FLASH: IMPROVING DOWNSTREAM EXPERIENCE UTILIZATION IN VISUAL MODEL-BASED REINFORCEMENT LEARNING THROUGH TASK-RELEVANT EPISODIC RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in visual model-based reinforcement learning have continually improved control performance through higher-fidelity generative world models and visual representations, often at additional computational cost. However, our probing experiments show that lightweight latent representations can retain substantial task-relevant information even when visual reconstruction is imperfect, suggesting that representation fidelity alone may not determine downstream control performance. These findings suggest that downstream utilization of task-relevant information already available in latent representations may itself be an important bottleneck. We therefore propose Fast Latent Anchoring via Salience and Hashing (FLASH), which uses salient learning events as retrieval anchors to identify structurally similar historical contexts through a hash-indexed episodic memory. FLASH jointly reuses these contexts in Actor-Critic learning, improving downstream experience utilization without replacing the underlying encoder or world-model architectures. Across Atari 100k, FLASH improves STORM's mean and median human-normalized scores by 4.71% and 8.37%, respectively, and DRAMA's by 8.33% and 5.35%. As a plug-and-play module that can be readily integrated into existing agents, FLASH achieves these performance gains with a 5.11–17.61% increase in wall-clock training time across the evaluated configurations.

1 INTRODUCTION

Visual model-based reinforcement learning (MBRL) relies on learned latent representations for prediction, policy learning, and value estimation from high-dimensional observations. Latent world-model agents compress visual observations into compact states and use learned dynamics to support downstream control (Hafner et al., 2019a; 2020; Zhang et al., 2023; Wang et al., 2025). Prior work has sought to improve these representations through reconstruction-based, contrastive, predictive, and reconstruction-free objectives (Hafner et al., 2019b; Laskin et al., 2020; Schwarzer et al., 2020; Stooke et al., 2021; Deng et al., 2022), while more recent generative world models aim to preserve finer visual detail (Alonso et al., 2024; Lee et al., 2026). Although these approaches can provide richer representations, higher-fidelity generative modeling can introduce additional computation without necessarily providing more task-relevant information for downstream control.

Limited control with a compact representation may arise because task-relevant information is absent from the latent state or because downstream policy and value learning does not effectively use information that remains encoded. When task-relevant information remains encoded, downstream learning may offer opportunities to improve control without replacing the underlying representation. In data-limited MBRL, policy and value learners are optimized using rewards and bootstrapped return targets from finite sampled or imagined trajectories (Hafner et al., 2020; Wang et al., 2025). Thus, the presence of decodable information does not guarantee its effective use in downstream control, making the experiences presented to these learners a distinct intervention point.

Based on these hypotheses, we analyze lightweight latent representations. Latent-similarity and supervised probing analyses indicate that selected task variables remain decodable under clean obser-

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.

vations despite imperfect visual reconstruction, while their decodability degrades substantially under background corruption. We also observe substantial performance variation across independent training runs in the original results of STORM (Zhang et al., 2023), a Transformer-based MBRL: in 7 of the 26 Atari 100k games, the difference between the highest- and lowest-scoring training seeds exceeds 1.0 in human-normalized score (HNS) (Appendix K). Together, these observations motivate investigating how downstream learning utilizes the information and experiences already available to the agent, as discussed in Section 3.

We propose Fast Latent Anchoring via Salience and Hashing (FLASH), a plug-and-play mechanism that improves downstream experience utilization by selectively reusing prior experiences during Actor-Critic learning. FLASH identifies salient learning events, retrieves similar historical contexts through a hash-indexed episodic memory, and incorporates the resulting groups into the same training batch, while leaving the underlying encoder, world-model, and Actor-Critic architectures unchanged.

Our contributions are threefold:

- We empirically demonstrate that lightweight latent representations retain decodable task-relevant information despite imperfect visual reconstruction (Section 3).
- We introduce FLASH, a new plug-and-play mechanism that retrieves structurally similar historical contexts around salient learning events and reuses them for Actor-Critic learning without modifying the underlying encoder or world model.
- Our experiments demonstrate performance improvements across diverse games with both STORM and DRAMA (Zhang et al., 2023; Wang et al., 2025), providing empirical evidence that more effective utilization of previously encountered experiences can enhance downstream control.

2 Related Work

Latent World Models and Representation Learning. Dreamer and DreamerV2 learn policies through latent imagination (Hafner et al., 2019a; 2020), while STORM and DRAMA use Transformer-based and Mamba-based dynamics, respectively (Zhang et al., 2023; Wang et al., 2025). Prior work has also improved visual representations through reconstruction (Hafner et al., 2019b), contrastive or predictive objectives (Laskin et al., 2020; Schwarzer et al., 2020), and reconstruction-free representation learning (Stooke et al., 2021; Deng et al., 2022); DIAMOND instead uses a diffusion world model to preserve visual detail (Alonso et al., 2024). These approaches primarily intervene in the learned representation or world model. FLASH instead retains the base agent's encoder and world-model architectures and changes the historical contexts used to initialize Actor-Critic imagination through targeted experience retrieval.

Episodic Control and Retrieval-Augmented Reinforcement Learning. MFEC uses nearest-neighbor lookup over episodic state-action memories to estimate values (Blundell et al., 2016), while NEC combines learned embeddings with a differentiable neural dictionary (Pritzel et al., 2017). EVA retrieves related past trajectories to augment parametric value estimates (Hansen et al., 2018), and later methods organize episodic memory through associative graphs or state abstraction (Zhu et al., 2020; Li et al., 2023). However, these value-centric uses of episodic memory do not directly address how previously encountered, related contexts can be reintroduced as training contexts for Actor-Critic imagination, rather than only as direct corrections to value estimates. FLASH differs in how retrieved experience enters learning: it retrieves structurally similar historical contexts around selected anchors and inserts them into the same imagination batch, rather than directly using retrieved returns or values for value estimation. Hash-indexed retrieval further avoids requiring a full nearest-neighbor search or an explicit associative graph.

Prioritized Experience Replay and Priority Signals. Prioritized Experience Replay uses TD-error to prioritize transitions (Schaul et al., 2015), while related methods incorporate value-based signals (Cao et al., 2019), curiosity (Kauvar et al., 2023), sequence-level priorities (Karimpanal & Bouffanais, 2018; Brittain et al., 2019), or actor-critic-specific criteria (Saglam et al., 2023). Prioritized Sweeping instead schedules backups according to changes in value estimates across updates (Moore & Atkeson, 1993). However, prioritizing high absolute state values can repeatedly select already-sustained high-value regions rather than transitions into them, while TD-error alone captures sur-

prise without indicating whether it coincides with a meaningful rise in estimated value along the trajectory. FLASH uses a different signal: normalized TD-error is combined multiplicatively with the signed temporal value difference $\Delta V_t = V(s_t) - V(s_{t-1})$, which compares consecutive states along a trajectory rather than successive estimates of the same state. The resulting score selects anchors rather than directly determining replay frequency; each selected anchor subsequently triggers retrieval of structurally similar contexts for joint Actor-Critic training.

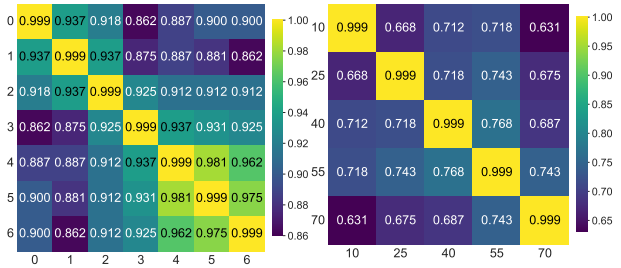
Appendix A provides further background and comparisons.

3 Problem Analysis and Motivation

The original STORM results show substantial variation in final evaluation performance across training seeds. In 7 of the 26 Atari 100k games, the difference between the highest and lowest scoring seeds exceeds 1.0 human-normalized score (HNS) (Appendix K). This motivates investigating how training experiences affect downstream control performance. If the encoder discards task-relevant information, downstream learning cannot recover it. We therefore examine whether task-relevant visual changes are reflected in the latent state and whether the corresponding variables remain decodable (Sections 3.1 and 3.2). Even when such information is retained, the latent representation may have limitations that affect its usefulness for downstream control. We then examine how this decodability depends on the surrounding visual context (Section 3.3).

3.1 Latent Sensitivity to Task-Relevant Visual Changes

In controlled Seaquest observations, we preserve the full frame and vary only either the collected-diver count or a diver's position; the complete observation, instead of a cropped diver region, is encoded. Consecutive latent states from ordinary trajectories have a mean cosine similarity of 0.997, whereas adjacent diver counts average 0.949, count differences of three or more average 0.893, and larger diver displacements reduce similarity to as low as 0.63 (Figure 1(a–b)). These patterns suggest sensitivity to the task-relevant element changes, but not semantic recoverability.



(a) Diver Count (0–6) (b) Diver Position

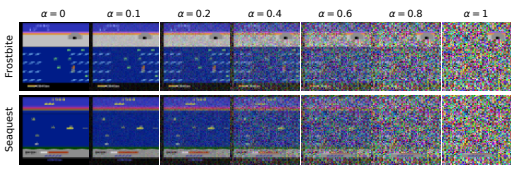
Figure 1: Pairwise cosine similarities of latent states when only the diver count or diver position varies, and background is fixed.

3.2 Decodability of Task-Relevant Information

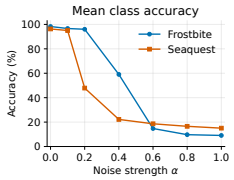
We next assess whether diver count, presence, and position are decodable from the latent representation using ground-truth labels derived from the Atari emulator's internal game-state memory (RAM). The encoder used for this analysis had never encountered states with four to six collected divers during MBRL training, although such high-count states are required for diver-return behavior in Seaquest (Zahavy et al., 2016). The probe itself is supervised on examples spanning all counts from 0 to 6, so this experiment does not test zero-shot generalization of the probe. As shown in Table 1, count prediction reaches 99.5% accuracy over counts 0–6 and 95.7% over the pooled 4–6 subset; diver presence reaches 92.4%, and position prediction has a mean error of 12.44 pixels. Complete protocols and validation results are reported in Appendix J. These results suggest that the selected task variables remain recoverable from the frozen latent representation, including for high-count states not encountered during encoder training.

3.3 Context Dependence of Latent Decodability

We next assess how changes in the surrounding visual context affect the decodability of task-relevant information, examining whether predictions rely on target-specific evidence or contextual cues. High clean-data decodability does not reveal whether prediction relies on target-specific evidence or on



(a) Background-noise examples



(b) Mean class accuracy

Figure 2: Background corruption and supervised decoding. (a) Representative Frostbite and Seaquest observations across noise strengths α. (b) Mean class accuracy of separately trained probes at each strength.

Table 1: Clean probing results in Seaquest using MLP probes trained on latent representations from a frozen encoder.

Target	Metric	Result
Count (0–6)	Acc.	99.5%
Count (4–6)	Acc.	95.7%
Presence	Acc.	92.4%
Position	Mean error	12.44 px

contextual cues that happen to co-vary with the target. For example, if an igloo stage generally increases over time, other time-varying visual factors could serve as predictive proxies and become unreliable when that relationship changes. We therefore preserve the target region while progressively corrupting the surrounding pixels with uniform noise (Figure 2(a)). A fresh probe is trained at each corruption strength, allowing it to adapt to whatever latent coordinates remain predictive under that condition; complete protocols are provided in Appendix J.4.

Despite condition-specific retraining, mean class accuracy at $\alpha = 1$ falls from 96.29% to 15.04% in Seaquest and from 98.27% to 9.07% in Frostbite (Figure 2(b)). Thus, the high clean-data decodability is not robust to large changes in the surrounding visual context. Because the corruption also moves observations away from the encoder's training distribution, this experiment does not isolate the mechanism behind the degradation; full curves, validation results, and class-wise results are reported in Appendix J.4.

Together, these analyses indicate that selected task-relevant information remains decodable from lightweight latent representations under clean observations, though its decodability can degrade under changes in visual context. These findings motivate investigating whether downstream experience utilization can be improved to better exploit task-relevant information already present in the latent representation and thereby enhance control performance.

4 METHODOLOGY

FLASH targets imagination-based Actor-Critic agents with learned embeddings for retrieval and replay-based rollout initialization. We instantiate FLASH on STORM (Zhang et al., 2023) and DRAMA (Wang et al., 2025); more broadly, it can augment latent world-model agents that learn from imagined trajectories (Hafner et al., 2023) and diffusion-based agents (Lee et al., 2026) using observation-encoder features, without modifying their encoder or world-model architectures.

FLASH has three components (Figure 3). First, salience-guided anchor selection identifies salient transitions and registers them as anchors (Section 4.1). Second, a hash-indexed episodic memory retrieves historical contexts with matching latent structure for each anchor (Section 4.2). Third, the retrieved groups are incorporated into the imagination batch together with randomly sampled contexts (Section 4.3). Appendix D details our FLASH procedure.

4.1 SALIENCE-GUIDED ANCHOR SELECTION

TD-error magnitude is commonly used as a proxy for how surprising a transition is in prioritized replay (Schaul et al., 2015). In FLASH, salience denotes an anchor-selection score that combines TD-error magnitude with the signed temporal value difference, rather than TD-error magnitude alone. During each world model training step, given a batch of sampled trajectories with value estimates $V(s_t)$ from the critic, we compute:

$$\delta_t^{\mathrm{TD}} = \left| r_t + \gamma\, V(s_{t+1})(1 - d_t) - V(s_t) \right|, \tag{1}$$

$$\Delta V_t = V(s_t) - V(s_{t-1}), \tag{2}$$

where r_t is the reward, γ is the discount factor, and d_t is the termination flag. To normalize across environments with varying reward scales, we maintain per-environment exponential moving averages (EMA) of the mean and variance:

$$\bar{\mu}_e \leftarrow (1-\alpha)\bar{\mu}_e + \alpha \cdot \mathbb{E}[\delta_e^{\mathrm{TD}}], \qquad \bar{\sigma}_e^2 \leftarrow (1-\alpha)\bar{\sigma}_e^2 + \alpha \cdot \mathrm{Var}[\delta_e^{\mathrm{TD}}], \tag{3}$$

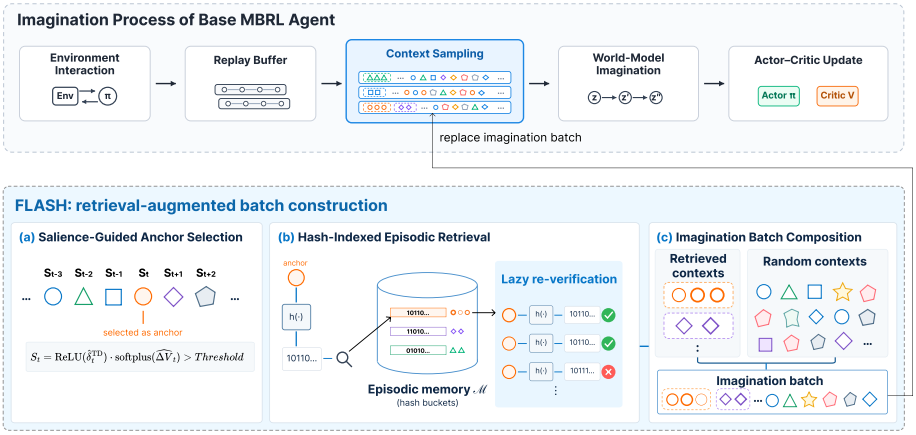


Figure 3: Overview of FLASH pipeline. FLASH is integrated into the base MBRL agent to improve the utilization of previously encountered experiences for downstream control. (a) Frames satisfying the salience criterion are selected as anchors. (b) Each anchor's latent state is hashed to retrieve candidates with matching hash keys, which are then re-encoded and re-hashed for verification. (c) Verified candidates and their anchors form retrieved context groups, which replace a subset of randomly sampled contexts in the imagination batch for Actor-Critic learning. Geometric shapes represent frames, and sequences of shapes denote context windows.

and analogously for ΔV. For each time step t in the sampled trajectories, we compute the salience score S_t using the EMA statistics from preceding batches and then update the statistics with the current batch. The salience score is:

$$S_t = \mathrm{ReLU}\left(\frac{\delta_t^{\mathrm{TD}} - \bar{\mu}_e}{\sqrt{\bar{\sigma}_e^2 + \epsilon}}\right) \cdot \mathrm{softplus}\left(\frac{\Delta V_t - \bar{\mu}_e^V}{\sqrt{\bar{\sigma}_e^{2,V} + \epsilon}}\right). \tag{4}$$

Here, ReLU is the rectified linear unit (Nair & Hinton, 2010), while $\mathrm{softplus}$ is its smooth approximation (Dugas et al., 2000). The ReLU term assigns zero contribution when the normalized TD-error is non-positive, while the softplus term modulates the score according to the signed temporal value difference. For each trajectory sequence in the batch, if the maximum salience score exceeds the threshold ($\max_{t \in \mathrm{seq}} S_t > \tau_z$), the replay-buffer pointer corresponding to $t^* = \mathrm{argmax}_{t \in \mathrm{seq}} S_t$ is shifted by Δ_{anchor} steps and registered as an active anchor $a = (p_{t^*+\Delta_{\mathrm{anchor}}}, e)$ in a first-in, first-out (FIFO) queue $\mathcal{A}$. The model-specific anchor offset is described in Appendix B.

4.2 Hash-Indexed Episodic Retrieval

During training, the world-model observation encoder and posterior head, jointly denoted by f_θ, map an observation o_t to a latent representation $z_t = f_\theta(o_t) \in \mathbb{R}^d$. FLASH uses this representation consistently for hash insertion, candidate re-verification, and Global Rebuild. Base-agent-specific details of the hashed representation are provided in Appendix C.

We apply a hash function $h : \mathbb{R}^d \to \{0, 1, \ldots, 2^B - 1\}$ to produce a discrete hash key:

$$h(z_t) = \sum_{i=1}^{B} \mathbf{1}\left[\left(W_{\mathrm{proj}}^\top (z_t - \mu)\right)_i > 0\right] \cdot 2^{i-1}, \tag{5}$$

where B is the number of hash bits, μ is a centering vector, and $W_{\mathrm{proj}} \in \mathbb{R}^{d \times B}$ is the projection matrix. Before the first Global Rebuild, we set $\mu = \mathbf{0}$ and sample W_{proj} from an isotropic Gaussian distribution, yielding standard random-hyperplane locality-sensitive hashing (LSH) (Charikar, 2002). At the first Global Rebuild, we instead compute μ from the current replay-buffer latents and set W_{proj} to the leading B principal directions, yielding a data-dependent PCA-based binary hash (Lin et al., 2010). Subsequent rebuilds refit this PCA basis to the current latent distribution. Each transition pointer (p_t, e), where p_t is the circular buffer index and e is the environment index, is inserted into a hash bucket $\mathcal{B}_{h(z_t)}$.

Neighbor Retrieval and Adaptive Index Maintenance. At each imagination step, where the agent generates imagined trajectories for policy training, FLASH pops up to M anchors from $\mathcal{A}$. For each anchor a with hash key k_a, we sample up to $m \times (n-1)$ candidate pointers from bucket $\mathcal{B}_{k_a}$, excluding the anchor itself. Because f_θ changes during training, previously stored bucket assignments may become inconsistent with the current representation. FLASH re-encodes each sampled candidate using the current world model and retains it only if its recomputed hash key matches k_a, yielding a verified neighbor set $\mathcal{N}_a$ with $|\mathcal{N}_a| \leq n$, including the anchor itself. The hit rate is defined as the fraction of sampled candidates whose recomputed keys remain consistent with the queried bucket. After retrieval is activated, if the average hit rate falls below threshold η, FLASH performs a Global Rebuild: the replay buffer is re-encoded using the current f_θ, the centering vector μ and PCA projection W_{proj} are recomputed, all hash buckets are cleared, and all transitions are re-hashed. A cooldown period limits the frequency of Global Rebuilds.

4.3 Imagination Batch Composition

The retrieved groups provide contexts for imagined trajectories used in Actor-Critic training. For each retrieved group $\mathcal{N}_a$, we extract the preceding L-step context window of observations and actions from the replay buffer (validating that no episode termination occurs within the window). Let $N_{\text{ret}} = \sum_a |\mathcal{N}_a|$ be the total number of retrieved contexts, including both anchors and their matched neighbors. These are concatenated with randomly-sampled contexts to form the imagination batch:

$$\mathcal{D}_{\text{imagine}} = \underbrace{\{(o_{t-L:t},\, a_{t-L:t})\}_{i=1}^{N_{\text{rand}}}}_{\text{random samples}} \cup \underbrace{\bigcup_{a \in \text{popped}} \{(o^{(j)}_{t-L:t},\, a^{(j)}_{t-L:t})\}_{j \in \mathcal{N}_a}}_{\text{retrieved contexts (anchors + neighbors)}}, \tag{6}$$

where $N_{\text{rand}} = \max\big(0,\, N_{\text{total}} - N_{\text{ret}}\big)$, and N_{total} is the base imagination batch size. We choose M and n such that $Mn \leq N_{\text{total}}$. Since each of the at most M anchor groups contains at most n contexts (including the anchor), $N_{\text{ret}} \leq Mn \leq N_{\text{total}}$. Consequently, $N_{\text{rand}} + N_{\text{ret}} = N_{\text{total}}$, so substituting retrieved contexts for random samples strictly preserves the physical batch size.

Group Weighting. Each retrieved group $\mathcal{N}_a$ is weighted such that its member weights sum to one, while every random sample receives $w = 1$. The unnormalized batch weight is:

$$W_{\text{sum}} = N_{\text{rand}} + M_{\text{valid}} \leq N_{\text{total}}, \tag{7}$$

where M_{valid} is the number of valid anchor groups. The inequality is strict when at least one group contains more than one context. Actor-Critic losses (policy, value, entropy) are normalized by W_{sum}:

$$\mathcal{L}_{\text{AC}} = \frac{1}{W_{\text{sum}}} \sum_{i \in \mathcal{D}_{\text{imagine}}} w_i \left[\mathcal{L}_{\text{actor}}(s_i, a_i) + \mathcal{L}_{\text{critic}}(s_i) - \beta\, \mathcal{H}(\pi(\cdot|s_i))\right]. \tag{8}$$

5 Experiments

5.1 Experimental Setup

We evaluate FLASH on Atari 100k, comprising 26 games with a standard budget of 100,000 environment steps (400,000 frames) per game. We compare STORM (Zhang et al., 2023) and DRAMA (Wang et al., 2025) with their FLASH-augmented counterparts under matched training settings. Our protocol uses four training seeds per game and evaluates each final checkpoint over 20 episodes for STORM and 10 for DRAMA, averaging returns first within each run and then across training seeds. For Freeway, STORM uses offline expert demonstrations following its original setup. To account for game-specific score scales, we report mean and median human-normalized scores (HNS) across games, where $\text{HNS} = (A - R)/(H - R)$ and A, R, and H denote the agent, random-policy, and human scores, respectively. FLASH uses a 50,000-step retrieval warmup; further implementation details are provided in Appendix C.

5.2 Main Performance and Analysis

As shown in Table 2, FLASH improves mean and median HNS on both STORM and DRAMA. Their larger gains than interquartile mean (IQM) motivate the game-level analyses, which examine

Table 2: Atari 100k performance. The "+ours" columns denote agents augmented with FLASH, and Δ denotes the difference between FLASH and its corresponding baseline.

Game	Random	Human	STORM	STORM+ours	Δ	DRAMA	DRAMA+ours	Δ
Alien	227.8	7127.7	1091.6	1147.9	+56.3	1051.5	972.2	-79.3
Amidar	5.8	1719.5	208.6	169.2	-39.4	140.3	130.2	-10.1
Assault	222.4	742.0	821.1	723.7	-97.4	383.5	449.9	+66.4
Asterix	210.0	8503.3	677.5	759.4	+81.9	1680	1397.5	-282.5
BankHeist	14.2	753.1	593.9	542.1	-51.8	458.8	278.8	-180
BattleZone	2360.0	37187.5	12212.5	13125	+912.5	10775	8600	-2175
Boxing	0.1	12.1	81.9	83.5	+1.6	63.2	75.3	+12.1
Breakout	1.7	30.5	12.4	12.8	+0.4	14.7	16.6	+1.9
ChopperCommand	811.0	7387.8	1688.8	1425	-263.8	1382.5	1550	+167.5
CrazyClimber	10780.5	35829.4	58890	58632.5	-257.5	76802.5	74437.5	-2365
DemonAttack	152.1	1971.0	181.6	155.1	-26.5	187.2	151	-36.2
Freeway	0.0	29.6	30.7	30.4	-0.3	15.7	16.1	+0.4
Frostbite	65.2	4334.7	1742.4	2054.9	+312.5	1311.5	1319.2	+7.7
Gopher	257.6	2412.5	6996.3	7982.7	+986.4	1934.5	2187.5	+253
Hero	1027.0	30826.4	11785.6	13229.2	+1443.6	7949.8	7987.1	+37.3
Jamesbond	29.0	302.8	520.6	603.8	+83.2	428.8	535	+106.2
Kangaroo	52.0	3035.0	2071.2	2258.8	+187.6	1350	1335	-15
Krull	1598.0	2665.5	6541	6800.9	+259.9	9076	9569	+493
KungFuMaster	258.5	22736.3	25168.3	25105	-63.3	16480	16137.5	-342.5
MsPacman	307.3	6951.6	2014.6	2015.9	+1.3	2460.2	2525.2	+65
Pong	-20.7	14.6	6.4	12.2	+5.8	8.8	14.7	+5.9
PrivateEye	24.9	69571.3	602.1	3.6	-598.5	100	-4.9	-104.9
Qbert	163.9	13455.0	3266.6	3429.4	+162.8	1446.2	1496.2	+50
RoadRunner	11.5	7845.0	15543.8	16845	+1301.2	11812.5	12537.5	+725
Seaquest	68.4	42054.7	459	497.8	+38.8	626	558	-68
UpNDown	533.4	11693.2	6817.8	6891.2	+73.4	5269.5	6038.8	+769.3
#Superhuman (↑)	0	N/A	9	8	-1	5	6	+1
Mean (↑)	0.000	1.000	1.105	1.157	+0.052	0.948	1.027	+0.079
Median (↑)	0.000	1.000	0.478	0.518	+0.040	0.374	0.394	+0.020
IQM (↑)	0.000	1.000	0.554	0.570	+0.016	0.374	0.390	+0.016
Optimality Gap (↓)	1.000	0.000	0.455	0.446	-0.009	0.547	0.540	-0.007

whether FLASH better exploits previously encountered successful experiences. Effects vary across backbones and environments. STORM uses uniform context sampling, whereas DRAMA uses dynamic frequency-based sampling (DFS). FLASH increments DFS counts for retrieved frames, but group weighting and interactions among related contexts can make their Actor-Critic contribution differ from ordinary samples. Backbone-specific latent representations and world-model dynamics may further affect retrieval relevance and imagined-trajectory quality.

In Pong, Figure 4 shows that FLASH retrieves experiences from similar configurations, including both similar and contrasting outcomes, such as scoring cases and a neighbor where the ball rebounds off the opponent's paddle. Joint reuse may help the Actor-Critic distinguish scoring opportunities from similar non-scoring situations, potentially contributing to the gain in Pong. In Frostbite, Figure 5 shows that after successful behavior is encountered during the shared warmup, FLASH achieves higher subsequent training returns than baseline; its final evaluation distribution has a higher upper tail and higher first and third quartiles, consistent with better exploitation of previously encountered successes. Appendix E analyzes this relationship across training seeds, while Appendix I examines the critic's responses to controlled igloo-stage changes.

Over the full 100k-step training budget, FLASH increases wall-clock training time by 5.11–7.41% for STORM and by 17.61% for DRAMA; detailed measurements are provided in Appendix C.1.

5.3 Ablation Studies

To assess FLASH design choices, we ablate neighbor retrieval ($n = 16$ vs. $n = 1$), the value signal ($\Delta V_t = V(s_t) - V(s_{t-1})$ vs. $V(s_t)$), and score combination (multiplicative vs. additive); details are provided in Appendix L.

While the main performance evaluation uses four training seeds per game, each ablation uses two to four paired training seeds per game. For each ablation, Full FLASH and the corresponding variant are evaluated using the same seeds, and their scores are averaged over these paired runs. Consequently, the reported Full FLASH scores may differ across ablations and from those in the main performance evaluation due to differences in the seed sets used for each comparison.

Neighbor Retrieval. FLASH with neighbor retrieval achieves higher mean HNS than the anchor-only variant, indicating that structurally similar historical contexts can provide useful information

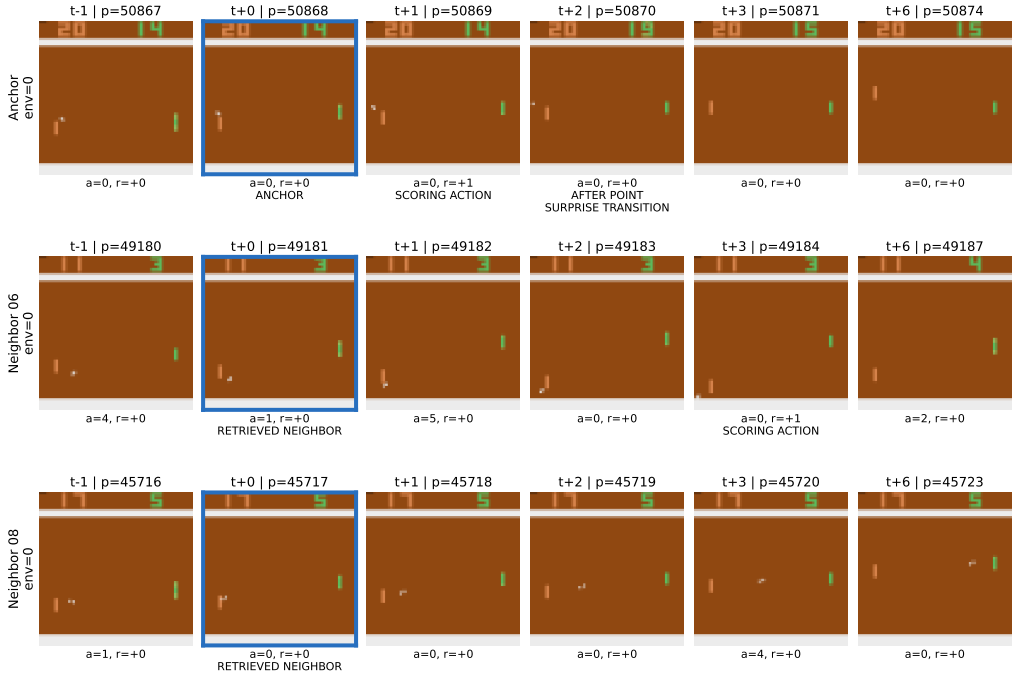


Figure 4: Selected actual replay trajectories for a scoring-related anchor and two of its 15 retrieved neighbors in Pong. Blue boxes mark the retrieved observations at $t = 0$; actions and rewards correspond to the outgoing transitions from each frame.

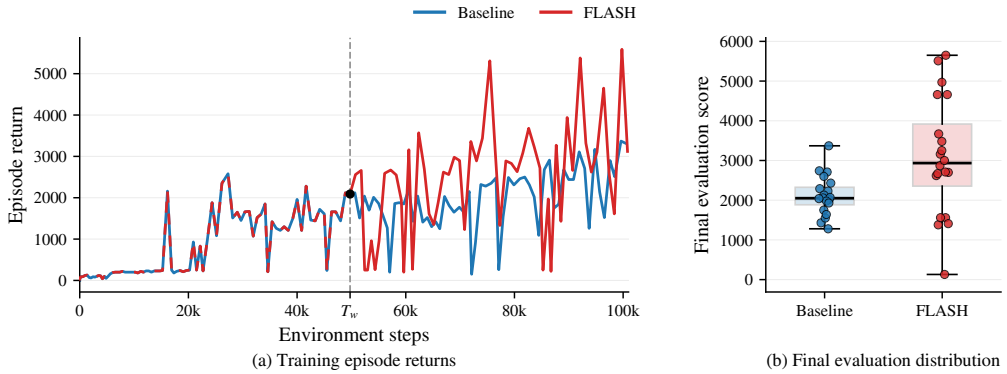


Figure 5: Training and evaluation performance on Frostbite for one matched training-seed pair. (a) Training episode returns, with Baseline and FLASH sharing the same training trajectory until retrieval activation at T_w. (b) Final evaluation return distributions over 20 episodes per method, using the corresponding checkpoints at 100k environment steps.

beyond the selected anchor alone. As illustrated in Pong, retrieved neighbors can include similar situations with contrasting outcomes, potentially helping the Actor-Critic distinguish outcome-relevant differences among recurring configurations. However, No neighbor performs better in several games, suggesting that such contrasts are not always equally informative: when outcome differences among retrieved contexts are not well captured by the retrieval representation or context window, these contrasts may be less informative for Actor-Critic learning. This motivates retrieval criteria that account not only for structural similarity but also for the behavioral relevance of the retrieved contexts.

Value Signal. Using temporal value difference yields higher mean HNS than using the state value $V(s_t)$, with notable gains in Gopher, KungFuMaster, and Pong. In these games, repeatedly executing actions that directly produce rewarding outcomes is important, making transitions into more valuable states potentially informative for anchor selection. In contrast, the absolute-value variant performs better in Krull, BankHeist, and MsPacman. MsPacman, for example, also contains tem-

Table 3: Ablations on STORM. We compare retrieval target $n = 16$ with anchor-only retrieval ($n = 1$), temporal value difference ΔV_t with state value $V(s_t)$, and multiplicative with additive score combination; complete results are in Tables 19, 20, and 21. Game scores use the paired training seeds of each comparison, while aggregate metrics use all games with paired results; IQM pools per-seed human-normalized scores. Bold indicates the best score within each comparison group (lower for Optimality Gap). Across all three comparisons, the default FLASH choices yield better aggregate metrics than their ablated counterparts, although individual games can favor the alternatives.

	Neighbor retrieval		Value signal		Combination	
Game / Metric	FLASH ($n = 16$)	Anchor only ($n = 1$)	ΔV_t	$V(s_t)$	S_t^{mult}	S_t^{add}
Assault	723.7	**833.9**	**751.7**	730	**751.7**	702.9
BankHeist	599.2	**622**	599.2	**659.2**	**599.2**	566.2
BattleZone	**13150**	10766.7	12600	**13450**	12600	**13275**
Boxing	**82.1**	78.4	**83.8**	81.6	**83.8**	76.9
ChopperCommand	1497.5	**1807.5**	1497.5	**1622.5**	1497.5	**1665**
Gopher	**7952**	4862	**7952**	4947.5	**7531**	2516
Kangaroo	2285	**2480**	2900	**3140**	**2900**	2885
Krull	**6716**	6075	6675.8	**7287**	**6783.5**	5904.5
KungFuMaster	**27535**	18785	**27535**	17935	**27535**	23222.5
MsPacman	**2059.5**	1838.2	1885.8	**2422.2**	1885.8	**2373.8**
Pong	**15.4**	8.7	**15.4**	7.5	**15.4**	12.2
#Superhuman (↑)	**8**	7	**9**	7	**9**	7
Mean (↑)	**1.164**	1.042	**1.178**	1.086	**1.163**	0.984
Median (↑)	**0.518**	0.501	**0.538**	0.452	**0.536**	0.458
IQM (↑)	**0.565**	0.538	**0.642**	0.591	**0.599**	0.565
Optimality Gap (↓)	**0.442**	0.456	**0.434**	0.450	**0.446**	0.455

porarily favorable states: after consuming a power pellet, the agent can obtain substantial rewards by eating otherwise threatening ghosts. Absolute-value selection can continue to emphasize experiences within such high-value states rather than primarily their entry transitions. These differences suggest that the relative usefulness of the two signals may depend on whether rewarding behavior is characterized more by value-increasing transitions or by exploiting already favorable states.

Score Combination. The multiplicative score S_t^{mult} achieves higher mean HNS than the additive score S_t^{add}, suggesting that jointly high TD-error and temporal value difference are generally useful for anchor selection. Across separate ablations, 7 of the 10 games where Additive performs better also perform better with Absolute Value. Both variants relax the default emphasis on transitions where surprise and positive value increase coincide: Additive can trigger when either component is strong, while Absolute Value can emphasize already high-value states without requiring a sharp value increase. This overlap suggests that some games may contain informative experiences outside such coincident events, for which a less restrictive selection criterion can be beneficial.

6 Conclusion

We investigated downstream experience utilization as an intervention point for data-efficient visual model-based reinforcement learning. Our analyses show that lightweight latent representations can retain substantial task-relevant information under clean observations even when visual reconstruction is imperfect, while also revealing that such decodability can depend strongly on the surrounding visual context. Motivated by these observations, we introduced FLASH, a plug-and-play mechanism that identifies salient learning events, retrieves structurally similar historical contexts through a PCA-based hash-indexed episodic memory, and reuses them in Actor-Critic learning without replacing the underlying encoder or world-model architecture. Across Atari 100k, FLASH improves aggregate performance for both STORM and DRAMA, while game-level analyses and ablations show that its benefits depend on which experiences are selected and jointly reused. These results support experience utilization as a complementary direction to improving representation and world-model quality: stronger downstream control can arise not only from learning richer representations, but also from making more effective use of informative experiences that the agent has already encountered.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing, and we have reviewed all AI-assisted work. We have not used generative AI tools for generating the core scientific content of this study, including the formulation of research ideas, the derivation of scientific conclusions, or the interpretation of results. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We acknowledge that all datasets used in this work are publicly available and comply with relevant licensing terms. No personally identifiable information (PII) or sensitive data was collected or used. We believe this technology holds significant potential for beneficial applications. We are committed to the responsible advancement of this field and encourage continued research to establish clear ethical guidelines for deployment.

REPRODUCIBILITY STATEMENT

We provide implementation details and experimental settings in the main paper and appendix. The source code and instructions for reproducing our experiments are included in the supplementary ZIP archive submitted with this paper.

REFERENCES

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

Charles Blundell, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z Leibo, Jack Rae, Daan Wierstra, and Demis Hassabis. Model-free episodic control. *arXiv preprint arXiv:1606.04460*, 2016.

Marc Brittain, Josh Bertram, Xuxi Yang, and Peng Wei. Prioritized sequence experience replay. *arXiv preprint arXiv:1905.12726*, 2019.

Xi Cao, Huaiyu Wan, Youfang Lin, and Sheng Han. High-value prioritized experience replay for off-policy reinforcement learning. In *2019 IEEE 31st international conference on tools with artificial intelligence (ICTAI)*, pp. 1510–1514. IEEE, 2019.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 380–388, 2002.

Fei Deng, Ingook Jang, and Sungjin Ahn. Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *International conference on machine learning*, pp. 4956–4975. PMLR, 2022.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in neural information processing systems*, 13, 2000.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019b.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Steven Hansen, Alexander Pritzel, Pablo Sprechmann, André Barreto, and Charles Blundell. Fast deep reinforcement learning using online adjustments from the past. *Advances in Neural Information Processing Systems*, 31, 2018.

Thommen George Karimpanal and Roland Bouffanais. Experience replay using transition sequences. *Frontiers in neurorobotics*, 12:32, 2018.

Isaac Kauvar, Chris Doyle, Linqi Zhou, and Nick Haber. Curious replay for model-based adaptation. *arXiv preprint arXiv:2306.15934*, 2023.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pp. 5639–5650. PMLR, 2020.

Jia-Hua Lee, Bor-Jiun Lin, Wei-Fang Sun, and Chun-Yi Lee. Edeline: Enhancing memory in diffusion-based world models via linear-time sequence modeling. *Advances in Neural Information Processing Systems*, 38:105587–105626, 2026.

Zhuo Li, Derui Zhu, Yujing Hu, Xiaofei Xie, Lei Ma, Yan Zheng, Yan Song, Yingfeng Chen, and Jianjun Zhao. Neural episodic control with state abstraction. *arXiv preprint arXiv:2301.11490*, 2023.

Ruei-Sung Lin, David A Ross, and Jay Yagnik. Spec hashing: Similarity preserving algorithm for entropy-based coding. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 848–854. IEEE, 2010.

Andrew W Moore and Christopher G Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine learning*, 13(1):103–130, 1993.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adria Puigdomenech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *International conference on machine learning*, pp. 2827–2836. PMLR, 2017.

Baturay Saglam, Furkan B Mutlu, Dogan C Cicek, and Suleyman S Kozat. Actor prioritized experience replay. *Journal of Artificial Intelligence Research*, 78:639–672, 2023.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International conference on machine learning*, pp. 9870–9879. PMLR, 2021.

Wenlong Wang, Ivana Dusparic, Yucheng Shi, Ke Zhang, and Vinny Cahill. Drama: Mamba-enabled model-based reinforcement learning is sample and parameter efficient. In *International Conference on Learning Representations*, volume 2025, pp. 75688–75712, 2025.

Tom Zahavy, Nir Ben-Zrihem, and Shie Mannor. Graying the black box: Understanding dqns. In *International conference on machine learning*, pp. 1899–1908. PMLR, 2016.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. Storm: Efficient stochastic transformer based world models for reinforcement learning. *Advances in Neural Information Processing Systems*, 36:27147–27166, 2023.

Guangxiang Zhu, Zichuan Lin, Guangwen Yang, and Chongjie Zhang. Episodic reinforcement learning with associative memory. In *International Conference on Learning Representations*, 2020.

APPENDIX CONTENTS

A Extended Related Work **14**
A.1 Latent World Models and Representation Learning 14
A.2 Episodic Memory and Retrieval-Augmented Reinforcement Learning 14
A.3 Priority Signals and Anchor Selection 15

B Anchor Offset and Context Alignment **16**

C Detailed Experimental Protocol and Hardware Assignment **17**
C.1 Wall-Clock Training Overhead 18

D Complete FLASH Procedure **19**

E Across-Seed Analysis of Exploitation in Frostbite **20**

F Complete Retrieval Group in Pong **22**

G Irreducible Probing Error due to RAM-Render Misalignment **25**

H Probing Accuracy of the FLASH Encoder **26**

I Controlled Igloo-Stage Evaluation: Protocol and Detailed Value Metrics **27**
I.1 Checkpoints and Shared Evaluation Contexts 27
I.2 Mean Correlations, Effect Sizes, and Bootstrap Uncertainty 27
I.3 Distributions and Matched-Context Changes 28
I.4 Variation Across Evaluation Episodes 29
I.5 Global Average Value Profiles 29
I.6 Interpretation and Scope 30

J Detailed Probing Experiments and Robustness Analysis **31**
J.1 Motivating Observation from Decoded Imagination 31
J.2 Preliminary Observation: The Cosine Similarity Clue 31
J.3 Probing the Latent Space 31
J.4 Background-Noise Strength Sweep with Separately Trained Probes 32

K Across-Seed Performance Variation in Atari 100k **37**

L Ablation Studies on Triggering and Retrieval **39**
L.1 Ablation 1: The Role of Neighbor Retrieval 39
L.2 Ablation 2: Value Difference vs. Absolute Value 39
L.3 Ablation 3: Multiplicative vs. Additive Combination 40

A Extended Related Work

This section expands the background and comparisons summarized in Section 2.

A.1 Latent World Models and Representation Learning

Latent world models encode visual observations into compact states and learn dynamics that generate imagined trajectories for policy and value learning. Dreamer (Hafner et al., 2019a) and DreamerV2 (Hafner et al., 2020) established Actor-Critic learning in latent imagination. STORM (Zhang et al., 2023), the primary base agent in our experiments, combines stochastic latent states with Transformer-based dynamics, while DRAMA (Wang et al., 2025) uses Mamba-based dynamics to improve sample and parameter efficiency.

A complementary line of work focuses on the representations used by these agents. Reconstruction-based world models learn visual representations jointly with generative objectives (Hafner et al., 2019b), while CURL (Laskin et al., 2020) introduces contrastive representation learning, SPR (Schwarzer et al., 2020) uses temporal predictive consistency, ATC (Stooke et al., 2021) decouples representation learning from policy optimization, and DreamerPro (Deng et al., 2022) adopts reconstruction-free prototypical representations. DIAMOND (Alonso et al., 2024) instead uses a diffusion-based world model to preserve complex visual details, emphasizing visual fidelity through a heavier generative formulation.

These approaches modify the representation-learning objective, latent dynamics, or generative world-model formulation to improve the states available for downstream control. FLASH targets a different intervention point. It retains the base agent's encoder and world-model architectures and instead changes which previously encountered historical contexts initialize Actor-Critic imagination. Thus, FLASH is complementary to methods that improve the representation or world model: rather than explicitly redesigning representation learning, it aims to make downstream learning better use of the representations and experiences already available to the base agent. A controlled Frostbite analysis further shows stronger average stage-value ordering in the FLASH critic for one matched checkpoint pair (Appendix I), providing a complementary downstream diagnostic of how task-relevant variation is reflected in the learned critic.

A.2 Episodic Memory and Retrieval-Augmented Reinforcement Learning

Model-Free Episodic Control (MFEC; Blundell et al., 2016) stores high observed returns associated with state-action pairs and uses nearest-neighbor lookup for value estimation. Neural Episodic Control (NEC; Pritzel et al., 2017) combines learned state embeddings with a differentiable neural dictionary to enable rapid value estimation from episodic experience. EVA (Hansen et al., 2018) retrieves related historical trajectories and uses them to adjust and augment parametric value estimates. ERLAM (Zhu et al., 2020) instead organizes episodic experience through an associative graph that propagates information across related trajectories, while subsequent work incorporates state abstraction into neural episodic control (Li et al., 2023).

The distinction relevant to FLASH is not merely whether past experience is retrieved, but where the retrieved information enters the learning pipeline. MFEC and NEC use episodic memories directly in value estimation, and EVA explicitly uses retrieved trajectories to augment value estimates. FLASH instead retrieves replay contexts around selected anchors and uses those contexts as initial conditions for imagined trajectories. The resulting imagined trajectories are optimized using the base agent's existing Actor-Critic objectives; retrieved returns or value estimates are not directly substituted into the critic target. Retrieval therefore affects learning through the composition of the Actor-Critic imagination batch.

FLASH also differs in its retrieval mechanism. Rather than performing a full nearest-neighbor search over the replay memory or constructing an explicit associative graph, it maps latent representations to discrete hash buckets and restricts candidate retrieval to matching buckets. Because the world-model representation evolves during training, stored bucket assignments can become stale. FLASH lazily re-encodes retrieved candidates for verification and triggers a Global Rebuild when the resulting hit rate falls below a threshold, thereby adapting the retrieval index to representation drift as described in Section 4.2.

A.3 Priority Signals and Anchor Selection

Prioritized Experience Replay (PER; Schaul et al., 2015) uses TD-error magnitude to prioritize replayed transitions. Subsequent approaches incorporate additional signals, including value-related criteria (Cao et al., 2019), model-based curiosity (Kauvar et al., 2023), sequence-level priorities (Karimpanal & Bouffanais, 2018; Brittain et al., 2019), and actor-critic-specific criteria (Saglam et al., 2023). Prioritized Sweeping (Moore & Atkeson, 1993) also uses value-related information, but schedules backups according to changes in value estimates induced by learning updates.

FLASH's anchor-selection signal differs along two dimensions. First, it uses the signed temporal value difference between consecutive states, $\Delta V_t = V(s_t) - V(s_{t-1})$. This should be distinguished from the value-change signal in Prioritized Sweeping: ΔV_t compares the critic's estimates of two temporally consecutive states along a trajectory, whereas Prioritized Sweeping considers how the estimate associated with a state changes as a consequence of an update. FLASH therefore uses temporal variation in estimated value as a signal for identifying salient locations in an experienced trajectory.

Second, FLASH combines normalized TD-error and signed temporal value difference multiplicatively through ReLU and softplus, rather than using either quantity alone or combining them additively as in value-aware replay approaches such as HVPER (Cao et al., 2019). The ReLU term gives a zero score when the normalized TD-error is non-positive, so a positive anchor score requires above-average TD-error. The softplus term then modulates this score according to the signed temporal value difference while remaining positive. The exact formulation and normalization procedure are given in Section 4.1.

Finally, the resulting score has a different role from a conventional replay priority. FLASH does not use it directly as a sampling probability or replay frequency. Instead, the score determines which experiences become anchors; each triggered anchor then initiates retrieval of structurally similar historical contexts, which are jointly incorporated into the Actor-Critic imagination batch. Appendix L evaluates the contributions of neighbor retrieval and alternative anchor-selection formulations.

B Anchor Offset and Context Alignment

We use an anchor offset of $\Delta_{\text{anchor}} = -2$. The offset is applied directly to the replay position of the selected salience score: if t^* maximizes the score in a sampled sequence, the registered pointer is p_{t^*-2}. Circular-buffer arithmetic is used for storage, and retrieved contexts must still contain valid replay entries without crossing an episode boundary.

For an anchor at position p, a length-L context contains the observations and recorded actions at positions $p - L + 1, \ldots, p$, including the anchor. With $L = 8$ and $p = t^* - 2$, this is the interval $t^* - 9, \ldots, t^* - 2$. After processing the recorded action at p, the world model predicts the next latent state, and the imagined policy chooses actions starting at $p + 1 = t^* - 1$. This alignment is shared by STORM's Transformer and DRAMA's Mamba implementations.

The offset places the imagination start before the selected high-salience state, so policy training can include preceding decisions. It is an implementation choice, not a guarantee that an action exactly two steps earlier caused the salience event; the architecture alone does not establish such a causal delay.

C Detailed Experimental Protocol and Hardware Assignment

Training and evaluation protocol. We follow the Atari 100k protocol in Section 5.1: 26 games, a training budget of 100,000 environment steps (400,000 emulator frames with action repeat 4), and four training seeds per game for the main comparison. The final checkpoint is evaluated for 20 episodes in STORM and 10 episodes in DRAMA. We average episode returns within each run before averaging over training seeds. Mean and median HNS are computed across games using the random-policy and human references in Table 2. Diagnostic and ablation experiments report their own seed coverage and aggregation rules in their respective appendices.

FLASH representation for hashing. For both STORM and DRAMA, f_θ consists of the world-model observation encoder and posterior head. Given the posterior logits $\ell_t \in \mathbb{R}^{K\times C}$ before uniform mixing, both implementations use

$$z_t = \mathrm{vec}(0.99\ \mathrm{softmax}(\ell_t) + 0.01/C)\,, \qquad K = C = 32, \quad d = 1024, \tag{9}$$

where softmax acts over the C categories of each of the K categorical variables. In code, this is the flattened posterior returned by `encode_obs` with `sample_mode=probs`. DRAMA's Mamba hidden state is not part of the hash input. Neither implementation samples a one-hot state for hashing; insertion, candidate re-verification, and Global Rebuild use the same posterior-probability representation.

Retrieval warmup and index initialization. The retrieval warmup target is $T_{\mathrm{warm}} = 50{,}000$ environment steps. Base-agent initialization is included within this interval. We initialize STORM from 1,024 replay steps and DRAMA from 1,032 steps. During retrieval warmup, FLASH accumulates hash entries and EMA statistics without registering or retrieving anchors. STORM's fixed-warmup path schedules a Global Rebuild at the last policy-training step before activation. For shared-warmup experiments, Baseline and FLASH resume from the same warmup checkpoint at the first episode boundary at or after the target. STORM rebuilds the index at this split before activating retrieval. DRAMA uses the hit-rate condition after activation. Subsequent rebuilds use the hit-rate threshold and cooldown in Table 4.

Experimental configuration. Table 4 summarizes the FLASH hyperparameters.

Table 4: FLASH hyperparameters for STORM and DRAMA. The target group size includes the anchor.

Parameter	Value
Retrieval warmup target T_{warm}	50,000 environment steps
Multiplicative salience threshold τ_z	3.5
EMA update rate α; numerical constant ϵ	0.01; 10^{-8}
Initial EMA mean; variance	0; 1
Hash input dimension d; hash bits B	1,024; 10
PCA sample cap per rebuild	100,000
Maximum anchors per imagination batch M	16
Target group size n; candidate multiplier m	16; 16
Maximum retrieved contexts	256
Imagination batch size N_{total}	1,024
Context length L; imagination horizon	8; 16
Anchor offset Δ_{anchor}	-2
Configured anchor weight	0.0625
Rebuild hit-rate threshold η	0.06
Rebuild cooldown	2,000 training-loop steps

Both configurations use the `retrieved` batch-reduction mode to replace random contexts one for one, so at most $Mn = 256$ retrieved contexts occupy the 1,024-context imagination batch. The configured anchor weight assigns mass 0.0625 to the anchor and shares the remaining mass equally among its neighbors; a singleton anchor receives weight one. For a full 16-member group, the configured weights are uniform at $1/16$ per member. Smaller groups retain unit total weight, consistent with Section 4.3, but need not have equal member weights in the implementation.

Table 5: Wall-clock training time on Atari 100k for STORM and DRAMA, with and without FLASH.

Agent / GPU	Baseline (min)	+FLASH (min)	Extra time (min)	Overhead (%)
STORM / RTX A6000	297	319	22	7.41
STORM / RTX PRO 6000	159	170	11	6.92
STORM / RTX 3090	411	432	21	5.11
DRAMA / RTX 3090	301	354	53	17.61

Environment and hardware. Both implementations use Gymnasium/ALE observations resized to 64×64 RGB pixels and PyTorch world models, with a replay capacity of 100,000 transitions. The world-model training batch contains 16 sequences, of length 64 in STORM and 128 in DRAMA. The timing comparison below pairs STORM and STORM+FLASH on each of an NVIDIA RTX A6000, an NVIDIA RTX PRO 6000, and an NVIDIA GeForce RTX 3090. We additionally compare DRAMA and DRAMA+FLASH on the NVIDIA GeForce RTX 3090.

C.1 Wall-Clock Training Overhead

FLASH retains the base model's encoder architecture while introducing additional computation for hash indexing, anchor selection, retrieval, and memory maintenance. To measure the practical training overhead, we compare the wall-clock training time of each baseline and its FLASH-augmented counterpart over the full 100k-step training budget, including retrieval warmup. For STORM, FLASH increases training time by 5.11–7.41% across three evaluated GPUs, corresponding to 11–22 additional minutes. For DRAMA, FLASH increases training time from 301 to 354 minutes on an NVIDIA GeForce RTX 3090, corresponding to an additional 53 minutes (17.61%). The detailed measurements are reported in Table 5.

D COMPLETE FLASH PROCEDURE

Algorithm 1 uses the notation of Section 4. World-model and Actor-Critic updates follow the base agent's update schedule.

Here, $\mathrm{Sample}_{\le k}(X)$ samples up to k available members of X without replacement, and $\mathrm{Take}_{\le k}$ retains up to k verified candidates. Retrieved contexts end at their associated replay pointers, following the indexing convention in Appendix B. The algorithm uses the group-weighted Actor-Critic objective in Eq. equation 8; Appendix C describes the configured implementation weights.

Algorithm 1 FLASH: Fast Latent Anchoring via Salience and Hashing

Require: Base agent, replay buffer $\mathcal{R}$, parameters with $Mn \le N_{\text{total}}$
Require: Initialization horizon T_{init}, budget T, retrieval activation T_{warm}
$\mathcal{A} \leftarrow \emptyset$, $\mathcal{B}_k \leftarrow \emptyset$ for all k; initialize EMA statistics
$\mu \leftarrow \mathbf{0}$; sample entries of W_{proj} independently from $\mathcal{N}(0,1)$
Collect T_{init} transitions into $\mathcal{R}$ using the base initialization procedure
for $t = T_{\text{init}}, \ldots, T-1$ **do**
  Collect (o_t, a_t, r_t, d_t); store at (p_t, e) and index by $h(f_\theta(o_t))$
  Sample a replay batch $\mathcal{D}$ and update the world model f_θ
  Compute δ_u^{TD}, ΔV_u, and S_u using the preceding EMA statistics (Eqs. equation 1–equation 4); then update the EMA
  if $t \ge T_{\text{warm}}$ **then**
    for all sequences $q \in \mathcal{D}$ with $\max_{u \in q} S_u > \tau_z$ **do**
      $u^* \leftarrow \mathrm{argmax}_{u \in q} S_u$; append $(p_{u^* + \Delta_{\text{anchor}}}, e)$ to $\mathcal{A}$ if indexed
    end for
  end if
  $\mathcal{P} \leftarrow \emptyset$; initialize valid retrieved groups $\mathcal{G} \leftarrow \emptyset$
  if $t \ge T_{\text{warm}}$ **then**
    $\mathcal{P} \leftarrow \mathrm{Pop}_{\le M}(\mathcal{A})$ in FIFO order
  end if
  for all valid anchors $a \in \mathcal{P}$ with registered key k_a **do**
    $\mathcal{C}_a \leftarrow \mathrm{Sample}_{\le m(n-1)}(\mathcal{B}_{k_a} \setminus \{a\})$
    Discard candidates without a valid length-L context
    $\mathcal{V}_a \leftarrow \{j \in \mathcal{C}_a : h(f_\theta(o_j)) = k_a\}$; record verification hit rate
    $\mathcal{N}_a \leftarrow \{a\} \cup \mathrm{Take}_{\le n-1}(\mathcal{V}_a)$; append $\mathcal{N}_a$ to $\mathcal{G}$
  end for
  $N_{\text{ret}} \leftarrow \sum_{\mathcal{N}_a \in \mathcal{G}} |\mathcal{N}_a|$; $N_{\text{rand}} \leftarrow N_{\text{total}} - N_{\text{ret}}$
  Form $\mathcal{D}_{\text{imagine}}$ from retrieved contexts and N_{rand} random contexts (Eq. equation 6)
  Set $w_i = 1$ for random contexts and assign member weights within each $\mathcal{N}_a$ to sum to one
  Generate imagined trajectories; update Actor-Critic using Eq. equation 8, with $W_{\text{sum}} = N_{\text{rand}} + |\mathcal{G}|$
  if scheduled index initialization is due, or retrieval is active, mean hit rate $< \eta$, and cooldown has elapsed **then**
    Re-encode $\mathcal{R}$, refit μ and W_{proj} by PCA, and rebuild all $\mathcal{B}_k$
  end if
end for

The hit rate averages the verified fraction over queried groups with candidates available for re-encoding; the implementation returns zero when there are no such candidates. Model-specific initialization and activation timing are described in Appendix C. If no valid group is retrieved, $N_{\text{ret}} = 0$, and the imagination batch consists entirely of randomly sampled contexts. In all cases, $N_{\text{ret}} \le Mn \le N_{\text{total}}$, so the batch contains exactly N_{total} contexts.

E ACROSS-SEED ANALYSIS OF EXPLOITATION IN FROSTBITE

This appendix provides the detailed Frostbite across-seed analysis summarized in Section 5.2. We use five matched training seeds (10, 710, 3710, 6000, and 6010), selecting the most recently created finished run with an evaluation for each seed and method, without ranking runs by their scores. STORM baselines are eligible regardless of creation date. The FLASH runs satisfy the fixed retrieval configuration: 50,000-step warmup, batch-size reduction by retrieved contexts, salience threshold 3.5, target group size 16. Each selected run was trained for 100k environment steps and evaluated for 20 episodes at its final checkpoint. We verify the individual evaluation returns against the logged mean. These are 20 evaluation episodes per trained model, not 20 independent training runs.

Success and analysis groups. We use episode return ≥ 800 as a score-based success criterion motivated by the approximately 800-point return associated with the first igloo completion in our observations. This is a proxy for successful behavior, rather than a direct detection of igloo completion. Low-return outcomes are defined separately by return ≤ 300. Both thresholds are applied identically to both methods. We report all five seed pairs and, separately, the four pairs in which both methods encountered success before retrieval activation. This subgroup is defined using pre-activation training experience, rather than final evaluation performance. Seed 6010 never reached the success threshold under either method in the recorded training episodes and remains in the full-sample results.

Table 6: Frostbite final mean evaluation returns and outcome rates for all five training-seed pairs. Each success or low-return rate uses 20 evaluation episodes, so 5 percentage points correspond to one episode. "Pre-50k" indicates whether both methods encountered a training episode with return ≥ 800 before retrieval activation. All checkpoints were evaluated after 100k training environment steps.

	Evaluation return		Success (%)		Low return (%)		
Seed	STORM	+FLASH	STORM	+FLASH	STORM	+FLASH	Pre-50k
10	2068.0	2206.0	95.0	90.0	5.0	10.0	Yes
710	2270.5	2601.0	100.0	95.0	0.0	5.0	Yes
3710	1904.0	2233.5	70.0	90.0	30.0	10.0	Yes
6000	2408.0	2570.5	95.0	100.0	5.0	0.0	Yes
6010	254.5	258.5	0.0	0.0	100.0	100.0	No

In Frostbite, we observe two distinct outcomes across five matched training-seed pairs (Table 6). In four pairs, both agents encountered successful behavior before retrieval activation, and FLASH achieved higher final mean evaluation returns in every pair. In the remaining pair, neither agent encountered successful behavior, and both achieved similarly low final returns. These results suggest that FLASH improves the utilization of previously encountered successful experiences rather than independently enabling the discovery of new successful behavior.

Aggregation and uncertainty. We first summarize the 20 evaluation episodes within each training run, then give each training seed equal weight. Table 7 reports differences between FLASH and STORM with percentile bootstrap 95% confidence intervals obtained from 10,000 resamples of matched training-seed pairs (bootstrap seed 2027). The resampling unit is the training-seed pair; evaluation episodes are not treated as independent training runs. With only five pairs overall and four in the pre-activation-success subgroup, these intervals are exploratory estimates of uncertainty.

Training-evaluation gap and final success reproduction. For each training seed i, we compute $G_i = R_i^{\text{train,best}} - \bar{R}_i^{\text{eval}}$, using the maximum recorded training episode return and the final checkpoint's mean evaluation return. The average gap decreases in both analysis groups (Table 7). This is a descriptive statistic: the training maximum and evaluation mean differ in selection, and a lower training maximum also reduces the gap. Indeed, the mean best training return is lower under FLASH in these runs, so the gap reduction alone cannot establish more effective exploitation.

Table 7: Aggregate Frostbite results. Differences are FLASH minus STORM; rate differences and their confidence intervals are in percentage points (pp). The gap is the maximum recorded training episode return minus the final mean evaluation return. Confidence intervals resample matched training-seed pairs.

Metric	STORM	+FLASH	Difference	95% CI for difference
All five training-seed pairs				
Mean evaluation return	1781.0	1973.9	$+192.9$	$[89.3, 296.5]$
Mean best training return	3152.0	2912.0	-240.0	$[-520.0, 40.0]$
Mean training-evaluation gap	1371.0	938.1	-432.9	$[-773.1, -92.7]$
Evaluation success (%)	72.00	75.00	$+3.00$ pp	$[-4.00, 12.00]$ pp
Low-return outcomes (%)	28.00	25.00	-3.00 pp	$[-12.00, 4.00]$ pp
Four pairs with success in both methods before retrieval activation				
Mean evaluation return	2162.6	2402.8	$+240.1$	$[150.3, 330.0]$
Mean best training return	3870.0	3570.0	-300.0	$[-580.0, 37.5]$
Mean training-evaluation gap	1707.4	1167.3	-540.1	$[-842.3, -118.9]$
Evaluation success (%)	90.00	93.75	$+3.75$ pp	$[-5.00, 13.75]$ pp
Low-return outcomes (%)	10.00	6.25	-3.75 pp	$[-13.75, 5.00]$ pp

The more consistent observation is that all four seed pairs with pre-activation success have higher final mean evaluation returns under FLASH. The reduction in low-return outcomes is less consistent: counts change from 1 to 2, 0 to 1, 6 to 2, and 1 to 0 for seeds 10, 710, 3710, and 6000, respectively. Thus, the net reduction from 8 to 5 low-return episodes is driven mainly by seed 3710. This supports reporting higher final returns after successful experience has been encountered, while limiting claims about uniform improvements in success reproduction or failure avoidance.

First success and training-based consolidation. We identify first success from completed training episodes. As a supplementary measure, we define training-based consolidation as the end of the first complete window of 10 training episodes, starting no earlier than the first successful episode, in which at least 8 episodes meet the success threshold. Elapsed time is measured from first success to that window's end. Environment steps are reconstructed from episode lengths and checked against replay-buffer lengths, rather than equating the W&B logging index with environment steps.

Table 8 shows that first success and this consolidation criterion occur at identical steps in the two methods for each of the four successful seed pairs. Both events precede retrieval activation. Consequently, this particular consolidation measure does not demonstrate a speedup attributable to FLASH. Seed 6010 never succeeds, so its consolidation time is undefined. Successful runs that did not consolidate before training ended would be treated as right-censored; none occur in this sample. These retrospective runs contain only final checkpoint evaluations, so they do not support measuring consolidation of intermediate evaluation performance.

Table 8: Learning-event timings in Frostbite, in environment steps. For each listed seed, STORM and STORM+FLASH have identical first-success and training-consolidation timings, so a single value is shown for both methods. All observed events precede FLASH activation at 50,000 steps. Dashes denote no successful training episode and hence an undefined consolidation time.

Seed	First success	Training consolidation	Steps from first success
10	17,172	29,169	11,997
710	14,036	40,916	26,880
3710	14,173	21,448	7,275
6000	13,206	38,545	25,339
6010	—	—	—

F Complete Retrieval Group in Pong

Figure 4 presents the complete retrieval group for the scoring-related Pong anchor analyzed in Section 5.2. The group consists of one anchor and 15 retrieved neighbors. The retrieved observations exhibit varying degrees of visual similarity to the anchor and include ordinary rally situations as well as situations with different subsequent outcomes. Among these neighbors, Neighbor 06 leads to a point, whereas Neighbor 08 shows a near-miss in which the ball contacts the opponent's paddle and reverses direction. These two neighbors were selected for the main-paper comparison because they illustrate how experiences associated with similar game situations can lead to different outcomes. The remaining neighbors exhibit other game configurations and trajectories, indicating that the retrieved group is not limited to scoring-related situations or near-misses. All displayed trajectories are actual replay experiences rather than imagined rollouts.



Figure 6: Complete retrieval group for the scoring-related Pong anchor (upper portion). Blue boxes mark the retrieved observations at $t = 0$; actions and rewards correspond to the outgoing transitions from each frame.

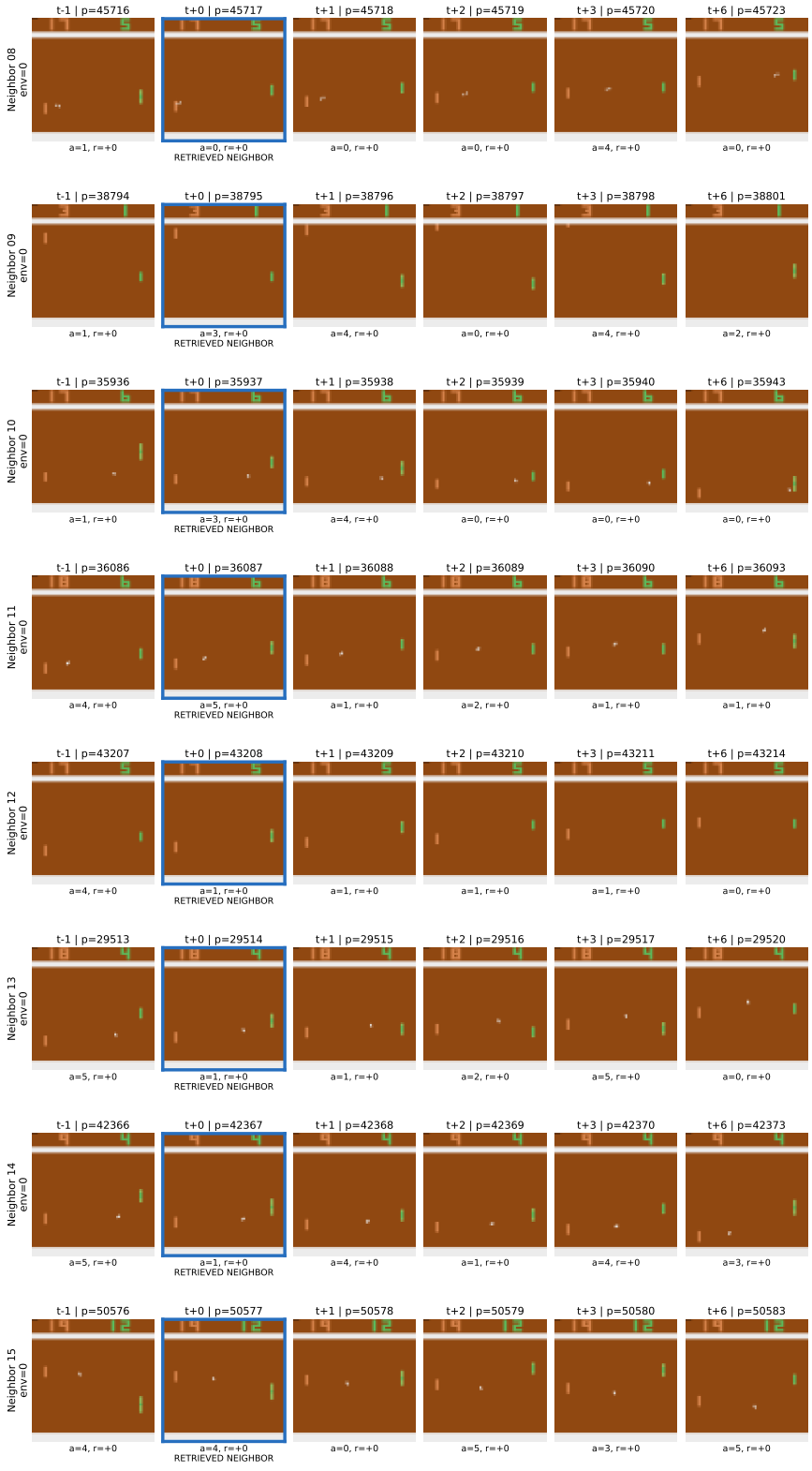
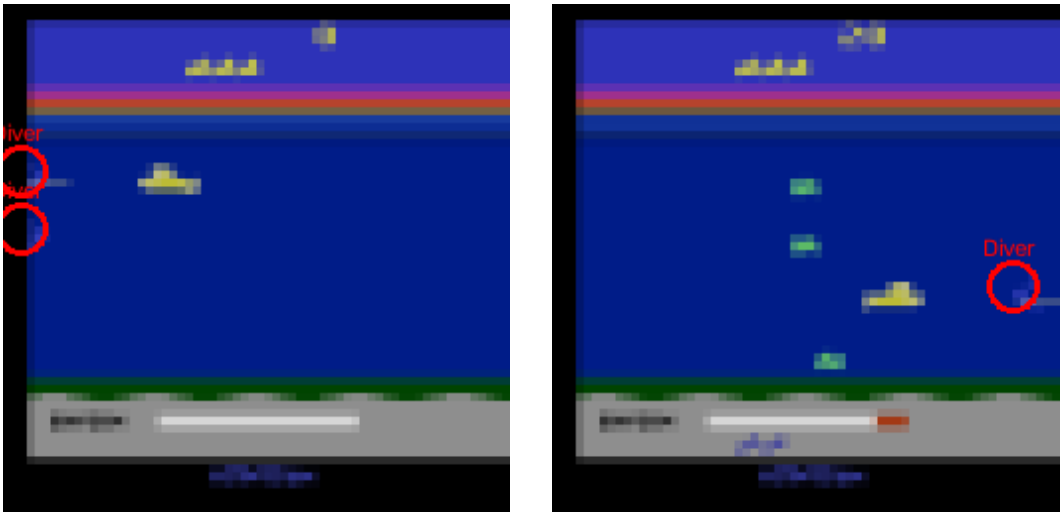


Figure 7: Complete retrieval group for the scoring-related Pong anchor (lower portion). Continuation of Figure 6, showing the remaining retrieved neighbors.

G IRREDUCIBLE PROBING ERROR DUE TO RAM-RENDER MISALIGNMENT



(a) Boundary emergence event.

(b) Spatial offset in RAM coordinate.

Figure 8: Visualizing the RAM-render misalignment. (a) A diver is registered in the RAM state as soon as it touches the screen boundary, even though only a negligible fraction of its sprite is visible at the edge (`frame_0040.png`). (b) The center of the RAM coordinate (red circle) does not perfectly align with the visual center of the diver sprite, demonstrating the intrinsic spatial misalignment (`frame_0120.png`).

While our probing MLP achieves a pixel distance error of 12.44 pixels on the original 160×210 Atari screen space, it is important to contextualize this value. The world model's encoder processes observations that have been downsampled to 64×64 pixels. In this resized 64×64 space, a 12.44 pixel distance on the original screen corresponds to a deviation of less than 5 pixels (≈ 4.98 pixels, based on the $64/160$ width scaling factor).

Furthermore, this remaining error is largely an irreducible artifact of the Atari 2600 emulation itself. There is a known spatial misalignment between the underlying RAM state coordinates and the exact pixel location where the sprites are rendered on the screen. For example, the RAM representation of a diver's position often exhibits minor positional offsets compared to its actual rendered visual center.

Additionally, this hardware-level asynchrony provides a strong justification for the 92.4% presence prediction accuracy. The Atari RAM strictly registers a diver as "present" the exact moment its coordinate enters the screen boundary, even though visually only an almost imperceptible fraction of the sprite is drawn at the extreme left or right edges. Conversely, when the submarine collects a diver, the diver's entity is immediately wiped from the RAM state; yet, its visual sprite typically persists on the screen for at least one additional frame, heavily overlapping with the submarine before fully disappearing. These edge-case boundary entering and collision events create a persistent temporal mismatch between what is rendered and what the RAM dictates, accounting for the remaining gap in presence accuracy.

H PROBING ACCURACY OF THE FLASH ENCODER

We also retain an earlier two-condition probing experiment on the FLASH encoder. The encoder was frozen, and separate MLPs were trained on clean observations and observations with complete background replacement by uniform noise. Each MLP was validated and evaluated under its own training corruption condition. These are separate runs from the baseline noise-strength sweep in Appendix J.4.

As shown in Table 9, the clean-trained FLASH probe achieves 99.08% full-dataset accuracy, while the separately noise-trained probe achieves 27.18%; their best validation accuracies are 97.03% and 25.41%. These runs use the 100,000-step checkpoint from `Frostbite-life_done-wm_2L512D8H-100k-seed3710_O` and the 1,854-frame dataset. The baseline endpoint runs attained 99.19% and 27.72%. The broadly similar endpoint pattern provides no evidence here for improved decoding under background noise with FLASH, but the absence of a FLASH strength sweep precludes a controlled estimate of an encoder-robustness difference. This diagnostic also does not establish the mechanism behind the value-function results in Section I.

Table 9: Earlier FLASH-encoder endpoint experiment in Frostbite. Clean and noisy columns come from separately trained MLPs; noise is present during both training and evaluation of the noisy probe. All entries use the full dataset, including probe-training frames, and are separate from the baseline strength sweep.

Igloo Stage	**Samples** (n)	**Clean Accuracy (%)**	**Noisy Accuracy (%)**
0	153	100.0	0.0
1	26	88.5	0.0
2	25	88.0	0.0
3	28	96.4	0.0
4	77	98.7	0.0
5	54	98.1	0.0
6	159	100.0	0.6
7	377	99.5	77.5
8	95	98.9	0.0
9	91	98.9	0.0
10	121	100.0	0.0
11	120	100.0	0.0
12	47	97.9	0.0
13	89	98.9	0.0
14	28	92.9	0.0
15	70	100.0	0.0
16	294	100.0	71.8
Overall	**1,854**	**99.1**	**27.2**

I CONTROLLED IGLOO-STAGE EVALUATION: PROTOCOL AND DETAILED VALUE METRICS

This appendix presents a controlled igloo-stage analysis of the critic's responses to task-relevant visual changes in Frostbite. We examine how predicted values vary with igloo construction stage and report both the average effect and its variation across observation-action contexts. This analysis uses one checkpoint pair and is separate from the across-training-seed evaluation-return analysis in Appendix E.

I.1 CHECKPOINTS AND SHARED EVALUATION CONTEXTS

We evaluate one Baseline-FLASH pair trained with the same random seed, using the world models and critics saved at 100,000 environment steps. Both models share the encoder architecture and remain fixed during evaluation.

We collect 20,000 observations and actions once using the Baseline policy in Frostbite, with action repeat 4 and observations resized to 64×64 pixels. This count refers to agent-step observations, not raw emulator frames. The dataset covers 19 completed episodes and part of a twentieth episode. Both models receive the same collected observations and action histories, so their predictions are compared on matched contexts drawn from the Baseline policy.

Context construction and stage intervention. We use RAM-derived igloo stages to divide the data into consecutive blocks with unchanged stage and a maximum length of 64 observations. Blocks do not cross episode boundaries, and those shorter than nine observations are excluded, leaving 320 valid blocks. For each block, we replace the igloo region in every observation with each of 17 stage templates (0–16). All pixels outside the igloo region and all actions remain identical across variants. The background can change over time within a block, but is held fixed across stage variants at each time step. No random background noise is injected.

For each variant, the first eight observations initialize the temporal context; subsequent observations are evaluated using their encoded latents and preceding observation-action histories. This yields $N = 17{,}421$ evaluated contexts. Each context denotes one evaluated time point with its history, tested under all 17 stage variants. These variants yield one correlation per context, rather than 17 independent evaluation samples.

Latent sampling. Stochastic latent sampling uses common random seeds across models and stage variants within each block. Each context-stage prediction uses one latent draw; values are not averaged over repeated draws.

I.2 MEAN CORRELATIONS, EFFECT SIZES, AND BOOTSTRAP UNCERTAINTY

Let $v_i^{(m)}(k)$ be the predicted value for model $m \in \{\mathrm{B}, \mathrm{F}\}$ at context i under stage k. We calculate

$$\rho_i^{(m)} = \mathrm{Spearman}\Big((0, \ldots, 16),\, (v_i^{(m)}(0), \ldots, v_i^{(m)}(16))\Big)\,, \tag{10}$$

and analogously Kendall's τ_b. Tied predictions use the standard tie handling of these rank correlations. If a model predicts a constant value across all 17 stages, its correlation is undefined. A context with an undefined correlation in either model is excluded from the paired analysis for that metric, rather than assigned zero. No correlations were undefined in this dataset, so all 17,421 contexts enter both analyses.

For either correlation $c \in \{\rho, \tau_b\}$, the mean and the paired absolute effect size are

$$\bar{c}^{(m)} = \frac{1}{N}\sum_{i=1}^{N} c_i^{(m)}, \qquad \overline{\Delta c} = \frac{1}{N}\sum_{i=1}^{N}\left(c_i^{(\mathrm{F})} - c_i^{(\mathrm{B})}\right). \tag{11}$$

Each evaluated context has equal weight in these point estimates. We estimate uncertainty using 10,000 paired episode-bootstrap replicates. In each replicate, we draw 20 episode identifiers with replacement and include all evaluated contexts from each selected episode, preserving the Baseline-FLASH pairing and the multiplicity of repeated episodes. We recompute the mean using the number

Table 10: Mean per-context correlations and paired absolute effect sizes for the evaluated checkpoint pair. Brackets give paired episode-bootstrap percentile 95% CIs from 10,000 replicates over 20 observed episodes (19 complete and one partial).

Statistic	Spearman's ρ	Kendall's τ_b
Baseline (STORM)	0.0756 $[0.0447, 0.1097]$	0.0397 $[0.0143, 0.0676]$
FLASH	0.1622 $[0.1269, 0.1988]$	0.1343 $[0.1063, 0.1634]$
FLASH − Baseline	0.0866 $[0.0506, 0.1213]$	0.0946 $[0.0670, 0.1213]$

Table 11: Distributions across 17,421 matched contexts. Q1 and Q3 are the 25th and 75th percentiles; IQR is Q3−Q1. Difference rows use per-context FLASH-minus-Baseline differences.

Metric	Distribution	Median	Q1	Q3	IQR
ρ	Baseline	0.1281	-0.4297	0.5857	1.0154
ρ	FLASH	0.2240	-0.3051	0.7077	1.0128
ρ	Difference	0.0490	-0.1886	0.3744	0.5630
τ_b	Baseline	0.0517	-0.3321	0.4118	0.7439
τ_b	FLASH	0.1637	-0.2404	0.5469	0.7873
τ_b	Difference	0.0701	-0.1715	0.3529	0.5244

of contexts in that replicate as the denominator, thereby accounting for unequal episode lengths. The 2.5th and 97.5th percentiles give the 95% confidence intervals. Resampling whole episodes keeps the observed within-episode temporal dependence intact instead of treating adjacent frames as independent observations.

Table 10 shows positive mean differences for both metrics: $\overline{\Delta\rho} = 0.0866$ and $\overline{\Delta\tau_b} = 0.0946$. Both difference intervals lie above zero, providing evidence of a positive mean effect under this episode-resampling analysis. The FLASH mean Spearman correlation is approximately 2.15 times the Baseline mean, but its absolute value (0.1622) remains weak. The absolute effect and its interval are therefore more informative than the relative increase alone. These intervals describe uncertainty over sampled evaluation episodes for two fixed checkpoints. They do not include variation across training seeds or repeated latent draws, and the 17,421 contexts are not independent training runs.

I.3 Distributions and Matched-Context Changes

Table 11 reports the median, quartiles, and interquartile range (IQR) of each model's per-context correlations and of their paired differences. The median paired difference is computed after subtracting values within each matched context; it need not equal the difference between the two models' separate medians. Quartiles describe variation across contexts, whereas the confidence intervals in Table 10 describe uncertainty in a mean.

Both medians increase, from 0.1281 to 0.2240 for Spearman and from 0.0517 to 0.1637 for Kendall. The median paired changes are also positive (0.0490 and 0.0701). Nevertheless, the distributions remain broad: both models' first quartiles are negative, and the middle 50% of their paired differences includes both negative and positive values. The Spearman IQR is approximately unchanged across models (1.0154 versus 1.0128), while the Kendall IQR increases (0.7439 versus 0.7873). Thus, a higher average does not imply uniformly improved ordering or a narrower spread across contexts.

FLASH has a higher Spearman correlation in 9,686 contexts (55.60%), a lower correlation in 7,733 (44.39%), and an equal correlation in two. The corresponding Kendall counts are 10,001 (57.41%), 7,392 (42.43%), and 28. Among contexts with a positive Spearman change, the mean increase is 0.4289; among those with a negative change, the mean decrease has magnitude 0.3422. The corresponding Kendall magnitudes are 0.3767 and 0.2867. Both the frequency and the conditional magnitude of positive changes therefore contribute to the positive overall means. These observations support improvement extending beyond a few isolated positive contexts, while also documenting substantial negative changes.

Table 12: Direction and magnitude of changes within matched contexts. Conditional mean changes average only the contexts with positive or negative change, respectively; all differences are FLASH minus Baseline.

Statistic	Spearman's ρ	Kendall's τ_b
Higher under FLASH	9,686 (55.60%)	10,001 (57.41%)
Lower under FLASH	7,733 (44.39%)	7,392 (42.43%)
Equal	2 (0.01%)	28 (0.16%)
Mean change where positive	$+0.4289$	$+0.3767$
Mean change where negative	-0.3422	-0.2867

Table 13: Paired mean correlation differences by evaluation episode. The final episode is partially observed. Positive differences denote higher average correlation under FLASH.

Episode	Evaluated contexts	Mean $\Delta\rho$	Mean $\Delta\tau_b$
1	783	-0.0006	$+0.0320$
2	1,048	-0.0292	$+0.0172$
3	957	$+0.0343$	$+0.0476$
4	1,010	$+0.0031$	$+0.0448$
5	861	$+0.1738$	$+0.1783$
6	1,008	$+0.1237$	$+0.1139$
7	907	$+0.2387$	$+0.2201$
8	793	$+0.1170$	$+0.1174$
9	896	$+0.1587$	$+0.1516$
10	890	$+0.1461$	$+0.1416$
11	896	$+0.0800$	$+0.1003$
12	952	-0.0569	-0.0224
13	956	$+0.0522$	$+0.0625$
14	444	$+0.1168$	$+0.0962$
15	848	$+0.1453$	$+0.1337$
16	929	$+0.1605$	$+0.1551$
17	712	$+0.0947$	$+0.0824$
18	857	$+0.1585$	$+0.1369$
19	1,010	$+0.0444$	$+0.0581$
20	664	$+0.0123$	$+0.0419$

I.4 Variation Across Evaluation Episodes

For each evaluation episode e, we average the paired differences over its n_e evaluated contexts. Table 13 reports these episode-specific means. The mean Spearman difference is positive in 17 of 20 episodes (85%), and the mean Kendall difference is positive in 19 of 20 (95%). The respective ranges are $[-0.0569,\ 0.2387]$ and $[-0.0224,\ 0.2201]$. Thus, the positive pooled effect occurs across multiple episodes, although individual episodes contain both positive and negative context-level changes. These counts are descriptive results for the same fixed checkpoint pair, not additional training-seed replications or separate significance tests. The pooled estimates in Table 10 weight contexts equally; they are not unweighted means of these 20 episode means.

I.5 Global Average Value Profiles

For completeness, we also average predicted values across all contexts at each stage, $\bar{v}^{(m)}(k) = N^{-1}\sum_i v_i^{(m)}(k)$, and then correlate this 17-point profile with stage index. This reverses the order of averaging and correlation relative to the per-context metric and answers a different question. Baseline and FLASH respectively obtain Spearman correlations of 0.9681 and 0.7819, and Kendall correlations of 0.8971 and 0.6765, for their global profiles. These are point estimates for the averaged profiles, not additional per-context means or bootstrap intervals.

Table 14 gives the complete profiles. The Baseline profile generally rises and reaches its maximum at stage 16, with some intermediate decreases. The FLASH profile peaks at stage 12, falls through stage 15, and rises again at stage 16. Thus, FLASH's higher mean per-context ordering coexists with

a lower correlation of its global average profile. Neither averaging procedure implies the result of the other. Differences in raw predicted-value magnitudes are descriptive and do not by themselves establish more accurate value estimates.

Table 14: Global average predicted value by igloo stage, using the same 17,421 contexts for both models. Bold entries mark each model's maximum.

Igloo stage	Baseline	FLASH
0	122.5811	120.9317
1	130.3185	134.1611
2	135.5643	143.4519
3	140.1353	147.9921
4	143.5138	153.3676
5	143.5928	161.1659
6	142.2332	164.6773
7	144.1948	169.2527
8	144.3742	171.7132
9	144.5918	175.1732
10	146.0023	177.5542
11	149.7183	180.4352
12	152.8047	**181.0360**
13	149.8857	174.6467
14	148.4644	169.3923
15	151.2220	165.3206
16	**158.5796**	174.8477

I.6 Interpretation and Scope

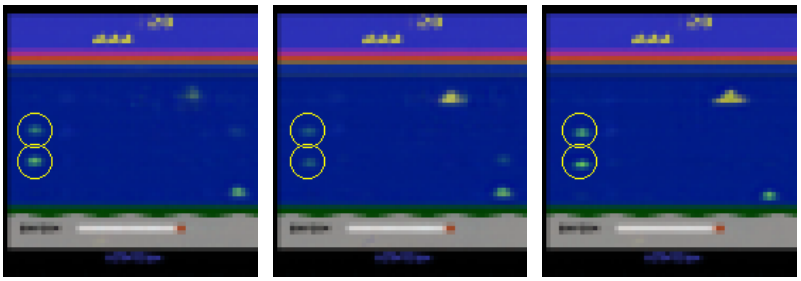
For this checkpoint pair on Baseline-collected contexts, FLASH exhibits higher mean and median stage-value correlations, positive paired mean differences with episode-bootstrap intervals above zero, and positive episode-mean changes in most sampled episodes. The absolute correlations nevertheless remain weak on average, and a substantial fraction of contexts show lower correlations under FLASH. These findings support an average improvement in ordering for the evaluated pair, rather than uniform robustness across backgrounds.

The diagnostic tests whether predicted values increase with igloo stage under a controlled visual intervention. It does not directly measure whether latent states are disentangled, and a negative correlation can still reflect sensitivity to stage changes. Moreover, a critic predicts future return under its policy; the experiment does not establish that true expected return must increase monotonically with stage in every context. Replacing image regions also does not guarantee that each modified observation-action sequence corresponds to a realizable environment trajectory. Consequently, these results alone neither establish that spatial entanglement has been removed nor identify the cause of the evaluation-return improvements discussed in Section 5.2. The background-noise probe classification experiment in Appendix H is a separate diagnostic and is not pooled with this value analysis.

J Detailed Probing Experiments and Robustness Analysis

This section provides the comprehensive experimental setup and extended results for the diagnostic analyses summarized in Section 3.

J.1 Motivating Observation from Decoded Imagination



(a) t (less blurry) (b) $t+1$ (more blurry) (c) $t+2$ (less blurry)

Figure 9: Consecutive decoded frames from an open-loop imagined trajectory. The fish (yellow circles) appears less blurry at t and $t+2$, and more blurry at $t+1$.

Imperfect visual reconstruction naturally raises the concern that a compact latent state may have discarded information needed for control. As an initial motivating observation, we inspect consecutive decoded frames from an open-loop imagined trajectory. Starting from an encoded observation, the world model predicts subsequent latent states using its learned transition dynamics and the imagined actions, without access to further real observations; each predicted latent state is then mapped back to image space using the world model's learned observation decoder. As shown in Figure 9, the highlighted fish appears less blurry at t, more blurry at $t+1$, and less blurry again at $t+2$. This non-monotonic change in decoded appearance motivated us to examine the latent representation directly rather than treating reconstruction fidelity as a direct measure of retained task-relevant information. Here, *blurry* refers only to decoded visual appearance, not to the information content of the latent state.

J.2 Preliminary Observation: The Cosine Similarity Clue

We construct controlled Seaquest observations in which the full visual observation is preserved except for either the number of collected divers or the position of a diver. Each complete observation, including the unchanged background, is passed through the encoder; the diver region is not encoded in isolation. We construct one set by varying the collected-diver count from 0 to 6 and another by moving a single diver while keeping the submarine and background fixed, and compare the resulting latent states using pairwise cosine similarity. As shown in Figure 1, cosine similarity decreases under these controlled target changes. For reference, consecutive latent states from ordinary trajectories have a mean cosine similarity of 0.997. Adjacent diver counts, which can differ across consecutive environment steps, have an average similarity of 0.949, while pairs differing by three or more counts average 0.893. Similarity also decreases with diver displacement, reaching values as low as 0.63. Relative to the high similarity observed under ordinary temporal variation, these patterns suggest that the controlled task-relevant changes induce appreciable changes in the latent representation. We use this analysis as a sensitivity test rather than evidence of semantic separation: whether the corresponding task variables can be recovered from the representation is evaluated directly through supervised probing in Appendix J.3.

J.3 Probing the Latent Space

Encoder and probing data. We test whether a lightweight MLP can predict diver counts, presence, and positions from the latent state z in Seaquest. The frozen encoder comes from a training run that never encountered states with four to six collected divers. These high-count states are relevant to completing the six-diver return behavior discussed in Section 3.2.

We construct the probing dataset from newly collected rollouts from the same run, independent of the encoder's original replay buffer, together with manually collected demonstrations containing four to six divers. The frozen encoder maps these observations to latents, and RAM states provide the supervised labels. The MLP is trained on examples spanning all counts from zero to six, including the high-count demonstrations. Thus, counts not encountered during encoder training are represented in the probe's supervised training data.

Collected-diver count. The clean count probe achieves 99.52% accuracy over all 59,077 frames, including its training frames, and 97.67% best validation accuracy. Over the 673 frames with counts four to six, its pooled full-dataset accuracy is 95.69% (644 correct). The individual accuracies for counts four, five, and six are 95.89%, 96.88%, and 90.91%, respectively. These results measure supervised decoding from a frozen encoder, not zero-shot generalization by the MLP to unseen labels. Dataset construction, splitting, and checkpoint selection are detailed in Appendix J.4.

Diver presence and position. Up to four divers can appear on screen simultaneously. To avoid assigning fixed output slots to particular lanes, the probe outputs an unordered set of four predictions, each comprising a presence probability and (x, y) coordinates. Hungarian matching associates predictions with ground-truth divers for the permutation-invariant matching loss.

The probe achieves 92.4% diver-presence accuracy and a mean position error of 12.44 pixels in the original 160×210 screen coordinates. This indicates that the probe can recover object presence and location on the evaluated data despite imperfect visual reconstruction. It does not establish complete information preservation or a disentangled representation.

RAM-render alignment and residual error. RAM-derived labels and rendered frames can be temporally or spatially misaligned. For example, the diver counter may update while a collected diver remains visible for an additional frame, and a sprite's rendered location may differ from its RAM coordinates. Appendix G illustrates such discrepancies.

These mismatches may contribute to residual probing errors, but the present experiments do not quantify their contribution. Information loss in the encoder and limitations of the probe remain possible additional sources of error.

J.4 Background-Noise Strength Sweep with Separately Trained Probes

High clean-data decodability does not reveal whether prediction relies on visual evidence specific to the target or on contextual cues that happen to co-vary with it in the observed data. For example, if an igloo stage generally increases as time passes within an episode, other visual factors that also change over time could serve as predictive proxies even if they do not directly reflect the igloo stage itself. Such proxy relationships may become unreliable when the target and contextual cues later vary differently from how they co-varied in previously encountered data. We therefore quantify whether a supervised probe can continue to decode a target whose own pixels remain intact while the surrounding visual context is progressively corrupted. For each game, the encoder is fixed across all seven noise strengths, while a new MLP is trained at each strength. Both probe-training and validation inputs have the same corruption strength as the final evaluation inputs. The quantitative sweep uses recorded observations without replacing igloo stages or diver counts; template-based stage replacement belongs to the separate critic analysis in Appendix I.

Data and labels. Frostbite uses 1,854 observations from `probing_data/Frostbite_20260822_201417.npz`, covering all 17 igloo stages. The label is 0 when RAM byte 77 equals 255, and otherwise equals that byte plus one. Seaquest combines 57,604 observations from `probing_data/Seaquest-life_done-wm_2L512D8H-100k-seed1_X_probing.npz` with 1,473 observations from `probing_data/Seaquest_20260822_133245.npz`, for 59,077 observations in total. RAM byte 62 gives the collected-diver count (0–6). The first Seaquest file contains counts 0–2; the additional file supplies examples spanning 0–6. Each observation is a 64×64 RGB image stored as unsigned 8-bit pixels, paired with its RAM state. The class counts are reported in Tables 16 and 17.

Frozen encoders and probe optimization. We use world-model checkpoints at 100,000 environment steps from `Frostbite-life_done-wm_2L512D8H-100k-seed3710` and `Seaquest-life_done-wm_2L512D8H-100k-seed1_X`. The world model is set to evaluation mode and its parameters are frozen. Latents are extracted with deterministic categorical mode selection (`sample_mode=mode`), yielding 1,024 input features. The probe is an MLP with widths $1024 \to 256 \to 64 \to K$, with ReLU after each hidden layer and $K = 17$ or 7. Each condition trains a fresh probe for 50 epochs using Adam, unweighted cross-entropy, and learning rate 0.005. Batch sizes are 256 for Frostbite and 4,096 for Seaquest. Probe initialization, minibatch shuffling, and data splitting use seed 42. The training budget and other hyperparameters are held fixed across strengths within each game. Training a separate probe at each corruption strength allows the probe to adapt to whatever latent coordinates remain predictive under that condition, rather than testing a clean-trained probe under a mismatched noisy input distribution.

Corruption operator and preserved regions. For original pixel $I_{p,c}$ at location p and color channel c, draw $U_{p,c}$ independently from the discrete uniform distribution on $\{0, \ldots, 255\}$. If B is the preserved bounding box, the input at strength α is

$$\widetilde{I}_{p,c}^{(\alpha)} = \begin{cases} I_{p,c}, & p \in B, \\ \mathrm{round}\left((1-\alpha)I_{p,c} + \alpha U_{p,c}\right), & p \notin B. \end{cases} \tag{12}$$

The implementation uses float32 mixing, NumPy rounding to the nearest integer, clipping to $[0, 255]$, and conversion back to unsigned 8-bit pixels; it then copies the original target pixels exactly. The resulting image is divided by 255 before encoding. Strength 0 leaves the observation unchanged, and strength 1 completely replaces pixels outside B with uniform noise. In zero-based input coordinates with exclusive upper bounds, B is $42 \le x < 58$, $10 \le y < 17$ in Frostbite (a 16×7 region), and $18 \le x < 46$, $53 \le y < 57$ in Seaquest (a 28×4 region). "Background" here denotes every pixel outside this box, including other objects and interface elements.

We evaluate $\alpha \in \{0, 0.1, 0.2, 0.4, 0.6, 0.8, 1\}$. A separate NumPy random-number generator with seed 42 produces the noise, preserving the same random pixel field for each observation across strengths without consuming the random state used for the data split. Corruption is applied once before latent extraction and train/validation splitting. Each frame therefore retains its sampled corruption throughout training and evaluation; noise is not resampled at every epoch. Original RAM labels are retained. Figure 2(a) illustrates the operator using separate display examples, rather than the full set of evaluated observations.

Splits, checkpoint selection, and metric definitions. A random permutation of frames assigns $\lfloor 0.2N \rfloor$ frames to validation and the remainder to training: 370/1,484 in Frostbite and 11,815/47,262 in Seaquest (validation/training). Within a game, the same split is used at every strength. The MLP checkpoint with the highest validation accuracy across the 50 epochs is selected. All per-class results and the reported overall and class-average curves are then computed on the *full dataset*, which includes the training frames. They are not held-out test accuracies. We separately report the best validation accuracy used for checkpoint selection; that validation set is also not an independent final test set. Splitting is by frame rather than by episode, so nearby frames from the same trajectory can occur in both partitions.

For class c, let N_c be its full-dataset sample count and C_c its number of correct predictions. Its class-conditional accuracy is $a_c = 100 C_c / N_c$. Overall accuracy is $100 \sum_c C_c / \sum_c N_c$, while mean class accuracy is the unweighted mean $K^{-1} \sum_c a_c$ (balanced accuracy). Thus, classes contribute in proportion to frequency to the overall metric, but equally to the class mean. There is one frozen encoder checkpoint per game and one probe-training run per strength, all with seed 42; no across-seed confidence intervals are estimated. The saved JSON files contain full-dataset class counts and accuracies, but not validation accuracies broken down by class.

Results at every noise strength. Figure 10 shows the overall-accuracy curves that complement the mean class accuracy in Figure 2(b). Table 15 reports both full-dataset aggregates and the selected validation accuracy for every condition. Tables 16 and 17 report all 119 Frostbite and 49 Seaquest class-strength combinations. Percentages are rounded to two decimal places; aggregates are computed from unrounded counts or class accuracies.

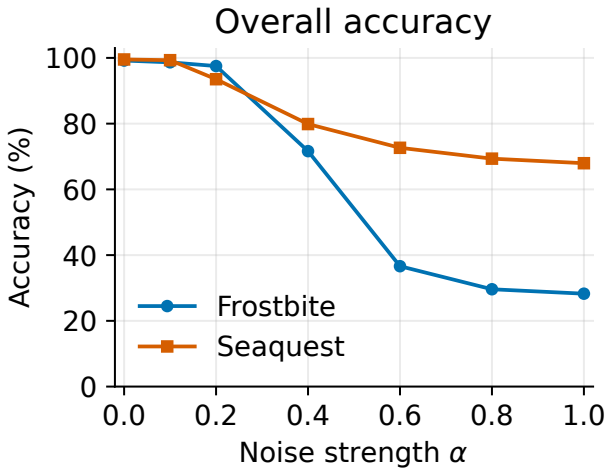


Figure 10: Overall classification accuracy across noise strengths. This metric weights classes by their sample frequency and is computed on the full probing dataset, including probe-training frames. Each point uses a separately trained MLP at the indicated strength with seed 42 and a fixed encoder per game. Lines connect the seven measured strengths. Mean class accuracy is shown in Figure 2(b); validation and class-wise results are reported in the tables 15-17.

Table 15: Complete noise-strength sweep (percent). Overall and mean class accuracy use the full dataset, including training frames. Best validation accuracy is the maximum over epochs used to select the probe. Each condition trains a separate MLP at the indicated noise strength.

	Frostbite			Seaquest		
α	Overall	Class mean	Best val.	Overall	Class mean	Best val.
0.0	99.19	98.27	96.76	99.52	96.29	97.67
0.1	98.65	96.65	95.14	99.30	95.02	96.59
0.2	97.52	96.02	87.57	93.52	47.86	91.32
0.4	71.63	59.04	51.08	79.87	22.19	79.40
0.6	36.62	14.75	30.27	72.65	18.62	72.61
0.8	29.61	9.70	25.95	69.35	16.55	69.22
1.0	28.26	9.07	24.05	67.96	15.04	68.00

Frostbite: stage-wise changes. At $\alpha = 0$, all class accuracies are at least 92.00%. Mean class accuracy remains 96.02% at $\alpha = 0.2$, but falls to 59.04% at $\alpha = 0.4$ and 14.75% at $\alpha = 0.6$. The deterioration varies by stage: at $\alpha = 0.4$, stage 1 has 11.54% accuracy while stage 16 retains 92.86%. At $\alpha = 0.6$, only stages 0, 6, 7, 10, 11, 13, and 16 have any correct predictions. At both $\alpha = 0.8$ and $\alpha = 1$, only stages 7 and 16 have correct predictions. Under complete background replacement, their class accuracies are 85.15% and 69.05%, with 0% for every other stage. The 28.26% overall accuracy is above the 20.33% constant-majority baseline (always predicting stage 7), while the 9.07% class mean is only modestly above the $100/17 = 5.88\%$ uniform-random baseline. The selected validation accuracy at this strength is 24.05%.

Seaquest: imbalance and minority-class performance. Class 0 accounts for 39,767 of 59,077 frames (67.31%), so overall accuracy can remain high even when most classes are decoded imperfact. At $\alpha = 0.1$, every class retains at least 90.00% accuracy. At $\alpha = 0.2$, overall accuracy is still 93.52%, but the class mean is 47.86%; counts 4 and 6 have no correct predictions, while counts 2, 3, and 5 have accuracies of 35.23%, 41.15%, and 69.06%. For every $\alpha \geq 0.4$, counts 2–6 have 0% accuracy. At $\alpha = 1$, count-0 and count-1 accuracies are 97.72% and 7.58%, respectively, yielding 67.96% overall accuracy and 15.04% mean class accuracy. The overall result is close to the 67.31% constant-majority baseline; it should not be called uniform-random chance, whose expected accu-

Table 16: Frostbite: every class at every noise strength. Entries are class-conditional accuracies (percent) over the full dataset, including probe-training frames. The count N_c is unchanged across columns. Each column uses a separately trained MLP.

Igloo stage	N_c	$\alpha = 0$	0.1	0.2	0.4	0.6	0.8	1.0
0	153	99.35	98.69	98.04	73.86	32.03	0.00	0.00
1	26	96.15	92.31	100.00	11.54	0.00	0.00	0.00
2	25	92.00	76.00	84.00	40.00	0.00	0.00	0.00
3	28	100.00	100.00	100.00	28.57	0.00	0.00	0.00
4	77	98.70	98.70	97.40	68.83	0.00	0.00	0.00
5	54	100.00	98.15	96.30	72.22	0.00	0.00	0.00
6	159	100.00	99.37	98.11	79.87	8.18	0.00	0.00
7	377	99.73	99.73	98.41	83.82	87.27	77.19	85.15
8	95	98.95	100.00	96.84	32.63	0.00	0.00	0.00
9	91	98.90	97.80	97.80	72.53	0.00	0.00	0.00
10	121	100.00	100.00	95.87	48.76	0.83	0.00	0.00
11	120	98.33	99.17	97.50	66.67	0.83	0.00	0.00
12	47	97.87	93.62	89.36	46.81	0.00	0.00	0.00
13	89	97.75	96.63	97.75	79.78	34.83	0.00	0.00
14	28	92.86	92.86	89.29	39.29	0.00	0.00	0.00
15	70	100.00	100.00	95.71	65.71	0.00	0.00	0.00
16	294	100.00	100.00	100.00	92.86	86.73	87.76	69.05
Overall	1,854	99.19	98.65	97.52	71.63	36.62	29.61	28.26
Class mean	–	98.27	96.65	96.02	59.04	14.75	9.70	9.07

Table 17: Seaquest: every class at every noise strength. Entries are class-conditional accuracies (percent) over the full dataset, including probe-training frames. The count N_c is unchanged across columns. Each column uses a separately trained MLP.

Diver count	N_c	$\alpha = 0$	0.1	0.2	0.4	0.6	0.8	1.0
0	39,767	99.82	99.70	97.24	91.26	91.21	93.43	97.72
1	17,003	99.35	99.02	92.32	64.09	39.10	22.43	7.58
2	1,374	95.78	94.76	35.23	0.00	0.00	0.00	0.00
3	260	95.38	93.46	41.15	0.00	0.00	0.00	0.00
4	146	95.89	91.78	0.00	0.00	0.00	0.00	0.00
5	417	96.88	96.40	69.06	0.00	0.00	0.00	0.00
6	110	90.91	90.00	0.00	0.00	0.00	0.00	0.00
Overall	59,077	99.52	99.30	93.52	79.87	72.65	69.35	67.96
Class mean	–	96.29	95.02	47.86	22.19	18.62	16.55	15.04

racy is $100/7 = 14.29\%$. The selected validation accuracy is 68.00%. The class-wise results also show that the fitted probe is not simply always predicting class 0.

Interpretation and scope. The decline persists when each MLP is trained on the same corruption strength on which it is evaluated. This rules out an evaluation-only change in noise strength for the probe as the protocol behind these results. It does not rule out a distribution shift for the frozen encoder, which was trained on observations collected during training, nor separate representation changes from finite probe capacity, optimization, class imbalance, or limited data. Because full-dataset accuracies include training frames and the validation set is used for checkpoint selection, the reported numbers do not estimate performance on an independent test set. The frame split and single-seed design further limit claims about generalization to new trajectories or training runs. These results establish degraded decoding by the specified supervised procedure as background corruption increases, not that target information has necessarily been erased or that spatial entanglement alone causes the degradation.

The corruption results complement the clean probing results in a specific way. Clean probing indicates that the evaluated task variables can be decoded from the latent state, whereas the corruption sweep suggests that this recoverability can depend strongly on surrounding visual context. One pos-

sible concern for downstream control is that policy and value learning from finite trajectories may associate useful outcomes with contextual features that co-vary with the task-relevant factor rather than with the factor alone. If those relationships later change, the previously learned association may become less reliable. The present corruption experiment does not establish that this mechanism occurs during MBRL training, but it motivates examining experience utilization as a downstream intervention in addition to representation quality.

K ACROSS-SEED PERFORMANCE VARIATION IN ATARI 100K

This appendix examines performance variation across training seeds using the original STORM results (Zhang et al., 2023). We analyze the reported final evaluation scores from five independent training runs for each of the 26 Atari 100k games. These results are separate from our four-seed baseline evaluation in Section 5.1 and are used here to characterize across-seed performance variation rather than to evaluate FLASH.

Measurement. For each game g, let $A_{g,s}$ denote the final evaluation score of the run with training seed s, and let R_g and H_g denote the random-policy and human reference scores, respectively. We measure across-seed variation as the difference between the maximum and minimum human-normalized scores (HNS):

$$\Delta_{\text{seed}}^{(g)} = \frac{\max_s A_{g,s} - \min_s A_{g,s}}{H_g - R_g}. \tag{13}$$

We use the random-policy and human reference scores in Table 2. The resulting $\Delta_{\text{seed}}^{(g)}$ measures the range of final evaluation performance across the five reported training seeds for each game.

Results. Table 18 reports the minimum and maximum evaluation scores and their human-normalized difference for all 26 games. In seven games (Boxing, CrazyClimber, Gopher, Jamesbond, Kangaroo, Krull, and RoadRunner), the difference exceeds 1.0 HNS. The variation is particularly pronounced in Gopher, where the reported scores range from 2,084 to 14,230, corresponding to a difference of 5.637 HNS. These results demonstrate that substantial differences in final evaluation performance can arise across independent training runs of the same agent under the same training configuration.

Table 18: Across-seed performance variation in the original STORM results on Atari 100k. Minimum and maximum scores are obtained from five reported training runs per game. Δ_{seed} denotes the difference between the maximum and minimum scores normalized by the human-random score gap. Bold values indicate games with $\Delta_{\text{seed}} > 1.0$.

Game	Min. score	Max. score	Δ_{seed}
Alien	627.0	1604.0	0.142
Amidar	155.1	254.4	0.058
Assault	691.6	1002.1	0.598
Asterix	825.0	1325.0	0.060
BankHeist	236.0	936.0	0.947
BattleZone	7500.0	17500.0	0.287
Boxing	71.5	90.5	**1.583**
Breakout	8.0	24.0	0.556
ChopperCommand	1400.0	2230.0	0.126
CrazyClimber	52520.0	83140.0	**1.222**
DemonAttack	145.0	186.0	0.023
Freeway	0.0	0.0	0.000
Frostbite	267.0	2231.0	0.460
Gopher	2084.0	14230.0	**5.637**
Hero	7081.0	13499.5	0.215
Jamesbond	370.0	780.0	**1.497**
Kangaroo	1020.0	7400.0	**2.139**
Krull	6694.0	9446.0	**2.578**
KungFuMaster	20600.0	33020.0	0.553
MsPacman	2177.0	3386.4	0.182
Pong	-7.4	20.8	0.799
PrivateEye	82.5	15010.0	0.215
Qbert	3987.5	5012.5	0.077
RoadRunner	12140.0	21280.0	**1.167**
Seaquest	394.0	746.0	0.008
UpNDown	4933.0	11750.0	0.611

Interpretation and limitations. The observed variation is consistent with differences arising from stochastic training trajectories, but it does not isolate their underlying causes. Changing the random seed can affect exploration, collected experiences, and the learned encoder, world-model, and Actor-Critic parameters. Therefore, the reported performance differences do not establish that downstream experience utilization alone accounts for the variation. Moreover, the maximum-minimum range is a descriptive statistic over five training seeds, rather than an estimate of the full performance distribution.

Together with the probing results in Section 3, these observations motivate investigating experience utilization as a downstream intervention point. FLASH addresses this question by selectively reusing previously encountered salient experiences and structurally similar historical contexts during Actor-Critic learning, without replacing the underlying encoder or world-model architecture. The across-seed variation reported here serves as motivation for this investigation, rather than evidence of FLASH's effectiveness.

L Ablation Studies on Triggering and Retrieval

To isolate and validate the impact of the core components introduced in FLASH, we conducted rigorous ablation studies.

L.1 Ablation 1: The Role of Neighbor Retrieval

In this setting, we preserve the exact triggering condition of FLASH (ReLU(TD-error) $\times$ Softplus(Value Difference)) but disable neighbor retrieval from the hash buckets. When an anchor is triggered, only its associated replay context is included in the imagination batch, without retrieving structurally similar historical contexts. Table 19 compares Full FLASH ($n = 16$) with No neighbor ($n = 1$) on STORM, using the default anchor setting for both variants. For each game, both variants use the same training seeds. The number of training seeds can differ from the baseline-FLASH comparison in Table 2.

Table 19: Neighbor retrieval ablation on STORM. Full FLASH uses retrieval target $n = 16$; No neighbor uses $n = 1$ (anchor only), both with the default anchor setting. The number of training seeds can differ across ablations and from Table 2. A dash indicates no paired results. Aggregate metrics summarize games with paired results using the Random/Human references in Table 2. Mean, Median, and Optimality Gap use human-normalized game means; IQM pools the unrounded per-seed human-normalized scores. Bold indicates the best score in each pair (lower for Optimality Gap).

Game	Full FLASH	No neighbor
Alien	**1147.9**	1031.9
Amidar	**182.4**	171.9
Assault	723.7	**833.9**
Asterix	**716.7**	668.3
BankHeist	599.2	**622**
BattleZone	**13150**	10766.7
Boxing	**82.1**	78.4
Breakout	13	**13.2**
ChopperCommand	1497.5	**1807.5**
CrazyClimber	62040	**62650**
DemonAttack	**138.8**	134
Freeway	**27.6**	27.1
Frostbite	2054.9	**2058.1**
Gopher	**7952**	4862
Hero	**13217.1**	12962.4
Jamesbond	**593.8**	498.8
Kangaroo	2285	**2480**
Krull	**6716**	6075
KungFuMaster	**27535**	18785
MsPacman	**2059.5**	1838.2
Pong	**15.4**	8.7
PrivateEye	-593.9	**177.5**
Qbert	**3702.9**	3406.7
RoadRunner	**17418.3**	16721.7
Seaquest	**490**	481.3
UpNDown	**6891.2**	6507.7
#Superhuman (↑)	**8**	7
Mean (↑)	**1.164**	1.042
Median (↑)	**0.518**	0.501
IQM (↑)	**0.565**	0.538
Optimality Gap (↓)	**0.442**	0.456

L.2 Ablation 2: Value Difference vs. Absolute Value

Here, we replace the signed temporal value difference ($V(s_t) - V(s_{t-1})$) in the triggering condition with the absolute state value ($V(s_t)$), while retaining normalization and multiplicative score combi-

nation. This tests whether anchor selection benefits from changes in estimated value rather than its level. Table 20 compares Full FLASH with this absolute-value variant on STORM, using retrieval target $n = 16$ and the default anchor setting for both. For each game, both variants use the same training seeds. The number of training seeds can differ from the neighbor-retrieval comparison.

Table 20: Value signal ablation on STORM. Full FLASH uses the signed temporal value difference $V(s_t) - V(s_{t-1})$; Absolute value uses the state value $V(s_t)$. Both use retrieval target $n = 16$, the default anchor setting, and multiplicative score combination. The number of training seeds can differ across ablations and from Table 2. A dash indicates no paired results. Aggregate metrics summarize games with paired results using the Random/Human references in Table 2. Mean, Median, and Optimality Gap use human-normalized game means; IQM pools the unrounded per-seed human-normalized scores. Bold indicates the best score in each pair (lower for Optimality Gap).

Game	Full FLASH	Absolute value
Alien	**1126.8**	1092.5
Amidar	**196.0**	173.1
Assault	**751.7**	730
Asterix	946.2	**947.5**
BankHeist	599.2	**659.2**
BattleZone	12600	**13450**
Boxing	**83.8**	81.6
Breakout	12	**12.3**
ChopperCommand	1497.5	**1622.5**
CrazyClimber	**62040**	53202.5
DemonAttack	138.8	**143.1**
Freeway	**27.6**	**27.6**
Frostbite	1498.8	**1659.8**
Gopher	**7952**	4947.5
Hero	13217.1	**13368.5**
Jamesbond	**593.8**	521.2
Kangaroo	2900	**3140**
Krull	6675.8	**7287**
KungFuMaster	**27535**	17935
MsPacman	1885.8	**2422.2**
Pong	**15.4**	7.5
PrivateEye	-593.9	**911.2**
Qbert	**4106.9**	2636.2
RoadRunner	**17637.5**	16602.5
Seaquest	**510**	442.5
UpNDown	**7972.5**	6006.5
#Superhuman (↑)	**9**	7
Mean (↑)	**1.178**	1.086
Median (↑)	**0.538**	0.452
IQM (↑)	**0.642**	0.591
Optimality Gap (↓)	**0.434**	0.450

L.3 Ablation 3: Multiplicative vs. Additive Combination

Let $x_t = \mathrm{ReLU}(z_t^{\mathrm{TD}})$ and $y_t = \mathrm{softplus}(z_t^{\Delta V})$, where the two z-scores use the preceding per-environment EMA statistics as in Eq. equation 4. Full FLASH uses $S_t^{\mathrm{mult}} = x_t y_t$ with threshold $\tau_{\mathrm{mult}} = 3.5$. The additive variant instead uses

$$S_t^{\mathrm{add}} = x_t + y_t, \qquad \tau_{\mathrm{add}} = 4.0. \tag{14}$$

Both transformed signals have unit coefficients; no additional mixing coefficient is used. The implementation selects the maximum score in each replay sequence and triggers when it reaches the corresponding threshold. The stored additive-run configurations use `score_combination=add` and `additive_z_score_threshold=4.0`; the multiplicative threshold remains 3.5.

The two thresholds are specified separately because addition and multiplication produce different score scales. These settings do not by themselves establish equal anchor-trigger rates. Without mea-

sured trigger counts, the comparison evaluates the combination rule together with its stated threshold, rather than isolating the rule at a matched retrieval frequency. Unlike the product, the additive score can trigger from the value-difference term even when the normalized TD-error is non-positive. Calibrating thresholds to matched trigger rates and studying alternative relative weights are possible follow-up experiments.

Table 21 compares the variants on STORM with retrieval target $n = 16$ and the same default anchor setting. Each game uses paired training seeds. The number of seeds and the corresponding Full FLASH means can differ across the three ablations and from Table 2.

Table 21: Score combination ablation on STORM. Full FLASH multiplies the ReLU-transformed normalized TD-error and softplus-transformed normalized temporal value difference; Additive sums these terms. Both use retrieval target $n = 16$ and the default anchor setting. The number of training seeds can differ across ablations and from Table 2. A dash indicates no paired results. Aggregate metrics summarize games with paired results using the Random/Human references in Table 2. Mean, Median, and Optimality Gap use human-normalized game means; IQM pools the unrounded per-seed human-normalized scores. Bold indicates the best score in each pair (lower for Optimality Gap).

Game	Full FLASH	Additive
Alien	1126.8	**1259**
Amidar	**196.0**	168.5
Assault	**751.7**	702.9
Asterix	946.2	**962.5**
BankHeist	**599.2**	566.2
BattleZone	12600	**13275**
Boxing	**83.8**	76.9
Breakout	**12**	11.5
ChopperCommand	1497.5	**1665**
CrazyClimber	**62040**	52885
DemonAttack	138.8	**156.9**
Freeway	27.6	**27.7**
Frostbite	**251**	226.5
Gopher	**7531**	2516
Hero	13108.8	**13410.5**
Jamesbond	593.8	**598.8**
Kangaroo	**2900**	2885
Krull	**6783.5**	5904.5
KungFuMaster	**27535**	23222.5
MsPacman	1885.8	**2373.8**
Pong	**15.4**	12.2
PrivateEye	-593.9	**194.6**
Qbert	**4106.9**	3816.9
RoadRunner	**17637.5**	16817.5
Seaquest	**510**	464.5
UpNDown	**7972.5**	6122
#Superhuman (↑)	**9**	7
Mean (↑)	**1.163**	0.984
Median (↑)	**0.536**	0.458
IQM (↑)	**0.599**	0.565
Optimality Gap (↓)	**0.446**	0.455