Align While Search: Belief-Guided Exploratory Inference for Test-Time World Alignment

Anonymous Author(s)
Affiliation
Address
email

Abstract

We introduce a test-time adaptive agent that performs exploratory inference through posterior-guided belief refinement without relying on gradient-based updates or additional training for LLM agent search operation under partial observability. Our agent maintains a structured belief over the environment state, iteratively updates it via action-conditioned observations and selects actions by maximizing predicted information gain over the belief space. We estimate information gain using a lightweight LLM-based surrogate and assess world alignment through a novel reward that quantifies the consistency between posterior belief and ground-truth environment configuration. Experiments show that our method outperforms inference-time scaling baselines such as prompt-augmented or retrieval-enhanced LLMs, in aligning with latent world states with significantly lower integration overhead.

1 Introduction

Agents operating in partially-observable environments continuously encounter incomplete information in the course of achieving their goals [10, 9]. The key capability in such settings is exploratory decision making. That is, the agent should not only act to achieve the target objective but also to collect information that refines its internal belief about the world. The interaction between action, observation, and belief refinement forms the basis of effective behavior under uncertainty.

In this paper, we consider exploratory decision making under uncertainty, focusing on the task of object search in embodied settings, which is the basic task for more advanced tasks, and examine inference-time world understanding under such partial observability. While large language models (LLMs) have shown promise in zero-shot task execution [3, 17], their static reasoning often fails to adapt to the unfolding dynamics of environments with partial observability. Prior approaches to such partial observability include inference-time scaling methods (e.g., prompt tuning [12], retrieval augmentation [11, 22] and static prompting) and train-time policy optimization via supervised or reinforcement learning (e.g., SFT [18], DPO [21] and MCTS-style planners [34, 4]). However, the former lacks adaptive interaction with the environment, while the latter requires substantial training cost and limits deployment flexibility.

To overcome such drawbacks, we propose a novel, lightweight but effective agent architecture that performs *inference-time exploratory reasoning* through *posterior-guided belief refinement*. Our core idea is as follows: the agent maintains a structured posterior belief over object locations, updates it based on observations, and selects actions that reflect this evolving belief. Thus, our behavior policy is driven not by memorized patterns but by a dynamically refined understanding of the world. Crucially, all our adaptation occurs at test time without gradient updates, fine-tuning, or additional environment models. Empirical results show that our inference-time alignment strategy outperforms search-based and world-model baselines, while achieving substantially lower computational overhead by operating exclusively at test time. Our proposed method generalizes across diverse object types, room layouts and interaction histories without requiring task-specific training or reward-based tuning. To the best of our knowledge, this is the first test-time inference agent that performs Bayesian posterior reasoning

Submitted to the First Exploration in AI Today Workshop at ICML (EXAIT at ICML 2025). Do not distribute.

entirely via language; simulating observations, updating structured belief distribution, and selecting actions that optimize epistemic utility. Our contributions are summarized below:

- We propose a lightweight object search agent that performs test-time exploratory learning through posterior belief refinement, requiring no training or gradient updates.
- Our agent selects actions by dynamically aligning its behavior with evolving beliefs, enabling efficient object localization under partial observability.
- Experiments show that our method surpasses inference-time scaling baselines and competes with train-time world models.

2 Motivation

Modern language agents trained with supervised trajectories often overfit to specific search behaviors seen in expert demonstrations. To validate this, we ran an experiment in which an LLM search agent trained with supervised learning is given diverse search missions (Appendix C). When the agent is given search tasks in seen environments, the agent succeeds with high probability as seen in Figure 1(left). On the other hand, when the agent is given search tasks in unseen test-time environments in which the room layout is similar to the trained layout but object locations are different, the agent persistently reproduces train-time room visitation sequences even in these test-time environments with 84.8 (=52.7+31.8) % out of total search sequences with some deviations, as seen in Figure 1(right). Notably, over 50% of failures occur in the case of train-like search sequences, indicating that the agent is not adapting based on what it observes but performs blindly according to what it expects. This highlights a core limitation of static imitation agent: the inability to dynamically adjust when confronted with uncertainty or distribution shift. To address this limitation, in the following section we propose an inference-time agent named *Align While Search (AWS)* that performs belief-driven search, bridging exploration and alignment via *posterior-guided belief refinement*, enabling agents to adaptively deviate from training habits to align with the true environment through interaction.

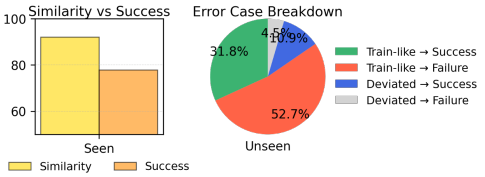

Figure 1: **Failures in Supervised Learning.** (**Left**) Similarity and success. (**Right**) Error case breakdown.

3 Proposed Method

Our inference-time aligning agent is inspired by Bayesian active learning in which the agent selects an action a that maximizes the posterior utility as follows:

$$a^* = \arg\max_{a \in \mathcal{A}} \mathbb{E}_{\hat{o} \sim p(\hat{o}|a)} \left[U\left(b_t, \hat{b}_{t+1}(b_t, a, \hat{o})\right) \right], \tag{1}$$

where $U(b_t, b_{t+1})$ is the posterior utility based on current belief b_t and updated belief b_{t+1} (a function of (b_t, a, o) with o being observation induced by a). Simply saying, the belief b_t can be the probability distribution of a target object over different locations. Our key idea is not to select action a to go to the location with the maximum probability in b_t as in conventional methods but to select action a to maximize the posterior utility considering both current b_t and updated b_{t+1} for more informative and robust action selection.To realize this idea, we exploit inference-time LLM-based belief simulation and posterior evaluation rather than relying on environment-tuned models or offline optimization, which enables our method to be plug-and-play compatible with existing world models.

3.1 Overview

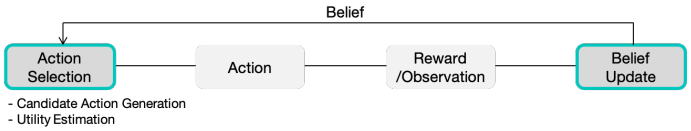

Figure 2: Inference-time loop of our agent

Our idea for accelerating alignment with the true world state and enabling fast search is to select an action that is expected to maximize the *information gain over a belief space*. Once alignment improves, the agent's belief becomes a reliable proxy for the environment, enabling accurate search purely based on beliefs. Thus, our method maintains and updates a hierarchical belief state over the

environment and selects the action by reasoning about the expected informativeness and alignment reward of each candidate action based on the maintained belief. Our agent runs an inference-time exploratory learning loop as shown in Figure 2, where two main operations of action selection and belief update alternate. This loop enables the agent to align with its environment and infer object locations solely depending on belief refinement without any offline fine-tuning. In the following sections, we detail each component: hierarchical belief design in §3.2, utility-based action selection and belief update rules in §3.3, and finally termination strategy in §3.4.

3.2 Hierarchical Belief Representation

We propose a structured belief system over the environment, represented as a hierarchical couple $(\mathcal{G}, \mathcal{S})$ corresponding to global- and object-level abstractions:

- $\mathcal{G}$: global hypotheses over user intent and scene layout, e.g.,

> "The user has a tendency to accumulate and display personal items, which may contribute to clutter: The presence of a houseplant, a statue, and a vase (previously observed) on the countertop and sinkbasin suggests that the user values aesthetics and may have a tendency to accumulate and display personal items. This could contribute to clutter and make it difficult for the user to maintain a tidy kitchen."

- $\mathcal{S}$: low object-level candidate locations, e.g.,

> mug: {cabinet: 0.145, drawer: 0.431, shelf: 0.425, countertop: 0.000, sinkbasin: 0.000, coffeemachine: 0.001, fridge: 0.001, garbagecan: 0.000, microwave: 0.000, stoveburner: 0.000, toaster: 0.000}

Note that the belief in $\mathcal{G}$ is stored as language, whereas the belief in $\mathcal{S}$ is stored as a probability distribution, as seen in the above examples. Concretely, the belief $b^{\mathcal{S}}$ in $\mathcal{S}$ is modeled as a categorical distribution over symbolic locations:

$$b^{\mathcal{S}}(k) = Pr(\text{target is located at } k), \ \ \forall k \in \mathcal{L}^{\mathcal{S}}, \tag{2}$$

where $\mathcal{L}^{\mathcal{S}}$ is the set of all symbolic locations of $\mathcal{S}$.

3.3 Exploratory Action Selection via Expected Utility

At each step, the agent must decide which unexplored symbolic location to visit next. We formulate this as an inference-time decision-making process that optimizes an expected utility function, balancing information gain and alignment.

Information Gain as Utility via LLM-Based Simulation If the belief $b^{\mathcal{S}}$ is perfect and sufficient statistic, the optimal action is to visit the location with maximum probability in $b^{\mathcal{S}}$ due to *maximum a posteriori* decision principle. Under partial observability and LLM-based operation, however, the belief is not perfect. To overcome this imperfectness and devise a robust action selection rule without additional training or gradient-based test-time tuning, we adopt the following parameter-free action selection method based on LLM-based simulation. We first take top K_a actions from current object-level belief $b^{\mathcal{S}}$ as candidate actions. For each candidate action a, we use an LLM-based observation predictor π_{pred} to sample K_o plausible observations $\{\hat{o}\}$ conditioned on the current belief and candidate action a. Then, we apply belief update to get an updated belief $b^{\mathcal{S}'}$ based on current belief, candidate action and predicted observation by using a belief updator BU, which will be explained shortly. Then, we define the step-wise reward as the information gain of candidate action, i.e., *expected reduction in entropy of the object-level belief in $\mathcal{S}$ after simulated belief update with predicted observation*, given by

$$\text{IG}(a) = \mathbb{E}_{\hat{o}} \left[H(b^{\mathcal{S}}) - H(b^{\mathcal{S}'} \mid \hat{o}) \right], \tag{3}$$

where $H(\cdot)$ denotes entropy over locations, $b^{\mathcal{S}}$ is the current object-level belief, and $b^{\mathcal{S}'}$ is the belief after a simulated update with predicted observation $\hat{o} \sim \pi_{pred}(\hat{o} \mid B, a)$ with current full belief B and candidate action a. Then, we choose the action from the candidate action set that yields the maximum information gain. Note that the object being found is equivalent to an impulse distribution $b^{\mathcal{S}}$. Hence, maximum entropy reduction at each step implies fastest search.

Belief Update Current full belief B on $(\mathcal{G}, \mathcal{S})$ denoted as $B = (B^{\mathcal{G}}, B^{\mathcal{S}})$ should be updated after observation o induced by action a. For this, we adopt the following two-step approach considering the linguistic nature of $B^{\mathcal{G}}$ and numeric nature of $B^{\mathcal{S}}$:

$$B^{\mathcal{G}} \xrightarrow{\pi^{g}_{BU},\, o} B^{\mathcal{G}'} \rightarrow B^{\mathcal{S}'}. \tag{4}$$

That is, we first update $B^{\mathcal{G}}$ to $B^{\mathcal{G}'}$ based on an LLM-based updator π^{g}_{BU} based on observation o. Then, from the update global belief $B^{\mathcal{G}'}$ we extract the updated object-level belief $B^{\mathcal{S}'}$ using one of the following two methods (see Appendix E.3 for details):

- **Similarity-Based Scaling:** Adjust belief scores using lexical similarity between candidate symbols and the updated global hypothesis.
- **LLM-Based Prior Scaling:** Use an LLM π^{s}_{BU} to predict which locations should be boosted or suppressed given a global hypothesis.

The above two variants yield different belief structures over time-ranging from localized refinement to semantic jumps, ultimately driving distinct exploration paths and belief shaping. The different belief-shaping behavior between the two methods will be discussed further in Section 4.6.

3.4 Instance-Level Grounding and Termination

After selecting a symbolic location (e.g., `cabinet`), the agent samples one unexplored instance (e.g., `cabinet3`) uniformly from the set of candidates associated with that symbol, and executes the corresponding action. If the instance is not directly observable (e.g., closed), an open action is issued beforehand. The search loop terminates either when the target object is found or when the average alignment score over the past k steps exceeds a predefined threshold.

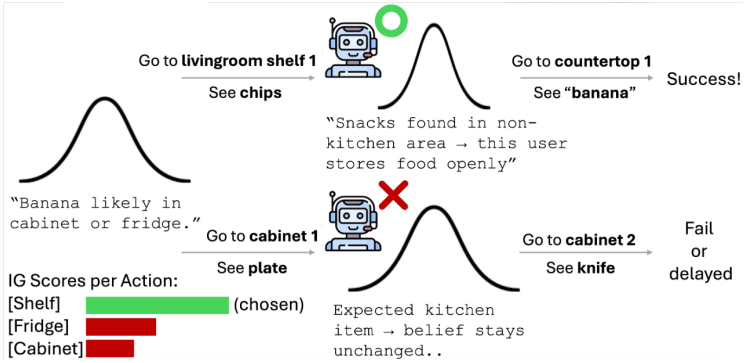

Figure 3: Overview of our Align-While-Search agent

Our strategy extends Bayesian active learning to symbolic domains by replacing feature-space uncertainty with semantic hypotheses (e.g., room-object mappings, user intent cues). Figure 3 illustrates our search agent. The agent begins with an initial belief composed of both semantic $\mathcal{G}$ and numeric $\mathcal{S}$, interacts with the environment to refine it through search and observation, and iteratively aligns its belief to the true environment. The entropy of the belief decreases while alignment improves over time, enabling effective inference-time adaptation.

4 Experiments

We evaluate the effectiveness of our belief-guided object search agent in simulated environments under partial observability. Our experiments are designed to answer three key questions:

- Does inference-time belief refinement improve search efficiency compared to static or prior-only baselines?
- Is the predicted information gain (IG) aligned with actual improvements in belief accuracy?
- What factors affect the success or failure of belief-based search?

4.1 Experimental Settings

Environments We evaluate our method on **ALFWorld** [26], a household object search benchmark that requires goal-directed navigation and interaction. We follow Song et al. [28] for trajectory construction and supervised training. Full dataset and environment details are provided in Appendix H.
Evaluation Metrics We report the *Success Rate* (SR(%)) as our primary metric. Two additional analysis metrics, *Average Steps* and *Final Belief Alignment*, are discussed in Section 4.3.
Baselines We compare our method (AWS) against three categories of baselines: (1) **Training-time world model** baselines, including ETO [28], WKM [20], and MPO [33]; (2) **Inference-time scaling** baseline, such as RAFA [14] and (3) **Preference alignment** baselines such as IPR [32], SteCa [30] (evaluated in Appendix I). Reproduction and implementation details are in Appendix G.

Paradigm	Method	ALFWorld Seen	ALFWorld Unseen	Average
Scaling-only LLM	*GPT family*			
	GPT-4 [1]	42.9	38.1	40.5
	GPT-4 + RAFA [14]	46.0	39.5	42.7 (+5.43%)
	GPT-4 + AWS (Ours)	**78.7**	**91.0**	**84.8** (+109%)
	LLaMA family			
	LLaMA-3.1-70B [5]	72.8	75.3	74.0
	LLaMA-3.1-70B + RAFA [14]	50.0	41.1	45.5 (-38.5%)
	LLaMA-3.1-70B + AWS (Ours)	**77.1**	**78.3**	**77.7** (+5.00%)
Fine-tuned World Model	*LLaMA family*			
	LLaMA-3.1-8B + SFT [36]	79.3	71.6	75.4
	LLaMA-3.1-8B + ETO [28]	77.1	76.4	76.7 (+1.73%)
	LLaMA-3.1-8B + WKM [20]	77.1	78.2	77.6 (+2.92%)
	LLaMA-3.1-8B + MPO [33]	80.7	81.3	81.0 (+7.43%)
	LLaMA-3.1-8B + AWS (Ours)	**87.5**	**85.3**	**86.4** (+14.6%)

(a) Comparison with training-time and inference-time LLM agents on ALFWorld. (**Bold** indicates the best in each column; shaded rows show our best variants.)

Base Model	Method	ALFWorld (Unseen)
(A) Fine-tuned Models		
LLaMA-3.1-8B [36]	Vanilla	81.3
	LLM-Scaling	88.8 (+9.20%)
	Similarity-Scaling	**91.7** (+12.8%)
(B) Instruction-tuned Models		
LLaMA-3.1-8B (base model)	Vanilla	46.2
	LLM-Scaling	52.9 (+14.5%)
	Similarity-Scaling	**67.9** (+46.8%)
LLaMA-3.1-70B (base model)	Vanilla	85.0
	LLM-Scaling	91.0 (+7.06%)
	Similarity-Scaling	**94.0** (+10.6%)
GPT-4o-mini (base model)	Vanilla	76.8
	LLM-Scaling	86.6 (+12.7%)
	Similarity-Scaling	**89.5** (+16.5%)
GPT-4 (base model)	Vanilla	**93.2**
	LLM-Scaling	92.0 (-1.29%)
	Similarity-Scaling	92.5 (-0.75%)

(b) Comparison of scaling strategies across base models on ALFWorld (MPO-inferred [33]).

Table 1: **Comparison of Performance on ALFWorld**: across different agent types **(a)** and scaling variants for each base model **(b)**.

Model Backbones Note that AWS requires π_{pred}, π^{g}_{BU} and π^{s}_{BU} and use the same LLM for all three with different prompting. We evaluate AWS using both open-source and proprietary LLMs: LLaMA2-7B [5], LLaMA3.1-8B [5], LLaMA3.1-70B [5], GPT-4o-mini [8], and GPT-4 [8]. Model usage and performance variations are detailed in Appendix H.

4.2 Main Results

We evaluate our method on ALFWorld, comparing against both inference-time scaling and train-time world model baselines. As shown in Table 1(a), our method AWS achieves the highest average performance across seen and unseen splits, outperforming both scaling-only LLMs and fine-tuned agents. Compared to prior world models such as WKM [20] and MPO [33], our method achieves comparable or better accuracy while avoiding training overhead. Notably, on ALFWorld unseen tasks, our method reaches 85.3% success rates, surpassing all fine-tuned baselines. Table 1(b) summarizes the performance of our two scaling strategies in Section 3.3, both of which yield comparable or consistent gains across diverse backbone LLMs-including LLaMA-3.1-8B-SFT(+12.8%), LLaMA-3.1-8B(+46.8%), LLaMA-3.1-70B(+10.6%), and GPT-4o-mini(+16.5%) on both fine-tuned and instruction-tuned models. Notably, our method synergizes with alternative world models such as MPO [33], achieving new state-of-the-art results of 94.0% on ALFWorld (Unseen). These results highlight that posterior-guided exploratory reasoning enables efficient inference-time adaptation and scalable deployment across environments and models.

4.3 Ablations

To understand the contribution of individual components, we conduct controlled ablations and strategy replacements. Table 2 reports the success rate and efficiency of alternative search strategies. We test four key variants: (1) random unexplored action(random), (2) no belief updates with greedy selection (fixed prior), (3) using a greedy action selection while updating beliefs (greedy), and (4) using an MCTS-style action selection without IG estimation (MCTS).

Our results show that belief updates are essential for maintaining alignment over time-removing them leads to significant drops in success and increases in episode length. Notably, belief updates alone do not yield gains (Flat Prior→Greedy), but becomes evident when coupled with structured exploration. Similarly, removing IG (Ours→MCTS) results in inefficient exploration, confirming that our lightweight IG predictor provides a strong signal. Together, these results highlight that posterior-guided exploration is most effective when belief updates, IG estimation, and structured search operate in concert, enabling accurate alignment inference-time behavior.

Method	Prior Update	IG	MCTS	SR (%)	Steps (↓)
Random Search	×	×	×	×	74.6	19.8
Flat Prior Search	✓	×	×	×	82.8	14.5
Greedy (No IG)	✓	✓	×	×	82.4	**11.7**
MCTS (No IG)	✓	✓	×	✓	<u>85.0</u>	14.7
Ours	✓	✓	✓	✓	**87.4**	<u>13.8</u>

Table 2: **Ablation Results.** Removing either belief update or IG reduces alignment and success rate, confirming both are essential for inference-time performance (best in **bold**, second best in <u>underline</u>).

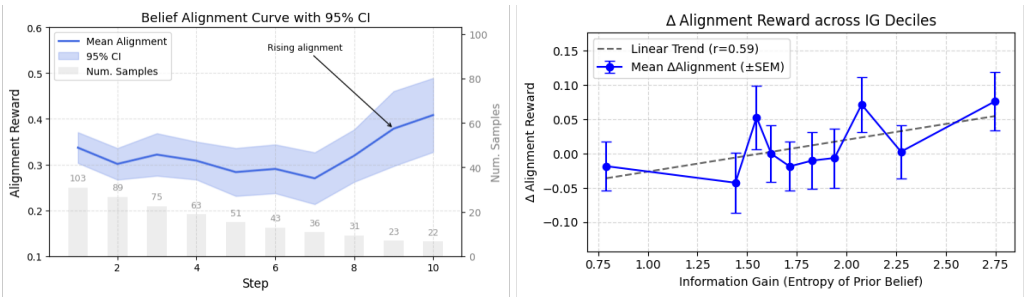

Figure 4: **Belief Alignment Analysis.** (**Left**) We plot belief alignment scores over time by tracking the agent's belief in the correct object location ($b_t(y^*)$) across interaction steps. Our method shows a notable rise in alignment during later stages, highlighting the benefit of inference-time refinement. Shaded regions indicate 95% confidence intervals. (**Right**) We group predicted information gain scores into deciles and measure the corresponding change in belief alignment (ΔR^{align}). Higher IG deciles consistently lead to larger improvements, validating the utility of IG-based action ranking.

We also compare our method against alignment-tuned agents (Table 4), such as IPR and SteCa in Appendix I. Our method achieves the best performance on unseen tasks, compared to both supervised and reinforcement-based preference tuning methods. This shows that our belief-guided adaptation offers good generalization compared to fixed alignment policies.

4.4 Analysis: Belif Alignment Curve (Alignment vs. Step)

To evaluate how effectively the agent refines its belief during exploration, we measure an alignment reward that reflects how well the predicted observation $\hat{o}_a$ matches the actual environment observation o_a after actually executing action a. Specifically, we define $\text{Align}(a) = \text{similarity}(\hat{o}_a, o_a)$, where $\hat{o}_a = \pi_{\text{pred}}(B, a)$ and the $\text{similarity}(\cdot, \cdot)$ function is described in Appendix E. Then we plot the alignment trajectory, i.e., the belief mass assigned to the ground-truth object location $b_t(y^*)$, over interaction steps. As shown in Figure 4(left), our method initially maintains moderate alignment, but exhibits a clear rising trend in the later steps, demonstrating that belief-guided exploration accumulates useful information over time. While early steps (1–6) show limited gains due to the agent's need to disambiguate possible hypotheses, steps 7–10 demonstrate consistent alignment improvement, even under high variance from reduced sample counts. The upward shift in the alignment curve, coupled with tight confidence intervals in the early stages, indicates that our agent learns to prioritize informative actions. Overall, this trajectory supports our hypothesis that inference-time belief refinement enables robust adaptation, particularly in underexplored or ambiguous environments.

4.5 Factors on Success of Belief-based Search

Ordinal IG vs. ΔR Alignment Consistency To evaluate whether LLM-predicted IG scores are reliable indicators of belief refinement, we analyze the correlation between predicted IG rankings and actual changes in belief alignment, defined as $\Delta R^{\text{align}} = R(b_{t+1}) - R(b_t)$, where $R(b)$ denotes the alignment reward (i.e., belief mass on the true location). We group predicted IG scores into deciles and compute the average ΔR^{align} per bin. As shown in Figure 4(right), higher IG bins are associated with greater alignment gains, forming a broadly monotonic trend. While per-step predictions are noisy, the global ordinal structure remains consistent-particularly in successful episodes where exploration is well-executed. These results validate the effectiveness of ordinal-scale LLM reasoning as a lightweight yet reliable proxy for belief-shifting action selection.

Exploration Value of Information Gain To evaluate the effectiveness of information gain as a guide for exploration, we analyze

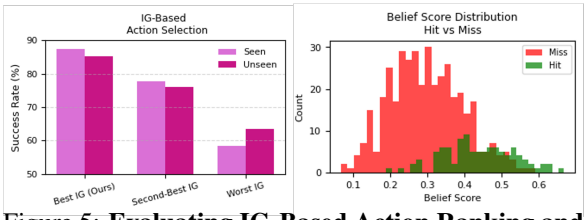

Figure 5: **Evaluating IG-Based Action Ranking and Belief Accuracy. (Left)** Success rate under different IG-based selection strategies. Performance consistently degrades when the selected action deviates from the best IG candidate, confirming the utility of IG in action ranking. **(Right)** Distribution of belief scores at visited locations, grouped by whether the target object was present (hit) or absent (miss). Higher belief mass aligns with true object presence, validating the belief prior as a reliable proxy for environmental structure.

how the choice of action-based on IG ranking-impacts downstream task success. Rather than relying on the absolute value of IG, we test whether selecting the top-ranked IG candidate among available actions leads to better outcomes. As shown in Figure 5(left), selecting the top-IG action yields the highest success rate (87% on seen and 85.3% on unseen environments). Performance drops when choosing the second-best IG candidate (77.85% / 76.31%) and degrades further with the lowest-IG choice (58.33% / 63.38%). These results indicate that IG serves as an effective relative ranking signal for epistemic utility, enabling the agent to prioritize informative actions that ultimately improve task performance.

Belief Prior vs. Ground Truth Matching. To evaluate how well the agent's belief prior aligns with actual object presence, we analyze the belief scores assigned to visited locations and compare them against whether the target object was found (*hit*) or not (*miss*).

For each exploration step, we extract the belief probability assigned to the visited receptacle and label the outcome based on object presence. As shown in Figure 5(right), hit locations receive substantially higher belief scores than miss locations. Specifically, the average belief score for hit cases was 0.452 (std = 0.133, $n = 117$), while for miss cases it was 0.314 (std = 0.143, $n = 555$). This difference is highly significant, with a t-test yielding $t = 10.06$ and $p = 3.98 \times 10^{-19}$. These results confirm that belief priors provide meaningful guidance for search, effectively distinguishing promising targets from distractors. Rather than serving as a passive estimate, the belief prior plays an active role in structuring efficient exploration, even under partial observability.

4.6 Case Study

To better understand how our belief-guided search enables effective exploration, we conduct a case study comparing two variants, similarity-based and LLM-based belief updates, on the same task. As shown in Figure 6, both variants successfully discover the target object, but they exhibit distinct belief dynamics and exploration strategies. The similarity-based agent gradually increases its cabinet belief through localized refinement (Figure 6a), maintaining high entropy early on before committing to high-probability regions (Figure 6c; *blue*). In contrast, the LLM-based agent exhibits a sharper belief shift (Figure 6b) and faster entropy reduction by leveraging global cues extracted from past object co-occurrence or spatial semantics (Figure 6c; *yellow*). These belief dynamics translate to different search paths (Figure 6d): the similarity-based agent narrows down its focus through nearby locations, while the LLM-based agent performs semantically informed jumps to higher-yield receptacles early in the episode. In Appendix Table 5, we further compare the step-by-step hypotheses evolution of both agents, highlighting how their reasoning structures emerge over time.

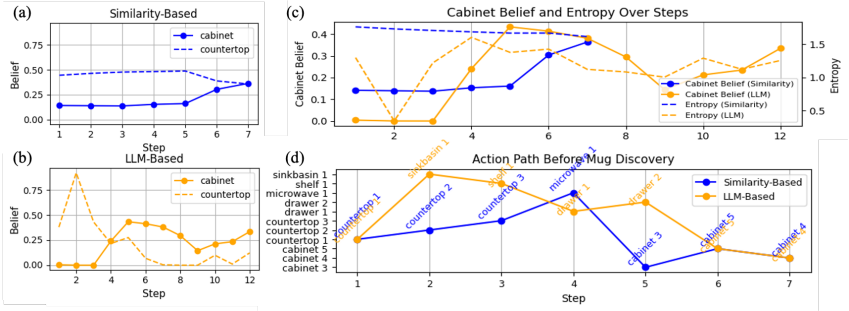

Figure 6: **Belief Update and Search Behavior Comparison.** (a) Similarity-based belief scaling: step-wise belief dynamics (b) LLM-based scaling: step-wise belief dynamics (c) For Similarity-based scaling, belief changes gradually while for LLM-based scaling, belief dynamics and cabinet-related entropy show more focused belief shaping. (d) The LLM agent explores semantically distant locations early, whereas the similarity-based agent refines search inlocalized clusters.

5 Conclusion and Future Work

We have presented a test-time adaptive agent that refines its belief through exploratory reasoning to align actions with evolving understanding of the world. Without training or fine-tuning, our method improves localization efficiency by selecting actions that maximize belief alignment. Beyond search, this framework offers a scalable foundation for inference-time adaptation.

As future work, we plan to extend our framework to other domains such as grid-based object interaction environments and interactive fiction benchmarks. Also, we plan to integrate procedural rule reasoning and recovery, enabling broader alignment across spatial and semantic domains.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Michael Ahn, Anthony Brohan, and et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Robotics: Science and Systems (RSS)*, 2022.

[3] Tom Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. *NeurIPS*, 2020.

[4] Luyu Gao et al. Pal: Program-aided language models. In *ICLR*, 2022.

[5] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[6] Andy Huang et al. Language models as agents. *arXiv preprint arXiv:2207.04612*, 2022.

[7] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[8] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[9] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. In *ICML*, 2018.

[10] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[11] Patrick Lewis, Ethan Perez, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*, 2020.

[12] Xinya Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*, 2021.

[13] Yafu Li, Xuyang Hu, Xiaoye Qu, Linjie Li, and Yu Cheng. Test-time preference optimization: On-the-fly alignment via iterative textual feedback. *arXiv preprint arXiv:2501.12895*, 2025.

[14] Zhihan Liu, Hao Hu, Shenao Zhang, Hongyi Guo, Shuqi Ke, Boyi Liu, and Zhaoran Wang. Reason for future, act for now: A principled framework for autonomous llm agents with provable sample efficiency. *arXiv preprint arXiv:2309.17382*, 2023.

[15] Ethan Mendes and Alan Ritter. Language models can self-improve at state-value estimation for better search. *arXiv preprint arXiv:2503.02878*, 2025.

[16] Seonghyeon Oh et al. Reward-specification-free policy adaptation via entropy alignment. *arXiv preprint arXiv:2306.11289*, 2023.

[17] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[18] Long Ouyang, Jeffrey Wu, et al. Training language models to follow instructions with human feedback. *NeurIPS*, 2022.

[19] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

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

[21] Ramin Rafailov et al. Direct preference optimization: Your language model is secretly a reward model. *ICLR*, 2023.

[22] Omer Ram et al. Contextual compression: Context compression improves multi-hop qa with open-domain retrieval. *ICLR*, 2023.

[23] Jürgen Schmidhuber. Curious model-building control systems. In *Proc. international joint conference on neural networks*, pages 1458–1463, 1991.

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

[25] Noah Shinn, Ke Hou, and et al. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

[26] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

[27] Gautam Singh, Skand Peri, Junghyun Kim, Hyunseok Kim, and Sungjin Ahn. Structured world belief for reinforcement learning in pomdp. In *International Conference on Machine Learning*, pages 9744–9755. PMLR, 2021.

[28] Yifan Song, Xiang Yue Da Yin, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization for llm agents, 2024. *URL https://arxiv.org/abs/2403.02502*.

[29] Qianru Sun, Yitong Shi, et al. Test-time training with self-supervision for generalization under distribution shifts. *ICML*, 2020.

[30] Mengqi Wang et al. Steca: Structured textual caching for long-horizon language agents. *arXiv preprint arXiv:2402.13330*, 2024.

[31] Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023.

[32] Charles Xiong et al. Ipr: Improving planning robustness of language agents with in-context action rewrites. *arXiv preprint arXiv:2402.08094*, 2024.

[33] Weimin Xiong, Yifan Song, Qingxiu Dong, Bingchan Zhao, Feifan Song, Xun Wang, and Sujian Li. Mpo: Boosting llm agents with meta plan optimization. *arXiv preprint arXiv:2503.02682*, 2025.

[34] Shinn Yao et al. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

[35] Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

[36] Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. *arXiv preprint arXiv:2310.12823*, 2023.

[37] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

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

[39] Yuke Zhu et al. Visual semantic planning using deep successor representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

A Notations

Notation Glossary

- $\mathcal{B}_t$: Belief state at timestep t, composed of $(\mathcal{G}_t, \mathcal{R}_t, \mathcal{S}_t)$
- $a \in \mathcal{A}_t$: Candidate symbolic action (e.g., "go to shelf") at time t
- $\hat{o}$: Predicted observation under action a, sampled from $P(o \mid a, \mathcal{B}_t)$
- $u(\mathcal{B}_t, \hat{o})$: Utility of the predicted observation $\hat{o}$ when incorporated into the belief state
- IG_a: Information gain induced by taking action a and observing $\hat{o}$
- Align_a: Alignment reward computed by comparing $\hat{o}$ to actual environment observation
- λ: Weighting factor to trade off information utility and alignment score
- $H(\cdot)$: Shannon entropy of a probability distribution
- $\mathcal{G}$: Global belief consisting of high-level language hypotheses
- $\mathcal{S}_s$: Subject-level belief distribution over locations for target object s
- $\mathcal{M}_t$: Set of object mentions parsed from observation o at time t
- $\text{sim}(m, h)$: Similarity function between parsed object mention m and hypothesis h
- $\text{symb}(a)$: Symbolic keyword extracted from action a (e.g., "cabinet" from "go to cabinet 2")
- π_{pred}: Observation prediction model
- π^g_{BU}, π^s_{BU}: Belief update model for global(g) and object(s)-level beliefs

B Related Works

LLM Agents via Prompting and Tuning. Language models have been adapted for decision-making through prompting-based agents [2, 6] and instruction tuning [18, 37]. Preference optimization methods like DPO [21] and RFT [35] improve alignment but remain static at test time. RAFA [14] augments LLMs via retrieval and sketch-based prompting but lacks belief refinement or uncertainty modeling. Other works-Reflexion [25], ETO [28], IPR [32]-train agents with feedback or trajectory memory. In the same vein, [15]. In contrast, our method performs no tuning or memory learning. It enables inference-time adaptation purely via posterior-guided belief refinement.

Symbolic Memory and World Models. Grounded agents often use symbolic memory [34], knowledge graphs [7], or learned world models [39]. These typically rely on static environments or pretraining. Meanwhile, Zhou et al. [38] proposes augmenting LLM agents with rule-based world alignment by explicitly learning environment-specific preconditions and effects, enabling improved planning via model-predictive control; however, their method requires repeated exploration with execution-time verification and external rule storage. Likewise, Singh et al. [27] introduces a probabilistic object-centric world model that maintains belief over latent object states via sequential Monte Carlo sampling, but assumes structured environment priors and is trained offline with oracle supervision. We maintain a structured belief over symbolic locations, updating it online through LLM-based posterior reasoning without external memory or rewards.

Exploration and Inference-Time Scaling. Exploration has been studied through intrinsic motivation [19], Bayesian surprise [23], and uncertainty-driven strategies. Recent test-time adaptation methods [29, 16] update models without retraining, but often lack structured belief modeling. TPO [13] adapts behavior through rollout-based optimization, but incurs high computational cost due to sampling and gradient updates. RAFA [14] is the only inference-time scaling baseline that combines LLM-based agentic planning with Bayesian adaptive control. It formulates interaction as a loop of plan-then-act, using past experience to reason over future value and select the next best action without fine-tuning. However, RAFA does not maintain structured belief or represent uncertainty over symbolic hypotheses, limiting its ability to model latent world dynamics. In contrast, our method performs symbolic belief refinement via posterior updates, leveraging LLMs to reason over task-relevant hypotheses and actively align with environment rules at inference time.

C Motivation: Experimental Detail

Objective. This experiment evaluates whether SFT agents reuse train-time search behaviors at test time and whether such behavior leads to successful task completion. We analyze both the degree of behavioral copying and its effectiveness across environments that differ from training.

Setup. We define two evaluation sets:

- **Seen Set (Same Room)**: Rooms and object locations identical to training.
- **Unseen Set (Similar Room)**: Rooms with similar layouts but different object placements.

Each trajectory begins with a natural language room description followed by a sequence of actions. From each trajectory, we extract a **search sequence**, defined as a maximal consecutive chain of `go to` actions. We then compute:

- **Room similarity**: Cosine similarity between the test-time room description and each training room description, using sentence embeddings (e.g., all-MiniLM-L6-v2).
- **Trajectory similarity**: Sequence similarity (via normalized edit distance) between the test search sequence and the most similar training trajectory from rooms with matching descriptions.

We retain only test trajectories with high room similarity (≥ 0.9) to isolate cases where the room appears similar, but object locations may differ.

Case Classification. We categorize each test trajectory based on (1) its trajectory similarity and (2) success outcome:

- **Train-like → Success**: Trajectory similarity ≥ 0.9, task succeeded.
- **Train-like → Failure**: Trajectory similarity ≥ 0.9, task failed.
- **Deviated → Success**: Trajectory similarity < 0.9, task succeeded.
- **Deviated → Failure**: Trajectory similarity < 0.9, task failed.

D Method: Definitions

Global Belief ($\mathcal{G}$). The global belief captures abstract hypotheses about how the environment is structured or how a user organizes objects in the household. Formally, $\mathcal{G} = g_1, g_2, g_3$ is a set of natural language statements generated by an LLM at the beginning of the episode. These statements reflect user-specific organizational tendencies (e.g., "kitchen is well-organized" or "coffee-related items are grouped near the counter") and serve as a latent prior that conditions downstream inference and action selection.

Subject Belief ($\mathcal{S}$). Given a search target s, the subject belief $\mathcal{S}_s$ is represented as a normalized categorical distribution over all symbolic locations (e.g., cabinet", drawer") and their instances (e.g., "cabinet3") in the environment. Each score $p(l \mid s)$ reflects the agent's current estimate of the likelihood that object s resides at location l. Subject priors are initialized from room beliefs and are progressively refined through both simulated and actual observations as the agent explores.

E Method: Implementation

E.1 Belief Update Pipeline

At each timestep t, the agent updates its belief state hierarchically, following the top-down structure of $(\mathcal{G}, \mathcal{S})$. These updates can occur in two modes: (1) *simulated updates* for evaluating hypothetical actions using predicted observations, and (2) *actual updates* after executing an action and receiving a true environment observation.

Step 1: Global Belief Update. Given an observation o (real or predicted), the agent invokes a language model to revise its global belief $\mathcal{G}$. This involves generating a new set of high-level hypotheses that reinterpret how the environment may be structured given o. This update enables adaptive abstraction, allowing the agent to reinterpret user-specific tendencies over time.

Step 2: Subject Belief Update. The subject-level belief $\mathcal{S}_s$ is refined based on the updated room scores. For a given subject s, the location prior $p(l \mid s)$ is adjusted such that locations associated with highly scored rooms receive a proportional boost. If $l \in r$, then:

$$p(l \mid s) \propto p(l \mid s) \cdot p(r)$$

This propagation ensures that high-level beliefs about the environment ultimately influence fine-grained search behavior.

This belief update process provides a flexible mechanism for belief propagation across abstraction levels, and serves as the foundation for estimating information gain and alignment in the action selection process.

E.2 Common Module

Global Hypothesis Initialization To bootstrap the agent's environment understanding, we use the following prompt to initialize global-level hypotheses:

Prompt:
You are helping an agent search for the object `"mug"` in a household environment.
Generate 3 plausible hypotheses about how the user organizes their space or uses the room.
Each hypothesis should describe user habits or room-level behaviors (e.g., "kitchen is well-organized", "cabinets contain cups").
Return as a numbered list.

This prompt is passed to the LLM once per episode. The result is used to instantiate a `GlobalBelief` object, which stores a list of hypotheses guiding subsequent inference.

Hypothesis Update from Observation The global hypotheses are revised based on new environment observations:

Prompt: Global Hypothesis Update
You previously assumed the following about the user:
```
1. kitchen is well-organized
2. cabinets contain cups
3. mugs are often in the sink
```

Now the agent observed: `"the sink is empty"`
Based on this new information, revise or expand the global hypotheses.
Return 3 updated hypotheses as a numbered list.

The updated hypotheses are used to refine prior scores or generate new priors.

E.3 Variants in Belief Adjustment

We experiment with two inference-time belief adjustment variants, both operating on top of the shared global belief.

LLM-Based Prior Scaling Given a global hypothesis (e.g., "cabinets contain cups") and the known list of room-level symbols (e.g., `["cabinet", "drawer", "sink"]`), we use the following prompt:

Prompt: LLM-Based Prior Scaling
Given the hypothesis: `"cabinets contain cups"`
Among the locations `["cabinet", "drawer", "sink"]`, decide which to boost or suppress.
Return only a JSON object with this format:
```
{"boost": [...], "suppress": [...]}
```

The returned result adjusts room scores: boosting adds +1.0, suppressing applies a 0.5 multiplier. The scores are then normalized. This process is repeated for each global hypothesis. Optionally, we log the decisions for interoperability.

Similarity-Based Prior Scaling. Upon receiving an environment observation $o \in \mathcal{O}_t$ after taking action $a \in \mathcal{A}_t$, we refine the room-level belief $\mathcal{R}_t$ by comparing observed object mentions against each room's symbolic hypotheses.

Let $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ denote the set of rooms, and $\mathcal{H}_r$ the set of symbolic hypotheses associated with room r. Given the extracted set of object mentions $\mathcal{M}_t = \{m_1, \ldots, m_k\}$ from the observation o, we compute the adjustment score s_r for each room $r \in \mathcal{R}$ as:

$$s_r = \sum_{m \in \mathcal{M}_t} \sum_{h \in \mathcal{H}_r} \text{sim}(m, h) + \delta_r(a)$$

where $\text{sim}(m, h) \in [0, 1]$ is a similarity function (e.g., cosine similarity), and the symbolic match bonus $\delta_r(a)$ is defined as:

$$\delta_r(a) = \begin{cases} 0.2 & \text{if symb}(a) \in r \\ 0 & \text{otherwise} \end{cases}$$

Room scores are then updated additively:

$$\text{score}_t(r) \leftarrow \text{score}_{t-1}(r) + s_r$$

Finally, scores are normalized to form a belief distribution:

$$\text{score}_t(r) \leftarrow \frac{\text{score}_t(r)}{\sum_{r' \in \mathcal{R}} \text{score}_t(r')}$$

This mechanism allows the agent to update $\mathcal{R}_t$ using soft semantic evidence derived from symbolic observations, without needing predefined rules.

E.4 Implementation Details

Symbolic Location Selection. The agent ranks top-k unexplored symbols from the belief prior (k=3), estimates information gain for each, and selects the best action.

Observation Sampling. For each action, we sample K=3 predicted observations using a frozen LLM (temperature 0.7), each containing 2–4 likely objects.

Belief Update. Upon observing o after action a, the agent updates its belief via a similarity-weighted rule:

$$b'^{\ell}(l) \propto b^{\ell}(l) \cdot \phi(o, l, a)$$

where $\phi(o, l, a)$ approximates $P(o \mid l, a)$ using model predictions or observation similarity. The belief hierarchy is updated recursively across levels. Belief is updated over normalized categorical priors via similarity- or LLM-based scaling.

Reward Computation. Alignment is computed as set precision and averaged over recent steps for convergence detection. We use a set-overlap score for similarity:

$$\text{Align}(a) = \frac{|\texttt{Predicted}(a) \cap \texttt{Actual}(a)|}{|\texttt{Actual}(a)|}$$

When used in forward planning, we approximate expected alignment via:

$$\mathbb{E}_{\hat{o} \sim \pi_{\text{pred}}(B, a)} \left[\text{sim}(\hat{o}, o)\right]$$

where o denotes a sampled candidate outcome. This allows us to estimate how well an action aligns with the latent world model under the current belief.

Termination. The loop ends when the target is found or when average alignment over the past 3 steps exceeds 0.75.

F Method: Algorithms

F.1 Full Algorithm

Algorithm 1: Inference-Time Belief Alignment with Multi-Level Update

Input: Target subject s, environment $\mathcal{E}$
Initial beliefs: Global B_0^g, Subject B_0^s
Predictor π_{pred}, LLM π_{LLM}
Parameters: max steps T, sample count N, top-k actions, threshold θ_{align}, weight λ
Output: Final belief state, alignment log, found/not-found status
Initialize tracker, set $t \leftarrow 0$
for $t = 1$ **to** T **do**
  $A^c \leftarrow$ Top-k unexplored symbols from B_t^s `// Candidate actions`
  foreach $a \in A^c$ **do**
    IGList $\leftarrow [\,]$
    for $i = 1$ **to** N **do**
      $\hat{o}_a^{(i)} \leftarrow \pi_{\text{pred}}(B_t^r, a)$ `// Predict observation`
      $B^{g'} \leftarrow B_t^g$.`update`$(\hat{o}_a^{(i)}, \pi_{\text{LLM}})$
      $B^{s'} \leftarrow B_t^s$.`adjust_from_global`$(B^{g'})$
      IGList.`append`$(H(B_t^s) - H(B^{s'}))$
    end
    $\mathbb{E}[\text{IG}](a) \leftarrow$ mean(IGList)
  end
  $a^* \leftarrow \arg\max_{a \in A^c} \mathbb{E}[\text{IG}](a)$
  $x^* \leftarrow$ `SelectInstance`(a^*)
  $o_t \leftarrow \mathcal{E}$.`step`$(x^*)$
  Log: tracker.`log_action`(x^*, o_t)
  $B_{t+1}^g \leftarrow B_t^g$.`update`$(o_t, \pi_{\text{LLM}})$
  $B_{t+1}^s \leftarrow B_t^s$.`adjust_from_global`$(B_{t+1}^g)$
  B_{t+1}^s.`update_with_observation`(s, x^*, o_t)
  $\hat{o}_{\text{align}} \leftarrow \pi_{\text{pred}}(B_{t+1}^r, x^*)$
  $r_{\text{align}} \leftarrow$ `AlignScore`$(\hat{o}_{\text{align}}, o_t)$
  tracker.`log_alignment`$(x^*, o_t, \hat{o}_{\text{align}}, r_{\text{align}})$
  if $r_{align} \geq \theta_{align}$ ***or*** $s \in o_t$ **then**
    return FOUND, tracker
  end
end
return NOT FOUND, tracker

F.2 Algorithm for LLM-based Observation Sampling

Algorithm 2: `Predictor.sample_observation()`

Input: Belief context $\mathcal{B}$, action a, samples N
Result: Predicted observations $\{\hat{o}^{(1)}, ..., \hat{o}^{(N)}\}$
for $i = 1$ **to** N **do**
  Construct prompt p_i with $\mathcal{B}$ and a
  $\hat{o}^{(i)} \leftarrow \pi_{\text{LLM}}(p_i)$ `// LLM response`
end
return $\{\hat{o}^{(1)}, ..., \hat{o}^{(N)}\}$

F.3 Algorithm for Belief Update in Detail

Algorithm 3: SubjectBelief.update_with_observation()

Input: Subject s, location x, observation o
Data: Belief $P(x|s)$ over $x \in \mathcal{X}$
Result: Updated $P'(x|s)$
$\mathcal{O}_x \leftarrow$ `parse_subjects_from`(o) `// Extract objects at` x
foreach $x' \in \mathcal{X}$ **do**
  if $x' = x$ **then**
    if $s \in \mathcal{O}_x$ **then**
      $P'(x'|s) \leftarrow P(x'|s) \cdot \alpha$ `// Boost`
    else
      $P'(x'|s) \leftarrow P(x'|s) \cdot \beta$ `// Decay`
    end
  else
    $P'(x'|s) \leftarrow P(x'|s)$
  end
end
Normalize $P'(x'|s)$ over x'

G Baseline Details

ETO Song et al. [28] introduces a novel learning framework for LLM agents that leverages both successful and failed trajectories to enhance performance. By iteratively collecting failure trajectories during exploration and applying contrastive learning (e.g., DPO loss) between failure-success pairs, ETO refines agent policies beyond traditional imitation learning.

WKM Qiao et al. [20] enhances LLM-based agents with structured task-level and state-level knowledge extracted from expert trajectories. At inference time, the agent leverages global task knowledge for high-level planning and dynamic state knowledge for step-wise action grounding, reducing hallucinated or inefficient behavior. This approach improves generalization across unseen tasks and environments by integrating procedural context directly into the action selection process. As the original paper does not report scores for LLaMA 3.1, we reproduced their method under the same environment and evaluation setting.

MPO Xiong et al. [33] introduces a plug-and-play framework that enhances LLM-based agents by providing high-level, abstract meta plans to guide task execution. By leveraging feedback from agent performance, MPO refines these meta plans, leading to improved task completion efficiency and generalization across unseen scenarios. WebShop results are not reported in the original paper, so we were unable to include them in the Table 1

IPR Xiong et al. [32] introduces a framework that enhances LLM agent training by providing step-level supervision through Monte Carlo-estimated rewards. By generating contrastive action pairs from discrepancies between agent actions and expert trajectories, IPR improves action efficiency and generalization across tasks.

STeCa Wang et al. [30] introduces a framework that enhances LLM agent learning by identifying suboptimal actions through step-level reward comparisons during exploration. By constructing calibrated trajectories using LLM-driven reflection, STeCa enables agents to learn from improved decision-making processes, leading to enhanced robustness in long-horizon tasks

RAFA Liu et al. [14] introduces a principled framework that integrates long-term reasoning and short-term acting for autonomous LLM agents. By modeling reasoning as Bayesian adaptive MDP planning and employing in-context learning for policy updates, RAFA achieves provable $\sqrt{T}$ regret and demonstrates strong empirical performance across multiple benchmarks.

Code repositories are as follows:

- ETO [28]: `https://github.com/Yifan-Song793/ETO`
- WKM [20]: `https://github.com/zjunlp/WKM`
- MPO [33]: `https://github.com/WeiminXiong/MPO`
- IPR [32]: `https://github.com/WeiminXiong/IPR`
- STeCa [30]: `https://github.com/WangHanLinHenry/STeCa`
- RAFA [14]: `https://github.com/agentification/RAFA_code`

H Experiment Configuration

Environment. We evaluate our method on two partially observable instruction-following benchmarks: ALFWorld. In ALFWorld, the agent is tasked with finding a target object (e.g., mug, toiletpaper, soap) within a simulated household, navigating and interacting with discrete actions (go, open, search) to gather natural language feedback (e.g., "You open cabinet 1. It is empty."). Observations do not explicitly name the target unless correctly found. We cap the maximum number of steps per episode at 30 for ALFWorld to enforce consistent evaluation with baselines.

Agent Configuration. We evaluate both open-source and proprietary LLMs as agents. For open-source models, we use LLaMA 2 and LLaMA 3.1–8B variants, testing both task-specific fine-tuned models and unmodified (zero-shot) versions. Proprietary models such as GPT-4o-mini and GPT-4 are accessed via the OpenAI API and used as-is without any additional adaptation. All models are prompted using a unified instruction-following format across tasks. We set the maximum output token length to 1024. Temperature is set to 0.7 for observation sampling tasks and defaults otherwise.

Task Protocol. In ALFWorld, we follow the standard evaluation splits introduced in prior work, using 140 seen and 134 unseen tasks from the reference benchmark. Each task is executed once without retries. Success is measured by correct object localization or item selection within the step limit.

Task type	# train	# seen	# unseen
Pick & Place	790	35	24
Examine in Light	308	13	18
Clean & Place	650	27	31
Heat & Place	459	16	23
Cool & Place	533	25	21
Pick Two & Place	813	24	17
All	3,553	140	134

Table 3: Six ALFWorld task types with heldout seen and unseen evaluation sets.

Computation Resources. All experiments were conducted using NVIDIA A100 (40GB) and RTX 3090 (24GB) GPUs on internal compute clusters. Experiments with open-source models were performed on 1–2 GPUs per run. Proprietary models (e.g., GPT-4) were accessed through the OpenAI API.

I Additional Experimental Result

We conduct additional experiments to evaluate the generality of our method across different alignment strategies. Also, we analyze the hypotheses' evolution over steps.

I.1 Comparison with Alignment-based Baselines.

Table 4 compares our method against prior alignment-tuned agents, including SFT, PPO, and recent RL-based approaches such as IPR and SteCa. Our method (AWS) achieves the highest performance on the ALFWorld benchmark, outperforming all prior alignment strategies on both seen and unseen tasks.

Method	ALFWorld (Unseen)
Llama-2-7B-Chat + PPO [24]	29.1
Llama-2-7B-Chat + SFT [36]	67.2
Llama-2-7B-Chat + RFT [35]	66.4
Llama-2-7B-Chat + Step-PPO [31]	69.7
Llama-2-7B-Chat + ETO [28]	72.4
Llama-2-7B-Chat + IPR [32]	74.7
Llama-2-7B-Chat + SteCa [30]	76.1
Llama-2-7B-Chat + AWS (Ours)	**76.8**

Table 4: **Comparison between different alignment strategies.** Our method (AWS) achieves the highest average performance across seen and unseen splits, outperforming prior supervised and RL-tuned baselines. Bold numbers indicate the best in each column.

I.2 Step-Wise Hypothesis Evolution

To better understand how each variant adapts global hypotheses over time, we present a step-wise comparison of inferred world structures during the mug search task (see Table 5). At each step, the agent forms hypotheses about room structure and object placement, conditioned on recent observations.

Step	Agent	Hypothesis Summary	Interpretation
1	Similarity	"Centralized kitchen counter"	Used for daily tools (sponge, spoon, knives)
	LLM	"Countertops used for daily activity"	Includes non-kitchen items (e.g., credit card, potato)
2	Similarity	"Multi-zone structure emerging"	Breakfast prep zone, Plant care zone
	LLM	"Countertop clutter"	No dedicated item location
3	Similarity	"Meal/snack prep inferred"	Tomato and potato → cooking zone emerging
	LLM	"Sinkbasin & countertop = temp workspace"	Multi-purpose usage (cluttered logic remains)
4	Similarity	"Storage strategy detected"	Soap + bottles categorized into cabinet
	LLM	"Zone cleaning in progress"	Begins forming structure (cleaning-based policy)
5	Similarity	"Gaps in storage logic"	Cabinets partially filled, inconsistent
	LLM	"Temporary vs. designated storage co-exist"	Sponge & bottle in cabinet → early categorization
6	Similarity	"Category mismatch in cabinet items"	Salt & pepper in different cabinets
	LLM	"Systematic labeling emerging"	Forks numbered → category grouping via labels
7	Similarity	"Storage pattern identified"	Mug observed in cabinet with soap → final confirmation
	LLM	"Consistent categorization & order"	Spatula 2 in drawer, cabinets tidied → structured conclusion

Table 5: Step-wise evolution of global hypotheses for each agent. The similarity-based variant exhibits gradual zone-based inference, while the LLM-based variant begins with cluttered understanding and quickly transitions to structured categorization.

Similarity-Based Variant. This agent builds structure incrementally. It first assumes the kitchen counter is a centralized utility zone, then refines this into functional subregions (e.g., breakfast prep, plant care) as more items are observed. Later, it detects inconsistencies (e.g., salt and pepper in different cabinets) and ultimately confirms a coherent storage pattern-such as a mug stored alongside cleaning items-highlighting a gradual, zone-based interpretation process.

LLM-Based Variant. The LLM-based agent starts with less grounded hypotheses, interpreting surfaces as generic and cluttered workspaces. However, as structural signals emerge (e.g., numbered forks, categorized placements), it transitions rapidly to a policy-driven world model based on cleaning routines or semantic labeling. By the final step, it exhibits consistent categorization and spatial order.

In summary, while both variants succeed, their reasoning paths differ: **Similarity-Based** infers structure gradually via bottom-up spatial cues, whereas **LLM-Based** generalizes early from sparse patterns, leading to faster semantic organization.