LEARN FROM FAILURE, ACT WITH MEMORY: REFLECTIVE AND PROSPECTIVE LEARNING FOR REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Exploration–exploitation coordination is a core challenge in reinforcement learning. Existing methods rely on static weighting or annealing variants, limiting state-wise exploration–exploitation balance. This causes premature convergence or ineffective exploration, impairing learning efficiency and performance. Essentially, the coordination problem is not about static allocation but an experience-based decision problem requiring agents to remember, reflect on, and leverage past successes and failures. Inspired by human retrospective memory and prospective anticipation, we propose an integrated adaptive exploration–exploitation framework, **MemoRL**. On the exploitation side, the Retrospective Module distills shared patterns from successes and identifies weaknesses from failures for targeted improvement. On the exploration side, the Prospective Module combines coarse global search with fine-grained state discrimination via sparsity-constrained latent representations. We theoretically justify proposed exploration mechanism. Furthermore, we introduce a policy-level ensemble to adaptively coordinate the prospective and retrospective modules, enabling a exploration–exploitation balance based on demands. We prove the coordinated strategy outperforms non-integrated strategies. MemoRL can be plugged into existing algorithms. We evaluate MemoRL across **56** environments against 23 baselines in total, where our method achieves better average performance than baselines. Notably, it achieves up to a **15**$\times$ improvement on Montezuma's Revenge. MemoRL improves time efficiency over baselines by **41.8%** on average.

1 INTRODUCTION

Reinforcement Learning (RL) (Zhao et al., 2022; Gan et al., 2024a; Yan et al., 2024; Shaheen et al., 2025; Bairaktaris & Johannssen, 2025) has achieved remarkable progress in robotic control Sharifi et al. (2025); Tang et al. (2025), game decision-making (Barros e Sa & Madeira, 2025; Zhao et al., 2022), and sequential tasks (Bein & Niv, 2025; Yan et al., 2025), but efficient exploration–exploitation coordination remains a fundamental challenge (Ahmed et al., 2019; Schulman et al., 2015; 2017; Bellemare et al., 2016). An agent must exploit experience while exploring environments for strategies, and this trade-off determines efficiency and performance. Without coordination, exploration and exploitation may misalign with needs, degrading performance: excessive exploration repeats low-value states, while premature exploitation causes convergence.

Although prior studies attempt to mitigate the exploration–exploitation trade-off, most rely on heuristic or static coordination, e.g., fixed weighting or predefined annealing (Szpruch et al., 2026; Sutton & Barto, 1999; Tokic, 2010; Mnih et al., 2015; Taiga et al., 2020; Dabney et al., 2021; Dann et al., 2022; Yan et al., 2023). This prevents the agent from adaptively prioritizing exploration or exploitation across different states, thereby inducing a systematic imbalance during training.

Effective coordination between exploration and exploitation requires understanding real-time decision demands and recalling historical experience. The agent should learn reusable policies from successes, but more importantly, analyze failures to identify weaknesses. Continuous retraining on failure modes until the agent learns to avoid failures is overlooked in existing work.

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

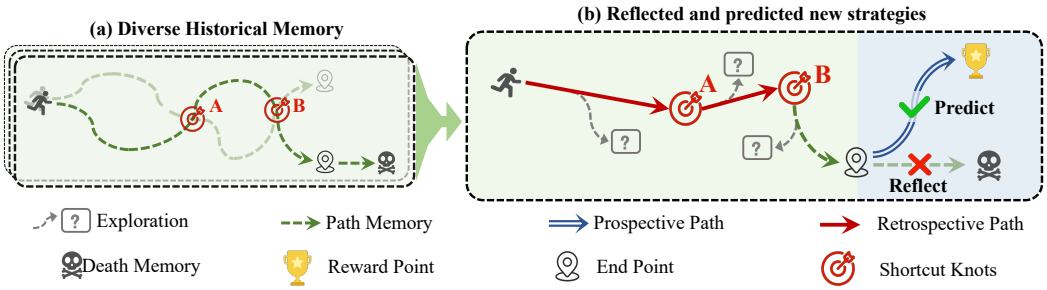


Figure 1: **Retrospective Memory Motivation**. MemoRL learns to avoid danger and updates shortcut policy by digesting advantageous-memory commonalities and reflecting on failed memory.

To enable real-time exploration–exploitation coordination based on current state and decision demands, inspired by human cognition, we revisit this problem via retrospective memory and prospective anticipation, proposing MemoRL. Specifically, on the exploitation side, we design a Retrospective Module that searches the shortest decision paths from successful experiences. Importantly, our Retrospective Module also emphasizes modeling failed experiences to identify decision deficiencies in the current policy and conduct targeted corrective training (see Fig. 1). On the exploration side, inspired by humans' ability to make prospective judgments based on experience, we introduce a Prospective Module. This module not only performs global exploration of novel states but also distinguishes similar yet nonidentical states at a fine-grained level using sparsity-constrained latent representations. This enables more precise exploration and targeted policy responses, a capability absent from prior methods (see Fig. 2). We further provide theoretical justification for the proposed prospective exploration mechanism. After explicitly modeling exploration and exploitation, we introduce a policy-level ensemble to dynamically coordinate the retrospective and prospective modules, enabling adaptive exploration–exploitation balance across states.

In summary, the main contributions of this paper are threefold:

- We propose an ensemble learning framework based on human retrospective and prospective memory, achieving adaptive exploration–exploitation coordination at the policy level, with theoretical performance guarantees.
- We introduce a prospective exploration module that integrates coarse global search with fine-grained state discrimination, enabling the agent to distinguish similar states and conduct more precise, targeted exploration beyond global novelty-driven methods.
- We propose a retrospective module that goes beyond imitation by extracting shared structures from successful trajectories to learn shorter decision paths, while leveraging failures to identify policy deficiencies and perform targeted correction.

2 Distinction from Prior Work

Related work is deferred to Appendix C. We summarize key differences below.

***Exploitation Methods*.** Existing methods mainly rely on imitation learning Oh et al. (2018); Bagaria et al. (2025); Cloud et al. (2026); Minoshima & Arai (2026), reproducing high-return policies while ignoring failures, which prevents understanding of when policies fail and identifying decision gaps. Instead, our retrospective module models both successes and failures, shortening decision paths via shared successful patterns and inferring missing policies from failures. *Fig. 1: motivation. Fig. 4 and Sec. 4.1: evidence of shortest-path learning and reflection.*

***Exploration Methods*.** Existing methods Burda et al. (2018b); Badia et al. (2020b); Kayal et al. (2025); Sukhija et al. (2025) improve state coverage via novelty-driven incentives but rely on coarse signals, lacking a fine-grained distinction between similar states. This causes missed subtle differences and limits exploration precision, treating exploration as a global behavior rather than a state-dependent decision. In contrast, our prospective module preserves coarse exploration while explicitly modeling fine-grained differences between similar states, balancing global coverage and local discrimination to enable more targeted exploration. *The motivation is shown in Fig. 2, and evidence of fine exploration in Fig. 5 & 6 and Sec. 4.1.*

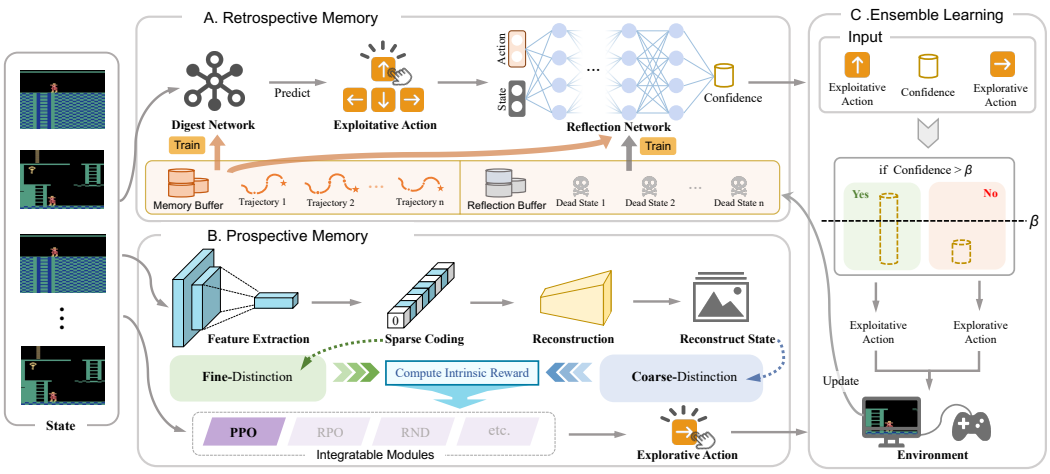


Figure 3: **MemoRL Framework.** A) The retrospective memory learns to memorize history "goood" trajectories and "dead" states, and yields real-time evaluation for current state. B) The prospective module boosts exploration via coarse-fine intrinsic motivation to accumulate diverse experiences for the memory. C) The two modules are coordinated with ensemble learning.

Coordination. Existing methods coordinate exploration and exploitation via fixed weights or decaying exploration, which are static controls independent of decision demands. In contrast, we employ ensemble learning to jointly model them at the policy level, enabling dynamic, state-dependent decisions.

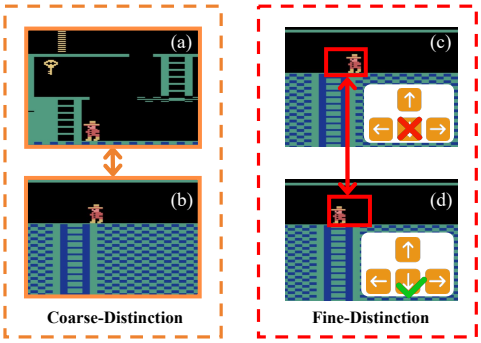


Figure 2: **Prospective Exploration Motivation**. Existing exploration methods distinguish only completely different states like (a) and (b), but fail on nearly identical states like (c) and (d) with different action importance.

3 METHOD

MemoRL introduces a Retrospective Module for exploitation and a Prospective Module for exploration, and achieves a dynamic balance between the two (Fig. 3). Sec. 3.1 presents the Retrospective Module: it extracts shared structures from successful trajectories to learn shorter, higher-reward paths, and models failed experiences to correct wrong decisions at key states. Sec. 3.2 presents the Prospective Module: a hierarchical mechanism with coarse and fine-grained explorers for global coverage and state discrimination, with theoretical soundness proof. Sec. 3.3 uses ensemble learning to adaptively collaborate both modules, achieving balanced exploration-exploitation, with a proof that the ensemble policy outperforms the original. RL and our method preliminaries are provided in Appendix. A.

3.1 RETROSPECTIVE MEMORY FOR EXPLOITATION

The Retrospective Memory module models successes and failures, compressing trajectories from successful experiences and using failures to identify gaps and apply targeted correction. The structure of the Retrospective Memory module is illustrated in Fig. 3A.

Success-Based Exploitation. We aim to guide learning by extracting shared path structures across trajectories, driving memory to converge to shorter, more optimal trajectories. Thus, we adapt to new environments by continually incorporating new experiences and eliminating outdated experiences, ensuring a reasonable memory capacity and avoiding excessive overhead. Based on this, we maintain a fixed-size bank of advantageous trajectories, $M = \{T_1, T_2, T_3, \ldots, T_i\}$, where each T_i contains a complete sequence of state-action-reward tuples:

$$T_i = \{(s_1, a_1, r_1), (s_2, a_2, r_2), \ldots, (s_i, a_i, r_i)(R_i, N_i)\},$$

where s_i, a_i, and r_i represent the state, action, and cumulative reward, respectively, N_i is the total length of the trajectory excluding terminal (death) states, and R_i is the total reward accumulated in the trajectory. If the trajectory ends in a non-terminal state, it is entirely stored.

The update strategy of the memory bank works as follows: if $R_i = R_k$, the trajectory with the shorter length, N_i or N_k, replaces the longer one; if $R_i < R_k$, the entire trajectory T_i is replaced by T_k. This enables the memory to gradually converge to a set of more optimal trajectories.

Digest Network. Based on the High-Value Trajectory Construction, the resulting set of advantageous trajectories not only exists as stored memory but is further transformed into structured decision constraints to guide the training of the Digest Network. Specifically, we use the set of advantageous trajectories M as the training set. For each state–action pair (s, a) from the high-value trajectories, the network takes the state s as input, outputs an action $\hat{a}$, and is optimized via the loss function:

$$\mathcal{L}_{\text{digest}}(\theta) = -\sum_{(s,a)\in M} \log \pi_\theta(a|s). \tag{1}$$

As shown in Fig. 1, the common states across different successful trajectories characterize the critical path toward the goal. As the memory bank continuously retains trajectories with higher returns and shorter paths during the update process, these trajectories gradually align at shared states. In this way, they implicitly form a more compact path structure. Based on this, the Digest Network learns shared state-action transitions, compressing redundant paths into a consistent decision pattern, enabling the policy to converge toward a shorter path.

Failure-Based Exploitation. Unlike existing explanation methods that focus solely on advantageous experiences, our approach also emphasizes learning from failures. The R-bank maintains state-action pairs one step before failure. These experiences arise from the Reflection Network's incorrect action assessments and serve as negative samples for the Reflection Network.

Reflection network. The reflection network is responsible for learning how to avoid failure states. To achieve this, a reliability level (C_ω) is predicted, indicating the confidence of the action $\hat{a}$ from the memory-based policy of the Digest Network. If $\hat{a}$ tends to lead the agent to failure, the reflection reliability will be decreased; conversely, if $\hat{a}$ contributes to the agent's success, the reflection reliability is increased. The reflection process continues until the agent can come over the failed process. The loss function is defined as follows:

$$loss_C(\omega) = \sum_{(s,a)\in M} \|C_\omega(s,a) - 1\|^2 + \sum_{(s,a)\in R} \|C_\omega(s,a) - 0\|^2, \tag{2}$$

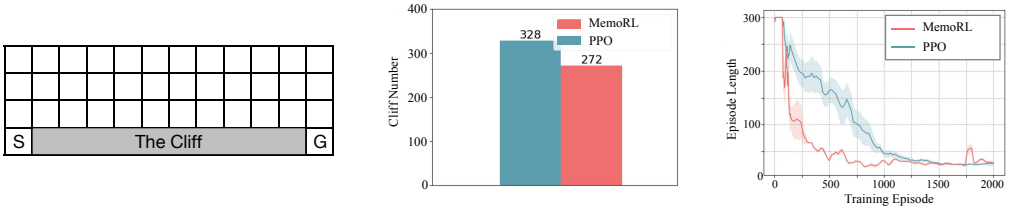
where M and R represent actions from M-bank and R-bank. We force the reliability of the action from the M-bank to 1 and the action from the R-bank to 0. This enhances the robustness of the overall decision-making system by providing dynamic feedback and optimizing prediction accuracy.

3.2 Prospective Module for Exploration

We propose a Prospective Module that establishes a new intrinsic motivation paradigm and a fine-grained distinction mechanism (Fig. 3B). This Module unifies the training of exploration policy networks with intrinsic motivation metrics. We provide proof of the retention of the optimal policy.

Coarse-Distinction Exploration. As illustrated in Fig. 3B, our intrinsic reward is based on a two-level distinction scheme for coarse state novelty and fine-grained distinctions between similar states, denoted as C-Discriminator and F-Discriminator, respectively. Given a raw image of a state as the input s_t, the C-Discriminator learns the feature representations $f_\theta(s_t)$ and reconstructs the image $\hat{s}_t = g_\phi(f_\theta(s_t))$. The reconstruction error between $\hat{s}_t$ and s_t serves as a coarse-grained measurement to identify novel images during training. As such, the states that have been explored many times can be distinguished from new states due to their different fitting degree in the reconstruction network.

Fine-Distinction Exploration. The C-Discriminator is helpful for novel state exploration, but reconstruction error cannot distinguish visually similar states, which is necessary in environments with sparse feedback. For instance, consecutive images may yield similar reconstruction errors while representing entirely different importance. As shown in Fig. 2c and Fig. 2d, the two states are visually similar, yet Fig. 2d has a larger actionable policy space and is therefore more important. Existing intrinsic reward methods cannot distinguish similar states with different importance. Therefore,



(a) CliffWalking (b) Number (c) Length

Figure 4: **Reflection & Shortest-Path Visualization** (No Prospective Module). (b) Total cliff falls; lower is better. Fewer falls with MemoRL verify effective failure reflection. (c) Goal-reaching steps during training. Fewer steps verify MemoRL's shortest-path distillation from successful trajectories.

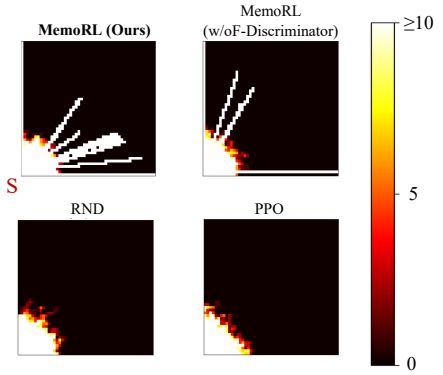


Figure 5: **Coarse Exploration Visualization.** In the reward-free environment, our coarse exploration module explores deeper than the baselines.

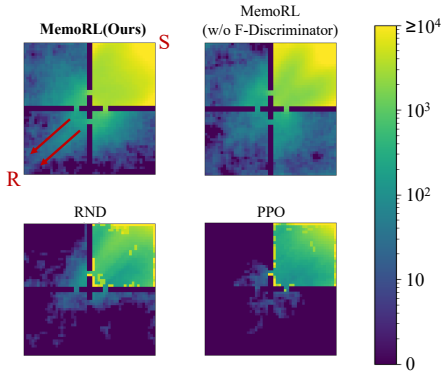


Figure 6: **Fine Exploration Visualization.** In the single-reward environment, MemoRL reaches rewards and forms stable paths via fine distinction.

we further propose an innovative F-Discriminator to support fine-grained state discrimination. The F-Discriminator poses a sparse constraint to the feature vector $f_\theta(s_t)$ with a ℓ_1-norm regularization.

Fine-Distinction: For trained states, due to the sparsity of their representation vector $f_\theta(s_t)$, the magnitude of the corresponding regularization term is smaller. In contrast, for untrained or novel states, the vector $f_\theta(s_t)$ will be dense, the magnitude of the regularization terms are larger. By analyzing the sparsity of $f_\theta(s_t)$, we can precisely distinguish between trained states, states similar to trained ones, and entirely new states. We unify the exploration policy's network training loss with the coarse-fine intrinsic motivation measure, formulating our new paradigm as below:

$$R_{int} = loss_{f,g}(\theta, \phi) = \frac{1}{2} \left\| s_t - g_\phi(f_\theta(s_t)) \right\|_2^2 + \lambda \left\| f_\theta(s_t) \right\|_1 , \tag{3}$$

f_θ and g_ϕ denote the feature extraction network (encoder) and reconstruction network (decoder).

This paradigm distinguishes the importance of closely related images, improving intrinsic reward accuracy and enabling more precise allocation of exploration effort. Furthermore, the unified paradigm integrates network loss and intrinsic reward evaluation, avoiding manual design across tasks and enhancing generalization for multi-task adaptation. The coarse-fine motivation achieves more precise state exploration and provides accurate learning samples for memory learning.

Proof of the retaining of policy optimality. Here, we aim to ascertain the impact of our intrinsic exploration reward on the policy's optimality. We start by introducing an assumption.

Assumption 1. *After k updates, we assume $0 \leq \alpha R_{int} \leq (1-\gamma)\min_s[Q^*(s,a^*) - Q^*(s,a^{sub})]$ holds, where Q^* is the optimal Q function, a^* and a^{sub} are optimal and suboptimal actions under the state s, respectively.*

Assumption 1 is reasonable as the networks are trained to minimize the losses; hence, the accompanying intrinsic rewards will be smaller enough after certain training steps. Under this assumption, we can prove that our new intrinsic motivation paradigm will retain the optimality of policy.

Table 1: Comparison of reward acquisition efficiency. We compare the time taken by our method and the RND method to achieve the same reward (Threshold). Our method requires significantly less training time than RND to reach the same reward. Time efficiency $= \frac{t_{\text{RND}} - t_{\text{MemoRL}}}{t_{\text{RND}}}$.

	Montezuma's Revenge	Alien	Hero	Frostbite	Qbert
Threshold	2000	1500	10000	1600	1000
Time efficiency	25%	31%	38%	86%	29%

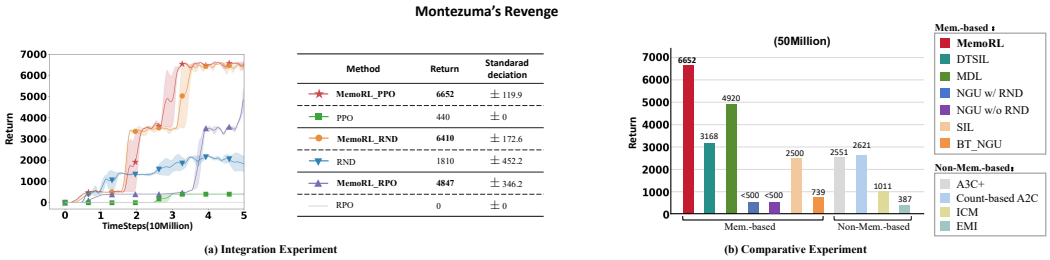


Figure 7: **Comparison in Montezuma's Revenge.** (a) Comparison between baseline methods before and after integration with MemoRL; (b) comparison with other advanced baseline models.

Theorem 1. *After k updates of the coarse-fine distinction network, under Assumption 1, the optimal action remains the same after adding the intrinsic rewards. That is, for any state s, we have*

$$\arg\max_a Q^*(s,a) = \arg\max_a Q_1^*(s,a), \tag{4}$$

where Q_1^* is the optimal Q function with intrinsic rewards.

The proof is given in Appendix B.1. Theorem 1 reveals the guarantee of intrinsic rewards based on prediction error, which has not been mentioned in existing work Bellemare et al. (2016); Pathak et al. (2017); Burda et al. (2018b) and points out a new direction for future intrinsic reward design.

3.3 Ensemble Learning for Exploration-Exploitation Balance

The Retrospective Memory module and the Prospective Module guide exploitation and exploration, respectively. We leverage ensemble learning to adaptively choose between exploiting memory and continue exploring. Denote the actions from the memory reflection as a_{mem} and actions from the original RL algorithm with our intrinsic rewards as a_{ori}. Given a policy π, e.g., PPO, we establish its greedy counterpart denoted as π_θ and define a combination function:

$$\pi_{new} = (1 - I_{\{C_\omega(s,a)\geq\beta\}})\pi + I_{\{C_\omega(s,a)\geq\beta\}}\pi_\theta, \tag{5}$$

where β is the reliability threshold, and $I_{\{C_\omega(s,a)\geq\beta\}} = 1$ when the prediction of memory reflection network $C_\omega(s,a) \geq \beta$ is larger than the threshold β.

Exploration-Exploitation trade-off: The ensemble mechanism adaptively switches between exploitation and exploration according to the confidence of the Reflection Network. Specifically, when the confidence score $C_\omega(s,a)$ for the action predicted by the Digest Network exceeds the threshold β, the agent follows the memory-based policy π_θ for exploitation. Otherwise, the agent falls back to the original RL policy π with intrinsic rewards to continue exploration. Note that we balance exploration and exploitation directly on the policy level instead of rewards, and that $I_{\{C_\omega(s,a)\geq\beta\}}$ is entirely based on the network's real-time output rather than being defined by human experience.

Proof of memory effectiveness. Through ensemble, MemoRL combines exploration and exploitation and adaptively selects between memory prediction or exploration based on the reliability index the Reflection Network provides. Our ensemble learning achieves adaptive coordination between exploration and exploitation. To underscore the efficacy of this approach, we present Theorem 2 that highlights the policy value of π_{new} is at least as good as that of π.

Theorem 2. *According to Eqn. (5), if 1) π_θ is a greedy policy for π, i.e., for any state s, we have $\pi_\theta(a^*|s) = 1$, where $a^* = \arg\max_a Q^\pi(s,a)$, and 2) for any state s, $C_\omega(s,a^*) > C_\omega(s,a)$ satisfies, where $a \in A \setminus \{a^*\}$, then*

$$V^{\pi_{new}}(s) \geq V^\pi(s).$$

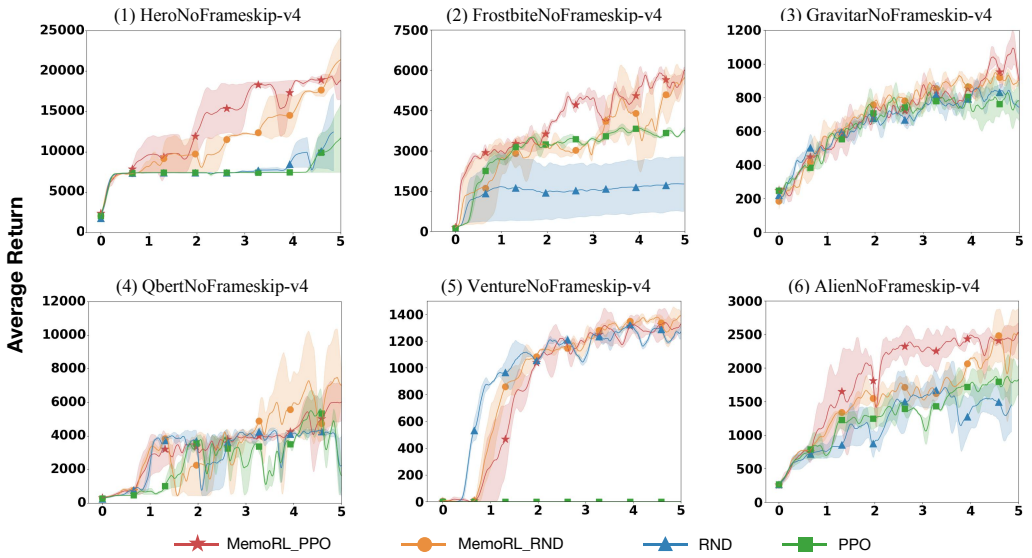


Figure 8: **Discrete-space Environments (Atari).** The x-axis represents timesteps in 10 million.

The proof is shown in Appendix B.2. This property confirms the effectiveness of ensemble policies, showcasing the robustness of the integrated learning approach. By merging both policies, MemoRL ensures that the newly derived policy does not compromise quality.

4 Experiments

Cliff Walking validates path compression and failure avoidance of the retrospective module. Four-Room and DarkChamber validate the fine-grained distinction of the prospective module. We then compare with baselines on Montezuma's Revenge, and evaluate generalization on 50+ environments covering continuous and discrete action spaces. Unless specified, MemoRL denotes the PPO-integrated version, and in the main text Atari experiments use 50M steps and 5 seeds.

4.1 Visualization Analysis

Visualization of ***Shortest-Path Learning and Reflective Capability***. We evaluate the Retrospective Module in Cliff Walking by comparing PPO with and without the module. Fewer cliff falls in Fig. 4(b) demonstrate effective failure reflection, while fewer goal-reaching steps in Fig. 4(c) verify accelerated shortest-path discovery from shared successful experiences.

Visualization of ***Fine State Distinction Visualization***. We evaluate the Prospective Module in Dark Chamber and Four Rooms with only this module enabled. In the reward-free Dark Chamber, MemoRL achieves superior exploration coverage and depth, while the full model further demonstrates deeper exploration than its variant without F-Discriminator. In Four Rooms, MemoRL achieves the largest exploration area and learns a stable reward acquisition path, whereas the variant without F-Discriminator is weaker in both coverage and quality. These results validate improved global and goal-directed exploration via fine-grained distinction.

4.2 Training Time and Efficiency Evaluation

We statistically evaluate the improvement efficiency based on the score curves of 12 environments in Fig. 8 and Fig. 9. Although MemoRL requires approximately 1.43× the training time of RND under the same training steps due to the additional exploitation module, its average improvement efficiency increases by 41.8 %, indicating that the extra time cost brings more efficient training gains. The efficiency comparison results are shown in Tab. 1.

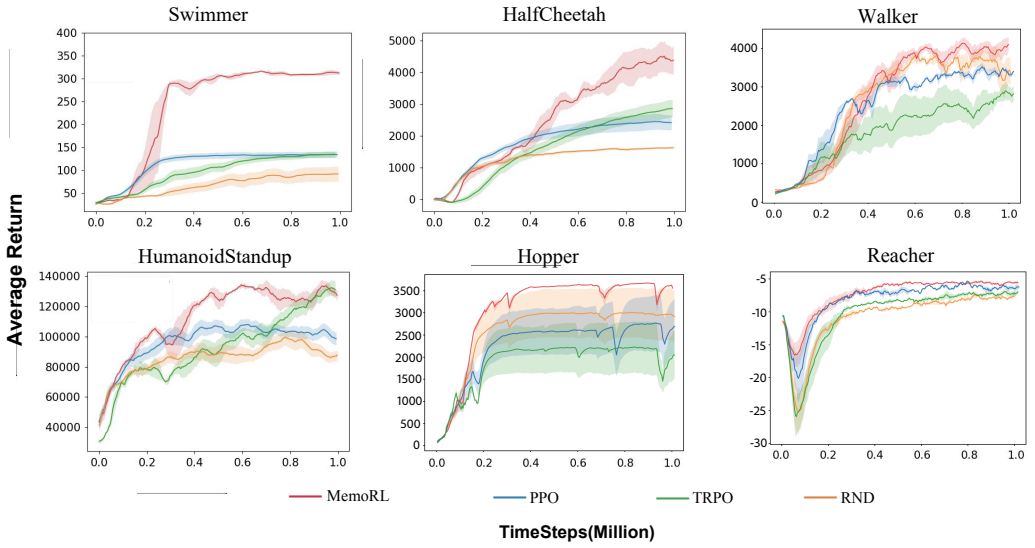


Figure 9: **Continuous Space Tasks (MuJuCo)**. The x-axis represents timesteps in millions.

Table 2: **Ablation Study on Atari**. Size indicates the exploration-module feature-vector dimension. Both Ours and Ours w/o use Size-256.

Method	Montezuma	Alien	Hero	Gravitar	Venture	Frostbite	Qbert
Base1 (PPO)	440.3 ± 0	1829.3±317.1	12798.1±7653.9	759.3±185.1	0	3973.1±38.2	2247.1±986.3
Base2 (RND)	1810.7 ± 452.2	1527.6±496.3	13769.8±8172.4	794.5±107.1	1219.7±19.3	1983.6±985.2	2305.1±1378.4
Size-64	2517.6 ± 59.1	2214.6± 547.3	13926.3±4517.9	779.3±52.0	1298.1±83.2	5241.9±437.5	3801.6±1571.9
Size-128	2751.2 ± 65.1	2116.4±67.9	19723.1±7654.2	875.4±105.9	1198.1±23.8	4973.5±679.1	4194.2±825.1
Size-512	4980.3 ± 391.7	**2790.3±317.5**	21982.7±4218.2	**1372.9±98.2**	1317.4±119.3	61437.±543.6	5923.1±1765.9
Size-1024	3761.5 ± 569.0	1834.9±412.6	18763.2±7681.5	1318.1±197.4	1251.9±39.2	5071.8±764.9	5214.1±1963.9
Size-2048	4518.3 ± 752.2	1634.9±874.9	15837.1±2371.9	1281.5±138.1	1053.8±126.4	4674.2±305.9	3641.2±719.1
Ours w/o Exploration	879.6 ± 370.2	1961.7 ± 189.5	14178.2 ± 4170.3	781.4 ± 57.7	571.2 ± 42.7	4017.8 ± 651.9	2817.4 ± 961.2
Ours w/o Exploitation	4274.2 ± 875.1	2461.9 ± 547.1	18914.6 ± 7931.5	1194.3± 371.9	1286.9 ± 93.1	4318.1 ± 798.2	4917.6 ± 533.7
Ours w/o Trade-off	4917.6 ± 975.3	2514.3 ± 579.1	20981.6 ± 5321.5	987.1 ± 217.3	1317.2 ± 134.1	5718.7 ± 819.0	5719.4 ± 1876.3
Ours (Size-256)	**6652.0 ± 119.9**	2623.9 ± 293.8	**22486.4 ± 3765.1**	1093.3 ± 97.6	**1397.8 ± 86.3**	**6274.5 ± 126.9**	**6983.1 ± 1234.1**

4.3 Main Experiments

We evaluate MemoRL on Montezuma's Revenge, which is one of the most challenging environments in RL against PPO, RND, RPO Gan et al. (2024b), representative memory methods Badia et al. (2020b); Campos et al. (2021); Oh et al. (2018); Guo et al. (2020); Ferret et al. (2020); Yan et al. (2023); Le et al. (2024), and other mainstream RL baselines Bellemare et al. (2016); Pathak et al. (2017); Kim et al. (2018); Kang et al. (2021); Hafner et al. (2020); Kooi et al. (2025); Pham et al. (2026); Kresse & Lampert (2025). As shown in Fig. 7, integrating MemoRL consistently improves score, convergence speed, and stability. Notably, PPO+MemoRL achieves a 15× score improvement. Fig. 7(b) further demonstrates that MemoRL outperforms existing memory methods by learning shared knowledge from past experiences instead of simply replaying behaviors.

4.4 Generalization Experiments

To demonstrate the generality of MemoRL, we evaluate it on Atari and MuJoCo. As Fig. 8, MemoRL significantly improves performance on five of six challenging Atari tasks beyond Montezuma, validating its effectiveness in discrete action spaces. Fig. 9 further shows strong performance in continuous action spaces (Mujuco), where PPO+MemoRL achieves nearly 3× improvement on Swimmer and 2× on HalfCheetah. Fig. 10 and Fig. 11 summarize improvements over PPO and RPO across environments, demonstrating consistent gains and strong robustness without task-specific hyperparameter tuning. We further compare our method with recent methods in Tab. 4 & 3 .

Table 3: Comparison Against **Recent Methods on Challenging Mujoco Tasks**.

Method	Year	HalfCheetah	Walker2d	Hopper
FP-PPO-M	2025	1500	2400	2100
DWC-PPO-M	2025	2100	1300	1900
CR-SAC	2026	–	4244	2574
CR-PPO	2026	1654	2587	2487
CR-A2C	2026	893	186	169
MemoRL+PPO	–	**4337**	**4293**	**3507**

Table 4: Comparison with **Recent Methods on Challenging Atari Tasks**.

Method	Year	Montezuma	Gravitar	Venture
DreamerV2	2021	81	**3789**	2
PQN	2025	0	1107	18
PQN-ResNet	2025	0	514	0
Hadamax-PQN	2025	0	1547	940
OPR-subset	2026	2500	1845	1380
MemoRL+PPO	–	**6652**	1093	**1397**

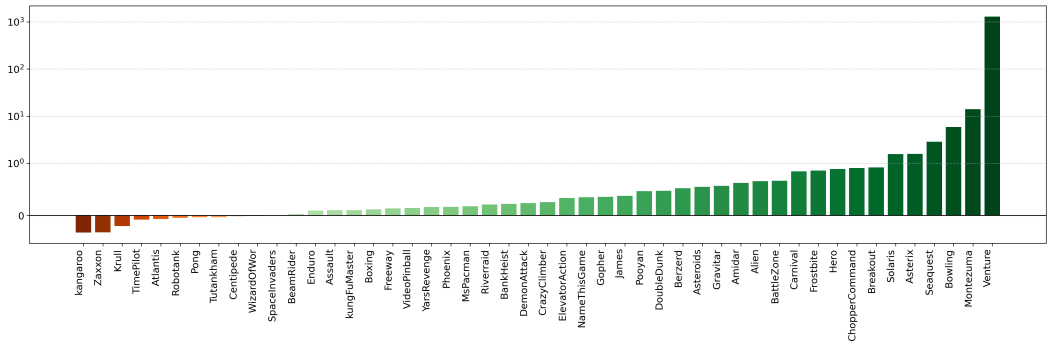


Figure 10: Summary of the results of MemoRL compared with PPO across **48** Atari environments. The y-axis denotes the normalized improvement of MemoRL against the baseline Gan et al. (2024b).

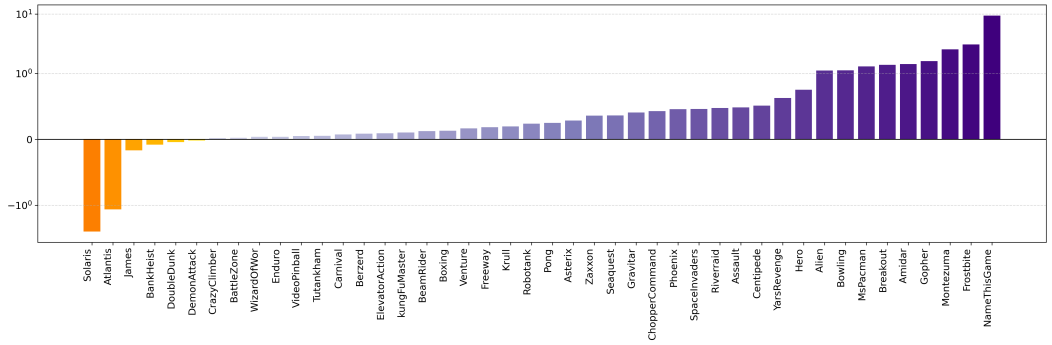


Figure 11: Summary of the results of MemoRL compared with RPO across **44** Atari environments. The y-axis denotes the normalized improvement of MemoRL against the baseline Gan et al. (2024b).

4.5 Ablation Study

We conduct ablation studies on Atari environments. As shown in Tab. 2, the sparse feature-vector dimension affects exploration performance, with Size-256 achieving the best results. In addition, using only retrospective exploitation or prospective exploration performs worse than the integrated model, validating the effectiveness of adaptive exploration-exploitation coordination.

5 Conclusion

We present MemoRL, a memory-enhanced RL framework inspired by retrospective and prospective memory. Retrospective reflection guides shortcut trajectories and avoids pitfalls, while intrinsic motivation generates high-quality exploration data for prospective memory construction with policy optimality preserved. Ensemble learning adaptively coordinates both mechanisms.

AI USE STATEMENT

In this work, generative AI tools were used mainly for language-related polishing.

REFERENCES

Joshua Achiam and Shankar Sastry. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In *International conference on machine learning*, pp. 151–160. PMLR, 2019.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in Neural Information Processing Systems*, pp. 5048–5058, 2017.

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperforming the Atari human benchmark. In *International Conference on Machine Learning*, pp. 507–517, 2020a.

Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martín Arjovsky, Alexander Pritzel, Andew Bolt, et al. Never give up: Learning directed exploration strategies. *arXiv preprint arXiv:2002.06038*, 2020b.

Akhil Bagaria, Anita De Mello Koch, Rafael Rodriguez-Sanchez, Sam Lobel, and George Konidaris. Intrinsically motivated discovery of temporally abstract graph-based models of the world. In *Reinforcement Learning Conference*, 2025.

Julius A Bairaktaris and Arne Johannssen. Outsmarting algorithms: A comparative battle between reinforcement learning and heuristics in atari tetris. *Expert Systems with Applications*, 277: 127251, 2025.

Gabriel Caldas Barros e Sa and Charles Andrye Galvão Madeira. Deep reinforcement learning in real-time strategy games: a systematic literature review: Gc barros e sá et al. *Applied Intelligence*, 55(4):243, 2025.

Andrew Barto, Marco Mirolli, and Gianluca Baldassarre. Novelty or surprise? *Frontiers in Psychology*, 4:907, 2013.

Oded Bein and Yael Niv. Schemas, reinforcement learning and the medial prefrontal cortex. *Nature Reviews Neuroscience*, 26(3):141–157, 2025.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems*, pp. 1471–1479, 2016.

John D Bransford and Marcia K Johnson. Contextual prerequisites for understanding: Some investigations of comprehension and recall. *Verbal Learning and Verbal Behavior*, 11(6):717–726, 1972.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018a.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018b.

Víctor Campos, Pablo Sprechmann, Steven Hansen, Andre Barreto, Steven Kapturowski, Alex Vitvitskyi, Adria Puigdomenech Badia, and Charles Blundell. Beyond fine-tuning: Transferring behavior in reinforcement learning. *arXiv preprint arXiv:2102.13515*, 2021.

Alex Cloud, Minh Le, James Chua, Jan Betley, Anna Sztyber-Betley, Sören Mindermann, Jacob Hilton, Samuel Marks, and Owain Evans. Language models transmit behavioural traits through hidden signals in data. *Nature*, 652(8110):615–621, 2026.

Will Dabney, Georg Ostrovski, and Andre Barreto. Temporally-extended ε-greedy exploration. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ONBPHFZ7zG4.

Chris Dann, Yishay Mansour, Mehryar Mohri, Ayush Sekhari, and Karthik Sridharan. Guarantees for epsilon-greedy reinforcement learning with function approximation. In *International conference on machine learning*, pp. 4666–4689. PMLR, 2022.

Rati Devidze, Parameswaran Kamalaruban, and Adish Singla. Exploration-guided reward shaping for reinforcement learning under sparse rewards. *Advances in Neural Information Processing Systems*, 35:5829–5842, 2022.

Sehar Shahzad Farooq, Hameedur Rahman, Samiya Abdul Wahid, Muhammad Alyan Ansari, Saira Abdul Wahid, and Hosu Lee. Curiosity-driven exploration in reinforcement learning: An adaptive self-supervised learning approach for playing action games. *Computers*, 14(10):434, 2025.

Johan Ferret, Olivier Pietquin, and Matthieu Geist. Self-imitation advantage learning. *arXiv preprint arXiv:2012.11989*, 2020.

Yaozhong Gan, Zhe Zhang, and Xiaoyang Tan. Smoothing advantage learning. In *AAAI Conference on Artificial Intelligence*, pp. 6657–6664, 2022.

Yaozhong Gan, Renye Yan, Zhe Wu, and Junliang Xing. Reflective policy optimization. *arXiv preprint arXiv:2406.03678*, 2024a.

Yaozhong Gan, Renye Yan, Zhe Wu, and Junliang Xing. Reflective policy optimization. *International Conference on Machine Learning*, 2024b.

Yijie Guo, Jongwook Choi, Marcin Moczulski, Shengyu Feng, Samy Bengio, Mohammad Norouzi, and Honglak Lee. Memory based trajectory-conditioned policies for learning from sparse rewards. *Advances in Neural Information Processing Systems*, 33:4333–4345, 2020.

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

Joshua Hare. Dealing with sparse rewards in reinforcement learning. *arXiv preprint arXiv:1910.09281*, 2019.

Yongxin Kang, Enmin Zhao, Kai Li, and Junliang Xing. Exploration via state influence modeling. In *AAAI Conference on Artificial Intelligence*, pp. 8047–8054, 2021.

Aya Kayal, Eduardo Pignatelli, and Laura Toni. The impact of intrinsic rewards on exploration in reinforcement learning. *Neural Computing and Applications*, 37(21):16269–16303, 2025.

Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. *arXiv preprint arXiv:1810.01176*, 2018.

Jacob E Kooi, Zhao Yang, and Vincent François-Lavet. Hadamax encoding: Elevating performance in model-free atari. *arXiv preprint arXiv:2505.15345*, 2025.

Fabian Kresse and Christoph H Lampert. Differentiable weightless controllers: Learning logic circuits for continuous control. *arXiv preprint arXiv:2512.01467*, 2025.

Hung Le, Kien Do, Dung Nguyen, and Svetha Venkatesh. Beyond surprise: Improving exploration through surprise novelty. In *AAMAS*, pp. 1084–1092, 2024.

Shuai Liu, Shuai Wang, Xinyu Liu, Amir H Gandomi, Mahmoud Daneshmand, Khan Muhammad, and Victor Hugo C De Albuquerque. Human memory update strategy: a multi-layer template update mechanism for remote visual monitoring. *IEEE Transactions on Multimedia*, 23:2188–2198, 2021.

Kota Minoshima and Sachiyo Arai. Autonomous reward shaping via self-generated trajectories for sparse-reward reinforcement learning. *IEEE Access*, 2026.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Jahir Sadik Monon, Deeparghya Dutta Barua, and Md Mosaddek Khan. Learning heterogeneous agent collaboration in decentralized multi-agent systems via intrinsic motivation. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pp. 2681–2683, 2025.

Isack Thomas Nicholaus and Dae-Ki Kang. Ftpsg: Feature mixture transformer and potential-based subgoal generation for hierarchical multi-agent reinforcement learning. *Expert Systems with Applications*, 270:126540, 2025.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International Conference on Machine Learning*, pp. 3878–3887, 2018.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*, 1:6, 2007.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pp. 2778–2787, 2017.

Mai Pham, Vikrant Vaze, and Peter Chin. Optimistic policy regularization. *arXiv preprint arXiv:2603.06793*, 2026.

N Rajasekhar, TK Radhakrishnan, and N Samsudeen. Exploring reinforcement learning in process control: a comprehensive survey. *International Journal of Systems Science*, 56(14):3528–3557, 2025.

Richard M Ryan and Edward L Deci. Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology*, 25(1):54–67, 2000.

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

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

Abdelrhman Shaheen, Anas Badr, Ali Abohendy, Hatem Alsaadawy, Nadine Alsayad, and Ehab H El-Shazly. Reinforcement learning in strategy-based and atari games: A review of google deepminds innovations. *arXiv preprint arXiv:2502.10303*, 2025.

Mojtaba Sharifi, Shreesh Tripathi, Yun Chen, Qiang Zhang, and Mahdi Tavakoli. Reinforcement learning methods for assistive and rehabilitation robotic systems: A survey. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2025.

Paul J Silvia. Curiosity and motivation. *The Oxford Handbook of Human Motivation*, pp. 157–166, 2012.

Thet Htar Su, Shaswot Shresthamali, and Masaaki Kondo. Quantum framework for reinforcement learning: Integrating the markov decision process, quantum arithmetic, and trajectory search. *Physical Review A*, 111(6):062421, 2025.

Bhavya Sukhija, Lenart Treven, Carmelo Sferrazza, Florian Dorfler, Pieter Abbeel, and Andreas Krause. Optimism via intrinsic rewards: Scalable and principled exploration for model-based reinforcement learning. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. *Robotica*, 17(2): 229–235, 1999.

Lukasz Szpruch, Tanut Treetanthiploet, and Yufei Zhang. Exploration-exploitation trade-off for continuous-time episodic reinforcement learning with linear-convex models. *The Annals of Applied Probability*, 36(2):959–1001, 2026.

Adrien Ali Taiga, William Fedus, Marlos C. Machado, Aaron Courville, and Marc G. Bellemare. On bonus based exploration methods in the arcade learning environment. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJewlyStDr.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *Annual Review of Control, Robotics, and Autonomous Systems*, 8(1):153–188, 2025.

Michel Tokic. Adaptive ε-greedy exploration in reinforcement learning based on value differences. In *Annual conference on artificial intelligence*, pp. 203–210. Springer, 2010.

Carel van Niekerk, Renato Vukovic, Benjamin Matthias Ruppik, Hsien-chin Lin, and Milica Gašić. Post-training large language models via reinforcement learning from self-feedback. *arXiv preprint arXiv:2507.21931*, 2025.

Renye Yan, Zhe Wu, Yuan Zhan, Pin Tao, Zongwei Wang, Yimao Cai, and Junliang Xing. Mnemonic dictionary learning for intrinsic motivation in reinforcement learning. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. IEEE, 2023.

Renye Yan, You Wu, Yaozhong Gan, Yunfan Yang, Zhaoke Yu, Zongxi Liu, Xin Zhang, Ling Liang, and Yimao Cai. Autoencoder reconstruction model for long-horizon exploration. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2024.

RenYe Yan, Jikang Cheng, Yaozhong Gan, Shikun Sun, You Wu, Yunfan Yang, Liang Ling, Jinlong Lin, Yeshuang Zhu, Jie Zhou, et al. Entropy-adaptive diffusion policy optimization with dynamic step alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1924–1934, 2025.

Daochen Zha, Kwei-Herng Lai, Kaixiong Zhou, and Xia Hu. Experience replay optimization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 4243–4249, 2019.

Enmin Zhao, Renye Yan, Jinqiu Li, Kai Li, and Junliang Xing. Alphaholdem: High-performance artificial intelligence for heads-up no-limit poker via end-to-end reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 4689–4697, 2022.

Xiujie Zhao, Piao Chen, and Loon Ching Tang. Condition-based maintenance via markov decision processes: A review. *Frontiers of Engineering Management*, 12(2):330–342, 2025.

A PRELIMINARIES

Markov Decision Process (MDP). A Markov Decision Process (MDP) (Zhao et al., 2025; Su et al., 2025; Rajasekhar et al., 2025;?), as described by Sutton and Barto (1999) and Puterman (2014), is defined by the tuple $\langle S, A, P, R, \gamma \rangle$, where S and A denote the sets of states and actions, respectively. The function P represents the state transition probabilities, describing the likelihood of transitioning from one state to another upon taking a specific action. The reward function R assigns a scalar value (often assumed non-negative) for each state-action pair, reflecting the immediate feedback received after the transition.

The discount factor $\gamma \in [0, 1)$ accounts for the agent's preference for immediate versus future rewards by adjusting the value of rewards over time. During interactions in the MDP, the agent generates sequences (trajectories) consisting of states, actions, and rewards: $\tau = (s_0, a_0, R(s_0, a_0)), \cdots, (s_t, a_t, R(s_t, a_t)), \cdots$. The value of a particular state-action pair under policy π, denoted as the state-action value function $Q^\pi(s_t, a_t)$, is calculated as follows:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{i=0}^{\infty} \gamma^i R(s_{t+i}, a_{t+i}) \mid s_t, a_t \right].$$

A policy defines the agent's behavior as a probability distribution over actions given the states. The policy $\pi(a_i \mid s)$ in this context is derived using a softmax function applied to the state-action value function:

$$\pi(a_i \mid s) = \frac{e^{Q(s,a_i)}}{\sum_k e^{Q(s,a_k)}}, \quad i = 1, 2, \cdots, n.$$

To ensure that the agent chooses actions that maximize its long-term accumulated rewards, policy optimization aims to derive a policy π that maximizes the expected cumulative discounted reward $\eta(\pi)$. This is given by:

$$\eta(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{i=0}^{\infty} \gamma^i R(s_i, a_i) \right].$$

The objective is to encourage the agent to make decisions that yield the highest total reward, balancing immediate and future returns.

Intrinsic-Based Reinforcement Learning. Extending the traditional Markov Decision Process (MDP) framework, intrinsic rewards Farooq et al. (2025); Monon et al. (2025); Nicholaus & Kang (2025) facilitate agent exploration, particularly in environments characterized by sparse extrinsic rewards. In such settings, relying solely on extrinsic feedback can lead to inefficient exploration and poor learning outcomes, as the agent may not receive enough signals to optimize its policy effectively.

Intrinsic rewards serve as an additional incentive for the agent, promoting the discovery of novel states and encouraging behaviors that might eventually lead to higher extrinsic rewards. The total reward function $R_{\text{total}}(s, a)$ is thus composed of both extrinsic rewards R_{ext} and intrinsic rewards R_{int}, which are combined to form a more comprehensive feedback mechanism:

$$R_{\text{total}}(s, a) = R_{\text{ext}}(s, a) + \alpha R_{\text{int}}(s, a).$$

The intrinsic reward Kayal et al. (2025); Sukhija et al. (2025); van Niekerk et al. (2025) $R_{\text{int}}(s, a)$ often reflects properties such as state novelty, prediction error, or curiosity. These intrinsic signals help agents maintain a high level of exploration, ensuring that less frequently visited states are considered, thus improving the learning of optimal policies over time. By incorporating intrinsic rewards, intrinsic-based reinforcement learning aims to balance exploration and exploitation, allowing agents to learn effectively even when extrinsic rewards are sparse or delayed. This approach significantly improves learning efficiency in complex environments where traditional methods struggle to provide sufficient guidance.

B ADDITIONAL PROOFS

B.1 PROOF OF THEOREM 1

Proof. Under Assumption 1, we define $C \triangleq (1-\gamma)\min_s[Q^*(s,a^*) - Q^*(s,a^{sub})]$. So, for any state s and action a, we omit (s,a), and have

$$R \leq R + \alpha R_{int} \leq R + C$$

Then

$$Q^*(s,a) \triangleq \max_\pi \mathbb{E}_\tau \left[\sum_t \gamma^t R_t | s,a\right] \leq \max_\pi \mathbb{E}_\tau \left[\sum_t \gamma^t (R_t + \alpha R_{int}) | s,a\right] \triangleq Q_1^*(s,a) \tag{6}$$

and

$$Q_1^*(s,a) = \max_\pi \mathbb{E}_\tau \left[\sum_t \gamma^t (R_t + \alpha R_{int}) | s,a\right] \leq \max_\pi \mathbb{E}_\tau \left[\sum_t \gamma^t (R_t + C) | s,a\right] \triangleq \hat{Q}^*(s,a) \tag{7}$$

According to the paper Gan et al. (2022), we have

$$\hat{Q}^*(s,a) \triangleq \max_\pi \mathbb{E}_\tau \left[\sum_t \gamma^t (R_t + C) | s,a\right] = Q^*(s,a) + \frac{C}{1-\gamma} \tag{8}$$

where Q^* is the optimal Q function. Since a^* and a^{sub} are optimal and suboptimal actions under the state s, respectively, we have $Q^*(s,a^*) > Q^*(s,a^{sub})$. Next, we will prove that $Q_1^*(s,a^*) > Q_1^*(s,a^{sub})$ holds.

For the optimal action a^* and suboptimal action a^{sub}, we have

$$Q^*(s,a^*) \leq Q_1^*(s,a^*) \leq \hat{Q}^*(s,a^*) = Q^*(s,a^*) + \frac{C}{1-\gamma}$$
$$Q^*(s,a^{sub}) \leq Q_1^*(s,a^{sub}) \leq \hat{Q}^*(s,a^{sub}) = Q^*(s,a^{sub}) + \frac{C}{1-\gamma}$$

By simplifying, we have

$$Q_1^*(s,a^*) - Q_1^*(s,a^{sub}) > Q^*(s,a^*) - Q^*(s,a^{sub}) - \frac{C}{1-\gamma}$$

By definition of C, we know $Q^*(s,a^*) - Q^*(s,a^{sub}) - \frac{C}{1-\gamma} > 0$ holds. Therefore, $Q_1^*(s,a^*) > Q_1^*(s,a^{sub})$ holds. □

B.2 PROOF OF THEOREM 2

Proof. Since the value of $C_\omega(s,a)$ is related to the state s and action a. Under state s, due to $C_\omega(s,a^*) > C_\omega(s,a)$, we can get

$$\begin{aligned}
&\sum_a I_{\{C_\omega(s,a)\geq\beta\}} \pi(a|s) Q^\pi(s,a) \\
\leq &I_{\{C_\omega(s,a^*)\geq\beta\}} \sum_a \pi(a|s) Q^\pi(s,a) \\
\leq &I_{\{C_\omega(s,a^*)\geq\beta\}} Q^\pi(s,a^*) \\
= &\sum_a I_{\{C_\omega(s,a)\geq\beta\}} \pi_\theta(a|s) Q^\pi(s,a)
\end{aligned}$$

Therefore, we get

$$
\begin{aligned}
V^{\pi}(s) &= \sum_{a} \pi(a|s) Q^{\pi}(s,a) \\
&= \sum_{a} \left(1 - I_{\{C_{\omega}(s,a) \geq \beta\}}\right) \pi(a|s) Q^{\pi}(s,a) + \sum_{a} I_{\{C_{\omega}(s,a) \geq \beta\}} \pi(a|s) Q^{\pi}(s,a) \\
&\leq \sum_{a} \left(1 - I_{\{C_{\omega}(s,a) \geq \beta\}}\right) \pi(a|s) Q^{\pi}(s,a) + \sum_{a} I_{\{C_{\omega}(s,a) \geq \beta\}} \pi^{*}(a|s) Q^{\pi}(s,a) \\
&= \sum_{a} \pi_{\text{new}}(a|s) Q^{\pi}(s) \\
&= E_{\pi_{\text{new}}}\{Q^{\pi}(s,a)|s_t = s\} \\
&= E_{\pi_{\text{new}}}\{r_t + \gamma V^{\pi}(s_{t+1})|s_t = s\} \\
&\leq E_{\pi_{\text{new}}}\{r_t + \gamma \sum_{a_{t+1}} \pi_{\text{new}}(a_{t+1}|s_{t+1}) Q^{\pi}(s_{t+1})|s_t = s\} \\
&\vdots \\
&\leq V^{\pi_{\text{new}}}(s).
\end{aligned}
$$

□

C Additional related works

C.1 Exploration–Exploitation Dilemma

The trade-off between exploration and exploitation is a core challenge in RL and has long constrained the performance of algorithms in complex environments. Although existing studies have attempted to alleviate this issue by using fixed weighting combinations or annealing mechanisms to adjust the exploration intensity, these methods fundamentally rely on static or decaying regulation and cannot adaptively adjust based on the decision-making needs of the current state. Consequently, agents often exhibit significant optimization conflicts during training, where exploration tends to drive the policy away from currently high-reward behaviors to cover unknown regions, whereas exploitation seeks to reinforce existing high-reward paths and suppress policy updates. These opposing optimization signals can lead to instability or even to the mutual cancellation of policy updates across stages. For example, overly aggressive exploration in the early stage makes it difficult to learn an effective policy, whereas excessive exploitation in the later stage limits the discovery of potentially better solutions. More critically, these methods generally lack the ability to leverage historical experience in a structured manner, making it difficult to extract decision-making guidance on "when to explore and when to exploit" from past interactions. As a result, the exploration-exploitation trade-off remains at the level of heuristic adjustment, failing to achieve truly adaptive synergy from the ground up.**

C.2 Memory-based RL

Current memory-based RL methods can be divided into two main categories, i.e., storage-based methods and experience replay. Storage-based methods, e.g., NGU Badia et al. (2020b) and BT-NGU Campos et al. (2021), retain past states and reuse the memory bank to help the agent discover novel states. However, these approaches are essentially state counting via a memory bank rather than carefully learning from past experiences. Experience replay methods, such as SIL Oh et al. (2018), DTSIL Guo et al. (2020) and ERO Zha et al. (2019), accelerate policy convergence by replaying past advantageous experiences. However, they often overlook the importance of exploring new policies and reflecting on failed experiences, which can lead to suboptimal policies.

In RL, policy optimization offers significant advantages in dense-reward environments, as it can directly optimize the policy function and leverage frequent reward signals to enable efficient policy updates, thereby converging quickly to an optimal policy. However, in sparse-reward environments, reward signals are limited and delayed, leading the agent to receive insufficient feedback most of

the time. This significantly increases the difficulty of policy evaluation, making policy updates unstable and inefficient. Sparse reward signals not only exacerbate the high-variance problem of policy gradient estimation but may also cause the policy to become trapped in suboptimal solutions or struggle to learn effective long-horizon policies, thereby undermining the performance of policy gradient methods in such environments.

Memory provides a promising direction for addressing sparse-reward problems, as humans learn from past experiences to avoid repeating mistakes in similar tasks Liu et al. (2021); Bransford & Johnson (1972); Andrychowicz et al. (2017). This fact suggests that leveraging experience facilitates efficient execution of new tasks, especially when rewards are sparse Pathak et al. (2017). Through memory mechanisms, an agent can leverage prior successful experiences to improve its policy in complex environments. This reduces its reliance on dense reward feedback, overcomes the limitations of traditional policy gradient methods in sparse-reward settings, and enhances both learning efficiency and policy stability.

Existing memory-based RL methods fall into two main categories, namely storage-based and experience replay. Storage-based methods, such as NGU, use a k-nearest-neighbor dynamic memory library to store embeddings of visited states and compute similarities to the current state. They then assign intrinsic rewards inversely proportional to these similarities, encouraging exploration of novel states while reducing revisits. However, this approach essentially implements state counting through a memory buffer and cannot leverage memory experience for prediction or policy guidance. Experience replay methods, such as self-imitation learning, reuse high-return experiences by repeatedly imitating state-action pairs whose returns exceed current value estimates. DTSIL further maintains a trajectory buffer to reproduce historical trajectories and guide exploration. Although these methods have achieved some success, they rely too heavily on existing policies, thereby limiting exploration of new policies and potentially leading to premature convergence to suboptimal solutions. Moreover, they overlook failed experiences, even though pitfalls often hide along the necessary paths of high-return trajectories. Learning to avoid dangers by reflecting on failures can increase the likelihood of approaching the optimal policy.

C.3 Existing Exploration Methods

Intrinsic motivation based on curiosity to guide exploration has drawn attention for addressing sparse and delayed rewards. These methods essentially measure state novelty and can be broadly categorized into two types: count-based and prediction error-based. Count-based methods Bellemare et al. (2016) track state visitation frequency and provide higher rewards for less-explored states. While simple and intuitive, they suffer from significant storage pressure in complex environments due to the large state space. In contrast, prediction error-based methods Burda et al. (2018b); Pathak et al. (2017) evaluate state density using neural networks and reward more on the low-density states. Hence, the experience of visited states can be propagated to nearby similar states. However, these methods evaluate similar states solely by visual appearance, leading to confusion about states that are nearly identical but have different potential values. A finer-grained metric that can distinguish the nuances among similar states is needed to improve intrinsic motivation.

C.4 Sparse Reward Environments

Sparse reward environments Hare (2019); Devidze et al. (2022) in RL are scenarios in which agents receive little to no feedback most of the time, receiving rewards only under specific conditions. This scarcity makes it difficult for agents to obtain sufficient guidance during training, leading traditional algorithms to perform poorly because they rely on frequent reward signals to update policies and value functions. To address this issue, researchers have proposed several approaches. First, improving exploration policies, such as using intrinsic rewards based on state novelty or prediction error, can encourage agents to explore previously unexplored regions. Second, imitation and inverse reinforcement learning are widely applied, allowing agents to learn from expert demonstrations and find optimal paths more efficiently. Additionally, memory-based methods replay and mimic past successful experiences, substituting sparse reward signals, accelerating policy optimization, and easing the difficulty of updates in sparse reward environments.

C.5 MEMORY MOTHEDS

Existing mainstream memory utilization methods can be broadly divided into memory storage and experience reuse. Below, we provide a more detailed overview of related work to help readers gain a better understanding.

Storage-based methods: Memory storage is the most straightforward and most intuitive method of memory utilization. It helps address the sparse reward problem by storing the agent's experienced states to measure state novelty. A typical example of this approach is the NGU Badia et al. (2020b) method, which aims to solve exploration challenges in sparse reward environments by storing past state experiences. NGU memorizes processed state experiences and uses a clustering-based method to manage the memory bank. By clustering visited states and using these cluster centers, the agent can identify and explore new states, providing additional intrinsic rewards based on state rarity. The agent receives higher intrinsic rewards when visiting rare states, encouraging the exploration of new areas. This memory comparison mechanism improves the agent's exploration efficiency and learning ability in sparse reward environments. However, the NGU algorithm has some limitations. First, managing and updating the cluster centers can become increasingly complex in high-dimensional state spaces, reducing computational efficiency. Additionally, clustering-based methods may incorrectly group similar states with different values in high-dimensional spaces, overlooking critical nuances and affecting policy optimization. Finally, memory storage methods mainly focus on constructing state counts via memory banks rather than using stored experiences to predict and guide future policies.

Experience Replay methods: Imitation learning Oh et al. (2018); Guo et al. (2020) based on experience replay is a memory utilization approach that centers on using the agent's past successful experiences to guide current policy optimization. Specifically, as the agent interacts with the environment, it stores information about its experienced states, actions, and corresponding rewards in a replay buffer. By analyzing this historical experience, particularly those associated with high rewards, the agent learns to imitate successful behaviors, reinforcing the positive effects of these experiences during policy optimization. Through replaying and mimicking high-reward actions, the agent improves decision-making quality, reduces reliance on random exploration, and accelerates policy convergence and improvement. By effectively leveraging valuable past experiences, such algorithms provide a more stable and efficient optimization path, helping the agent quickly learn and adapt to complex environments. Despite significant progress in imitation learning based on experience replay, this approach has three notable limitations. First, these methods place too much emphasis on reproducing past high-value behaviors while falling short in exploring new policies. This can weaken the agent's exploration capabilities, causing it to miss the global optimal policy and become stuck in local optima. Second, these methods primarily focus on leveraging successful experiences while neglecting the value of failed experiences. Failed experiences are equally important for policy optimization. Human learning can extract valuable insights from both positive and negative experiences. In reinforcement learning, paths with high rewards often hide potential risks or pitfalls. Thus, effectively using failed experiences to avoid repeated mistakes is crucial for improving policy optimization. Third, these methods focus on imitating past successful behaviors rather than truly understanding the experiences and using them to predict future decisions. As a result, the agent often merely replicates past behavior patterns rather than using those experiences to anticipate future scenarios and outcomes.

C.6 INTRINSIC MOTIVATION

In addressing the problem of sparse and delayed rewards, intrinsic motivation models Ryan & Deci (2000); Badia et al. (2020a); Oudeyer & Kaplan (2007) have gained increasing attention for their ability to enhance exploration efficiency. Intrinsic motivation methods simulate intrinsic drives in organisms, such as curiosity, to guide agents in exploration, thereby sustaining exploration efforts in environments lacking clear extrinsic rewards. These approaches can be broadly categorized into two types: count-based policies and prediction error-based policies.

Count-based Intrinsic Motivation: Count-based policies Bellemare et al. (2016) track the frequency of an agent's state visits, using a decreasing function of state counts as exploration rewards.

The basic principle is that the agent should receive higher rewards for visiting less frequently explored states, encouraging exploration of under-explored areas. The simplicity and intuitiveness of this mechanism have made it widely applicable in simple reinforcement learning environments. However, as the complexity of the environment increases, particularly in high-dimensional state spaces, the number of states grows rapidly. In such cases, the cost of maintaining and updating state counts significantly rises. Specifically, state counting requires substantial memory to record the visitation frequency of each state. Additionally, frequently updating the counts in dynamic environments may decrease computational efficiency. This increases resource consumption and may slow the agent's real-time decision-making, reducing its practicality in complex environments.

Error-based Intrinsic Motivation: Prediction error-based policies Burda et al. (2018b); Pathak et al. (2017); Burda et al. (2018a); Achiam & Sastry (2017); Silvia (2012) leverage neural network errors to assess state visit density, forming curiosity-driven exploration policies. Specifically, these methods calculate the agent's prediction error in a given state to estimate how frequently the state has been visited. Visits to low-density states are encouraged, while visits to high-density states are reduced, promoting exploration of unknown or novel states. This approach eliminates the dependence on storage and computational efficiency inherent in count-based intrinsic reward methods. It significantly improves exploration quality in the early stages of a task, allowing the agent to gather more information and experience. By this mechanism, the agent can more flexibly address exploration challenges in complex environments, actively seeking under-explored states. This policy's effectiveness lies in broadening exploration and enabling the agent to adapt to environmental changes, thereby improving learning efficiency quickly. However, existing intrinsic motivation models often focus on enhancing exploration quality while paying less attention to exploitation. This tendency can lead to diminishing exploration returns in later stages as the agent's interaction with the environment deepens, causing exploration efficiency to decline significantly—a phenomenon known as the "intrinsic motivation fading problem" Barto et al. (2013); Oudeyer & Kaplan (2007). Additionally, current prediction error-based intrinsic motivation methods tend to overlook subtle differences between similar states, which can result in missed opportunities to explore critical states and, consequently, potential policy improvements.

D Supplementary Experiment Details

D.1 Visualization of Experimental Environments Setup

Dark Chamber. To demonstrate the exploration capabilities of various algorithms in a reward-free environment, we constructed the *Dark Chamber* environment, as shown in Figure 12. This environment is 50×50 in size, with no rewards or penalties. The agent starts exploring from the bottom left, where unexplored areas are represented as black fog, and the color of explored areas gradually lightens based on exploration density.

Four Rooms. To further evaluate the agent's exploration ability in a complex environment with sparse rewards and obstacles, we selected the *Four Rooms* environment for comparative visualization experiments, as shown in Figure 13. The agent starts from point S in the upper right, explores the environment, avoids the cross-shaped wall in the middle, and passes through the small doorway to reach the goal in the lower left for a reward.

Cliff Walking. The Cliff Walking environment, filled with cliff traps, was selected as the visualization platform to evaluate the memory-reflection mechanism, as shown in Figure 4(a). In this environment, the gray areas represent cliffs, and the agent must move from point S to point G. To succeed, the agent needs to learn to avoid dangers while reaching goal G using a relatively short path, making this environment highly suitable for verifying the effectiveness of the memory-reflection mechanism.

D.2 Algorithm Description

The algorithmic process of MemoRL can be seen in Algorithm 1

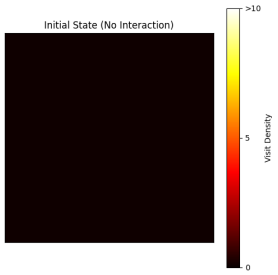


Figure 12: **Dark Chamber environment** This environment contains no rewards or traps and is designed as a testing platform to evaluate the algorithm's exploration capabilities under reward-free conditions.

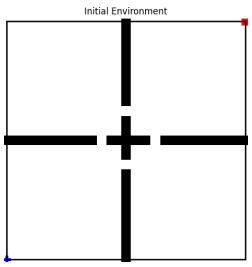


Figure 13: **Four Rooms environment** The environment features a central obstacle of four walls, with only one door the same size as the agent for it to pass through. The agent starts at the red point in the top right corner and must navigate through the obstacles to reach the blue point in the bottom left corner to obtain the only reward. As a setting with sparse rewards and obstacles, this environment is ideal for evaluating an algorithm's ability to explore challenging scenarios.

D.3 Parameter Settings

Table 5 presents detailed parameter settings for the primary experimental platform, Atari.

Table 5: Hyperparameter Table

Hyperparameter	Value	Hyperparameter	Value
MaxStepPerEpisode	4500	Entropy	0.001
ExtCoef	2.0	Epoch	4
LearningRate	1e-4	MiniBatch	4
NumEnv	32	PPOEps	0.1
NumStep	128	IntCoef	1.0
Gamma	0.999	StickyAction	True
IntGamma	0.99	ActionProb	0.25
Lambda	0.95	UpdateProportion	0.25
StableEps	1e-8	ObsNormStep	50
StateStackSize	4	ReliabilityThreshold	0.85
PreProcHeight	84	M-BankSize	10
ProProcWidth	84	R-BankSize	5000
UseGAE	True	M-BankBatchSize	1
UseGPU	True	R-BankBatchSize	128
UseNorm	False	CommonBufferSize	100000
UseNoisyNet	False	OriPolicyEnvNum	16
ClipGradNorm	0.5	ExploitUpdate	50

Algorithm 1 MemoRL: Retrospective-Prospective Ensemble Reinforcement Learning

Require: Environment $\mathcal{E}$, base RL policy π, memory policy π_θ, memory bank M, reflection bank R, Reflection Network C_ω, reliability threshold β, intrinsic reward coefficient α, encoder f_θ, decoder g_ϕ
Initialize: Initialize policy π, memory bank M, reflection bank R, feature encoder f_θ, reconstruction decoder g_ϕ, and Reflection Network C_ω
for $t = 0, 1, 2, \ldots$ **do**
    1. Trajectory Collection
    Collect trajectories τ using the current ensemble policy π_{new}
    2. Retrospective Module
    Success-Based Exploitation:
    Update advantageous trajectory memory bank M using trajectory return R_i and trajectory length N_i
    Train the Digest Network using state-action pairs from M:

$$\mathcal{L}_{\text{digest}}(\theta) = -\sum_{(s,a)\in M} \log \pi_\theta(a|s)$$

    Failure-Based Reflection:
    Store failure-related state-action pairs into reflection bank R
    Train the Reflection Network $C_\omega(s, a)$:

$$\mathcal{L}_C(\omega) = \sum_{(s,a)\in M} \|C_\omega(s,a) - 1\|^2 + \sum_{(s,a)\in R} \|C_\omega(s,a) - 0\|^2$$

    3. Prospective Module
    Encode state s_t using $f_\theta(s_t)$ and reconstruct:

$$\hat{s}_t = g_\phi(f_\theta(s_t))$$

    Compute coarse intrinsic reward using reconstruction error
    Compute fine-grained distinction reward using sparse latent regularization
    Compute intrinsic reward:

$$R^{\text{int}} = \frac{1}{2}\|s_t - g_\phi(f_\theta(s_t))\|_2^2 + \lambda\|f_\theta(s_t)\|_1$$

    Update exploration policy π using intrinsic reward R^{int}
    4. Ensemble Learning for Exploration-Exploitation Balance
    Obtain memory action a_{mem} from π_θ
    Compute confidence score:

$$C_\omega(s, a_{\text{mem}})$$

    Construct ensemble policy:

$$\pi_{\text{new}} = (1 - I\{C_\omega(s, a_{\text{mem}}) \geq \beta\})\pi + I\{C_\omega(s, a_{\text{mem}}) \geq \beta\}\pi_\theta$$

    if $C_\omega(s, a_{\text{mem}}) \geq \beta$ **then**
        Execute exploitation action from π_θ
    else
        Execute exploration action from π
    end if
    5. Policy Optimization
    Update the underlying RL policy π using collected trajectories and intrinsic reward R^{int}
end for