WHEN STORED ACTIONS LOSE INFORMATION: FINITE-PRECISION LIKELIHOOD REPLAY IN SQUASHED PPO

Anonymous authors
Paper under double-blind review

ABSTRACT

A stored action is not always a faithful record of the sample that produced it. In a tanh-squashed policy, several large Gaussian samples can become the same float32 action, exactly -1 or $+1$. Applying the inverse transform cannot recover the original sample. We study a Stable-Baselines3 PPO pathway that nevertheless reconstructs this value when evaluating action likelihoods. Its likelihood ratio initially equals one because collection and optimisation use the same reconstruction, but its gradient already differs from the gradient at the sampled value. Direct logging detects boundary collisions within the first eight environment interactions. Across three MuJoCo tasks, storing the sampled pre-tanh value substantially reduces peak approximate KL despite equal or greater boundary incidence. On HalfCheetah, the median falls from 1.26×10^6 to 0.0361. A separate PPO implementation reproduces the difference under a shared saturation stressor. We also show how a measured transform threshold predicts boundary exposure and how exploration initialisation changes that exposure. The practical distinction is between avoiding information loss and preserving the information needed for replay. Store the sampled latent when later updates evaluate its likelihood. When only a rounded action remains, model the probability of its rounding cell rather than treating a clamped inverse as the original sample.

1 INTRODUCTION

Proximal Policy Optimisation (PPO) collects actions with one policy and reuses those samples across several optimisation epochs (Schulman et al., 2017). Each update compares the likelihood of a collected sample under the current and collection policies. That comparison requires a consistent record of what was sampled. We examine a case in which the rollout buffer retains the action sent to the environment but loses information needed to reconstruct the sample used by the policy.

Consider a policy that samples a Gaussian value $u = 12$ and applies tanh to keep the action within $[-1, 1]$. On the float32 pathway studied here, the stored action is exactly 1.0. A later inverse calculation clamps that action and returns $\widehat{u} = 8.3178$, not 12. Every sufficiently large positive sample produces the same stored action. The clamp avoids an infinite inverse, but it cannot identify which sample was drawn.

This matters before an update becomes visibly unstable. In the studied Stable-Baselines3 (SB3) pathway, collection and optimisation both evaluate the likelihood at the reconstructed value. Their ratio therefore starts at one. Nevertheless, differentiating that ratio evaluates the likelihood gradient at $\widehat{u}$, not at u. The first update can already differ; subsequent updates can produce different ratios and clipping decisions. Checking that the initial ratio equals one does not detect this error.

We call repeated likelihood evaluation of a fixed on-policy rollout *likelihood replay*. This is PPO's reuse of its current rollout, not an off-policy replay buffer. We distinguish two questions throughout the paper. *Exposure* asks whether sampled actions enter a region where finite storage loses substantial information. *Replay representation* asks what information the buffer retains and what likelihood the update evaluates. Reducing exploration can limit exposure. Retaining the sampled value addresses the representation problem even when exposure occurs.

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.

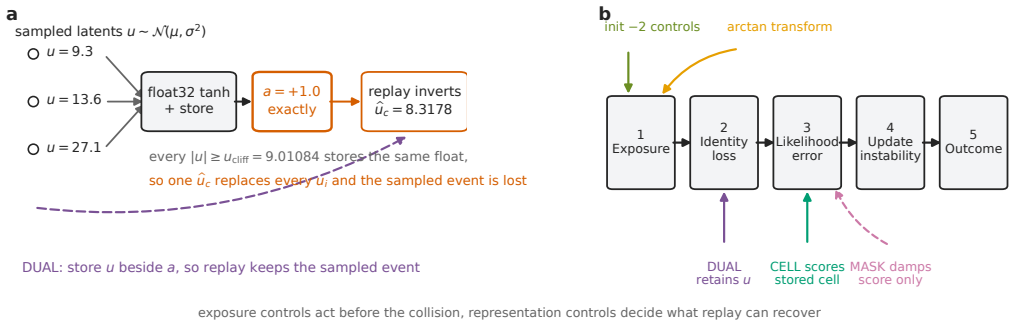


Figure 1: **Where replay loses the sampled value.** (a) Several pre-tanh values become the same stored action. DUAL retains the sampled value alongside that action; reconstructed replay replaces it with a clamped inverse. (b) The proposed sequence from boundary exposure to likelihood error and unstable updates, with the intervention points marked. Exposure does not make every run fail.

The main experiments use tanh squashing with generalised state-dependent exploration (gSDE) (Raffin et al., 2022). This is an optional SB3 configuration, not its default unsquashed PPO policy. We compare reconstructed replay with two alternatives: storing the sampled latent, and assigning probability to the set of latent values represented by the stored action. The comparisons use three MuJoCo tasks, a common source snapshot, and five seeds per method. A separate implementation tests the same representation change under a shared saturation stressor.

Our contribution is the diagnosis and measurement of a storage-induced likelihood error, not the invention of latent caching or integrated action probabilities. We derive the resulting gradient error, directly record when information is lost, and test interventions that separate exposure from replay representation. The results establish a numerical failure mechanism and a way to remove its reconstruction error. They do not establish a universal improvement in task return.

2 Background and Related Work

What PPO compares. For a state s_t, sampled action a_t, and advantage estimate $\widehat{A}_t$, PPO maximises

$$L^{\text{clip}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\widehat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\widehat{A}_t\right)\right], \qquad r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}. \tag{1}$$

The advantage estimates whether the action was better or worse than the baseline prediction; ϵ controls clipping. The old policy is fixed throughout the optimisation epochs. Implementations usually form the ratio by exponentiating a difference of log likelihoods. Clipping changes which surrogate term supplies the gradient. It does not check whether the two likelihoods describe the sample that was collected.

How a squashed policy generates an action. At a fixed state, write one Gaussian coordinate as $U \sim \mathcal{N}(\mu_\theta(s), \sigma_\theta(s)^2)$, with density $p_\theta(u \mid s)$. The pre-transform value U is the *latent*; the bounded action is $A = T(U)$, where $T(u) = \tanh(u)$. In exact arithmetic,

$$\log \pi_\theta(a \mid s) = \log p_\theta(u \mid s) - \log |T'(u)|, \qquad u = T^{-1}(a). \tag{2}$$

The second term accounts for the change of scale introduced by the transform. With independent action coordinates, log likelihoods are summed across coordinates. In gSDE, learned features and a sampled exploration matrix make the Gaussian scale state dependent. The initial log standard deviation sets the exploration-matrix scale, not the effective standard deviation at every state (Raffin et al., 2022; Stable-Baselines3 Contributors, 2025).

Relation to earlier work. PPO's sensitivity to implementation details and unstable likelihood ratios is established (Engstrom et al., 2020; Andrychowicz et al., 2021; Wang & Gao, 2025; Garg et al., 2021). Our question is earlier in the computation: does the retained action still identify the latent at which the likelihood is evaluated? SAC-style implementations already retain a latent within a forward

calculation (Haarnoja et al., 2018; Raffin et al., 2021). PPO additionally needs that value to survive storage and later optimisation epochs.

Integrated action probabilities are also established. CAPG integrates Gaussian tails for clipped actions (Fujita & Maeda, 2018); marginal policy gradients formalise the corresponding treatment of information-losing action transforms (Eisenach et al., 2019). Discretised likelihoods assign mass to bins rather than density to a representative point (Salimans et al., 2017). We apply these distinctions to cells created by floating-point storage and measure their consequences during PPO training.

3 How finite precision changes likelihood replay

3.1 The buffer stores a rounded action, not its latent

Let Q_p round an action to storage precision p. An idealised forward calculation stores

$$A_p = Q_p(T(U)). \tag{3}$$

Uppercase symbols denote random variables and lowercase symbols their realised values. The map $Q_p \circ T$ is many-to-one: a stored float represents a set of latent values. Near the boundary, that set can extend through an entire tail of the Gaussian.

Actual library kernels approximate T before storing its output. We therefore measure their saturation thresholds rather than deriving them from the dtype alone. On the audited PyTorch float32 CPU pathway, the first positive input that stores $+1$ is $\tau_\kappa = 9.01084$, where κ identifies the kernel and dtype. SB3 clamps the inverse input using machine epsilon, $\varepsilon_{32} = 2^{-23}$, giving

$$\widehat{u}_c = \operatorname{atanh}(1 - \varepsilon_{32}) \simeq 8.3178. \tag{4}$$

Even the shallowest positive boundary sample is reconstructed about 0.693 lower. The gap grows for deeper samples. Interior rounding also matters because the inverse sensitivity is $|d \operatorname{atanh}(a)/da| = (1 - a^2)^{-1}$.

3.2 An initial ratio of one does not imply a correct gradient

The relevant SB3 calls are short. During collection, `ActorCriticPolicy.forward` passes the squashed action to `log_prob`. With squashed gSDE, that method applies the clamped inverse. During optimisation, `evaluate_actions` passes the stored action through the same path. This behaviour is present in the versioned upstream source (Stable-Baselines3 Contributors, 2025); the experiments use an instrumented local version. Appendix B separates the source audit from the experimental modifications.

Write $\ell_\theta(v) = \log p_\theta(v \mid s)$ for the Gaussian log density. At fixed state and fixed evaluation points, the parameter-independent transform correction cancels from each old–new ratio. The two ratios are therefore

$$\widehat{r} = \exp\{\ell_\theta(\widehat{u}) - \ell_{\theta_{\text{old}}}(\widehat{u})\}, \qquad r_u = \exp\{\ell_\theta(u) - \ell_{\theta_{\text{old}}}(u)\}. \tag{5}$$

The first reconstructs a point; the second evaluates the retained latent. Both equal one at $\theta = \theta_{\text{old}}$.

Let $\delta = \widehat{u} - u$. Holding the samples and Gaussian scale fixed gives

$$\ell_\theta(\widehat{u}) - \ell_\theta(u) = -\frac{\delta\{2(u - \mu_\theta) + \delta\}}{2\sigma_\theta^2}, \tag{6}$$

$$\partial_{\mu_\theta} \log \widehat{r} - \partial_{\mu_\theta} \log r_u = \frac{\delta}{\sigma_\theta^2}. \tag{7}$$

A log-likelihood derivative is called a *score*. Equation (7) shows that the score error is present even when the ratio values agree. A small scale amplifies it. For a nonzero advantage, the first unclipped PPO gradient inherits this difference.

A numerical example makes the distinction concrete. Set $u = 12$, $\mu_{\text{old}} = 12$, and $\sigma = 1$. At the old policy, the sampled location score is 0, while the reconstructed score is approximately -3.6822. After a hypothetical mean change to 12.1, the sampled-latent ratio is 0.995, but the reconstructed ratio is 0.689. With PPO's $[0.8, 1.2]$ interval, the latter crosses the lower boundary. For a negative

advantage, the two evaluations select different surrogate branches. This is an analytical example, not an additional training result.

In general, defining $\Delta\ell_\theta = \ell_\theta(\widehat{u}) - \ell_\theta(u)$ yields

$$\log \widehat{r} - \log r_u = \Delta\ell_\theta - \Delta\ell_{\theta_{\text{old}}}. \tag{8}$$

Appendix A gives the branch conditions and derivation. A ratio guard can limit the update it receives. It cannot recover a sample that the buffer discarded.

3.3 Two alternatives represent different retained information

DUAL retains the sampled latent. DUAL stores both a_p and u, computes the collection likelihood at u, and reuses that same value during optimisation. It does not resample or reconstruct the latent. This removes the evaluation-point error in Equation (7). It preserves the sampled floating-point value, not an ideal real-number sample of infinite precision.

CELL assigns probability to the stored action's rounding cell. Under the idealised map in Equation (3), let a_p^- and a_p^+ be the rounding-cell edges, defined by midpoints to neighbouring representable actions. Their inverse images are $u_p^- = T^{-1}(a_p^-)$ and $u_p^+ = T^{-1}(a_p^+)$. Then

$$P_{\theta,p}^{\text{cell}}(a_p \mid s) = \Phi\left(\frac{u_p^+ - \mu_\theta(s)}{\sigma_\theta(s)}\right) - \Phi\left(\frac{u_p^- - \mu_\theta(s)}{\sigma_\theta(s)}\right), \tag{9}$$

where Φ is the standard normal distribution function. At $+1$, the upper latent edge is $+\infty$, so the mass is a Gaussian tail probability. CELL uses its logarithm in both terms of the PPO ratio.

This formula is exact for the specified rounding model. It is not an exact likelihood of every native transform kernel. For example, the ideal float32 positive-cell edge is $b = \operatorname{atanh}(1-2^{-25}) \simeq 9.01091$, slightly different from the measured PyTorch threshold τ_κ. We use b for the cell model and τ_κ for the measured exposure calculation.

The distinction changes the gradient. Define the normalised tail depth $z = (\mu - b)/\sigma \geq 0$ and the reconstruction gap $g = b - \widehat{u}_c > 0$. With φ denoting the standard normal density,

$$\partial_\mu \log P_{\theta,p}^{\text{cell}}(+1) = \frac{\varphi(z)}{\sigma\Phi(z)}, \qquad \partial_\mu \ell_\theta(\widehat{u}_c) = -\frac{z}{\sigma} - \frac{g}{\sigma^2}. \tag{10}$$

At fixed scale, the cell score tends to zero as the mean moves farther into the positive tail; the reconstructed score grows in magnitude. Their signs also differ. CELL can therefore avoid a large reconstructed score while leaving the action saturated. Stable updates and useful control are separate outcomes. Figure 2 illustrates the score comparison.

DUAL and CELL are not identical estimators. For a fixed transform and a weight that depends only on its recorded output, conditioning the latent score on that output gives the corresponding cell score. Their weighted expectations then agree before PPO clipping; their clipped, finite-sample updates need not agree. Appendix A states the assumptions. The identity does not establish an unbiased trajectory gradient for history-dependent exploration; our comparisons retain the same gSDE likelihood model and isolate the replay evaluation point.

4 Experimental design

Primary comparison. RECON is the reconstructed-replay parent: PPO with gSDE, coordinatewise tanh, noise resampling every four interactions, and initial exploration log standard deviation 0.0. CELL changes the likelihood to Equation (9); DUAL retains the sampled latent. We compare all three on HalfCheetah-v4, Walker2d-v5, and Ant-v4 with seeds $\{71, 0, 1, 2, 3\}$, giving 45 primary runs. They share one scientific source snapshot and training pipeline, with a requested budget of one million interactions per run. Standard PPO is an unsquashed performance reference, not a one-factor control for RECON.

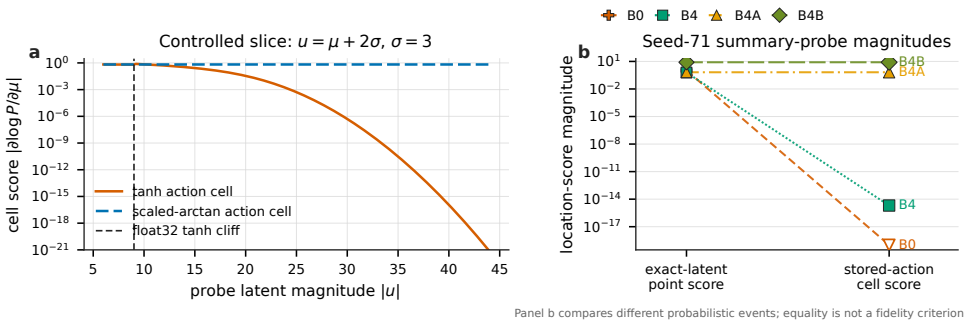


Figure 2: **Why the scored event matters.** (a) Controlled score comparison on the slice $u = \mu + 2\sigma$, $\sigma = 3$, for tanh and scaled arctan. (b) Sampled-latent and cell-score magnitudes at archived seed-71 summary probes. These probabilities describe different events, so equal scores are not the criterion for correctness. The probes combine final-quartile run summaries; they are not paired individual action samples. The B0 (RECON) cell score is a counterfactual calculation.

Direct measurements and independent reproduction. Five RECON and five CELL HalfCheetah runs log sampled and reconstructed latents before optimiser minibatches, together with exact stored-boundary status and likelihood diagnostics. Two shorter RECON runs, seeds 71 and 2, log collection events at individual interactions for a budget of 51,200. A telemetry-on/off check reports bitwise agreement in all 55 monitored fields under fixed source code.

The independent implementation shares no SB3 code. Its RECON and DUAL arms receive the same predetermined additive mean offset, increasing from zero to $+12.0$ over 200,000 interactions and then held fixed. The offset enters both sampling and likelihood evaluation. Three seeds per method and a one-million-interaction budget were specified before these runs. This comparison tests behaviour under controlled saturation, not the frequency of saturation in ordinary training.

Supporting controls. An earlier 110-run suite covers exploration initialisation, actor family, transform, and arithmetic precision. Its source snapshot differs from the primary comparison, so its outcomes are reported separately. RECON-INIT2 changes only the initial exploration log standard deviation to -2; CELL-INIT2 combines that initialisation with cell likelihoods. The value -2 is the base PPO setting in the gSDE paper, which also reports environment-specific overrides (Raffin et al., 2022). A separate five-seed MASK control scales reconstructed gradients without replacing their likelihood values. Appendix C summarises these controls.

Outcomes and aggregation. We report numerical failures, update diagnostics, and task returns separately. A failure is a non-finite training quantity or arithmetic exception. Failed runs have no endpoint return. The *boundary fraction* counts samples with at least one coordinate stored exactly at -1 or $+1$, not merely close to a boundary. Within-run fractions average equal-size logged minibatches; each caption specifies the aggregation across seeds.

We use SB3's approximate-KL diagnostic $\widehat{D} = \mathrm{mean}[(r-1) - \log r]$ and write $\widehat{D}_{\mathrm{peak}}$ for the largest finite recorded value in a run. Very large values indicate extreme ratios, not a trustworthy numerical estimate of the true policy KL. We show all five seeds for the primary diagnostic rather than using significance tests to summarise this small sample. Hyperparameters, software, and evaluation details appear in Appendix B.

5 Results

5.1 Information is lost during collection

The two fine-grained runs first store a boundary action at interactions 8 (seed 71) and 3 (seed 2), before any optimisation. At the first minibatch, after 2,048 interactions, both record latent and log-likelihood errors above 10^{-3}. Gradient norms first exceed 100 at interactions 8,192 and 6,144, respectively. Neither run reaches approximate KL above 10 or a non-finite value within the short

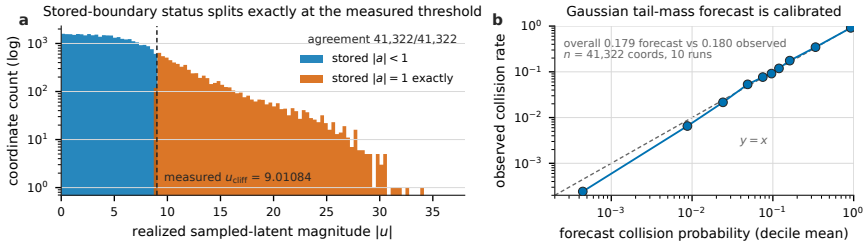


Figure 3: **Predicting boundary exposure.** (a) The measured PyTorch float32 threshold separates stored-boundary and interior coordinates in the captured sample. (b) Predicted and observed boundary fractions grouped by forecast decile. The lowest observed rate is displayed at the axis floor. The 41,322 coordinates are deduplicated, event-triggered observations from ten runs.

budget. These observations establish the order of the measured events, not a complete path to failure in those two runs.

Across the five longer RECON runs, the mean exact-boundary sample fraction is 0.579. The within-minibatch 95th percentile of absolute latent error, averaged within runs and then across seeds, is 4.14. CELL has a higher boundary fraction, 0.953, and an action-only reconstruction would have an average minibatch 95th-percentile error of 16.28. CELL does not use that reconstructed point for its likelihood.

The archived ratio telemetry also includes a *one-sided* diagnostic that changes the current evaluation point but retains the reconstructed old log likelihood. It is not r_u in Equation (5). We do not use its clipping or branch disagreements as evidence that two consistent same-policy ratios differ before an update. The representation claim rests on the gradient calculation and the RECON–DUAL interventions.

5.2 The measured threshold predicts captured boundary exposure

For a Gaussian coordinate and a symmetric saturation threshold τ_κ, define

$$h(\mu, \sigma; \tau_\kappa) = \Pr_{U \sim \mathcal{N}(\mu, \sigma^2)}(|U| \geq \tau_\kappa) = \Phi\left(\frac{-\tau_\kappa - \mu}{\sigma}\right) + \Phi\left(\frac{\mu - \tau_\kappa}{\sigma}\right). \tag{11}$$

This is the boundary probability under the measured threshold model. It requires neither inverse reconstruction nor a likelihood ratio.

On 41,322 deduplicated coordinates from the ten direct-logging runs, $|u| \geq \tau_\kappa$ agrees with stored $|a_p| = 1$ on every coordinate. The mean forecast is 0.179, compared with an observed fraction of 0.180; Figure 3 shows the binned comparison. The snapshots are event triggered. These values describe the captured coordinates, not an unconditional collision rate or an independent sample of entire training runs.

5.3 Retaining the latent reduces extreme updates across tasks

Table 1 compares the three replay representations. RECON has the largest median peak approximate KL on every task. The separation spans several orders of magnitude on HalfCheetah and Walker2d and is smaller on Ant. The two numerical failures in the 45-run comparison both occur under RECON. All 15 DUAL runs finish, with peak approximate KL below 0.2.

CELL and DUAL reach at least as much boundary incidence as RECON. Boundary incidence alone does not match the depth of the latent tails, but DUAL also records the reconstruction error that action-only replay would have incurred. On HalfCheetah, its median per-run maximum is 37.4. Those errors are measured but never used by DUAL, whose replay uses the retained latent throughout. Figure 4 shows the seed-level exposure and update diagnostics.

Returns do not follow the diagnostic ordering uniformly. DUAL improves the completed-run HalfCheetah mean over RECON, but CELL has the highest Walker2d mean and all three Ant means

Table 1: **Primary replay comparison: five seeds per method and task.** Boundary fraction and $\widehat{D}_{\text{peak}}$ are medians across seeds. Return is the mean deterministic endpoint among completed runs, with that sample size in parentheses. Failed runs contribute to diagnostics up to failure, but have no endpoint return.

Task	Method	Boundary fraction	$\widehat{D}_{\text{peak}}$	Failures	Return (n)
HalfCheetah	RECON	0.564	1.26×10^6	1/5	233.3 (4)
	CELL	0.953	0.0273	0/5	992.3 (5)
	DUAL	0.950	0.0361	0/5	1043.0 (5)
Walker2d	RECON	0.836	4.8×10^5	1/5	1187.7 (4)
	CELL	0.981	0.0780	0/5	2897.0 (5)
	DUAL	0.978	0.0943	0/5	1859.1 (5)
Ant	RECON	0.707	0.1775	0/5	-8.4 (5)
	CELL	0.949	0.0276	0/5	-15.2 (5)
	DUAL	0.955	0.0362	0/5	-13.4 (5)

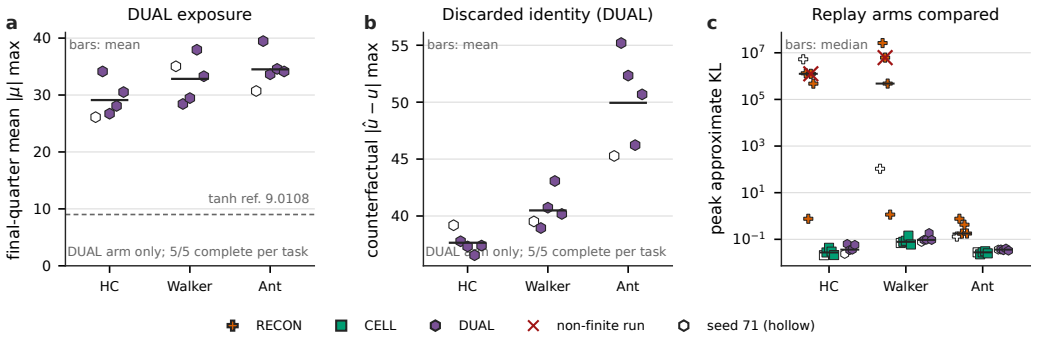


Figure 4: **Replay representation across three tasks.** Each point is one seed; hollow markers identify discovery seed 71. (a) DUAL boundary exposure and (b) the reconstruction error action-only replay would have incurred. These errors are diagnostics, not DUAL inputs; bars in these panels are five-seed means. (c) Peak approximate KL by method and task on a logarithmic scale, with median bars. Crosses identify the two numerical failures, both under RECON.

remain negative. The numerical result is therefore more consistent than the performance result: preserving the latent removes reconstruction error, not every cause of poor control.

5.4 A SEPARATE IMPLEMENTATION REPRODUCES THE DIFFERENCE

Under the common mean-offset stressor, all three independent RECON runs become non-finite, at interactions 475,136, 503,808, and 618,496. Their median peak approximate KL is 2.34×10^{30}. The maximum reconstruction-induced log-likelihood errors range from 1,409 to 2,855 nats. All three DUAL runs complete, with median peak approximate KL 4.75 and zero evaluation-point mismatch. Both methods reach full stored-boundary exposure. This reproduction supports the representation mechanism beyond the SB3-derived code, while remaining a deliberately stressed comparison.

5.5 PRECISION AND INITIALISATION CHANGE EXPOSURE, NOT THE STORED RECORD

The earlier precision comparison separates moving the saturation threshold from preserving the sample (Figure 5). Every tested precision arm crosses its own forward threshold. Wider arithmetic therefore delays, but does not eliminate, the observed exposure. The float32 latent-retention arm keeps approximate KL small after crossing.

Initialisation provides another exposure control. In the supporting suite, all 15 principal RECON runs and all 15 CELL runs cross the recorded policy-mean reference $|\mu| > 9.0108$. None of the 15 CELL-INIT2 runs or the five HalfCheetah RECON-INIT2 runs crosses that reference. This is

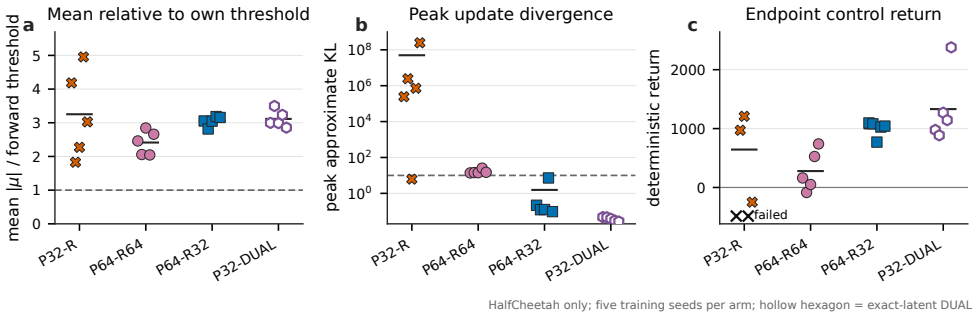


Figure 5: **More precision and more retained information are different interventions.** The earlier HalfCheetah comparison uses five seeds per arm. Each arithmetic arm crosses its own measured forward threshold. P32-DUAL retains the latent alongside the float32 action and keeps the update diagnostic small after crossing. Hollow hexagons identify DUAL; crosses mark failed P32-R (float32 reconstructed-replay) endpoints. This source snapshot is separate from the primary comparison in Table 1.

a *mean-based diagnostic*: staying below it does not imply that every Gaussian sample avoids the boundary. Table 3 reports the corresponding HalfCheetah outcomes. The improved returns under RECON-INIT2 also show why CELL-INIT2's performance cannot be attributed to cell likelihoods alone.

6 Practical implications and scope

Preserve the value that the later likelihood needs. Store the sampled pre-transform latent alongside the bounded action, and use it for both collection and optimisation likelihoods. Store it without reducing its precision and keep it fixed during gradient evaluation. For a parameter-independent transform, its Jacobian correction cancels from the PPO ratio. A correction required elsewhere should be evaluated from the retained latent, not from an inverted boundary action. This change adds one latent array; it does not require a different environment action.

Measure exposure on the deployed pathway. Monitor exact boundary equality in the buffer dtype, separately from near-boundary action rates and policy-mean thresholds. Measure the forward threshold for the actual kernel: our reported PyTorch and NumPy CPU thresholds differ at the same dtype (Table 2). Wider precision and lower initial exploration are useful exposure controls, not guarantees that an action-only buffer identifies the sampled latent. When only the stored action remains, a cell likelihood requires an explicit rounding or kernel model.

What the evidence establishes. The analysis identifies an evaluation-point error. The primary comparisons show substantially smaller extreme-update diagnostics after removing that error, including runs with substantial boundary exposure. Neither the five-seed comparisons nor the three-seed stressed reproduction estimates how prevalent the problem is across applications. Fine-grained event order uses two seeds, and seed 71 was the discovery seed. The thresholds were measured on CPU; no CUDA threshold is claimed.

The study leaves temporal gSDE likelihood modelling unchanged. It also does not establish identical clipped objectives for CELL and DUAL, an exact kernel likelihood for the ideal midpoint-cell model, or better return on every task. These boundaries separate the demonstrated storage mechanism from broader questions about exploration and policy optimisation.

7 Conclusion

A squashed action stored in finite precision need not identify the latent that produced it. Clamping the inverse yields a finite number, not a recovered sample. When both PPO likelihoods use that reconstruction, their initial ratio is one but the first gradient can already be wrong for the sampled

latent. Direct measurements locate the information loss during collection, and representation-level comparisons reduce extreme updates without reducing boundary incidence. Initialisation and precision change exposure; retaining the latent preserves the information needed for replay. The likelihood used in an update should match what the rollout actually records.

AI USE STATEMENT

AI agents were used to generate initial drafts of some sections, improve the clarity of the writing, and assist with the development of mathematical arguments and proofs. The authors reviewed and revised the AI-assisted drafts and rechecked the mathematical claims and supporting arguments. OpenAI Codex supported these activities, while Google Gemini assisted with a final manuscript review. The authors take full responsibility for the final content.

REPRODUCIBILITY STATEMENT

Appendices A–C give the mathematical assumptions, implementation changes, training settings, metric definitions, and supporting controls. The supplementary materials document resolved configurations, commands, source hashes, run outcomes, saved telemetry, figure data, and validation scripts. Primary and earlier-source experiments are identified separately.

REFERENCES

Marcin Andrychowicz et al. What matters in on-policy reinforcement learning? a large-scale empirical study. *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=nIAxjsniDzg.

Carson Eisenach, Haichuan Yang, Ji Liu, and Han Liu. Marginal policy gradients: A unified family of estimators for bounded action spaces with applications. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkgqFiAcFm.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1etN1rtPB.

Yasuhiro Fujita and Shin-ichi Maeda. Clipped action policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1597–1606, 2018. URL https://proceedings.mlr.press/v80/fujita18a.html.

Saurabh Garg, Joshua Zhanson, Emilio Parisotto, Adarsh Prasad, Zico Kolter, Zachary Lipton, Sivaraman Balakrishnan, Ruslan Salakhutdinov, and Pradeep Ravikumar. On proximal policy optimization's heavy-tailed gradients. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3610–3619, 2021. URL https://proceedings.mlr.press/v139/garg21b.html.

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.

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.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth exploration for robotic reinforcement learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1634–1644, 2022. URL https://proceedings.mlr.press/v164/raffin22a.html.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. PixelCNN++: Improving the PixelCNN with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017. URL https://arxiv.org/abs/1701.05517.

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.

Stable-Baselines3 Contributors. Stable-Baselines3, version 2.7.1: Policy and distribution source code. Versioned source code, 2025. URL https://github.com/DLR-RM/stable-baselines3/tree/v2.7.1/stable_baselines3/common. Tag v2.7.1; stable_baselines3/common/policies.py and stable_baselines3/common/distributions.py. Accessed 24 September 2026.

Tao Wang and Sicun Gao. Numerical pitfalls in policy gradient updates, 2025. URL https://openreview.net/forum?id=u4dORXVAnx. Submitted to ICLR 2025.

A Probability models and proofs

A.1 Rounding cells and computed-kernel cells

Let $\mathcal{F}_p \subset [-1, 1]$ be the representable actions and let Q_p use round-to-nearest with a fixed tie rule. The cell of a_p is $C_p(a_p) = \{a : Q_p(a) = a_p\}$. For a continuous, strictly increasing $T : \mathbb{R} \to (-1, 1)$ with limits ± 1, the composite $Q_p \circ T$ cannot be injective: its domain is infinite and its codomain finite. If $b_a < 1$ is the midpoint below $+1$, then its positive boundary preimage is $[T^{-1}(b_a), \infty)$, up to the endpoint tie convention. The negative case is analogous. Integrating the Gaussian over this preimage gives Equation (9).

For the native computed map K_p, the corresponding mass would instead be

$$P^K_{\theta,p}(a_p \mid s) = \int_{\{u:K_p(u)=a_p\}} p_\theta(u \mid s)\, du. \tag{12}$$

The midpoint-cell implementation uses the ideal $Q_p \circ T$ model; it does not enumerate every native-kernel preimage. The Gaussian integrals also use a continuous sampling model, not the discrete law of a floating-point random-number generator. Accordingly, unbounded latent-error statements below concern the continuous Gaussian model. Real finite latent dtypes have finite ranges, but still do not permit unique reconstruction from a boundary action.

A.2 Ratio error and clipping

Proposition 1 (Equal initial ratios, different gradients) *Fix $s, u, \widehat{u}$ and the old policy. For Gaussian densities with positive scales, Equation* (8) *holds. At identical old and current parameters both ratios equal one, while their log-ratio derivatives with respect to the current mean differ by δ/σ^2_θ. If $r_u \in (1-\epsilon, 1+\epsilon)$, a log-ratio error exceeding $\log(1+\epsilon) - \log r_u$ crosses the upper clip boundary; an error below $\log(1-\epsilon) - \log r_u$ crosses the lower boundary. These crossings change the active branch for positive and negative advantages, respectively.*

Proof. Expanding $\log \widehat{r} - \log r_u$ groups the two current-policy densities and the two old-policy densities, giving Equation (8). Gaussian expansion gives Equation (6); differentiation at fixed $u, \widehat{u}, \sigma_\theta$ gives Equation (7). At equal policies the log-ratio errors cancel, but the old term is fixed when differentiating. The clipping conditions follow by adding the stated error to $\log r_u$ and using the sign of $\widehat{A}$ in Equation (1). With a common fixed scale, $\log \widehat{r} - \log r_u = \delta(\mu_\theta - \mu_{\text{old}})/\sigma^2$. A nonzero mean change therefore produces errors of arbitrarily large magnitude along the relevant half-infinite boundary preimage in the continuous model. □

A.3 THE RELATION BETWEEN RETAINED-LATENT AND CELL SCORES

Let B be a fixed, parameter-independent map into a finite set, and write $P_\theta(a) = \int_{B^{-1}(a)} p_\theta(u)\,du$. Assume differentiation under the integral is valid and $P_\theta(a) > 0$. Then

$$\nabla_\theta \log P_\theta(a) = \frac{\int_{B^{-1}(a)} p_\theta(u)\nabla_\theta \log p_\theta(u)\,du}{P_\theta(a)} = \mathbb{E}_\theta[\nabla_\theta \log p_\theta(U) \mid B(U) = a]. \tag{13}$$

For any fixed cell-measurable weight G for which the products are integrable, multiplying by $G(a)$ and summing gives

$$\mathbb{E}_\theta[G(B(U))\nabla_\theta \log p_\theta(U)] = \mathbb{E}_\theta[G(A)\nabla_\theta \log P_\theta(A)], \qquad A = B(U). \tag{14}$$

This is a finite-partition instance of marginal policy gradients (Eisenach et al., 2019). It applies to CELL with $B = Q_p \circ T$, or to an exact computed-cell model with $B = K_p$. One cannot replace one map with the other without changing the model. PPO clipping is nonlinear, and trajectory-dependent advantages under correlated exploration need not be functions of a single recorded cell. Equation (14) is therefore not a general equivalence claim for the training algorithms.

A.4 TAIL BEHAVIOUR AND THE LIMITS OF SCALAR DAMPING

Lemma 1 (Boundary-cell and reconstructed scores) *Fix a cell edge b, reconstruction $\widehat{u}_c < b$, and scale $\sigma > 0$. Let $z = (\mu - b)/\sigma \geq 0$ and $g = b - \widehat{u}_c$. The positive-cell mass is $\Phi(z)$. Its location and log-scale scores are*

$$\frac{\varphi(z)}{\sigma\Phi(z)}, \qquad -\frac{z\varphi(z)}{\Phi(z)},$$

whereas the reconstructed Gaussian scores are

$$-\frac{z}{\sigma} - \frac{g}{\sigma^2}, \qquad (z + g/\sigma)^2 - 1.$$

As $z \to \infty$ at fixed σ, both cell scores vanish and the magnitudes of both reconstructed scores diverge.

Proof. Differentiate $\log \Phi(z)$ using $\partial_\mu z = 1/\sigma$ and $\partial_{\log\sigma} z = -z$. For the point density, substitute $(\widehat{u}_c - \mu)/\sigma = -(z + g/\sigma)$ into the Gaussian scores. The limits follow from $\Phi(z) \to 1$ and $z\varphi(z) \to 0$. The cell location score is bounded by $\sqrt{2/\pi}/\sigma$ for fixed scale; this is not a bound uniform over $\sigma \downarrow 0$. □

The ratio of location-score magnitudes is

$$d(z, \sigma) = \frac{\varphi(z)}{\Phi(z)(z + g/\sigma)}. \tag{15}$$

It depends on scale as well as depth. A function of z alone cannot match it across scales. Moreover, the two location scores have opposite signs in this positive-tail regime. Nonnegative scalar damping cannot reproduce the cell score direction even if it matches the magnitude at one point.

B IMPLEMENTATION AND REPRODUCIBILITY

Training settings. The main configuration requests one million interactions, rollout length 2,048, minibatch size 128, ten optimisation epochs, learning rate 3×10^{-4}, discount 0.99, generalised-advantage parameter 0.95, clip range 0.2, entropy coefficient zero, value coefficient 0.5, and maximum gradient norm 0.5. The policy uses two 128-unit tanh hidden layers. gSDE resamples every four interactions. The primary environments and seeds are specified in Section 4; the direct HalfCheetah comparisons use observation normalisation. Resolved per-run configurations specify the full normalisation settings and wrappers.

Software and source checks. The reported platform is Python 3.12.2, PyTorch 2.11.0+cpu, SB3 2.7.1, Gymnasium 1.2.0, NumPy 1.26.4, and Linux x86-64. Arithmetic and storage are float32 unless a precision arm specifies float64. The immutable metadata does not identify the CPU model. The local standard-action PPO configuration matches upstream SB3 through two full updates in the reported parity test; this test is distinct from the squashed-gSDE call-path audit. All scientific intervention flags are off by default and selected explicitly in resolved configurations.

Table 2: First positive native input reported to store tanh exactly as $+1$ on the audited CPU kernels. These are arithmetic measurements, not universal dtype constants.

Dtype	PyTorch CPU	NumPy CPU
float16	4.50781	4.50781
bfloat16	3.46875	not available
float32	9.01084	10.0
float64	19.0615	18.9903

Collection and replay audit. The collection method is `ActorCriticPolicy.forward`; the replay method is `evaluate_actions`. Both pass the action to `log_prob`. In the gSDE distribution, a configured bijector is inverted before the Gaussian density is evaluated. The inverse clamps using `finfo(dtype).eps`. The separate 10^{-6} constant regularises the Jacobian correction, not that inverse clamp (Stable-Baselines3 Contributors, 2025). These method names and the pinned version replace line numbers that change between the local fork and upstream.

DUAL and CELL implementation. DUAL adds a rollout array for the sampled pre-tanh values and passes the retained, detached values to collection and replay likelihood evaluation. The old likelihood is collected at that same latent. A stable tanh Jacobian, when required, can be evaluated as $2\{\log 2 - u - \mathrm{softplus}(-2u)\}$; it cancels from a fixed-latent PPO ratio. With N stored transitions, action dimension d, and float32 latents, the added latent array contains $4Nd$ bytes, excluding metadata and temporary tensors. No runtime-overhead measurement is claimed. The zero entropy coefficient keeps an entropy-estimation change out of these comparisons. CELL constructs predecessor/successor midpoints and uses log-space CDF differences or narrow-cell quadrature to avoid subtracting nearly equal probabilities. The reported unit tests cover boundary tails, ordered cells, finite log masses, same-policy ratios, and gradient checks.

Independent implementation. The reproduction is a single-file squashed-Gaussian PPO implementation, not gSDE, with the same conceptual rollout, network, optimiser, discount, advantage, and clipping settings above. It runs on CPU in float32 and uses a 10^{-6} inverse clamp. Both methods log the latent, but RECON's optimiser does not read it. Import checks report no dependency on SB3 or the local fork. The shared mean offset affects neither rewards nor observations. A pre-run check compares retained-latent likelihoods through two code paths over 512 samples; the DUAL identity-error field is zero by construction, not an independent random measurement.

Evaluation and source separation. Task returns use saved checkpoints and common deterministic evaluation. Failed endpoints remain absent rather than receiving an invented return. The primary table uses 45 runs; the five MASK runs and six independent reproduction runs are separate. The older 110-run suite is also separate because its source snapshot differs. Its last complete telemetry rollout ends at 999,424 interactions under the registered one-million-interaction checkpoint label. The two historical kettle endpoints are kept distinct: transition 199 for the mechanism lineage and transition 200 for the qualified competence task. The latter uses 1 kg of water, a 20°C start and ambient temperature, a 100°C target, and one-second steps. Action a gives $P = 1500(a + 1)$ W; heat capacity is 4184 J/(kg K), loss is $5(T - 20)$ W, step reward is $-10^{-6}P$, and terminal reward adds $-0.01|100 - T_{200}|$.

Artifact record. The supplementary run records associate each outcome with its configuration, command, source hash, terminal state, evaluation, and telemetry. The sanitised package contains a deduplicated source snapshot and analysis inputs; large checkpoints and full payloads are separately checksum indexed. Source hashes and the run inventory, rather than campaign names alone, define which results can be compared. The manuscript reports the saved outcomes; it does not pool incompatible snapshots.

Clamp choice. The SB3 float32 machine-epsilon clamp returns 8.3178. A nominal 10^{-6} clamp has real-arithmetic inverse 7.2543, while the tightest interior float32 action, $1 - 2^{-24}$, has inverse 8.6643. Literal clamp constants are themselves rounded on execution, so exact native values must

Table 3: **Earlier HalfCheetah initialisation comparison.** Deterministic return is completed-run mean $\pm$ standard error. All settings use observation normalisation. The initial log scale is the gSDE exploration parameter, not a state-independent effective policy scale.

Method	Initial log scale	Completed	Return
Standard PPO	not applicable	5/5	2682.5 ± 598.5
RECON	0.0	3/5	643.7 ± 451.9
CELL	0.0	5/5	1050.7 ± 135.8
RECON-INIT2	-2	5/5	3475.6 ± 1006.7
CELL-INIT2	-2	5/5	3785.8 ± 825.3

be measured in the implementation. Every finite reconstruction still selects one value from a tail of possible samples.

C SUPPORTING CONTROLS AND CONDITIONAL PERFORMANCE

Initialisation and return. Table 3 retains the earlier HalfCheetah comparison. It must not be pooled with Table 1. In the principal three-task suite, RECON completes 13/15 runs and CELL completes 15/15. The mean-based tanh reference is crossed in 15/15 runs for each, compared with 0/15 for CELL-INIT2. The separate HalfCheetah RECON-INIT2 control completes 5/5 and has no mean-reference crossing. These are not zero-tail- probability claims.