Agent-BRACE: Decoupling Beliefs from Actions in Long-Horizon Tasks via Verbalized State Uncertainty

Joykirat Singh[1] **Zaid Khan**[1] **Archiki Prasad**[1] **Justin Chih-Yao Chen**[1]
Akshay Nambi[3] **Hyunji Lee**[1] **Elias Stengel-Eskin**[2] **Mohit Bansal**[1]

[1]UNC Chapel Hill [2]The University of Texas at Austin [3]Microsoft Research

Abstract

Large language models (LLMs) are increasingly deployed on long-horizon tasks in partially observable environments, where they must act while inferring and tracking a complex environment state over many steps. This leads to two challenges: *partial observability* requires maintaining uncertainty over unobserved world attributes, and *long interaction history* causes context to grow without bound, diluting task-relevant information. A principled solution to both challenges is a *belief state*: a posterior distribution over environment states given past observations and actions, which compactly encodes history for decision making regardless of episode length. In LLM agents, however, the open-ended nature of text makes it unclear how to represent such a distribution. Therefore, we introduce `Agent-BRACE`: **Agent** **B**elief state **R**epresentation via **A**bstraction and **C**onfidence **E**stimation, a method that decouples an LLM agent into a *belief state model* and a *policy model*, jointly optimized via reinforcement learning. The belief state model produces a structured approximation of the belief distribution: a set of atomic natural language claims about the environment, each annotated with an ordinal verbalized certainty label ranging from certain to unknown. The policy model conditions on this compact, structured approximate belief rather than the full history, learning to select actions under explicit uncertainty. Across long-horizon, partially observable embodied language environments, `Agent-BRACE` achieves an average absolute improvement of +14.5% (Qwen2.5-3B-Instruct) and +5.3% (Qwen3-4B-Instruct), outperforming strong RL baselines while maintaining a near-constant context window independent of episode length. Further analysis shows that the learned belief becomes increasingly calibrated over the course of an episode as evidence accumulates.[1]

1 Introduction

Large language models (LLMs) are increasingly being deployed as agents in long-horizon, partially observable tasks like software engineering [Yang et al., 2024, Jimenez et al., 2024], web navigation [Zhou et al., 2023, Deng et al., 2023, He et al., 2024], or research [Lu et al., 2024, Novikov et al., 2025]. These models must act while inferring complex world state from incomplete observations over many steps – a setting that is traditionally modeled as a partially observable Markov Decision Process (POMDP) [Åström, 1965]. In this framing, an optimal policy needs to only condition on the *belief state*, a posterior distribution over possible environment states given the history of past/current observations and past actions. The belief state admits two complementary interpretations: *(1)* the distribution represents uncertainty over the state the agent is in, accounting for unobserved variables; *(2)* it serves as a sufficient statistic for the prior interaction history $\mathcal{H}_t$, allowing the agent to track observations over time. Current LLM agents differ from traditional POMDP approaches in that they

[1]Codebase: https://github.com/joykirat18/Agent-BRACE

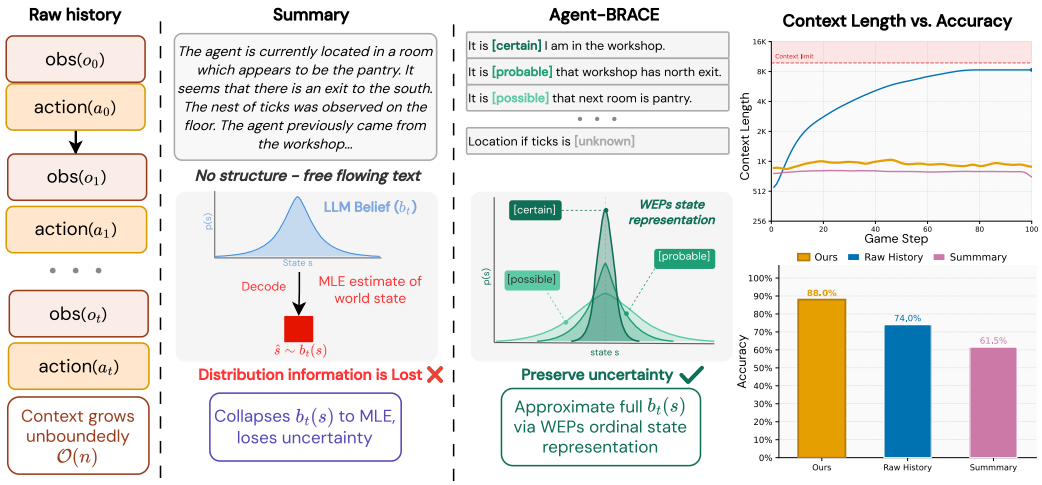

Figure 1: Three approaches to context management in long-horizon POMDP agents. **Raw history** (left), grows unbounded as $\mathcal{O}(n)$. **Summary-based belief** (middle) compresses history into a summary but collapses the belief distribution to a single MLE point estimate $\hat{s} \sim b_t(s)$, discarding uncertainty. **Agent-BRACE** (right) represents the belief as WEP-annotated statements (*confirmed*, *probable*, *possible*, etc.), approximating the full distribution $b_t(s)$, with near-constant context window. `Agent-BRACE` (Qwen3-4B-Instruct) outperforms both baselines in accuracy while maintaining constant context length (right panel).

generally represent both actions and observations in text. This enables interaction with open-ended, unstructured environments that lack predefined action or observation schema, but complicates encoding an explicit belief state and introduces its own challenges. First, without a sufficient statistic of history, LLM-based policies must be conditioned on the raw interaction trajectory, leading to inefficient representation (Fig. 1; *Raw history*), with the context length growing linearly in the episode length, thus increasing computational cost and diluting task-relevant signals with spurious details [Liu et al., 2024, Chung et al., 2025] (Fig. 1; *Context Length vs Accuracy*). Second, while POMDP approaches for large or continuous state spaces are well studied (e.g., particle filters, predictive state representations) [Silver and Veness, 2010, Hafner et al., 2020, Gregor et al., 2019], the open-ended nature of text poses its own challenges: it is unclear how to encode a distribution in text over a compositional state space. Indeed, past work either relies on the LLM's internal representation as a belief proxy [Kamel et al., 2025] – which lacks interpretability and limits external verification – or externalizes belief into a free-form natural language summary [Zhou et al., 2025, Yu et al., 2025], which is more interpretable but collapses the belief distribution $b_t(s)$ into a single point estimate.

To tackle these challenges and preserve uncertainty in belief states for LLM agents, we introduce `Agent-BRACE`: **Agent** **B**elief state **R**epresentation via **A**bstraction and **C**onfidence **E**stimation, a training method that represents an agent's belief as text while simultaneously encoding uncertainty via verbalized probability estimates. Following the POMDP formalization, `Agent-BRACE` decouples an LLM agent into two modules: a *belief state model* and a *policy model*, training them jointly using reinforcement learning (RL). As shown in Fig. 2 (*belief state update*), at each step t the belief state model takes as input the goal (G), the previous belief (b_t), and the new observation (o_{t+1}), and produces an updated approximate belief (b_{t+1}) represented as a set of atomic natural language claims. Crucially, each claim is annotated with a certainty label drawn from the Words of Estimative Probability (WEP) scale [van Tiel et al., 2022, Tang et al., 2026, Sileo and Moens, 2023]; an ordered Likert style vocabulary (*confirmed* $\succ$ *almost certain* $\succ$ *probable* $\succ$ *possible* $\succ$ *unlikely* $\succ$*doubtful* $\succ$ *unknown*) that is grounded in how humans express uncertainty in natural language. Prior work has shown LLMs can meaningfully produce and differentiate between such verbalized uncertainty expressions [Lin et al., 2022, Tian et al., 2023, Stengel-Eskin et al., 2024]. This yields a belief approximation that captures uncertainty and uses a discrete scale that LLMs can reliably produce and update. Since the belief state b_t is a sufficient approximation of the full history, the policy model can select an action conditioned on (G, b_t, o_t) rather than on the history $\mathcal{H}_t$ – replacing an ever-growing trajectory with a compact, bounded representation. In `Agent-BRACE`, the belief state model and the

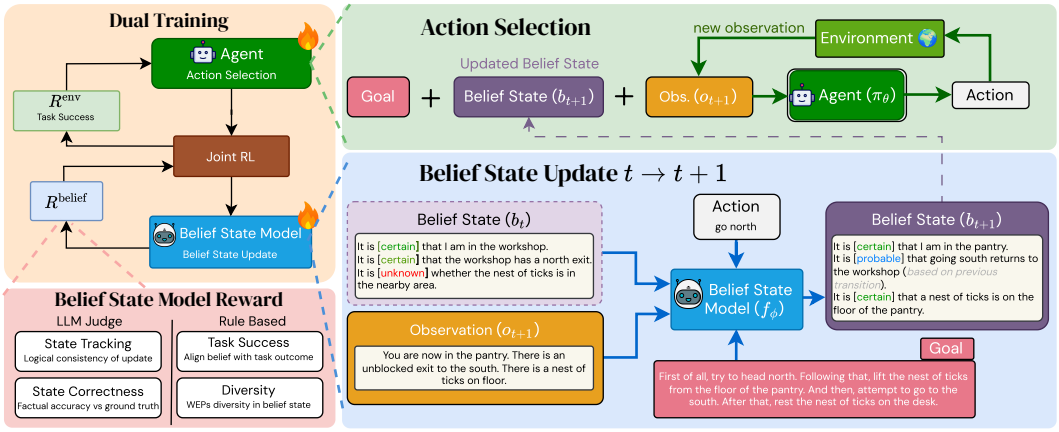

Figure 2: Overview of `Agent-BRACE`. The agent is decomposed into a *belief state model* f_ϕ and a *policy model* π_θ, jointly optimized via PPO (*Dual Training*). At each step t, f_ϕ consumes the goal G, previous belief b_t, and new observation o_{t+1} to produce an updated belief b_{t+1} with WEPs-based certainty labels (*Belief State Update*). The policy π_θ then selects an action a_t conditioned on (G, b_{t+1}, o_{t+1}) rather than the full history $\mathcal{H}_t$ (*Action Selection*). The belief model is trained with a composite reward R^{belief}, while the policy model is trained with a binary environment reward R^{env}.

policy model are jointly trained via PPO [Schulman et al., 2017]. The policy model is optimized to maximize the binary environment reward (+1 for success, 0 for failure), providing the primary signal for action selection. The belief state model is optimized via a combination of complementary rewards, each targeting a different failure mode in belief quality: state tracking ensures logical consistency [Zou et al., 2026, Yuan et al., 2026], state correctness ensures factual grounding [Zhao et al., 2026], diversity prevents uncertainty collapse [Leng et al., 2024], discounted success aligns belief quality with task outcomes, and format ensures structural consistency. Ablations in Section 4 confirm the importance of each reward.

We train and evaluate `Agent-BRACE` on various long-horizon, partially observable embodied language tasks. Specifically, `Agent-BRACE` is trained on Quest, a task from the *TextWorld* [Côté et al., 2018] environment, using Qwen2.5-3B-Instruct [Qwen Team, 2024] and Qwen3-4B-Instruct [Qwen Team, 2025] as base models and evaluate on three *TextWorld* environments: Quest, Treasure, and Cooking. `Agent-BRACE` outperforms all baselines, including ReAct [Yao et al., 2022], Direct-Action (RL trained), ReAct (RL trained), MEM1 [Zhou et al., 2025], and PABU [Jiang et al., 2026], achieving average accuracies of 72.8% and 79.3% on Qwen2.5-3B-Instruct and Qwen3-4B-Instruct, respectively – an average absolute improvement[2] of +14.5% over the strongest RL-trained baseline (Direct-Action (RL)) on Qwen2.5-3B-Instruct and +5.3% on Qwen3-4B-Instruct. Crucially, `Agent-BRACE` maintains a near constant context window while achieving the best performance. `Agent-BRACE` also demonstrates strong generalization, achieving consistently high performance on Treasure and Cooking tasks despite being trained only on Quest. Moreover, we show that `Agent-BRACE` can be extended to other tasks, with +2.85% improvement over the strongest RL-trained baseline on ALFWorld [Shridhar et al., 2020b]. Our ablations confirm that each component contributes meaningfully: joint training, belief-state rewards, and an expressive WEP label set each play a critical role – removing any one leads to meaningful degradation. Further analysis shows that the belief becomes better calibrated over the course of an episode, with Brier score [Glenn et al., 1950] decreasing from 0.40 to 0.28 and the fraction of *confirmed* claims growing from 21% to 52% as evidence accumulates.

2 Methodology: Agent-BRACE

In this section, we introduce our method in detail (Fig. 2). `Agent-BRACE` addresses two core challenges in long-horizon agentic tasks: linear growth of past history and the absence of belief representation under partial observability. To tackle this, `Agent-BRACE` jointly trains a *belief state*

[2]All improvements reported in this paper are absolute unless otherwise stated.

model and a *policy model* via PPO, where the belief state model maintains a structured uncertainty-aware belief that serves as a sufficient approximation of the history for downstream action selection.

2.1 Environment and Agentic Task

We focus on partially-observable environments modeled as POMDPs defined by the tuple $\mathcal{M} = (S, T, A, \Omega, O, R, \gamma)$, where S is the set of latent environment states, $T : S \times A \to \Delta(S)$ is the state-transition distribution, A is the natural language action space, Ω is the observation space, $O : S \times A \to \Delta(\Omega)$ is the observation distribution, $R : S \times A \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discount factor. Since the current observation o_t is not a sufficient statistic for the environment state s_t [Kaelbling et al., 1998], an optimal policy π must condition on the full history $\mathcal{H}_t = \{G, o_0, a_0, \ldots, o_t\}$, or an equivalent belief state $b_t = P(s_t|\mathcal{H}_t)$, to maximize expected cumulative rewards [Åström, 1965]. We consider agentic tasks where an LLM pursues a goal G by interacting with the environment ϵ, until the objective is achieved or a step budget is reached.

2.2 Decoupled Architecture: Belief State Model and Policy Model

As shown in Fig. 2 (*Dual Training*), our approach parameterizes an agent with two jointly optimized components: a *belief state model* for state estimation and a *policy model* for action selection:

Belief State Model (f_ϕ): This model is a learnable belief-update function, constructing and maintaining an approximate belief representation from raw environment observations. Each belief state is represented as a set of statements, where each statement is annotated with a WEP-based uncertainty label. As shown in Fig. 2 (*Belief State Update*), f_ϕ consumes the goal G, the current belief state b_t, and the new observation o_{t+1} to produce an updated belief state $b_{t+1} = f_\phi(G, b_t, o_{t+1})$.

Policy Model (π_θ): This targets the long-horizon history challenge by conditioning action selection on the compact belief b_{t+1} generated by the belief state model rather than the full history $\mathcal{H}_t$. The next action becomes $\pi_\theta(G, b_{t+1}, o_{t+1})$.

2.3 Belief State Representation

As shown in Fig. 1 (*Summary*), representing belief states as natural language summaries collapses the distribution over environment state s_t into a single point estimate, discarding the *uncertainty* that belief state policies rely on in partially observable settings. Such summaries are unstructured, producing free-flowing prose with no separation between distinct facts, making it difficult for the policy model to locate and extract task-relevant signals. This problem compounds over time as summaries grow to accommodate new observations [Kang et al., 2025], increasing both the length and density of interleaved facts the policy must parse. Instead, we represent belief state b_t as *a set of verbalized belief statements*: atomic natural language claims about individual aspects of the environment, each annotated with an explicit uncertainty label (example shown in Fig. 2) drawn from the Words of Estimative Probability (WEP) scale [Kent, 1964] – an ordered Likert-style vocabulary grounded in how human naturally express uncertainty: *confirmed* $\succ$ *almost certain* $\succ$ *probable* $\succ$ *possible* $\succ$ *unlikely* $\succ$*doubtful* $\succ$ *unknown*. This yields a structured approximation of the belief distribution $b_t(s)$ that is both interpretable and reliably produced by LLMs [Lin et al., 2022, Tian et al., 2023, Xiong et al., 2023].

To define the belief space, we specify set of structured slots that every belief state must populate (e.g, agent location, inventory, etc.); further details in Appendix A. While specific values and their instantiations are learned from task experience rather than hand-specified, automatically discovering belief dimensions in fully open-ended environments remains future work. We ablate this structured belief space in Appendix C, finding only a minor drop in performance without it, suggesting that the joint training provides a strong foundation for belief learning, with structured belief states delivering additional gains.

2.4 Joint Training Procedure

As shown in Fig. 2, we train the belief state model f_ϕ and policy model π_θ jointly via Proximal Policy Optimization (PPO) [Schulman et al., 2017]. Prior to PPO, we perform supervised finetuning (SFT) on the belief state model using GPT 5.4 mini [OpenAI, 2026] trajectories to enforce a structured

belief state representation (SFT detail in Appendix B; SFT stage ablation in Appendix C). The policy model π_θ is trained to maximize the binary environment reward R^{env}. Simultaneously, the belief state model is optimized to maximize the composite reward R^{belief}. The clipped objective is given by:

$$\mathcal{L}_{\text{policy/belief}}(\theta) = -\hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{1}$$

where $r_t(\theta)$ is the probability ratio. For the policy model, $\hat{A}_t$ is estimated via Generalized Advantage Estimation (GAE) [Schulman et al., 2015] using a learned critic $V(s_t)$. To reduce the computational cost, we approximate the advantage: $\hat{A}_t^{\text{belief}} = \frac{R_t^{\text{belief}} - \mu_r}{\sigma_r}$, where μ_r and σ_r are the mean and standard deviation of rewards computed over rollouts from the same task. Both modules are trained via the same objective, but differ in advantage estimation: the policy model uses GAE with a learned critic $V(s_t)$, while the belief state model uses GRPO style group-normalized returns [Shao et al., 2024].

2.5 Reward Design

`Agent-BRACE` is trained with two distinct reward signals: R^{belief} for the belief state model and R^{env} for the policy model, optimized jointly so that the belief representations are shaped by the policy's decision-making needs. Both rewards are summarized in Table 1. The final belief state model reward $R_t^{\text{belief}} = r_t^{\text{format}} \times \frac{1}{4}\left(r_t^{\text{st}} + r_t^{\text{sc}} + r_t^{\text{div}} + r_t^{\text{success}}\right)$. Full detail in Appendix D.

Table 1: Reward components for the belief state model and policy model.

Reward	**Symbol**	**Purpose**
Belief State Model (R^{belief})		
State Tracking	r_t^{st}	Measures the logical consistency of the belief update $b_{t-1} \rightarrow b_t$ given observation o_t.
State Correctness	r_t^{sc}	Ensures that the belief state claim, along with its uncertainty score, remains grounded to the environment states.
Diversity	r_t^{div}	Encourages use of the full WEP vocabulary via entropy $H(b_t)$ of the label histogram
Format	r_t^{format}	Enforces structured output; acts as a multiplicative gate zeroing all other rewards for invalid outputs
Task Success	r_t^{success}	Propagates task outcome to the belief module via $\gamma^t \times \mathbf{1}[\text{success}]$
Policy Model (R^{env})		
Task Success	R^{env}	Binary reward from the environment ($+1$ for success, 0 for failure); primary reinforcement signal for action selection

3 Experimental Setup and Results

3.1 Setup

Models. To evaluate `Agent-BRACE`, we adopt instruction tuned models, Qwen2.5-3B-Instruct [Qwen Team, 2024] and Qwen3-4B-Instruct [Qwen Team, 2025] as our base models. Both the belief state and policy models are initialized from the same base model. Additionally, we use Qwen3-30B-A3B-Instruct [Qwen Team, 2025] as the judge to evaluate state tracking and correctness reward.

Datasets. We train and evaluate on *TextWorld* [Côté et al., 2018], which provides the flexibility to generate multiple types of text-based games. We construct three different tasks using *TextWorld* environment: (i) **Quests:** A text adventure game environment where agents navigate rooms, manipulate objects, and solve quests via natural language; (ii) **Cooking:** This task takes place in a typical house and consists in finding the right food item and cooking it; (iii) **Treasure:** The agent spawns in a randomly generated maze and must find a specific object which is mentioned in the objective displayed when the game starts. `Agent-BRACE` and other training baselines were trained only on Quest and evaluated on all three datasets. Full details are provided in Appendix E.

Baselines. We compare with several strong baselines **(1) Base Model:** off the shelf instruction tuned model; **(2) ReAct** [Yao et al., 2022]: Interleaved reasoning and action selection; **(3) Direct-Action**

Table 2: Performance comparison of `Agent-BRACE` against baselines across three TextWorld environments (Quest, Treasure, Cooking) on Qwen2.5-3B-Instruct and Qwen3-4B-Instruct. Acc. denotes task success rate (%, higher is better); Steps denotes average number of steps taken (lower is better). Blue rows are inference-only (no training); `Agent-BRACE` and other baselines are trained with Quest only, while Treasure and Cooking are out-of-domain (OOD) tasks.

Method	Quest		Treasure		Cooking		Average	
	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓
Qwen2.5-3B-Instruct								
Base Model	4.0	96.1	7.5	93.2	2.5	98.1	4.7	95.8
ReAct	23.0	37.6	37.0	33.6	27.5	38.4	29.2	36.5
Direct-Action (RL)	56.0	35.8	67.5	32.6	51.5	46.1	58.3	38.2
ReAct (RL)	46.5	34.2	55.0	32.7	34.5	44.4	45.3	37.1
MEM1	29.5	62.9	30.0	47.7	52.5	48.0	37.3	52.9
PABU	73.0	37.0	72.5	34.4	33.0	73.1	59.5	48.2
`Agent-BRACE`	**78.5**	37.3	**81.5**	32.1	**58.5**	60.3	**72.8**	43.3
Qwen3-4B-Instruct								
Base Model	61.5	32.3	65.0	30.3	69.5	34.1	65.3	32.2
ReAct	60.5	12.6	69.5	10.3	13.5	24.4	47.8	15.8
Direct-Action (RL)	74.0	29.6	72.5	28.0	**75.5**	31.9	74.0	29.8
ReAct (RL)	75.5	18.2	74.0	16.5	13.0	40.6	54.2	25.0
MEM1	61.5	50.2	63.5	31.4	10.0	10.0	45.0	30.5
PABU	82.2	29.1	73.5	37.2	32.5	75.6	62.7	47.3
`Agent-BRACE`	**88.0**	30.5	**81.0**	30.0	69.0	44.6	**79.3**	35.0

(RL): PPO trained model that directly outputs actions, using the same final environment reward as `Agent-BRACE`; **(4) ReAct (RL):** PPO trained model that additionally outputs its thinking inside *<think> ... </think>* tokens before taking an action; **(5) MEM1** [Zhou et al., 2025]: RL framework that maintains a compact shared state for memory consolidation and reasoning – integrating prior memory with new observations while strategically discarding irrelevant or redundant information; **(6) PABU** [Jiang et al., 2026]: Belief-state framework that compactly represents an agent's state by explicitly modeling task progress and selectively retaining past actions and observation.

Implementation Details. The maximum number of turns during training is set to be 15, and during inference, to test the long-horizon capability of the method, we set the maximum number of turns to be 100. Additional hyperparameter details are available in Appendix F.

3.2 Main Results

Agent-BRACE outperforms other baselines. Table 2 presents the main results across three *TextWorld* environments for both Qwen2.5-3B-Instruct and Qwen3-4B-Instruct. Overall, `Agent-BRACE` achieves the highest average accuracy across all baselines with 72.8% on Qwen2.5-3B-Instruct and 79.3% on Qwen3-4B-Instruct, an absolute improvement of +14.5% and +5.3% over the strongest RL-trained baseline, Direct-Action (RL), respectively. On Qwen2.5-3B-Instruct, `Agent-BRACE` outperforms ReAct (RL) by +27.5%, demonstrating that interleaved chain-of-thought reasoning alone is insufficient under partial observability. Against MEM1, `Agent-BRACE` improves by +35.5% on Qwen2.5-3B-Instruct and +34.3% on Qwen3-4B-Instruct, confirming that summary-based compression discards task-critical signals. `Agent-BRACE` on average also outperforms PABU by +13.3% on Qwen2.5-3B-Instruct and +16.6% on Qwen3-4B-Instruct. The improvements are consistent across both models and suggest a clear pattern: baselines that treat history as a sufficient statistic, whether through raw context (Direct Action, ReAct), summarization (MEM1), or progress-aware compression (PABU), cannot maintain an uncertainty approximation over world state. `Agent-BRACE`'s improvement stems from explicit representation of an approximate belief state via WEP annotations and jointly optimizing the belief state model with the policy, so that the agent learns to act under uncertainty rather than from a single point estimate of the world.

Agent-BRACE generalizes to held-out TextWorld Environments. `Agent-BRACE` is exclusively trained on Quest, yet achieves strong and consistent performance across all three tasks (Table 2).

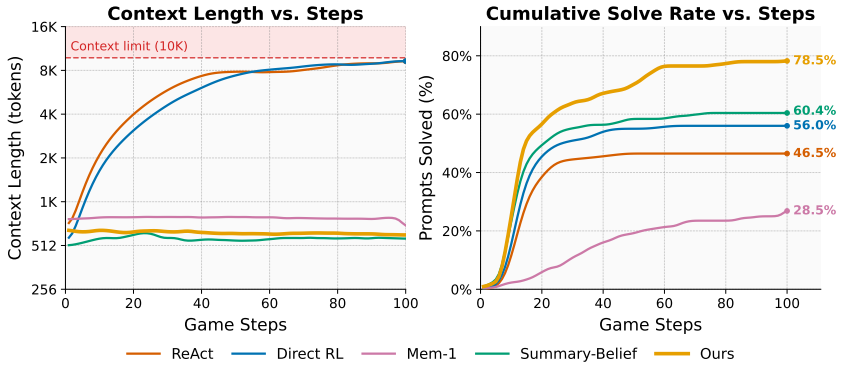

Figure 3: `Agent-BRACE` maintains a near constant context window while achieving the highest solve rate (78.5%). Comparison of context length growth (left) and cumulative solve rate (right) across methods with maximum 100 game steps on Quest using Qwen2.5-3B-Instruct.

It transfers most effectively to Treasure (81.5% on Qwen2.5-3B-Instruct and 81.0% on Qwen3-4B-Instruct), which shares Quest's navigation structure. On cooking, which requires sequential sub-goal completion and is structured differently from Quest, `Agent-BRACE` still outperforms most baselines (+7.0% over Direct-Action and +6.0% over MEM1 on Qwen2.5-3B-Instruct). MEM1, ReAct (RL), and PABU all collapse on Cooking under Qwen3-4B-Instruct (10.0%, 13.0%, and 32.5%), confirming that history-based and progress-aware representations are brittle under the non-monotonic sub-goal structure of Cooking. `Agent-BRACE`, maintaining per-claim uncertainty without assuming a linear progress signal, remains robust across both settings without per-task engineering.

Agent-BRACE maintains a bounded context window and enhanced solve rate. Fig. 3 plots context length (left) and cumulative solve rate (right) across methods over 100 steps. ReAct and Direct-Action (RL) grow linearly, eventually exceeding the context limit, while `Agent-BRACE` maintains a near constant context window. Crucially, despite operating within a similar context budget as MEM1 and Summary-Belief (ablation run where belief state model is replaced with LLM summarizer), `Agent-BRACE` achieves a substantially higher cumulative solve rate of 78.5% vs 28.5% and 60.4% respectively. The gap isolates the contribution of approximating the belief distribution via WEP labels from context bounding alone.

Agent-BRACE performance on ALFWorld. While Quest, Cooking, and Treasure share common navigation and inventory structure, we wanted to test `Agent-BRACE` generalization to tasks with qualitatively different actions. ALFWorld [Shridhar et al., 2020b] is built on top of the ALFRED [Shridhar et al., 2020a] household dataset that requires the agent to execute multi-step object manipulation (e.g., pick, clean, heat, cool, place) with different observation and action structures from the TextWorld environment. Table 3 shows the performance of `Agent-BRACE` (Qwen3-4B-Instruct) on ALFWorld, averaged over three evaluation runs. `Agent-BRACE` achieves the highest accuracy of 30.71%, outperforming the strongest RL-trained baseline Direct-Action (RL) by +2.85% and the Base Model by +6.42%. It also outperforms memory based baseline MEM1 by 5%. These results suggest that `Agent-BRACE`'s structured belief representation generalizes to other tasks.

Table 3: Performance comparison of `Agent-BRACE` (Qwen3-4B-Instruct) against baselines on ALFWorld environment. Blue rows are inference-only.

Method	Acc.↑	Steps↓
Base Model	24.3	20.1
ReAct	16.4	9.6
Direct-Action (RL)	27.9	19.1
ReAct (RL)	22.1	10.6
MEM1	25.7	42.9
`Agent-BRACE`	**30.7**	39.1

4 Ablation and Analysis

To understand the importance of each component of `Agent-BRACE`, we do an ablation study, specifically **(1)** `Agent-BRACE` **(Limited WEP):** Training the belief state model to only capture two levels of uncertainty – *confirmed* and *unknown*; **(2)** `Agent-BRACE` **(Summary-Belief)**: Instead of training belief state to capture both abstraction of past history and uncertainty, it only summarizes past history

Table 4: Ablation analysis of `Agent-BRACE` using Qwen2.5-3B-Instruct and Qwen3-4B-Instruct. Each component is a variation of `Agent-BRACE`.

Method	Quest		Treasure		Cooking		Average	
	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓
Qwen-2.5-3B-Instruct								
`Agent-BRACE`	**78.5**	37.3	**81.5**	32.1	58.5	60.3	**72.8**	43.3
- Limited WEP	76.0	42.4	71.0	40.1	58.0	66.1	68.3	49.5
- No State Reward	78.0	39.4	73.0	40.2	**63.0**	53.7	71.3	44.5
- Frozen belief model	66.5	43.5	59.5	46.2	28.0	80.3	51.3	56.7
- Summary-Belief	60.4	24.0	54.3	34.1	3.5	18.3	39.4	25.5
Qwen3-4B-Instruct								
`Agent-BRACE`	**88.0**	30.5	**81.0**	30.0	**69.0**	44.6	**79.3**	35.1
- Limited WEP	79.0	35.6	69.0	40.7	48.0	73.9	65.3	50.1
- No State Reward	59.5	44.4	64.5	45.1	58.5	57.6	60.8	45.0
- Frozen belief model	77.5	34.7	57.0	50.1	35.5	74.1	56.7	53.3
- Summary-Belief	61.5	47.1	36.0	66.4	38.5	67.4	45.3	60.3

$\mathcal{H}_k$; **(3)** `Agent-BRACE` **(Frozen Belief model):** The belief state model remains static and only the policy model is trained; **(4)** `Agent-BRACE` **(No State Reward):** The belief model is only trained on discounted success reward and other state relevant rewards are removed.

Belief uncertainty becomes better calibrated over the course of an episode. Fig. 4 analyzes the uncertainty representations learned by the belief model. To compute the Brier scores, we map each WEP label to its nominal probability following the ordinal scale (e.g., *confirmed* ≈ 1.0, *probable* ≈ 0.75, *possible* ≈ 0.50, *doubtful* ≈ 0.25, *unknown* ≈ 0.0), and compute the Brier score [Glenn et al., 1950] against binary ground truth from the LLM judge (Qwen3-30B-A3B-Instruct-2507); 0 indicates perfect calibration and 0.25 corresponds to random chance brier score. Brier score decreases steadily from 0.4 at step 0 to below 0.28 by step 14, a reduction of approximately 0.12 points, and remains well below the random baseline of 0.25 throughout, while the fraction of *confirmed* claims grows from 21% to 52% – confirming that the belief model progressively sharpens its uncertainty estimate as evidence accumulates. Additional calibration analysis (Appendix G) reveals that the belief model begins underconfident but becomes increasingly calibrated over the course of training, with high-confidence labels (*almost certain*, *confirmed*) remaining well-calibrated throughout – a safer failure mode than overconfidence [Stengel-Eskin et al., 2024]. Appendix K provides qualitative examples of uncertainty labels being correctly assigned and progressively resolved as evidence accumulates.

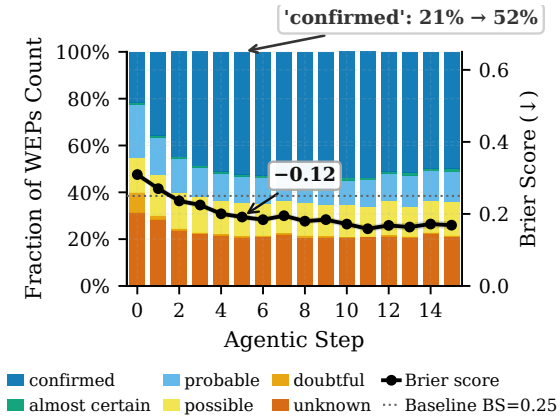

Figure 4: Brier score drops from 0.40 → 0.28 while *confirmed* claims grow 21% to 52%, confirming progressive calibration as evidence accumulates. WEP label distribution (bars, left axis) and mean Brier Score (line, right axis) across agent steps for Qwen3-4B-Instruct (`Agent-BRACE`) on Quest dataset.

Joint training of belief model and policy drives performance gains. Table 4 presents the results of various ablations of `Agent-BRACE` on Qwen2.5-3B-Instruct and Qwen3-4B-Instruct. When the belief model is frozen and restricted to summarizing past observations (Summary-Belief), the average performance drops by 33.4% and 34.0% for Qwen2.5-3B-Instruct and Qwen3-4B-Instruct, respectively. The drop is most severe on Cooking (58.5% → 3.5% on Qwen2.5-3B-Instruct; 69.0% → 38.5% on Qwen3-4B-Instruct), because Cooking requires fine-grained uncertainty tracking over sequential sub-goals and collapsing the belief distribution is particularly damaging when the agent must reason over multiple possible world states simultaneously. Similarly, freezing the belief model and training only the policy (Frozen belief model) shows an average accuracy drop by 21.5%

absolute points on Qwen2.5-3B-Instruct and 22.6% points on Qwen3-4B-Instruct, demonstrating the importance of optimizing the belief and policy models rather than treating belief as a fixed module.

Belief quality depends on reward design, label granularity, and task-specific supervision. As seen in Table 4, removing the state-grounding rewards (state correctness, state tracking, diversity, and format) degrades average accuracy by 1.5% on Qwen2.5-3B-Instruct and 18.5% on Qwen3-4B-Instruct, confirming that explicit grounding signals are critical for downstream performance. Collapsing the 7-level WEP scale to a binary certain/unknown drops average accuracy from 79.3% $\rightarrow$ 65.3% on Qwen3-4B-Instruct and 72.8% $\rightarrow$ 68.3% on Qwen2.5-3B-Instruct, demonstrating that uncertainty granularity directly impacts policy performance. Additionally we also ablated task-specific belief supervision (Appendix C), i.e, both the SFT warm-start and domain-structured prompting, which causes a moderate drop from 79.3% $\rightarrow$ 69.3% on Qwen3-4B-Instruct, yet without belief supervision, `Agent-BRACE` still outperforms most trained baselines, confirming that joint RL alone provides sufficient signal for structured world-state tracking.

5 Related Work

Belief state estimation has a long history in sequential decision making under partial observability. Classical POMDP solvers [Kaelbling et al., 1998, Smallwood and Sondik, 1973] maintain exact belief distributions over finite, discrete state spaces with known transition and observation models. To scale to continuous state spaces, Monte Carlo methods such as particle filters approximate the belief distribution [Silver and Veness, 2010, Thrun, 1999]. When transition and observation models are unknown, deep RL approaches learn belief representations as distributions over latent states via variational inference [Hafner et al., 2020, Gregor et al., 2019]. However, these approaches assume a pre-specified, continuous state space and learn implicit latent representations, assumptions that break down in text-based environments where the state space is open-ended and interpretability is desirable.

Increasingly, LLMs form the backbone of interactive agents for long-horizon, partially observable tasks such as software and web navigation. ReAct [Yao et al., 2022] interleaves chain-of-thought reasoning with actions, conditioning on a growing interaction history. Several memory-based approaches [Yu et al., 2025, Kang et al., 2025, Zhang et al., 2025, Xu et al., 2025] address the resulting context burden: MEM1 [Zhou et al., 2025] selectively compresses past interactions through summarization. Although these methods reduce the burden of long contexts, they still treat history as a sufficient representation of the agent's state, without modeling uncertainty over the environment.

A complementary line of work directly addresses belief representation in LLM agents. One approach uses internal latent states as an implicit proxy for belief [Kamel et al., 2025], sacrificing interpretability. At the other extreme, StateAct [Rozanov and Rei, 2025], ABBEL [Lidayan et al., 2025], and PABU [Jiang et al., 2026] represent agent state as structured natural language summaries, but collapse the belief distribution to a single maximum-likelihood estimate, discarding uncertainty inherent to belief states. `Agent-BRACE` addresses it by externalizing belief in structured text while preserving uncertainty via per-claim WEP annotations and jointly optimizing the belief state model and the policy model via reinforcement learning.

6 Conclusion

We introduce Agent-BRACE, a training method that jointly trains a belief state model and a policy model via reinforcement learning. The belief state model produces a structured approximation of the belief distribution as atomic natural language claims annotated with a Words of Estimative Probability label on an ordered scale. Conditioning the policy on this compact belief rather than the raw interaction history simultaneously addresses two challenges that arise when LLM agents operate in POMDP-style environments: representing uncertainty over an open-ended state space, and bounding the context required for action selection. `Agent-BRACE` attains the highest average performance on three TextWorld environments on both Qwen2.5-3B-Instruct and Qwen3-4B-Instruct. Despite training only on Quest, the method transfers to the held-out Treasure and Cooking tasks, suggesting the structured belief representation captures task-relevant attributes without per-task engineering. Ablations confirm that joint optimization, the graded WEP scale, and the state tracking/correctness rewards each contribute meaningfully. Further analysis shows the learned belief is well calibrated and sharpens as evidence accumulates, with Brier score dropping as the steps progress.

Acknowledgments

This work was supported by Microsoft Agentic AI Research and Innovation (AARI) grant program, NDSEG PhD Fellowship, NSF-AI Engage Institute DRL-2112635, NSF-CAREER Award 1846185, and an Apple PhD Fellowship. The views contained in this article are those of the authors and not of the funding agency.

References

Karl Johan Åström. Optimal control of markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965.

Andy Chung, Yichi Zhang, Kaixiang Lin, Aditya Rawal, Qiaozi Gao, and Joyce Chai. Evaluating long-context reasoning in llm-based webagents. *arXiv preprint arXiv:2512.04307*, 2025.

Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, pages 41–75. Springer, 2018.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

W Brier Glenn et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Karol Gregor, Danilo Jimenez Rezende, Frederic Besse, Yan Wu, Hamza Merzic, and Aaron van den Oord. Shaping belief states with generative environment models for rl. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6864–6890, 2024.

Haitao Jiang, Lin Ge, Hengrui Cai, and Rui Song. Pabu: Progress-aware belief update for efficient llm agents. *arXiv preprint arXiv:2602.09138*, 2026.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024. URL https://arxiv.org/abs/2310.06770.

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.

Adam Kamel, Tanish Rastogi, Michael Ma, Kailash Ranganathan, and Kevin Zhu. Emergent world beliefs: Exploring transformers in stochastic games. *arXiv preprint arXiv:2512.23722*, 2025.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. Acon: Optimizing context compression for long-horizon llm agents. *arXiv preprint arXiv:2510.00615*, 2025.

Sherman Kent. Words of estimative probability (2nd edition). *Studies in Intelligence*, 8(4):49–65, 1964.

Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiaxin Huang. Taming overconfidence in llms: Reward calibration in rlhf. *arXiv preprint arXiv:2410.09724*, 2024.

Aly Lidayan, Jakob Bjorner, Satvik Golechha, Kartik Goyal, and Alane Suhr. Abbel: Llm agents acting through belief bottlenecks expressed in language. *arXiv preprint arXiv:2512.20111*, 2025.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

OpenAI. Introducing gpt-5.4, 2026. URL https://openai.com/index/introducing-gpt-5-4/.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Nikolai Rozanov and Marek Rei. Stateact: Enhancing llm base agents via self-prompting and state-tracking. In *Proceedings of the 1st Workshop for Research on Agent Language Models (REALM 2025)*, pages 367–385, 2025.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020a. URL https://arxiv.org/abs/1912.01734.

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*, 2020b.

Damien Sileo and Marie Francine Moens. Probing neural language models for understanding of words of estimative probability. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (* SEM 2023)*, pages 469–476, 2023.

David Silver and Joel Veness. Monte-carlo planning in large pomdps. *Advances in neural information processing systems*, 23, 2010.

Richard D Smallwood and Edward J Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations research*, 21(5):1071–1088, 1973.

Elias Stengel-Eskin, Peter Hase, and Mohit Bansal. Lacie: Listener-aware finetuning for calibration in large language models. *Advances in Neural Information Processing Systems*, 37:43080–43106, 2024.

Zhisheng Tang, Ke Shen, and Mayank Kejriwal. An evaluation of estimative uncertainty in large language models. *npj Complexity*, 3(1):8, 2026.

Sebastian Thrun. Monte carlo pomdps. *Advances in neural information processing systems*, 12, 1999.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, 2023.

Bob van Tiel, Uli Sauerland, and Michael Franke. Meaning and use in the expression of estimative probability. *Open Mind*, 6:250–263, 11 2022. ISSN 2470-2986. doi: 10.1162/opmi_a_00066. URL https://doi.org/10.1162/opmi_a_00066.

Ruiyi Wang and Prithviraj Ammanabrolu. A practitioner's guide to multi-turn agentic reinforcement learning, 2025. URL https://arxiv.org/abs/2510.01132.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*, 2023.

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

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024. URL https://arxiv.org/abs/2405.15793.

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

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*, 2025.

Zhenhang Yuan, Shenghai Yuan, and Lihua Xie. Rpms: Enhancing llm-based embodied planning through rule-augmented memory synergy. *arXiv preprint arXiv:2603.17831*, 2026.

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

Yibo Zhao, Jiapeng Zhu, Zichen Ding, and Xiang Li. Grace: Reinforcement learning for grounded response and abstention under contextual evidence, 2026. URL https://arxiv.org/abs/2601.04525.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

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

Deyu Zou, Yongqiang Chen, Jianxiang Wang, Haochen Yang, Mufei Li, James Cheng, Pan Li, and Yu Gong. Reducing belief deviation in reinforcement learning for active reasoning, 2026. URL https://arxiv.org/abs/2510.12264.

A Belief State Structure

In *TextWorld* environment, the belief state tracks five critical dimensions: (i) the agent's current location, (ii) topological room connections, (iii) states of observed objects, (iv) inventory contents, and (v) progress relative to specific sub-goals. To ensure a clean separation of concerns between the belief model and the policy, the belief state is strictly prohibited from generating plans, intentions, or hypothetical future actions. This ensures that b_k serves exclusively as an approximation of the current environment states, leaving the decision making to the policy model. Prompt 1 shows the full prompt used for generating the belief state.

B Belief Model SFT on Teacher Trajectories

We first perform SFT on the belief model f_ϕ using a teacher dataset $\mathcal{D}_{\text{teacher}}$. These trajectories are generated by GPT 5.4 mini using Prompt 1, which is prompted to perform state tracking. For each transition $(G, b_{t-1}, o_t) \to b_k$, we optimize the standard cross entropy loss:

$$\mathcal{L}_{\text{SFT}}(\phi) = -\mathbb{E}_{(g,b_{k-1},o_k,b_k)\sim\mathcal{D}_{\text{teacher}}}\left[\sum_t \log f_\phi(b_{k,t} \mid g, b_{k-1}, o_k, b_{k,<t})\right] \tag{2}$$

This stage is necessary to teach the belief model to utilize the Likert certainty scale and adhere to the structured representation.

To verify that the SFT phase in belief state model training only bootstraps structural formatting rather than transferring task knowledge, Fig. 6 tracks individual reward components across belief state model training (Appendix I). Format compliance remains consistently high (>0.96) from the first training step, while state correctness, diversity, and task rewards all begin near their lower bounds and rise steadily, confirming that belief quality is learned through RL, not inherited from the SFT teacher.

C Belief State without Task-Specific Supervision

To decouple the contribution of task-specific belief supervision, we run an additional ablation: `Agent-BRACE` without belief supervision. First, the belief state model is initialized directly from the base model (Qwen3-4B-Instruct) rather than from an SFT checkpoint trained on belief state annotations from GPT 5.4 mini, eliminating the supervised warm-start. Second, the belief state prompt template is made fully domain-agnostic: all game-specific entity names like room and object identifiers in the in-context examples are either removed or replaced with abstract placeholders, removing the domain-specific belief prompt (ref. Prompt 5 for full prompt). Under this configuration, the belief state model must bootstrap a meaningful world-state representation entirely through PPO training, guided only by the task reward signal and belief state model rewards.

As shown in Table 5, removing belief supervision leads to consistent degradation across all three tasks – average accuracy drops from 79.3% to 69.3%. Comparing against the full Qwen3-4B-Instruct baseline suite from Table 2, the ablation (- Belief supervision) at 69.3% average accuracy still outperforms PABU (62.7%), ReAct RL (54.2%), MEM1 (45.0%), ReAct (47.8%), and the Base Model (65.3%), falling only a little behind Direct-Action RL (74.0%) among trained baselines. Critically, the degradation here is moderate rather than catastrophic: the system is still able to retain substantial task-solving capability even without any belief supervision. This confirms that the SFT warm-start and domain-structured prompting are important catalysts for belief state model learning, but the joint RL training alone can provide a sufficiently strong training signal to teach structured world-state tracking from a general-purpose language prior. This result is encouraging for real-world applicability, where it is not always feasible to define task-relevant states.

D Reward Design

To train `Agent-BRACE`, we define two distinct reward signals: one for the belief state model (R^{belief}) and one for the policy model(R^{env}) – which are optimized jointly so that the belief representation is shaped by the policy's decision-making needs.

Table 5: Ablation of belief state supervision on Qwen3-4B-Instruct. `Agent-BRACE` vs. a variant with no SFT initialization and no domain-specific belief prompt structure (- Belief Supervision).

Method	Quest		Treasure		Cooking		Average	
	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓
`Agent-BRACE`	**88.0**	30.5	**81.0**	30.0	**69.0**	44.6	**79.3**	35.1
- Belief Supervision	79.0	34.9	74.0	38.1	55.0	62.4	69.3	45.1

Belief State Model Rewards (R_t^{belief}): As shown in Fig. 2, the belief state model is trained using a composite reward consisting of five signals. To show the importance of these rewards, we ablate belief state relevant rewards in Section 4.

- **State Tracking Reward** (r_t^{st}): This reward assesses the logical consistency of the belief update $b_{t-1} \to b_t$ given observation o_t. An LLM judge (ref. Prompt 2) assesses whether new information in o_k is incorporated in the updated belief. This reward counts, N_{new}: new facts in o_t correctly added; N_{missing}: new facts absent or wrong in b_t; N_{stale}: prior beliefs contradicted by o_t but left unchanged; N_{total}: total claims in b_t. The reward is the product of coverage of new information and freshness of retained beliefs: $r_t^{\text{st}} = \frac{N\text{new}}{N_{\text{new}}+N_{\text{missing}}} \times \left(1 - \frac{N_{\text{stale}}}{N_{\text{total}}}\right)$.
- **State Correctness Reward** (r_t^{sc}): This reward ensures that the belief state claim, along with its uncertainty score, remains grounded to the environment states. An LLM judge parses b_t into tuples $(subject, predicate, certainty)$ (ref. Prompt 3), then verifies each tuple against s_t, classifying it as fully correct, partially correct (the underlying fact is true, but the certainty label is miscalibrated) and incorrect (ref. Prompt 4). Therefore final $r_t^{\text{sc}} = \frac{N\text{correct}+0.5\times N_{\text{partial}}}{N_{\text{ver}}}$, where $N_{\text{ver}} = N_{\text{correct}} + N_{\text{partial}} + N_{\text{incorrect}}$.
- **Format Reward** (r_t^{format}): This reward assess adherence to the structured representation. To mitigate reward hacking and degradation of the structure, we apply a multiplicative gating signal that enforces these constraints, assigning zero reward to structurally invalid outputs.
- **Diversity Reward** (r_t^{div}): Each uncertainty score in the belief state is matched against an ordered keyword list (e.g., *confirmed* ≈ 1.0, *probable* ≈ 0.75, *possible* ≈ 0.50, *doubtful* ≈ 0.25, *unknown* ≈ 0.0) and mapped to one of 7 canonical levels. Shannon entropy $H = -\sum p_i \log p_i$ is computed over the resulting label distribution and normalized to $[0, 1]$. The reward is maximized when each individual claim's uncertainty is spread evenly across the WEP vocabulary and minimized when they collapse onto a single label.
- **Discounted Success Reward** (r_t^{success}) A time step-discounted success reward ($\gamma^t \times \mathbf{1}[\text{success}]$) is assigned to each belief state based on the agent's final task outcome.

The final reward assigned to a belief state is $R_t^{\text{belief}} = r_t^{\text{format}} \times \frac{1}{4}\left(r_t^{\text{st}} + r_t^{\text{sc}} + r_t^{\text{div}} + r_t^{\text{success}}\right)$. The format reward acts as a multiplicative gate, zeroing out all other rewards for structurally invalid outputs and preventing reward hacking.

Policy Model Reward (R^{env}): The policy model π_θ is optimized to maximize the policy reward, which is derived from the external environment. This signal (+1 for task success, 0 for task failure) provides the primary reinforcement for action selection, ensuring that the policy learns to take the next best action.

E Dataset Details

All tasks are built on the *TextWorld* environment, a procedural text-game generator that produces fully observable game graphs alongside natural-language descriptions. We construct three benchmark suites with controlled difficulty curricula: Quest, Treasure, and Cooking. All three suites share the same split sizes: 1,000 training games, 100 validation games, and 200 test games, generated with non-overlapping base seeds (10000 / 20000 / 30000, respectively). Training games span easier difficulty levels, while validation and test games sample progressively harder configurations to measure out-of-distribution generalization. We also extend our dataset to the ALFWorld environment.

E.1 Quest

Task. The agent must navigate a multi-room environment, locate and manipulate objects (keys, containers, doors), and collect a designated target object. Winning requires executing the full quest sequence in the correct order.

Generation. Games are generated with `tw-make custom`, parametrized by a `rooms:objects:questlength` triplet that directly controls world complexity. Training games cycle across four configurations (Table 6). Validation and test games cycle across five harder configurations ranging from 6 rooms / 6 objects / 8-step quests up to 8 rooms / 12 objects / 13-step quests.

Table 6: Basic (Quest) training configurations.

Configuration	Rooms	Objects	Quest Steps
2:3:3	2	3	3
2:4:4	2	4	4
4:4:4	4	4	4
4:6:6	4	6	6

E.2 Treasure

Task. The agent is placed inside a procedurally generated maze and must locate a named treasure object (*e.g.*, a latchkey) hidden in a random room. The objective is given in natural language at the start of each episode.

Generation. Games are generated with `tw-make tw-treasure_hunter -level` L, where the level integer (1–30) jointly governs world size, container nesting depth, and the number of distractor objects. Training games cycle across levels $\{1, 2, 4, 6, 8\}$ (easy band); Validation and test games cycle across levels $\{14, 16, 18, 20, 22, 25, 28, 30\}$.

E.3 Cooking

Task. The agent must find a recipe posted in a kitchen cookbook and execute it: navigate to relevant rooms, gather the required ingredients, apply the correct preparation steps (opening containers, cutting, cooking), and finally prepare and eat the meal.

Generation. Games are generated with `tw-make tw-cooking`, parametrized as `recipe:take:go:flags`, where `recipe` is the number of required ingredients, `take` is the number of objects to pick up, `go` is the number of rooms, and `flags` encode which mechanics are active: `o` (openable containers), `c` (cooking appliance), `t` (cutting board), `d` (limited inventory / drop required). Training configurations cycle across four settings (Table 7). Validation and test configurations cycle across five harder settings (4–5 ingredients, 9–12 rooms).

Table 7: Cooking training configurations.

Configuration	Rooms	Ingredients	Mechanics
3:3:9:oc	9	3	open, cook
4:4:6:oct	6	4	open, cook, cut
4:4:6:octd	6	4	open, cook, cut, drop
4:4:9:oct	9	4	open, cook, cut

E.4 ALFWorld

The training and testing datasets for ALFWorld are directly taken from Wang and Ammanabrolu [2025].

F Implementation Details and Hyperparameters

Our codebase is built on top of Wang and Ammanabrolu [2025] and all training is run on a single node with 4 NVIDIA GPUs (A100).

F.1 Models Used

To train and evaluate `Agent-BRACE`, we adopt instruction-tuned models, Qwen2.5-3B-Instruct[3] and Qwen3-4B-Instruct[4] as our base models. Additionally, we use Qwen3-30B-A3B-Instruct-2507[5] as the judge to evaluate state tracking and correctness reward.

F.2 Belief State Model: Supervised Pre-training (SFT)

Before joint RL training, the belief-state LM is fine-tuned with supervised learning on belief-state trajectories generated from teacher demonstrations. Hyperparameters are summarised in Table 8.

Table 8: Belief-state SFT hyperparameters.

Hyperparameter	**Value**
Base model	Qwen3-4B-Instruct
Max sequence length	4096
Training epochs	5
Per-device batch size	4
Gradient accumulation steps	8
Effective batch size	32
Learning rate	2×10^{-5}
Precision	BF16
LoRA rank (r)	64
LoRA α	128
Save steps	100

F.3 Policy PPO Training

The policy is trained with Proximal Policy Optimization (PPO) using a Generalized Advantage Estimation (GAE) critic. Table 9 lists the PPO hyperparameters used.

F.4 Joint Belief-State PPO Training

During RL, the belief-state LM is updated jointly with the policy after every rollout batch. Belief-model training hyperparameters are given in Table 10.

G Belief States are underconfident but improve over training

Fig. 5 analyzes the calibration of WEP labels at early (steps 0-4) and late (steps 10-15) training stages. For each WEP label emitted by the belief state model, we measure the empirical truth rate – the fraction of claims carrying that label that are independently verified as true by the LLM judge (Qwen3-30B-A3B-Instruct-2507) and compare it against the *nominal probability* that the label represents on the WEP scale (e.g., *confirmed* $\approx$ 1.0, *probable* $\approx$ 0.75, *possible* $\approx$ 0.50, *doubtful* $\approx$ 0.25, *unknown* $\approx$ 0.0; shown in gray). A perfectly calibrated belief model would have these match exactly; a model whose empirical truth rate consistently exceeds the nominal probability is underconfident; it assigns lower-confidence labels to beliefs that are in fact more often true than those labels suggest (e.g., labeling something *possible* when it is actually true 84% of the time). Fig. 5 shows that this is precisely what occurs: at early training, claims labeled *unknown* are verified true

[3] https://huggingface.co/Qwen/Qwen2.5-3B-Instruct
[4] https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507
[5] https://huggingface.co/Qwen/Qwen3-30B-A3B-Instruct-2507

Table 9: Policy PPO hyperparameters (shared across environments).

Hyperparameter	Value
Algorithm	
Advantage estimator	GAE
Discount factor γ	1.0
KL penalty coefficient	0.01
Clip ratio ε	0.2
Optimization	
Actor learning rate	1×10^{-6}
Critic learning rate	1×10^{-5}
Train batch size	16
PPO mini-batch size	16
Rollout	
Rollout temperature	0.7
Validation temperature	0.4
Rollout samples per prompt (n)	1
Policy tensor-parallel size	2
Maximum episode steps (training)	15
Maximum episode steps (evaluation)	100
Reward shaping	single (terminal)

Table 10: Joint belief-state RL hyperparameters.

Hyperparameter	Value
Optimization	
Belief LM learning rate	1×10^{-5}
Gradient update steps per batch	1
PPO mini-batch size	4
PPO clip ratio ε	0.2
Dual-clip upper bound c	3.0
Policy loss mode	vanilla PPO
Value function	disabled
Inference (vLLM subprocess)	
Max generation tokens	1,024
vLLM max model length	4,096
vLLM max batched tokens	4,096

68% of the time, far above the nominal probability of approximately 0, shrinking to 54% by late training. This indicates the model increasingly reserves *unknown* for genuinely uncertain claims rather than as a default label. At the high-confidence end, *almost certain* and *confirmed* remain well calibrated throughout ($\geq$91% and $\geq$95% respectively). In sequential decision making under partial observability, underconfidence is a safer failure mode than overconfidence [Stengel-Eskin et al., 2024]: an agent that hedges will continue to explore and gather evidence, whereas an overconfident agent risks committing to an incorrect world model and acting on it irreversibly.

H Statistical Reliability of Main Results

To assess the statistical reliability of `Agent-BRACE`, we report mean accuracy and standard deviation across three independent runs for all methods trained on Quest and evaluated on all three TextWorld environments. Table 11 reports the result for Qwen3-4B-Instruct across three independent runs.

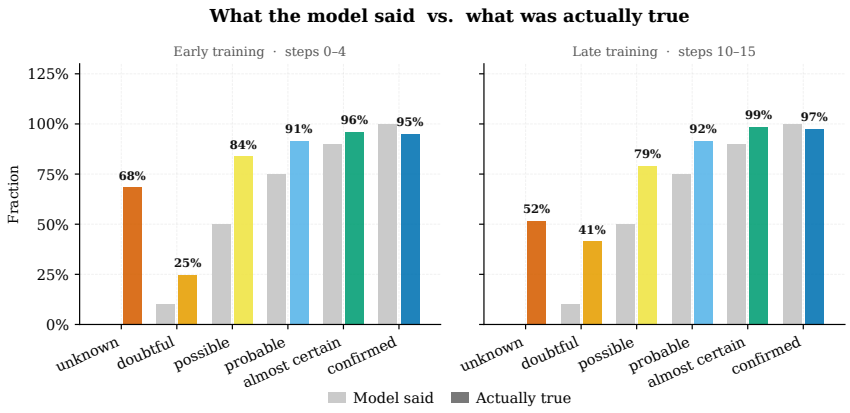

Figure 5: Calibration of WEP labels at early (0-4) and late (10-15) steps. For each WEP label emitted by the belief model, the colored bar shows the fraction of those claims independently verified as true, while the grey bar shows the nominal probability that the label represents on the WEP scale. The belief model is systematically underconfident; it assigns conservative labels to beliefs that are more often true than the label implies.

Table 11: Performance comparison of `Agent-BRACE` against baselines across three TextWorld environments (Quest, Treasure, Cooking) on Qwen3-4B-Instruct with standard deviation across three independent runs.

Method	Quest		Treasure		Cooking	
	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓
Base Model	61.5 ± 0.0	32.2 ± 0.1	66.0 ± 0.9	29.9 ± 0.3	69 ± 1.4	33.9 ± 0.2
ReAct	61.0 ± 0.6	12.8 ± 0.2	69.5 ± 0.9	10.0 ± 0.2	13.2 ± 0.3	24.4 ± 0.0
Direct-Action (RL)	74 ± 0.8	29.8 ± 0.7	72.5 ± 0.3	28.1 ± 0.1	76 ± 0.6	31.5 ± 0.3
ReAct (RL)	75.5 ± 0.6	18.3 ± 0.2	74.0 ± 0.0	16.5 ± 0.0	13.0 ± 0.0	40.6 ± 0.0
PABU	83.0 ± 0.9	26.4 ± 2.4	70.7 ± 3.0	37.8 ± 1.4	32.5 ± 0.3	72.1 ± 3.1
`Agent-BRACE`	88 ± 0.9	30.8 ± 0.5	81 ± 0.6	29.9 ± 0.2	68.7 ± 0.6	44.7 ± 0.2

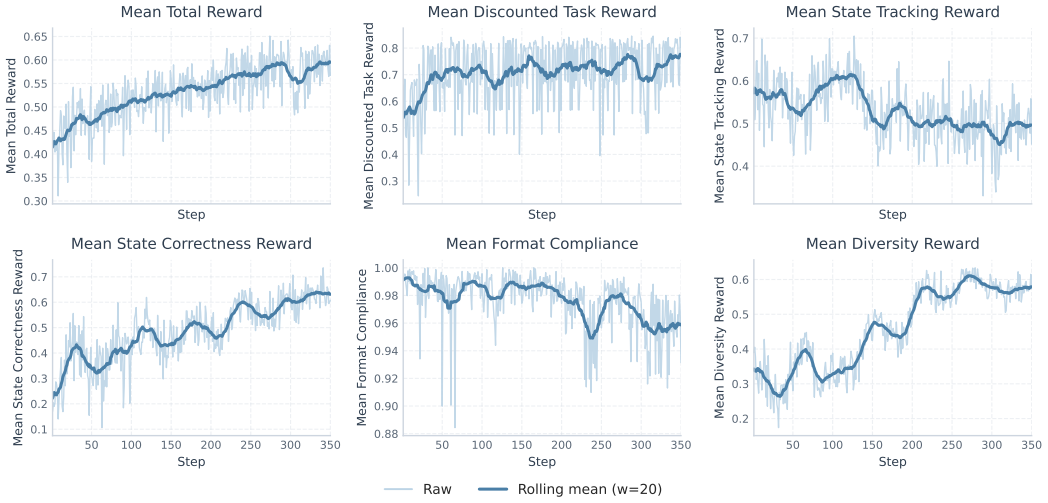

Figure 6: Reward component trajectories across PPO training steps for the belief state model (Qwen3-4B-Instruct).

I Belief State Model Training

Fig. 6 plots the five reward components along with the mean total reward across belief state model training steps. If the SFT cold start phase were distilling task knowledge from GPT 5.4 mini, we would expect state correctness and total task reward to begin high and plateau quickly, with minimal learning signal during PPO. Instead, we observe the opposite: format compliance starts near 0.98 and remains consistently high throughout training, confirming that SFT successfully bootstrapped structured adherence to the WEP-annotated belief format. In contrast, state correctness begins near 0.2 and rises steadily to 0.65, diversity reward rises from 0.3 to 0.65, and similarly discounted task reward from 0.55 to 0.78. This pattern demonstrates that factual belief quality, uncertainty diversity, and task performance are learned through RL interaction with the environment, not inherited from the teacher model. The SFT phase contributes only to the format; all substantive learning is attributed to the RL training signal.

J `Agent-BRACE` without state correctness and tracking reward

Table 12 ablates the LLM judged belief state rewards by setting r_t^{st} and r_t^{sc} to zero while keeping all other components fixed. Removing both the rewards degrades the average accuracy by 4% (79.3% $\rightarrow$ 75.3%). Notably, on average the ablation still outperforms all baselines reported in Table 2, demonstrating that the removal of LLM judge reward does not catastrophically degrade the performance of `Agent-BRACE`.

Table 12: Ablation of LLM judged based belief state rewards. `Agent-BRACE` vs. a variant with no State correctness and tracking reward.

Method	Quest		Treasure		Cooking		Average	
	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓	Acc.↑	Steps↓
`Agent-BRACE`	**88.0**	30.5	**81.0**	30.0	69.0	44.6	**79.3**	35.1
- r_t^{st} and r_t^{sc} reward	72.0	40.2	72.0	36.8	**82.**0	33.6	75.3	36.8

K Qualitative Examples

To illustrate the properties of our jointly trained belief state, we present representative examples from our method on cooking tasks.

Example 1 illustrates calibrated uncertainty and its correct resolution: after a single `open plain door` action, the south exit transitions from *possible* to *confirmed*, while genuinely unknown quantities such as the cookbook location and shelf contents remain marked as unknown rather than being hallucinated.

Example 2 demonstrates that the belief state tracks world-state changes without over-committing to unseen information. After opening the sliding patio door, the model correctly upgrades that exit to *confirmed open*, while unvisited north and west exits remain *possible* with "destination not yet observed" — reflecting the agent's actual epistemic boundary.

Example 3 shows fine-grained inventory tracking across multiple items. The belief state simultaneously records five items with exact processing descriptors (*chopped burned*, *sliced burned*) while correctly deferring on recipe requirements and counter contents, both marked *unknown* pending cookbook consultation.

Prompt 1: Belief State Generation/Update

You are playing a text-based game. Given the goal, your previous belief state, and the current observation, produce an updated belief state capturing what you know and how confidently you know it.

Goal: `{goal}` **Previous belief state:** `{previous_belief_state}` **Current observation:** `{current_obs}`

Output **ONLY** a belief state within `<belief_state>` `</belief_state>` tags.

— STRICT RULES —

Your belief state **MUST NOT** contain: (i) any next action, plan, or intention; (ii) forward-looking phrases: "I will", "I should", "my next step", "I plan to", etc.; (iii) any recommendation about which command to execute. **ONLY** record what you have already observed or can directly infer from past observations.

— UPDATE RULES —

- If the current observation **CONFIRMS** a previous bullet → upgrade it to "confirmed".
- If the current observation **CONTRADICTS** a previous bullet → replace it immediately.
- Never carry forward a stale bullet that conflicts with a direct observation.

— CERTAINTY SCALE —

Every bullet **MUST** contain exactly one of these markers, used naturally in the sentence:

confirmed / certain	directly observed this turn
almost certain	observed previously, no contradicting evidence since
probable	inferred from goal structure or strong pattern
possible	no visit yet, some contextual reason to believe
unlikely	visited nearby rooms, no supporting evidence found
doubtful	contradicting evidence exists
unknown	no evidence

— MANDATORY COVERAGE —

You **MUST** cover each of the following in at least one bullet: current location; each known exit and where it leads; each goal-relevant object (its location and state); your inventory; progress toward each sub-goal.

— FORMAT —

One bullet per distinct fact, starting with "- ". Every bullet contains exactly one certainty marker. No JSON, no percentages, no key-value pairs.

Examples:

- It is *confirmed* that I am in the kitchen.
- The east exit *almost certainly* leads to the hallway based on prior exploration.
- The key is *probably* still in the living room where I last saw it.
- It is *possible* the chest in the bedroom contains the goal item, though I have not visited.
- The couch is *ruled out* from the bedroom – I visited and did not observe it there.
- It is *doubtful* the garden door is unlocked given every other door here has been locked.
- It is *confirmed* that I am carrying only the brass key.

Prompt 2: State-Tracking Reward

You are evaluating a belief-state update in a text-based game.

Previous belief state: `{prev_belief}` **New observation:** `{new_obs}` **New belief state:** `{new_belief}`

Step 1 — Identify (brief, one line each):

- **New facts:** list each distinct fact the observation reveals (e.g. "player moved to kitchen", "door is open")
- **Missing:** which of those new facts are absent or wrong in the new belief state
- **Stale:** which prior beliefs does the observation contradict that were left unchanged

Step 2 — Count (integers):

- N_{new} = number of new facts correctly captured
- N_{missing} = number of new facts missing or wrong
- N_{stale} = number of stale/contradicted priors left unchanged
- N_{total} = total claims in the new belief state

Step 3 — Compute:

- If $N_{\text{total}} = 0$: score $= 0.0$
- Otherwise:

$$\text{coverage} = \frac{N_{\text{new}}}{\max(1,\ N_{\text{new}} + N_{\text{missing}})}$$

$$\text{staleness} = \frac{N_{\text{stale}}}{N_{\text{total}}}$$

$$\text{score} = \text{coverage} \times (1 - \text{staleness}) \quad \text{clamped to } [0.00, 1.00]$$

End with exactly `<score>X.XX</score>` where `X.XX` is a decimal in $[0.00, 1.00]$.

Prompt 3: Claim Extraction

You are analysing a belief state from a text-based game.

Belief state: `{belief_state}`

Task — Extract every specific factual claim from the belief state. List each claim on its own line using this exact format:

`CLAIM: <subject> | <predicate> | <certainty-label>`

Examples:

- `CLAIM: player location | in the kitchen | certain`
- `CLAIM: key | on the table in the library | probable`
- `CLAIM: east exit from kitchen | leads to hallway | almost certain`
- `CLAIM: chest | open | possible`

List **ALL** claims now (one per line):

Prompt 4: State Correctness

You are verifying factual claims from a belief state against the true game world state.

True game world state (ground truth): `{raw_state}` **Claims to verify:** `{claims}`

Instructions — For each claim decide:

- **Correct** — the underlying fact is true *and* the certainty label is appropriate.
- **Incorrect** — the underlying fact is false (label does not matter).
- **Partially correct** — the fact is true but the certainty label is badly miscalibrated (e.g. marked *certain* for something only *probable* is supported by evidence, or *unknown* for something directly observable in the true state).
- **Unverifiable** — the ground truth does not contain enough information to confirm or deny the claim.

Scoring — Let:

- $N_{\text{verifiable}} = \text{Correct} + \text{Incorrect} + \text{Partially correct}$
- $N_{\text{correct}} = \text{number of Correct verdicts}$
- $N_{\text{partial}} = \text{number of Partially correct verdicts}$

$$\text{score} = \frac{N_{\text{correct}} + 0.5 \times N_{\text{partial}}}{N_{\text{verifiable}}} \qquad (N_{\text{verifiable}} = 0 \Rightarrow \text{score} = 0.0)$$

Provide a brief per-claim verdict, then end with exactly `<score>X.XX</score>` where `X.XX` is a decimal in $[0.00, 1.00]$.

Prompt 5: Belief State Generation/Update (No Belief Supervision)

You are playing a text-based game. Given the goal, your previous belief state, and the current observation, produce an updated belief state capturing what you know and how confidently you know it.

Goal: `{goal}` **Previous belief state:** `{previous_belief_state}` **Current observation:** `{current_obs}`

Output **ONLY** a belief state within `<belief_state>` `</belief_state>` tags.

— STRICT RULES —

Your belief state **MUST NOT** contain: (i) any next action, plan, or intention; (ii) forward-looking phrases: "I will", "I should", "my next step", "I plan to", etc.; (iii) any recommendation about which command to execute. **ONLY** record what you have already observed or can directly infer from past observations.

— UPDATE RULES —

- If the current observation **CONFIRMS** a previous bullet → upgrade it to "confirmed".
- If the current observation **CONTRADICTS** a previous bullet → replace it immediately.
- If an expected element is not observed when new information is obtained → downgrade its likelihood.
- Never carry forward a stale bullet that conflicts with a direct observation.

— CERTAINTY SCALE —

Every bullet **MUST** contain exactly one of these markers:

confirmed / certain	directly observed this turn
almost certain	observed previously, no contradicting evidence since
probable	inferred from prior observations or consistent patterns
possible	no direct evidence either way
unlikely	weak or missing supporting evidence
doubtful	contradicting evidence exists
unknown	no evidence

— FORMAT —

One bullet per distinct fact, starting with "- ". Every bullet contains exactly one certainty marker from the scale above. No JSON, no percentages, no key-value pairs.

Examples:

- It is *confirmed* that I am in a situation where a surface and an object are present.
- An element previously observed is *almost certainly* still present as there is no contradicting evidence.
- There is a *probable* relationship between two elements based on prior observations.
- It is *possible* that additional elements exist that have not yet been observed.
- An expected element is *ruled out* in the current context as it is not observed.
- It is *doubtful* that a previously assumed property holds given new evidence.
- It is *confirmed* that I am associated with an element obtained earlier.

Example 1: Uncertainty correctly encoded and resolved

Two consecutive steps as the agent opens a door in the pantry (`game cooking_30003`).

Step A — *Observation:*

> You are now in the pantry. A normal one. What's that over there? It looks like it's a shelf. The shelf is wooden. However, the shelf, like an empty shelf, has nothing on it. There is a closed plain door leading south. You are carrying nothing.

Belief state:

- It is *confirmed* that I am in the pantry.
- The south exit is *possible* through the closed plain door, as indicated by the door description.
- The shelf is *probable* in the pantry, currently empty as observed.
- Whether the shelf contains any items is *unknown*.
- It is *confirmed* that I am carrying nothing.
- The location of the cookbook is *unknown*, as it has not been observed in the pantry.
- Whether the recipe is accessible from here is *unknown*.

Action: `open plain door` ✓

Step B — *Observation:*

> You are now in the pantry. You open plain door. You are carrying nothing.

Belief state:

- It is *confirmed* that I am in the pantry.
- The south exit is *confirmed* by the open plain door leading south.
- The shelf is *probable* in the pantry, currently empty as observed.
- Whether the shelf contains any items is *unknown*.
- It is *confirmed* that I am carrying nothing.
- The location of the cookbook is *unknown*, as it has not been observed here.
- It is *confirmed* that the plain door is open, as directly observed.

Action: `go south` ✓

Example 2: Belief state tracks world-state changes across actions

The agent navigates through a corridor with a sliding patio door (`game cooking_30002`).

Step A — *Observation:*

> You are now in the corridor. A normal one. There is a closed sliding patio door leading east. There is an exit to the north. Don't worry, there is no door. You need an exit without a door? You should try going south. There is an exit to the west. You are carrying nothing.

Belief state:

- It is *confirmed* that I am in the corridor.
- The east exit is *possible* through a closed sliding patio door, though it is currently closed and not open.
- Whether the sliding patio door can be opened is *unknown*, as it has not yet been interacted with.
- The north exit is *possible*, destination not yet observed.
- The west exit is *possible*, destination not yet observed.
- It is *confirmed* that I am carrying nothing.

Action: `open sliding patio door` ✓

Step B — *Observation:*

> You are now in the corridor. You open sliding patio door. You are carrying nothing.

Belief state:

- It is *confirmed* that I am in the corridor.
- The east exit is *possible* through an open sliding patio door, since it is explicitly open.
- The sliding patio door is *confirmed* open, as directly observed after opening it.
- The north and west exits are *possible*, destinations not yet observed.
- It is *confirmed* that I am carrying nothing.

Action: `go east` ✓

Example 3: Multi-item inventory tracking with fine-grained state labels

After several cooking steps, the agent holds five distinct items (`game cooking_30003`).

Observation:

> You are now in the kitchen. You take the red apple from the counter. You are carrying: a red apple, a knife, a chopped burned red hot pepper, a sliced burned yellow potato and a cookbook.

Belief state:

- It is *confirmed* that I am in the kitchen.
- The north open plain door is *confirmed*.
- The red apple is *confirmed* in my inventory, taken from the counter.
- The chopped burned red hot pepper is *confirmed* in my inventory.
- The sliced burned yellow potato is *confirmed* in my inventory.
- The cookbook is *confirmed* in my inventory.
- The knife is *confirmed* in my inventory.
- Whether any items remain on the counter is *unknown*.
- The required preparation state of the red apple is *unknown*, as the cookbook has not yet been read.

Action: `slice red apple with knife` ✓