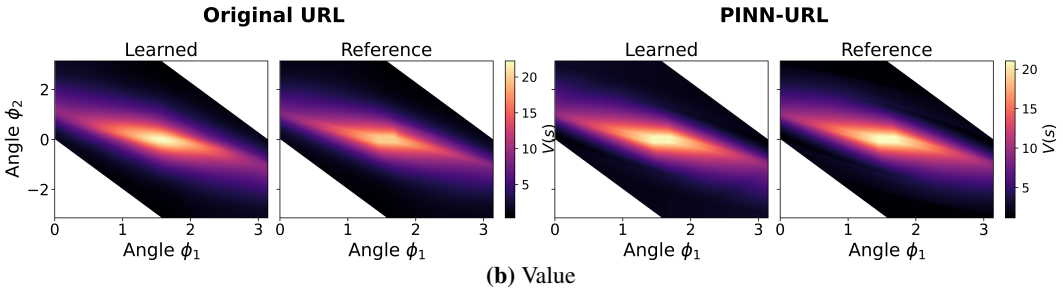


(a) Density

Figure 6: Representative StandUp density fields for Original URL and PINN-URL. For each method, the learned density is shown together with its policy-dependent simulation reference. Reference densities are obtained from discounted empirical state visitation under the policies used for reference construction. The panels are intended to compare the spatial organization and support of the learned densities with their numerical references rather than to provide an analytical ground-truth error measure.

Both methods recover structured state-density fields over the regions visited by their learned policies. In the displayed example (Figure 6), PINN-URL reproduces the large-scale organization of its simulation reference more closely than Original URL. This qualitative comparison matches the lower StandUp density residual of PINN-URL, while the common residual bank provides the quantitative method-level comparison.

E.3 StandUp Value Fields



(b) Value

Figure 7: Representative StandUp value fields for Original URL and PINN-URL, together with their policy-dependent Monte Carlo references. For each method, the learned field is shown alongside its corresponding reference field. The Monte Carlo estimator uses the Umbrella reward and the learned density associated with the corresponding reference construction. Learned and reference fields are shown on a common scale within each method-wise comparison. These panels are qualitative diagnostics rather than analytical ground-truth value errors.

Original URL shows larger localized departures from its Monte Carlo reference, whereas the displayed PINN-URL field reproduces the principal high- and low-value regions and their large-scale gradients more closely (Figure 7). This matches the final median StandUp value-residual MSE, which decreases from 5.049×10^{-3} for Original URL to 8.061×10^{-5} for PINN-URL. The field

figures are representative spatial diagnostics; the residual study provides the across-run quantitative comparison on a common state bank.

F Ablation Details

The entropy and architecture–learning sensitivity studies each use ten independent training runs per configuration, with seeds $0, \ldots, 9$, and 1.2×10^6 training iterations. Reported dispersions are sample standard deviations across training runs. The entropy study changes the composition of the Umbrella entropy term, whereas the architecture–learning study exchanges the value/density representation and learning configuration.

F.1 Entropy Terms

The full URL effective reward is

$$r_U(s,a) = r(s,a) - \widetilde{\alpha}\left[\log p(s) + \log \pi(a \mid s)\right], \qquad \widetilde{\alpha} = 0.01.$$

Entropy objectives over state-visitation distributions have also been studied as exploration criteria in reinforcement learning (Hazan et al., 2019); URL differs by coupling the state-density and policy-entropy contributions directly in the Umbrella reward.

The entropy study compares three configurations:

$$\begin{aligned} \text{Full Entropy:} &\quad r_U = r - \widetilde{\alpha}(\log p + \log \pi), \\ \text{Policy only :} &\quad r_U = r - \widetilde{\alpha} \log \pi, \\ \text{No Entropy:} &\quad r_U = r. \end{aligned}$$

Policy only removes only the state-density term from the effective reward; density learning itself remains active. No Entropy sets $\widetilde{\alpha} = 0$. Table 2 gives the final deterministic-return statistics for all method–environment combinations.

Figure 8 shows the MVMC return trajectories. Both methods exhibit the same ordering: Full Entropy $>$ Policy only $>$ No Entropy. Removing only the density contribution reduces return but preserves successful learning, whereas setting $\widetilde{\alpha} = 0$ yields zero final return in every run of both methods. The comparison between Policy only and No Entropy therefore supports an important role for the policy-entropy contribution under the tested MVMC configuration, while the improvement from Policy only to Full Entropy shows an additional contribution from the density term. Because the study does not include a density-only entropy condition, these comparisons do not constitute a complete factorial decomposition of the two entropy terms. StandUp shows a different pattern. Figure 9 reports the complete return trajectories for Original URL and PINN-URL.

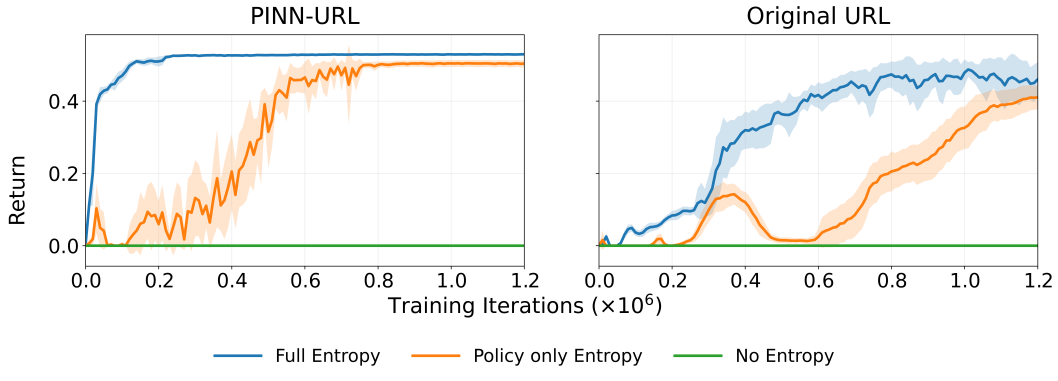


Figure 8: Return in MVMC for three entropy configurations of the Umbrella reward, for PINN-URL (left) and Original URL (right): the full reward, a policy-only Curves show the mean across 10 independent training runs per condition; shaded regions indicate pointwise 95% confidence intervals for the mean.

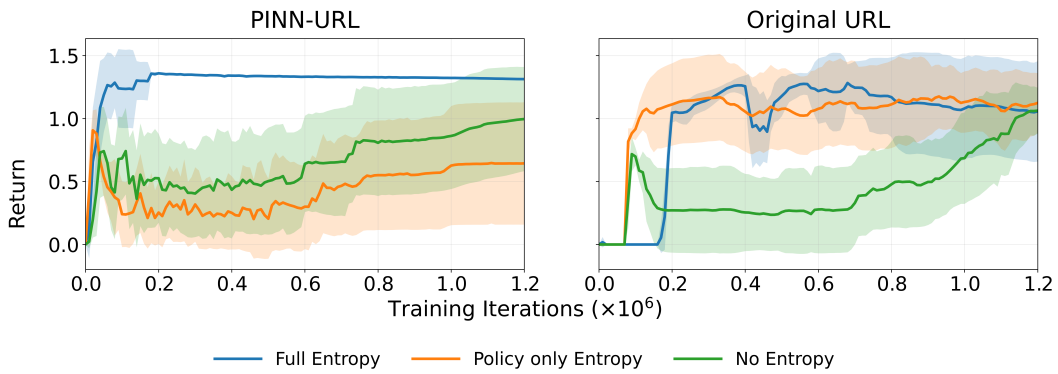


Figure 9: StandUp entropy-composition sensitivity. Curves show mean deterministic returns across ten independent training runs; shaded regions indicate pointwise 95% confidence intervals for the mean. Full Entropy retains the complete joint state–action entropy term; Policy only removes the state-density contribution from the effective reward while retaining density learning; No Entropy sets $\widetilde{\alpha} = 0$.

For Original URL on StandUp, the three final means are similar relative to their substantial run-to-run dispersion, and there is no monotone advantage of the full joint entropy formulation. PINN-URL behaves differently: Full Entropy reaches 1.313 ± 0.004, whereas Policy only and No Entropy end at 0.644 ± 0.486 and 0.996 ± 0.413, respectively. All uncertainties reported here are 95% confidence interval half-widths for the mean across ten independent training runs. The full entropy formulation also exhibits substantially smaller across-seed variability for PINN-URL in StandUp under the tested configuration. The effect of entropy is therefore method- and benchmark-dependent. MVMC favors the full joint entropy term for both learning procedures. In StandUp, Original URL shows no corresponding monotone ordering, whereas full-entropy PINN-URL is markedly more consistent than either reduced-entropy variant. We therefore treat the entropy formulation as part of the integrated method configuration rather than attributing the full PINN-URL result to either entropy contribution in isolation.

F.2 Swapped Architectures

The architecture–learning study crosses the value/density representation with the corresponding learning configuration while leaving the policy architecture in the same two-hidden-layer Tanh family used in the core methods. The two tested combinations are

PINN architecture + Original learning:	SIREN value/density representation with Original URL semi-gradient learning,
Original architecture + PINN learning:	Original ELU value/density representation with PINN-URL residual-based learning.

The first combination uses the Original URL semi-gradient value and density updates, one sampled action per state, fixed-rate Adam optimization, and no PINN value-focused sampling, cosine scheduling, or gradient clipping. The second uses exact finite-action value and density residuals, shared collocation sampling, additional value-focused states, AdamW/Adam optimization, cosine schedules, and branch-specific clipping. Consequently, the experiment crosses two *bundles*; it is not a factorial isolation of activation function, network depth, residual loss, sampling strategy, or optimizer individually. Figure 10 shows the complete deterministic-return trajectories, and Table 6 summarizes their final and best group-mean performance.

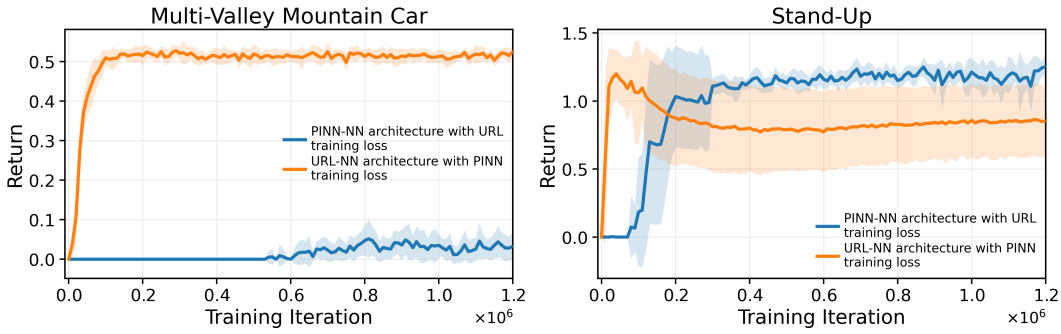


Figure 10: Return for the swapped value and density architectures in MVMC (left) and StandUp (right): SIREN networks trained as in Original URL, and the ELU networks of Original URL trained as in PINN-URL. Curves show the mean across 10 runs per condition; shaded regions denote one standard deviation.

Table 6: Architecture–learning swap return summary. Entries are means ± sample standard deviations across ten independent training runs. "Best" denotes the checkpoint with the highest across-run mean return, not the mean of independently selected per-run maxima.

Environment	Swap	Final return	Best group mean
MVMC	PINN architecture + Original learning	0.032 ± 0.034	0.052 ± 0.048 at 0.81M
MVMC	Original architecture + PINN learning	0.524 ± 0.013	0.527 ± 0.013 at 0.29M
StandUp	PINN architecture + Original learning	1.253 ± 0.094	1.253 ± 0.094 at 1.20M
StandUp	Original architecture + PINN learning	0.854 ± 0.247	1.200 ± 0.179 at 0.04M

The MVMC ordering strongly favors the Original architecture combined with the PINN-URL learning configuration. The reverse tendency appears in StandUp: the PINN architecture with Original URL learning remains strong at the endpoint, whereas the Original architecture with PINN-URL learning rises rapidly to a high-return regime and then deteriorates substantially over the remainder of training. The reversal shows that neither the field architecture nor the learning configuration has a benchmark-independent advantage when considered as these tested bundles.

F.3 Swap Configurations

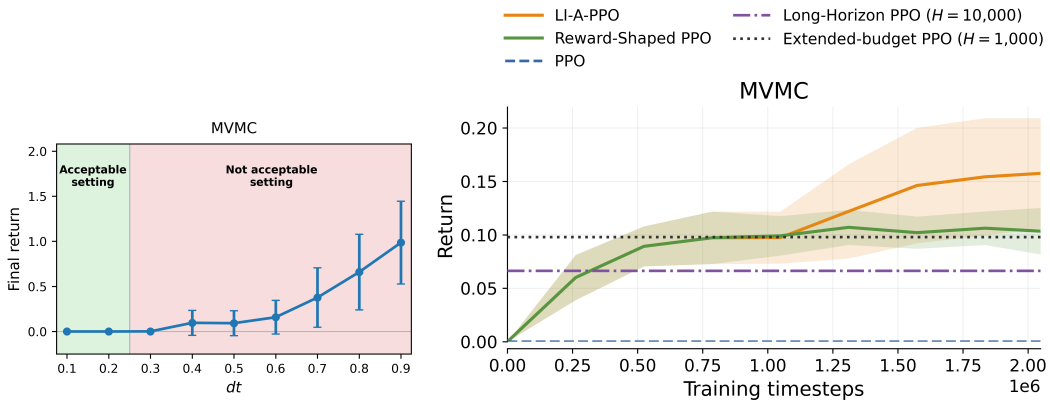
All four cells use seeds 0–9, 1.2×10^6 training iterations, a main batch size of 10,000, $\gamma = 0.95$, $\widetilde{\alpha} = 0.01$, and the reflecting representation. Table 7 gives the configuration-specific optimizer settings. The ten-run comparison used these selected configurations without further tuning on seeds 0–9. For the Original-architecture + PINN-learning MVMC cell, the additional value states cover the MVMC reward strip and use the same local weighting form as the PINN-URL value-focused residual, but the final swap configuration uses 10,000 extra states and $\lambda_{\text{extra}} = 0.3$. For the corresponding StandUp cell, 10,000 additional admissible goal-region states are used with $\lambda_{\text{extra}} = 1$. These sampling settings differ from the core PINN-URL configuration: the swaps with PINN learning use 10,000 additional value states, whereas the core PINN-URL experiments use 2,500. The PINN-architecture + Original-learning MVMC configuration underwent more learning-rate tuning during development than its StandUp counterpart; the StandUp configuration largely inherited the corresponding Original URL optimization settings. This asymmetry is another reason to interpret the study as a comparison of the final tested bundles rather than as a clean causal decomposition. The swaps show a benchmark-dependent dependence on both field representation and learning configuration. They do not isolate the effect of architecture or residual-based learning individually. Such attribution would require experiments in which architecture, objective, sampling, and optimization are varied independently.

Table 7: Key optimization settings for the final architecture–learning swaps.

Swap	Env.	Initial learning rates	Additional settings
PINN arch. + Original learning	MVMC	$\eta_\pi = 1.25\times10^{-6}$ $\eta_V = 1.5\times10^{-6}$ $\eta_p = 2\times10^{-5}$	Adam; fixed learning rates; no gradient clipping; no extra-value batch
PINN arch. + Original learning	StandUp	$\eta_\pi = 1\times10^{-6}$ $\eta_V = 1\times10^{-6}$ $\eta_p = 1\times10^{-7}$	Adam; fixed learning rates; no gradient clipping; no extra-value batch
Original arch. + PINN learning	MVMC	$\eta_\pi = 1\times10^{-4}$ $\eta_V = 6\times10^{-4}$ $\eta_p = 2.5\times10^{-4}$	Cosine minima: $\eta_{\pi,\min} = 1\times10^{-5}$, $\eta_{V,\min} = 1\times10^{-5}$, $\eta_{p,\min} = 5\times10^{-6}$. Clips $(\pi, V, p) = (0, 5, 5)$. 10,000 extra value states; $\lambda_{\text{extra}} = 0.3$
Original arch. + PINN learning	StandUp	$\eta_\pi = 3\times10^{-5}$ $\eta_V = 5\times10^{-4}$ $\eta_p = 2.5\times10^{-4}$	Cosine minima: $\eta_{\pi,\min} = 3\times10^{-6}$, $\eta_{V,\min} = 5\times10^{-5}$, $\eta_{p,\min} = 5\times10^{-6}$. Clips $(\pi, V, p) = (0.5, 1, 1)$. 10,000 extra value states; $\lambda_{\text{extra}} = 1$

G Relaxed-Constraint Baselines

Proximal Policy Optimization (PPO) was evaluated at the target benchmark setting and under additional discretization, horizon, and training-signal variants. The vanilla baseline uses Stable-Baselines3 PPO (Raffin et al., 2021), trained from random initialization on the original sparse-reward environments. Reward-Shaped PPO and Local-Imitation-Assisted PPO (LI-A-PPO) are included as separate MVMC diagnostics and are not interpreted as single-factor modifications of the vanilla baseline. Figure 11 summarizes the MVMC outcomes across these variants and across the time-step sweep; the individual studies follow.



(a) Temporal discretization **(b)** Return trajectories at $\Delta t = 0.1$

Figure 11: PPO diagnostics on MVMC. (a) Final deterministic discounted return after 2.048M interactions in the fixed-step-discount sweep. Points and error bars show the mean and 95% confidence intervals across 20 runs; shading marks the target regime, $\Delta t \leq 0.2$. (b) Deterministic return trajectories at $\Delta t = 0.1$ for Reward-Shaped PPO and LI-A-PPO, with shaded bands showing mean $\pm$ sample standard deviation across 20 runs. Horizontal lines indicate the final mean returns of PPO ($H = 1000$, 2.048M interactions), Long-Horizon PPO ($H = 10{,}000$, 6.016M interactions), and Extended-budget PPO ($H = 1000$, 6.016M interactions). Reward-Shaped PPO and LI-A-PPO are shown through their 2.048M-interaction training budgets. The horizontal references represent results obtained with different training budgets; panel (b) is not an equal-budget comparison.

Table 8: Vanilla PPO configurations at the target benchmark setting. Both actor and critic use the indicated two-hidden-layer architecture.

Setting	MVMC	StandUp
Training interactions	2,048,000	2,048,000
Episode horizon	1000	2000
Parallel environments	4	4
Steps per environment / rollout	1024	1000
Rollout size	4096	4000
Minibatch size	256	500
PPO epochs	10	10
Learning rate	3×10^{-4}	3×10^{-4}
GAE λ	0.95	0.95
PPO clip range	0.2	0.2
Entropy coefficient	0	0.01
Value-loss coefficient	0.5	0.5
Maximum gradient norm	0.5	0.5
Actor / critic hidden layers	Tanh $[64, 64]$	ReLU $[128, 128]$

G.1 PPO AT THE BENCHMARK SETTING

The target benchmark discretization is

$$\Delta t = 0.1,$$

with episode horizons

$$H_{\mathrm{MVMC}} = 1000, \qquad H_{\mathrm{StandUp}} = 2000.$$

The environments do not terminate when the rewarded region is reached; episodes are truncated only by these external horizons. PPO uses the per-step discount

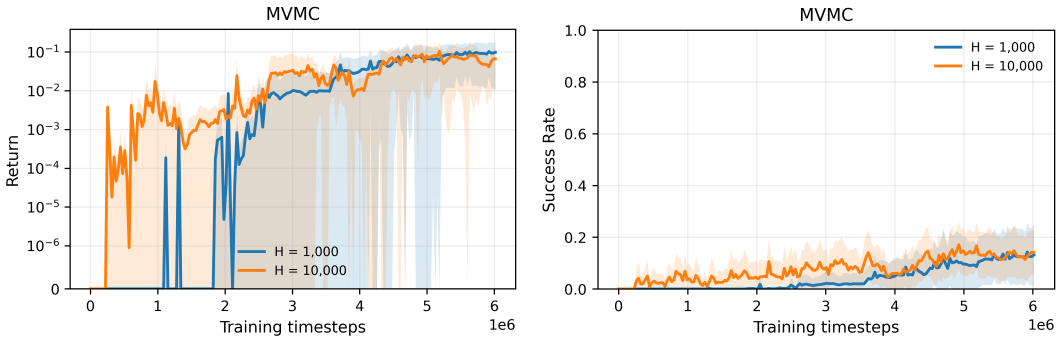
$$\gamma_{\mathrm{step}} = \gamma_{\mathrm{cont}}^{\Delta t}, \qquad \gamma_{\mathrm{cont}} = 0.95,$$

so at the target discretization

$$\gamma_{\mathrm{step}} = 0.95^{0.1} \simeq 0.994884.$$

This matches the continuous-time discount convention used when evaluating the benchmark rather than applying 0.95 independently at every numerical environment step. The final vanilla PPO baselines contain 20 independent training runs, with seeds $0, \ldots, 19$. Their selected configurations are summarized in Table 8. Advantage estimates use generalized advantage estimation (GAE) (Schulman et al., 2016).

Systematic PPO tuning. The near-zero PPO result at $\Delta t = 0.1$ was not based on a single default configuration. Before selecting the final multi-run baselines, we performed a broad vanilla-PPO screen that varied learning rate, entropy regularization, network activation and width, rollout length, vector-environment count, clipping/update conservatism, and GAE settings while keeping the original reward, dynamics, Δt, initial-state distribution, and episode horizon unchanged. The first stage contained 12 configurations and two development seeds for each environment, giving 24 MVMC and 24 StandUp runs of 1.024×10^6 interactions each. None of the 24 MVMC runs encountered a positive sparse reward during training. StandUp produced partial learning signals and was therefore investigated more deeply. A second stage evaluated 12 focused configurations on four new development seeds, with 2.048×10^6 interactions per run. Promising policies were checked on independent initial-state banks, followed by a reserved-seed confirmation stage before the final configuration was selected. The final vanilla PPO configurations were therefore selected after systematic environment-specific tuning rather than taken directly from the Stable-Baselines3 defaults. For the reported figures, saved checkpoints are reevaluated on 250 episodes using a fresh evaluation bank shared across directly compared policies. Deterministic actions define the primary return curves; stochastic-policy evaluations are retained as diagnostics. Aggregation is performed across independently trained policies, not across evaluation episodes.



(a) Discounted return **(b)** Goal-reaching success

Figure 12: Deterministic-policy MVMC horizon diagnostics for the paired $H = 1000$ and $H = 10000$ PPO experiments with extended budget. Curves show arithmetic means across 20 training runs, with shaded bands denoting 95% confidence intervals of the mean. Confidence intervals are clipped to the valid range of each metric. The horizon intervention is paired; all other training settings are matched.

G.2 Extended Budget and Horizon

Additional PPO runs examine the effects of a larger interaction budget and episode horizon. These experiments are diagnostic studies and are not equal-budget comparisons with URL or PINN-URL. For these horizon diagnostics, the goal-reaching success rate is the fraction of evaluation episodes in which the policy enters the rewarded region at least once within the evaluation horizon. For MVMC, a final paired horizon study uses the same tuned PPO configuration for

$$H = 1000 \qquad \text{and} \qquad H = 10000.$$

The two experiments each contain 20 training runs and 6.016×10^6 environment interactions. Apart from the training and evaluation horizon, their configuration is matched:

$$n_{\text{env}} = 8, \qquad n_{\text{steps}} = 4000, \qquad |\mathcal{B}| = 256, \qquad N_{\text{epoch}} = 20,$$
$$\eta = 3 \times 10^{-4}, \qquad \gamma_{\text{cont}} = 0.975, \qquad \lambda_{\text{GAE}} = 0.995,$$
$$\epsilon_{\text{clip}} = 0.2, \qquad c_{\text{ent}} = 0, \qquad c_V = 0.5, \qquad \|\nabla\|_{\max} = 0.25.$$

Both actor and critic use Tanh $[64, 64]$ networks. The two MVMC studies share the same 189 rollout-aligned evaluation checkpoints and common evaluation-bank construction.

As Figure 12 shows, the extended MVMC runs attain sustained nonzero return under a substantially larger interaction budget and the tuned horizon-study configuration. These experiments therefore rule out the stronger interpretation that PPO is intrinsically unable to learn the task. Because the interaction budget and training configuration differ from the vanilla baseline, however, they are treated as separate diagnostics rather than equal-budget replacements for the target-discretization comparison. StandUp was also evaluated with a final $H = 10000$ characterization using 20 runs and 6.0×10^6 environment interactions. Its configuration uses four parallel environments, 1000 steps per environment, minibatches of 500, 10 PPO epochs, learning rate 3×10^{-4}, $\gamma_{\text{cont}} = 0.95$, GAE $\lambda = 0.95$, clipping 0.2, entropy coefficient 0.02, and ReLU $[128, 128]$ actor and critic; its return and success rate training curves are shown in Figure 13.

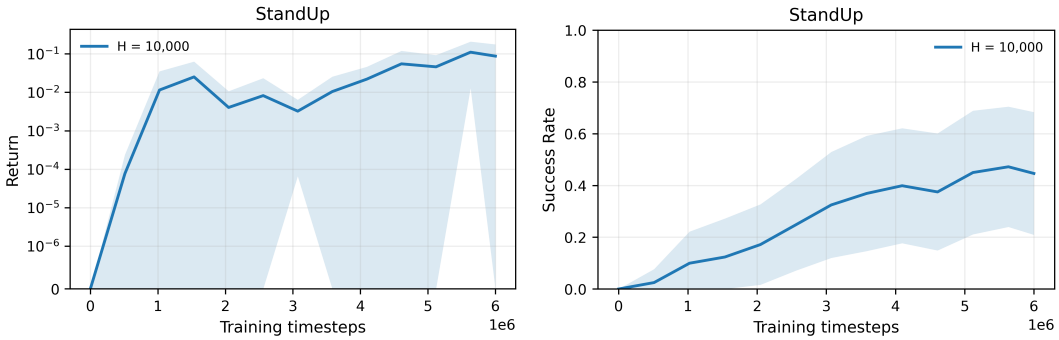
G.3 Reward Shaping

Reward-Shaped PPO is an MVMC alternative in which PPO is trained with a potential-based dense reward while evaluation continues to use the original sparse task reward. Let

$$E(x, u) = \frac{1}{2}u^2 + g\,h(x)$$

denote mechanical energy. The normalized energy-progress term is

$$e(x, u) = \text{clip}\left(\frac{E(x, u) - E_{\min}}{E_{\text{goal}} - E_{\min}}, 0, 1\right),$$



(a) Discounted return **(b)** Goal-reaching success

Figure 13: Deterministic-policy StandUp $H = 10000$ characterization across 20 independent training runs. Curves show arithmetic means, with shaded bands denoting 95% confidence intervals of the mean. Confidence intervals are clipped to the valid range of each metric. This is a supplementary long-horizon diagnostic; unlike the MVMC experiment above, it is not presented as a matched $H = 2000$ versus $H = 10000$ intervention.

with the implemented normalization constants

$$E_{\min} = -4.96844390051971 \times 10^{-4}, \qquad E_{\text{goal}} = 6.428984088157917 \times 10^{-4}.$$

The position-closeness term is

$$c(x) = 1 - \frac{\max(|x| - 0.05, 0)}{0.99 - 0.05},$$

with the implemented potential

$$\Phi(x, u) = e(x, u)\left[\alpha + (1 - \alpha)c(x)\right], \qquad \alpha = 0.75.$$

The PPO training reward is

$$r_{\text{train}}(s_t, a_t, s_{t+1}) = r_{\text{sparse}}(s_{t+1}) + \beta\left[\gamma_{\text{step}}\Phi(s_{t+1}) - \Phi(s_t)\right], \qquad \beta = 10.$$

The shaping wrapper does not modify the environment transition, initial-state distribution, action space, or episode horizon. The 20-run experiment uses $\Delta t = 0.1$, $H = 1000$, 2.048×10^6 interactions, four environments with 1024 steps each, minibatches of 256, 10 PPO epochs, learning rate 3×10^{-4}, Tanh $[64, 64]$ actor and critic, and entropy coefficient 0.01. Evaluation uses the original sparse MVMC reward; the shaped reward is used only during training.

G.4 Local Imitation

Local-Imitation-Assisted PPO (LI-A-PPO) uses the Reward-Shaped PPO configuration above throughout training and activates an additional actor-side objective after a rollout-aligned switch at

$$T_{\text{switch}} = 786{,}432$$

environment interactions. The shaped reward is unchanged across this switch. Training before T_{switch} is identical to Reward-Shaped PPO. At the switch, the current policy is copied and held fixed as the reference policy π_{ref}. Subsequent PPO updates retain the standard clipped policy loss, value loss, and entropy term, while adding local imitation and reference-policy regularization to the actor objective. For rollout observations in the local region

$$|x| < 0.15,$$

a physics-based controller evaluates the two one-step actions and returns a preferred action

$$a^\star(s) = \arg\min_{a \in \{0,1\}} J_a(s).$$

Its confidence is

$$\kappa(s) = \left|\tanh\left(\frac{J_1(s) - J_0(s)}{0.05}\right)\right|.$$

The local imitation loss over a PPO minibatch of size B is

$$\mathcal{L}_{\text{im}} = \frac{1}{B}\sum_{i=1}^{B} \mathbf{1}\{|x_i| < 0.15\}\kappa(s_i)\left[-\log \pi_\theta\big(a^\star(s_i) \mid s_i\big)\right].$$

Outside the local region, the policy is regularized toward the policy frozen at the switch:

$$\mathcal{L}_{\text{KL}} = \frac{1}{B}\sum_{i=1}^{B} \mathbf{1}\{|x_i| \geq 0.15\} D_{\text{KL}}\left(\pi_{\text{ref}}(\cdot \mid s_i) \,\|\, \pi_\theta(\cdot \mid s_i)\right).$$

The post-switch optimization objective is therefore

$$\mathcal{L}_{\text{LI-A}} = \mathcal{L}_{\text{PPO}} + \lambda_{\text{im}}\mathcal{L}_{\text{im}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}},$$

with

$$\lambda_{\text{im}} = 0.10, \qquad \lambda_{\text{KL}} = 5.0.$$

The controller supplies only an auxiliary training target. Its action is never executed in the environment, either during training or evaluation. LI-A-PPO uses no offline demonstration dataset, pretrained checkpoint, or behavior-cloning warm start. The PPO value loss is unchanged, and the controller does not modify the environment dynamics or the sparse evaluation reward. The final experiment contains 20 training runs. For every seed, the saved LI-A-PPO checkpoint at T_{switch} was verified to match the paired Reward-Shaped PPO checkpoint exactly in policy and optimizer state, and the frozen reference-policy snapshot was verified to reproduce the same switch policy. Thus, the divergence between the two methods begins only after the prescribed assistance switch.

G.5 Time Step

To examine sensitivity to the temporal discretization, vanilla PPO was trained at

$$\Delta t \in \{0.1, 0.2, \ldots, 0.9\}.$$

Each value of Δt uses 20 independent training runs. In the continuous-discount sweep, the continuous discount is kept at $\gamma_{\text{cont}} = 0.95$, so that

$$\gamma_{\text{step}}(\Delta t) = 0.95^{\Delta t}.$$

A separate fixed-step-discount sweep was also performed, holding the per-step PPO discount at its $\Delta t = 0.1$ value. This auxiliary sweep distinguishes effects of changing the numerical dynamics from those induced only through the corresponding change in per-step discounting. The benchmark remains $\Delta t = 0.1$. The labels in Figure 14 distinguish the finer $\Delta t \leq 0.2$ regime from coarser discretizations only for visual orientation. Changing Δt changes the discrete-time control problem, so performance at larger time steps is not evidence of performance on the target formulation.

G.6 Learned PPO Value Functions

Figure 15 shows representative MVMC critic fields for the four PPO variants considered in the diagnostic study. These critic fields are diagnostic only. PPO and Long-Horizon PPO estimate returns under the original sparse reward, whereas Reward-Shaped PPO and LI-A-PPO estimate returns under the shaped training reward. Their absolute value scales are therefore not directly comparable across the two groups.

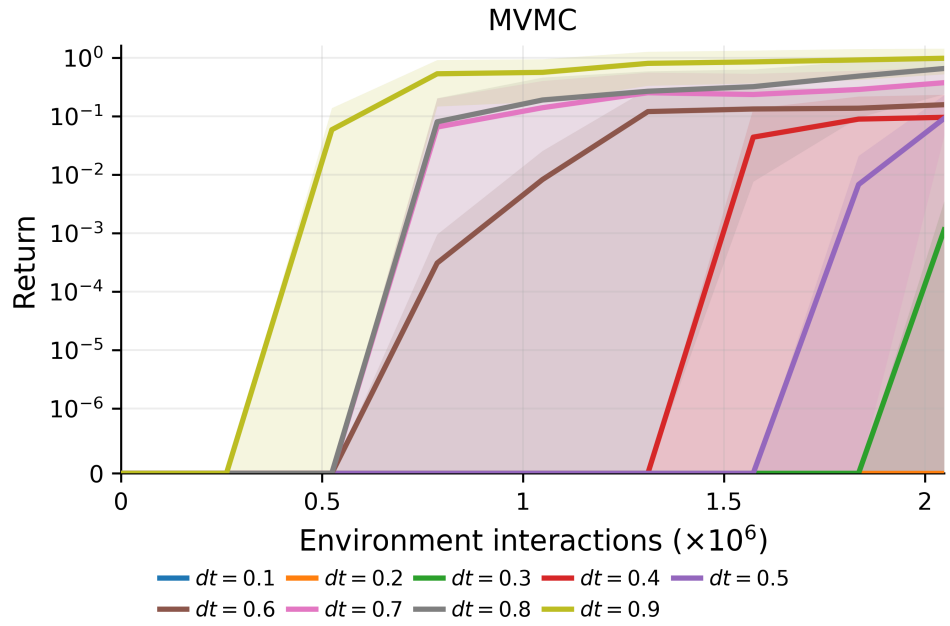


Figure 14: PPO return behavior across numerical time steps. The target benchmark uses $\Delta t = 0.1$; larger time steps modify the numerical control problem and are therefore treated as sensitivity diagnostics rather than equivalent versions of the target task. Curves show arithmetic means across 20 independently trained policies without smoothing or interpolation. The shaded bands denoting 95% confidence intervals of the mean.

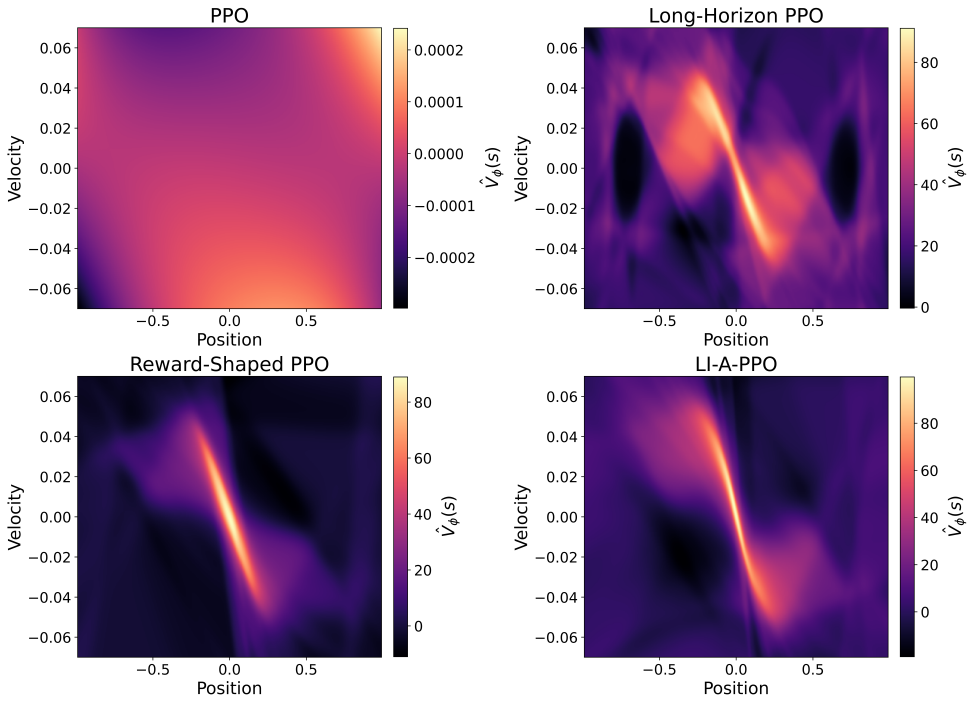


Figure 15: Representative MVMC critic fields for PPO, Long-Horizon PPO, Reward-Shaped PPO, and LI-A-PPO. Absolute value scales should not be compared directly between sparse-reward PPO and the reward-shaped variants because their critics are trained against different reward signals.