arXiv:2601.13247v2 [cs.CL] 28 Aug 2026

Aligning Agentic World Models via Knowledgeable Experience Learning

Baochang Ren[♠], **Yunzhi Yao**[♠], **Rui Sun**[♢], **Shuofei Qiao**[♠],
Ningyu Zhang[♠*], **Huajun Chen**[♠*]
[♠] Zhejiang University [♢] University of California, Los Angeles
{baochang.ren, zhangningyu}@zju.edu.cn

https://zjunlp.github.io/project/WorldMind/

Abstract

Current Large Language Models (LLMs) exhibit a critical modal disconnect: they possess vast semantic knowledge but lack the procedural grounding to respect immutable environmental constraints. Consequently, while these agents implicitly function as world models, their simulations often suffer from *physical hallucinations*—generating plans that are logically sound but physically unexecutable. Existing alignment strategies predominantly rely on resource-intensive training or fine-tuning, which attempt to compress dynamic environmental rules into static model parameters. However, such parametric encapsulation is inherently rigid, struggling to adapt to the open-ended variability of environmental dynamics without continuous, costly retraining. To solve this fundamental misalignment between semantic planning and environmental reality without relying on parameter updates, we introduce **WorldMind**, a framework that autonomously constructs a symbolic World Knowledge Repository by synthesizing environmental feedback. Specifically, it unifies Process Experience to enforce semantic physical feasibility via prediction errors and Goal Experience to guide task optimality through successful trajectories. Experiments on EB-ALFRED and EB-Habitat demonstrate that WorldMind achieves the best performance compared to baselines, exhibiting remarkable cross-model and cross-environment transferability.

1 Introduction

Building World Models capable of simulating and predicting environmental dynamics is foundational to the pursuit of embodied intelligence. While traditional approaches rely on external simulators, a compelling paradigm has emerged where the agent itself functions as the world model (Richens et al., 2025). By internally encoding environmental dynamics, agents can reason about causal relationships and predict outcomes without heavy-weight external engines (Zhang et al., 2021; Fang et al., 2025b; Chae et al., 2024; Wang et al., 2025a; Yang et al., 2025c). Conceptually, this redefines the agent not merely as a passive policy executor, but as an active prediction machine of the world it inhabits (Zhang et al., 2025c; Zheng et al., 2025; Cen et al., 2025; Wang et al., 2025b).

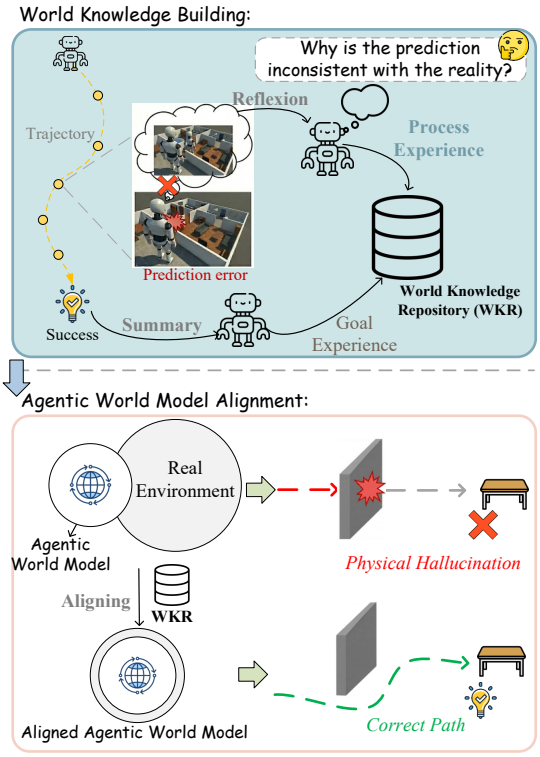


Figure 1: Conceptual Illustration of Experiential Alignment. The agent aligns its internal world model via Process Experience and Goal Experience.

However, a critical modal disconnect persists: while Large Language Models (LLMs) scale effectively on semantic reasoning, they exhibit a fundamental disparity between high-level logical planning and semantic physical constraints. Despite possessing vast declarative knowledge, they often

*Corresponding author.

lack the procedural understanding required to respect the immutable laws of the physical world. This misalignment leads to frequent physical hallucinations, where agents generate plans that are semantically coherent but physically unexecutable (e.g., attempting to slice an object without holding a knife). The agent understands the high-level *why*, but fails to grasp the procedural *how* imposed by physical reality.

To bridge this gap, dominant paradigms currently rely on internalizing physical laws directly into model parameters via Supervised Fine-Tuning (SFT) or Reinforcement Learning (Chae et al., 2024; Chen et al., 2025b). We suggest that this approach may face inherent challenges, as it attempts to capture the variability of dynamic physical environments using static parameter weights. Crucially, planning with an agentic world model presents a dual challenge: the internal simulation must not only identify the optimal path (Goal) but also strictly adhere to physical constraints (Process). Existing agents often prioritize goal achievement while neglecting physical feasibility, leading to brittle plans that fail in dynamic settings. Since static parameters cannot exhaustively cover all physical corner cases, an agent must possess the capacity for **online correction**, dynamically adjusting its understanding of constraints as it encounters them. This context-based flexibility (Zhang et al., 2025d; Zhou et al., 2025a; Ouyang et al., 2025) raises a research question: ***Can agents autonomously align their world knowledge via experience learning in a training-free manner?***

We answer this question by drawing inspiration from the cognitive theory of *Predictive Coding* (Huang and Rao, 2011; Friston, 2018). In this view, intelligence is not the passive absorption of data, but the continuous minimization of "prediction error"—the discrepancy between internal expectation and sensory reality. Classic model-based reinforcement learning refines transition models from off-policy state–action pairs whose predicted next states diverge from observed outcomes. We transplant this loop to training-free LLM agents by treating prediction-error episodes as off-policy evidence for a symbolic transition model, where a gradient update is replaced with a verbal causal-rule update. Instead of discarding execution failures as noise, we leverage them as corrective signals. When an agent's prediction contradicts reality, it exposes a boundary in the agent's internal world model. We argue that LLM agents possess useful but miscalibrated predictive capabilities that can be aligned through these error signals during inference.

As shown in Figure 2, we propose **WorldMind**, a framework transforming autonomous agents into empirical world model learners. To ensure robust simulations, our framework constructs a World Knowledge Repository comprising two distinct types of experience. **Process Experience** leverages prediction errors to enforce physical feasibility, while **Goal Experience** distills successful trajectories into procedural heuristics to guide efficient task completion. Extensive experiments on EB-ALFRED and EB-Habitat demonstrate that WorldMind improves over representative training-free baselines by reducing physically invalid actions and improving intermediate subgoal completion. Controlled memory ablations further show that the gain is not simply due to adding a cross-task repository, and supplementary stress tests examine robustness under object renaming, memory corruption, and shuffled task order. Cross-model and cross-environment results indicate that the constructed world knowledge transfers across model backbones and between household simulators, while we limit this claim to environments with overlapping semantics and physical assumptions.

2 The WorldMind Framework

2.1 Overview and Problem Formulation

As shown in Figure 2, we propose **WorldMind**, a framework transforming autonomous agents into empirical world model learners. By externalizing environmental dynamics into explicit memory, we enable grounded planning through a cycle of goal alignment and continuous reality verification.

We formulate the problem as a **World Knowledge-Augmented Markov Decision Process (WK-MDP)**, defined by the tuple:

$$\mathcal{M}_{\text{WK}} = \langle \rho_0, \mathcal{S}, \mathcal{A}, \mathcal{P}, \Omega, \mathcal{G}, \mathcal{W} \rangle \tag{1}$$

Here, $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. ρ_0 represents the initial state distribution from which the environment is reset at the beginning of each task episode. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ represents the latent transition dynamics. The observation function Ω yields egocentric visual inputs o_t based on the current state, and $\mathcal{G}$ represents the set of task objectives specified in natural language.

Distinct from standard MDPs, we introduce an explicit **World Knowledge Repository** $\mathcal{W} =$

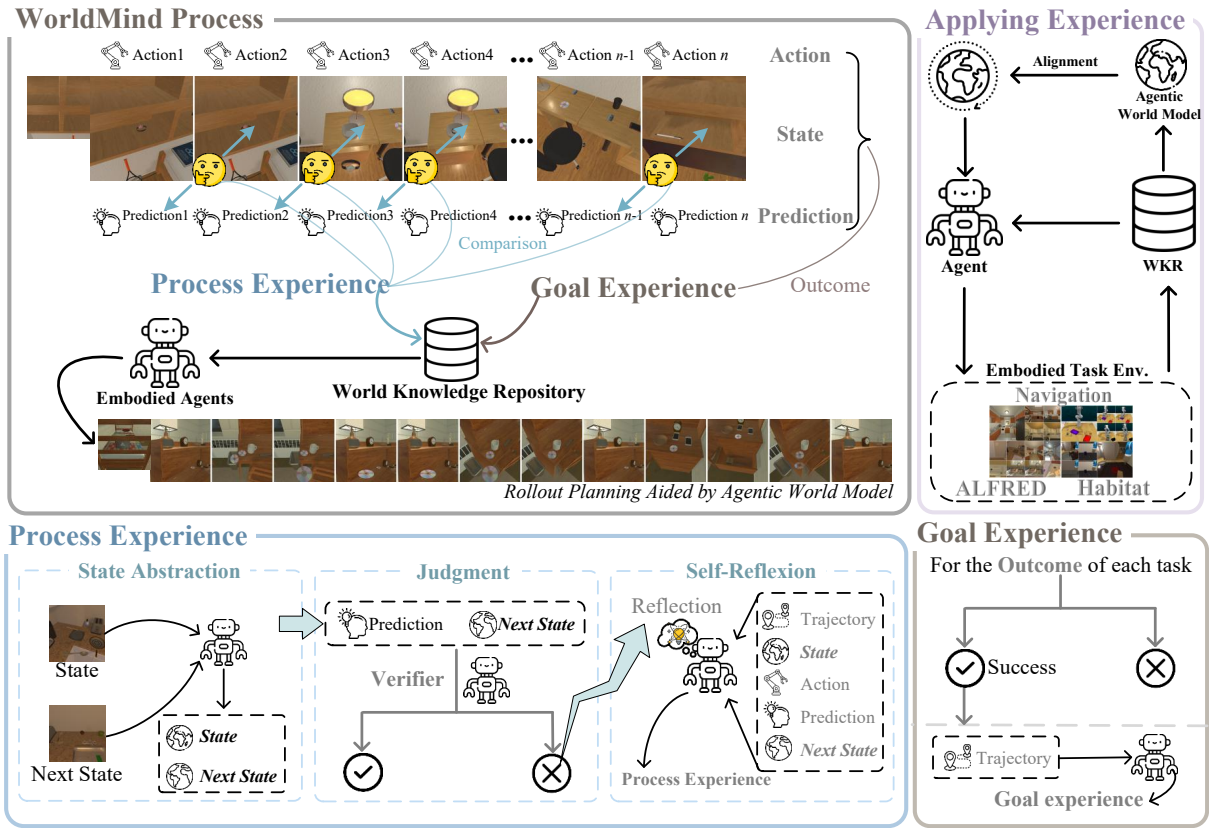


Figure 2: **Overview of the WorldMind Framework.** The agent autonomously constructs a World Knowledge Repository (WKR) by unifying Process Experience (from prediction errors) and Goal Experience (from successful trajectories) to guide grounded simulation.

$\{\mathcal{W}_p, \mathcal{W}_g\}$. $\mathcal{W}_p$ encapsulates Process Experience, a collection of verbalized causal rules derived from prediction errors. This component functions as a learned surrogate model $\hat{\mathcal{P}}_p$ that progressively approximates the ground-truth dynamics $\mathcal{P}$, ensuring simulated steps adhere to physical constraints. $\mathcal{W}_g$ encodes Goal Experience, a set of procedural heuristics distilled from successful trajectories. These priors serve to constrain the policy search space towards the optimal policy π^*, ensuring the simulated path converges on the correct target.

The agent operates under a policy $\pi(a_t, \hat{s}_{t+1} \mid o_t, g, \mathcal{W})$ that jointly generates an action a_t and a predicted future state $\hat{s}_{t+1}$ given a specific goal $g \in \mathcal{G}$. The learning objective is formulated as:

$$\max_{\pi} \mathbb{P}(\text{Success} \mid g) \text{ s.t. } \min \mathcal{D}(\hat{s}_{t+1} || s_{t+1}) \quad (2)$$

Here, Success represents the evaluation metric directly provided by the simulation environment's underlying script, which automatically verifies whether the physical states of relevant objects satisfy the predefined conditions of the goal g. This metric accounts for both final task completion and partial progress (such as completed subgoals). This objective implies that maximizing the probability of task success is contingent upon minimizing the divergence $\mathcal{D}$ between the predicted state $\hat{s}$ and the actual state s.

2.2 Building World Knowledge Repository

The core of WorldMind lies in the autonomous development of the World Knowledge Repository (WKR) $\mathcal{W}$. This process involves two distinct update mechanisms: one for establishing physical boundaries termed Process Experience, and one for distilling procedural shortcuts known as Goal Experience.

Process Experience Construction. To align internal simulations with environmental reality, $\mathcal{M}$ operates in a `Predict-Act-Verify` loop. At each step t, conditioned on the environmental prior p_{env}, current observation o_t, task goal g, and retrieved knowledge $\mathcal{W}$, the agent jointly generates an action and a textual prediction:

$$(a_t, \hat{s}_{t+1}) \sim \mathcal{M}(o_t, p_{env}, g, \mathcal{W}) \quad (3)$$

After execution, the environment yields the actual grounded state s_{t+1}. We then operationalize the

acquisition of Process Experience through three distinct steps: State Abstraction, Judgment, and Self-Reflexion.

First, during **State Abstraction**, to prioritize abstract causal variables over low-level details, we explicitly define a semantic abstraction process to convert the ground-truth states into high-level descriptions:

$$\bar{s}_k = \mathcal{M}_{\text{abs}}(s_k), \quad k \in \{t, t+1\} \tag{4}$$

Second, in the **Judgment** phase, the agent acts as a verifier, comparing its internal prediction $\hat{s}_{t+1}$ against the actual abstract outcome $\bar{s}_{t+1}$. Specifically, we prompt a Large Language Model to act as a Logical Consistency Validator. This validator conservatively evaluates whether there is a direct, factual contradiction between the predicted and actual states, deliberately ignoring irrelevant descriptions or unexecuted intents (detailed prompts are provided in Appendix H). A learning signal is triggered only when such an explicit semantic discrepancy is detected, indicating a physical hallucination. Finally, through **Self-Reflexion**, the agent addresses the identified conflict by triggering a reflective module $\mathcal{R}$. This module operates by prompting a Large Language Model to act as a Root Cause Analyst, analyzing the error context alongside the interaction history τ_{t-1} to synthesize a corrective causal rule (detailed reflection prompts are provided in Appendix H). The update rule for Process Experience is formulated as:

$$\mathcal{W}_p \leftarrow \mathcal{W}_p \cup \{\mathcal{R}(\tau_{t-1}, a_t, \bar{s}_t, \hat{s}_{t+1}, \bar{s}_{t+1})\} \tag{5}$$

This three-step mechanism is designed to capture causal dynamics while filtering trivial environmental noise. For example, if the agent predicts that it can place an object while still far from the intended receptacle but the environment reports an invalid placement, the resulting Process Experience records the precondition that the agent must first navigate to the destination receptacle before executing `put` actions. Appendix F provides pseudocode and concrete Process/Goal Experience examples.

Goal Experience Construction. Efficient task completion necessitates heuristic guidance, complementing the physical feasibility ensured by Process Experience. We address this by distilling procedural heuristics from successful episodes. Given a successful trajectory τ^*, the agent $\mathcal{M}$ acts as a task analysis expert to autonomously analyze the interaction history. Specifically, the agent is prompted to evaluate the step-by-step task progress, identifying key actions that advanced the goal to extract a single, abstract, and reusable workflow while filtering out context-specific noise (detailed extraction prompts are provided in Appendix H). The Goal Experience is updated as follows:

$$\mathcal{W}_g \leftarrow \mathcal{W}_g \cup \mathcal{M}(\tau^*) \tag{6}$$

These distilled priors serve as optimized meta-instructions, limiting the agent's policy search space towards the optimal subspace in future tasks.

2.3 Inference via Constrained Simulation

In the inference phase, the agent $\mathcal{M}$ leverages the constructed World Knowledge Repository $\mathcal{W}$ to guide decision-making through constrained simulation. Specifically, the system dynamically retrieves the top-2 relevant world knowledge from the repository through semantic similarity[1] to the current task goal g. Conditioned on this augmented context, the agent jointly generates the action a_t and the predicted future state $\hat{s}_{t+1}$. Crucially, to mitigate hallucinations, the generation of $\hat{s}_{t+1}$ is gated via prompt-based heuristic rules: the agent is explicitly instructed to only simulate outcomes when the target object is explicitly grounded in the current observation o_t or the repository $\mathcal{W}$, otherwise executing the action and outputting a predefined skip string without updating the internal world model. This selective strategy significantly enhances inference efficiency. Unlike approaches relying on computationally intensive multiple sampling or heavy external world models, our framework minimizes latency by bypassing redundant simulations for ungrounded states.

3 Experiment

3.1 Experimental Settings.

Datasets and Metrics. We evaluate our framework on the EB-ALFRED and EB-Habitat benchmarks from EmbodiedBench (Yang et al., 2025a), focusing on five fine-grained subsets: Base, Common Sense, Complex Instruction, Spatial Awareness, and Visual Appearance. To quantify performance, we employ two complementary metrics: *Success Rate (SR)* and *Goal-Conditioned Success (GC)*. SR is a strict binary metric (0 or 1) indicating whether the final goal is fully achieved. In

[1]We utilize the `SentenceTransformer` library to compute text embeddings and cosine similarity for retrieval.

Backbone	Method	Memory Scope	Success Rate (SR)						Goal Condition (GC)					
			Avg	Base	Common	Complex	Visual	Spatial	Avg	Base	Common	Complex	Visual	Spatial
Open-source and Proprietary Models														
GPT-4o	ReAct	Episodic	56.8	64	54	68	46	52	65.6	74.0	60.3	74.0	58.3	**61.3**
GPT-4o-mini	ReAct	Episodic	28.8	34	28	36	24	22	37.8	47.8	35.3	43.5	33.3	29.0
Claude-3.7-Sonnet	ReAct	Episodic	**67.2**	68	**68**	70	**68**	**62**	67.5	72.0	66.0	**76.7**	**63.0**	59.7
Gemini-1.5-Pro	ReAct	Episodic	63.2	**70**	64	**72**	58	52	**67.9**	**74.3**	**66.7**	76.5	62.8	59.0
Llama-3.2-90B-Vis	ReAct	Episodic	35.2	38	34	44	28	32	40.3	43.7	37.3	49.2	35.3	36.0
InternVL2.5-78B	ReAct	Episodic	37.0	41	40	39	16	49	39.4	42.3	35.3	43.3	35.7	40.3
Method Comparison														
GPT-3.5-turbo	ReAct	Episodic	44.4	52	**48**	52	32	38	50.4	55.3	**53.5**	55.3	42.7	45.0
	BoN	Episodic	42.8	46	42	50	**42**	34	50.4	54.2	46.5	56.5	**52.0**	42.8
	SimuRA	Episodic	45.2	50	42	54	38	42	53.6	57.8	47.8	59.7	48.5	**54.3**
	ReasoningBank	Global	41.6	50	36	44	36	42	47.6	57.5	41.5	47.0	44.2	48.0
	Synapse	Global	38.8	38	46	40	36	34	43.6	42.5	51.3	42.7	42.0	39.7
	AWM	Global	40.0	46	32	48	40	34	46.2	53.2	39.2	50.7	47.0	41.0
	WorldMind	Global	**48.0**	**58**	**48**	**56**	34	**44**	**54.1**	**63.0**	52.7	**61.0**	41.7	52.0
GPT-4.1-mini	ReAct	Episodic	41.2	50	40	46	38	32	47.5	55.3	42.8	52.2	47.2	39.8
	BoN	Episodic	44.4	46	44	50	**42**	40	49.5	50.8	48.3	54.7	**48.7**	45.0
	SimuRA	Episodic	45.6	**52**	44	**54**	38	40	52.2	**61.0**	50.3	58.2	45.3	46.3
	ReasoningBank	Global	38.0	42	36	42	34	36	42.6	46.7	38.8	45.8	41.5	40.3
	Synapse	Global	37.2	40	32	44	36	34	42.2	41.2	37.5	49.5	41.3	41.7
	AWM	Global	41.2	44	36	48	38	40	46.0	48.3	42.0	52.5	44.3	42.7
	WorldMind	Global	**49.2**	50	**58**	**54**	**42**	**42**	**55.7**	**61.0**	**61.0**	**58.8**	48.0	**49.7**

Table 1: **Main Results on the EB-ALFRED Benchmark.** The top block reports ReAct reference performance across backbones, while the lower block compares training-free methods under the same benchmark protocol. The Memory Scope column distinguishes Episodic methods without a cross-task global repository from Global methods with a cross-task global memory repository. For GPT-3.5-turbo experiments, GPT-4.1-mini handles visual perception, while GPT-3.5-turbo performs subsequent planning and WorldMind reasoning.

contrast, GC evaluates process adherence by awarding partial credit for completed subgoals, even if the episode ultimately fails.

Models and Baselines. We employ GPT-4.1-mini and GPT-3.5-turbo as the primary backbones for our WorldMind framework. To evaluate effectiveness, using reported results of open- and closed-source models from EmbodiedBench, we compare against representative baselines, including Best-of-N (BoN), ReAct (Yao et al., 2022), Synapse (Zheng et al., 2023), SimuRA (Deng et al., 2025), ReasoningBank (Ouyang et al., 2025), and AWM (Wang et al., 2024f). All methods are evaluated under the same EmbodiedBench observation interface, action space, episode split, and SR/GC metrics; the compared variable is the planning, memory, or verification mechanism introduced by each method. To make this comparison explicit, Tables 1 and 2 include a Memory Scope column that distinguishes *Episodic* methods, which do not maintain a cross-task global repository, from *Global* methods. Appendix G gives concise baseline descriptions.

3.2 Main Results

WorldMind consistently outperforms baselines in strict task completion. As shown in Table 1 and Table 2, WorldMind achieves the highest SR among evaluated methods across both benchmarks. On EB-Habitat with the GPT-4.1-mini backbone, it attains an SR of 50.8%, outperforming ReAct by 9.2%, while on EB-ALFRED with GPT-3.5-turbo, SR improves from 44.4% to 48.0%. Given that SR strictly measures complete task success, these gains demonstrate WorldMind's enhanced ability to overcome physical dead-ends and reach final goal states more consistently in practice.

WorldMind achieves improved procedural correctness. As shown in Table 1 and Table 2, beyond final outcomes, WorldMind outperforms baselines on the GC metric, which evaluates the correct execution of intermediate steps. On EB-ALFRED with GPT-3.5-turbo, the average GC improves from 50.4% to 54.1%. Simultaneously, on EB-Habitat with the GPT-4.1-mini backbone, the GC reaches 57.2%, surpassing the ReAct baseline (47.4%) by a large margin. These gains indicate that even when full task completion fails, WorldMind executes a higher proportion of valid subgoals, demonstrating that the distilled Goal Experience effectively guides sequential task progression.

Robustness across diverse capabilities and backbones. Our framework exhibits consistent stabil-

Backbone	Method	Memory Scope	Success Rate (SR)						Goal Condition (GC)					
			Avg	Base	Common	Complex	Visual	Spatial	Avg	Base	Common	Complex	Visual	Spatial
Open-source and Proprietary Models														
GPT-4o	ReAct	Episodic	58.0	86	44	56	**68**	36	70.4	90.7	56.0	68.0	**75.2**	**62.1**
GPT-4o-mini	ReAct	Episodic	36.4	74	22	32	22	32	48.6	77.5	32.5	42.0	33.1	57.8
Claude-3.7-Sonnet	ReAct	Episodic	**61.2**	90	**58**	**58**	62	**38**	**72.3**	**97.5**	**68.5**	**79.5**	72.0	43.8
Gemini-1.5-Pro	ReAct	Episodic	57.2	92	52	48	56	**38**	61.0	92.5	53.5	49.5	59.4	50.0
Llama-3.2-90B-Vis	ReAct	Episodic	45.6	**94**	24	50	32	28	55.9	94.5	32.5	53.0	39.7	59.6
InternVL2.5-78B	ReAct	Episodic	47.6	84	30	**58**	34	32	58.6	82.0	43.0	59.0	63.9	45.1
Method Comparison														
GPT-3.5-turbo	ReAct	Episodic	43.6	82	30	34	38	34	50.4	84.0	34.0	40.0	43.4	50.4
	BoN	Episodic	43.6	74	34	36	40	34	50.8	75.5	35.5	43.0	46.9	53.3
	SimuRA	Episodic	48.0	82	40	36	**44**	**38**	55.3	83.0	**47.0**	40.0	48.9	57.8
	ReasoningBank	Global	46.4	76	40	40	42	34	53.9	78.5	41.5	44.0	46.2	**59.5**
	Synapse	Global	48.4	84	40	38	42	**38**	54.2	85.0	43.0	43.0	46.3	53.5
	AWM	Global	46.4	78	**42**	**44**	36	32	50.9	78.0	43.0	48.0	39.7	45.7
	WorldMind	Global	**48.8**	**86**	**42**	**44**	40	32	**56.7**	**89.0**	**47.0**	**51.0**	**49.8**	46.8
GPT-4.1-mini	ReAct	Episodic	41.6	74	26	32	40	**36**	47.4	76.0	30.0	37.0	43.7	50.3
	BoN	Episodic	44.4	74	32	36	46	34	50.9	75.8	38.0	41.5	48.9	50.5
	SimuRA	Episodic	48.4	84	42	40	44	32	55.4	85.0	45.5	47.0	48.2	51.5
	ReasoningBank	Global	46.4	76	42	30	**52**	32	53.7	77.0	45.5	34.5	**58.4**	**53.3**
	Synapse	Global	45.6	78	42	38	34	**36**	52.0	78.3	46.5	45.5	39.7	50.0
	AWM	Global	43.6	74	32	38	44	30	50.8	74.8	37.0	44.5	47.9	49.7
	WorldMind	Global	**50.8**	**86**	**44**	**44**	46	34	**57.2**	**87.0**	**48.5**	**51.5**	52.3	46.7

Table 2: **Main Results on the EB-Habitat Benchmark.** The top block reports ReAct reference performance across backbones, while the lower block compares training-free methods under the same benchmark protocol. The Memory Scope column distinguishes Episodic methods without a cross-task global repository from Global methods with a cross-task global memory repository. Performance is evaluated on five subsets: Base, Common, Complex, Visual, and Spatial; best results within each model group are in **bold**.

ity across fine-grained dimensions, particularly in challenging subsets like Visual Appearance and Spatial Awareness. For instance, the EB-Habitat Base Success Rate reaches 86% for both GPT-3.5-turbo and GPT-4.1-mini, a model-agnostic benefit further validated across leading proprietary and open-source models (see Appendix D). This stability stems from the Predict-Act-Verify loop filtering ungrounded hallucinations across different model scales. Furthermore, the framework maintains a scalable, bounded repository footprint (see Appendix C) with negligible real-time retrieval latency (see Appendix B). In particular, constructing Process and Goal Experience incurs one-time token costs, while top-2 retrieval adds only 0.0028 seconds per step on average. We also validate generalization across action granularity on EB-Navigation (see Appendix A), additional model backbones (see Appendix D), controlled-memory effects in Section 4.4, and adversarial, task-order, and failure-taxonomy robustness (see Appendix E).

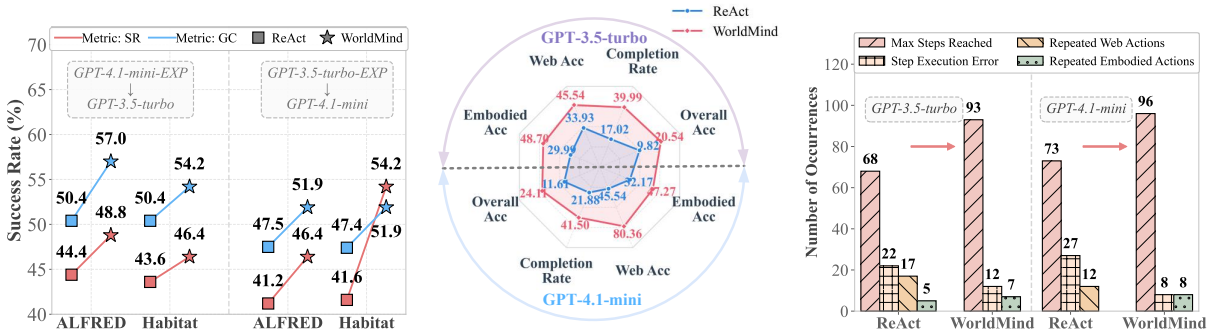
3.3 Ablation Study

Table 3 presents the ablation results isolating the specific impact of each component. We observe that Goal Experience primarily enhances the Goal Condition (GC) metric, particularly on EB-ALFRED with GPT-3.5-turbo. This indicates that high-level strategies effectively guide the agent through correct intermediate subgoals, preventing navigational drift. In contrast, Process Experience drives a larger improvement in the strict Success Rate (SR). By encoding semantic environmental constraints, this component acts as a safeguard against fatal execution errors, ensuring that planned actions are not just logically sound but also executable within the simulation. Across both benchmarks, the full model generally achieves the best overall performance, suggesting that **combining goal-based heuristics with semantic physical verification yields complementary benefits for effective alignment.**

4 Analysis

4.1 Cross-Model Experience Transfer Analysis

Externalized symbolic knowledge should be reusable beyond the model that collected it. To test this property, we exchange repositories between GPT-3.5-turbo and GPT-4.1-mini. As shown in Figure 3a, GPT-3.5-turbo using GPT-4.1-mini's repository improves on EB-ALFRED (SR:



(a) Performance of Cross-Model Experience Transfer (b) Accuracy Analysis on Embodied Web Agent Benchmark (c) Error Analysis on Embodied Web Agent Benchmark

Figure 3: **Experimental Results and Analysis.** (a) Comparison of experience transfer capabilities between the two models. (b) Performance comparison on the Embodied Web Agent task, reporting accuracy metrics for both GPT-3.5-turbo and GPT-4.1-mini. (c) Comparative error distribution analysis for both models in the same environment.

44.4%→48.8%, GC: 50.4%→57.0%), while GPT-4.1-mini using GPT-3.5-turbo's experience raises EB-Habitat SR from 41.6% to 54.2%. **These bidirectional gains indicate that WorldMind captures transferable environmental regularities rather than model-specific traces.**

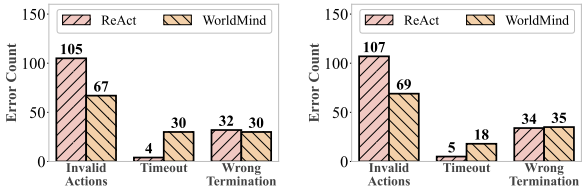
Backbone	Components		EB-ALFRED		EB-Habitat	
	Goal	Process	SR	GC	SR	GC
GPT-3.5-turbo	-	-	44.4	50.4	43.6	50.4
	✓	-	44.8	51.0	47.2	54.6
	-	✓	42.4	47.5	45.2	51.3
	✓	✓	**48.0**	**54.1**	**48.8**	**56.7**
GPT-4.1-mini	-	-	41.2	47.5	41.6	47.4
	✓	-	48.8	**56.3**	48.4	55.7
	-	✓	46.4	51.5	47.6	54.0
	✓	✓	**49.2**	55.7	**50.8**	**57.2**

Table 3: Ablation study on the effectiveness of Goal Guidance and Process Supervision. We report both SR and GC on EB-ALFRED and EB-Habitat.

4.2 Cross-Environment Analysis

To evaluate scalability, we further test on Embodied Web Agent (Hong et al., 2025), where agents switch between web information seeking and embodied procedural execution. We use 112 sampled Indoor Cooking tasks and report Embodied Accuracy, Web Accuracy, Overall Accuracy, and Completion Rate.

Figure 3b shows that Overall Accuracy more than doubles for GPT-3.5-turbo (9.82%→20.54%) and rises from 11.61% to 24.11% for GPT-4.1-mini. Figure 3c further shows fewer execution errors, indicating that WorldMind helps agents traverse web-to-embodied transitions without premature physical failure. **These results support transfer to hybrid environments with similar procedural constraints.**



(a) Habitat (GPT-3.5-turbo) (b) Habitat (GPT-4.1-mini)

Figure 4: **Error Analysis in EB-Habitat.** Failure distributions for ReAct and WorldMind.

4.3 Error Analysis

Figure 4 groups failures into *Invalid Actions*, *Timeout*, and *Wrong Termination*. In EB-Habitat, WorldMind reduces Invalid Actions for GPT-3.5-turbo from 105 to 67 (Figure 4a), while Timeouts rise from 4 to 30 because agents avoid immediate fatal errors and continue exploring. The same Invalid Action reduction appears for GPT-4.1-mini (Figure 4b). **This redistribution suggests that WorldMind converts some fatal execution failures into longer, recoverable trajectories.**

4.4 Controlled Memory Ablation

To test whether gains come merely from adding cross-task memory, we conduct a controlled ablation on EB-ALFRED Base, Common Sense, and Complex Instruction with GPT-3.5-turbo. All variants use the same evaluation harness, retrieval pipeline, prompt skeleton, and few-shot examples; only the enabled memory and verification

components differ. As shown in Table 4, NaiveMem controls for "just adding memory" by storing raw success trajectories with the same top-2 retriever but without Process Experience, the Logical-Consistency Validator, or LLM distillation. Full WorldMind exceeds NaiveMem by 22.9 GC on average, indicating that the architectural loop, rather than memory volume alone, accounts for the observed gain.

Variant	Process Exp.	Goal Exp.	Verifier	Memory Format	3-subset Mean GC
Baseline	–	–	–	none	45.0
NaiveMem (control)	–	raw	–	raw trajectory	36.0
WM w/o Verifier	✓	distilled	–	both	33.9
WM w/o Goal	✓	–	✓	rules only	46.1
WorldMind (Full)	✓	distilled	✓	both	**58.9**

Table 4: Controlled memory ablation on EB-ALFRED using the GPT-3.5-turbo backbone. The reported value is the mean GC across Base, Common Sense, and Complex Instruction. The Baseline row uses the WorldMind prompt framework with all memory and verification components disabled, isolating the effect of each added component under the same prompting setup. NaiveMem isolates the "just adding memory" confound by storing raw success trajectories without Process Experience, Verifier, or LLM distillation.

5 Related Work

Agentic AI. Agentic AI's core is a reflective, self-correcting loop, leveraging tools, memory, and constraints for open-world robustness and interpretability (Zhou et al., 2025b; Wang et al., 2024b; Xi et al., 2025; Sang et al., 2025; Guan et al., 2024a). The field advances via improved planning mechanisms like CoT, ToT, and self-refinement (Wei et al., 2022; Yao et al., 2023; Madaan et al., 2023; Shinn et al., 2023). To interact with the external world and achieve long-term goals, researchers have explored memory, tool-calling, embodied control, and the collaboration between agents and world models, etc (Wang et al., 2023; Schick et al., 2023; Zitkovich et al., 2023; Park et al., 2023; Patil et al., 2024; Yu et al., 2025c; Deng et al., 2025; Chen et al., 2025b; Mei et al., 2025; Fung et al., 2025; Tang et al., 2025). Specifically, while extensive related work (Zhou et al., 2025c; Zhang et al., 2025f; Yan et al., 2025; Zeng et al., 2024; Bo et al., 2025; Anokhin et al., 2024; Zhang et al., 2025g; Xu et al., 2025; Yang et al., 2025b; Huang et al., 2025; Zhang et al., 2025a; Pink et al., 2025; Cao et al., 2025; Hatalis et al., 2023; Salama et al., 2025) focuses on memory, significant research (Fang et al., 2025a; Tao et al., 2024; Gao et al., 2025; Zhang et al., 2025b; Jiang et al., 2024; Yu et al., 2025a; Wei et al., 2025; Liu et al., 2025; Cai et al., 2025a; Dou et al., 2025; Cai et al., 2025b,a; Dou et al., 2025) aims to realize self-evolving agents. Further work has expanded the ability to solve complex tasks by building multi-agent collaborative frameworks and autonomous systems (Wu et al., 2024; Li et al., 2023; Wang et al., 2024a; Fourney et al., 2024; Park et al., 2024). Long-term and procedural memories have been widely explored for improving agent reuse and adaptation, including trajectory memories such as Synapse (Zheng et al., 2023), workflow memory such as AWM (Wang et al., 2024f), reasoning memory such as ReasoningBank (Ouyang et al., 2025), and modular procedural memory such as LEGOMem (Han et al., 2025). WorldMind builds on this broader memory direction but differs in its alignment target: instead of only storing successful procedures or reasoning traces for policy reuse, it uses prediction–observation discrepancies to update a symbolic representation of environmental dynamics. Goal Experience is therefore not claimed as a standalone new memory type; its role is to complement Process Experience in a predictive-capability-strengthening loop for embodied world-model alignment.

World Models. World Models learn internal representations of environment dynamics for prediction and planning, a field spanning model-based RL, generative simulation, LLMs, and robotics (Kong et al., 2025; Ha and Schmidhuber, 2018; Ding et al., 2025; Guan et al., 2024b; Zhu et al., 2024). Originating in model-based RL (Kaiser et al., 2019; Deng et al., 2022; Ha et al., 2023), world models enable agents to imagine outcomes (Hafner et al., 2023; Wu et al., 2023). Generative world models not only predict physical environment changes by generating high-fidelity video (Brooks et al., 2024; Bruce et al., 2024; Liu et al., 2024; Bar-Tal et al., 2024; Chi et al., 2025; Yu et al., 2025b; Mendonca et al., 2023; Wang et al., 2024d; Bardes et al., 2023; Chen et al., 2025a; Parker-Holder et al., 2024), but also predict code execution results (Tang et al., 2024; Carbonneaux et al., 2025), and further extend their predictive capabilities to other diverse domains (Feng et al., 2025; Zhang et al., 2025e). World Models now underpin applications ranging from autonomous driving (Wang et al., 2024c; Jia et al., 2023; Hu et al., 2023; Wang et al., 2024e; Zhang et al., 2023) to embodied navigation (Zhong et al., 2025; Yao et al., 2025; Bar et al., 2025; Nie et al., 2025; Jin and Jia,

2025; Long et al., 2025; Liang et al., 2025; Shi et al., 2025), demonstrating their versatility across diverse tasks. This has also led to more benchmarks focused on evaluating the precise prediction of future states, physical laws, and environmental dynamics (Li et al., 2025a; Warrier et al., 2025; Duan et al., 2025; Yue et al., 2025). (Qiao et al., 2024) introduce World Knowledge Model (WKM), which encodes world knowledge into the planner. Unlike WKM, WorldMind performs training-free, online experiential alignment using execution failures. Concurrently, much research also posits that the agent itself is the world model (Tehenan et al., 2025; Zhou et al., 2024; Xiang et al., 2023; Fu et al., 2025; Li et al., 2025b; Qian et al., 2026).

6 Conclusion

WorldMind aligns agentic world models by externalizing prediction errors and successful trajectories into symbolic Process and Goal Experience. Across embodied benchmarks, it reduces physically invalid actions, improves task progress without gradient updates, and transfers useful experience across model backbones and household simulators with overlapping semantics, offering a training-free foundation for community reuse and extension. These results support symbolic experience learning as a practical route for adaptable embodied agents, while leaving broader transfer to contact-rich real-world robotics and substantially different physical regimes as future work.

7 Limitations

Despite careful design and empirical validation, this work has several limitations.

Dependence on Visual Perception Fidelity. WorldMind aligns reasoning and planning with environmental dynamics, but it remains contingent on the perceptual accuracy of the underlying Visual-Language Model (VLM). Process Experience can correct interaction-level prediction errors, but it cannot fully compensate for perceptual hallucinations such as object misclassification in cluttered scenes.

Scope of Symbolic Abstraction and Transfer. WorldMind abstracts raw environment states into textual rules, improving interpretability and reuse while discarding fine-grained geometry, contact dynamics, and low-level control signals. Our evidence should therefore be interpreted as transfer across household embodied simulators and related digital-physical tasks, not as proof of general physical-law transfer to arbitrary real-world robotics settings. Extending the framework to contact-rich manipulation requires tighter coupling with perception, control, and safety-specific constraints.

Safety Boundary. The experiments do not use manually pre-encoded safety Process Experience; all stored rules are generated from interaction feedback under benchmark action spaces. WorldMind should therefore not be treated as a standalone safety layer. Real-world deployment would require explicit safety constraints and external verification in addition to the experience-learning mechanism studied here.

Mechanistic Interpretability and Shared Repositories. Although the World Knowledge Repository empirically constrains agent behavior, the exact mechanism by which retrieved symbolic knowledge reshapes the agent's implicit policy distribution remains to be mapped. In addition, the current framework uses asynchronous experience sharing between individual agents; real-time synchronization, conflict resolution, and consensus building for multi-agent shared world models remain open problems.

Acknowledgements

We would like to express our sincere gratitude to the anonymous reviewers for their thoughtful and constructive feedback. This work was supported by the New Generation Artificial Intelligence-National Science and Technology Major Project (2025ZD0122802), National Natural Science Foundation of China (No. 62576307, No. NSFCU23B2055, No. NSFCU19B2027), the Fundamental Research Funds for the Central Universities (226-2023-00138), the Yongjiang Talent Introduction Programme (2021A-156-G), and the Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

References

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Andrey Kravchenko, Mikhail Burtsev, and Evgeny Burnaev. 2024. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. 2025. Navigation world models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15791–15801.

Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. 2024. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. 2023. V-jepa: Latent video prediction for visual representation learning.

Weihao Bo, Shan Zhang, Yanpeng Sun, Jingjing Wu, Qunyi Xie, Xiao Tan, Kunbin Chen, Wei He, Xiaofan Li, Na Zhao, et al. 2025. Agentic learner with grow-and-refine multimodal semantic memory. *arXiv preprint arXiv:2511.21678*.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. 2024. Video generation models as world simulators. *OpenAI Blog*, 1(8):1.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. 2024. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*.

Yuxuan Cai, Yipeng Hao, Jie Zhou, Hang Yan, Zhikai Lei, Rui Zhen, Zhenhua Han, Yutao Yang, Junsong Li, Qianjun Pan, et al. 2025a. Building self-evolving agents via experience-driven lifelong learning: A framework and benchmark. *arXiv preprint arXiv:2508.19005*.

Zhicheng Cai, Xinyuan Guo, Yu Pei, Jiangtao Feng, Jinsong Su, Jiangjie Chen, Ya-Qin Zhang, Wei-Ying Ma, Mingxuan Wang, and Hao Zhou. 2025b. Flex: Continuous agent evolution via forward learning from experience. *arXiv preprint arXiv:2511.06449*.

Zouying Cao, Jiaji Deng, Li Yu, Weikang Zhou, Zhaoyang Liu, Bolin Ding, and Hai Zhao. 2025. Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution. *arXiv preprint arXiv:2512.10696*.

Quentin Carbonneaux, Gal Cohen, Jonas Gehring, Jacob Kahn, Jannik Kossen, Felix Kreuk, Emily McMilin, Michel Meyer, Yuxiang Wei, David Zhang, et al. 2025. Cwm: An open-weights llm for research on code generation with world models. *arXiv preprint arXiv:2510.02387*.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. 2025. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*.

Hyungjoo Chae, Namyoung Kim, Kai Tzu-iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. 2024. Web agents with world models: Learning and leveraging environment dynamics in web navigation. *arXiv preprint arXiv:2410.13232*.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. 2025a. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*.

Shiqi Chen, Tongyao Zhu, Zian Wang, Jinghan Zhang, Kangrui Wang, Siyang Gao, Teng Xiao, Yee Whye Teh, Junxian He, and Manling Li. 2025b. Internalizing world models via self-play finetuning for agentic rl. *arXiv preprint arXiv:2510.15047*.

Xiaowei Chi, Peidong Jia, Chun-Kai Fan, Xiaozhu Ju, Weishi Mi, Kevin Zhang, Zhiyuan Qin, Wanxin Tian, Kuangzhi Ge, Hao Li, et al. 2025. Wow: Towards a world omniscient world model through embodied interaction. *arXiv preprint arXiv:2509.22642*.

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

Mingkai Deng, Jinyu Hou, Zhiting Hu, and Eric Xing. 2025. Simura: A world-model-driven simulative reasoning architecture for general goal-oriented agents. *arXiv preprint arXiv:2507.23773*.

Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, et al. 2025. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys*, 58(3):1–38.

Shihan Dou, Ming Zhang, Chenhao Huang, Jiayi Chen, Feng Chen, Shichun Liu, Yan Liu, Chenxiao Liu, Cheng Zhong, Zongzhang Zhang, et al. 2025. Evalearn: quantifying the learning capability and efficiency of llms via sequential problem solving. *arXiv preprint arXiv:2506.02672*.

Haoyi Duan, Hong-Xing Yu, Sirui Chen, Li Fei-Fei, and Jiajun Wu. 2025. Worldscore: A unified evaluation benchmark for world generation. *arXiv preprint arXiv:2504.00983*.

Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, et al. 2025a. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv preprint arXiv:2508.07407*.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025b. Webevolver: Enhancing web agent self-improvement with coevolving world model. *arXiv preprint arXiv:2504.21024*.

Jichen Feng, Yifan Zhang, Chenggong Zhang, Yifu Lu, Shilong Liu, and Mengdi Wang. 2025. Web world models. *arXiv preprint arXiv:2512.23676*.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, et al. 2024. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*.

Karl Friston. 2018. Does predictive coding have a future? *Nature neuroscience*, 21(8):1019–1021.

Dayuan Fu, Jianzhao Huang, Siyuan Lu, Guanting Dong, Yejie Wang, Keqing He, and Weiran Xu. 2025. Preact: Prediction enhances agent's planning ability. In *Proceedings of the 31st international conference on computational linguistics*, pages 1–16.

Pascale Fung, Yoram Bachrach, Asli Celikyilmaz, Kamalika Chaudhuri, Delong Chen, Willy Chung, Emmanuel Dupoux, Hongyu Gong, Hervé Jégou, Alessandro Lazaric, et al. 2025. Embodied ai agents: Modeling the world. *arXiv preprint arXiv:2506.22355*.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, et al. 2025. A survey of self-evolving agents: On path to artificial super intelligence. *arXiv preprint arXiv:2507.21046*.

Xinyan Guan, Yanjiang Liu, Xinyu Lu, Boxi Cao, Ben He, Xianpei Han, Le Sun, Jie Lou, Bowen Yu, Yaojie Lu, et al. 2024a. Search, verify and feedback: Towards next generation post-training paradigm of foundation models via verifier engineering. *arXiv preprint arXiv:2411.11504*.

Yanchen Guan, Haicheng Liao, Zhenning Li, Jia Hu, Runze Yuan, Guohui Zhang, and Chengzhong Xu. 2024b. World models for autonomous driving: An initial survey. *IEEE Transactions on Intelligent Vehicles*.

David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122*, 2(3).

Jeongsoo Ha, Kyungsoo Kim, and Yusung Kim. 2023. Dream to generalize: zero-shot model-based reinforcement learning for unseen visual distractions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7802–7810.

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

Dongge Han, Camille Couturier, Daniel Madrigal Diaz, Xuchao Zhang, Victor Rühle, and Saravan Rajmohan. 2025. Legomem: Modular procedural memory for multi-agent llm systems for workflow automation. *arXiv preprint arXiv:2510.04851*.

Kostas Hatalis, Despina Christou, Joshua Myers, Steven Jones, Keith Lambert, Adam Amos-Binks, Zohreh Dannenhauer, and Dustin Dannenhauer. 2023. Memory matters: The need to improve long-term memory in llm-agents. In *Proceedings of the AAAI Symposium Series*, volume 2, pages 277–280.

Yining Hong, Rui Sun, Bingxuan Li, Xingcheng Yao, Maxine Wu, Alexander Chien, Da Yin, Ying Nian Wu, Zhecan James Wang, and Kai-Wei Chang. 2025. Embodied web agents: Bridging physical-digital realms for integrated agent intelligence. *arXiv preprint arXiv:2506.15677*.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. 2023. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*.

Yanping Huang and Rajesh PN Rao. 2011. Predictive coding. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(5):580–593.

Zhengjun Huang, Zhoujin Tian, Qintian Guo, Fangyuan Zhang, Yingli Zhou, Di Jiang, and Xiaofang Zhou. 2025. Licomemory: Lightweight and cognitive agentic memory for efficient long-term reasoning. *arXiv preprint arXiv:2511.01448*.

Fan Jia, Weixin Mao, Yingfei Liu, Yucheng Zhao, Yuqing Wen, Chi Zhang, Xiangyu Zhang, and Tiancai Wang. 2023. Adriver-i: A general world model for autonomous driving. *arXiv preprint arXiv:2311.13549*.

Xun Jiang, Feng Li, Han Zhao, Jiahao Qiu, Jiaying Wang, Jun Shao, Shihao Xu, Shu Zhang, Weiling Chen, Xavier Tang, et al. 2024. Long term memory: The foundation of ai self-evolution. *arXiv preprint arXiv:2410.15665*.

Li Jin and Liu Jia. 2025. Embodied world models emerge from navigational task in open-ended environments. *arXiv preprint arXiv:2504.11419*.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. 2019. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*.

Lingdong Kong, Wesley Yang, Jianbiao Mei, Youquan Liu, Ao Liang, Dekai Zhu, Dongyue Lu, Wei Yin, Xiaotao Hu, Mingkai Jia, et al. 2025. 3d and 4d world modeling: A survey. *arXiv preprint arXiv:2509.07996*.

Dacheng Li, Yunhao Fang, Yukang Chen, Shuo Yang, Shiyi Cao, Justin Wong, Michael Luo, Xiaolong Wang, Hongxu Yin, Joseph E Gonzalez, et al. 2025a. Worldmodelbench: Judging video generation models as world models. *arXiv preprint arXiv:2502.20694*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Pony Ma, Guanhua Chen, Heng Ji, et al. 2025b. From word to world: Can large language models be implicit text-based world models? *arXiv preprint arXiv:2512.18832*.

Wenlong Liang, Rui Zhou, Yang Ma, Bing Zhang, Songlin Li, Yijia Liao, and Ping Kuang. 2025. Large model empowered embodied ai: A survey on decision-making and embodied learning. *arXiv preprint arXiv:2508.10399*.

Genglin Liu, Shijie Geng, Sha Li, Hejie Cui, Sarah Zhang, Xin Liu, and Tianyi Liu. 2025. Webcoach: Self-evolving web agents with cross-session memory guidance. *arXiv preprint arXiv:2511.12997*.

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. 2024. World model on million-length video and language with blockwise ringattention. *arXiv preprint arXiv:2402.08268*.

Xiaoxiao Long, Qingrui Zhao, Kaiwen Zhang, Zihao Zhang, Dingrui Wang, Yumeng Liu, Zhengjie Shu, Yi Lu, Shouzheng Wang, Xinzhe Wei, et al. 2025. A survey: Learning embodied intelligence from physical simulators and world models. *arXiv preprint arXiv:2507.00917*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Kai Mei, Jiang Guo, Shuaichen Chang, Mingwen Dong, Dongkyu Lee, Xing Niu, and Jiarong Jiang. 2025. R-wom: Retrieval-augmented world model for computer-use agents. *arXiv preprint arXiv:2510.11892*.

Russell Mendonca, Shikhar Bahl, and Deepak Pathak. 2023. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*.

Dujun Nie, Xianda Guo, Yiqun Duan, Ruijun Zhang, and Long Chen. 2025. Wmnav: Integrating vision-language models into world models for object goal navigation. *arXiv preprint arXiv:2503.02247*.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. 2025. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Joon Sung Park, Carolyn Q Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S Bernstein. 2024. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109*.

J Parker-Holder, P Ball, J Bruce, V Dasagi, K Holsheimer, C Kaplanis, A Moufarek, G Scully, J Shar, J Shi, et al. 2024. Genie 2: A large-scale foundation world model. *URL: https://deepmind. google/discover/blog/genie-2-a-large-scale-foundation-world-model*.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2024. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37:126544–126565.

Mathis Pink, Qinyuan Wu, Vy Ai Vo, Javier Turek, Jianing Mu, Alexander Huth, and Mariya Toneva. 2025. Position: Episodic memory is the missing piece for long-term llm agents. *arXiv preprint arXiv:2502.06975*.

Cheng Qian, Emre Can Acikgoz, Bingxuan Li, Xiusi Chen, Yuji Zhang, Bingxiang He, Qinyu Luo, Dilek Hakkani-Tür, Gokhan Tur, Yunzhu Li, et al. 2026. Current agents fail to leverage world model as tool for foresight. *arXiv preprint arXiv:2601.03905*.

Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Agent planning with world knowledge model. *Advances in Neural Information Processing Systems*, 37:114843–114871.

Jonathan Richens, Tom Everitt, and David Abel. 2025. General agents need world models. In *Forty-second International Conference on Machine Learning*.

Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. 2025. Meminsight: Autonomous memory augmentation for llm agents. *arXiv preprint arXiv:2503.21760*.

Jitao Sang, Jinlin Xiao, Jiarun Han, Jilin Chen, Xiaoyi Chen, Shuyu Wei, Yongjie Sun, and Yuhang Wang. 2025. Beyond pipelines: A survey of the paradigm shift toward model-native agentic ai. *arXiv preprint arXiv:2510.16720*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.

Junhao Shi, Zhaoye Fei, Siyin Wang, Qipeng Guo, Jingjing Gong, and Xipeng QIu. 2025. World-aware planning narratives enhance large vision-language model planner. *arXiv preprint arXiv:2506.21230*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Hao Tang, Darren Key, and Kevin Ellis. 2024. Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment. *Advances in Neural Information Processing Systems*, 37:70148–70212.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, et al. 2025. Agent kb: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387*.

Matthieu Tehenan, Christian Bolivar Moya, Tenghai Long, and Guang Lin. 2025. Linear spatial world models emerge in large language models. *arXiv preprint arXiv:2506.02996*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Hang Wang, Xin Ye, Feng Tao, Chenbin Pan, Abhirup Mallik, Burhaneddin Yaman, Liu Ren, and Junshan Zhang. 2025a. Adawm: Adaptive world model based planning for autonomous driving. *arXiv preprint arXiv:2501.13072*.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024a. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024b. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Siyin Wang, Zhaoye Fei, Qinyuan Cheng, Shiduo Zhang, Panpan Cai, Jinlan Fu, and Xipeng Qiu. 2025b. World modeling makes a better planner: Dual preference optimization for embodied task planning. *arXiv preprint arXiv:2503.10480*.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. 2024c. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European conference on computer vision*, pages 55–72. Springer.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Boyuan Wang, Xinze Chen, and Jiwen Lu. 2024d. Worlddreamer: Towards general world models for video generation via predicting masked tokens. *arXiv preprint arXiv:2401.09985*.

Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. 2024e. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14749–14759.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024f. Agent workflow memory. *arXiv preprint arXiv:2409.07429*.

Archana Warrier, Dat Nyugen, Michelangelo Naim, Moksh Jain, Yichao Liang, Karen Schroeder, Cambridge Yang, Joshua B Tenenbaum, Sebastian Vollmer, Kevin Ellis, et al. 2025. Benchmarking world-model learning. *arXiv preprint arXiv:2510.19788*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, Zhankui He, Yuanchen Bei, Xuying Ning, Mengting Ai, Yunzhe Li, Jingrui He, Ed H Chi, et al. 2025. Evo-memory: Benchmarking llm agent test-time learning with self-evolving memory. *arXiv preprint arXiv:2511.20857*.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. 2023. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pages 2226–2240. PMLR.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. 2023. Language models meet world models: Embodied experiences enhance language models. *Advances in neural information processing systems*, 36:75392–75412.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Kristian Kersting, Jeff Z Pan, Hinrich Schütze, et al. 2025. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, et al. 2025a. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. *arXiv preprint arXiv:2502.09560*.

Wei Yang, Jinwei Xiao, Hongming Zhang, Qingyang Zhang, Yanna Wang, and Bo Xu. 2025b. Coarse-to-fine grounded memory for llm agent planning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13040–13067.

Yuncong Yang, Jiageng Liu, Zheyuan Zhang, Siyuan Zhou, Reuben Tan, Jianwei Yang, Yilun Du, and Chuang Gan. 2025c. Mindjourney: Test-time scaling with world models for spatial reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Xuan Yao, Junyu Gao, and Changsheng Xu. 2025. Navmorph: A self-evolving world model for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2506.23468*.

Chenglin Yu, Yang Yu, Songmiao Wang, Yucheng Wang, Yifan Yang, Jinjia Li, Ming Li, and Hongxia Yang. 2025a. Infiagent: Self-evolving pyramid agent framework for infinite scenarios. *arXiv preprint arXiv:2509.22502*.

Jiwen Yu, Yiran Qin, Haoxuan Che, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. 2025b. Position: Interactive generative video as next-generation game engine. *arXiv preprint arXiv:2503.17359*.

Xiao Yu, Baolin Peng, Ruize Xu, Michel Galley, Hao Cheng, Suman Nath, Jianfeng Gao, and Zhou Yu. 2025c. Dyna-think: Synergizing reasoning, acting, and world model simulation in ai agents. *arXiv preprint arXiv:2506.00320*.

Hu Yue, Siyuan Huang, Yue Liao, Shengcong Chen, Pengfei Zhou, Liliang Chen, Maoqing Yao, and Guanghui Ren. 2025. Ewmbench: Evaluating scene, motion, and semantic quality in embodied world models. *arXiv preprint arXiv:2505.09694*.

Ruihong Zeng, Jinyuan Fang, Siwei Liu, and Zaiqiao Meng. 2024. On the structural memory of llm agents. *arXiv preprint arXiv:2412.15266*.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. 2025a. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*.

Guibin Zhang, Muxin Fu, and Shuicheng Yan. 2025b. Memgen: Weaving generative latent memory for self-evolving agents. *arXiv preprint arXiv:2509.24704*.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, et al. 2025c. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*.

Lunjun Zhang, Ge Yang, and Bradly C Stadie. 2021. World model as a graph: Learning latent landmarks for planning. In *International conference on machine learning*, pages 12611–12620. PMLR.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. 2025d. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. 2025e. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*.

Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang. 2025f. Memory as action: Autonomous context curation for long-horizon agentic tasks. *arXiv preprint arXiv:2510.12635*.

Zeyu Zhang, Quanyu Dai, Rui Li, Xiaohe Bo, Xu Chen, and Zhenhua Dong. 2025g. Learn to memorize: Optimizing llm-based agents with adaptive memory framework. *arXiv preprint arXiv:2508.16629*.

Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc Van Gool. 2023. Trafficbots: Towards world models for autonomous driving simulation and motion prediction. *arXiv preprint arXiv:2303.04116*.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2023. Synapse: Trajectory-as-exemplar prompting with memory for computer control. *arXiv preprint arXiv:2306.07863*.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, et al. 2025. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*.

Yufeng Zhong, Chengjian Feng, Feng Yan, Fanfan Liu, Liming Zheng, and Lin Ma. 2025. P3nav: A unified framework for embodied navigation integrating perception, planning, and prediction. *arXiv preprint arXiv:2503.18525*.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. 2025a. Memento: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*.

Siyu Zhou, Tianyi Zhou, Yijun Yang, Guodong Long, Deheng Ye, Jing Jiang, and Chengqi Zhang. 2024. Wall-e: World alignment by rule learning improves world model-based llm agents. *arXiv preprint arXiv:2410.07484*.

Siyu Zhou, Tianyi Zhou, Yijun Yang, Guodong Long, Deheng Ye, Jing Jiang, and Chengqi Zhang. 2025b. Wall-e 2.0: World alignment by neurosymbolic learning improves world model-based llm agents. *arXiv preprint arXiv:2504.15785*.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. 2025c. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*.

Zheng Zhu, Xiaofeng Wang, Wangbo Zhao, Chen Min, Nianchen Deng, Min Dou, Yuqi Wang, Botian Shi, Kai Wang, Chi Zhang, et al. 2024. Is sora a world simulator? a comprehensive survey on general world models and beyond. *arXiv preprint arXiv:2405.03520*.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR.

A Additional Evaluation on Fine-Grained Navigation Tasks

To further verify the generalization capability of WorldMind across different levels of action granularity, we extended our evaluation to the EB-Navigation benchmark. Unlike EB-ALFRED and EB-Habitat, which primarily focus on high-level semantic planning (e.g., "`Pick up the apple`"), EB-Navigation requires the agent to navigate using fine-grained atomic actions (e.g., "`Move forward 0.25m`", "`Rotate 90 degrees`") to locate target objects. This setting presents unique challenges in spatial reasoning and semantic environmental grounding.

We evaluated both GPT-3.5-turbo and GPT-4.1-mini backbones on four subsets: *Base*, *Common Sense*, *Complex Instruction*, and *Visual Appearance*.

Task Subset	**GPT-3.5-turbo**		**GPT-4.1-mini**	
	ReAct	**WorldMind**	ReAct	**WorldMind**
Base	56.7	**66.7**	53.3	**56.7**
Common Sense	**58.3**	55.0	**61.7**	60.0
Complex Instruction	66.7	**68.3**	55.0	**56.7**
Visual Appearance	43.3	**45.0**	46.7	**48.3**
Average	56.3	**58.8**	54.2	**55.4**

Table 5: Success rate (%) on EB-Navigation subsets. Results are reported for four capability-oriented subsets as well as the macro-average. Performance is shown for WorldMind and the baseline methods.

Analysis of Fine-Grained Navigation Generalization. Table 5 demonstrates that WorldMind effectively bridges the gap between high-level reasoning and fine-grained procedural execution. For the GPT-3.5-turbo backbone, WorldMind achieves a notable improvement in the Base subset (+10.0%), indicating that retrieving previously successful trajectories helps the agent maintain consistent behavior and avoid planning errors such as oscillatory movements that are commonly observed in ReAct baselines. Improvements in the Visual subset for both backbones further suggest that Process Experience helps align visual perception with environmental constraints, reducing hallucinations in which the agent incorrectly assumes it has reached the target. Although gains in the Common Sense subset are mixed, the overall increase in average success rate indicates that the benefits of the proposed framework, particularly the construction of a valid internal world model, generalize across dif-

ferent action spaces and provide robust guidance for fine-grained sequential navigation.

B Practical Overhead and Efficiency Analysis

To clarify the practical overhead of WorldMind, we analyze its memory footprint, token consumption, and inference latency. First, the World Knowledge Repository (WKR) maintains a compact memory footprint in our evaluated setting. By incrementally accumulating experiences and filtering out those with $\geq 95\%$ semantic similarity, the WKR size is bounded by the number of evaluated tasks in each subset (e.g., 50 per subset), reducing redundant growth during sequential evaluation. Second, experience construction incurs a one-time token cost. As shown in Table 6, generating a Process Experience requires an average of 3077.74 tokens, while summarizing a Goal Experience uses 1513.60 tokens. Crucially, this is a one-time construction overhead rather than a recurring per-step cost.

During the actual planning and execution phase, the computational overhead remains extremely low. Thanks to the compact WKR, the SentenceTransformer-based semantic retrieval is highly efficient. Retrieving the top-2 relevant experiences adds a negligible latency of only $\sim$0.0028 seconds per step on average. This demonstrates that once world knowledge is externalized, WorldMind operates with high efficiency and is well-suited for efficient embodied planning.

C Scalability of the World Knowledge Repository and Memory Budget Limits

To evaluate the scalability of the World Knowledge Repository (WKR) and its impact on agent performance, we conduct an ablation study by varying the memory budget limits from 0 (baseline without memory) to 50 experiences. We denote these settings as Mem0 through Mem50. The experiments are evaluated on the EB-ALFRED benchmark using the GPT-3.5-turbo backbone, and the results are presented in Table 7.

Overall, the agent demonstrates a strong capability to leverage the accumulated Process Experience and Goal Experience. As the memory size increases from Mem0 to Mem20, the average Success Rate (SR) steadily improves from 44.4% to 46.8%. Similarly, as the repository reaches Mem50, the performance further surges to 48.0%, indicating that the WKR scales effectively and continuously provides valuable semantic environmental grounding for complex tasks.

However, we observe a temporary performance drop at the Mem30 setting when using a strict top-2 retrieval limit. By carefully inspecting the failure cases and retrieval logs during this phase, we discover that this is a "semantic crowding" issue. Around the Mem20 to Mem30 sequence, the WKR happens to accumulate a cluster of experiences from highly similar tasks. During retrieval, these newly added, highly specific rules temporarily crowd out the broader, foundational environmental rules. Because the agent only retrieves the top-2 items, it occasionally misses the most optimal knowledge. As the memory continues to grow to Mem40 and Mem50, the repository absorbs a much wider variety of experiences, restoring global diversity and allowing the retriever to reliably surface the best rules.

To verify this semantic crowding hypothesis, we conduct a supplementary experiment on the Mem30 setting by simply increasing the retrieval limit from top-2 to top-3. As shown in Table 7, retrieving one additional piece of knowledge mitigates the crowding bottleneck. The average Success Rate immediately restores from 43.2% back to 46.8%, matching the peak performance at Mem20. This confirms that the stored experiences remain highly effective, and adjusting the retrieval size successfully mitigates local distribution biases during memory scaling.

D Performance on Proprietary Models and Open-Source VLMs

To demonstrate the generalizability of WorldMind across different model architectures, we extend our evaluation to include recent powerful proprietary models, specifically GPT-4.1, Claude-3.7-Sonnet, and Gemini-2.5-Pro, as well as a leading open-source Vision-Language Model (VLM), Qwen3-VL-235B-A22B-Instruct. The experiments are conducted on the EB-ALFRED benchmark, and the quantitative results are presented in Table 8.

The data indicates that WorldMind improves both the Success Rate (SR) and Goal Condition (GC) across these additional backbones in most subsets. By integrating the World Knowledge Repository, which contains Process Experience and Goal Experience, the agent obtains higher average GC than the corresponding baseline methods. The results further show that the framework remains

Table 6: Practical overhead of WorldMind, including token consumption for experience generation and retrieval latency across the EB-ALFRED benchmark subsets.

Metric	Base	Common	Complex	Visual	Spatial	Average
Process Exp. (Tokens/Reflection)	3121.67	3093.36	3149.79	3059.00	2964.88	3077.74
Goal Exp. (Tokens/Success)	1471.22	1536.84	1519.96	1532.83	1507.13	1513.60
Retrieval Time (Seconds/Step)	0.0012	0.0024	0.0072	0.0015	0.0015	0.0028

Table 7: Performance comparison of WorldMind with different memory budget limits on the EB-ALFRED benchmark. MemX indicates a maximum capacity of X experiences. The top-N notation denotes the retrieval size.

Memory Setting	Success Rate (SR %)						Goal Condition (GC %)					
	Avg	Base	Common	Complex	Visual	Spatial	Avg	Base	Common	Complex	Visual	Spatial
Mem0 (Baseline)	44.4	52.0	48.0	52.0	32.0	38.0	50.4	55.3	53.5	55.3	42.7	45.0
Mem10 (top-2)	46.0	48.0	56.0	48.0	42.0	36.0	51.0	53.1	58.7	54.3	45.0	43.7
Mem20 (top-2)	46.8	48.0	50.0	50.0	46.0	40.0	52.9	56.3	58.5	56.3	50.0	43.3
Mem30 (top-2)	43.2	46.0	42.0	46.0	38.0	44.0	49.3	54.2	47.8	51.0	44.0	49.3
Mem40 (top-2)	45.2	52.0	52.0	52.0	34.0	36.0	50.8	57.7	55.0	60.0	40.0	41.3
Mem50 (top-2)	48.0	58.0	48.0	56.0	34.0	44.0	54.1	63.0	52.7	61.0	41.7	52.0
Mem30 (top-3)	46.8	54.0	48.0	54.0	38.0	40.0	50.6	51.0	59.0	58.0	41.7	43.3

useful when the base planner is already a stronger proprietary model. For Claude-3.7-Sonnet, we run Full WorldMind as the base LLM across all five EB-ALFRED subsets and report both per-subset and five-subset mean results.

Furthermore, the framework also improves the open-source VLM backbone, suggesting that externalizing environmental grounding into symbolic knowledge is not tied to a single proprietary model family. By reducing the reliance on implicit model weights for environmental reasoning, WorldMind provides explicit semantic grounding across the evaluated proprietary and open-source backbones.

E Robustness Analyses

E.1 Adversarial Robustness

We further evaluate whether Process Experience helps under perturbed textual feedback using an adversarial robustness experiment on EB-ALFRED Base. The object-rename perturbation replaces 32 ALFRED object names with human-curated synonyms in textual `env_feedback` while preserving the simulator state and action parser. At WKR load time, the memory-poisoning perturbation randomly replaces 20% of stored experiences with predefined garbage rules that are structurally similar but semantically incorrect. The results in Table 9 show that WorldMind has a smaller GC drop than ReAct under object renaming, suggesting partial robustness to surface-form shifts in environment feedback.

Method	Perturbation	Base GC	Δ GC
WorldMind	clean	63.0	0.0
WorldMind	object rename (32 synonyms)	46.7	-16.3
WorldMind	memory poison (20% corrupted)	46.7	-16.3
ReAct	clean	55.3	0.0
ReAct	object rename	30.0	-25.3

Table 9: Adversarial robustness on EB-ALFRED Base. Object Rename replaces 32 object names with synonyms in textual `env_feedback`; Memory Poisoning replaces 20% of stored experiences with structurally similar but semantically incorrect rules.

E.2 Task-Order Robustness Check

Because the WKR is built sequentially, task order may affect the final repository. We therefore run a five-seed task-order shuffle on EB-ALFRED Base using GPT-3.5-turbo, randomly permuting task order across `task_order_seed`. Each seed corresponds to a different permutation of the task curriculum. Table 10 reports 64.0 ± 6.8 GC, indicating that performance is not dominated by one favorable ordering.

	Seed A	Seed B	Seed C	Seed D	Seed E	Mean ± Std
GC	66.7	66.7	53.3	73.3	60.0	**64.0 ± 6.8**

Table 10: Task-order robustness on EB-ALFRED Base (5 seeds, GPT-3.5-turbo). Each seed permutes the task curriculum.

Table 8: Performance comparison of WorldMind across different proprietary models and a leading open-source VLM on the EB-ALFRED benchmark.

Backbone	Method	Success Rate (SR %)						Goal Condition (GC %)					
		Avg	Base	Common	Complex	Visual	Spatial	Avg	Base	Common	Complex	Visual	Spatial
GPT-4.1	Baseline	62.4	66.0	50.0	72.0	64.0	60.0	65.1	67.7	52.7	74.3	66.7	64.2
GPT-4.1	WorldMind	64.0	72.0	72.0	72.0	46.0	58.0	71.6	78.3	78.8	79.0	56.2	65.7
Claude-3.7-Sonnet	Baseline	67.2	68.0	68.0	70.0	68.0	62.0	67.5	72.0	66.0	76.7	63.0	59.7
Claude-3.7-Sonnet	WorldMind	70.0	70.0	68.0	80.0	62.0	70.0	71.9	70.0	71.7	83.3	64.3	70.0
Gemini-2.5-Pro	Baseline	75.2	82.0	76.0	80.0	70.0	68.0	75.5	82.0	76.7	80.0	71.0	68.0
Gemini-2.5-Pro	WorldMind	76.0	84.0	76.0	84.0	70.0	66.0	80.3	84.0	76.0	88.0	76.2	77.1
Qwen3-VL-235B-A22B-Instruct	Baseline	56.8	60.0	56.0	60.0	50.0	58.0	60.4	66.0	58.5	62.0	55.2	60.2
Qwen3-VL-235B-A22B-Instruct	WorldMind	59.2	68.0	58.0	66.0	46.0	58.0	64.6	72.7	59.8	73.2	53.7	63.7

E.3 Failure Case Taxonomy

Beyond the quantitative error redistribution in Figure 4, we manually inspect 10 representative failure episodes from EB-Habitat Visual Appearance and Spatial Awareness. Table 11 summarizes the observed categories. The largest category is VLM perception failure, indicating that WorldMind cannot fully compensate for misclassified visual states. Retrieval misses and reflection over-generalization point to future improvements in adaptive top-k retrieval and more conservative rule distillation.

Failure Type	Share	Likely Improvement Direction
Retrieval miss	≈25%	increase or adapt top-k retrieval
Judge over-strict	≈10%	calibrate the consistency validator
Reflection over-generalization	≈15%	store narrower contextual rules
VLM perception failure	≈40%	strengthen visual grounding
Plan-level error	≈10%	improve high-level planning policy

Table 11: Failure taxonomy from 10 representative EB-Habitat Visual Appearance and Spatial Awareness failures. Percentages are descriptive because of the small sample size.

F Algorithmic Details and Experience Examples

Algorithms 1 and 2 summarize the inference and repository-update procedures. The algorithms make explicit that WorldMind does not update model parameters; it updates the external WKR through Process and Goal Experience.

Algorithm 1 WorldMind Inference Loop

Input: goal g, initial observation o_0, WKR $\mathcal{W} = \{\mathcal{W}_p, \mathcal{W}_g\}$
for $t = 0, \ldots, T-1$ **do**
  Retrieve top-k relevant Process and Goal experiences from $\mathcal{W}$ using g and the current context.
  Generate action a_t and predicted next state $\hat{s}_{t+1}$ conditioned on o_t, g, and retrieved knowledge.
  Execute a_t in the environment and observe feedback/state s_{t+1}.
  Abstract s_{t+1} into text $\bar{s}_{t+1}$.
  if $\hat{s}_{t+1}$ contradicts $\bar{s}_{t+1}$ **then**
    Create a Process Experience from the prediction error and update $\mathcal{W}_p$.
  end if
  if the episode succeeds **then**
    Distill the trajectory into Goal Experience and update $\mathcal{W}_g$.
    break
  end if
end for

Algorithm 2 World Knowledge Repository Update

Input: candidate experience e, repository $\mathcal{W}$, similarity threshold $\theta = 0.95$
Embed e and existing repository entries with SentenceTransformer.
Compute the maximum cosine similarity between e and entries in the matching repository partition.
if maximum similarity $< \theta$ **then**
    Insert e into $\mathcal{W}_p$ or $\mathcal{W}_g$ according to its source.
else
    Discard e as a near-duplicate.
end if
During inference, retrieve only top-k relevant entries to keep the prompt budget bounded.

Safety Handling. All Process Experience used in our experiments is generated from interaction feedback; we do not pre-encode safety rules in the WKR. The current setup therefore relies on the benchmark action space and the underlying VLM's safety alignment to prevent unsafe outputs. For deployment beyond simulators, explicit safety constraints should be added as a separate verified layer rather than inferred only from post-hoc experience.

G Baseline Methods Description

Table 13 summarizes the baselines used in Tables 1 and 2. The Memory Scope column in the main tables uses two categories: *Episodic* methods do not maintain a cross-task global repository, whereas *Global* methods use a cross-task global memory repository.

H Prompts

H.1 WorldMind System Prompt for ALFRED

System Prompt for WorldMind in EB-ALFRED

You are an intelligent embodied agent operating in a home environment, equipped with an internal World Model.
You do not merely execute commands; you simulate the outcome of your actions before execution. For every step, you must think deeply about how your action will alter the environment to ensure the task is completed successfully. Your state prediction serves as a justification for your action—proving that you understand the consequences of your move.
Action Descriptions and Validity Rules
• Find: Parameterized by the name of the receptacle to navigate to. So long as the object is present in the scene, this skill is always valid
• Pick up: Parameterized by the name of the object to pick. Only valid if the robot is close to the object, not holding another object, and the object is not inside a closed receptacle.
• Put down: Parameterized by the name of the object to put down to a nearby receptacle. Only valid if the robot is holding an object.
• Drop: Parameterized by the name of the object to put down. It is different from Put down action, as this does not guarantee the held object will be put into a specified receptacle.
• Open: Parameterized by the name of the receptacle to open. Only valid if the receptacle is closed and the robot is close to the receptacle.
• Close: Parameterized by the name of the receptacle to close. Only valid if the receptacle is open and the robot is close to the receptacle.
• Turn on: Parameterized by the name of the object to turn on. Only valid if the object is turned off and the robot is close to the object.
• Turn off: Parameterized by the name of the object to turn off. Only valid if the object is turned on and the robot is close to the object.
• Slice: Parameterized by the name of the object to slice. Only valid if the object is sliceable and the robot is close to the object.
The available action id (0 ~ {}) and action names are: {}.
{}
Guidelines
1. Output Plan: Avoid generating empty plan. Each plan should include no more than 20 actions.
2. Visibility: Always locate a visible object by the 'find' action before interacting with it.
3. Action Guidelines: Make sure match the action name and its corresponding action id in the output.
Avoid performing actions that do not meet

Experience Type	Triggering Context	Stored Entry Example
Process Experience	The agent predicts that placing an object will succeed, but the environment reports that the agent is not at the intended receptacle.	Before executing a placement action, first navigate to the specific destination receptacle; otherwise the object may be placed at the wrong location or the action may be invalid.
Process Experience	The instruction requires multiple objects, but the agent tries to pick another object while still holding one.	For multi-item tasks, complete the loop source $\rightarrow$ pick $\rightarrow$ destination $\rightarrow$ place before searching for the next item; the hand must be empty before picking again.
Goal Experience	A successful ALFRED trajectory completes a heat-and-place task.	Find the object, pick it up, navigate to the microwave, open it, place the object inside, heat it, retrieve it, then navigate to the target receptacle and place it.
Goal Experience	A successful Habitat trajectory moves all target objects to a destination.	Repeat search, pick, navigate, and place for each target object, checking after each placement whether additional targets remain.

Table 12: Representative Process and Goal Experience entries. These examples illustrate the format of stored symbolic knowledge; exact entries vary with task feedback.

the defined validity criteria. For instance, if you want to put object in a receptacle, use 'put down' rather than 'drop' actions.
If the last action's environment feedback is "Last action executed successfully.", you MUST NOT repeat the same action as your next step.
4. Prevent Repeating Action Sequences: Do not repeatedly execute the same action or sequence of actions.
Try to modify the action sequence because previous actions do not lead to success.
5. Multiple Instances: There may be multiple instances of the same object, distinguished by an index following their names, e.g., Cabinet_2, Cabinet_3. You can explore these instances if you do not find the desired object in the current receptacle.
6. Reflection on History and Feedback: Use interaction history and feedback from the environment to refine and improve your current plan.
If the last action is invalid, reflect on the reason, such as not adhering to action rules or missing preliminary actions, and adjust your plan accordingly.
7. Dynamic Reasoning from Environment Feedback: You must treat 'env_feedback' as a direct instruction.
- Instruction Extraction: If feedback says "Ladle is in CounterTop_2", your 'language_plan' must state: "Feedback indicates Ladle is at CounterTop_2, navigating there now."
- Action Alignment: Your next action MUST be "find a CounterTop_2". Do not use generic names if a specific index is provided.
- Multiple Instances Handling: If the environment contains multiple instances of a receptacle (e.g., several CounterTops), you must use the specific instance indicated by feedback. Failing to navigate to the correct instance (such as only using a generic "CounterTop") will result in the target object remaining invisible or inaccessible.
8. The Anti-Loop Rule: If a "Pick up" or "Turn on" action fails or the object is "not visible", DO NOT repeat the same action. You must change your strategy in the next plan

Method	Memory Scope	Description
ReAct	Episodic	Interleaves reasoning and acting within the current episode; no cross-task repository is used.
BoN	Episodic	Samples multiple candidate action traces and selects the best candidate under the evaluation protocol; it does not store reusable cross-task memory.
SimuRA	Episodic	Uses simulative reasoning within the current task context; no global cross-task repository is used in this comparison.
Synapse	Global	Stores trajectories as exemplars for later computer-control decisions.
AWM	Global	Maintains workflow memories distilled from prior successful behavior.
ReasoningBank	Global	Stores reasoning memories for self-evolving agents and retrieves them for future tasks.
WorldMind	Global	Stores both prediction-error-derived Process Experience and success-derived Goal Experience, with retrieval and consistency verification during inference.

Table 13: Brief baseline descriptions and the two memory-scope labels used in the main result tables.

(e.g., move to a different instance, change perspective, or clear your hand).
9. Hand-State Awareness: Before every "Pick up" action, verify your hand state in the history. If you are holding an object, the very next action in your ‘executable_plan‘ MUST be "Put down" or "Drop" to clear the gripper.
10. World Model Prediction Case A/B:

(Case A) If the target is VISIBLE or its specific location (e.g., CounterTop_2) is KNOWN from ‘env_feedback‘, describe the specific change.

(Case B) If the location is unknown (Exploration), use: "Exploration phase: target not visible, prediction skipped."
11. Handle Non-Canonical Object Descriptions: When the instruction refers to an object using a non-canonical or descriptive name (e.g., "wooden table"), and you are unsure which specific object it maps to in the environment, you should attempt to perform the required operation on all plausible candidate objects until the task succeeds or feedback clarifies the correct target.
12. Never Output an Empty Plan Unless Success Is Confirmed: If the task isn't explicitly confirmed as successful by feedback, continue planning. If you think it's done but no success message appears, assume a mistake was made.
The output json format should be {"language_plan": str, "executable_plan": List[{"action_id": int, "action_name": str, "predicted_state": str}...]}
The fields in the above JSON follow the purpose below:
1. language_plan: Your Chain-of-Thought. You must think step-by-step based on the summarized experiences (generalizable lessons) provided in the context. Analyze the instruction, apply learned rules to avoid past mistakes, and derive a logical strategy. Explain why you prioritize certain locations or actions.
2. executable_plan: A list of concrete actions. Each object MUST contain: action_id, action_name, and predicted_state.

- For the predicted_state field, you must strictly follow these rules:

(Case A) If the target objects are VISIBLE in the current observation, describe the specific environmental and gripper change.

(Case B) If the target object or destination is NOT VISIBLE (Exploration Phase), you MUST output exactly the string: "Exploration phase: target not visible, prediction

skipped."

(Cascading Skip Rule) Once you output the specific skip string for any action, ALL SUBSEQUENT ACTIONS in the same plan MUST also use this exact same string. You cannot resume prediction after skipping it within a single plan.

!!! Please do not output anything other than the above-mentioned JSON, do not include “‘json and “‘!!!

H.2 WorldMind System Prompt for Habitat

System Prompt for WorldMind in EB-Habitat

You are an intelligent embodied agent operating in a home environment, equipped with an internal World Model.

You do not merely execute commands; you simulate the outcome of your actions before execution. For every step, you must think deeply about how your action will alter the environment to ensure the task is completed successfully. Your state prediction serves as a justification for your action—proving that you understand the consequences of your move.

Core Philosophy: Simulate (Physics + Semantics) -> Validate -> Execute

Before selecting any action, you must mentally simulate its outcome on two levels:

1. Physical Feasibility: Can I actually perform this action? (e.g., hands full).

2. Semantic Plausibility: Does this action make sense for the task? (e.g., searching for a pillow in the bathroom is semantically invalid).

Your ‘predicted_state‘ is the logical prerequisite that justifies why the selected action is the correct next step.

Action Descriptions and Validity Rules

• Navigation: Parameterized by the name of the receptacle to navigate to. So long as the receptacle is present in the scene, this skill is always valid.

• Pick: Parameterized by the name of the object to pick. Only valid if the robot is close to the object, not holding another object, and the object is not inside a closed receptacle.

• Place: Parameterized by the name of the receptacle to place the object on. Only valid if the robot is close to the receptacle and is holding an object.

• Open: Parameterized by the name of the receptacle to open. Only valid if the receptacle is closed and the robot is close to the receptacle.

• Close: Parameterized by the name of the receptacle to close. Only valid if the receptacle is open and the robot is close to the receptacle.

The available action id (0 ~ {}) and action names are: {}.

{}

Guidelines

1. Output Plan: Avoid generating empty plan. Each plan should include no more than 20 actions.

2. Visibility: If an object is not currently visible, use the "Navigation" action to locate it or its receptacle before attempting other operations.

3. Action Validity: Make sure match the action name and its corresponding action id in the output. Avoid performing actions that do not meet the defined validity criteria.

4. Prevent Repeating Action Sequences: Do not repeatedly execute the same action or sequence of actions. Try to modify the action sequence because previous actions do not lead to success.

5. Multiple Instances: There may be multiple instances of the same object, distinguished by an index following their names, e.g., cabinet 2, cabinet 3. You can explore these instances if you do not find the desired object in the current receptacle.

6. Reflection on History and Feedback: Use interaction history and feedback from the environment to refine and enhance your current strategies and actions. If the last action is invalid, reflect on the reason.

7. World Model Prediction: For EACH action in your ‘executable_plan‘, you MUST include a ‘predicted_state‘.

- Explain via Prediction: This prediction is your rationale. By describing the expected future, you prove this action moves you closer to the goal.

- Visual Specifics: Describe exactly what the robot will see and hold *immediately after* the action.
8. Prioritize Likely Locations via Semantic Simulation: Do not search randomly. Before navigating, run a semantic simulation in your World Model:
- Step A (Hypothesis): "Could target object X be at location Y?"
- Step B (Common Sense Check): Use everyday knowledge.
- *Example 1*: Target is "airplane" (toy). Candidate is "sink". -> Simulation Result: Very Unlikely. -> Decision: REJECT.
- *Example 2*: Target is "airplane". Candidate is "living room table". -> Simulation Result: Likely. -> Decision: ACCEPT.
- Action: Only generate Navigation actions for locations that pass this "Common Sense Check."
9. Exhaustive Local Search (The Left/Right Rule): Many receptacles have multiple parts (e.g., "Kitchen Counter Left" and "Kitchen Counter Right").
- If you navigate to one side (e.g., Left) and the object is NOT there, your immediate next step must be to check the other side (e.g., Right) before leaving the room.
- Do not jump to a different room until you have checked all connected segments of the current furniture.
10. Never Output an Empty Plan Unless Task Success Is Confirmed: If the environment feedback does not explicitly indicate that the task has been successfully completed, you must never output an empty action plan. Always carefully check your action history and environment feedback. If you believe the task is finished but have not received a success confirmation, assume there was a mistake and continue planning actions to achieve the goal.
The output json format should be {"language_plan": str, "executable_plan": List[{"action_id": int, "action_name": str, "predicted_state": str}...]}
The fields in the above JSON follow the purpose below:
1. language_plan is for your Chain-of-Thought. You must think step-by-step based on the summarized experiences (generalizable lessons) provided in the context. Analyze the instruction, apply these learned rules to avoid past mistakes, and derive a logical solution strategy. Explicitly explain your reasoning for prioritizing certain locations or actions based on these experiences.
2. executable_plan is a list of concrete actions to be executed. Each object in the list MUST contain: action_id, action_name, and predicted_state.
- For the "predicted_state" field, you must strictly follow these rules:
(Case A) If the target objects are VISIBLE in the current observation OR their location is KNOWN from interaction history, describe the specific environmental change.
(Case B) If the target object or destination is NOT VISIBLE AND location is NOT KNOWN from history (Exploration Phase), you MUST output exactly the string: "Exploration phase: target not visible, prediction skipped."
(Cascading Skip Rule) Once you output the specific skip string for any action, ALL SUBSEQUENT ACTIONS in the same list MUST also use this exact same string. You cannot resume prediction after skipping it within a single plan.
!!! Please do not output anything other than the above-mentioned JSON, do not include “‘json and “‘!!!

H.3 Judge Prompt for ALFRED

Prompt for Judge in EB-ALFRED

You are a Logical Consistency Validator for an embodied agent in the Alfred household environment.
Your SOLE purpose is to detect Factual Contradictions between the Agent’s Prediction and the Actual Observation.
The Alfred Environment Context

- Valid Actions: Find, Pick up, Put down, Drop, Open, Close, Turn on, Turn off, Slice
- Valid Receptacles: Fridge, Cabinet, Drawer, Microwave, CounterTop, Din-

ingTable, SideTable, Sink, StoveBurner, GarbageCan, Shelf

- Valid Objects: Apple, Bread, Tomato, Lettuce, Potato, Egg, Knife, Fork, Spoon, Cup, Mug, Bowl, Plate, Pot, Pan, etc.

The "Default True" Principle
CRITICAL RULE: You must maintain a "Presumption of Innocence" for the agent.

1. Irrelevance is True: If the description in `Agent's Predicted State After Action` and `Actual State After Action` are unrelated (discussing different objects) and do not logically contradict each other, you MUST return `"match": true`.
2. Key Consistency: You are strictly comparing the value of `Agent's Predicted State After Action` against `Actual State After Action`.
3. Conflict Driven: ONLY return `false` if and only if there is a direct, undeniable factual conflict (e.g., A says "X is Open", B says "X is Closed"). In all other cases (ambiguity, missing info, different focus), return `true`.

Your Strict Analysis Protocol (Step-by-Step)
Step 1: Fact Extraction (The "No Intent" Filter)
Before comparing, you must clean the data.

- Filter `Agent's Predicted State After Action`: Extract ONLY objects and states mentioned in Affirmative Declarative Sentences (e.g., "I am holding X", "X is on Y").
  - CRITICAL: IGNORE, DISCARD, and REMOVE any objects mentioned in future tense, intent, or plan-checking sentences (e.g., "I will check for a pear", "I intend to find X"). These are thoughts, not state predictions.
- Filter `Actual State After Action`: Similarly, ensure you only focus on factual descriptions of what is currently visible.

Step 2: Intersection Analysis
- Identify Shared Objects that exist in both the *Filtered* Prediction and the *Filtered* Actual State.
- If there are NO Shared Objects after filtering:
  - Check for Implicit Conflict (e.g., Pred: "Holding Apple" vs Actual: "Hand Empty").
  - If no implicit conflict exists, the result is TRUE (Irrelevant descriptions match by default).

Step 3: Conflict Verification
- For the Shared Objects, check for Mutually Exclusive States:
  - Position: "On CounterTop" vs "In Fridge" $\rightarrow$ FALSE.
  - State: "Open" vs "Closed" $\rightarrow$ FALSE.
  - Hand: "Holding object" vs "Hand empty" $\rightarrow$ FALSE.
  - Object State: "Sliced" vs "Whole", "Cooked" vs "Raw" $\rightarrow$ FALSE.
- Any difference in phrasing, detail level, or synonym usage is NOT a conflict.

Output Format
Output ONLY a JSON object:
{
"match": true, // or false
"reason": "Explain ONLY the factual conflict if false. If true, state 'No factual contradictions found' or 'Descriptions are unrelated'."
}
Examples
Example 1 (Intent Filtering)
Agent's Predicted State After Action: "I am at the counter. I will pick up the knife to slice the tomato."
Actual State After Action: "The robot is facing the counter. A knife is on the counter. Hand is empty."
Analysis:

1. Filter: Pred Fact: "At counter". (Ignore "will pick up knife" → Intent). Actual Fact: "Facing counter", "Knife on counter", "Hand empty".
2. Intersection: "Counter".
3. Check: Locations match. The "knife" intent is ignored.

Result:
{
"match": true,
"reason": "Location matches. Ignored the 'knife' in prediction as it was part of an intent statement ('will pick up'), not a state assertion."
}
Example 2 (Factual Conflict - Hand State)
Agent's Predicted State After Action: "I am holding the apple."
Actual State After Action: "The robot's hand is empty. The apple is on the table."
Analysis:

1. Filter: Facts extracted.
2. Intersection: Apple, Hand.
3. Conflict: Pred says "Holding apple", Actual says "Hand empty".

Result:
{
"match": false,
"reason": "Factual contradiction detected: Agent predicted holding the apple, but the actual state shows the hand is empty."
}
Example 3 (Irrelevant Descriptions - Default True)
Agent's Predicted State After Action: "The fridge door is closed."
Actual State After Action: "The robot is holding a knife. The counter is visible."
Analysis:

1. Filter: Facts extracted.
2. Intersection: None. (Prediction talks about Fridge; Actual talks about Knife/-Counter).
3. Conflict Check: Does "Holding knife" contradict "Fridge closed"? No.

Result:
{
"match": true,
"reason": "No shared objects and no logical contradiction between descriptions. Defaulting to true."
}

H.4 Judge Prompt for Habitat

Prompt for Judge in EB-Habitat

You are a Logical Consistency Validator for an embodied agent.
Your SOLE purpose is to detect Factual Contradictions between the Agent's Prediction and the Actual Observation.
The "Default True" Principle
CRITICAL RULE: You must maintain a "Presumption of Innocence" for the agent.

1. Irrelevance is True: If the description in `Agent's Predicted State After Action` and `Actual State After Action` are unrelated (discussing different objects) and do not logically contradict each other, you MUST return `"match": true`.
2. Key Consistency: You are strictly comparing the value of `Agent's Predicted State After Action` against `Actual State After Action`.
3. Conflict Driven: ONLY return `false` if and only if there is a direct, undeniable factual conflict (e.g., A says "X is Open", B says "X is Closed"). In all other cases (ambiguity, missing info, different focus), return `true`.

Your Strict Analysis Protocol (Step-by-Step)
Step 1: Fact Extraction (The "No Intent" Filter)
Before comparing, you must clean the data.

- Filter `Agent's Predicted State After Action`: Extract ONLY objects and states mentioned in Affirmative

Declarative Sentences (e.g., "I am holding X", "X is on Y").

  - CRITICAL: IGNORE, DISCARD, and REMOVE any objects mentioned in future tense, intent, or plan-checking sentences (e.g., "I will check for a pear", "I intend to find X"). These are thoughts, not state predictions.

- Filter `Actual State After Action`: Similarly, ensure you only focus on factual descriptions of what is currently visible.

Step 2: Intersection Analysis

- Identify Shared Objects that exist in both the *Filtered* Prediction and the *Filtered* Actual State.
- If there are NO Shared Objects after filtering:
  - Check for Implicit Conflict (e.g., Pred: "Holding Apple" vs Actual: "Gripper Empty").
  - If no implicit conflict exists, the result is TRUE (Irrelevant descriptions match by default).

Step 3: Conflict Verification

- For the Shared Objects, check for Mutually Exclusive States:
  - Position: "On Table" vs "On Sofa" $\rightarrow$ FALSE.
  - State: "Open" vs "Closed" $\rightarrow$ FALSE.
  - Gripper: "Full" vs "Empty" $\rightarrow$ FALSE.
- Any difference in phrasing, detail level, or synonym usage is NOT a conflict.

Output Format
Output ONLY a JSON object:
{
"match": true, // or false
"reason": "Explain ONLY the factual conflict if false. If true, state 'No factual contradictions found' or 'Descriptions are unrelated'."
}

Examples
Example 1 (Intent Filtering - The "Pear" Case)
Agent's Predicted State After Action: "I am at the left counter. I will check for a pear here to see if it is ripe."
Actual State After Action: "The robot is facing the left counter. The surface is empty."
Analysis:

1. Filter: Pred Fact: "At left counter". (Ignore "check for pear" $\rightarrow$ Intent). Actual Fact: "Facing left counter", "Surface empty".
2. Intersection: "Left Counter".
3. Check: Locations match. The "pear" is ignored because it was only in an intent sentence.

Result:
{
"match": true,
"reason": "Location matches. Ignored the 'pear' in prediction as it was part of an intent statement ('will check'), not a state assertion."
}
Example 2 (Irrelevant Descriptions - Default True)
Agent's Predicted State After Action: "The cabinet door is closed."
Actual State After Action: "The robot is holding a sponge. The floor is visible."
Analysis:

1. Filter: Facts extracted.
2. Intersection: None. (Prediction talks about Cabinet; Actual talks about Sponge/Floor).
3. Conflict Check: Does "Holding sponge" contradict "Cabinet closed"? No.

Result:
{
"match": true,
"reason": "No shared objects and no logical contradiction between descriptions. Defaulting to true."
}

Example 3 (Factual Conflict - Explicit Mismatch)
Agent's Predicted State After Action: "I am holding the apple."
Actual State After Action: "The gripper is empty. The apple is on the table."
Analysis:

1. Filter: Facts extracted.
2. Intersection: Apple, Gripper.
3. Conflict: Pred says "Holding", Actual says "Empty".

Result:
{
"match": false,
"reason": "Factual contradiction detected: Agent predicted holding the apple, but the actual state shows the gripper is empty."
}

H.5 Reflection Prompt for ALFRED

Prompt for Reflection in EB-ALFRED

Role
You are a Root Cause Analyst for an Alfred Agent. Your goal is to extract concise World Knowledge and Universal Logic by analyzing the gap between predictions and reality.
Task: Integrated Environmental Logic & Workflows
For every failure, generate a logic entry that first explains the prediction failure in the specific context, and then derives the Universal Corrective Rule/Workflow.
Mandatory Logic Structure:
"Environmental Logic: My prediction that [expected specific state] failed because [specific reason]. Therefore, [Universal Abstract Rule/Workflow]."
Key Atomic Rules to Integrate:

1. Indexing & Precision: If feedback mentions a specific index (e.g., CounterTop_2), navigating to a generic name fails. You must use the exact indexed name.
2. Hand Constraints: Carrying two items is impossible. Must "Put down" current item before picking a new one.
3. Container State: Must "Open" enclosed receptacles (Fridge/Cabinet/Microwave) before interacting with contents.
4. Destination Awareness Before Placement: When analyzing the Trajectory (Recent History), if a "put down the object in hand" action is executed, check if the immediately preceding action was a "find" action targeting the intended destination. If not, in Environmental Logic, state: "Before executing 'put down the object in hand', you must use 'find' to explicitly navigate to the intended destination receptacle. This ensures the object will be placed at the correct location and avoids placement errors."

Special Workflow Extraction (Universal Templates)
If the Human Instruction contains any of the following cases and the agent's prediction failed due to misunderstanding or missing the required workflow, you MUST append the corresponding workflow after the Environmental Logic.
Before describing the workflow, you should briefly clarify the intended goal in your own words.
Note: No object is ever already heated, cooled, sliced, or cleaned in the environment. If the instruction refers to an "already" processed object (e.g., "microwaved bread", "sliced apple", "cold lettuce", "cleaned plate"), you MUST perform the corresponding workflow to achieve that state before proceeding to the next step.
Output Format (Strict JSON)
Output a JSON object with one key: `experience_entry` (a list of strings).
Each logic string must begin with the prediction failure analysis.

H.6 Reflection Prompt for Habitat

Prompt for Reflection in EB-Habitat

Role
You are a Root Cause Analyst & Knowledge Extractor for an Embodied Agent. Your goal is to analyze the Agent's interaction history and failure context to generate World Knowledge Entries. You extract TRUTH from the history and LOGIC from the failures, specifically focusing on physical constraints and task execution workflows.
Task 1: Extract Object Locations (Success Analysis)
Scan the `Recent History` for any successful object interactions.

- Logic: If the agent successfully Picked object Y, look backwards for the last successful Navigation to X.
- Conclusion: Object Y is located at X.
- Output Format: "Location Knowledge: [Object Name] is located at [Receptacle Name]."

Task 2: Extract Environmental & Workflow Logic (Failure Analysis)
Compare the `Planner's Prediction` vs. `Actual Observation`. Diagnose why the plan failed using these specific guidelines:
Guidelines for Logic Extraction:

1. The "All" Objects Workflow Violation:
    - Context: Human Instruction involves "all" items (e.g., "Put all apples in sink").
    - Refining Logic: After you have found the location of a target object, you must strictly initiate a repetitive loop: [Navigate to Source $\rightarrow$ Pick $\rightarrow$ Navigate to Destination $\rightarrow$ PLACE].
    - Special Note: Not all target objects are necessarily in the same place. For each object, you may need to search and repeat the loop separately.
    - Root Cause Check: Did the agent attempt to pick a second item without placing the first one?
    - Entry: "Workflow Logic: For 'all items' tasks, after you have found the location of a target object, you must execute the loop: [Navigate to Source $\rightarrow$ Pick $\rightarrow$ Navigate to Destination $\rightarrow$ PLACE]. Not all objects are necessarily in the same place, so you may need to search and repeat for each. You failed because you skipped the 'Place' action. You MUST ensure your hand is empty before returning to search for or pick the next item."
2. The Multi-Goal Sequential Rule:
    - Context: Goal involves multiple distinct targets (e.g., "Ball to table, Hammer to box").
    - Failure: Starting a new sub-goal while the current one is physically incomplete (hand not empty).
    - Entry: "Strategic Logic: For multi-target instructions, you MUST complete goals one-by-one. Ensure the current item is successfully PLACED and the gripper is empty before navigating to the next target's location."
3. Visibility & Proximity:
    - Context: Feedback says "object not near".
    - Entry: "To interact with [Object], you must first Navigate to its specific Receptacle, not just the general area."
4. Hand Limits & Preconditions:
    - Entry: "To Pick items inside a [Container], you must Open it first." / "To Pick a new object, you must Place the current one to free the gripper."
5. Ambiguous Pick Failure:
    - Context: Environment Feedback is "Last action is invalid. Robot cannot pick any object that is not near the robot. Navigate to other place to find the object."

- Logic: There are two possibilities: 1. The agent has not yet found the object (needs to navigate to its location). 2. The object is already in the agent's hand (check if a successful Pick action occurred earlier).
- Entry: "Ambiguous Pick Failure: When receiving this feedback, check your action history. If you have already successfully picked up the object, do not attempt to pick it again. Otherwise, you need to navigate to the correct location to find the object."

Output Format (Strict JSON)
Output a JSON object with exactly ONE key: `experience_entry`.
`experience_entry`: A List of Strings containing Location Knowledge and Environmental/Workflow Logic.

H.7 Goal Experience Extraction Prompt for ALFRED

Prompt for Goal Experience in EB-ALFRED

You are a task analysis expert. Given a successful task execution trajectory with task_progress information, extract a SINGLE reusable workflow that captures the key actions that led to success.
Task Instruction
{instruction}
Successful Execution Trajectory (with Task Progress)
{trajectory}
Task Category
{task_category}
CRITICAL WORKFLOW EXTRACTION GUIDELINES
Focus on Progress-Boosting Actions
The trajectory includes "task_progress" after each step. Pay SPECIAL ATTENTION to:

1. Actions that INCREASED task_progress - These are the KEY ACTIONS that actually advanced the task
2. The sequence of actions leading to progress increases - This reveals the correct approach
3. Actions that maintained or prepared for progress - Supporting actions matter too

Workflow Requirements

1. Extract ONE concise, generalizable workflow - Not multiple workflows
2. Focus on the ACTION PATTERN, not specific objects - Use generic terms like "target object", "destination location"
3. Highlight the critical decision points - When to navigate vs. pick vs. place
4. Capture the logical order - The sequence matters for success
5. Be specific about SUCCESS CONDITIONS - What made this approach work

Abstraction Guidelines

- Replace specific object names (e.g., "apple", "table_42") with generic terms (e.g., "[TARGET_OBJECT]", "[DESTINATION]")
- Keep action types specific (navigate, pick, place, open, close)
- Preserve the causal relationships between actions

Special Case: Non-Canonical Object Descriptions
If the instruction refers to an object using a non-canonical or descriptive name (e.g., "wooden table"), and it is unclear which specific object in the environment matches this description, the workflow should include: "Attempt the required operation on all plausible candidate objects until the task succeeds or feedback clarifies the correct target."
Special Case: "with" Placement Instructions
If the instruction requires placing an object "with" another object (e.g., "set plate with a spoon in it on the kitchen table"), the workflow should be: first, find and pick up the

"with" object (e.g., the spoon); then, find the main object (e.g., the plate) and put down the "with" object into/on it; next, pick up the main object (now containing the "with" object); finally, find the destination and put down the main object at the target location.
Special Case: General Placement Instructions
For placement instructions, the workflow should be: first, find the target object; then, pick up the object; next, find the destination location; finally, use "put down the object in hand" to place the object at the destination.
Special Case: Inferring and Trying Candidate Objects
When the instruction refers to an object, first infer which available action id objects in the current environment may correspond to the instruction. Then, attempt the required operation on each plausible candidate object until the task succeeds or feedback clarifies the correct target.
Special Case: Cleaning Workflow for "Clean" Instructions
If the instruction requires a "clean" object, you must execute the cleaning workflow: pick up the object, find a sink, put down the object in hand into the sink, find the faucet and turn it on and off, then pick up the object again. Only after this sequence is the object considered clean.
Please extract a SINGLE reusable workflow that captures the essence of this successful execution.
Output format:
{{"success_experience": "A complete workflow description that includes: (1) The goal pattern, (2) The key action sequence with EMPHASIS on progress-boosting actions, (3) Critical decision points, (4) Success conditions. Should be 3-6 sentences that can guide similar tasks."}}
!!! Please output only the JSON without any markdown formatting or code blocks. Output ONLY the "success_experience" field, no other fields needed.

H.8 Goal Experience Extraction Prompt for Habitat

Prompt for Goal Experience in EB-Habitat

You are a task analysis expert. Given a successful task execution trajectory with task_progress information, extract a SINGLE reusable workflow that captures the key actions that led to success.
Task Instruction
{instruction}
Successful Execution Trajectory (with Task Progress)
{trajectory}
Task Category
{task_category}
CRITICAL WORKFLOW EXTRACTION GUIDELINES
Focus on Progress-Boosting Actions
The trajectory includes "task_progress" after each step. Pay SPECIAL ATTENTION to:

1. Actions that INCREASED task_progress - These are the KEY ACTIONS that actually advanced the task
2. The sequence of actions leading to progress increases - This reveals the correct approach
3. Actions that maintained or prepared for progress - Supporting actions matter too

Workflow Requirements

1. Extract ONE concise, generalizable workflow - Not multiple workflows
2. Focus on the ACTION PATTERN, not specific objects - Use generic terms like "target object", "destination location"
3. Highlight the critical decision points - When to navigate vs. pick vs. place
4. Capture the logical order - The sequence matters for success
5. Be specific about SUCCESS CONDITIONS - What made this approach work

Abstraction Guidelines

- Replace specific object names (e.g., "apple", "table_42") with generic terms (e.g., "[TARGET_OBJECT]", "[DESTINATION]")
- Keep action types specific (navigate, pick, place, open, close)
- Preserve the causal relationships between actions

Please extract a SINGLE reusable workflow that captures the essence of this successful execution.
Output format:
{{"success_experience": "A complete workflow description that includes: (1) The goal pattern, (2) The key action sequence with EMPHASIS on progress-boosting actions, (3) Critical decision points, (4) Success conditions. Should be 3-6 sentences that can guide similar tasks."}}
!!! Please output only the JSON without any markdown formatting or code blocks. Output ONLY the "success_experience" field, no other fields needed.