GROUNDED IN IMAGINATION: BELIEF-GUIDED EMBODIED TASK PLANNING IN UNCERTAIN, DYNAMIC ENVIRONMENTS

Anonymous authors
Paper under double-blind review

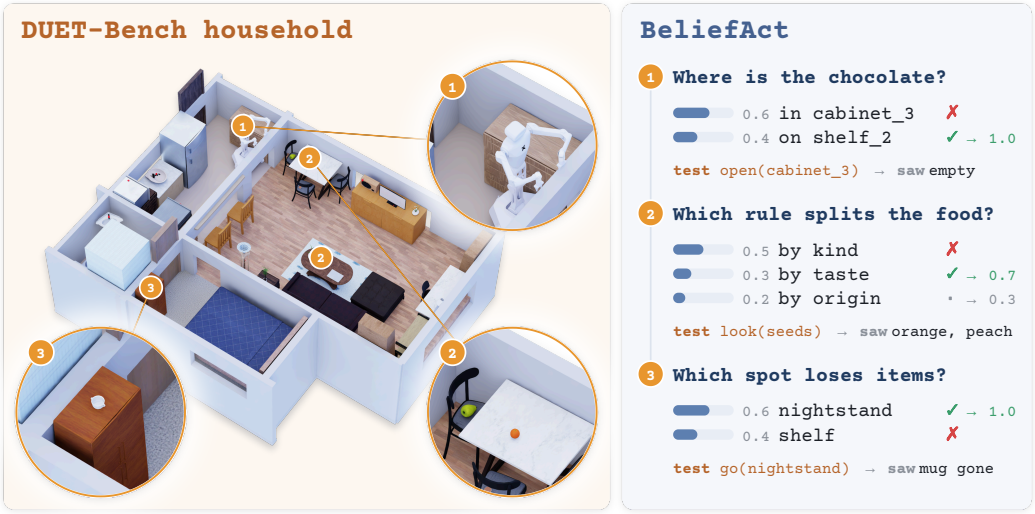

Figure 1: **Grounded in imagination: beliefs for uncertain, dynamic environments.** DUET-Bench introduces uncertainty about object locations (1), the intended food splitting rule (2), and changes outside the robot's view (3), illustrated on the left. BeliefAct represents these task-relevant unknowns as free-form questions with competing, probability-weighted answers (right). The agent uses what it observes after opening a cabinet, inspecting example foods, or revisiting a storage spot to revise its beliefs and choose what to do next.

ABSTRACT

A robot in a household faces heterogeneous uncertainty: unknown object locations, unstated arrangement preferences, and a housemate who moves things by unclear habits. Such settings require planning that is *grounded in imagination*: driven not only by the observed state, but also by beliefs about task-relevant unknowns. This leads to two challenges: identifying and representing the task-relevant unknowns among the unbounded possibilities of an open world and revising them from new observations. Existing works pre-specify both the state space and the update rules, and so neither scale nor transfer. We propose BeliefAct, an LLM-based planning framework that identifies and represents task-relevant beliefs as free-form questions with probability-weighted answers, a representation that generalizes beyond a fixed set. To facilitate belief updates, it pairs each hypothesis with a test action that would settle it and the outcome expected if it holds, compares them against observations to reweight hypotheses. To systematically test planning under heterogeneous uncertainty, we introduce DUET-Bench, a benchmark built on BEHAVIOR-1K with eight household task families, where the initial configuration, the goal interpretation, or an external process is uncertain. BeliefAct achieves the highest overall success rate at the fewest rounds and the lowest interaction cost. It also transfers to two-agent Communicative Watch-and-Help, matching task completion with about 60% fewer LLM calls.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Agentic robotics pairs the semantic knowledge of generalist language models with a repertoire of pre-trained robot skills, decomposing novel long-horizon tasks into sequences of skill executions (Ichter et al., 2023; Huang et al., 2023; Zhang et al., 2024b). This works well when when all information required for planning is available, but household environments rarely provide it. A robot asked to tidy a living room must infer the grouping its owner intends from the few items already put away. Seeing the cat wander in, it should clear the dishes off the coffee table first, before they end up on the floor. In both cases, planning solely from concrete observations and explicit instructions is insufficient. Planning should instead be *grounded in imagination*: the agent should reason explicitly about what could be true beyond what it has seen, or could become true while it acts, and use its own interactions to decide which hypotheses to retain (Figure 1). We believe language models already carry the commonsense priors such reasoning requires: where the remote is likely kept, what a pair of seed items suggests about the intended grouping, and what a cat is likely to do to a stack of dishes.

However, turning those priors into a usable belief raises two questions. First, which unknowns matter, and how should they be represented? The space of uncertainty is unbounded, while only a fraction of it affects the task. The relevant facts differ across tasks, e.g., an intended grouping, or a cat's habits, and can shift within a single task as it progresses. Existing belief representations sidestep this issue by fixing the question ahead of time, committing to one kind of uncertainty and a belief format built for it, such as a distribution over an enumerated set of receptacles or a hypothesis about a partner's intent. Such representations do not readily transfer when the relevant unknown is of another kind.

Second, how should a belief be maintained as the task progresses? As the robot moves and the scene changes, observations accumulate and shift the plausibility of competing hypotheses. Not finding a mug in one cabinet redistributes belief over the rest, while seeing the cat on the coffee table strengthens the hypothesis that dishes there are at risk. The robot may also take actions to resolve uncertainties. Where multiple hypotheses are maintained, prior work either regenerates them from scratch at each step (Seo et al., 2026) or revises them using predetermined rules, such as eliminating states ruled out by an observation (Kumar et al., 2026; Wang et al., 2026) or combining evidence with a fixed update formula (Tang et al., 2025), which is not a generalizable solution.

To answer these questions, we propose BeliefAct, a planning framework that identifies and represents heterogeneous task-relevant beliefs, and revises them from what the agent observes as it acts. The agent represents uncertainty as free-form questions paired with probability-weighted answers. Questions single out the few unresolved aspects that bear on the next decision, phrased in whatever terms the task requires rather than drawn from a fixed schema, while weighted answers approximate a distribution over the possibilities instead of committing to the most likely reading (Wang et al., 2026). Each hypothesis is further paired with a test action that would resolve it and the outcome expected if it holds. In other words, the agent writes the transition rules for its own belief on the fly, just as it writes the belief itself. At each step, the agent compares what its hypotheses predicted against what it actually observed, reweights them by reasoning about what the observation implies for each, and revises the question set itself, posing new questions as the task progresses and setting aside those that no longer bear on it. Because each hypothesis names the action that would settle it, the agent can also act for information rather than progress, choosing to resolve a question before it acts on the answer.

Existing benchmarks each focus on specific aspects of uncertainty: object locations (Shridhar et al., 2021; Tang et al., 2025), goal requirements (Kant et al., 2022; Wu et al., 2023; Ramachandruni & Chernova, 2025), dynamic changes introduced during execution (Yin et al., 2025; Yuan et al., 2026), possibly from another agent (Puig et al., 2021; Zhang et al., 2024b; Chang et al., 2025). To systematically evaluate an agent's ability to handle heterogeneous uncertainty, we introduce DUET-Bench, a dynamics- and uncertainty-aware embodied task planning benchmark built in BEHAVIOR-1K scenes (Li et al., 2023), with the source of uncertainty spanning the initial configuration, the goal interpretation, and the external processes.

Across the 240 episodes in DUET-Bench, BeliefAct achieves the highest overall success rate (86.7%) and the lowest mean rounds and interaction cost among the evaluated methods. Its round savings over the best baseline increase from 8% overall to 11% on tasks with hidden goal rules or external dynamics. Ablations of question formulation, competing answers, and belief transition consistently lower success and increase both rounds and cost. The same belief interface also supports two-agent cooperation on Communicative Watch-and-Help (C-WAH) (Zhang et al., 2024b), with 60% fewer

LLM calls. In summary, our contributions are a free-form QA belief representation, its interaction-grounded transition, and DUET-Bench for evaluating planning under heterogeneous uncertainty.

2 Related Work

Belief Modeling for Embodied Task Planning Most LLM-based embodied task planners act directly from the current context (Yao et al., 2023; Shinn et al., 2023; Huang et al., 2023; Ichter et al., 2023; Liang et al., 2023). A growing line of work instead argues that planning benefits from an explicit belief. Within this line, some methods maintain only a single most-likely estimate (Yoneda et al., 2024; Rozanov & Rei, 2024; Kim et al., 2024; Chen et al., 2023), while others maintain a distribution over multiple candidate states (Tang et al., 2025; Liao et al., 2026; Singh et al., 2026; Kumar et al., 2026). In multi-hypothesis belief-based planning, some works use learned models to obtain belief distributions over a pre-declared set (Zhao et al., 2025; Azran et al., 2026; Wang et al., 2026; Zhao et al., 2023), then hand updating and planning to a symbolic solver, which must be hand-specified per domain and does not generalize to unseen predicates or object types. Kumar et al. (2026) exposes the belief to LLMs for planning, but the belief itself remains over an pre-declared set. Singh et al. (2026) and Liao et al. (2026) avoid this rigidity by generating beliefs in natural language, but the former forces a template on the belief, and the latter only generate beliefs in reaction to an observation. Seo et al. (2026) extracts genuinely free-form hypotheses from the planner's own reasoning traces, but does not maintain them across steps, regenerating them from scratch at every step. In contrast, our approach maintains a belief expressed as free-form, task-relevant question-answer pairs, revised across the episode by the LLM itself rather than a fixed rule.

Embodied Task Planning in Dynamic Environments and Multi-Agent Cooperation To operate in dynamic environments with external changes, existing embodied task planning approaches primarily detect and react to change after it occurs, rather than anticipating it beforehand (Yan et al., 2025; Yin et al., 2025). Such anticipation appears instead in the multi-agent literature, where the source of change is an agent whose behavior can be modeled and predicted (Puig et al., 2021; 2023; Zhang et al., 2024a; Cross et al., 2024; Zhang et al., 2025). These methods maintain an explicit hypothesis about a partner's goal (Puig et al., 2021; 2023) or evolving strategy (Zhang et al., 2024a; Cross et al., 2024; Zhang et al., 2025), revising it based on the partner's observed actions. Most methods collapse belief to a single estimate (Zhang et al., 2024a; Cross et al., 2024; Zhang et al., 2025; Seo et al., 2025), with few exceptions such as NOPA (Puig et al., 2023), which maintains a weighted set over candidate beliefs. However, the belief always only concerns the partner's intent. An exception, CoBel-World (Wang et al., 2025), unifies physical-state and partner-mental-state belief within a single representation, but commits to one deterministic value without representing competing hypotheses. Our approach makes no such assumption on the type of uncertainties, extending the same free-form, LLM-generated belief used for static uncertainty to the dynamic, multi-agent case.

Benchmarks for Embodied Planning under Uncertainty Existing embodied task planning benchmarks mostly focus on specific aspects of uncertainty. Some hide where objects are in a partially observable setting (Shridhar et al., 2021; Tang et al., 2025); others leave the intended arrangement to be inferred from commonsense or user preference (Kant et al., 2022; Wu et al., 2023; Ramachandruni & Chernova, 2025). A separate line introduces dynamic changes, but the changes are arbitrary perturbations rather than the effects of a consistent underlying process, leaving detection and recovery as the only available response (Yin et al., 2025; Yuan et al., 2026; Wang & Li, 2026; Chen et al., 2026). Multi-agent benchmarks add a second actor, but focus on coordination rather than uncertainty, inferring intended division of labor and confirming via communication (Puig et al., 2021; Zhang et al., 2024b; Chang et al., 2025; Padmakumar et al., 2022). No existing benchmark therefore evaluates one agent under heterogeneous uncertainty, spanning the initial configuration, the goal interpretation, and the external processes that change the world.

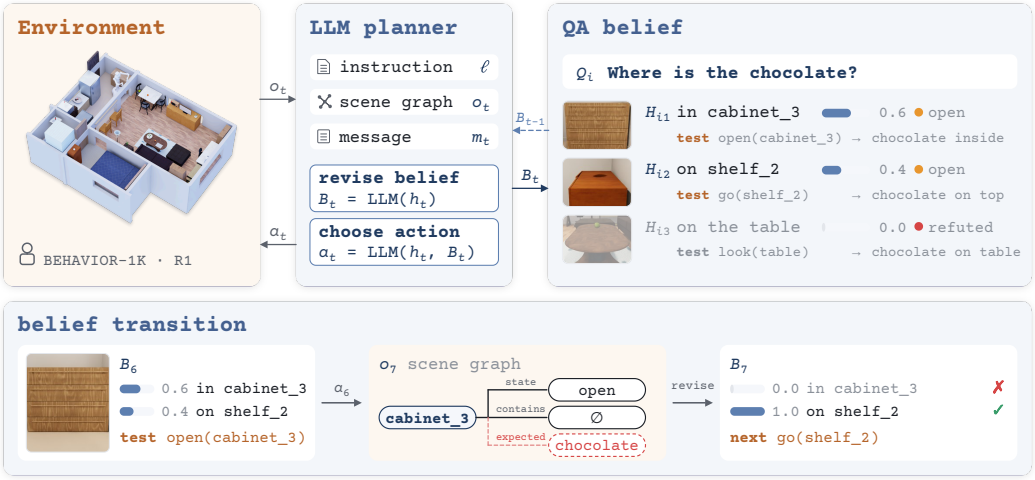

Figure 2: **BeliefAct updates beliefs to guide each action.** *Top:* Given the task instruction, current observation, and retained context, the LLM updates $\mathcal{B}_{t-1}$ to $\mathcal{B}_t$ and selects a_t in a single call. Each question retains competing answers with probabilities, test actions and expected observations, and statuses indicating whether they are open, supported, or refuted. *Bottom:* Opening the cabinet reveals it is empty, contradicting the expected presence of chocolate. The model refutes the cabinet hypothesis, shifts probability to the shelf, and chooses to visit it next.

3 METHOD

We propose BeliefAct, an uncertainty-aware embodied task planning framework that guides an LLM task planner with a persistent belief of the task-relevant unknowns in the environment (Fig. 2). We represent the belief with a free-form question-answer (QA) set (Section 3.2), which is grounded in expected interaction outcomes and explicitly transitioned in each round (Section 3.3).

3.1 PROBLEM DEFINITION

We formulate the problem of high-level embodied planning as a finite-horizon Partially Observable Markov Decision Process (POMDP). The agent receives partial observations of the environment, with unseen object states, task completion conditions, and dynamic changes being potentially unknown.

At round t, the agent takes in a language context $h_t = (\ell, o_t, s_t, a_{0:t-1}, m_{t-1}, \mathcal{A}_t)$, with ℓ denoting the task instruction, o_t the observation in the form of a cumulative scene graph (Rosinol et al., 2020) containing observed objects and their spatial relations, s_t the agent's current ego status, and $a_{0:t-1}$ the history action trajectory. m_{t-1} is a message the agent wrote to itself at the previous step, enabling retaining information across rounds within the episode. $\mathcal{A}_t$ is the set of currently valid actions supplied by the environment. Each $a \in \mathcal{A}_t$ is an observation, navigation, or manipulation primitive annotated with its cost c_a. The agent needs to output a valid action $a_t \in \mathcal{A}_t$ with an associated cost of c_{a_t}, and a message m_t to pass on useful information. The objective is to complete the task within H rounds while limiting the cumulative cost $c_\Sigma = \sum_t c_{a_t}$.

$$(a_t, m_t) = \text{LLM}(h_t) \tag{1}$$

The observation and execution interfaces are specified in Appendix C.

3.2 QA-BASED BELIEF REPRESENTATION

We design our belief representation to optimize for flexibility and scalability in an open world with heterogeneous uncertainty: it should apply to unknowns of diverse natures, and represent only those that affect the current decision. A fixed form such as a probabilistic scene graph (Zhao et al., 2023) that enumerates every object in the scene satisfies neither. To this end, we propose a QA-based belief, prompting the LLM to pose and answer its own questions in natural language, eliciting its world knowledge (Fung et al., 2025) in a form suited to its decision making.

At round t, we define the QA-based belief as the following collection:

$$\mathcal{B}_t = \Big((Q_i, \{(H_{ij}, p_{t,ij})\}_j), \{\mathcal{T}_{t,ij}\}_{j \in \mathcal{J}_{t,i}} \Big)_{i=1}^{N_t}, \quad \mathcal{T}_{t,ij} = (\tau_{t,ij}, e_{t,ij}). \tag{2}$$

Here, Q_i denotes a question about an unknown aspect of the world, H_{ij} a hypothesized answer, and $p_{t,ij}$ its probability at round t. The test $\mathcal{T}_{t,ij}$ pairs an available action $\tau_{t,ij}$ with the observation $e_{t,ij}$ expected after that action if H_{ij} is true. The N_t retained questions constitute $\mathcal{B}_t$. The index j ranges over the answers retained for Q_i, while $\mathcal{J}_{t,i}$ indexes its current candidate answers. For each nonempty $\mathcal{J}_{t,i}$, probabilities satisfy $p_{t,ij} \geq 0$ and $\sum_{j \in \mathcal{J}_{t,i}} p_{t,ij} = 1$. Details are specified in Appendix B.2.

We leave question wording and the evolving question set to the model, while prompting it to address any success-critical hidden factor stated in the task instruction from the first round. Common questions can concern unseen object states (e.g., "Where is the bottle of face cream?"), task completion conditions (e.g., "Which food belongs to the intended meal set?"), and dynamic changes (e.g., "Which storage spot loses items while unobserved?"). The model chooses these questions from the task and available evidence, including any success-critical hidden factor identified by the task instruction ℓ.

For each hypothesis, the model describes in natural language what it expects an available test action to reveal if that hypothesis is true. The expectation can describe changes to the scene or information revealed by the action. We retain the test and expectation with the answer so that later observations can be interpreted against the original prediction.

3.3 Interaction-Grounded Belief Transition and Action Selection

In each round, the agent maintains the belief defined in Section 3.2 throughout task execution, using new observations to reassess its questions and hypotheses. It then selects an action based on those hypotheses, obtaining feedback and observations for the next round.

Belief Transition. At round $t > 0$, the LLM assesses the latest observation o_t, ego state s_t, message m_t including previous belief $\mathcal{B}_{t-1}$, and history actions $a_{0:t-1}$, then updates relevant QA pairs. For each successfully executed test, we retain its associated hypothesis, original expectation, and execution round in the transmissible execution records $\mathcal{R}_t \in m_t$. For a test executed in the preceding round, $a_{t-1} = \tau_{t-1,ij}$, the model compares the observation o_t with the expectation $e_{t-1,ij}$ under H_{ij}. We write $\mathcal{B}_t$ for the updated belief, including unchanged content carried over from $\mathcal{B}_{t-1}$:

$$\mathcal{B}_t = \mathrm{LLM}(h_t). \tag{3}$$

The model uses this assessment to update the probabilities $p_{t,ij}$ of competing answers and their statuses $\sigma_{t,ij} \in \{\mathrm{open}, \mathrm{supported}, \mathrm{refuted}\}$. An answer is open while it remains undiscovered, supported when observations match its expectation, and refuted when the model judges it inconsistent with observed evidence. Refuted answers leave the candidate set $\mathcal{J}_{t,i}$ but remain listed with zero probability mass. It can retain unresolved alternatives, introduce questions or answers about newly relevant unknowns, and revise the test pairs $\mathcal{T}_{t,ij}$. The question status $z_{t,i} \in \{\mathrm{active}, \mathrm{closed}\}$ records the model's decision to close a resolved question or reopen it when its conclusion requires reconsideration. Closed questions remain available as summaries.

Belief Conditioned Action Rollout. The model selects its next action using the updated hypotheses, task objective, action costs, and remaining round budget. It can gather evidence about an unresolved question or proceed with task execution under its current assumptions.

The next action a_t is conditioned on the updated belief $\mathcal{B}_t$:

$$a_t = \mathrm{LLM}(h_t, \mathcal{B}_t). \tag{4}$$

The model also identifies any carried questions informing its choice and states whether the action tests or exploits an answer or advances task completion. These annotations describe the decision; the framework generates $\mathcal{R}_{t+1}$ after successful execution by matching the action to the recorded tests. Executing a_t provides action feedback and the observation o_{t+1}, which inform the next belief update. Appendix B specifies the update protocol, and Appendix B.3 provides the prompts.



Figure 3: **DUET-Bench: eight tasks spanning uncertainty in state, goals, and dynamics.** We construct 240 episodes in BEHAVIOR-1K scenes, with 30 episodes per task. *Top:* Static tasks hide target locations or provide example foods from which the intended food splitting rule or meal must be inferred. *Bottom:* Dynamic tasks add scripted object relocations, with unknown rules determining which items move, where they go, and when. In division of labor, an unseen helper delivers an unknown subset of targets. Question marks identify hidden factors; arrows indicate the affected planning decisions. Paired views show changes from t to $t + k$, with orange boxes marking affected locations.

4 Benchmark

We introduce DUET-Bench (Dynamics and Uncertainty-Aware Embodied Tasks Benchmark), comprising eight tasks (four static and four dynamic) with 30 fixed episodes per task, for 240 episodes in total (Figure 3). Static tasks involve unknown object locations or rules defining a correct outcome; dynamic tasks introduce external activity governed by unknown rules.

4.1 Benchmark Setup

Episodes run in OmniGibson using BEHAVIOR-1K scenes (Li et al., 2023) with an R1 robot. The planner receives a cumulative textual scene graph and selects navigation, observation, or manipulation primitives. Previously observed object states remain in the graph and can become outdated after unseen relocations. Appendix C specifies the interface and cost model.

4.2 Static Tasks: Constructing State and Goal Uncertainty

The *single-* and *multi-object search* tasks require grasping each named target at least once. We place targets outside the starting room, with multi-object targets spanning at least two rooms.

We construct uncertainty about the intended outcome in *food splitting rule* and *meal composition* using manually specified semantic tables. In food splitting, six foods must be divided equally between two destinations, each seeded with one example. A hidden attribute, such as food type or preparation, determines the correct partition. We retain seed pairs consistent with at least three grouping rules and food sets that uniquely identify the intended rule under the equal-size constraint.

Meal composition requires collecting foods for an unspecified meal, such as breakfast or dinner. A food–meal affinity table assigns each food to its highest-weight category. We place one matching food at the destination as a cue and scatter two targets with three distractors elsewhere. Distractors belong to other categories but have affinity to the selected meal. Success requires both targets and no distractor at the destination.

4.3 Dynamic Tasks: Constructing External Changes

Dynamic tasks add scripted object relocations to explicit collection or storage goals. Rules are fixed per episode, and external activity is revealed through later observations of the affected objects.

In *division of labor* and *visitor intent*, the robot must gather all targets at one destination. For division of labor, we assign an unseen helper a subset selected by a hidden food attribute, leaving the remaining targets to the robot. Helper deliveries begin after the robot's first delivery. For visitor intent, we select an undisclosed relocation pattern: delivery to the destination, removal from it, movement between other surfaces, or a mixture.

Trigger rule retains the collection goal but hides when a housemate acts. We select a periodic schedule or a trigger tied to the robot's first delivery of a target, first opening of an object, or entry into a designated room. Each triggered move brings a target to the destination if it already contains a target; otherwise, it moves one to another surface.

Unstable storage requires placing all targets at any of three interchangeable storage spots. We program delayed removals from one undisclosed spot, occurring only while it is out of view. Success requires declaring completion with every target stored and no removal pending.

We filter dynamic episodes in simplified event simulations for identifiable rules and for cost or success differences between policies based on different assumptions about those rules. Appendix C.2 provides the full task specifications and construction criteria.

5 Experiments

5.1 Experimental Setup

We evaluate on all 240 frozen episodes of DUET-Bench, with 30 episodes per task. The results separate two object-search tasks (60 episodes), two static rule tasks (60), and four dynamic tasks (120), distinguishing uncertainty about object locations from uncertainty about goals and external changes. All four methods use Gemini 3.8 Flash with *low* thinking and temperature 1.0, a 200-round horizon, and the shared primitive executor and cost specification in Appendix C.1.

We equip all methods with a self-written auxiliary memory note m_t for cross-round information passing, as a parallel to BeliefAct's persistent belief and execution records, as described in Section 3.3. At round t, the output note m_{t-1} from the last round is fed as input. Baselines are prompted to write history action outcomes, task progress, and future plans to m_t.

We also evaluate on the 50 tasks in C-WAH benchmark with two communicating agents and partial symbolic observations (Zhang et al., 2024b; Puig et al., 2021). Both agents run the evaluated method, using the same backbone and decoding settings as above. We run one episode per method–task pair with a 250-step limit. The C-WAH adaptation selects high-level plans, with environment-specific action granularity and belief storage (Appendix A.4).

5.2 Baselines and Evaluation Metrics

Naive directly selects actions from context; Bayesian Scene Graph (**BSG**) maintains location distributions inspired by LLM-MCTS (Zhao et al., 2023); **PCE** (Seo et al., 2026) evaluates proposed actions under alternative scenarios. On C-WAH, we compare Naive, BSG, and PCE.

On DUET-Bench, we report **success rate (SR)**, mean **Rounds** (failures counted as 200), and movement-equivalent **Cost**. C-WAH reports SR, completion **Steps**, per-agent travel (**Dist.**) and communication (**Comm**), and **LLM calls** per episode. Please refer to Appendix A for details.

5.3 Quantitative Analysis

DUET-Bench Performance. Overall, BeliefAct combines the highest overall success rate with the fewest rounds and lowest interaction cost (Table 1). The efficiency gains concentrate in tasks where hidden rules or external changes affect action choice, with round savings over Naive increasing from 8% overall to 11% on these tasks.

Table 1: Results on DUET-Bench episodes with Gemini 3.8 Flash. BSG: Bayesian scene graph; Rnds: rounds. **Bold**/underlined: best/second-best across Tables 1–3.

	Object search (n=60)			Static rule tasks (n=60)			Dynamic tasks (n=120)			Overall (n=240)		
Method	SR (%)	Rnds	Cost (m)	SR (%)	Rnds	Cost (m)	SR (%)	Rnds	Cost (m)	SR (%)	Rnds	Cost (m)
Naive	83.3	77.1	**137.2**	76.7	86.5	275.5	89.2	51.1	141.6	84.6	66.5	174.0
BSG	76.7	81.3	144.4	**80.0**	91.6	287.1	**92.5**	48.6	138.7	85.4	67.5	177.2
PCE	83.3	**76.1**	149.5	71.7	98.7	283.4	87.5	56.0	151.1	82.5	71.7	183.8
Ours	**85.0**	78.4	153.5	78.3	**79.1**	**274.6**	91.7	**44.1**	**125.3**	**86.7**	**61.4**	**169.7**

Table 2: Results on C-WAH benchmark.

Method	Steps	Dist. (m)	Comm	LLM calls
Naive	39.8	33.7	0.74	48.0
BSG	43.3	37.3	**0.51**	50.1
PCE	41.7	34.4	1.72	60.5
Ours	**39.2**	**33.2**	0.58	**24.0**

Table 3: Belief representation and transition ablations on DUET-Bench.

Variant	SR (%) ↑	Rounds ↓	Cost (m) ↓
fixed questions	86.3	65.3	182.6
single answer	84.6	63.0	177.0
no transition	84.2	65.5	184.5
direct transition	85.8	64.7	175.1
Ours (full)	**86.7**	**61.4**	**169.7**

In object search tasks, BeliefAct achieves the highest object-search success rate, with comparable rounds but somewhat higher interaction cost. We attribute this efficiency gap to its tendency to verify hypotheses through exploration, which can lead to more searching than task completion requires.

On static tasks regarding unclear rules, and dynamic tasks with active environment changes, BeliefAct leads both efficiency metrics on rule and dynamic tasks, with BSG solving one more episode in each block. Its advantage persists on jointly solved episodes: in trigger rule, all methods succeed, yet BeliefAct needs 23% fewer rounds than the best baseline. The case studies show how belief updates change the action sequence, correcting the inferred food splitting rule or deliberately triggering an external intervention (Section 5.4).

C-WAH Benchmark Performance. On C-WAH, BeliefAct maintains full task completion and comparable steps and travel with 50% fewer LLM calls than Naive (Table 2). Naive, PCE, and BeliefAct replan at high-level plan boundaries; BeliefAct combines belief and action in one call, whereas Naive uses two (Appendix A.1).

5.4 Qualitative Analysis

We analyze BeliefAct's belief updates and action choices in three DUET-Bench episodes, one each from food splitting rule, visitor intent, and trigger rule. Each case compares all four methods on the same frozen episode, indexed by its BEHAVIOR-1K scene name and episode ID (e.g., `Merom`, `s16`). Hidden rules and event logs are used only for retrospective verification; not as inputs to agents.

A new question after an unmet completion prediction. In food splitting rule (`Merom`, `s16`), one seed food on each countertop provides evidence about how six remaining foods should be divided. BeliefAct adopts a baked-versus-other grouping, assigning it probability 0.95 in its response, and eventually places all eight foods. Its final placement predicts task completion, but the episode continues. The model then opens a new question about the grouping, proposes a cooked-dishes-versus-cold-snacks interpretation, and identifies the sugar cookie and fried chicken as the two items to swap. This partition matches the hidden labels on all eight foods. After the swap, the episode completes in 24 rounds with 90.6 m of interaction cost. On the same episode, all three baselines exhaust 200 rounds and incur 597–649 m.

Revising an answer from incidental evidence. In visitor intent (`Rs_garden`, `s04`), the model initially believes that the visitor is not interfering and predicts that placed foods will remain on the

breakfast table. While searching the garden for a missing chocolate bar, it instead observes a waffle that it had already placed on the breakfast table. The model rewrites its visitor hypothesis to account for this relocation; subsequent observations reveal the cookie and chocolate bar at the same garden coffee table. It then explicitly uses the revised account—the visitor moves foods from the breakfast table to that coffee table—to decide where to retrieve the remaining items. Evidence collected for an object-location question thus updates an answer about the visitor's behavior. BeliefAct finishes in 28 rounds despite four undone placements; Naive, BSG, and PCE require 54, 47, and 172 rounds, respectively, with 8, 7, and 25 undone placements.

Choosing actions before resolving the trigger. In trigger rule (`Ihlen`, `s45`), a housemate delivers items when the destination table is occupied but hides them when it is empty. BeliefAct initially maintains four possible triggers: periodic activity, placement, opening a container, and entering a room. This uncertainty already affects action order: it starts a pile on the destination table before opening cupboards. After the first placement, it waits to test whether placement triggers a delivery; no item moves, and the model marks that answer as refuted. It next opens a cupboard with the explicit expectation of a delivery, then opens a refrigerator. On returning to the table, it observes two delivered items. It subsequently opens the last cupboard deliberately to trigger another delivery. The episode completes in 31 rounds, compared with 67 for Naive, 92 for BSG, and 49 for PCE. All four methods receive three deliveries, so the lower round count does not come from receiving more assistance.

5.5 Ablations of Belief Representation and Transition

The four ablations use the same 240 episodes, model settings, horizon, and metrics as the main comparison. Table 3 separates the QA representation (Section 3.2) from belief transition (Section 3.3). Every ablated variant lowers aggregate success and increases both mean rounds and interaction cost.

Question formulation and competing answers. *Fixed questions* replaces model-proposed questions with task-specific templates, increasing mean rounds from 61.4 to 65.3 and cost from 169.7 to 182.6 m. The effect is strongest across the dynamic tasks: rounds rise from 44.1 to 52.7, reaching 56.3 versus 35.0 on trigger rule. Choosing what to ask is therefore part of efficient planning, especially when an unknown rule determines how actions change the world. *Single answer* is asked to keep one answer per question and to remove probabilities. Its success rate falls from 86.7% to 84.6%, while rounds and cost rise to 63.0 and 177.0 m. Keeping competing answers available for action selection improves both task completion and efficiency.

Persistent belief and interaction-grounded updates. *No transition* rebuilds the structured belief each round while retaining auxiliary memory and action history. Mean rounds increase from 61.4 to 65.5 and cost from 169.7 to 184.5 m, showing the value of carrying hypotheses beyond free-form memory. *Direct transition* updates the previous QA belief but replaces it in full, without answer-level tests, expectations, statuses, or execution records (Appendix A.3). It still requires 64.7 rounds and 175.1 m; across the dynamic tasks, its 48.3 rounds exceed the full method's 44.1. Carrying belief forward is therefore only part of the gain: the full transition mechanism further improves decisions by retaining structured evidence for subsequent updates.

6 Conclusion

In this paper, we presented BeliefAct, an LLM-based planning framework that represents task-relevant uncertainty as free-form questions with probability-weighted answers, pairs each hypothesis with a test action and the outcome expected if it holds, and revises the belief by comparing prediction against observation. We introduced DUET-Bench, a benchmark of eight household task families with heterogeneous uncertainties spanning the initial configuration, the goal interpretation, or an external process. BeliefAct achieved the highest overall success rate at the fewest rounds and the lowest interaction cost on DUET-Bench, and the same approach transferred to two-agent Communicative Watch-and-Help at equal task completion with substantially fewer LLM calls. Future work will explore self-improvement across episodes, where the mismatches between prediction and observation that drive revision within an episode also refine which questions the agent poses, and which test actions it trusts to settle them, in later tasks.

AI USE STATEMENT

In this work, we used generative AI tools to help develop theoretical models and conceptual frameworks, formulate mathematical claims, propose and refine hypotheses, design and provide feedback on research methodology and experiments, implement methods, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to assist with translation. The generation of synthetic datasets, provision of critical ingredients for proving mathematical claims, assistance in writing proofs, and cleaning or reformatting datasets are not applicable to this work.

Additionally, we used generative AI tools to create or modify scientific figures or images, draft parts of the manuscript, summarize and analyze existing literature, discover research topics and identify gaps, brainstorm research ideas, source and search for information, improve the readability of the manuscript, identify relevant literature, and format references.

We have reviewed all AI-assisted work. Research ideas, hypotheses, conceptual frameworks, and mathematical formulations suggested or refined with generative AI were critically assessed by the authors for scientific validity and consistency with the research problem. AI-assisted implementations were inspected, executed, and tested against the intended methodology and experimental setup. Literature-related outputs, including summaries, references, and claims about prior work, were manually checked against the original sources. AI-assisted interpretation of experimental results was verified against the underlying experimental data, and all scientific figures and manuscript text were reviewed and revised by the authors before inclusion. Generative AI tools were used as research and writing assistants and did not replace author judgment in determining the methodology, conducting experiments, or drawing the final scientific conclusions.

We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Implementation details, benchmark specifications, and evaluation settings are provided in Appendices A–C.

REFERENCES

Guy Azran, Michael Navat, and Sarah Keren. Bridging learned visual perception and symbolic belief-space planning via probabilistic grounding. In *International Conference on Neuro-Symbolic Systems (NeuS)*, 2026.

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John Turner, Eric Undersander, and Tsung-Yen Yang. PARTNR: A benchmark for planning and reasoning in embodied multi-agent tasks. In *The Thirteenth International Conference on Learning Representations*, 2025.

Pei-An Chen, Yong-Ching Liang, Jia-Fong Yeh, Hung-Ting Su, Yi-Ting Chen, Min Sun, and Winston Hsu. ADAPT: Benchmarking commonsense planning under unspecified affordance constraints. *arXiv preprint arXiv:2604.14902*, 2026.

Siwei Chen, Anxing Xiao, and David Hsu. LLM-State: Open world state representation for long-horizon task planning with large language model. *arXiv preprint arXiv:2311.17406*, 2023. URL https://arxiv.org/abs/2311.17406.

Logan Cross, Violet Xiang, Agam Bhatia, Daniel L. K. Yamins, and Nick Haber. Hypothetical minds: Scaffolding theory of mind for multi-agent tasks with large language models. *arXiv preprint arXiv:2407.07086*, 2024.

Pascale Fung, Yoram Bachrach, Asli Celikyilmaz, Kamalika Chaudhuri, Delong Chen, Willy Chung, Emmanuel Dupoux, Hongyu Gong, Hervé Jégou, Alessandro Lazaric, Arjun Majumdar, Andrea Madotto, Franziska Meier, Florian Metze, Louis-Philippe Morency, Théo Moutakanni, Juan Pino, Basile Terver, Joseph Tighe, Paden Tomasello, and Jitendra Malik. Embodied AI agents: Modeling the world. *arXiv preprint arXiv:2506.22355*, 2025. doi: 10.48550/arXiv.2506.22355. URL https://arxiv.org/abs/2506.22355.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Tomas Jackson, Noah Brown, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1769–1782. PMLR, 2023.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T. Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as I can, not as I say: Grounding language in robotic affordances. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 2023.

Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. Housekeep: Tidying virtual households using commonsense reasoning. In *European Conference on Computer Vision (ECCV)*, 2022.

Minsoo Kim, Jongyoon Kim, Jihyuk Kim, and Seung-won Hwang. QuBE: Question-based belief enhancement for agentic LLM reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21403–21423. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.1193.

Shubham Kumar, Harshit Kumar, Narendra Ahuja, and Saurabh Jha. Towards a belief-based world model for LLM agents. *arXiv preprint arXiv:2609.00455*, 2026.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. BEHAVIOR-1K: A benchmark for embodied AI with 1,000 everyday activities and realistic simulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 80–93. PMLR, 2023. URL https://proceedings.mlr.press/v205/li23a.html.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

Junfeng Liao, Qizhou Wang, Jianing Zhu, Bo Du, Rui Yan, and Xiuying Chen. Belief memory: Agent memory under partial observability. *arXiv preprint arXiv:2605.05583*, 2026.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. TEACh: Task-driven embodied agents that chat. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 2017–2025, 2022.

Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B. Tenenbaum, Sanja Fidler, and Antonio Torralba. Watch-and-help: A challenge for social perception and human-AI collaboration. In *International Conference on Learning Representations*, 2021.

Xavier Puig, Tianmin Shu, Joshua B. Tenenbaum, and Antonio Torralba. NOPA: Neurally-guided online probabilistic assistance for building socially intelligent home assistants. *arXiv preprint arXiv:2301.05223*, 2023.

Kartik Ramachandruni and Sonia Chernova. Personalized robotic object rearrangement from scene context. In *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2025.

Antoni Rosinol, Arjun Gupta, Marcus Abate, Jingnan Shi, and Luca Carlone. 3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Proceedings of Robotics: Science and Systems*, 2020. doi: 10.15607/RSS.2020.XVI.079. URL https://www.roboticsproceedings.org/rss16/p079.html.

Nikolai Rozanov and Marek Rei. StateAct: Enhancing LLM base agents via self-prompting and state-tracking. *arXiv preprint arXiv:2410.02810*, 2024.

SeungWon Seo, SeongRae Noh, Junhyeok Lee, SooBin Lim, Won Hee Lee, and HyeongYeop Kang. REVECA: Adaptive planning and trajectory-based validation in cooperative language agents using information relevance and relative proximity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

SeungWon Seo, SooBin Lim, Seongrae Noh, Haneul Kim, and HyeongYeop Kang. From assumptions to actions: Turning LLM reasoning into uncertainty-aware planning for embodied agents. In *The Fourteenth International Conference on Learning Representations*, 2026.

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Joykirat Singh, Zaid Khan, Archiki Prasad, Justin Chih-Yao Chen, Akshay Nambi, Hyunji Lee, Elias Stengel-Eskin, and Mohit Bansal. Agent-BRACE: Decoupling beliefs from actions in long-horizon tasks via verbalized state uncertainty. *arXiv preprint arXiv:2605.11436*, 2026.

Wenjing Tang, Xinyu He, Yongxi Huang, Yunxiao Xiao, Cewu Lu, and Panpan Cai. Tru-POMDP: Task planning under uncertainty via tree of hypotheses and open-ended POMDPs. In *Advances in Neural Information Processing Systems*, volume 38, pp. 55367–55396. Curran Associates, Inc., 2025. doi: 10.52202/085713-1851.

Meiqi Wang and Shichao Li. Compiling and benchmarking task-state horizons for embodied agents. *arXiv preprint arXiv:2608.08036*, 2026.

Ruiyang Wang, Hao-Lun Hsu, Swarajh Mehta, Jiwoo Kim, Zhihao Dou, and Miroslav Pajic. GAVEL: Graph world models for verified and efficient long-horizon LLM task planning. *arXiv preprint arXiv:2609.19315*, 2026.

Zhimin Wang, Duo Wu, Shaokang He, Jinghe Wang, Linjia Kang, Jing Yu, Kai Zhu, Jiawei Li, and Zhi Wang. Collaborative belief reasoning with LLMs for efficient multi-agent collaboration. *arXiv preprint arXiv:2509.21981*, 2025.

Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. TidyBot: Personalized robot assistance with large language models. *Autonomous Robots*, 2023.

Zhijie Yan, Shufei Li, Zuoxu Wang, Lixiu Wu, Han Wang, Jun Zhu, Lijiang Chen, and Jihong Liu. Dynamic open-vocabulary 3D scene graphs for long-term language-guided mobile manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Xu Yin, Min-Sung Yoon, Yuchi Huo, Kang Zhang, and Sung-Eui Yoon. LangPert: Detecting and handling task-level perturbations for robust object rearrangement. *arXiv preprint arXiv:2504.09893*, 2025.

Takuma Yoneda, Jiading Fang, Peng Li, Huanyu Zhang, Tianchong Jiang, Shengjie Lin, Ben Picker, David Yunis, Hongyuan Mei, and Matthew R. Walter. Statler: State-maintaining language models for embodied reasoning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15083–15091, 2024. doi: 10.1109/ICRA57147.2024.10610634.

Hao Yuan, Yuxin Wang, Lei Ji, and Zhiwei Yu. From failures to supervision: DynamicEnvPlan for robust long-horizon embodied planning. *arXiv preprint arXiv:2608.00613*, 2026.

Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, Xiaojun Chang, Junge Zhang, Feng Yin, Yitao Liang, and Yaodong Yang. ProAgent: Building proactive cooperative agents with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024a.

Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. In *The Twelfth International Conference on Learning Representations*, 2024b.

Hongxin Zhang, Zeyuan Wang, Qiushi Lyu, Zheyuan Zhang, Sunli Chen, Tianmin Shu, Behzad Dariush, Kwonjoon Lee, Yilun Du, and Chuang Gan. COMBO: Compositional world models for embodied multi-agent cooperation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Linfeng Zhao, Willie McClinton, Aidan Curtis, Nishanth Kumar, Tom Silver, Leslie Pack Kaelbling, and Lawson L. S. Wong. Seeing is believing: Belief-space planning with foundation models as uncertainty estimators. *arXiv preprint arXiv:2504.03245*, 2025.

Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 31967–31987. Curran Associates, Inc., 2023. doi: 10.52202/075280-1387.

Table 4: DUET method interfaces. All use the same observations and executor.

Method	Memory cap	Plan in memory	Candidate and room summaries	Parse attempts
Naive	160 words	Yes	No candidate list or room-summary block	5
BSG	160 words	Yes	No candidate list or room-summary block	5
PCE	120 words	No	Own candidate list; no room-summary block	5
BeliefAct	160 words	Yes	Advisory candidate IDs and observed-room summaries	3

A Experimental Details

A.1 Configurations and Baseline Implementations

All DUET runs use Gemini 3.8 Flash, temperature 1.0, low thinking, JSON output, a maximum output budget of 65,536 tokens, and a 40-round history window. The horizon is 200 dispatched actions. The API sampling seed and the Python/NumPy/PyTorch runtime seed are set to zero. Each method has one reported run per frozen episode. Runs use Python 3.12.3, Isaac Sim 6.0.1, OmniGibson, and `google-genai` 2.10.0, on RTX A5000 or RTX A6000.

Table 4 records differences retained in the main comparison. Naive, BSG, and BeliefAct use the same auxiliary memory instructions: entity-level action/evidence records, a task summary, and an ordered plan. PCE retains its 120-word entity/task format. Memory is rewritten by the model, truncated to its configured cap, and carried to the next decision; an empty or invalid note retains the previous one. The main comparison therefore evaluates the implemented systems with these interface differences. All four ablations retain BeliefAct's context, memory, and parser settings.

Naive. Naive directly selects an action from the observation, recent history, and auxiliary memory in one call.

Bayesian Scene Graph (BSG). BSG maintains a distribution over candidate locations of the target object, similar to the object-location belief in LLM-MCTS (Zhao et al., 2023). Each round, the model marks observed locations, assigns zero probability to inspected locations found empty, adds newly revealed locations, and renormalizes the distribution. In the same call, it directly selects an action conditioned on this distribution. The program stores the returned list and supplies it in the next round, without Bayesian updates or numerical normalization. There is no separate table keyed by each target in multi-object search episodes. BSG uses the same scene-graph observations, primitive executor, and auxiliary memory format as BeliefAct, while representing only object-location uncertainty.

PCE. PCE (Seo et al., 2026) proposes an action, expands uncertain assumptions into alternative scenarios, and scores the resulting choices by likelihood, goal progress, and cost. A grasp targeting an object already in the scene graph bypasses further evaluation; otherwise a verifier constructs a binary scenario tree of depth at most three and an evaluator scores its action leaves. Likelihood, gain, and cost ratings from one to five are mapped to $[0, 1]$, and the selected leaf maximizes likelihood times gain minus cost. Cost ratings use distance bins $[0, 1]$, $(1, 3]$, $(3, 6]$, $(6, 10]$, and $(10, \infty)$ metres, including manipulation surcharges. Unusable trees or scores fall back to the proposed action. Memory is returned with the proposal, so a decision requires one or three calls before retries.

C-WAH decision procedures. The planner chooses a high-level exploration, container inspection, grasp, placement, or communication plan. The executor expands physical plans into environment actions until completion or abandonment; it does not query the LLM at every environment step. Naive, PCE, and BeliefAct share this replanning trigger. BeliefAct generates belief and action in one call, whereas Naive makes a reasoning call followed by an action-extraction call. PCE uses those two calls and, when its verification gate opens, one verifier and one evaluator call. Thus the call-count comparison includes differences in the methods' implemented decision procedures. We

use the released PCE implementation; the environment and belief-state adaptation are specified in Appendix A.4.

A.2 Evaluation Metrics

Success rate (SR) is the fraction of episodes satisfying the task predicate within the horizon. On DUET-Bench, Rounds averages the number of dispatched actions for successes and 200 for failures. Cost averages the actual movement-equivalent interaction cost in both cases, without a failure penalty or cap. Navigation uses measured travel distance; panoramic scans and doorway crossings cost 2 m; grasping and placing cost approach distance plus 3 m; waiting costs 1 m. Appendix C.1 gives the complete primitive cost specification. Paired efficiency is the median baseline/BeliefAct ratio over episodes solved by both methods. In the DUET tables, bold and underlining indicate the best and second-best values.

On C-WAH, Steps is the mean length of the longer of the two agents' action sequences. Dist. and Comm are mean travel distance and message count per agent. LLM calls is the mean number of successful provider requests per episode for planning, message generation, and replies, excluding failed API attempts. Calls logged per job shard are allocated uniformly to its episodes, and records from discarded launches are excluded.

A.3 Ablation Implementations

Fixed questions. A manually specified question list is defined for each of the eight tasks. The search tasks ask where each target is; the other tasks ask about the food splitting rule, intended meal, helper's assigned food subset, unstable spot, visitor behavior, or trigger, together with remaining target locations. These templates use public task information and do not contain the episode's hidden answer. The framework fixes question identifiers and text, rejects unmatched questions, and ignores closure requests. Answer probabilities, tests, statuses, and execution records retain the full method's update rules.

Single answer. Each question retains the first non-refuted answer in the response; an alternative replaces the previous answer. Probability fields are removed from the output contract and display. Tests, expectations, answer statuses, refuted-answer history, question closure/reopening, and execution records remain available. Unmentioned questions are still retained.

No transition. The structured store is reset before every decision, including its active questions, refuted answers, closed questions, and execution records. The model generates a fresh belief for the current action. Auxiliary memory and the 40-round action history remain available, so earlier hypotheses can still be conveyed through the model's memory note.

Direct transition. The model reads its previous question–answer text and probabilities, updates them from the current context, and returns the complete belief to retain. Each response replaces that state: omitted entries disappear, probabilities are not normalized by the framework, and answer-level tests, expectations, statuses, and execution records are absent. The top-level action expectation and auxiliary memory remain; `purpose` refers to a question rather than a question–answer pair. This variant removes the structured transition machinery, not the model's ability to update its belief.

A.4 C-WAH Adaptation

C-WAH uses the VirtualHome executable version 2.3.0 and the original PCE test set. Its 50 tasks span five task types and two apartments, with two agents sharing a goal. Each agent receives its partial symbolic observation, the latest ten completed plans, and three dialogue turns. The evaluated Gemini settings are temperature 1.0, low thinking, and a 65,536-token output budget; the episode limit is 250 environment steps.

The shared execution interface and per-method decision procedures are described in Appendix A.1.

Each BeliefAct agent maintains its own textual QA state. A response containing questions replaces the live question/answer set, while omitted content disappears from that set; a response containing

no questions leaves it unchanged. Refuted-answer and closed-question records persist separately. Probabilities are displayed as supplied without program normalization. Model-written statuses and explicit closure govern the state, and completed plans are matched to answer-level test text to produce execution records. There are no `basis` or `purpose` fields in this adaptation. These storage rules differ from the incremental merge used on DUET.

Messages consume environment steps and reach the partner in its next observation. Selecting a message in BeliefAct or Naive triggers a separate call to generate its body; PCE can generate a message through its verified action branches. Replying to an incoming message also uses a call.

A.5 Episode Accounting

Parsing attempts do not dispatch actions or consume rounds. If all attempts fail, the framework retries the decision; three consecutive such failures terminate the episode. For PCE, four food splitting rule episodes ended this way and count as failures.

Interrupted attempts without a final result are rerun from the start; completed episodes, including task failures, are retained. Every reported method has a result for every frozen episode. For C-WAH, call accounting additionally excludes API log entries preceding the valid launch.

B Method Implementation

This section specifies the implementation of BeliefAct used on DUET-Bench. The agent retains its QA belief $\mathcal{B}$, execution records $\mathcal{R}$, and auxiliary memory within an episode, as components of m_t. All three are reset at the start of a new episode. Appendix A.4 describes the C-WAH adaptation.

B.1 One Response per Decision

The input contains the task instruction, robot state, cumulative scene graph, the latest 40 rounds of action history, and retained memory. We append the carried belief, a list of action candidates, and summaries of visited rooms and unexplored doors derived from the agent's observations. Candidate identifiers are reassigned each round. Their rows specify actions and arguments; distances and action charges appear in the scene graph and task interface. The list is advisory: a valid action outside it can also be executed.

One LLM response supplies the new or updated entries in `belief`, followed by the action and its arguments; no second call receives a programmatically merged belief. Each entry contains a question identifier and text (`id`, `q`), an `answers` list, cited observation rounds (`saw`), and an optional `closed` flag. Each answer specifies `id`, `text`, an absolute probability `p`, a candidate action `test`, its expected observation `expect`, and `status`. The response also contains `reasoning`, `memory`, and three action annotations: `basis`, `purpose`, and a top-level `expect`.

`basis` names questions in the incoming belief that informed the action. `purpose` names a question–answer pair when the action tests or exploits it, `execute` for task progress, or `none`. These are model reports made before execution. The top-level `expect` predicts the action's outcome; the answer-level expectations are the ones retained in execution records. The framework validates the action, merges the belief output, and dispatches the model's choice without reranking it.

B.2 Persistent Belief and Execution Records

Questions are matched by identifier, and answers by identifier within a question. Unmentioned questions remain unchanged. Within an updated question, omitted answers retain their old probabilities, and omitted test, expectation, or status annotations are carried forward where applicable. A model-marked `refuted` answer leaves the candidate set and is recorded separately; `supported` does not force probability one. The model closes a question with `closed:true` and can reopen its stored snapshot by resubmitting its identifier. Refuting every answer does not automatically close a question.

After merging, missing or invalid probabilities share the nonnegative mass remaining after the supplied and retained values. If their total is at most 10^{-9}, all candidate answers receive equal

probability; otherwise, the values are divided by their total only when it differs from one by more than 0.005. This is numerical normalization of model estimates. The framework does not infer support or refutation from observations, close questions at a confidence threshold, or evict questions. More than ten active questions produces a reminder. The prompt displays the latest six refuted answers per question and eight closed-question summaries; these display limits do not erase the stored history.

A test identifier is resolved to its action and arguments, so it remains identifiable when the candidate list changes. After a successful dispatch, the framework records the execution round and the expectation attached at execution time for every answer whose test matches that action. Thus one action can produce several records, independently of the single pair named in `purpose`. Failed actions remain in action history but produce no test record. From the next decision, each still-open answer displays its executed test and original expectation. Editing the expectation alone does not alter that record; selecting a different test clears it. The LLM compares the available observations with these expectations and decides which probabilities, statuses, or questions to update.

B.3 Prompt Instructions

The prompt leaves question wording to the model, while requiring questions to concern unknown world states. Facts already stated by the task or scene graph belong in memory. Unless the task fixes all possibilities, an answer must cover unobserved alternatives. The following excerpts specify when to open a question, how evidence affects its status, and what to submit:

> "If the task sentence describes something hidden that decides success, open an entry about exactly that in your first round, one answer per possibility."
>
> "You maintain each answer's `status` yourself: 'open' until evidence arrives, 'supported' when what you saw matches its `expect`, 'refuted' when what you saw rules it out. Cite in `saw` the rounds whose observations you relied on."
>
> "Send an entry only when you create it or change it; entries you leave out are kept unchanged."

Task-specific additions expose `wait` or `declare_done` only where available.

C Benchmark Specification

C.1 Observations, Actions, and Feedback

DUET-Bench uses a kinematic R1 robot in OmniGibson. Navigation and manipulation are implemented as high-level primitives, so the experiments evaluate planning with simulator-derived perception and scripted execution. The LLM receives text rather than images. Visibility-filtered scene-graph nodes expose observed objects' categories, positions, distances, open states, and observation rounds; edges describe support, containment, room membership, and doors connecting visited rooms. Opening a receptacle additionally reveals its contents. Previously observed nodes persist outside the field of view. An object's support or containment relation is revised when that object is observed again; seeing only its former receptacle does not establish its absence.

Each dispatched action includes its built-in observation, followed by cost accounting and task-side events and completion checks. External relocations therefore occur after that action's observation and enter the planner's context only after a later observation. Visitor intent checks completion before allowing another visitor move; unstable storage evaluates success upon declaration. The hidden rules, event logs, and evaluator diagnostics are not sent to the model. Multi-object search exposes collected-target progress; rule and dynamic tasks provide no intermediate correctness counts. Continuing after an unsuccessful completion attempt can itself be evidence, as specified in the task instructions.

Here d is the measured approach distance. Flat charges replace displacement charges; manipulation surcharges also apply to failed attempts. Every dispatched action consumes one round, including a zero-cost declaration. All methods share the primitive executor, cost rules, and task predicates. For sleeping movable objects, both perception and the success judge use a downward support ray with a 5 cm tolerance; other objects use the simulator's relation evaluator.

Table 5: Primitive actions and movement-equivalent costs.

Action	Effect	Cost (m)
`navigate`	Approach a named object	d
`navigate_through`	Cross a named open doorway	2
`grasp`, `place`	Pick up an object; place it OnTop or Inside a reference object	$d + 3$
`open`, `close`	Change a named object's open state	d
`scan_360`	Observe in four directions	2
`wait`	Observe in place; division of labor, visitor intent, and trigger rule tasks only	1
`declare_done`	End and evaluate unstable storage	0

Table 6: Task specifications. Each task contains 30 episodes.

Task (scenes)	Configuration and hidden quantities	Success condition
Single-object search (6)	One target outside the starting room; only its category is given.	Grasp the target.
Multi-object search (6)	Two to four targets outside the starting room and across at least two rooms.	Grasp every target at least once; putting one down preserves its collected status.
Food splitting rule (14)	Two destinations, one seed food on each, and six targets; an unknown food attribute determines a 3:3 partition.	All six targets occupy their assigned destinations.
Meal composition (20)	One cue food, two targets, and three distractors; an unknown meal category determines membership.	Both targets and no distractor occupy the destination.
Division of labor (5)	Six foods; an unknown attribute selects three or four for an unseen helper to deliver. Deliveries follow the robot's first delivery, reactive delays of one or two actions, and a period of four to eight actions.	All six foods are simultaneously at the destination.
Unstable storage (5)	Six objects and three storage spots. One spot has a fixed removal schedule with delays of one to six actions; removals wait until the spot is out of view.	Declare completion with all six stored and no removal pending.
Visitor intent (5)	Six foods; a fixed program delivers, removes, moves laterally, or mixes these behaviors. Activity is periodic and reacts to robot placements.	All six foods are simultaneously at the destination, checked before the next visitor move.
Trigger rule (6)	Seven foods, three initially in named closed containers. Activity follows a period of three to five actions, a first placement, a first opening, or entry into a designated room.	All seven foods are simultaneously at the destination.

C.2 Task Construction and Success Conditions

Table 6 specifies the eight tasks. Each has 30 frozen episodes; the scene counts refer to those episodes. Object models, placements, hidden labels, and event parameters are fixed for each episode. Task instructions disclose the goal and the relevant interaction rules, while withholding the realized hidden values.

Static uncertainty. Search episodes are drawn from verified object placements, excluding targets in the starting room; multi-object targets also have at least 1 m pairwise separation. Food splitting rule and meal composition use manually specified semantic tables. The food-attribute table assigns 88 food categories values for kind, preparation, origin, taste, and size class. A seed pair must permit at least three attribute-based explanations; the six targets must distinguish the chosen explanation,

with at least three target discoveries required under the least informative discovery order. The rule is fixed by the stored labels, not by an online language-model judge.

The meal table assigns 54 food categories affinity weights over breakfast, brunch, lunch, afternoon tea, dinner, barbecue, and picnic. Each food belongs to its unique highest-weight category. For a sampled meal, three matching foods are drawn by affinity; the strongest becomes the cue and the other two become targets. Three distractors belong to other categories but have normalized affinity at least 0.10 for the selected meal. The generators check category coverage and confusable alternatives, and place foods without encoding their group in their location.

External dynamics. Dynamic episode specifications store complete event schedules and object-selection orders. Apparent randomness, including mixed visitor behavior and intermittent storage removals, is compiled into these schedules rather than sampled anew during execution. In division of labor, the helper moves only its designated subset and never removes delivered items. Visitor programs can deliver to the destination, remove from it, move between other surfaces, or mix these actions; periodic intervals are four to eight actions, placement reactions take one or two actions, and moves have a four-action cooldown, extended to six after removal. In trigger rule, each triggered move delivers an accessible object when the destination contains a target and moves one to another surface when it is empty. External actors do not move the robot's held object.

Simplified event simulations filter the dynamic candidates before simulator evaluation. Episodes of division of labor retain at least four initial rule hypotheses and require the oracle to save at least 5 m against specified uninformed policies; each incorrect rule also incurs at least 5 m of regret. Trigger-rule episodes require the policy matched to the true trigger to outperform the alternatives by at least 5 m and produce at least two interventions after the first delivery. Visitor-intent episodes require informative interventions, different preferred policies for helpful and obstructive behavior, and feasible completion despite removals. Unstable-storage episodes require the oracle to be strictly more efficient than the tested safe policies and reject immediate placement-and-declaration. The visitor and storage filters impose no common numerical cost margin. These are construction-time checks, distinct from the 200-round evaluation.

C.3 Frozen Episode Selection

For each static task, a pool of 50 episodes is ordered by a seeded hash within scenes and selected by round-robin traversal across scenes; the first 30 are retained (ordering seed 20260919). For each dynamic task, the first 30 episode identifiers in sorted order are taken from pools that passed the construction and simulator checks. The resulting hidden-value frequencies need not be exactly balanced. All methods are evaluated on the same frozen episodes.

D Worked Examples

The following excerpts come from the reported DUET runs. Round indices are zero-based. We distinguish the agent's visible evidence and output from the hidden episode specification, which is used only for retrospective checking.

D.1 Testing a Hypothesis and Choosing the Next Object

In division of labor, the robot must collect six foods while an unseen helper delivers an unknown subset. By round 4, the robot has delivered bacon. Its question `helper_sort` asks which foods the helper collects. The model assigns probabilities 0.4, 0.4, and 0.2 to egg items, meat/seafood, and another categorization. All three answers propose `wait`, with different expectations: an egg item appears at the destination, a meat/seafood item appears, or a food from another category appears. The selected action is `wait`, with `basis=[helper_sort]` and `purpose=helper_sort.egg_items`.

That action's observation shows the omelet at the destination, where the robot did not place it. The next input includes the updated scene-graph relation and three execution records, one for each matching test. The egg-item record repeats the expectation written before execution. At round 5, the model outputs the fields below.

```
{"belief": [{"id": "helper_sort", "answers": [
  {"id": "egg_items", "p": 1.0, "status": "supported"},
  {"id": "meat_seafood", "p": 0.0, "status": "refuted"},
  {"id": "unknown_sort", "p": 0.0, "status": "refuted"}
], "saw": [4], "closed": true}],
 "basis": ["helper_sort"],
 "purpose": "execute",
 "action": "grasp",
 "args": {"target_name": "sv_obj_3_ctdwth"}}
```


The probability and status changes, including explicit question closure, are present in the model's original response. The framework reports no normalization or automatic closure for this update. The selected object is the meatball; the robot places it in round 6 and then collects the mussel, completing the episode in nine rounds. The hidden program delivers the omelet, egg, and hard-boiled egg, matching the model's proposed subset. The example shows how the model uses an observed delivery to divide the remaining work; it does not establish that its inferred category generalizes beyond these items.

D.2 Updating a Different Question from Incidental Evidence

In visitor intent, the robot has placed a waffle on the breakfast table and is searching for a chocolate bar. At round 14 it enters the garden; its round-15 scan serves the chocolate-bar location question. The scan instead reveals the waffle on a garden coffee table. This observation becomes available in the round-16 input, together with the robot's earlier placement history.

The model then rewrites answer `n2` of `npc_behavior`, changing its account from the visitor not interfering to the visitor having moved the waffle. It grasps the waffle with `basis=[npc_behavior]` and `purpose=npc_behavior.n2`. That action reveals the cookie and chocolate bar at the same coffee table. At round 17, the answer is rewritten as a movement pattern from the breakfast table to the garden coffee table; after replacing the waffle, the robot navigates back there in round 18 to retrieve the remaining foods.

This is a change in the text of the same answer identifier. Its probability and status were already 1.0 and `supported`, and remain so across these updates; the model does not report a probability reversal or name the hidden visitor type. The episode finishes in 28 rounds. The trace illustrates evidence collected for one question informing another, while also showing that semantic changes need not be reflected in the model's confidence or status labels.