Cognitive Predictive Processing: A Human-inspired Framework for Adaptive Exploration in Open-World Reinforcement Learning

Boheng Liu[1], **Ziyu Li**[1] *, **Chenghua Duan**[1], **Yutian Liu**[1],
Zhuo Wang[1], **Xiuxing Li**[1], **Qing Li**[1], **Xia Wu**[1,2]
[1]Beijing Institute of Technology
[2]Tangshan Research Institute, Beijing Institute of Technology
boheng@bit.edu.cn, ziyuli@bit.edu.cn

Abstract

Open-world reinforcement learning challenges agents to develop intelligent behavior in vast exploration spaces. Recent approaches like LS-Imagine have advanced the field by extending imagination horizons through jumpy state transitions, yet remain limited by fixed exploration mechanisms and static jump thresholds that cannot adapt across changing task phases, resulting in inefficient exploration and lower completion rates. Humans demonstrate remarkable capabilities in open-world decision-making through a chain-like process of task decomposition, selective memory utilization, and adaptive uncertainty regulation. Inspired by human decision-making processes, we present Cognitive Predictive Processing (CPP), a novel framework that integrates three neurologically-inspired systems: a phase-adaptive cognitive controller that dynamically decomposes tasks into exploration, approach, and completion phases with adaptive parameters; a dual-memory integration system implementing dual-modal memory that balances immediate context with selective long-term storage; and an uncertainty-modulated prediction regulator that continuously updates environmental predictions to modulate exploration behavior. Comprehensive experiments in MineDojo demonstrate that these human-inspired decision-making strategies enhance performance over recent techniques, with success rates improving by an average of 4.6% across resource collection tasks while reducing task completion steps by an average of 7.1%. Our approach bridges cognitive neuroscience and reinforcement learning, excelling in complex scenarios that require sustained exploration and strategic adaptation while demonstrating how neural-inspired models can solve key challenges in open-world AI systems.

1 Introduction

Open-world reinforcement learning (RL) challenges agents to develop intelligent behavior in environments characterized by unbounded exploration spaces, diverse interaction possibilities, and limited perceptual information [1, 31, 35]. These environments demand flexible decision-making capabilities that can generalize across varied scenarios and objectives while operating under substantial uncertainty [12]. Contemporary embodied AI systems struggle with efficient exploration, contextual adaptation, and progressive learning in such complex domains [8], highlighting the need for more sophisticated cognitive architectures that can emulate human-inspired reasoning and adaptation [26].

Recent advancements in model-based reinforcement learning have made significant progress toward addressing these challenges [1, 11, 16, 28, 29, 31, 45]. DreamerV3 [16] employs world models to

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

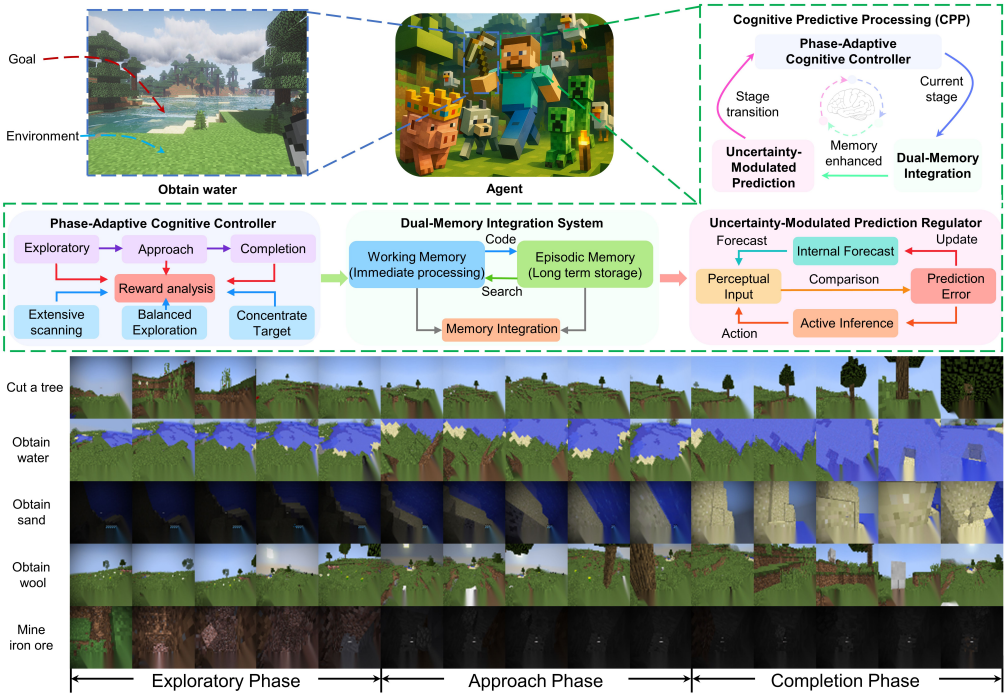


Figure 1: **Cognitive Predictive Processing architecture for open-world reinforcement learning.** The figure shows our integrated architecture that enhances agent exploration through three key components. Minecraft imagery and content shown in this figure are the property of Mojang Studios and Microsoft Corporation, used for academic illustration purposes.

improve sample efficiency through imagination-based planning, achieving impressive results across diverse domains. Building on this foundation, LS-Imagine [28] introduces jumpy state transitions guided by affordance maps, extending imagination horizons beyond the short-term constraints of previous approaches. While these methods have advanced the state of the art, they rely on fixed exploration mechanisms and static jump thresholds that cannot adapt to changing task phases. This limitation leads to inefficient exploration patterns and behavioral stagnation, particularly in complex scenarios requiring sustained exploration and strategic reasoning.

Humans excel in open-world environments through sophisticated cognitive processes, and unlike the fixed exploration mechanisms of existing algorithms, human behavior can be characterized by *three complementary cognitive strategies* that enable adaptive decision-making in a cascading process[15, 21]. First, when confronting complex tasks, humans naturally decompose problems into distinct phases [4, 13], transforming global uncertainty into manageable local challenges. This phase-specific decomposition allows for targeted optimization of exploration parameters tailored to each environmental context, creating a foundation for efficient problem-solving. Second, operating within this phase-structured framework, human behavior has been effectively modeled using selective dual-memory systems [39] that dynamically balance innovation and conservation. In these models, working memory focuses on immediate environmental features while episodic memory stores structured experiential. Third, as phases progress, humans have been observed to regulate exploration through predictive processing [41] that establishes adaptive feedback loops. By continuously comparing expected outcomes with actual environmental feedback, this mechanism dynamically adjusts exploration strategies, increasing precision in familiar contexts while promoting flexibility in novel situations. These integrated cognitive mechanisms, operating in a sequential yet complementary manner, enable humans to overcome precisely the static exploration strategies and contextual adaptation challenges that limit current AI approaches in open-world environments.

Inspired by human decision-making processes, we present Cognitive Predictive Processing (CPP), a framework that enhances open-world reinforcement learning by integrating principles from cognitive

neuroscience [23]. The phase-adaptive cognitive controller segments problems into exploration, approach, and completion phases with distinct computational characteristics, monitoring reward patterns and environmental features to determine phase transitions [42]. The dual-memory integration system parallels human working and episodic memory systems [10], providing contextual awareness and selective retention of significant experiences. The uncertainty-modulated prediction regulator applies hierarchical predictive coding [6], generating environmental predictions and responding to uncertainty through precision-weighted integration of sensory inputs and internal models. Figure 1 shows the integrated CPP architecture and how its key components interact to enable adaptive exploration in open-world environments.

Our contributions include: (1) Cognitive Predictive Processing, a novel framework that integrates human-inspired decision-making mechanisms into open-world reinforcement learning; (2) a phase-adaptive cognitive controller that draws inspiration from humans' natural problem decomposition into distinct phases, enabling adaptive exploration parameters across different task phases; (3) a dual-memory integration system inspired by human selective memory systems that balance immediate context with critical historical experiences, providing both short-term working memory and selective long-term storage; (4) an uncertainty-modulated prediction regulator that leverages principles from human predictive processing that adaptively responds to environmental uncertainty, adjusting exploration behavior through environmental prediction. Across multiple resource collection tasks, our framework not only improves success rates by an average of 4.6% but also reduces task completion steps by an average of 7.1%, demonstrating consistent performance advantages in structured tasks that require contextual adaptation and strategic planning.

2 Related Work

2.1 Open-World Reinforcement Learning

Open-world reinforcement learning confronts challenges of vast state spaces, sparse rewards, and long-horizon tasks. Recent advances in model-based approaches have significantly improved capabilities in such environments. DreamerV3 [16] extends earlier world model techniques [17, 18] by learning predictive representations from limited experiences, though these models typically struggle with limited temporal horizons [14]. Several approaches address long-horizon planning challenges: LS-Imagine [28] introduces jumpy imagination for efficient extended planning, while STEVE-1 [31], MineDreamer [46], and VPT [1] leverage language models or human demonstrations to bootstrap policies for complex environments like Minecraft. Despite these advances, exploration in open-world environments remains challenging, with most methods employing fixed exploration mechanisms that fail to adapt to changing task demands [19, 34]. Traditional exploration strategies including intrinsic motivation [27, 32] and novelty-seeking [36] show promise but employ static parameters and heuristics that cannot adapt to different phases of the learning process [33]. This contrasts with human exploration, which dynamically adjusts strategies based on task progress and environmental feedback. HyperX [49] introduces uncertainty-guided exploration through novelty bonuses and prediction discrepancies in meta-reinforcement learning contexts, though these exploration signals diminish during training and focus on cross-task adaptation rather than continuous within-task modulation. Recent approaches like SpatialVLA [37] and skill-oriented frameworks [45] implement more structured exploration but still rely on fixed exploration mechanisms rather than truly adaptive reasoning. Unlike these approaches, our work introduces a human-inspired framework that dynamically adjusts exploration strategies based on task progress, directly addressing the gap between static computational methods and flexible exploration in open-world environments.

2.2 Cognitive Mechanisms in Intelligent Systems

Cognitive-inspired approaches have shown considerable success in various AI domains through cognitive architectures like ACT-R [38] and SOAR [25] for structured problem-solving, while memory-augmented systems [7, 30] have improved sample efficiency in reinforcement learning tasks. Meanwhile, Botvinick et al. [3] highlight that structured cognitive processes remain crucial for developing capable RL agents. However, these traditional approaches have not been effectively applied for open-world exploration, where environments present unique challenges including vast exploration spaces, sparse rewards, and complex interactions. Human cognition excels in such real-world scenarios through flexible phase decomposition [42], memory integration [5, 24, 39],

and uncertainty prediction [6], offering valuable inspiration for open-world reinforcement learning. Our work addresses this research gap by integrating cognitive mechanisms into a unified framework specifically designed for open-world reinforcement learning, significantly enhancing performance in complex environments that have previously challenged conventional methods.

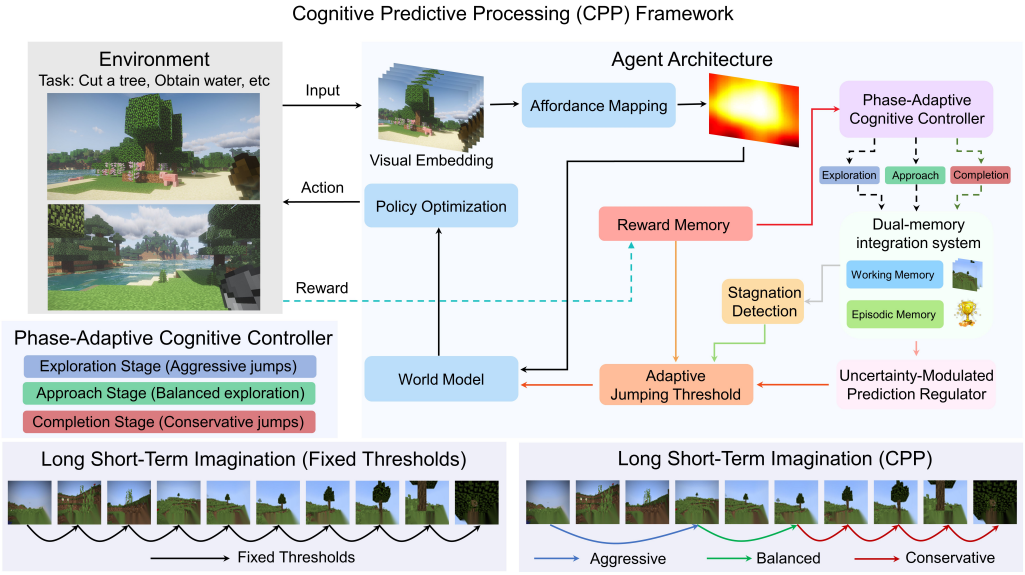


Figure 2: **Cognitive Predictive Processing framework.** The figure shows our integrated architecture with three key components: Phase-Adaptive Cognitive Controller for phase-based task decomposition, Dual-Memory Integration System for selective experience utilization, and Uncertainty-Modulated Prediction Regulator for uncertainty regulation.

3 Method

3.1 Problem formulation

We formulate cognitive predictive processing in open-world reinforcement learning as an enhanced partially observable Markov decision process (POMDP). Our approach operates on visual observations o_t while generating control signals a_t, receiving sparse environmental rewards r_t, and lacking access to internal environment states.

Our cognitive framework augments the traditional state-action space with a cognitive state vector $\psi_t = (m_t, \phi_t)$, where m_t represents the dual-memory state and ϕ_t indicates the cognitive phase. The world model learns on tuples $\{D_t = (o_t, a_t, r_t, \mathcal{M}_t, \psi_t, j_t, \Delta_t, G_t)\}$, where $\mathcal{M}_t$ represents the affordance map, j_t indicates the jumping flag, Δ_t denotes time steps between transitions, and G_t represents cumulative discounted rewards.

Our memory system maintains both short-term working memory $\mathcal{W}_t = \{(o_{t-k}, a_{t-k}, r_{t-k})\}_{k=1}^{K}$ with capacity K, and long-term episodic memory $\mathcal{E}_t = \{(o_i, a_i, r_i, s_i)\}_{i \in \mathcal{I}}$, where s_i (detailed in Appendix A.2) represents the surprise value and $\mathcal{I}$ denotes indices of significant experiences. The surprise value s_i quantifies experience significance through reward prediction error, observation prediction error, and cognitive state transitions.

The phase-adaptive cognitive controller decomposes tasks into phases $\phi_t \in \{\phi^{explore}, \phi^{approach}, \phi^{complete}\}$, each with corresponding exploration parameters $\eta(\phi_t) = (w^{explore}(\phi_t), b^{jump}(\phi_t))$ that control exploration weight and jump bias. Phase transitions follow $P(\phi_{t+1}|\phi_t, o_t, r_t, \mathcal{W}_t, \mathcal{E}_t) \propto f_\phi(o_t, r_t, \mathcal{W}_t, \mathcal{E}_t)$, where f_ϕ integrates observations, rewards and memory information.

Precision weighting $\omega_t = f_\omega(\varepsilon_{t-1:t-N})$ modulates the balance between sensory input and predictions over a window of N previous steps. Policy learning occurs on trajectories $\{(h_t, z_t, a_t, r_t, \psi_t, j_t, \Delta_t, G_t)\}$, where h_t represents the deterministic recurrent state and z_t represents the stochastic state. The mapping $(h_t, z_t) = f_{enc}(o_t, \mathcal{M}_t)$ is performed by the world model's encoder, transforming observations and affordance maps into internal representations.

3.2 Overview of cognitive predictive processing

In this section, we present the details of Cognitive Predictive Processing, which enhances open-world reinforcement learning through the following algorithm steps:

1. **Phase-adaptive cognitive controller** (Sec. 3.3): We continuously monitor task progress through reward patterns and behavioral sequences, dynamically classifying the current state into exploration, approach, or completion phases with corresponding parameter configurations.
2. **Dual-memory integration system** (Sec. 3.4): We implement a two-stream memory system that combines short-term working memory for immediate context with selective long-term episodic memory for significant experiences.
3. **Uncertainty-modulated prediction regulator** (Sec. 3.5): The agent maintains a hierarchical predictive model generating expectations about environmental states, with prediction errors driving both learning and adaptive parameter tuning based on uncertainty estimates.
4. **Phase-adaptive world modeling** (Sec. 3.6): The world model and policy are tested with phase-specific parameters that adapt to the current cognitive phase, integrating memory retrievals and predictive signals for context-sensitive decision-making.
5. **Data collection**: During interaction, we collect enhanced data including cognitive states, memory activations, and prediction errors alongside traditional observations and rewards, feeding this information back into the learning process.
6. Iterate Steps $1-5$.

The integrated architecture is illustrated in Figure 2, showing how our cognitive mechanisms interact to enhance agent exploration.

3.3 Phase-adaptive cognitive controller

Inspired by human problem-solving strategies, we implement a cognitive controller that decomposes complex tasks into meaningful phases. This controller continuously monitors task progress and adaptively classifies the current state into one of three cognitive phases:

$$\phi_t \in \{\phi^{explore}, \phi^{approach}, \phi^{complete}\} \tag{1}$$

Each phase represents a distinct cognitive mode with specialized parameter configurations that optimize the exploration-exploitation trade-off. For each phase, we define specific exploration parameters:

$$\eta(\phi_t) = (w^{explore}(\phi_t), b^{jump}(\phi_t)) \tag{2}$$

where $w^{explore}(\phi_t)$ controls the exploration weight and $b^{jump}(\phi_t)$ determines the jump bias, with high exploration during the exploration phase, balanced parameters in the approach phase, and low exploration with conservative jump bias in the completion phase.

The phase classification process integrates multiple information sources:

$$P(\phi_{t+1}|\phi_t, o_t, r_t, \mathcal{W}_t, \mathcal{E}_t) \propto f_\phi(o_t, r_t, \mathcal{W}_t, \mathcal{E}_t) \tag{3}$$

where f_ϕ is a phase classification function, phase transitions are governed by a progress tracking mechanism detailed in Appendix A.1.

3.4 Dual-memory integration system

Our memory architecture simulates human cognitive processes by implementing parallel memory systems. Working memory provides immediate context while episodic memory stores significant experiences. These systems interact to guide decision-making:

$$m_t = (\mathcal{W}_t, \mathcal{E}_t) \tag{4}$$

The working memory $\mathcal{W}_t$ maintains a limited capacity buffer of recent experiences:

$$\mathcal{W}_t = \{(o_{t-k}, a_{t-k}, r_{t-k})\}_{k=1}^{K} \tag{5}$$

where $K = 20$ represents capacity constraint simulating human working memory limitations. Working memory provides immediate context for decision-making through a similarity-based attention mechanism:

$$\alpha_t(k) = \frac{\exp(\cos(f(o_t), f(e.o)))}{\sum_{j=1}^{K} \exp(\cos(f(o_t), f(e.o)))} \tag{6}$$

where $\cos(f(o_t), f(e.o))$ measures the cosine similarity between current observation and episodic memories, f extracts feature representations from observations, and $\alpha_t(k)$ represents the attention weight for experience at time $t - k$.

The episodic memory $\mathcal{E}_t$ selectively stores experiences based on their surprise value s_i:

$$\mathcal{E}_t = \mathcal{E}_{t-1} \cup \{(o_t, a_t, r_t, \mathcal{C}_t)\} \text{ if } s_t > \tau_t \tag{7}$$

where s_t quantifies experience significance through reward prediction error and cognitive state transitions, and τ_t (defined in Appendix A.2) is a dynamic threshold that adapts based on recent experiences.

This dual-modal memory system enables the agent to simultaneously leverage immediate context through working memory and historical successes through episodic memory, creating a cognitive foundation that supports higher-level reasoning processes. Detailed memory retrieval and consolidation mechanisms are provided in Appendix A.2.

3.5 Uncertainty-modulated prediction regulator

Our uncertainty-modulated prediction regulator implements a hierarchical model that generates expectations about future states and establishes adaptive feedback loops, where prediction errors not only drive learning but also dynamically adjust exploration strategies. The system maintains predictions at multiple levels:

$$\begin{aligned} \hat{o}_{t+1} &= f_o(h_t, z_t, a_t, \phi_t) \\ \hat{r}_{t+1} &= f_r(h_t, z_t, a_t, \phi_t) \end{aligned} \tag{8}$$

where f_o and f_r are neural network prediction functions that generate next-observation and next-reward forecasts respectively. These functions form the core of the world model's predictive capabilities, enabling anticipatory reasoning across different cognitive phases. Prediction errors between expectations and observations drive both learning and adaptive behavior:

$$\begin{aligned} \varepsilon_t^o &= ||o_t - \hat{o}_t||_2 \\ \varepsilon_t^r &= |r_t - \hat{r}_t| \end{aligned} \tag{9}$$

These errors are combined into an uncertainty estimate that modulates exploration parameters:

$$\mathcal{U}_t = \omega_o \cdot \varepsilon_t^o + \omega_r \cdot \varepsilon_t^r + \omega_{trend} \cdot \varepsilon_{trend}^r \tag{10}$$

where ε^r_{trend} represents the reward trend error (formally defined in Appendix A.3). Higher uncertainty encourages broader exploration, while lower uncertainty enables more focused behavior. This mechanism adapts jump thresholds based on environmental predictability, fostering effective exploration in new areas while exploiting familiar regions. The detailed implementation of uncertainty estimation and parameter adaptation is provided in Appendix A.3.

3.6 Phase-adaptive world modeling

Our integrated cognitive architecture enhances both world model learning and policy optimization by incorporating phase-specific parameters, memory retrievals, and uncertainty estimates into a unified framework. The world model architecture extends typical model-based reinforcement learning with cognitive conditioning:

$$\begin{aligned} \hat{z}_t &\sim p_\theta(\hat{z}_t | h_t, \phi_t) \\ \hat{r}_t &\sim p_\theta(\hat{r}_t | h_t, z_t, \phi_t) \\ \hat{o}_t &\sim p_\theta(\hat{o}_t | h_t, z_t, \phi_t) \end{aligned} \tag{11}$$

Each component is conditioned on the current cognitive phase ϕ_t, enabling specialized processing for different reasoning phases. The jumping mechanism incorporates phase-specific thresholds and uncertainty estimates to adaptively balance exploration:

$$j_t = \mathbb{I}[\max(a_t) > \theta_t^{jump}(\phi_t, \mathcal{U}_t)] \tag{12}$$

where j_t is the jumping indicator and θ_t^{jump} is the adaptive threshold described in Appendix A.3.

Memory retrievals provide relevant historical context for current decisions:

$$\mathcal{R}_t = \{e_j\}_{j=1}^{K_r} \text{ where } j \in \text{argsort}_{e \in \mathcal{E}_t}(\cos(f(o_t), f(\phi_t))) \tag{13}$$

For behavior learning, we adapt the actor-critic approach with memory-enhanced and phase-aware components:

$$\begin{aligned} a_t &\sim \pi_\psi(a_t | h_t, z_t, \mathcal{R}_t, \phi_t) \\ V_t &= v_\psi(h_t, z_t, \mathcal{R}_t, \phi_t) \end{aligned} \tag{14}$$

This integrated approach transforms how the agent processes information and makes decisions across different task phases, enabling adaptive exploration that becomes increasingly focused as tasks progress from exploration to completion.

4 Experiments

This section details our comprehensive evaluation of the Cognitive Predictive Processing framework in open-world reinforcement learning scenarios. Our evaluation focuses on two key objectives: (1) comparing CPP against state-of-the-art approaches in overall task performance, and (2) analyzing the contribution of individual cognitive components to agent capabilities.

4.1 Experimental setup

4.1.1 Benchmark and tasks

We evaluate our approach on the challenging MineDojo benchmark [9], which provides a diverse set of open-world tasks in Minecraft. Following previous work, we focus on five representative tasks requiring different exploration and interaction strategies: (1) harvest log in plains, (2) harvest water with bucket, (3) harvest sand, (4) shear sheep, and (5) mine iron ore. For fair comparison, we maintain the same environmental configurations across all methods, including identical starting conditions, maximum episode lengths, and reward structures as detailed in Table 1. All experimental results represent averages from three independent runs to ensure statistical reliability.

Table 1: Details of evaluation tasks in MineDojo benchmark.

Task	Description	Initial Tools	Initial Environment	Max Steps
Harvest log in plains	"Cut a tree"	None	Plains biome	1000
Harvest water	"Obtain water"	Bucket	Near water source	1000
Harvest sand	"Obtain sand"	None	Near beach	1000
Shear sheep	"Obtain wool"	Shears	Sheep within 15 blocks	1000
Mine iron ore	"Mine iron ore"	Stone pickaxe	Underground cave system	2000

Table 2: Success rates (%) across MineDojo tasks (mean ± std over 3 runs). Results for VPT, STEVE-1, and DreamerV3 are from LS-Imagine [28], while LS-Imagine and CPP results represent our evaluations.

Model	Cut a tree	Obtain water	Obtain sand	Obtain wool	Mine iron ore
VPT [1]	6.97	0.61	12.99	1.94	0.00
STEVE-1 [31]	57.00	6.00	37.00	3.00	0.00
DreamerV3 [16]	53.33	55.72	59.88	25.13	**16.79**
LS-Imagine [28]	88.33±3.21	75.67±3.93	57.33±4.69	56.67±4.43	10.66±1.51
CPP (Ours)	**90.00±2.65**	**80.33±3.79**	**60.00±4.58**	**69.33±6.04**	12.00±1.83

4.1.2 Baselines and implementation

We compare CPP against state-of-the-art methods in open-world reinforcement learning: DreamerV3 [16], LS-Imagine [28], STEVE-1 [31], and VPT [1]. Our implementation builds upon the LS-Imagine codebase, integrating our cognitive components while maintaining the core world model architecture. All models are trained for 1×10^6 environment steps. Detailed implementation specifics and hyperparameter settings are provided in Appendix B.

4.2 Performance evaluation

4.2.1 Overall task success

Table 2 demonstrates that CPP outperforms baseline methods in the majority of evaluated tasks. In the harvest log task in plains, CPP achieves a 90.00% success rate compared to 88.33% for LS-Imagine, representing a 1.9% relative improvement. The performance differential increases in the harvest water task, where CPP reaches 80.33% success rate compared to LS-Imagine's 75.67%, constituting a 6.2% relative improvement. For the obtain sand task, CPP delivers a consistent improvement of 4.7% beyond LS-Imagine. The most significant performance gain appears in the obtain wool task with a 22.3% relative improvement, demonstrating CPP's efficacy in scenarios involving mobile targets. Unlike static resources in other tasks, sheep movement introduces additional uncertainty, requiring continuous adaptation of exploration strategies and contextual awareness to track and approach dynamic objectives.

In the mine iron ore task, DreamerV3 achieves 16.79% success rate compared to CPP's 12.00%. This specific difference stems from DreamerV3's architecture that particularly excels in enclosed spaces with limited visibility. Despite this, CPP still outperforms LS-Imagine in this complex scenario, demonstrating a 12.6% relative improvement. Across all other tasks, CPP consistently outperforms both DreamerV3 and LS-Imagine, confirming our framework's effectiveness in most open-world environments while revealing specific optimization opportunities for future work.

4.2.2 Sample efficiency

Beyond success rates, Table 3 reveals CPP's efficiency characteristics across different task types. CPP excels in structured interaction tasks, achieving first success in the harvest log task in just 340.48 steps compared to 363.27 steps for LS-Imagine and 711.22 steps for DreamerV3, a 52.1% reduction versus the latter. This efficiency extends to water collection (3.6% reduction) and wool gathering (18.2% reduction), where CPP's phase-adaptive controller effectively transitions between exploration modes. DreamerV3's fixed exploration parameters prove advantageous in homogeneous resource

Table 3: Average steps to first success across MineDojo tasks (mean ± std over 3 runs). Results for VPT, STEVE-1, and DreamerV3 are from LS-Imagine [28], while LS-Imagine and CPP results represent our evaluations.

Model	Cut tree	Get water	Get sand	Get wool	Mine iron
VPT [1]	963.32	987.65	880.54	987.49	-
STEVE-1 [31]	752.47	989.07	770.40	992.36	-
DreamerV3 [16]	711.22	628.79	**548.76**	841.14	**1789.06**
LS-Imagine [28]	363.27±28.45	473.05±35.62	612.27±42.18	558.04±31.27	1874.07±127.35
CPP (Ours)	**340.48±22.13**	**455.81±29.84**	567.84±38.76	**456.45±39.52**	1867.53±115.68

Table 4: Ablation results showing success rates (%) across tasks (mean ± std over 3 runs). "PACC": Phase-Adaptive Cognitive Controller. "DMIS": Dual-Memory Integration System. "UMPR": Uncertainty-Modulated Prediction Regulator.

Model Variant	Cut a tree	Obtain water	Obtain sand	Obtain wool	Mine iron ore
Full CPP	90.00±2.65	80.33±3.79	60.00±4.58	69.33±6.04	12.00±1.83
w/o PACC	84.33±4.16	72.67±5.28	54.33±6.12	52.00±7.35	7.33±3.21
w/o DMIS	86.67±3.51	75.00±4.73	57.67±5.46	61.67±5.89	8.67±3.06
w/o UMPR	87.33±3.28	76.67±4.52	55.00±5.83	63.00±5.24	9.33±2.94

identification tasks like sand collection (548.76 versus 567.84 steps) and iron mining (1789.06 versus 1867.53 steps). Notably, water collection shows different patterns: water bodies possess distinctive visual features (blue coloration, reflective properties, flowing patterns) that create strong perceptual signals our dual-memory system effectively encodes and retrieves, whereas sand blocks have more subtle visual characteristics that can be confused with similar terrain elements, making DreamerV3's consistent exploration more effective for such visually ambiguous but static resources. This complementary performance highlights CPP's design intention: while DreamerV3 excels in direct resource identification through consistent exploration, CPP prioritizes adaptive behavior across changing task phases. This tradeoff delivers superior performance in most realistic scenarios requiring contextual adaptation, especially in complex tasks with dynamic elements. Different architectural approaches optimize for different environmental characteristics, explaining the observed performance patterns. Additional trajectory analysis in Appendix C further illustrates these task-specific efficiency patterns.

4.3 Ablation studies

To understand the contribution of each cognitive component, we perform ablation studies by systematically removing individual mechanisms from the full CPP framework. Table 4 presents these results across all tasks. Removing the cognitive controller produces the largest performance drops, particularly in complex tasks with 24.99% relative decrease in shear sheep task. The dual-memory system shows substantial impact on water collection (5.33 percentage points) and shear sheep (7.66 percentage points) tasks, reflecting its importance for contextual adaptation. While our uncertainty quantification uses weighted linear combinations with fixed parameters rather than full Bayesian approaches, this design choice balances computational efficiency with biological plausibility, as neuroscience evidence suggests that uncertainty representations in biological systems often manifest as relatively simple combinations of prediction errors [48]. The consistent performance contributions across diverse tasks validate this simplified yet effective approach.

Each ablated variant maintains reasonable performance, demonstrating that individual cognitive mechanisms provide complementary capabilities. However, the full integration yields optimal results across all task categories, confirming that human-inspired cognitive mechanisms work synergistically in open-world reinforcement learning.

5 Conclusions and limitations

Our experimental results demonstrate that Cognitive Predictive Processing enhances reinforcement learning in open-world environments through three synergistic components: a phase-adaptive cognitive controller for task decomposition, a dual-memory integration system for experience utilization, and an uncertainty-modulated prediction regulator for uncertainty regulation. Compared to state-of-the-art methods, CPP significantly improves success rates (by 6.2% in water collection and 22.3% in wool collection tasks) while reducing completion steps by up to 6.3%, confirming the effectiveness of integrating human-inspired cognitive principles into reinforcement learning systems.

Despite these advances, several limitations remain. A notable limitation appears in the mine iron ore task, where our method did not match DreamerV3's performance, suggesting that complex navigation in confined spaces with limited visibility requires specialized mechanisms beyond our current implementation. Additionally, our three-phase decomposition framework (exploration, approach, completion) is well-suited for goal-directed tasks with spatial components, but may face challenges in certain task categories: highly unstructured environments where phase boundaries blur significantly, extremely long-horizon tasks requiring numerous intermediate subgoals where simple three-phase decomposition may be insufficient, and highly adversarial settings requiring rapid replanning due to dynamic obstacles.

Future work could address these limitations through enhanced spatial reasoning components, broader environment evaluations, and closer integration with neuroscientific models. A particularly promising direction involves enabling adaptive learning rules where the system could "learn how to learn" through meta-gradient approaches, allowing learning parameters to evolve based on accumulated task experience rather than remaining static [47]. Furthermore, extending CPP to complex multi-stage goals (such as crafting tasks requiring sequential resource collection) would leverage our dual-memory system's ability to organize experiences across subtasks, while adaptation to goal-conditioned settings could be achieved by conditioning phase classification and memory retrieval on goal embeddings. Extending CPP to real-world applications, particularly in domains such as assistive robotics where adaptive exploration strategies could benefit navigation and manipulation tasks in human-centered environments, represents an exciting and impactful future direction for this research.

Acknowledgement

This work was supported by the State Key Program of National Natural Science of China (Grant No. 62236001), Hebei Natural Science Foundation (Grant No. F2024105033), and Beijing Municipal Natural Science Foundation (Grant No. L247011).

References

[1] Baker B, Akkaya I, Zhokov P, et al. Video pretraining (vpt): Learning to act by watching unlabeled online videos[J]. Advances in Neural Information Processing Systems, 2022, 35: 24639-24654.

[2] Bavard S, Lebreton M, Khamassi M, et al. Reference-point centering and range-adaptation enhance human reinforcement learning at the cost of irrational preferences[J]. Nature communications, 2018, 9(1): 4503.

[3] Botvinick M, Ritter S, Wang J X, et al. Reinforcement learning, fast and slow[J]. Trends in cognitive sciences, 2019, 23(5): 408-422.

[4] Botvinick M M, Niv Y, Barto A G. Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective[J]. cognition, 2009, 113(3): 262-280.

[5] Chandra S, Sharma S, Chaudhuri R, et al. Episodic and associative memory from spatial scaffolds in the hippocampus[J]. Nature, 2025: 1-13.

[6] Chang H, Tang W, Wulf A M, et al. Sleep microstructure organizes memory replay[J]. Nature, 2025: 1-9.

[7] Chateau-Laurent H, Alexandre F. Contextual Control of Hopfield Networks in a Hippocampal Model[C]//Proceedings of the Annual Meeting of the Cognitive Science Society. 2024, 46.

[8] Fan L, Liang M, Li Y, et al. Evidential active recognition: Intelligent and prudent open-world embodied perception[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024: 16351-16361.

[9] Fan L, Wang G, Jiang Y, et al. Minedojo: Building open-ended embodied agents with internet-scale knowledge[J]. Advances in Neural Information Processing Systems, 2022, 35: 18343-18362.

[10] Fan Y, Wang M, Fang F, et al. Two-dimensional neural geometry underpins hierarchical organization of sequence in human working memory[J]. Nature Human Behaviour, 2025, 9(2): 360-375.

[11] Feng T, Wang X, Zhou Z, et al. EvoAgent: Agent Autonomous Evolution with Continual World Model for Long-Horizon Tasks[J]. arXiv preprint arXiv:2502.05907, 2025.

[12] Fujimoto T, Suetterlein J, Chatterjee S, et al. Assessing the impact of distribution shift on reinforcement learning performance[J]. arXiv preprint arXiv:2402.03590, 2024.

[13] Gershman S J. Deconstructing the human algorithms for exploration[J]. Cognition, 2018, 173: 34-42.

[14] Guo Z, Thakoor S, Pîslar M, et al. Byol-explore: Exploration by bootstrapped prediction[J]. Advances in neural information processing systems, 2022, 35: 31855-31870.

[15] Hassabis D, Kumaran D, Summerfield C, et al. Neuroscience-inspired artificial intelligence[J]. Neuron, 2017, 95(2): 245-258.

[16] Hafner D, Pasukonis J, Ba J, et al. Mastering diverse control tasks through world models[J]. Nature, 2025: 1-7.

[17] Hafner D, Lillicrap T, Ba J, et al. Dream to control: Learning behaviors by latent imagination[J]. arXiv preprint arXiv:1912.01603, 2019.

[18] Hafner D, Lillicrap T, Norouzi M, et al. Mastering atari with discrete world models[J]. arXiv preprint arXiv:2010.02193, 2020.

[19] Hafner D, Pasukonis J, Ba J, et al. Mastering diverse domains through world models[J]. arXiv preprint arXiv:2301.04104, 2023.

[20] Hu J, Lepori M A, Franke M. Linking forward-pass dynamics in Transformers and real-time human processing[J]. arXiv preprint arXiv:2504.14107, 2025.

[21] Lee D, Seo H, Jung M W. Neural basis of reinforcement learning and decision making[J]. Annual review of neuroscience, 2012, 35(1): 287-308.

[22] Kaefer K, Stella F, McNaughton B L, et al. Replay, the default mode network and the cascaded memory systems model[J]. Nature Reviews Neuroscience, 2022, 23(10): 628-640.

[23] Kriegeskorte N, Douglas P K. Cognitive computational neuroscience[J]. Nature neuroscience, 2018, 21(9): 1148-1160.

[24] Kveim V A, Salm L, Ulmer T, et al. Divergent recruitment of developmentally defined neuronal ensembles supports memory dynamics[J]. Science, 2024, 385(6710): eadk0997.

[25] Laird J E. The Soar cognitive architecture[M]. MIT press, 2019.

[26] LeCun Y. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27[J]. Open Review, 2022, 62(1): 1-62.

[27] Li H, Yang X, Wang Z, et al. Auto mc-reward: Automated dense reward design with large language models for minecraft[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024: 16426-16435.

[28] Li J, Wang Q, Wang Y, et al. Open-World Reinforcement Learning over Long Short-Term Imagination[C]. ICLR, 2025.

[29] Li Z, Xu X, Xu Z, et al. LARM: Large Auto-Regressive Model for Long-Horizon Embodied Intelligence[J]. arXiv preprint arXiv:2405.17424, 2024.

[30] Li Z, Zhu D, Hu Y, et al. Neural episodic control with state abstraction[J]. arXiv preprint arXiv:2301.11490, 2023.

[31] Lifshitz S, Paster K, Chan H, et al. Steve-1: A generative model for text-to-behavior in minecraft[J]. Advances in Neural Information Processing Systems, 2023, 36: 69900-69929.

[32] Liu B, Li X, Zhang J, et al. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems[J]. arXiv preprint arXiv:2504.01990, 2025.

[33] Luo S, Jiang M, Zhang S, et al. Experiment-free exoskeleton assistance via learning in simulation[J]. Nature, 2024, 630(8016): 353-359.

[34] Meng Y, Bing Z, Yao X, et al. Preserving and combining knowledge in robotic lifelong reinforcement learning[J]. Nature Machine Intelligence, 2025: 1-14.

[35] Parmar J, Chouhan S, Raychoudhury V, et al. Open-world machine learning: applications, challenges, and opportunities[J]. ACM Computing Surveys, 2023, 55(10): 1-37.

[36] Peng X B, Guo Y, Halper L, et al. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters[J]. ACM Transactions On Graphics (TOG), 2022, 41(4): 1-17.

[37] Qu D, Song H, Chen Q, et al. SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Model[J]. arXiv preprint arXiv:2501.15830, 2025.

[38] Ritter F E, Tehranchi F, Oury J D. ACT−R: A cognitive architecture for modeling cognition[J]. Wiley Interdisciplinary Reviews: Cognitive Science, 2019, 10(3): e1488.

[39] Sun W, Advani M, Spruston N, et al. Organizing memories for generalization in complementary learning systems[J]. Nature neuroscience, 2023, 26(8): 1438-1448.

[40] Tamura M, Spellman T J, Rosen A M, et al. Hippocampal-prefrontal theta-gamma coupling during performance of a spatial working memory task[J]. Nature communications, 2017, 8(1): 2182.

[41] Trudel N, Scholl J, Klein-Flügge M C, et al. Polarity of uncertainty representation during exploration and exploitation in ventromedial prefrontal cortex[J]. Nature Human Behaviour, 2021, 5(1): 83-98.

[42] Wang Y, Chiew V. On the cognitive process of human problem solving[J]. Cognitive systems research, 2010, 11(1): 81-92.

[43] Wittmann M K, Lin Y, Pan D, et al. Basis functions for complex social decisions in dorsomedial frontal cortex[J]. Nature, 2025: 1-11.

[44] Yang S, Nachum O, Du Y, et al. Foundation models for decision making: Problems, methods, and opportunities[J]. arXiv preprint arXiv:2303.04129, 2023.

[45] Yuan H, Zhang C, Wang H, et al. Skill reinforcement learning and planning for open-world long-horizon tasks[J]. arXiv preprint arXiv:2303.16563, 2023.

[46] Zhou E, Qin Y, Yin Z, et al. Minedreamer: Learning to follow instructions via chain-of-imagination for simulated-world control[J]. arXiv preprint arXiv:2403.12037, 2024.

[47] Sousa M, Bujalski P, Cruz B F, et al. A multidimensional distributional map of future reward in dopamine neurons[J]. Nature, 2025: 1-9.

[48] Trudel N, Scholl J, Klein-Flügge M C, et al. Polarity of uncertainty representation during exploration and exploitation in ventromedial prefrontal cortex[J]. Nature Human Behaviour, 2021, 5(1): 83-98.

[49] Zintgraf L M, Feng L, Lu C, et al. Exploration in approximate hyper-state space for meta reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2021: 12991-13001.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract and introduction accurately present our contributions: CPP framework with three components, performance improvements (4.6% success rate increase, 7.1% step reduction), and applications. These claims are supported by experiments in Sec. 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Sec. 5, acknowledging simplified cognitive approximations and performance limitations in complex navigation tasks. Additional failure mode analysis is provided in Appendix C with visualizations in Figures 4-5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Mathematical formulation is presented in Sec. 3 (equations 1-14) with implementation details in appendices (equations 15-28). All components are formally specified with proper notation.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes] .

Justification: Sec. 4.1 and Appendix B details the MineDojo benchmark, tasks, and baselines. Appendix B contains hyperparameters (Table 5) and environment setup. Algorithm 1 provides complete pseudocode.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We provide detailed algorithmic descriptions (Algorithm 1), mathematical formulations (Sec. 3, equations 1-14), hyperparameters (Table 5), and implementation details (Appendix B) sufficient for reproduction. The code will not be publicly released due to commercial considerations.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 4.1 describes evaluation tasks and baselines. Training details are in Appendix B. Hyperparameters are in Table 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: We report standard deviations for all main experimental results based on three independent runs with different random seeds.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix B.2 specifies "NVIDIA L40s GPUs" and "1×10^6 environment steps" for training.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We followed the NeurIPS Code of Ethics. Research focuses on reinforcement learning in gaming environments without sensitive data or harmful applications. Minecraft content is appropriately credited to Mojang/Microsoft.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This work is not related to any private or personal data, and there's no explicit negativesocial impacts.This research advances fundamental AI capabilities without direct harmful applications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: Paper does not release models or data with misuse risks. Research focuses on reinforcement learning in Minecraft environments only.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

Justification: Minecraft content attribution in Figure 1, MineDojo benchmark [9] properly cited, and all baselines (DreamerV3, LS-Imagine, STEVE-1, VPT) referenced [16]-[1].

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No human subjects involved; all experiments conducted in simulation environments.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Research methodology is based on reinforcement learning with cognitive mechanisms; no LLMs used.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Detailed Implementation of Cognitive Mechanisms

A.1 Phase-Adaptive Cognitive Controller

Phase transitions in the cognitive controller are governed by a progress tracking mechanism that integrates reward trends and memory activations:

$$\text{Progress}_t = \frac{\sum_{i=t-k}^{t} r_i}{k} + \max_{e \in \mathcal{E}_t} \cos(f(o_t), f(e.o)) \tag{15}$$

where $\cos(f(o_t), f(e.o))$ measures the similarity between current observation and episodic memories, k is the reward averaging window size. When progress exceeds phase-specific thresholds, the system transitions to the next cognitive phase:

$$\phi_t = \begin{cases} \phi^{approach} & \text{if } \phi_{t-1} = \phi^{explore} \text{ and Progress}_t > \tau^{explore} \\ \phi^{complete} & \text{if } \phi_{t-1} = \phi^{approach} \text{ and Progress}_t > \tau^{approach} \\ \phi_{t-1} & \text{otherwise} \end{cases} \tag{16}$$

where $\tau^{explore} = 0.4$ and $\tau^{approach} = 0.7$ are threshold values for phase transitions. The system implements stagnation detection based on consecutive low-reward episodes:

$$\text{Stagnation}_t = \mathbb{I}\left[\sum_{i=t-m}^{t} \mathbb{I}[r_i < \epsilon] > \lambda \cdot m\right] \tag{17}$$

where $m = 50$ is the detection window, $\epsilon = 0.05$ represents the reward significance threshold below which steps are considered non-productive, and $\lambda = 0.8$ is the required proportion of low-reward steps. Upon detecting stagnation, the system reverts to exploration:

$$\phi_t = \phi^{explore} \text{ if Stagnation}_t = 1 \tag{18}$$

This dynamic adaptation ensures recovery from unproductive behavioral patterns by resetting to exploratory strategies when necessary.

A.2 Dual-memory integration system

The surprise value for episodic memory encoding quantifies the significance of an experience through three components:

$$s_t = |r_t - \hat{r}_t| + ||o_t - \hat{o}_t||_2 + \mathbb{I}[\phi_t \neq \phi_{t-1}] \tag{19}$$

where $\hat{r}_t$ is the predicted reward based on current policy, $\hat{o}_t$ is the predicted observation, and $\mathbb{I}[\phi_t \neq \phi_{t-1}]$ is an indicator function that equals 1 when a cognitive phase transition occurs.

The surprise threshold τ_t adapts based on a moving average of recent surprise values:

$$\tau_t = \beta \cdot \tau_{t-1} + (1 - \beta) \cdot \text{median}(\{s_{t-k}\}_{k=1}^{K_s}) \tag{20}$$

where $\beta = 0.9$ is a smoothing factor and $K_s = 50$ is the window size for computing the median surprise. This mechanism ensures that as the agent improves, only increasingly significant experiences are stored.

Episodic memory retrieval operates through similarity-based access:

$$\text{sim}(o_t, e.o, \phi_t, e.\phi) = \gamma \cdot \cos(f(o_t), f(e.o)) + (1 - \gamma) \cdot \mathbb{I}[\phi_t = e.\phi] \tag{21}$$

where f extracts feature representations from observations, $\cos$ computes cosine similarity, and $\gamma = 0.7$ balances the importance of observation similarity versus cognitive phase matching. The top-K retrieval operation selects the most relevant episodic memories:

$$\mathcal{R}_t = \{e_j\}_{j=1}^{K_r} \text{ where } j \in \text{argsort}_{e \in \mathcal{E}_t}(\text{sim}(o_t, e.o, \phi_t, e.\phi)) \tag{22}$$

with $K_r = 5$ representing the number of retrieved memories. These retrieved memories provide relevant historical context that enhances decision-making, particularly during similar situations or cognitive phases encountered previously.

A.3 Uncertainty-modulated prediction regulator

Our uncertainty-modulated prediction regulator calculates prediction errors across multiple channels to estimate environmental uncertainty:

$$\begin{aligned} \varepsilon_t^o &= ||o_t - \hat{o}_t||_2 \\ \varepsilon_t^r &= |r_t - \hat{r}_t| \\ \varepsilon_{trend}^r &= \left| \frac{1}{N_r} \sum_{i=t-N_r}^{t} r_i - r_{threshold} \right| \end{aligned} \tag{23}$$

where $N_r = 20$ is the reward history window size and $r_{threshold} = 0.05$ is the reward significance threshold. These errors contribute to the overall uncertainty estimate:

$$\mathcal{U}_t = \omega_o \cdot \varepsilon_t^o + \omega_r \cdot \varepsilon_t^r + \omega_{trend} \cdot \varepsilon_{trend}^r \tag{24}$$

with fixed weights $\omega_o = 0.5$, $\omega_r = 0.3$, and $\omega_{trend} = 0.2$.

Uncertainty directly modulates the jump threshold used for extended planning:

$$\theta_t^{jump} = \theta_{base}^{jump}(\phi_t) + \mathcal{T}_t - \mathcal{S}_t \tag{25}$$

where $\theta_{base}^{jump}(\phi_t)$ represents phase-specific base thresholds as detailed in Table 5.

The trend adjustment $\mathcal{T}_t$ responds to recent reward patterns:

$$\mathcal{T}_t = \begin{cases} -0.1 & \text{if } \bar{r}_t < 0.05 \\ 0.1 & \text{if } \bar{r}_t > 0.2 \\ 0.0 & \text{otherwise} \end{cases} \tag{26}$$

where $\bar{r}_t$ is the moving average of recent rewards, computed as the arithmetic mean over a sliding window of the 20 most recent reward signals.

The stagnation adjustment increases linearly with consecutive no-reward steps:

$$\mathcal{S}_t = \min(0.15, 0.01 \cdot c_t) \tag{27}$$

The stagnation counter c_t tracks periods without significant rewards:

$$c_t = \begin{cases} 0 & \text{if } r_t > 0.05 \\ c_{t-1} + 1 & \text{otherwise} \end{cases} \tag{28}$$

This adaptive threshold mechanism encourages broader exploration during periods of high uncertainty or reward stagnation, while enabling more focused behavior when the environment becomes predictable.

Table 5: Key hyperparameters of the Cognitive Predictive Processing framework.

Component	Parameter	Value
Cognitive Controller	Exploration Phase Threshold ($\tau^{explore}$)	0.4
	Approach Phase Threshold ($\tau^{approach}$)	0.7
	Exploration Weight ($w^{explore}_{\text{explore}}$)	0.8
	Exploration Weight ($w^{explore}_{\text{approach}}$)	0.5
	Exploration Weight ($w^{explore}_{\text{complete}}$)	0.2
	Stagnation Detection Window (m)	50
	Base Jump Threshold	0.6
	Jump Threshold Explore Phase ($\theta^{jump}_{base}(\phi^{explore})$)	0.5
	Jump Threshold Approach Phase ($\theta^{jump}_{base}(\phi^{approach})$)	0.6
	Jump Threshold Complete phase ($\theta^{jump}_{base}(\phi^{complete})$)	0.7
Dual-Memory System	Working Memory Capacity (K)	20
	Surprise Threshold Base (τ_{base})	0.3
Prediction Regulator	Observation Error Weight (ω_o)	0.5
	Reward Error Weight (ω_r)	0.3
	Transition Error Weight (ω_{trend})	0.2

B Detailed Experimental Setup

B.1 Environmental Configuration Details

We configured the MineDojo environment with standardized parameters across all evaluation tasks to ensure fair comparison between methods. All environments shared common settings including appropriate break speed multipliers (100), consistent reward structures (1 point upon target acquisition), and task-specific prompts for affordance map generation. Tasks were initialized with relevant starting conditions as described in Table 1, with identical environmental parameters maintained across all compared methods. For each task, we utilized dedicated fine-tuned UNet models to generate task-appropriate affordance maps, with fast reset disabled for consistency. All experimental results were averaged across three independent runs with same random seeds to ensure statistical reliability.

B.2 Baseline Implementation Details

Our evaluation methodology utilized the LS-Imagine codebase as the implementation foundation for our CPP framework. For a rigorous comparative evaluation, we directly incorporated the reported performance metrics for DreamerV3, VPT, and STEVE-1 from the LS-Imagine paper [28], rather than re-implementing these methods. To ensure fair comparison, we implemented CPP using identical training procedures, hyperparameters, and model architecture components as described in the LS-Imagine repository, maintaining consistency in imagination horizon (15 steps), network dimensions (encoder: cnn-depth 96, mlp-units 1024; decoder: same configuration), optimizer settings, and training duration (1×10^6 steps). This approach allowed us to isolate the performance improvements attributable to our cognitive components while maintaining methodological consistency with established benchmarks, and all experiments were conducted on NVIDIA L40s GPUs.

B.3 Hyperparameter Configuration

Our CPP implementation integrates the cognitive components with carefully selected parameters detailed in Table 5. These parameter selections were informed by cognitive science principles that model human problem-solving strategies. For other world model parameters, we used the same configuration as the LS-Imagine baseline to ensure fair comparison across all evaluated methods, including network dimensions, optimizer settings, and training duration.

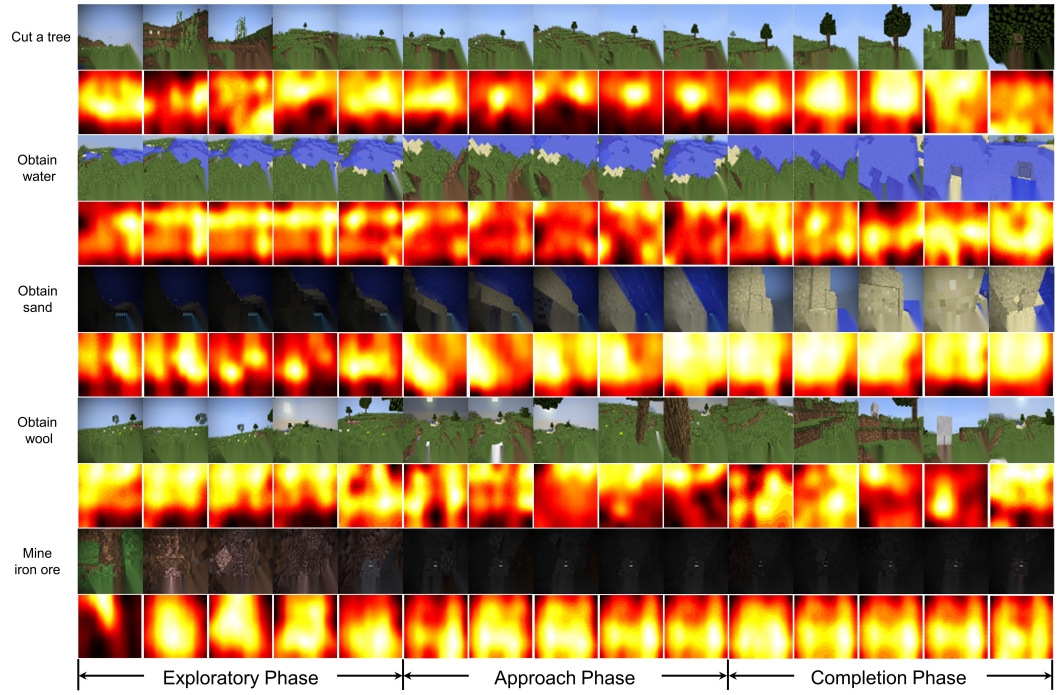


Figure 3: **CPP agent behavior across five MineDojo tasks.** Each task progression shows three cognitive phases: exploration (left), approach (middle), and completion (right). Top rows show agent observations while middle rows display affordance maps.

C Extended Performance Analysis

C.1 Trajectory Analysis with Affordance Maps

Figure 3 showcases CPP's adaptive exploration through affordance maps that evolve from diffuse patterns during exploration to concentrated activation during completion. This progression directly correlates with the decreasing exploration weight parameters across cognitive phases. The obtain wool task demonstrates particularly clear transitional patterns. Importantly, when target objects are initially occluded (such as sheep obscured by forests), the stagnation detection mechanism (Equations 17-18) identifies lack of progress and maintains the exploration phase with diffuse affordance patterns until visual detection occurs, at which point the maps transition to concentrated activation. These visualizations provide evidence that CPP's cognitive mechanisms enhance exploration coherence, supporting the quantitative performance improvements observed.

C.2 Completion Trajectory Analysis

Figure 4 illustrates CPP failure modes while Figure 5 presents LS-Imagine failure cases. In the harvest log task (top row), the agent becomes trapped in terrain cavities (highlighted in red), rendering affordance mapping ineffective. The obtain sand task (middle row) failures occur when environmental generation creates water bodies without adjacent sand blocks. In the obtain wool task (bottom row), the agent fails to identify a black sheep (highlighted in red) despite direct visibility, with affordance map activation indicating perceptual models trained on white sheep failed to generalize to variant appearances. LS-Imagine exhibits different failure patterns, predominantly exploration timeouts across all tasks. Most notably in the water collection task (red highlight), LS-Imagine identifies water sources but exhibits behavioral stagnation, repeatedly executing similar actions without progress. This highlights a critical advantage of CPP's stagnation detection mechanism, which dynamically resets exploration parameters when detecting repetitive behaviors. The comparative analysis reveals that while CPP enhances exploration efficiency, the failure modes demonstrate remarkable parallels

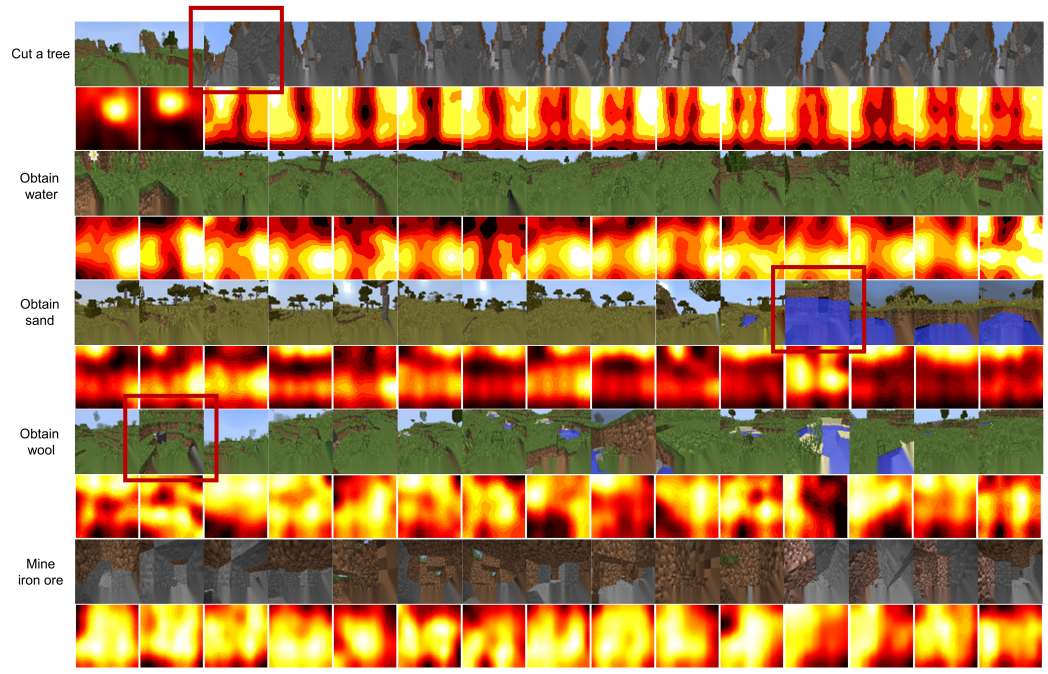


Figure 4: **CPP failure modes across tasks.** Red highlights indicate critical failure points: terrain cavity in the harvest log task preventing further exploration; water without adjacent sand in the obtain sand task challenging environmental expectations; and unrecognized black sheep in the obtain wool task revealing perceptual limitations after training primarily with white sheep.

to human cognitive processing errors. Similar to how human and Transformer processing are affected by equivalent stimulus properties [20], CPP's inability to recognize black sheep or becoming trapped in terrain cavities mirrors human perceptual biases and environmental fixation.

C.3 Reward Pattern Analysis

Figure 6 presents intrinsic reward patterns across our experimental tasks, revealing three key advantages of CPP. First, CPP achieves initial high-value rewards more rapidly (within first 50 steps), establishing effective exploration trajectories earlier. Second, during middle phases (steps 400-700) when both methods encounter "reward deserts," CPP maintains more frequent non-zero signals, demonstrating the effectiveness of our dual-memory system in guiding exploration during limited feedback. Finally, in later phases (steps 700-1000), CPP exhibits more continuous, goal-oriented reward patterns with concentrated high-values between steps 800-900, while LS-Imagine displays more intermittent peaks. This confirms how our cognitive phase transition effectively shifts from exploration to completion phases. The MineCLIP score comparison (Figure 6 f) quantitatively validates these observations, showing CPP's superior performance with statistically significant improvements ($p < 0.05$) in the water collection and sheep shearing tasks, where non-overlapping error bars indicate reliable performance advantages.

D Hyperparameter Sensitivity Analysis for Cognitive Predictive Processing

We conducted sensitivity analysis on two critical parameters of our CPP framework to validate design choices and assess robustness. For exploration weight parameters (Figure 7 a), we tested five configurations controlling phase-progression patterns. Our default implementation (B: 0.8→0.5→0.2) achieves optimal performance with 90.0% success rate and 340 completion steps. Configurations with flat parameters (D) or reverse progression (E) show significantly degraded performance, confirming that phase-based adaptation is essential for effective task completion.

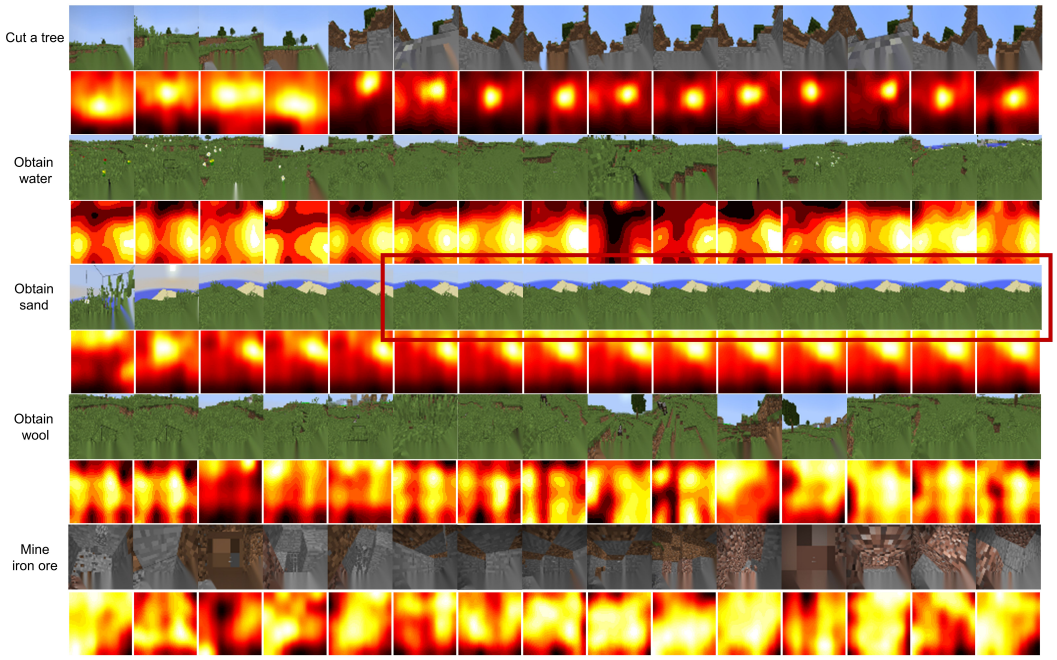


Figure 5: **LS-Imagine failure modes across tasks.** Red highlight in sand collection task shows behavioral stagnation where the agent fails to progress despite locating sand.

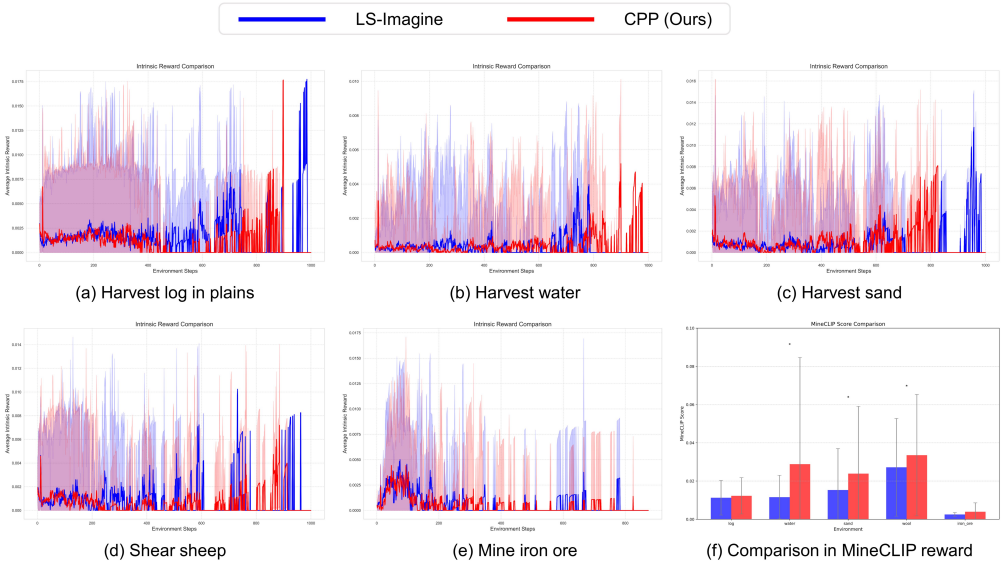
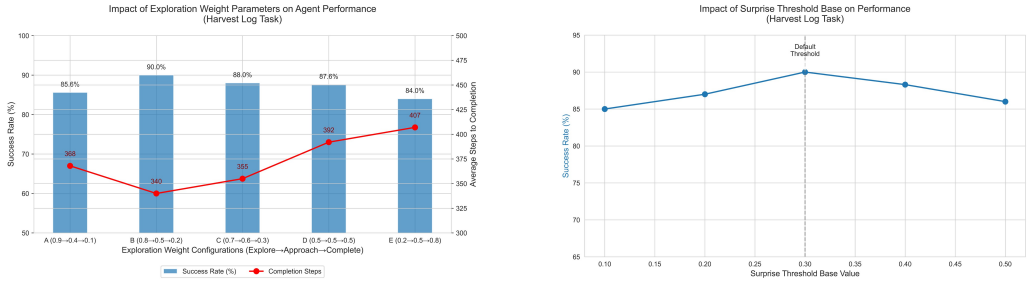


Figure 6: **Reward pattern comparison between CPP and LS-Imagine.** CPP (red) shows earlier high-value rewards, more consistent signals during exploration phases, and more concentrated reward patterns in later phases compared to LS-Imagine (blue).



(a) (b)

Figure 7: **Hyperparameter sensitivity analysis.** (a) Exploration weight parameters across five configurations, showing success rates and completion steps. Configuration B represents our default implementation. (b) Surprise threshold base impact on success rate, with vertical line indicating default value (0.3). All analyses conducted on the harvest log in plains task.

The surprise threshold base parameter governs episodic memory selectivity and exhibits an inverted U-shaped relationship with performance (Figure 7 b). The default value of 0.3 achieves peak success rate, while values that are too low (0.1-0.2) or too high (0.5-0.7) both reduce performance through different mechanisms. These results validate our parameter choices and demonstrate that CPP's cognitive mechanisms operate optimally within specific ranges that align with neurologically-inspired principles.

E Algorithm Pseudocode

This appendix provides the complete pseudocode for the Cognitive Predictive Processing framework for open-world reinforcement learning.

Algorithm 1 Cognitive Predictive Processing Framework

Initialize world model parameters θ, policy parameters ψ, working memory $\mathcal{W}_0 = \emptyset$, episodic memory $\mathcal{E}_0 = \emptyset$, cognitive phase $\phi_0 = \phi^{explore}$, precision weights $\omega = (\omega_o, \omega_r, \omega_{trend})$.
Initialize replay buffer $\mathcal{B}$ with random experience.
while not converged **do**
  Sample transitions from $\mathcal{B}$.
  Update world model parameters θ.
  for episode **do**
    Receive initial observation o_0.
    Initialize states h_0, z_0.
    for time step $t = 0 \cdots T$ until termination **do**
      Compute Progress$_t$ using Eq. (15).
      Update cognitive phase ϕ_t according to Eq. (16).
      Check for stagnation via Eq. (17) and reset phase if needed via Eq. (18).
      Set exploration parameters $\eta(\phi_t) = (w^{explore}(\phi_t), b^{jump}(\phi_t))$ from Eq. (2).
      Update working memory $\mathcal{W}_t = \{(o_{t-k}, a_{t-k}, r_{t-k})\}_{k=1}^{K}$ per Eq. (5).
      Compute surprise value s_t using Eq. (19).
      Adapt surprise threshold τ_t using Eq. (20).
      if $s_t > \tau_t$ **then**
        Update episodic memory $\mathcal{E}_t = \mathcal{E}_{t-1} \cup \{(o_t, a_t, r_t, s_t)\}$ via Eq. (7).
      end if
      Retrieve relevant memories $\mathcal{R}_t$ using similarity function in Eq. (21), Eq. (22).
      Generate predictions $\hat{o}_{t+1}, \hat{r}_{t+1}$ using Eq. (8).
      Compute prediction errors $\varepsilon_t^o, \varepsilon_t^r, \varepsilon_{trend}^r$ via Eq. (23).
      Calculate uncertainty $\mathcal{U}_t = \omega_o \cdot \varepsilon_t^o + \omega_r \cdot \varepsilon_t^r + \omega_{trend} \cdot \varepsilon_{trend}^r$ from Eq. (24).
      Adjust jump threshold θ_t^{jump} using Eq. (25, 26, 27, 28).
      Determine jump flag $j_t = \mathbb{I}[\max(a_t) > \theta_t^{jump}(\phi_t, \mathcal{U}_t)]$ via Eq. (12).
      Select action $a_t \sim \pi_\psi(a_t | h_t, z_t, \mathcal{R}_t, \phi_t)$ using Eq. (14).
      Execute action a_t, observe o_{t+1}, r_{t+1}.
      Update cognitive state $\psi_t = (m_t, \phi_t)$ where $m_t = (\mathcal{W}_t, \mathcal{E}_t)$ per Eq. (4).
      Store transition $(o_t, a_t, r_t, \mathcal{M}_t, \psi_t, j_t, \Delta_t, G_t)$ in $\mathcal{B}$.
      if $j_t = 1$ **(Jumpy transition detected) then**
        Construct long-term data with jumpy transition information.
      end if
    end for
    Sample transitions batch from $\mathcal{B}$.
    Update world model by conditioning on cognitive phase ϕ_t via Eq. (11).
    Compute phase-specific value function $V_t = v_\psi(h_t, z_t, \mathcal{R}_t, \phi_t)$ via Eq. (14).
    Update policy with phase-specific entropy regularization.
  end for
end while