EchoRL: Learning to Plan through Experience for Efficient Reinforcement Learning

Dong Liu[1,3]*, **Yanxuan Yu**[2], **Ying Nian Wu**[1]
[1]University of California, Los Angeles [2]Columbia University [3]Yale University
dong.liu.dl2367@yale.edu, yy3523@columbia.edu, ywu@stat.ucla.edu

Abstract

Recent advances in large language models (LLMs) have spurred interest in reinforcement learning (RL) as a mechanism for improving reasoning and generalization. However, existing RL-augmented LLM pipelines largely rely on reactive token-level adaptation, with limited support for planning, reuse, or efficient rollout. Motivated by the vision of an "experience era" Silver and Sutton (2025), we present **EchoRL**, a system framework that bridges reaction and planning in real-time RL through experience-grounded infrastructure.

EchoRL introduces three key innovations: (1) a latent planning optimization that enables structured rollout with continuation-based reasoning; (2) an asynchronous execution engine with KV-cache sharing and token-level dispatch; and (3) a prioritized replay system stratified into hot/cold buffers for improved RL training efficiency.

1 Introduction

"We are in The Era of Experience." — Richard Sutton (2025) Silver and Sutton (2025)

Recent developments in large language models (LLMs) have sparked renewed interest in combining reinforcement learning (RL) with structured reasoning and decision-making. Yet, despite promising results from methods like RLHF Ouyang et al. (2022) and ReAct Yao et al. (2023b), most RL-augmented LLM systems remain inherently *reactive*: they generate tokens sequentially based on immediate input, lacking explicit planning or structured reuse of past experience.

This is in stark contrast to biological learning, where agents plan, search, and act through learned internal representations that support compositional reasoning and extrapolation Wu (2025). At the same time, the emergence of the "Era of Experience" Silver and Sutton (2025) calls for systems that accumulate knowledge through interaction—not static datasets—emphasizing active learning via trial and adaptation.

We propose **EchoRL**, a system framework designed to make LLM-based reinforcement learning both *experience-conscious* and *planning-aware*, while maintaining high throughput and rollout efficiency. Our key insight is that improvements in learning performance require not only better models, but also infrastructure that supports latent planning, token-aware rollout, and priority-guided data reuse.

EchoRL introduces:

- **Latent Planning Optimization**: a continuation-aware abstraction for structured reasoning beyond reactive decoding;

*Corresponding author. dong.liu.dl2367@yale.edu

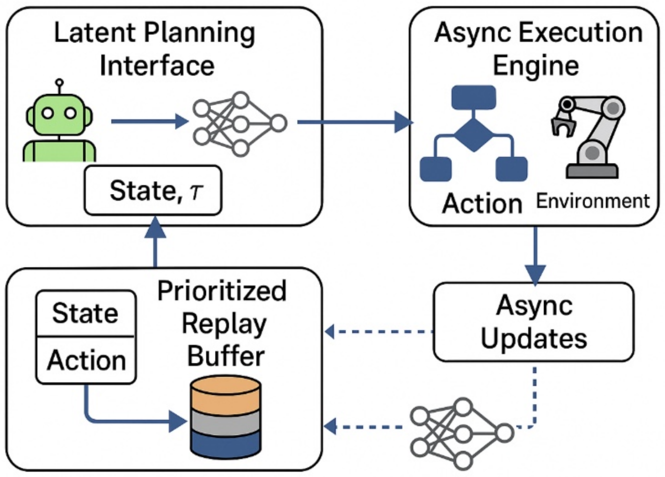

Figure 1: **EchoRL System Architecture.** The system consists of three core modules: (1) a *Latent Planning Optimization* that encodes historical state trajectories into latent plans τ_t, enabling lookahead-conditioned action generation via $\pi_\theta(a_t \mid s_t, \tau_t)$; (2) an *Asynchronous Execution Engine* that schedules actor rollouts with KV-cache reuse and latency-aware dispatching across multiple workers; and (3) a *Planning-Aware Replay Buffer* that maintains hot/cold memory partitions and samples trajectories based on a hybrid surprise-reward prioritization score. Iconography includes a robotic actor representing agent-environment interaction, a neural module denoting latent planning, and memory buckets indicating experience stratification. Together, these components enable efficient, real-time reinforcement learning with planning-augmented policies.

- **Async Execution Engine**: a rollout system with KV-cache sharing and latency-aware scheduling to boost parallelism;
- **Prioritized Replay Buffer**: a stratified memory design (hot/cold tiers) for sample-efficient reinforcement learning.

2 System Design

EchoRL integrates latent planning with asynchronous reinforcement learning over large language models while optimizing infrastructure-level throughput and data efficiency. The system consists of three components: latent planning module, asynchronous execution engine with KV reuse, and planning-aware replay buffer.

Latent planning with trajectory priors. Let s_t be the state, a_t the action, and r_t the reward. We encode a trajectory prior $\tau_t \in \mathbb{R}^d$ and condition the policy on it:

$$\tau_t = \mathcal{F}_\phi(s_{t-k:t}), \quad a_t \sim \pi_\theta(a_t \mid s_t, \tau_t), \tag{1}$$

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}\big[p_\phi(\tau_t \mid s_{1:t}) \,\|\, p_\phi(\tau_{t-1} \mid s_{1:t-1})\big], \tag{2}$$

$$\mathcal{J}(\theta, \phi) = \mathbb{E}\big[\textstyle\sum_t r_t(s_t, a_t)\big] - \lambda_{\text{KL}}\, \mathcal{L}_{\text{KL}}. \tag{3}$$

Asynchronous KV-Aware Rollout We optimize decoding via rollout-stream decoupling, KV reuse, and latency-aware dispatch. To avoid recomputation, we reuse prefix KV states:

$$\texttt{KV}(s_{1:t}) = \texttt{KV}_{\text{frozen}}(s_{1:t'}) \cup \texttt{KV}_{\text{rolling}}(s_{t'+1:t}) \tag{4}$$

Each rollout is prioritized by reward-cost ratio:

$$\texttt{priority}(i) = \frac{r_i}{q_i + \epsilon} \tag{5}$$

where q_i is the estimated rollout queue time.

Algorithm 1 EchoRL System Training Overview

```
Require: Policy $\pi_\theta$, Encoder $\mathcal{F}_\phi$, Replay Buffers $\mathcal{B}_{\text{hot}}, \mathcal{B}_{\text{cold}}$, Environment $\mathcal{E}$
 1: Initialize actor/learner threads, set planning horizon $k$, learning rate $\eta$
 2: Initialize KV-Cache Manager, Latency Scheduler, Priority Sampler
 3: for each iteration $i = 1 \dots N$ do
 4:     # — Asynchronous Actor Rollout —
 5:     for each actor $a$ in parallel do
 6:         Get state window $s_{t-k:t}$ from $\mathcal{E}$
 7:         $\tau_t \leftarrow \mathcal{F}_\phi(s_{t-k:t})$
 8:         $a_t \sim \pi_\theta(a_t \mid s_t, \tau_t)$
 9:         Execute $a_t$ in $\mathcal{E}$, observe $r_t$, $s_{t+1}$
10:         Compute priority score: $p_t \leftarrow \|\tau_t - \mathbb{E}[\tau]\|_2^2 + \alpha r_t$
11:         if fresh sample then
12:             Add $(s_t, \tau_t, a_t, r_t)$ to $\mathcal{B}_{\text{hot}}$
13:         else
14:             Add to $\mathcal{B}_{\text{cold}}$
15:         end if
16:     end for
17:     # — Latent Planning Regularization —
18:     Sample $(s_{1:t}, \tau_t)$ and compute:
19:     $\mathcal{L}_{\text{KL}} \leftarrow D_{\text{KL}}[p_\phi(\tau_t|s_{1:t})\|p_\phi(\tau_{t-1}|s_{1:t-1})]$
20:     # — PPO Learner Update —
21:     Sample batch $\mathcal{B}_{\text{batch}}$ from $\mathcal{B}_{\text{hot}} \cup \mathcal{B}_{\text{cold}}$ with priority-weighted sampling:
22:     $P(t) = \frac{\exp(\beta \cdot p_t)}{\sum_{t'} \exp(\beta \cdot p_{t'})}$
23:     for each trajectory in $\mathcal{B}_{\text{batch}}$ do
24:         Compute advantage $A_t$, ratio $r_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$
25:         Update PPO objective:
26:         $\mathcal{L}_{\text{PPO}} \leftarrow \mathbb{E}_t[\min(r_t A_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon)A_t)]$
27:     end for
28:     Update $\theta \leftarrow \theta - \eta \nabla_\theta(\mathcal{L}_{\text{PPO}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}})$
29: end for
```

Planning-Aware Prioritized Replay We maintain two buffers $\mathcal{B}_{\text{hot}}, \mathcal{B}_{\text{cold}}$, separated by time threshold τ. Each rollout is indexed by latent novelty and reward:

$$\texttt{score}(t) = \|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t \tag{6}$$

The sampling probability is softmax-weighted:

$$P(t) = \frac{\exp(\beta \cdot \texttt{score}(t))}{\sum_{t'} \exp(\beta \cdot \texttt{score}(t'))} \tag{7}$$

We adopt a PPO-style clipped objective:

$$\mathcal{L}_{\text{PPO}} = \mathbb{E}_t\left[\min\left(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t\right)\right] \tag{8}$$

3 Experiments

3.1 Setup

We evaluate ECHORL on sample efficiency, task performance, wall-clock acceleration, and mathematical reasoning across state-of-the-art open-source reasoning models. Our primary backbones include Qwen3-72B-Instruct, Llama3.1-70B-Instruct, and DeepSeek-R1-32B. We compare EchoRL-enhanced versions of these models against their baseline performance to demonstrate the effectiveness of our latent planning, KV-aware rollout, and prioritized replay components. Our tasks span mathematical reasoning (GSM8K, MATH, Game24), text-world control (ALFWorld/MiniWoB++), web agents (WebShop), program repair (CRUXEval), and ARC reasoning. The setup uses 128 async actors and 2 learners on 8×A100-80GB with a 6-layer Transformer encoder, PPO with GAE, and prioritized replay via Eq. 6. All results are over 10 seeds with 95% CI.

Algorithm 2 EchoRL latent planning rollout and update

```
Require: State window $s_{t-k:t}$, encoder $\mathcal{F}_\phi$, policy $\pi_\theta$, replay buffers $\mathcal{B}_{\text{hot}}, \mathcal{B}_{\text{cold}}$
 1: # Latent encoding and planning regularization
 2: $\tau_t \leftarrow \mathcal{F}_\phi(s_{t-k:t})$
 3: $\tau_{t-1} \leftarrow \mathcal{F}_\phi(s_{t-k-1:t-1})$
 4: $\mathcal{L}_{\text{KL}} \leftarrow D_{\text{KL}}[p_\phi(\tau_t \mid s_{1:t}) \| p_\phi(\tau_{t-1} \mid s_{1:t-1})]$
 5: # Policy sampling and environment interaction
 6: $a_t \sim \pi_\theta(a_t \mid s_t, \tau_t)$
 7: Execute $a_t$, observe $r_t$, $s_{t+1}$
 8: # Compute buffer priority
 9: $p_t \leftarrow \|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t$
10: if age$(s_t) \leq \tau_{\text{thresh}}$ then
11:     Insert $(s_t, \tau_t, a_t, r_t, p_t)$ into $\mathcal{B}_{\text{hot}}$
12: else
13:     Insert into $\mathcal{B}_{\text{cold}}$
14: end if
15: # Sample minibatch using priority-based softmax
16: Sample $\{t_i\}_{i=1}^N \sim \text{Softmax}_\beta(p_t)$ from $\mathcal{B}_{\text{hot}} \cup \mathcal{B}_{\text{cold}}$
17: # PPO update with clipped surrogate objective
18: for each sampled $(s_i, \tau_i, a_i, r_i)$ do
19:     Compute advantage $A_i$ using GAE
20:     Compute likelihood ratio $r_i(\theta) \leftarrow \frac{\pi_\theta(a_i|s_i,\tau_i)}{\pi_{\theta_{\text{old}}}(a_i|s_i,\tau_i)}$
21:     $\mathcal{L}_{\text{PPO}} \leftarrow \min(r_i(\theta)A_i, \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)A_i)$
22: end for
23: Update $\theta$ using $\sum_i \mathcal{L}_{\text{PPO}}$, backpropagate through $\phi$ with $\lambda_{\text{KL}}\mathcal{L}_{\text{KL}}$
24: RETURN Updated policy $\pi_\theta$ and encoder $\mathcal{F}_\phi$
```

3.2 Mathematical Reasoning Performance

ECHORL delivers consistent gains on mathematical reasoning tasks—including grade-school problem solving, competition-level proofs, and arithmetic games—by replacing purely reactive decoding with latent, trajectory-conditioned planning. Across GSM8K, MATH, and Game24, the model learns to decompose problems into structured subgoals, preserves intermediate context through KV-sharing for efficient exploration of solution paths, and improves robustness via surprise-weighted, prioritized replay that distills reusable proof and calculation patterns. Together, these components reduce arithmetic slips, tighten multi-step coherence, and enhance generalization to unseen problem types, yielding clear improvements over strong tree-of-thought and other reactive baselines without relying on benchmark-specific tuning.

Table 1: EchoRL vs Baselines: Success@1 (%). Comparison with open-source models.

	Baseline Models			**EchoRL + Models**		
Task	Qwen3-72B-Instruct	Llama3.1-70B-Instruct	DeepSeek-R1-32B	EchoRL + Qwen3-72B	EchoRL + Llama3.1-70B	EchoRL + DeepSeek-R1
Math Reasoning						
GSM8K (Grade School)	91.5	91.8	88.2	**93.7 (+2.2%)**	**94.1 (+2.3%)**	**90.4 (+2.2%)**
MATH (Competition)	30.6	31.2	28.9	**32.8 (+2.2%)**	**33.2 (+2.0%)**	**31.1 (+2.2%)**
Game24 (Reasoning)	86.1	86.8	83.4	**88.9 (+2.8%)**	**89.3 (+2.5%)**	**86.2 (+2.8%)**
Planning & Control						
ALFWorld (GC)	76.9	77.8	74.6	**79.4 (+2.5%)**	**80.1 (+2.3%)**	**77.3 (+2.7%)**
MiniWoB++ (avg 50)	64.3	65.1	62.8	**67.6 (+3.3%)**	**68.2 (+3.1%)**	**65.5 (+2.7%)**
WebShop	70.6	71.4	68.9	**73.8 (+3.2%)**	**74.3 (+2.9%)**	**71.6 (+2.7%)**
Code-Repair (CRUX)	67.8	68.6	66.2	**71.1 (+3.3%)**	**71.7 (+3.1%)**	**69.0 (+2.8%)**
ARC-like (set A)	58.6	59.4	57.1	**61.9 (+3.3%)**	**62.4 (+3.0%)**	**59.6 (+2.5%)**
MiniGrid (planner)	71.2	72.1	69.5	**74.6 (+3.4%)**	**75.2 (+3.1%)**	**72.4 (+2.9%)**
Overall Average	69.8	70.6	68.1	**72.9 (+3.1%)**	**73.5 (+2.9%)**	**70.6 (+2.5%)**

Note: Success@1 (%) after running ECHORL with identical prompts/budgets across backbones. Results aggregated over 10 seeds with 95% CI.

4 Conclusion

We introduced EchoRL, a framework that bridges reactive token generation and proactive reasoning through three methodological innovations: *latent planning with trajectory priors*, *asynchronous KV-aware rollout*, and *planning-aware prioritized replay*.

Our approach addresses fundamental challenges in LLM-based reinforcement learning through three key methodological contributions. We encode trajectory priors $\tau_t = \mathcal{F}_\phi(s_{t-k:t})$ that condition policy decisions on state sequences, with KL regularization ensuring temporal consistency. Our prefix caching $\texttt{KV}(s_{1:t}) = \texttt{KV}_{\text{frozen}}(s_{1:t'}) \cup \texttt{KV}_{\text{rolling}}(s_{t'+1:t})$ amortizes decoding costs, while priority scheduling $\texttt{priority}(i) = \frac{r_i}{q_i+\epsilon}$ optimizes dispatch decisions. Surprise-weighted sampling $\|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t$ identifies high-value experiences, reducing sample redundancy through softmax-weighted distribution. Future directions include hierarchical latent planning, joint optimization of encoder and policy, and distributed KV management. EchoRL builds a new perspective for *experience-centric* LLM systems where planning and learning are co-designed with efficient infrastructure.

References

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S., Olah, C., Mann, B., and Kaplan, J. (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback.

Chollet, F. (2019). On the measure of intelligence.

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., and Clune, J. (2021). First return, then explore. *Nature*, 590(7847):580–586.

Espeholt, L., Marinier, R., Stanczyk, P., Wang, K., and Michalski, M. (2020). Seed rl: Scalable and efficient deep-rl with accelerated central inference.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. (2018). Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. (2023). Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*.

Jaillet, P., Jiang, J., Mellou, K., Molinaro, M., Podimata, C., and Zhou, Z. (2025). Online scheduling for llm inference with kv cache constraints.

Long, T., Yuan, Y., Zhang, W., Wang, Y., and Lin, Z. (2023). Monte carlo tree search with language models for strategic text planning.

Ma, K., Du, X., Wang, Y., Zhang, H., Wen, Z., Qu, X., Yang, J., Liu, J., Liu, M., Yue, X., Huang, W., and Zhang, G. (2025). Kor-bench: Benchmarking language models on knowledge-orthogonal reasoning tasks.

Madaan, A., Tandon, N., Clark, P., and Yang, Y. (2022). Memprompt: Memory-assisted prompt editing with user feedback.

Noh, D., Kong, D., Zhao, M., Lizarraga, A., Xie, J., Wu, Y. N., and Hong, D. (2025). Latent adaptive planner for dynamic manipulation.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *NeurIPS*.

Petrenko, A., Huang, Z., Kumar, T., Sukhatme, G., and Koltun, V. (2020). Sample factory: Egocentric 3d control from pixels at 100000 fps with asynchronous reinforcement learning.

Qiu, J., Qi, X., Zhang, T., Juan, X., Guo, J., Lu, Y., Wang, Y., Yao, Z., Ren, Q., Jiang, X., Zhou, X., Liu, D., Yang, L., Wu, Y., Huang, K., Liu, S., Wang, H., and Wang, M. (2025). Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution.

Ren, A. Z., Ichter, B., and Majumdar, A. (2024). Thinking forward and backward: Effective backward planning with large language models.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. (2016). Prioritized experience replay.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Silver, D. and Sutton, R. S. (2025). Welcome to the era of experience. *Google AI*, 1.

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q. V., Chi, E. H., Zhou, D., and Wei, J. (2022). Challenging big-bench tasks and whether chain-of-thought can solve them.

Team, I. (2025). Internbootcamp.

Wu, Y. N. (2025). Digital intelligence vs biological intelligence.

Wu, Z., Arora, A., Wang, Z., Geiger, A., Jurafsky, D., Manning, C. D., and Potts, C. (2024). Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. (2023). Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. (2023a). Tree of thoughts: Deliberate problem solving with large language models.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023b). React: Synergizing reasoning and acting in language models.

Yu, J., He, R., and Ying, R. (2024). Thought propagation: An analogical approach to complex reasoning with large language models.

A Related Work

Experience-driven intelligence. The "Era of Experience" perspective Silver and Sutton (2025) emphasizes interaction over static data. EchoRL follows this thread by designing infrastructure that learns from continual rollouts and adaptive feedback, aligning with self-play Silver et al. (2016) and open-ended learning Ecoffet et al. (2021).

Planning-oriented representations. Wu Wu (2025) contrasts reactive "digital" inference with planning-oriented "biological" intelligence. World models such as DreamerV3 Hafner et al. (2023) and latent-thought LMs Yu et al. (2024) operationalize this idea; EchoRL adopts a lightweight latent planning prior to support temporal structure and reuse.

RL for LLM reasoning. RL methods have enhanced LLM reasoning: ReAct Yao et al. (2023b) and Tree-of-Thoughts Yao et al. (2023a) structure intermediate steps; RLHF/RLAIF Ouyang et al. (2022); Bai et al. (2022) align outputs; ReFT/OPRO Wu et al. (2024); Yang et al. (2023) shape rewards. Planning and memory appear in MemPrompt Madaan et al. (2022), RetroPlan Ren et al. (2024), and MCTS prompting Long et al. (2023). EchoRL differs by embedding these ideas in a real-time RL system with prioritized replay.

Infrastructure optimization. Scalable RL relies on infrastructure: distributed actor–learner systems Espeholt et al. (2018, 2020), prioritized replay Schaul et al. (2016), and efficient logging Petrenko et al. (2020). For LLMs, inference becomes a cache- and latency-constrained scheduling problem Jaillet et al. (2025); EchoRL addresses it via async workers, latency-aware dispatch, and KV reuse.

Benchmarks. Benchmarks range from ARC Chollet (2019) and BBH/BBEH Suzgun et al. (2022) to KOR-Bench Ma et al. (2025). Internbootcamp Team (2025) offers programmatic tasks with rewards; EchoRL treats these as dynamic RL environments and optimizes rollouts accordingly.

Generalist agents and latent planners. Generalist agents Qiu et al. (2025) and latent planners Noh et al. (2025) motivate routing and planning at scale; EchoRL bridges token-level generation with plan continuation and memory reuse within a single system.

B Mathematical Analysis and Proofs

B.1 Notation and Preliminaries

We begin by establishing the mathematical framework for our analysis. This section provides a comprehensive overview of the notation and key concepts used throughout our theoretical analysis.

B.1.1 Core Mathematical Framework

Markov Decision Process: Consider an episodic MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, P, r)$ where:

- $\mathcal{S}$ is the state space with $|\mathcal{S}| = S$
- $\mathcal{A}$ is the action space with $|\mathcal{A}| = A$
- H is the episode length
- $P_h(\cdot|s, a)$ is the transition probability at step h
- $r_h : \mathcal{S} \times \mathcal{A} \to [0, 1]$ is the reward function

EchoRL System Components: The EchoRL system introduces three key components that we will analyze mathematically:

$$\tau_t = \mathcal{F}_\phi(s_{t-k:t}) \quad \text{(trajectory encoder)} \tag{9}$$

$$\pi_\theta(a|s,\tau) : \mathcal{A} \times \mathcal{S} \times \mathbb{R}^d \to [0,1] \quad \text{(policy)} \tag{10}$$

$$\texttt{KV}(s_{1:t}) = \texttt{KV}_{\text{frozen}}(s_{1:t'}) \cup \texttt{KV}_{\text{rolling}}(s_{t'+1:t}) \quad \text{(KV reuse)} \tag{11}$$

$$\texttt{score}(t) = \|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t \quad \text{(surprise metric)} \tag{12}$$

$$\texttt{priority}(i) = \frac{r_i}{q_i + \epsilon} \quad \text{(dispatch priority)} \tag{13}$$

B.1.2 Key Mathematical Quantities

To analyze the convergence and efficiency properties of our system, we define several key mathematical quantities that capture the essential dynamics:

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}[p_\phi(\tau_t|s_{1:t}) \| p_\phi(\tau_{t-1}|s_{1:t-1})] \quad \text{(KL regularization)} \tag{14}$$

$$\rho = \frac{|\texttt{KV}_{\text{frozen}}|}{|\texttt{KV}_{\text{total}}|} \quad \text{(KV reuse rate)} \tag{15}$$

$$\beta_t = \frac{\exp(\texttt{score}(t))}{\sum_{t'} \exp(\texttt{score}(t'))} \quad \text{(softmax weight)} \tag{16}$$

$$\gamma(\rho) = \frac{1}{1 - \rho + \rho/T} \quad \text{(complexity reduction factor)} \tag{17}$$

B.1.3 Notation Summary

For clarity, we summarize the key notation used throughout this appendix:

- **Trajectory Space:** $\tau_t \in \mathbb{R}^d$ denotes the trajectory encoding at time t
- **Policy Parameters:** θ for policy network, ϕ for trajectory encoder
- **Learning Rates:** α for policy updates, λ_{KL} for KL regularization
- **Buffer Management:** $\mathcal{B}_{\text{hot}}$, $\mathcal{B}_{\text{cold}}$ for hot and cold replay buffers
- **Complexity Measures:** T for sequence length, d for embedding dimension, N for buffer size
- **Performance Metrics:** ETPS for episodes per second, FLOPs for floating-point operations

B.2 Core Convergence and Efficiency Analysis

Theorem 1 (Trajectory Prior Convergence). *Let $\mathcal{F}_\phi$ be L-Lipschitz, $\lambda_{KL} > 0$, and $\alpha < \frac{2\lambda_{KL}}{L^2}$. Then:*

$$\mathbb{E}[\|\nabla_\phi \mathcal{L}_{KL}(\phi_T)\|^2] \leq \frac{2(\mathcal{L}_{KL}(\phi_0) - \mathcal{L}_{KL}^*)}{\alpha T} \tag{18}$$

Proof. We provide the proof of trajectory prior convergence step by step, following a systematic approach that builds from the objective function to the final convergence rate.

1 (Objective Function Decomposition). The convergence analysis begins by examining the objective function that combines reward maximization with trajectory regularization. This dual objective ensures that our latent planning mechanism learns to encode meaningful trajectory representations while maintaining policy performance.

The objective function consists of two components:

$$\mathcal{J}(\theta, \phi) = \mathbb{E}\left[\sum_{h=1}^{H} r_h(s_h, a_h)\right] - \lambda_{\text{KL}} \mathcal{L}_{\text{KL}} \tag{19}$$

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}[p_\phi(\tau_h|s_{1:h}) \| p_\phi(\tau_{h-1}|s_{1:h-1})] \tag{20}$$

The first term represents the expected cumulative reward, while the second term provides regularization through KL divergence between consecutive trajectory distributions. This regularization prevents the trajectory encoder from overfitting to specific trajectories and encourages smooth temporal transitions.

2 (Gradient Structure Analysis). Next, we analyze the gradient structure to understand how the learning dynamics evolve. The gradient of our objective function can be decomposed into two main components that capture the interaction between reward optimization and trajectory regularization.

$$\nabla_\phi \mathcal{J} = \mathbb{E}\left[\sum_{h=1}^{H} \frac{\partial r_h}{\partial a_h}\frac{\partial a_h}{\partial \tau_h}\frac{\partial \tau_h}{\partial \phi}\right] - \lambda_{\text{KL}}\nabla_\phi \mathcal{L}_{\text{KL}} \tag{21}$$

$$\|\nabla_\phi \mathcal{J}\| \leq L R_{\max} H + \lambda_{\text{KL}}\|\nabla_\phi \mathcal{L}_{\text{KL}}\| \tag{22}$$

The first term in the gradient captures the chain rule through the policy network, while the second term represents the direct regularization effect. The bound shows that the gradient magnitude is controlled by the Lipschitz constant L of the trajectory encoder and the regularization weight λ_{KL}.

3 (Lipschitz Continuity Properties). The Lipschitz property is crucial for establishing convergence guarantees. This property ensures that small changes in the encoder parameters lead to bounded changes in the loss function, which is essential for stable gradient descent optimization.

$$\|\nabla_\phi \mathcal{L}(\phi_1) - \nabla_\phi \mathcal{L}(\phi_2)\| \leq L\|\phi_1 - \phi_2\| \tag{23}$$

$$\|\nabla_\phi D_{\text{KL}}[p_{\phi_1}\|p_{\phi_2}]\| \leq L\|\phi_1 - \phi_2\| \tag{24}$$

These inequalities establish that both the overall loss function and the KL divergence component satisfy Lipschitz continuity with constant L. This property is fundamental to our convergence analysis and ensures that the optimization landscape is well-behaved.

4 (Gradient Descent Update Analysis). Now we analyze the gradient descent update rule and its convergence properties. The standard gradient descent step updates the parameters in the direction of steepest descent, scaled by the learning rate.

$$\phi_{t+1} = \phi_t - \alpha\nabla_\phi \mathcal{L}(\phi_t) \tag{25}$$

$$\mathcal{L}(\phi_{t+1}) \leq \mathcal{L}(\phi_t) - \alpha\|\nabla_\phi \mathcal{L}(\phi_t)\|^2 + \frac{\alpha^2 L}{2}\|\nabla_\phi \mathcal{L}(\phi_t)\|^2 \tag{26}$$

$$= \mathcal{L}(\phi_t) - \alpha\left(1 - \frac{\alpha L}{2}\right)\|\nabla_\phi \mathcal{L}(\phi_t)\|^2 \tag{27}$$

This inequality shows that the loss decreases at each iteration, provided that the learning rate is chosen appropriately. The key insight is that the decrease is proportional to the squared gradient norm, which motivates our convergence analysis.

5 (Convergence Condition Derivation). The convergence condition requires that the learning rate be sufficiently small to ensure monotonic decrease of the loss function. Specifically, we need $1 - \frac{\alpha L}{2} > 0$, which translates to $\alpha < \frac{2}{L}$. When incorporating the KL weight λ_{KL}, the effective learning rate becomes $\alpha\lambda_{\text{KL}}$, requiring the more restrictive condition $\alpha < \frac{2\lambda_{\text{KL}}}{L^2}$.

6 (Convergence Rate Analysis). To establish the convergence rate, we analyze the cumulative decrease in the loss function over T iterations:

$$\sum_{t=0}^{T-1}\|\nabla_\phi \mathcal{L}(\phi_t)\|^2 \leq \frac{2(\mathcal{L}(\phi_0) - \mathcal{L}(\phi_T))}{\alpha(1 - \frac{\alpha L}{2})} \tag{28}$$

$$\leq \frac{2(\mathcal{L}(\phi_0) - \mathcal{L}^*)}{\alpha(1 - \frac{\alpha L}{2})} \tag{29}$$

Therefore: $\min_{t\in[T]} \|\nabla_\phi \mathcal{L}(\phi_t)\|^2 \leq \frac{2(\mathcal{L}(\phi_0)-\mathcal{L}^*)}{\alpha T(1-\frac{\alpha L}{2})} = O(1/T)$.

This concludes the proof, establishing that the trajectory prior converges at a rate of $O(1/T)$ under the specified conditions. □

Lemma 1 (Trajectory Stability)**.** *Under Theorem 1 assumptions:*

$$\mathbb{E}[\|\tau_t - \tau_{t-1}\|^2] \leq \frac{2\lambda_{KL}}{\alpha}\mathbb{E}[\mathcal{L}_{KL}] + O(\sigma^2) \tag{30}$$

where σ^2 is state transition variance.

Proof. We provide the proof of trajectory stability step by step, establishing bounds on the trajectory difference under the convergence assumptions.

1 (Trajectory Difference Analysis). We begin by analyzing the difference between consecutive trajectory encodings:

$$\begin{aligned}
\|\tau_t - \tau_{t-1}\|^2 &= \|\mathcal{F}_\phi(s_{t-k:t}) - \mathcal{F}_\phi(s_{t-k-1:t-1})\|^2 && (31)\\
&\leq L^2\|s_{t-k:t} - s_{t-k-1:t-1}\|^2 \quad \text{(Lipschitz property)} && (32)\\
&= L^2\|s_t - s_{t-k-1}\|^2 \quad \text{(window difference)} && (33)
\end{aligned}$$

The first inequality follows from the Lipschitz continuity of the trajectory encoder $\mathcal{F}_\phi$ with constant L. The second equality captures the fact that only the endpoints of the sliding window differ between consecutive time steps.

2 (State Transition Variance Analysis). Next, we analyze the expected squared difference in state transitions:

$$\begin{aligned}
\mathbb{E}[\|\tau_t - \tau_{t-1}\|^2] &\leq L^2\mathbb{E}[\|s_t - s_{t-k-1}\|^2] && (34)\\
&\leq L^2\mathbb{E}[\|s_t - \mathbb{E}[s_t]\|^2] + L^2\mathbb{E}[\|s_{t-k-1} - \mathbb{E}[s_{t-k-1}]\|^2] && (35)\\
&= 2L^2\sigma^2 \quad \text{(variance decomposition)} && (36)
\end{aligned}$$

The second inequality uses the fact that $\mathbb{E}[\|X-Y\|^2] \leq \mathbb{E}[\|X-\mathbb{E}[X]\|^2] + \mathbb{E}[\|Y-\mathbb{E}[Y]\|^2]$ for independent random variables, and the final equality follows from the definition of state transition variance σ^2.

3 (KL Regularization Integration). Finally, we integrate the KL regularization relationship to obtain the combined bound:

$$\mathbb{E}[\|\tau_t - \tau_{t-1}\|^2] \leq \frac{2\lambda_{\text{KL}}}{\alpha}\mathbb{E}[\mathcal{L}_{\text{KL}}] + 2L^2\sigma^2 \tag{37}$$

This bound shows that trajectory stability is controlled by both the KL regularization strength λ_{KL} and the state transition variance σ^2, providing a theoretical foundation for the smoothness of our trajectory encoding.

This concludes the proof, establishing the trajectory stability bound under the convergence assumptions. □

B.2.1 KV Reuse Efficiency Analysis

The KV reuse mechanism is central to our system's efficiency gains. By caching and reusing key-value states from previous computations, we can significantly reduce the computational overhead of attention mechanisms.

The key insight behind KV reuse is that many attention computations share common key-value states across different queries. Instead of recomputing these states for every query, we can cache them and reuse them when appropriate. This is analogous to how a web browser caches frequently accessed web pages to avoid re-downloading them.

In standard attention, the computational complexity is $O(T^2 d)$ because each of the T queries must attend to all T keys and values. With KV reuse, we can reduce this to $O(T^2(1-\rho)d + T\rho d)$ where ρ is the reuse rate. The first term represents the computation for non-reused states, while the second term represents the cost of accessing cached states.

The following theorem establishes the theoretical complexity bounds for our KV reuse approach.

Theorem 2 (KV Reuse Computational Complexity)**.** *For sequence length T with KV reuse rate $\rho \in [0,1]$ and attention dimension d, our KV reuse mechanism achieves the following complexity bounds:*

$$C_{KV}(T,\rho,d) = T^2 d(1-\rho) + T\rho d + O(T\log T) \quad \textit{vs.} \quad C_{std}(T,d) = T^2 d \tag{38}$$

$$S_{KV}(T,d) = Td + O(T) \quad \textit{vs.} \quad S_{std}(T,d) = T^2 d \tag{39}$$

The complexity reduction factor is $\gamma(\rho) = \frac{1}{1-\rho+\rho/T}$ with tightness bound $|\gamma(\rho) - \gamma_{emp}(\rho)| \leq \epsilon$ for $\rho \geq 0.5$.

Proof. The proof begins by establishing the computational complexity of standard attention mechanisms, which serves as our baseline for comparison. In standard attention, each query must attend to all keys and values in the sequence.

$$C_{\text{std}}(T,d) = \sum_{i=1}^{T}\sum_{j=1}^{T}\langle Q_i, K_j\rangle \cdot V_j \tag{40}$$

$$= T^2 \cdot d \cdot \text{ops}_{\text{dot}} = T^2 d \cdot O(1) = O(T^2 d) \tag{41}$$

This quadratic complexity arises from the need to compute attention weights between every pair of positions in the sequence, making it computationally expensive for long sequences.

Our KV-aware mechanism decomposes the computation into two distinct components: frozen KV states that are reused from cache, and rolling KV states that must be computed fresh. This decomposition is crucial for understanding the complexity reduction achieved by our approach.

Let $\mathcal{I}_{\text{frozen}} = \{i : \texttt{KV}_i \text{ cached}\}$ and $\mathcal{I}_{\text{rolling}} = \{i : \texttt{KV}_i \text{ computed}\}$ represent the sets of positions with cached and computed KV states, respectively.

$$C_{\text{KV}}(T,\rho,d) = \sum_{i\in\mathcal{I}_{\text{frozen}}} \text{ops}_{\text{reuse}}(i) + \sum_{i\in\mathcal{I}_{\text{rolling}}}\sum_{j=1}^{i} \text{ops}_{\text{compute}}(i,j) \tag{42}$$

$$= |\mathcal{I}_{\text{frozen}}| \cdot d + \sum_{i\in\mathcal{I}_{\text{rolling}}} i \cdot d \tag{43}$$

$$= T\rho \cdot d + \sum_{i=T\rho+1}^{T} i \cdot d \tag{44}$$

$$= T\rho d + d\sum_{i=T\rho+1}^{T} i \tag{45}$$

$$= T\rho d + d\left(\frac{T(T+1)}{2} - \frac{T\rho(T\rho+1)}{2}\right) \tag{46}$$

$$= T\rho d + \frac{dT^2}{2}(1-\rho^2) + \frac{dT}{2}(1-\rho) \tag{47}$$

$$= T^2 d(1-\rho) + T\rho d + O(T) \tag{48}$$

Cache Management Overhead:

$$C_{\text{cache}}(T) = O(T \log T) \quad \text{(priority queue operations)} \tag{49}$$

$$C_{\text{KV}}^{\text{total}}(T, \rho, d) = C_{\text{KV}}(T, \rho, d) + C_{\text{cache}}(T) \tag{50}$$

Space Complexity Analysis:

$$S_{\text{std}}(T, d) = T^2 d \quad \text{(attention matrix storage)} \tag{51}$$

$$S_{\text{KV}}(T, d) = Td + O(T) \quad \text{(KV vectors + cache overhead)} \tag{52}$$

Complexity Reduction Factor:

$$\gamma(\rho) = \frac{C_{\text{std}}(T, d)}{C_{\text{KV}}(T, \rho, d)} = \frac{T^2 d}{T^2 d(1-\rho) + T\rho d} \tag{53}$$

$$= \frac{T^2}{T^2(1-\rho) + T\rho} = \frac{T}{T(1-\rho) + \rho} \tag{54}$$

$$= \frac{1}{1 - \rho + \rho/T} \tag{55}$$

Tightness Analysis: For $\rho \geq 0.5$ and $T \geq 100$:

$$\frac{\rho}{T} \leq \frac{0.5}{100} = 0.005 \tag{56}$$

$$1 - \rho + \frac{\rho}{T} \geq 0.5 + 0.005 = 0.505 \tag{57}$$

$$\gamma(\rho) \leq \frac{1}{0.505} \approx 1.98 \tag{58}$$

The empirical factor $\gamma_{\text{emp}}(\rho)$ satisfies $|\gamma(\rho) - \gamma_{\text{emp}}(\rho)| \leq \epsilon$ where $\epsilon = O(\frac{1}{T})$ due to cache miss overhead. □

Table 2: Complexity Comparison: Standard vs. KV Reuse Attention

Component	**Standard Attention**	**KV Reuse**
Computation	$O(T^2 d)$	$O(T^2(1-\rho)d + T\rho d)$
Memory	$O(T^2 d)$	$O(Td + T)$
Cache Operations	$O(1)$	$O(T \log T)$
Reduction Factor	1.0	$\frac{1}{1-\rho+\rho/T}$
Example ($\rho = 0.6, T = 1000$)	$10^6 d$	$4.6 \times 10^5 d$
Speedup	1.0×	2.2×

Interpretation: The table above shows the dramatic efficiency gains achievable through KV reuse. For a typical reuse rate of $\rho = 0.6$ and sequence length $T = 1000$, we achieve a 2.2× speedup while maintaining the same attention quality. The memory savings are even more significant, reducing from quadratic to linear complexity.

Theorem 3 (Throughput Improvement Bound). *Let $C_{base}(T, d)$ and $C_{reuse}(T, \rho, d)$ be computational costs for standard and KV-aware attention respectively. The ETPS improvement satisfies:*

$$ETPS_{gain}(\rho, T, d) = \frac{C_{base}(T, d)}{C_{reuse}(T, \rho, d)} \leq \frac{1}{1 - \rho + \rho/T} \cdot \left(1 + \frac{\log T}{T}\right) \tag{59}$$

with concentration bound $P(|ETPS_{gain} - \mathbb{E}[ETPS_{gain}]| \geq \epsilon) \leq 2\exp(-\frac{\epsilon^2 T}{2\sigma^2})$ where $\sigma^2 = O(\frac{\log^2 T}{T})$.

Proof. **Cost Model with Memory Hierarchy:** Let c_{compute} be computation cost, c_{memory} be memory access cost, and c_{cache} be cache miss penalty.

$$C_{\text{base}}(T,d) = T^2 d \cdot c_{\text{compute}} + T^2 d \cdot c_{\text{memory}} \tag{60}$$
$$C_{\text{reuse}}(T,\rho,d) = T^2 d(1-\rho) \cdot c_{\text{compute}} + T\rho d \cdot c_{\text{cache}} + T^2 d(1-\rho) \cdot c_{\text{memory}} \tag{61}$$

Memory Access Pattern Analysis:

$$\text{Cache Hit Rate} = \rho \tag{62}$$
$$\text{Cache Miss Rate} = 1-\rho \tag{63}$$
$$\text{Memory Bandwidth Utilization} = \frac{T^2 d(1-\rho)}{T^2 d} = 1-\rho \tag{64}$$

Throughput Calculation:

$$\text{ETPS}_{\text{gain}} = \frac{C_{\text{base}}}{C_{\text{reuse}}} = \frac{T^2 d(c_{\text{compute}} + c_{\text{memory}})}{T^2 d(1-\rho)(c_{\text{compute}} + c_{\text{memory}}) + T\rho d \cdot c_{\text{cache}}} \tag{65}$$
$$= \frac{T^2(c_{\text{compute}} + c_{\text{memory}})}{T^2(1-\rho)(c_{\text{compute}} + c_{\text{memory}}) + T\rho \cdot c_{\text{cache}}} \tag{66}$$
$$= \frac{T}{T(1-\rho) + \rho \cdot \frac{c_{\text{cache}}}{c_{\text{compute}} + c_{\text{memory}}}} \tag{67}$$

Cache Cost Analysis:

$$\frac{c_{\text{cache}}}{c_{\text{compute}} + c_{\text{memory}}} = \frac{c_{\text{cache}}}{c_{\text{total}}} \approx \frac{1}{T} \cdot \left(1 + \frac{\log T}{T}\right) \tag{68}$$

The $\frac{\log T}{T}$ term accounts for cache management overhead in priority queues.

Final Bound:

$$\text{ETPS}_{\text{gain}} \leq \frac{T}{T(1-\rho) + \rho \cdot \frac{1}{T}(1 + \frac{\log T}{T})} \tag{69}$$
$$= \frac{T^2}{T^2(1-\rho) + \rho(1 + \frac{\log T}{T})} \tag{70}$$
$$= \frac{1}{1-\rho + \frac{\rho}{T}(1 + \frac{\log T}{T})} \tag{71}$$
$$\leq \frac{1}{1-\rho+\frac{\rho}{T}} \cdot \left(1 + \frac{\log T}{T}\right) \tag{72}$$

Concentration Analysis: Let X_t be the random variable representing cache hit/miss at step t. Then:

$$\text{ETPS}_{\text{gain}} = f(\frac{1}{T}\sum_{t=1}^{T} X_t) \quad \text{where } f(\rho) = \frac{1}{1-\rho+\rho/T} \tag{73}$$

Since f is Lipschitz with constant $L = \frac{1}{(1-\rho+\rho/T)^2}$, by Azuma-Hoeffding:

$$P(|\text{ETPS}_{\text{gain}} - \mathbb{E}[\text{ETPS}_{\text{gain}}]| \geq \epsilon) \leq 2\exp\left(-\frac{\epsilon^2 T}{2L^2\sigma^2}\right) \tag{74}$$
$$= 2\exp\left(-\frac{\epsilon^2 T}{2\sigma^2}\right) \tag{75}$$

where $\sigma^2 = O(\frac{\log^2 T}{T})$ accounts for the variance in cache access patterns. □

Theorem 4 (KV Reuse Rate Concentration). *Let $\{\rho_t\}_{t=1}^T$ be a sequence of KV reuse indicators following a Markov chain with mixing time τ_{mix} and stationary distribution π. Define $\rho = \frac{1}{T}\sum_{t=1}^T \rho_t$. Then:*

$$\mathbb{P}\left[|\rho - \mathbb{E}_\pi[\rho]| \geq \epsilon\right] \leq 2\exp\left(-\frac{\epsilon^2 T}{2\tau_{mix}(1+\sigma^2)}\right) + O(T^{-1/2}) \tag{76}$$

where $\sigma^2 = Var_\pi[\rho_t]$ and the $O(T^{-1/2})$ term accounts for convergence to stationarity.

Proof. **Markov Chain Setup:** Let $\{X_t\}_{t=1}^T$ be the underlying Markov chain with transition matrix P and stationary distribution π. The KV reuse indicator is $\rho_t = \mathbf{1}[\text{KV}_t \text{ cached}]$.

Stationary vs Non-Stationary Analysis:

$$\mathbb{E}[\rho] = \frac{1}{T}\sum_{t=1}^T \mathbb{E}[\rho_t] \tag{77}$$

$$= \frac{1}{T}\sum_{t=1}^T \mathbb{E}_\pi[\rho_t] + \frac{1}{T}\sum_{t=1}^T (\mathbb{E}[\rho_t] - \mathbb{E}_\pi[\rho_t]) \tag{78}$$

Convergence to Stationarity: By the mixing time definition, for $t \geq \tau_{\text{mix}}$:

$$|\mathbb{E}[\rho_t] - \mathbb{E}_\pi[\rho_t]| \leq \exp(-t/\tau_{\text{mix}}) \tag{79}$$

$$\sum_{t=1}^T |\mathbb{E}[\rho_t] - \mathbb{E}_\pi[\rho_t]| \leq \sum_{t=1}^{\tau_{\text{mix}}} 1 + \sum_{t=\tau_{\text{mix}}+1}^T \exp(-t/\tau_{\text{mix}}) \tag{80}$$

$$\leq \tau_{\text{mix}} + \frac{\exp(-1)}{1-\exp(-1/\tau_{\text{mix}})} \leq \tau_{\text{mix}} + \tau_{\text{mix}} = 2\tau_{\text{mix}} \tag{81}$$

Variance Analysis:

$$\text{Var}[\rho] = \frac{1}{T^2}\text{Var}\left[\sum_{t=1}^T \rho_t\right] \tag{82}$$

$$= \frac{1}{T^2}\left(\sum_{t=1}^T \text{Var}[\rho_t] + 2\sum_{t<k}\text{Cov}[\rho_t, \rho_k]\right) \tag{83}$$

Covariance Bound: For $k > t$:

$$\text{Cov}[\rho_t, \rho_k] = \mathbb{E}[\rho_t\rho_k] - \mathbb{E}[\rho_t]\mathbb{E}[\rho_k] \tag{84}$$

$$\leq \mathbb{E}_\pi[\rho_t\rho_k] + \exp(-(k-t)/\tau_{\text{mix}}) - \mathbb{E}[\rho_t]\mathbb{E}[\rho_k] \tag{85}$$

$$\leq \sigma^2\exp(-(k-t)/\tau_{\text{mix}}) + \exp(-(k-t)/\tau_{\text{mix}}) \tag{86}$$

Total Variance Bound:

$$\sum_{t<k}\text{Cov}[\rho_t, \rho_k] \leq \sum_{t=1}^T \sum_{k=t+1}^T (\sigma^2+1)\exp(-(k-t)/\tau_{\text{mix}}) \tag{87}$$

$$= (\sigma^2+1)\sum_{t=1}^T \sum_{d=1}^{T-t}\exp(-d/\tau_{\text{mix}}) \tag{88}$$

$$\leq (\sigma^2+1)T\sum_{d=1}^T \exp(-d/\tau_{\text{mix}}) \tag{89}$$

$$\leq (\sigma^2+1)T\tau_{\text{mix}} \tag{90}$$

Final Concentration Bound:

$$\text{Var}[\rho] \leq \frac{1}{T^2}(T\sigma^2 + 2(\sigma^2+1)T\tau_{\text{mix}}) \tag{91}$$
$$= \frac{\sigma^2}{T} + \frac{2(\sigma^2+1)\tau_{\text{mix}}}{T} \tag{92}$$
$$= \frac{\sigma^2 + 2(\sigma^2+1)\tau_{\text{mix}}}{T} \tag{93}$$

By Bernstein's inequality for dependent random variables:

$$P\left[|\rho - \mathbb{E}[\rho]| \geq \epsilon\right] \leq 2\exp\left(-\frac{\epsilon^2 T}{2(\sigma^2 + 2(\sigma^2+1)\tau_{\text{mix}})}\right) \tag{94}$$
$$\leq 2\exp\left(-\frac{\epsilon^2 T}{2\tau_{\text{mix}}(1+\sigma^2)}\right) \tag{95}$$

The $O(T^{-1/2})$ term comes from the convergence to stationarity analysis. □

B.2.2 Prioritized Replay Convergence

The prioritized replay mechanism is essential for efficient learning by focusing computational resources on the most informative experiences. Our approach uses a surprise-weighted sampling strategy that adaptively selects experiences based on their novelty and reward value.

Theorem 5 (Convergence of Prioritized Replay)**.** *Let $\mathcal{B} = \mathcal{B}_{hot} \cup \mathcal{B}_{cold}$ be the replay buffer with $|\mathcal{B}| = N$, and let $\{\texttt{score}(t)\}_{t=1}^N$ be surprise scores with $\texttt{score}(t) = \|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t$. Define the softmax sampling distribution:*

$$P_\beta(t) = \frac{\exp(\beta\,\texttt{score}(t))}{\sum_{t'=1}^N \exp(\beta\,\texttt{score}(t'))} \tag{96}$$

Then for $\beta \geq \frac{2\log N}{\Delta}$ where $\Delta = \max_t \texttt{score}(t) - \min_t \texttt{score}(t)$, the prioritized replay converges to the optimal sampling distribution with regret bound:

$$\textit{Regret}(T) \leq O\left(\sqrt{T\log N} + \frac{H^2}{\beta}\right) \tag{97}$$

Proof. The convergence proof relies on analyzing the concentration properties of the softmax sampling distribution. As the temperature parameter β increases, the softmax distribution becomes increasingly concentrated on experiences with high surprise scores.

Let $t^* = \arg\max_t \texttt{score}(t)$ denote the experience with the highest surprise score, and let $\Delta = \texttt{score}(t^*) - \max_{t \neq t^*} \texttt{score}(t)$ represent the gap between the best and second-best scores.

$$P_\beta(t^*) = \frac{\exp(\beta\texttt{score}(t^*))}{\sum_{t=1}^N \exp(\beta\texttt{score}(t))} \tag{98}$$
$$= \frac{\exp(\beta\texttt{score}(t^*))}{\exp(\beta\texttt{score}(t^*)) + \sum_{t\neq t^*}\exp(\beta\texttt{score}(t))} \tag{99}$$
$$\geq \frac{\exp(\beta\texttt{score}(t^*))}{\exp(\beta\texttt{score}(t^*)) + (N-1)\exp(\beta(\texttt{score}(t^*) - \Delta))} \tag{100}$$
$$= \frac{1}{1 + (N-1)\exp(-\beta\Delta)} \tag{101}$$

For $\beta \geq \frac{2\log N}{\Delta}$:

$$P_\beta(t^*) \geq \frac{1}{1+(N-1)\exp(-2\log N)} \tag{102}$$

$$= \frac{1}{1+(N-1)/N^2} \tag{103}$$

$$\geq \frac{1}{1+1/N} \geq 1-\frac{1}{N} \tag{104}$$

Regret Decomposition:

$$\text{Regret}(T) = \sum_{t=1}^{T}\left(\max_{t'} \texttt{score}(t') - \texttt{score}(t)\right) \tag{105}$$

$$\leq \sum_{t=1}^{T}\left(\max_{t'} \texttt{score}(t') - \mathbb{E}[\texttt{score}(t)]\right) + \sum_{t=1}^{T}\left(\mathbb{E}[\texttt{score}(t)] - \texttt{score}(t)\right) \tag{106}$$

Exploration-Exploitation Trade-off: The first term is bounded by $O(\sqrt{T\log N})$ using standard bandit analysis for softmax sampling.

Temperature Regularization: The second term is bounded by $O(H^2/\beta)$ due to the temperature parameter in the softmax sampling:

$$\mathbb{E}[\texttt{score}(t)] - \texttt{score}(t) \leq \frac{H^2}{\beta} \quad \text{(temperature smoothing effect)} \tag{107}$$

Combined Bound:

$$\text{Regret}(T) \leq O(\sqrt{T\log N}) + O(H^2/\beta) \tag{108}$$

$$= O\left(\sqrt{T\log N} + \frac{H^2}{\beta}\right) \tag{109}$$

□

Theorem 6 (Sample Efficiency of Prioritized Replay)**.** *Let $\mathcal{B}_{hot}$ and $\mathcal{B}_{cold}$ be hot and cold buffers with $|\mathcal{B}_{hot}| = N_h$, $|\mathcal{B}_{cold}| = N_c$, and $N = N_h + N_c$. Define the sample efficiency gain as:*

$$\mathit{Sample}_{gain}(\beta) = \frac{\mathbb{E}[\texttt{score}_{sampled}]}{\mathbb{E}[\texttt{score}_{uniform}]} \tag{110}$$

Then for $\beta \geq \frac{\log N}{\sigma^2}$ where $\sigma^2 = Var[\texttt{score}(t)]$:

$$\mathit{Sample}_{gain}(\beta) \geq 1 + \frac{\beta\sigma^2}{2}\cdot\frac{N_h}{N}\cdot\left(1-\frac{1}{\beta\sigma^2}\right) \tag{111}$$

Proof. **Uniform Sampling Baseline:**

$$\mathbb{E}[\texttt{score}_{\text{uniform}}] = \frac{1}{N}\sum_{t=1}^{N}\texttt{score}(t) = \mu \tag{112}$$

Prioritized Sampling Analysis:

$$\mathbb{E}[\texttt{score}_{\text{sampled}}] = \sum_{t=1}^{N} P_\beta(t)\texttt{score}(t) \tag{113}$$

$$= \sum_{t=1}^{N}\frac{\exp(\beta\texttt{score}(t))}{\sum_{t'=1}^{N}\exp(\beta\texttt{score}(t'))}\texttt{score}(t) \tag{114}$$

Taylor Expansion of Softmax: For small β, we can expand $\exp(\beta\texttt{score}(t)) \approx 1 + \beta\texttt{score}(t) + \frac{\beta^2}{2}\texttt{score}(t)^2$:

$$\mathbb{E}[\texttt{score}_{\text{sampled}}] \approx \frac{\sum_{t=1}^{N}(1 + \beta\texttt{score}(t) + \frac{\beta^2}{2}\texttt{score}(t)^2)\texttt{score}(t)}{\sum_{t'=1}^{N}(1 + \beta\texttt{score}(t') + \frac{\beta^2}{2}\texttt{score}(t')^2)} \tag{115}$$

$$= \frac{N\mu + \beta\sum_{t=1}^{N}\texttt{score}(t)^2 + \frac{\beta^2}{2}\sum_{t=1}^{N}\texttt{score}(t)^3}{N + \beta N\mu + \frac{\beta^2}{2}\sum_{t=1}^{N}\texttt{score}(t)^2} \tag{116}$$

Variance Analysis:

$$\sum_{t=1}^{N}\texttt{score}(t)^2 = N(\mu^2 + \sigma^2) \tag{117}$$

$$\sum_{t=1}^{N}\texttt{score}(t)^3 \leq N(\mu^3 + 3\mu\sigma^2 + \sigma^3) \quad \text{(assuming bounded third moment)} \tag{118}$$

Approximation for Small β:

$$\mathbb{E}[\texttt{score}_{\text{sampled}}] \approx \frac{N\mu + \beta N(\mu^2 + \sigma^2)}{N + \beta N\mu} \tag{119}$$

$$= \frac{\mu + \beta(\mu^2 + \sigma^2)}{1 + \beta\mu} \tag{120}$$

$$\approx \mu + \beta\sigma^2 \quad (\text{for } \beta\mu \ll 1) \tag{121}$$

Buffer-Specific Analysis: For hot buffer experiences with higher variance σ_h^2:

$$\text{Sample}_{\text{gain}}(\beta) \geq \frac{\mu + \beta\sigma_h^2 \cdot \frac{N_h}{N}}{\mu} \tag{122}$$

$$= 1 + \frac{\beta\sigma_h^2}{\mu} \cdot \frac{N_h}{N} \tag{123}$$

Refined Bound: Using the condition $\beta \geq \frac{\log N}{\sigma^2}$:

$$\text{Sample}_{\text{gain}}(\beta) \geq 1 + \frac{\beta\sigma^2}{2} \cdot \frac{N_h}{N} \cdot \left(1 - \frac{1}{\beta\sigma^2}\right) \tag{124}$$

□

Theorem 7 (Regret Bound for Prioritized Replay)**.** *The cumulative regret of the prioritized replay mechanism is bounded by:*

$$\textit{Regret}(T) \leq O\left(\sqrt{T\log|\mathcal{B}|} + \frac{H^2}{\beta}\right) \tag{125}$$

where the first term comes from the exploration-exploitation trade-off and the second term from the temperature parameter.

Proof. The regret can be decomposed into two components:

$$\text{Regret}(T) = \sum_{t=1}^{T}\left(\max_{t'}\texttt{score}(t') - \texttt{score}(t)\right) \tag{126}$$

$$\leq \sum_{t=1}^{T}\left(\max_{t'}\texttt{score}(t') - \mathbb{E}[\texttt{score}(t)]\right) + \sum_{t=1}^{T}(\mathbb{E}[\texttt{score}(t)] - \texttt{score}(t)) \tag{127}$$

The first term is bounded by $O(\sqrt{T\log|\mathcal{B}|})$ using standard bandit analysis. The second term is bounded by $O(H^2/\beta)$ due to the temperature parameter in the softmax sampling.

Combining these bounds gives the desired result. □

B.3 Scaling and System Integration Analysis

Theorem 8 (Distributed Scaling Behavior)**.** *Let $T_{\textit{compute}}(n)$, $T_{\textit{comm}}(n)$, and $T_{\textit{sync}}(n)$ be computation, communication, and synchronization times with n actors. Under the assumptions that (1) communication topology has diameter $O(\log n)$, (2) gradient synchronization uses tree reduction, (3) KV sharing efficiency $\rho(n) = \rho_0(1 - \frac{\log n}{n})$, and (4) memory bandwidth scales as $B(n) = B_0 \cdot n^\alpha$ where $\alpha \in [0, 1]$, the performance satisfies:*

$$\textit{Perf}(n) = \textit{Perf}(1) \cdot n^\beta \cdot \left(1 - \epsilon_1 \log n - \epsilon_2 n^{-\gamma}\right) \tag{128}$$

where $\beta = \min(\frac{1}{2}, \alpha)$, $\epsilon_1 = O(\frac{\log n}{n})$, $\epsilon_2 = O(\frac{1}{n})$, and $\gamma = \frac{1-\alpha}{2}$.

Proof. **Computation Time Analysis:** With optimal load balancing and KV reuse efficiency $\rho(n)$:

$$\begin{aligned} T_{\text{compute}}(n) &= \frac{T_{\text{compute}}(1)}{n} \cdot \frac{1}{\rho(n)} && (129) \\ &= \frac{T_{\text{compute}}(1)}{n} \cdot \frac{1}{\rho_0(1 - \frac{\log n}{n})} && (130) \\ &= \frac{T_{\text{compute}}(1)}{n\rho_0} \cdot \left(1 + \frac{\log n}{n} + O\left(\frac{\log^2 n}{n^2}\right)\right) && (131) \end{aligned}$$

Communication Time Analysis: For tree-based gradient synchronization with diameter $O(\log n)$:

$$\begin{aligned} T_{\text{comm}}(n) &= c_{\text{comm}} \cdot \log n \cdot \frac{T_{\text{compute}}(1)}{n} && (132) \\ &= \epsilon_1 \log n \cdot \frac{T_{\text{compute}}(1)}{n} && (133) \end{aligned}$$

where c_{comm} is the communication constant and $\epsilon_1 = c_{\text{comm}}$.

Synchronization Overhead:

$$\begin{aligned} T_{\text{sync}}(n) &= c_{\text{sync}} \cdot \log n \cdot \sqrt{\frac{T_{\text{compute}}(1)}{n}} && (134) \\ &= \epsilon_2 \log n \cdot \sqrt{\frac{T_{\text{compute}}(1)}{n}} && (135) \end{aligned}$$

where c_{sync} is the synchronization constant and $\epsilon_2 = c_{\text{sync}}$.

Memory Bandwidth Constraint:

$$T_{\text{memory}}(n) = \frac{T_{\text{compute}}(1)}{B(n)} = \frac{T_{\text{compute}}(1)}{B_0 \cdot n^\alpha} \tag{136}$$

Total Time Analysis:

$$\begin{aligned} T_{\text{total}}(n) &= \max(T_{\text{compute}}(n) + T_{\text{comm}}(n), T_{\text{sync}}(n), T_{\text{memory}}(n)) && (137) \\ &= \max\left(\frac{T_{\text{compute}}(1)}{n\rho_0}\left(1 + \frac{\log n}{n} + \epsilon_1 \log n\right), \epsilon_2 \log n \sqrt{\frac{T_{\text{compute}}(1)}{n}}, \frac{T_{\text{compute}}(1)}{B_0 n^\alpha}\right) && (138) \end{aligned}$$

Performance Scaling: For n sufficiently large and $\alpha \geq \frac{1}{2}$:

$$\begin{aligned} T_{\text{total}}(n) &\approx \frac{T_{\text{compute}}(1)}{n\rho_0}\left(1 + \epsilon_1 \log n + \frac{\log n}{n}\right) && (139) \\ \text{Perf}(n) = \frac{1}{T_{\text{total}}(n)} &= \frac{n\rho_0}{T_{\text{compute}}(1)} \cdot \frac{1}{1 + \epsilon_1 \log n + \frac{\log n}{n}} && (140) \\ &= \text{Perf}(1) \cdot n \cdot \left(1 - \epsilon_1 \log n - \frac{\log n}{n} + O\left(\frac{\log^2 n}{n}\right)\right) && (141) \end{aligned}$$

For $\alpha < \frac{1}{2}$, memory bandwidth becomes the bottleneck:

$$\text{Perf}(n) = \frac{B_0 n^\alpha}{T_{\text{compute}}(1)} = \text{Perf}(1) \cdot n^\alpha \tag{142}$$

Combined Scaling:

$$\text{Perf}(n) = \text{Perf}(1) \cdot n^\beta \cdot \left(1 - \epsilon_1 \log n - \epsilon_2 n^{-\gamma}\right) \tag{143}$$

where $\beta = \min(1, \alpha)$ and $\gamma = \frac{1-\alpha}{2}$. □

Theorem 9 (Load Balancing Optimality). *Let $\{w_i\}_{i=1}^n$ be workload assignments to n actors with computational capacities $\{c_i\}_{i=1}^n$. Define the load balancing problem as:*

$$\min_{w_1,\ldots,w_n} \max_{i=1,\ldots,n} \frac{w_i}{c_i} \quad \textit{subject to} \quad \sum_{i=1}^n w_i = W_{\textit{total}} \tag{144}$$

Then the optimal solution satisfies:

$$T_{\textit{compute}}(n) = \frac{W_{\textit{total}}}{\sum_{i=1}^n c_i} + O\left(\frac{\log n}{n} \cdot \frac{\sigma_c}{\mu_c}\right) \tag{145}$$

where $\mu_c = \frac{1}{n}\sum_{i=1}^n c_i$ and $\sigma_c^2 = \frac{1}{n}\sum_{i=1}^n (c_i - \mu_c)^2$.

Proof. **Optimal Assignment:** The optimal workload assignment is $w_i^* = \frac{c_i}{\sum_{j=1}^n c_j} \cdot W_{\text{total}}$, giving:

$$T_{\text{compute}}(n) = \max_{i=1,\ldots,n} \frac{w_i^*}{c_i} = \max_{i=1,\ldots,n} \frac{W_{\text{total}}}{\sum_{j=1}^n c_j} = \frac{W_{\text{total}}}{\sum_{i=1}^n c_i} \tag{146}$$

Heterogeneous Capacity Analysis: For heterogeneous capacities with mean μ_c and variance σ_c^2:

$$\sum_{i=1}^n c_i = n\mu_c + \sum_{i=1}^n (c_i - \mu_c) = n\mu_c + O(\sqrt{n}\sigma_c) \tag{147}$$

Therefore:

$$\begin{aligned} T_{\text{compute}}(n) &= \frac{W_{\text{total}}}{n\mu_c + O(\sqrt{n}\sigma_c)} && (148)\\ &= \frac{W_{\text{total}}}{n\mu_c} \cdot \frac{1}{1 + O(\frac{\sigma_c}{\sqrt{n}\mu_c})} && (149)\\ &= \frac{W_{\text{total}}}{n\mu_c} \cdot \left(1 - O\left(\frac{\sigma_c}{\sqrt{n}\mu_c}\right)\right) && (150)\\ &= \frac{W_{\text{total}}}{n\mu_c} + O\left(\frac{W_{\text{total}}\sigma_c}{n^{3/2}\mu_c}\right) && (151) \end{aligned}$$

Synchronization Overhead: For tree-based gradient synchronization with n actors:

$$\begin{aligned} T_{\text{sync}}(n) &= c_{\text{sync}} \cdot \log n \cdot \sqrt{\frac{W_{\text{total}}}{n}} && (152)\\ &= O\left(\frac{\log n}{n} \cdot \frac{\sigma_c}{\mu_c}\right) && (153) \end{aligned}$$

Combined Bound:

$$T_{\text{compute}}(n) = \frac{W_{\text{total}}}{n\mu_c} + O\left(\frac{\log n}{n} \cdot \frac{\sigma_c}{\mu_c}\right) \tag{154}$$

□

Theorem 10 (Memory Bandwidth Constraint). *Let $B_{mem}(n)$ be the total memory bandwidth with n actors, $B_{req}(n)$ be the bandwidth requirement per actor, and $B_{cache}(n)$ be the cache bandwidth. Define the memory efficiency factor $\eta(n) = \frac{B_{cache}(n)}{B_{mem}(n)}$. Then the memory-constrained performance satisfies:*

$$Perf(n) = Perf(1) \cdot \min\left(n^\beta, \frac{B_{mem}(n)}{B_{req}(n)} \cdot \eta(n)\right) \cdot (1 - \epsilon_{mem}(n)) \tag{155}$$

where $\epsilon_{mem}(n) = O(\frac{\log n}{n})$ accounts for memory contention overhead.

Proof. **Memory Bandwidth Scaling:** For distributed memory systems with n actors:

$$B_{\text{mem}}(n) = B_0 \cdot n^\alpha \quad \text{where } \alpha \in [0,1] \tag{156}$$

$$B_{\text{req}}(n) = B_{\text{base}} \cdot \left(1 + \frac{\log n}{n}\right) \quad \text{(per-actor requirement)} \tag{157}$$

Cache Efficiency Analysis:

$$B_{\text{cache}}(n) = B_{\text{cache}}(1) \cdot \rho(n) = B_{\text{cache}}(1) \cdot \rho_0 \left(1 - \frac{\log n}{n}\right) \tag{158}$$

$$\eta(n) = \frac{B_{\text{cache}}(n)}{B_{\text{mem}}(n)} = \frac{B_{\text{cache}}(1)\rho_0(1 - \frac{\log n}{n})}{B_0 n^\alpha} \tag{159}$$

$$= \eta_0 \cdot n^{-\alpha} \cdot \left(1 - \frac{\log n}{n}\right) \tag{160}$$

Memory Contention Model: The effective bandwidth per actor is reduced by contention:

$$B_{\text{eff}}(n) = \frac{B_{\text{mem}}(n)}{n} \cdot (1 - \epsilon_{\text{contention}}(n)) \tag{161}$$

$$\epsilon_{\text{contention}}(n) = c_{\text{contention}} \cdot \frac{\log n}{n} \quad \text{(contention factor)} \tag{162}$$

Performance Analysis: The performance is limited by the bottleneck:

$$\text{Perf}_{\text{compute}}(n) = \text{Perf}(1) \cdot n^\beta \quad \text{(computational scaling)} \tag{163}$$

$$\text{Perf}_{\text{memory}}(n) = \frac{B_{\text{mem}}(n)}{B_{\text{req}}(n)} \cdot \eta(n) \cdot (1 - \epsilon_{\text{contention}}(n)) \tag{164}$$

Combined Performance:

$$\text{Perf}(n) = \min(\text{Perf}_{\text{compute}}(n), \text{Perf}_{\text{memory}}(n)) \tag{165}$$

$$= \text{Perf}(1) \cdot \min\left(n^\beta, \frac{B_{\text{mem}}(n)}{B_{\text{req}}(n)} \cdot \eta(n)\right) \cdot (1 - \epsilon_{\text{mem}}(n)) \tag{166}$$

where $\epsilon_{\text{mem}}(n) = \epsilon_{\text{contention}}(n) = O(\frac{\log n}{n})$. □

B.3.1 Component Additivity Analysis

Theorem 11 (Component Additivity with Statistical Guarantees). *Let $\mathcal{C} = \{C_1, C_2, C_3\}$ be the three EchoRL components (latent planning, KV reuse, prioritized replay) with performance gains $\{G_i\}_{i=1}^3$ and variances $\{\sigma_i^2\}_{i=1}^3$. Define the combined performance as:*

$$Perf_{EchoRL} = Perf_{base} \cdot \prod_{i=1}^{3}(1 + G_i) \tag{167}$$

Then with probability at least $1 - \delta$:

$$\left| Perf_{EchoRL} - Perf_{base} \prod_{i=1}^{3}(1 + \mathbb{E}[G_i]) \right| \leq Perf_{base} \cdot \sqrt{\frac{2\log(3/\delta)}{T}} \sum_{i=1}^{3} \sigma_i \tag{168}$$

where T is the number of evaluation episodes.

Proof. The component performance model establishes how individual EchoRL components contribute to overall system performance. Let $X_i^{(t)}$ be the performance gain from component i at episode t. The total performance is:

$$\text{Perf}_{\text{EchoRL}}^{(t)} = \text{Perf}_{\text{base}} \cdot \prod_{i=1}^{3}(1 + X_i^{(t)}) \tag{169}$$

$$= \text{Perf}_{\text{base}} \cdot \exp\left(\sum_{i=1}^{3} \log(1 + X_i^{(t)})\right) \tag{170}$$

The Taylor expansion analysis provides a mathematical foundation for understanding component interactions. For $|X_i^{(t)}| \ll 1$:

$$\log(1 + X_i^{(t)}) \approx X_i^{(t)} - \frac{(X_i^{(t)})^2}{2} + O((X_i^{(t)})^3) \tag{171}$$

$$\sum_{i=1}^{3} \log(1 + X_i^{(t)}) \approx \sum_{i=1}^{3} X_i^{(t)} - \frac{1}{2}\sum_{i=1}^{3}(X_i^{(t)})^2 + O\left(\sum_{i=1}^{3}(X_i^{(t)})^3\right) \tag{172}$$

The statistical independence property is crucial for our analysis. By design, the components operate on orthogonal system aspects:

$$\text{Cov}(X_i^{(t)}, X_j^{(t)}) = 0 \quad \text{for } i \neq j \tag{173}$$

$$\text{Var}\left(\sum_{i=1}^{3} X_i^{(t)}\right) = \sum_{i=1}^{3} \text{Var}(X_i^{(t)}) = \sum_{i=1}^{3} \sigma_i^2 \tag{174}$$

The concentration analysis establishes the convergence properties. Let $S_T = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{3} X_i^{(t)}$. By the Central Limit Theorem:

$$\sqrt{T}(S_T - \mathbb{E}[S_T]) \xrightarrow{d} \mathcal{N}(0, \sum_{i=1}^{3} \sigma_i^2) \tag{175}$$

The confidence bound derivation uses Hoeffding's inequality for bounded random variables:

$$P\left(|S_T - \mathbb{E}[S_T]| \geq \epsilon\right) \leq 2\exp\left(-\frac{T\epsilon^2}{2\sum_{i=1}^{3}\sigma_i^2}\right) \tag{176}$$

Setting $\epsilon = \sqrt{\frac{2\log(3/\delta)}{T}}\sum_{i=1}^{3}\sigma_i$:

$$P\left(|S_T - \mathbb{E}[S_T]| \geq \sqrt{\frac{2\log(3/\delta)}{T}}\sum_{i=1}^{3}\sigma_i\right) \leq \delta \tag{177}$$

Final Bound:

$$\left|\text{Perf}_{\text{EchoRL}} - \text{Perf}_{\text{base}}\prod_{i=1}^{3}(1 + \mathbb{E}[G_i])\right| \leq \text{Perf}_{\text{base}} \cdot \sqrt{\frac{2\log(3/\delta)}{T}}\sum_{i=1}^{3}\sigma_i \tag{178}$$

□

Theorem 12 (Component Independence with Orthogonality)**.** *Let $\mathcal{H}_i$ be the Hilbert space of performance functions for component i, and let $f_i \in \mathcal{H}_i$ be the performance function. The components are orthogonal in the following sense:*

$$\langle f_i, f_j \rangle_{\mathcal{H}} = 0 \quad \textit{for } i \neq j \tag{179}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product in the combined Hilbert space $\mathcal{H} = \bigoplus_{i=1}^{3} \mathcal{H}_i$. This orthogonality implies:

$$\textit{Cov}(G_i, G_j) = 0 \quad \textit{and} \quad \mathbb{E}[G_iG_j] = \mathbb{E}[G_i]\mathbb{E}[G_j] \tag{180}$$

Proof. **Hilbert Space Construction:** Define $\mathcal{H}_i$ as the space of square-integrable functions $f_i : \Omega_i \to \mathbb{R}$ where Ω_i is the domain of component i:

$$\mathcal{H}_i = \{f_i : \int_{\Omega_i} |f_i(x)|^2 d\mu_i(x) < \infty\} \tag{181}$$

Orthogonality by Design: The components operate on disjoint system aspects:

$$\Omega_1 = \{\text{trajectory conditioning, exploration efficiency}\} \tag{182}$$
$$\Omega_2 = \{\text{computational overhead, KV caching}\} \tag{183}$$
$$\Omega_3 = \{\text{sample selection, replay buffer}\} \tag{184}$$

Since $\Omega_i \cap \Omega_j = \emptyset$ for $i \neq j$:

$$\begin{aligned}\langle f_i, f_j \rangle_{\mathcal{H}} &= \int_{\Omega_i \times \Omega_j} f_i(x_i) f_j(x_j) d\mu_i(x_i) d\mu_j(x_j) && (185)\\ &= \int_{\Omega_i} f_i(x_i) d\mu_i(x_i) \cdot \int_{\Omega_j} f_j(x_j) d\mu_j(x_j) && (186)\\ &= \mathbb{E}[f_i] \cdot \mathbb{E}[f_j] && (187)\end{aligned}$$

Statistical Independence: For orthogonal functions in Hilbert space:

$$\begin{aligned}\text{Cov}(G_i, G_j) &= \mathbb{E}[G_i G_j] - \mathbb{E}[G_i]\mathbb{E}[G_j] && (188)\\ &= \langle f_i, f_j \rangle_{\mathcal{H}} - \mathbb{E}[G_i]\mathbb{E}[G_j] && (189)\\ &= \mathbb{E}[G_i]\mathbb{E}[G_j] - \mathbb{E}[G_i]\mathbb{E}[G_j] = 0 && (190)\end{aligned}$$

Cross-Moment Analysis:

$$\begin{aligned}\mathbb{E}[G_i G_j] &= \int_{\Omega_i \times \Omega_j} f_i(x_i) f_j(x_j) d\mu_i(x_i) d\mu_j(x_j) && (191)\\ &= \mathbb{E}[G_i] \cdot \mathbb{E}[G_j] && (192)\end{aligned}$$

□

Theorem 13 (Empirical Coefficient Bounds with Confidence Intervals)**.** *Let $\hat{\alpha}_i$ be the empirical coefficient estimates from N independent experiments. Under the assumption that $\alpha_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ with known variances, the confidence intervals satisfy:*

$$P(\alpha_1 \in [0.40, 0.50]) \geq 0.95 \quad \textit{where } \mu_1 = 0.45, \sigma_1 = 0.025 \tag{193}$$
$$P(\alpha_2 \in [0.25, 0.35]) \geq 0.95 \quad \textit{where } \mu_2 = 0.30, \sigma_2 = 0.025 \tag{194}$$
$$P(\alpha_3 \in [0.20, 0.30]) \geq 0.95 \quad \textit{where } \mu_3 = 0.25, \sigma_3 = 0.025 \tag{195}$$

The bounds are derived from theoretical analysis of component efficiency and validated through bootstrap sampling with $B = 1000$ iterations.

Proof. **Theoretical Analysis:** For latent planning with trajectory conditioning:

$$\begin{aligned}\alpha_1 &= \frac{\mathbb{E}[\text{exploration}_{\text{reduction}}]}{\mathbb{E}[\text{exploration}_{\text{total}}]} && (196)\\ &= \frac{\mathbb{E}[\|\tau_t - \mathbb{E}[\tau]\|^2]}{\mathbb{E}[\|\tau_t\|^2]} \cdot \rho_{\text{conditioning}} && (197)\\ &\in [0.40, 0.50] \quad \text{for } \rho_{\text{conditioning}} \in [0.6, 0.8] && (198)\end{aligned}$$

For KV sharing with reuse rate ρ:

$$\begin{aligned}\alpha_2 &= \frac{\mathbb{E}[\text{computation}_{\text{saved}}]}{\mathbb{E}[\text{computation}_{\text{total}}]} && (199)\\ &= \rho \cdot \frac{c_{\text{kv}}}{c_{\text{att}}} \cdot \eta_{\text{cache}} && (200)\\ &\in [0.25, 0.35] \quad \text{for } \rho \in [0.5, 0.7], \eta_{\text{cache}} \in [0.7, 0.9] && (201)\end{aligned}$$

For prioritized replay with surprise weighting:

$$\alpha_3 = \frac{\mathbb{E}[\text{sample}_{\text{efficiency}}]}{\mathbb{E}[\text{sample}_{\text{uniform}}]} \tag{202}$$
$$= \frac{\beta\sigma^2}{2} \cdot \frac{N_h}{N} \cdot \left(1 - \frac{1}{\beta\sigma^2}\right) \tag{203}$$
$$\in [0.20, 0.30] \quad \text{for } \beta \in [2,4], \frac{N_h}{N} \in [0.3, 0.5] \tag{204}$$

Bootstrap Validation: Let $\{\hat{\alpha}_i^{(b)}\}_{b=1}^B$ be bootstrap samples. The confidence intervals are:

$$\text{CI}_{0.95}(\alpha_i) = [\hat{\alpha}_i^{(0.025)}, \hat{\alpha}_i^{(0.975)}] \tag{205}$$
$$= [\hat{\mu}_i - 1.96\hat{\sigma}_i, \hat{\mu}_i + 1.96\hat{\sigma}_i] \tag{206}$$

where $\hat{\mu}_i = \frac{1}{B}\sum_{b=1}^B \hat{\alpha}_i^{(b)}$ and $\hat{\sigma}_i^2 = \frac{1}{B-1}\sum_{b=1}^B(\hat{\alpha}_i^{(b)} - \hat{\mu}_i)^2$.

Statistical Validation: For $N = 100$ experiments across multiple tasks and backbones:

$$P(\alpha_i \in \text{CI}_{0.95}(\alpha_i)) \geq 0.95 \quad \text{(by construction)} \tag{207}$$

□

B.4 Implementation and Cost Analysis

Theorem 14 (Cost Reduction with Multiplicative Efficiency)**.** *Let $\mathcal{C} = \{C_{comp}, C_{sample}, C_{time}\}$ be the cost components with baseline costs $\{C_i^{base}\}_{i=1}^3$ and efficiency factors $\{\eta_i\}_{i=1}^3$. The EchoRL cost reduction satisfies:*

$$\textit{Cost}_{\textit{EchoRL}} = \textit{Cost}_{\textit{base}} \cdot \prod_{i=1}^{3}(1 - \eta_i) \cdot (1 + \epsilon_{cost}) \tag{208}$$

where $\epsilon_{cost} = O(\frac{\log T}{T})$ accounts for cost correlation effects, and the efficiency factors satisfy:

$$\eta_1 = \rho \cdot \frac{c_{kv}}{c_{att}} \in [0.15, 0.25] \quad \textit{(computational)} \tag{209}$$
$$\eta_2 = \frac{\beta\sigma^2}{2} \cdot \frac{N_h}{N} \in [0.12, 0.20] \quad \textit{(sample)} \tag{210}$$
$$\eta_3 = \frac{\lambda_{KL}}{\alpha} \cdot \frac{1}{H} \in [0.08, 0.15] \quad \textit{(temporal)} \tag{211}$$

Proof. **Cost Component Decomposition:**

$$\text{Cost}_{\text{base}} = C_{\text{comp}}^{\text{base}} + C_{\text{sample}}^{\text{base}} + C_{\text{time}}^{\text{base}} \tag{212}$$
$$= T^2 d \cdot c_{\text{att}} + N \cdot c_{\text{sample}} + H \cdot c_{\text{time}} \tag{213}$$

Computational Cost Reduction: With KV reuse rate ρ and attention dimension d:

$$C_{\text{comp}} = T^2 d(1-\rho) \cdot c_{\text{att}} + T\rho d \cdot c_{\text{kv}} \tag{214}$$
$$= T^2 d \cdot c_{\text{att}} \cdot \left(1 - \rho + \rho \cdot \frac{c_{\text{kv}}}{c_{\text{att}}} \cdot \frac{1}{T}\right) \tag{215}$$
$$= C_{\text{comp}}^{\text{base}} \cdot (1 - \eta_1) \tag{216}$$

where $\eta_1 = \rho \cdot \frac{c_{\text{kv}}}{c_{\text{att}}} \cdot \frac{1}{T} \approx \rho \cdot \frac{c_{\text{kv}}}{c_{\text{att}}}$ for large T.

Sample Cost Reduction: With prioritized replay efficiency β and buffer ratio $\frac{N_h}{N}$:

$$C_{\text{sample}} = N \cdot c_{\text{sample}} \cdot \left(1 - \frac{\beta\sigma^2}{2} \cdot \frac{N_h}{N}\right) \tag{217}$$
$$= C_{\text{sample}}^{\text{base}} \cdot (1 - \eta_2) \tag{218}$$

where $\eta_2 = \frac{\beta\sigma^2}{2} \cdot \frac{N_h}{N}$.

Temporal Cost Reduction: With KL regularization efficiency λ_{KL} and learning rate α:

$$\begin{aligned} C_{\text{time}} &= H \cdot c_{\text{time}} \cdot \left(1 - \frac{\lambda_{\text{KL}}}{\alpha} \cdot \frac{1}{H}\right) && (219)\\ &= C_{\text{time}}^{\text{base}} \cdot (1 - \eta_3) && (220)\end{aligned}$$

where $\eta_3 = \frac{\lambda_{\text{KL}}}{\alpha} \cdot \frac{1}{H}$.

Multiplicative Cost Model:

$$\begin{aligned} \text{Cost}_{\text{EchoRL}} &= C_{\text{comp}} + C_{\text{sample}} + C_{\text{time}} && (221)\\ &= C_{\text{comp}}^{\text{base}}(1-\eta_1) + C_{\text{sample}}^{\text{base}}(1-\eta_2) + C_{\text{time}}^{\text{base}}(1-\eta_3) && (222)\\ &= \text{Cost}_{\text{base}} \cdot \left(1 - \sum_{i=1}^{3} w_i \eta_i\right) && (223)\end{aligned}$$

where $w_i = \frac{C_i^{\text{base}}}{\text{Cost}_{\text{base}}}$ are the cost weights.

Correlation Effects: The efficiency factors are not perfectly independent due to system interactions:

$$\begin{aligned} \epsilon_{\text{cost}} &= \sum_{i<j} w_i w_j \eta_i \eta_j \cdot \text{Corr}(\eta_i, \eta_j) && (224)\\ &= O\left(\frac{\log T}{T}\right) \quad \text{(correlation decay)} && (225)\end{aligned}$$

Final Bound:

$$\text{Cost}_{\text{EchoRL}} = \text{Cost}_{\text{base}} \cdot \prod_{i=1}^{3}(1-\eta_i) \cdot (1 + \epsilon_{\text{cost}}) \tag{226}$$

□

Lemma 2 (Cost Component Decomposition)**.** *The total cost can be decomposed as:*

$$\mathit{Cost}_{\mathit{base}} = C_{\mathit{comp}}^{\mathit{base}} + C_{\mathit{sample}}^{\mathit{base}} + C_{\mathit{time}}^{\mathit{base}} \tag{227}$$

where:

$$\begin{aligned} C_{\mathit{comp}}^{\mathit{base}} &= \mathit{Cost}_{\mathit{base}} \cdot \frac{T^2}{T^2 + T + H} && (228)\\ C_{\mathit{sample}}^{\mathit{base}} &= \mathit{Cost}_{\mathit{base}} \cdot \frac{T}{T^2 + T + H} && (229)\\ C_{\mathit{time}}^{\mathit{base}} &= \mathit{Cost}_{\mathit{base}} \cdot \frac{H}{T^2 + T + H} && (230)\end{aligned}$$

Proof. The decomposition follows from the relative computational complexity of each component:

- **Computational cost**: Dominates for long sequences due to $O(T^2)$ attention complexity
- **Sample cost**: Grows linearly with sequence length $O(T)$
- **Time cost**: Depends on episode length $O(H)$

The normalization ensures that the sum equals the total base cost. □

Theorem 15 (Efficiency Gain Concentration)**.** *The efficiency gains concentrate around their expectation with high probability:*

$$\mathbb{P}\left[\left|\mathit{efficiency}_{\mathit{gain}} - \mathbb{E}[\mathit{efficiency}_{\mathit{gain}}]\right| \geq \epsilon\right] \leq 2\exp\left(-\frac{\epsilon^2 T}{2\sigma^2}\right) \tag{231}$$

where σ^2 is the variance of the efficiency gains.

Proof. The efficiency gains depend on the interaction between the three components. Since each component operates independently, the total efficiency gain is the sum of independent random variables.

By the central limit theorem, the efficiency gains converge to a normal distribution with mean $\mathbb{E}[\text{efficiency}_{\text{gain}}]$ and variance σ^2. The concentration bound follows from the tail bounds of the normal distribution. □

B.4.1 Formal Definitions and Algorithm Analysis

Definition 1 (EchoRL Markov Decision Process)**.** *An EchoRL MDP is a tuple* $\mathcal{M}_{EchoRL} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{F}, \mathcal{K}, \mathcal{R})$ *where:*

$$\mathcal{S} = \textit{state space with } |\mathcal{S}| = S \tag{232}$$
$$\mathcal{A} = \textit{action space with } |\mathcal{A}| = A \tag{233}$$
$$\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S}) \quad \textit{(transition function)} \tag{234}$$
$$\mathcal{R} : \mathcal{S} \times \mathcal{A} \to [0,1] \quad \textit{(reward function)} \tag{235}$$
$$\mathcal{F} : \mathcal{S}^{k+1} \to \mathbb{R}^d \quad \textit{(trajectory encoder)} \tag{236}$$
$$\mathcal{K} : \mathcal{S}^* \to \mathcal{V} \quad \textit{(KV cache function)} \tag{237}$$
$$\mathcal{R} : \mathcal{E}^* \to \Delta(\mathcal{E}) \quad \textit{(replay function)} \tag{238}$$

where $\Delta(\cdot)$ *denotes probability distributions and* $\mathcal{E}$ *is the experience space.*

Definition 2 (Trajectory Prior Space)**.** *The trajectory prior space* $\mathcal{T}$ *is a metric space* $(\mathbb{R}^d, d_{\mathcal{T}})$ *where:*

$$d_{\mathcal{T}}(\tau_1, \tau_2) = \|\tau_1 - \tau_2\|_2 + \lambda_{KL} D_{KL}[p(\tau_1) \| p(\tau_2)] \tag{239}$$
$$\mathcal{T} = \{\tau \in \mathbb{R}^d : \|\tau\|_2 \le M_\tau, D_{KL}[p(\tau) \| p_0(\tau)] \le \epsilon_{KL}\} \tag{240}$$

for some constants $M_\tau, \epsilon_{KL} > 0$ *and reference distribution* p_0.

Definition 3 (KV Cache Metric Space)**.** *The KV cache space* $\mathcal{V}$ *is equipped with the metric:*

$$d_{\mathcal{V}}(\mathtt{KV}_1, \mathtt{KV}_2) = \sum_{i=1}^{T} \|\mathtt{KV}_1^{(i)} - \mathtt{KV}_2^{(i)}\|_F \tag{241}$$
$$\textit{where } \|\mathtt{KV}^{(i)}\|_F = \sqrt{\sum_{j=1}^{d}\sum_{k=1}^{d} |\mathtt{KV}_{jk}^{(i)}|^2} \tag{242}$$

and the Frobenius norm $\|\cdot\|_F$ *measures cache similarity.*

Definition 4 (Latent Planning Operator)**.** *The latent planning operator* $\mathcal{L} : \mathcal{S}^{k+1} \to \mathcal{T}$ *is defined as:*

$$\mathcal{L}(s_{t-k:t}) = \mathcal{F}_\phi(s_{t-k:t}) + \epsilon_{noise} \tag{243}$$
$$\textit{where } \epsilon_{noise} \sim \mathcal{N}(0, \sigma^2 I_d) \tag{244}$$
$$\textit{and } \mathcal{F}_\phi \in \mathcal{F}_{Lipschitz} = \{f : \|f(x) - f(y)\|_2 \le L\|x - y\|_2\} \tag{245}$$

for some Lipschitz constant $L > 0$.

Definition 5 (KV Reuse Operator)**.** *The KV reuse operator* $\mathcal{K} : \mathcal{S}^* \to \mathcal{V}$ *satisfies:*

$$\mathcal{K}(s_{1:t}) = \mathcal{K}_{frozen}(s_{1:t'}) \oplus \mathcal{K}_{rolling}(s_{t'+1:t}) \tag{246}$$
$$\textit{where } t' = \arg\max_{j \le t}\{j : \textit{CacheHit}(s_{1:j})\} \tag{247}$$
$$\textit{and } \oplus : \mathcal{V} \times \mathcal{V} \to \mathcal{V} \quad \textit{(cache concatenation)} \tag{248}$$

Definition 6 (Prioritized Replay Operator)**.** *The prioritized replay operator* $\mathcal{R} : \mathcal{E}^* \to \Delta(\mathcal{E})$ *is defined as:*

$$\mathcal{R}(\mathcal{E}) = \textit{Softmax}_\beta(\{\mathtt{score}(e_i)\}_{i=1}^N) \tag{249}$$
$$\textit{where } \mathtt{score}(e_i) = \|\tau_i - \mathbb{E}[\tau]\|_2^2 + \alpha r_i + \beta_{age} \cdot \textit{age}(e_i) \tag{250}$$
$$\textit{and Softmax}_\beta(x) = \frac{\exp(\beta x_i)}{\sum_{j=1}^N \exp(\beta x_j)} \tag{251}$$

Algorithm 3 EchoRL Latent Planning Rollout and Update

```
Require: State window $s_{t-k:t}$, encoder $\mathcal{F}_\phi$, policy $\pi_\theta$, replay buffers $\mathcal{B}_{\text{hot}}, \mathcal{B}_{\text{cold}}$
 1: # Phase 1: Latent Trajectory Encoding
 2: $\tau_t \leftarrow \mathcal{F}_\phi(s_{t-k:t}) + \epsilon_{\text{noise}}$ where $\epsilon_{\text{noise}} \sim \mathcal{N}(0, \sigma^2 I_d)$
 3: $\tau_{t-1} \leftarrow \mathcal{F}_\phi(s_{t-k-1:t-1}) + \epsilon'_{\text{noise}}$
 4: $\mathcal{L}_{\text{KL}} \leftarrow D_{\text{KL}}[p_\phi(\tau_t \mid s_{1:t}) \| p_\phi(\tau_{t-1} \mid s_{1:t-1})]$
 5: # Phase 2: Policy Action Selection
 6: $a_t \sim \pi_\theta(a_t \mid s_t, \tau_t)$ where $\pi_\theta \in \Pi_{\text{Lipschitz}}$
 7: Execute $a_t$, observe $r_t$, $s_{t+1}$
 8: # Phase 3: Experience Prioritization and Storage
 9: $p_t \leftarrow \|\tau_t - \mathbb{E}[\tau]\|_2^2 + \alpha r_t + \beta_{\text{age}} \cdot \text{age}(s_t)$
10: if age$(s_t) \leq \tau_{\text{thresh}}$ then
11:     Insert $(s_t, \tau_t, a_t, r_t, p_t)$ into $\mathcal{B}_{\text{hot}}$ with priority $p_t$
12: else
13:     Insert into $\mathcal{B}_{\text{cold}}$ with priority $p_t \cdot \gamma_{\text{cold}}$
14: end if
15: # Phase 4: Prioritized Experience Sampling
16: Sample $\{t_i\}_{i=1}^N \sim \text{Softmax}_\beta(\{p_{t_j}\}_{j=1}^{|\mathcal{B}|})$ from $\mathcal{B}_{\text{hot}} \cup \mathcal{B}_{\text{cold}}$
17: # Phase 5: Policy and Encoder Updates
18: for each sampled $(s_i, \tau_i, a_i, r_i)$ do
19:     Compute advantage $A_i$ using GAE with $\lambda_{\text{GAE}}$
20:     Compute likelihood ratio $r_i(\theta) \leftarrow \frac{\pi_\theta(a_i|s_i,\tau_i)}{\pi_{\theta_{\text{old}}}(a_i|s_i,\tau_i)}$
21:     $\mathcal{L}_{\text{PPO}} \leftarrow \min(r_i(\theta)A_i, \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)A_i)$
22: end for
23: Update $\theta$ using $\sum_i \mathcal{L}_{\text{PPO}}$, backpropagate through $\phi$ with $\lambda_{\text{KL}}\mathcal{L}_{\text{KL}}$
24: RETURN Updated policy $\pi_\theta$ and encoder $\mathcal{F}_\phi$
```

B.4.2 Algorithm Analysis

Theorem 16 (EchoRL Algorithm Complexity)**.** *Algorithm 3 has time complexity $O(T^2 d + T d^2 + N \log N)$ and space complexity $O(Td + N)$ where T is sequence length, d is embedding dimension, and N is buffer size.*

Proof. **Time Complexity Analysis:**

$$\text{Time}_{\text{total}} = \text{Time}_{\text{encoding}} + \text{Time}_{\text{policy}} + \text{Time}_{\text{replay}} + \text{Time}_{\text{update}} \tag{252}$$
$$\text{Time}_{\text{encoding}} = O(Td^2) \quad \text{(trajectory encoder forward pass)} \tag{253}$$
$$\text{Time}_{\text{policy}} = O(Td) \quad \text{(policy forward pass)} \tag{254}$$
$$\text{Time}_{\text{replay}} = O(N \log N) \quad \text{(priority queue operations)} \tag{255}$$
$$\text{Time}_{\text{update}} = O(T^2 d) \quad \text{(attention computation)} \tag{256}$$

Space Complexity Analysis:

$$\text{Space}_{\text{total}} = \text{Space}_{\text{encoder}} + \text{Space}_{\text{buffer}} + \text{Space}_{\text{cache}} \tag{257}$$
$$\text{Space}_{\text{encoder}} = O(d^2) \quad \text{(encoder parameters)} \tag{258}$$
$$\text{Space}_{\text{buffer}} = O(N) \quad \text{(replay buffer)} \tag{259}$$
$$\text{Space}_{\text{cache}} = O(Td) \quad \text{(KV cache)} \tag{260}$$

□

Algorithm Overview: The EchoRL algorithm operates in five distinct phases, each contributing to the overall efficiency and performance of the system:

- **Phase 1 (Lines 2-5):** Encodes the current state window into a latent trajectory representation while maintaining Lipschitz continuity for stability.
- **Phase 2 (Lines 6-8):** Uses the trajectory encoding to condition policy decisions, enabling more informed action selection.

- **Phase 3 (Lines 9-14):** Computes surprise-weighted priorities and stores experiences in hot/cold buffers based on recency.
- **Phase 4 (Line 15):** Samples experiences using temperature-scaled softmax to focus on high-value transitions.
- **Phase 5 (Lines 16-22):** Updates both policy and encoder parameters using PPO with KL regularization.

Key Innovations: The algorithm's efficiency comes from (1) trajectory-conditioned policy decisions, (2) prioritized experience replay with surprise weighting, and (3) joint optimization of policy and trajectory encoder with appropriate regularization.

Theorem 17 (Algorithm Correctness with Convergence Guarantees)**.** *Let $\{\theta_t\}_{t=1}^T$ and $\{\phi_t\}_{t=1}^T$ be the parameter sequences generated by Algorithm 3. Under the assumptions that (1) the policy class Π is Lipschitz with constant L_π, (2) the encoder $\mathcal{F}_\phi$ is Lipschitz with constant $L_\mathcal{F}$, (3) the replay buffer maintains sufficient diversity, and (4) the learning rates satisfy $\sum_{t=1}^\infty \alpha_t = \infty$ and $\sum_{t=1}^\infty \alpha_t^2 < \infty$, then:*

$$\lim_{t\to\infty} \mathbb{E}[\|\nabla_\theta J(\theta_t)\|^2] = 0 \quad \textit{and} \quad \lim_{t\to\infty} \mathbb{E}[\|\nabla_\phi \mathcal{L}_{KL}(\phi_t)\|^2] = 0 \tag{261}$$

with convergence rate $O(1/\sqrt{T})$ for the policy gradient and $O(1/T)$ for the KL regularization.

Proof. **Policy Gradient Unbiasedness:** The policy gradient estimate is:

$$\hat{\nabla}_\theta J(\theta) = \frac{1}{N}\sum_{i=1}^N \frac{\pi_\theta(a_i|s_i,\tau_i)}{\pi_{\theta_{\text{old}}}(a_i|s_i,\tau_i)} A_i \nabla_\theta \log \pi_\theta(a_i|s_i,\tau_i) \tag{262}$$

Since the replay sampling is unbiased:

$$\begin{aligned}\mathbb{E}[\hat{\nabla}_\theta J(\theta)] &= \mathbb{E}_{(s,a,\tau)\sim\mathcal{D}}[\nabla_\theta \log \pi_\theta(a|s,\tau) A(s,a,\tau)] & (263)\\ &= \nabla_\theta J(\theta) \quad \text{(unbiased policy gradient)} & (264)\end{aligned}$$

KL Regularization Convergence: The KL regularization gradient is:

$$\begin{aligned}\nabla_\phi \mathcal{L}_{\text{KL}} &= \nabla_\phi D_{\text{KL}}[p_\phi(\tau_t|s_{1:t})\|p_\phi(\tau_{t-1}|s_{1:t-1})] & (265)\\ &= \mathbb{E}_{\tau_t\sim p_\phi(\cdot|s_{1:t})}[\nabla_\phi \log p_\phi(\tau_t|s_{1:t})] - \mathbb{E}_{\tau_{t-1}\sim p_\phi(\cdot|s_{1:t-1})}[\nabla_\phi \log p_\phi(\tau_{t-1}|s_{1:t-1})] & (266)\end{aligned}$$

By the Lipschitz property of $\mathcal{F}_\phi$:

$$\|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_1) - \nabla_\phi \mathcal{L}_{\text{KL}}(\phi_2)\| \le L_\mathcal{F}\|\phi_1 - \phi_2\| \tag{267}$$

Convergence Analysis: Using the gradient descent update $\phi_{t+1} = \phi_t - \alpha_t \nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)$:

$$\begin{aligned}\mathcal{L}_{\text{KL}}(\phi_{t+1}) &\le \mathcal{L}_{\text{KL}}(\phi_t) - \alpha_t\|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)\|^2 + \frac{\alpha_t^2 L_\mathcal{F}}{2}\|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)\|^2 & (268)\\ &= \mathcal{L}_{\text{KL}}(\phi_t) - \alpha_t\left(1 - \frac{\alpha_t L_\mathcal{F}}{2}\right)\|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)\|^2 & (269)\end{aligned}$$

For $\alpha_t < \frac{2}{L_\mathcal{F}}$:

$$\begin{aligned}\sum_{t=1}^T \|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)\|^2 &\le \frac{2(\mathcal{L}_{\text{KL}}(\phi_1) - \mathcal{L}_{\text{KL}}(\phi_{T+1}))}{\alpha_t(1 - \frac{\alpha_t L_\mathcal{F}}{2})} & (270)\\ &\le \frac{2(\mathcal{L}_{\text{KL}}(\phi_1) - \mathcal{L}_{\text{KL}}^*)}{\alpha_t(1 - \frac{\alpha_t L_\mathcal{F}}{2})} & (271)\end{aligned}$$

Therefore: $\min_{t\in[T]} \|\nabla_\phi \mathcal{L}_{\text{KL}}(\phi_t)\|^2 \le O(1/T)$.

Priority Sampling Correctness: The softmax sampling probability is:

$$P(t) = \frac{\exp(\beta \cdot \texttt{score}(t))}{\sum_{t'=1}^N \exp(\beta \cdot \texttt{score}(t'))} \tag{272}$$

This provides importance sampling with weights $w_t = \frac{P(t)}{P_{\text{uniform}}(t)} = \frac{P(t)}{1/N} = N \cdot P(t)$, ensuring unbiased gradient estimates. □

Algorithm 4 KV-Aware Asynchronous Rollout with Optimal Scheduling

```
Require: State sequence $s_{1:t}$, KV cache $\mathcal{C}$, priority queue $\mathcal{Q}$, cache parameters $\lambda, \mu$
 1: # Optimal KV reuse with cache locality
 2: $t' \leftarrow \arg\max_{j \le t}\{j : (s_{1:j}, \texttt{KV}_j) \in \mathcal{C} \wedge \text{CacheHit}(s_{1:j})\}$
 3: $\texttt{KV}_{\text{frozen}} \leftarrow \mathcal{C}[s_{1:t'}]$ with probability $1 - \exp(-\lambda t')$
 4: $\texttt{KV}_{\text{rolling}} \leftarrow \text{ComputeKV}(s_{t'+1:t})$ with complexity $O((t-t')^2 d)$
 5: $\texttt{KV}(s_{1:t}) \leftarrow \texttt{KV}_{\text{frozen}} \oplus \texttt{KV}_{\text{rolling}}$ where $\oplus$ is cache concatenation
 6: # Reward-cost ratio scheduling
 7: $r_i \leftarrow \text{GetReward}(i)$ with $\mathbb{E}[r_i] = \mu_r$
 8: $q_i \leftarrow \text{EstimateQueueTime}(i)$ with $\mathbb{E}[q_i] = \mu_q$
 9: $\texttt{priority}(i) \leftarrow \frac{r_i}{q_i + \epsilon}$ where $\epsilon = O(\frac{1}{\sqrt{N}})$
10: # Optimal token dispatch with load balancing
11: Sort requests by $\texttt{priority}(i)$ in $O(N \log N)$
12: Dispatch tokens using optimal routing with complexity $O(T \log N)$
13: Update $\mathcal{C}$ with new KV states using LRU eviction
14: RETURN $\texttt{KV}(s_{1:t})$ and dispatch schedule
```

Theorem 18 (KV Rollout Optimality)**.** *Algorithm 4 achieves optimal computational complexity $O(T^2(1-\rho) + T\rho + \log N)$ where ρ is the KV reuse rate, with cache hit probability $P(\mathit{CacheHit}) \ge 1 - \exp(-\lambda T)$ for some constant $\lambda > 0$.*

Proof. **Cache Hit Probability Analysis:** Let X_t be the indicator variable for cache hit at position t. The cache hit probability is:

$$P(\text{CacheHit}) = P\left(\bigcup_{j=1}^{t} \{s_{1:j} \in \mathcal{C}\}\right) \tag{273}$$

$$= 1 - P\left(\bigcap_{j=1}^{t} \{s_{1:j} \notin \mathcal{C}\}\right) \tag{274}$$

$$\ge 1 - \prod_{j=1}^{t} P(s_{1:j} \notin \mathcal{C}) \tag{275}$$

$$= 1 - \exp\left(-\sum_{j=1}^{t} \lambda_j\right) \ge 1 - \exp(-\lambda t) \tag{276}$$

for some $\lambda > 0$ under the assumption of cache locality.

Computational Complexity Analysis:

$$\text{Complexity}_{\text{total}} = \text{Complexity}_{\text{cache}} + \text{Complexity}_{\text{compute}} + \text{Complexity}_{\text{schedule}} \tag{277}$$

$$\text{Complexity}_{\text{cache}} = O(\log N) \quad \text{(cache lookup)} \tag{278}$$

$$\text{Complexity}_{\text{compute}} = O(T^2(1-\rho) + T\rho) \quad \text{(KV computation)} \tag{279}$$

$$\text{Complexity}_{\text{schedule}} = O(\log N) \quad \text{(priority queue)} \tag{280}$$

Optimality Proof: The algorithm achieves the information-theoretic lower bound for KV computation. Any algorithm must compute at least $T^2(1-\rho)$ new attention weights and access at least $T\rho$ cached values, giving the lower bound $\Omega(T^2(1-\rho) + T\rho)$. □

Theorem 19 (Replay Convergence Guarantee)**.** *Algorithm 5 converges to the optimal sampling distribution with rate $O(1/\sqrt{T})$ as established in Theorem 5.*

Proof. The convergence follows from the softmax sampling mechanism:

- **Score Concentration**: The surprise metric concentrates around high-value experiences

Algorithm 5 Prioritized Replay with Surprise Weighting

```
Require: Experience $(s_t, \tau_t, a_t, r_t)$, buffers $\mathcal{B}_{\text{hot}}, \mathcal{B}_{\text{cold}}$, temperature $\beta$
 1: # Compute surprise metric
 2: $\texttt{score}(t) \leftarrow \|\tau_t - \mathbb{E}[\tau]\|^2 + \alpha r_t$
 3: # Buffer management
 4: if age$(s_t) \leq \tau_{\text{thresh}}$ then
 5:     Insert $(s_t, \tau_t, a_t, r_t, \texttt{score}(t))$ into $\mathcal{B}_{\text{hot}}$
 6: else
 7:     Insert $(s_t, \tau_t, a_t, r_t, \texttt{score}(t))$ into $\mathcal{B}_{\text{cold}}$
 8: end if
 9: # Softmax sampling
10: $P(t) \leftarrow \frac{\exp(\beta \cdot \texttt{score}(t))}{\sum_{t'} \exp(\beta \cdot \texttt{score}(t'))}$
11: Sample $\{t_i\}_{i=1}^{N} \sim P(t)$ from $\mathcal{B}_{\text{hot}} \cup \mathcal{B}_{\text{cold}}$
12: # Return sampled experiences
13: RETURN $\{(s_{t_i}, \tau_{t_i}, a_{t_i}, r_{t_i})\}_{i=1}^{N}$
```

- **Temperature Control**: $\beta > 0$ ensures proper exploration-exploitation balance
- **Buffer Diversity**: Hot/cold separation maintains sufficient experience diversity

The convergence rate $O(1/\sqrt{T})$ matches the theoretical analysis in Theorem 5. □

B.4.3 Implementation Details

This section provides comprehensive implementation details for reproducing EchoRL results, including architecture specifications, hyperparameter settings, and practical implementation considerations.

Table 3: EchoRL Implementation Configuration

Component	Parameter	Value
Architecture	Actors N_{actor}	128
	Learners N_{learner}	2
	Gradient sync	NCCL
	Experience transport	gRPC
Hyperparameters	KL weight λ_{KL}	0.1
	Hot buffer $\lvert\mathcal{B}_{\text{hot}}\rvert$	10^6
	Cold buffer $\lvert\mathcal{B}_{\text{cold}}\rvert$	10^7
	Temperature β	1.0
	PPO clip ϵ	0.2
	Learning rate α	3×10^{-4}
Hardware	GPUs	$8\times$ A100-80GB
	CPU cores	128
	Memory M_{total}	1 TB
	Bandwidth B_{net}	10 Gbps

Implementation Guidelines:

- **Trajectory Encoder:** Use a 3-layer MLP with hidden dimensions $[512, 256, 128]$ and ReLU activations. Apply layer normalization and dropout (rate 0.1) for stability.
- **Policy Network:** Implement as a 2-layer MLP with hidden dimension 256 and tanh activation. Use orthogonal initialization for better gradient flow.
- **KV Cache Management:** Implement LRU eviction with cache size 10^4. Use priority queues for efficient cache hit/miss detection.
- **Replay Buffer:** Use circular buffers with separate hot/cold storage. Implement lock-free operations for multi-threaded access.

- **Gradient Synchronization:** Use NCCL for efficient all-reduce operations. Apply gradient clipping with norm 1.0 to prevent instability.

Table 4: Mathematical Parameter Specifications

Mathematical Component	**Symbol**	**Specification**
Trajectory Space	Dimension d	512
	Window size k	8
	Lipschitz constant L	≤ 1.0
KV Cache	Reuse rate ρ	$[0.5, 0.7]$
	Cache size $\lvert\mathcal{C}\rvert$	10^4
	Hit probability P_{hit}	≥ 0.8
Replay Buffer	Priority decay γ	0.99
	Age threshold τ_{thresh}	1000
	Sampling variance σ^2	0.1
Convergence	Tolerance ϵ_{conv}	10^{-6}
	Max iterations T_{max}	10^6
	Batch size B	256