Evaluating Long-Horizon Robot Planning under Uncertainty

Harsh Muriki*[1], Gayathri Aravindan[1], Yuchen Zheng[1], Sonia Chernova[1]

***Abstract*—Household robots must successfully navigate imperfect circumstances, both in their internal models and the external world. We introduce a benchmark of seven long-horizon household tasks in Habitat that vary initial robot memory and environment configurations. We evaluate adaptations of ReAct, VLM-TAMP, and Tru-POMDP within a shared execution framework across 90 tasks–environment–memory combinations, totaling 1,350 episodes. Our evaluation measures task success, partial completion, and action efficiency, along with rule-based analysis of execution traces. ReAct achieves the highest overall task success (45.3%) and completion (71.7%), while having accurate initial memory does not consistently yield the highest performance for every planner. We identify recurring placement errors, precondition errors, and redundant actions through trace analysis. These findings highlight differences in how the evaluated planners use initial knowledge and respond to uncertainty.**

***Index Terms*—Task and Motion Planning, Uncertainty, Planning, VLM**

I. Introduction

In real homes, robots rarely hold accurate beliefs about their surroundings. An object remembered on a countertop may have been moved into a cabinet, replaced with a similar item, or removed entirely. Each situation demands a different response: searching for the object, selecting a substitute, revising the plan, or recognizing that the goal is infeasible. Robust planning therefore requires robots to gather information, update their internal world representations, and adapt their plans as new observations become available [1], [2].

Task and motion planning (TAMP) connects discrete high-level task decisions with geometrically feasible robot motions [3]; under partial observability, these decisions must also account for what the robot knows about the environment. Existing approaches address this challenge through different mechanisms. ReAct [4] interleaves reasoning and action, while VLM-TAMP [5] uses vision-language-model-generated subgoals to guide geometric planning. LLaMAR studies feedback-driven planning under partial observability [6], and Tru-POMDP [7] explicitly models uncertainty through hypotheses and belief-space planning. Although these approaches offer different mechanisms for responding to uncertainty, their comparative strengths and limitations are difficult to characterize when evaluated on different tasks, environments, and execution frameworks. Benchmarks such as PARTNR [8] and Embodied Agent Interface [9] examine collaboration, embodied decision making, and execution errors. We investigate a complementary question: *How do different planning approaches respond when initial robot memory and environment configuration are systematically varied within a shared execution framework?*

In this work, we investigate how different planning architectures respond to uncertainty in long-horizon household tasks. Specifically, we examine two complementary sources of uncertainty: *initial memory uncertainty*, arising from incomplete or outdated knowledge of the environment, and *environmental variation*, including unavailable objects, acceptable substitutions, hidden targets, and distractors. By systematically varying these conditions within a shared execution framework, we evaluate whether planners achieve their goals, how efficiently they adapt, and what causes execution failures.

We introduce a benchmark of seven long-horizon household tasks implemented in Habitat Sim [10]–[12] using PARTNR's high-level robot skills. We adapt three planning approaches—ReAct, VLM-TAMP, and Tru-POMDP—to a common execution framework and evaluate them across 90 task–environment–memory combinations, with five repetitions per combination and planner, totaling 1,350 episodes. We measure task success, partial completion, and action efficiency relative to human reference executions, and analyze execution traces to identify recurring failure and inefficiency patterns.

Our results show that overall performance and sensitivity to uncertainty differ across the three planner adaptations. ReAct achieves the highest task success (45.3%) and completion (71.7%), outperforming approaches that incorporate explicit symbolic or belief-space planning. However, accurate initial memory does not consistently improve performance across planners, and all three approaches exhibit substantial execution inefficiencies. Trace analysis further reveals distinct limitations, including incorrect object placements, omitted actions, redundant exploration, and premature task termination. Together, these findings highlight that explicit uncertainty representations and replanning mechanisms alone do not guarantee robust or efficient execution.

In summary, our main contributions are:

- A benchmark of seven long-horizon household tasks with controlled variations in initial robot memory and environmental conditions.
- A comparative evaluation of three planner adaptations, evaluated across 1,350 episodes using success, completion and action-efficiency metrics.
- A trace-based analysis identifying recurring failure and inefficiency patterns.

*Corresponding author

[1]Georgia Institute of Technology, Atlanta, GA, USA {vmuriki3, garavindan3, fzheng, chernova}@gatech.edu.

II. Related Work

A. Task Planning Under Uncertainty

Robotic task planning under uncertainty requires reasoning with imperfect information and adapting plans as new evidence becomes available. Belief-space approaches maintain probabilistic state estimates and interleave planning, information gathering, and execution. CoCo-TAMP [13] incorporates language-model commonsense knowledge into belief estimation, while Tru-POMDP [7] combines LLM-generated hypotheses with Bayesian belief updates and online POMDP planning. CLUE [14] refines task-relevant hypotheses through closed-loop perception and interaction, while CURE [15] estimates uncertainty in LLM-generated plans to inform whether tasks should be executed autonomously or completed with human assistance.

B. Language- and Vision-Language-Based Task Planning

Large language models (LLMs) provide semantic knowledge for translating natural-language instructions into robot action sequences, while vision-language approaches incorporate perceptual observations directly into the planning process. SayCan [16] combines language-based skill selection with learned affordances, while ProgPrompt [17] grounds plan generation through prompts specifying available actions, objects, and executable examples. The LLM planners in PARTNR-Planner [8] select perception and action tools using task instructions and an updated world graph.

Hybrid frameworks combine language-model reasoning with structured representations and classical planning. LLM+P [18] translates natural-language planning problems into PDDL and invokes a classical planner. VLM-TAMP [5] uses VLM-generated subgoals to guide task and motion planning, querying the VLM again when a subgoal cannot be refined into a feasible plan. NESYRO [19] combines language-model code generation with symbolic verification and interactive validation to establish task-relevant preconditions.

III. Benchmark Tasks

We introduce a benchmark comprising **seven** long-horizon household tasks implemented in Meta's Habitat simulator. The tasks span water fetching, package preparation, book collection, grocery organization, guest table preparation, meal delivery, and household cleanup. Each requires multi-step execution across multiple rooms and furniture locations, involving object transport, spatial placement, and object-state manipulation. All tasks draw from a shared library of twelve high-level robot skills, implemented using PARTNR-Planner [8]. BASE task instructions are provided in Appendix VIII.

To systematically evaluate planning under uncertainty, we vary two aspects of each episode: the robot's initial memory and the environment configuration.

Initial robot memory. Each episode initializes the planner's internal world representation under one of three memory conditions. *Accurate memory* provides correct task-relevant information about object locations, containment relations, and states. *Incomplete memory* omits selected task-relevant information that must be acquired during execution. *Outdated memory* retains previously valid information that no longer matches the environment, requiring the planner to revise its representation using observations and action feedback.

Environment configurations. We evaluate robustness to environmental variation across five scenario configurations, summarized in Table I. These configurations vary object availability, accessibility, and the presence of distractors. In the baseline configuration (**BASE**), all target objects are present and directly accessible, with no distractors. The remaining configurations introduce distinct challenges or sources of task execution uncertainty: **SUB** replaces target objects with acceptable substitutes, requiring the robot to identify alternative ways to satisfy task goals; **ABS** removes target objects without providing suitable substitutes, requiring the robot to accept that a task goal is infeasible; **HID** hides target objects inside closed receptacles, requiring additional actions to locate and retrieve them; and **DIS** introduces irrelevant objects that may interfere with target identification and selection. Each configuration is instantiated for applicable tasks and evaluated under all three memory conditions.

TABLE I
FIVE ENVIRONMENT CONFIGURATIONS USED TO EVALUATE ROBUSTNESS.

Scenario	Target	Substitute	Absent	Hidden	Distractor
BASE	✓				
SUB		✓			
ABS			✓		
HID	✓			✓	
DIS	✓				✓

In ABS scenarios, a desirable response is to search plausible locations, report the unavailable resource, and complete the remaining achievable task requirements rather than repeatedly attempting an infeasible action. Evaluation criteria requiring unavailable objects are excluded, and only the remaining achievable requirements are evaluated.

We evaluate each approach using task success, task completion, and execution efficiency. Let $i \in \{1, \ldots, N\}$ index evaluation episodes, K_i denote the number of benchmark-defined evaluation criteria for episode i, and $z_{i,j} \in \{0, 1\}$ indicate whether criterion j is satisfied.

Task success and completion. We measure task completion as the fraction of evaluation criteria satisfied:

$$C_i = \frac{1}{K_i} \sum_{j=1}^{K_i} z_{i,j}.$$

An episode is considered successful (task success, $S_i = 1$) only if all criteria are satisfied; otherwise, $S_i = 0$. We report the mean task success rate, $\frac{1}{N} \sum_i S_i$, and mean task completion rate, $\frac{1}{N} \sum_i C_i$. Both metrics range from 0 to 1 and weight all episodes equally. For episodes that time out, we use the latest recorded evaluation outcomes and execution counts.

Execution efficiency. We adapt Success weighted by Path Length (SPL) [20] to measure efficiency in terms of high-level robot skill attempts rather than navigation distance. Specifically, we define action-based SPL ($\mathrm{SPL}_{\mathrm{act}}$), which combines binary task success with action efficiency relative to a human reference execution:

$$\mathrm{SPL}_{\mathrm{act}} = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{A_i^{\mathrm{ref}}}{\max(A_i, A_i^{\mathrm{ref}})},$$

where A_i is the number of skill attempts recorded during episode i, and A_i^{ref} is the corresponding reference action count from a human demonstration of the task variant.

Because binary success does not capture partial progress in long-horizon tasks, we additionally introduce task-completion-weighted SPL ($\text{TC-SPL}_{\mathrm{act}}$), which replaces binary success with the fraction of evaluation criteria satisfied:

$$\text{TC-SPL}_{\mathrm{act}} = \frac{1}{N}\sum_{i=1}^{N} C_i \frac{A_i^{\mathrm{ref}}}{\max(A_i, A_i^{\mathrm{ref}})}.$$

This metric rewards partial task completion while penalizing unnecessary skill attempts.

Failure and inefficiency analysis. To characterize the sources of execution failures and inefficiencies, we apply rule-based analysis to episode traces, following prior work on embodied-agent error analysis [9], [21]. We identify recurring patterns of failure and avoidable or redundant actions, and compare their prevalence across planners, initial memory conditions, and environment configurations. Categories are not mutually exclusive; their definitions are provided in Appendix VIII.

IV. Experimental Setup

A. Compared Methods

We adapt three existing planning approaches to the Habitat/PARTNR framework, using a shared environment representation and library of executable robot skills to enable consistent comparison.

ReAct adaptation. We extend the zero-shot ReAct planner [4] provided by PARTNR-Planner [8] to incorporate visual observations alongside task instructions, world graph descriptions, and action history. At each decision step, the VLM generates a textual rationale and selects the next high-level robot skill. Execution feedback and updated observations are incorporated into subsequent decisions, enabling the planner to adapt its actions during execution. We refer to this multimodal adaptation as **ReAct**.

VLM-TAMP adaptation. We adapt the hierarchical planning approach of VLM-TAMP [5] to Habitat. Given task instructions, scene images, and the robot's current scene graph, a VLM generates intermediate subgoals expressed as grounded predicates. A symbolic planner then uses a hand-crafted PDDL domain [22] to generate executable skill sequences. Execution feedback triggers symbolic replanning or VLM-based subgoal revision when necessary. Our adaptation preserves this hierarchical architecture while using PARTNR's predefined skills for execution and incorporating the robot's initial memory and subsequent observations. We refer to this approach as **VLM-TAMP**.

Tru-POMDP adaptation. We adapt Tru-POMDP [7] to Habitat. The approach maintains uncertainty over task goals and unobserved object locations using an LLM-based *Tree of Hypotheses*, which constructs a weighted particle belief from textual scene observations, initial robot memory, and interaction history. Following each action, the belief is updated by predicting action effects, eliminating hypotheses inconsistent with observations, and generating new hypotheses when the remaining probability mass falls below a threshold. A DESPOT-style belief-tree search implemented in Python selects actions using a deterministic symbolic transition model and an adapted rollout policy. We extend the planning domain to support exploration and object-state changes. We refer to this adaptation as **Tru-POMDP**.

We evaluate all three approaches across seven household tasks, varying initial robot memory and environment configurations. Each applicable task–memory–environment combination is repeated five times per approach, yielding 1,350 episodes in total. All approaches use **GPT-5.6 Luna** with high reasoning effort as their underlying language or vision-language model, selected to balance planning accuracy, inference cost, and latency.

V. Results and Discussion

We compare the three planning approaches in terms of overall task performance, sensitivity to initial memory and environmental variation, and recurring failure and inefficiency patterns. Together, these analyses characterize how different planning architectures handle uncertainty during long-horizon household tasks.

A. Overall Task Performance

Figure 1 compares the three planners across task success, task completion, and action-weighted efficiency metrics. ReAct achieves the highest overall performance, with a task success rate of 45.3% and mean task completion of 71.7%, compared with 29.1% and 48.3% for VLM-TAMP, and 32.7% and 56.2% for Tru-POMDP. These results indicate that ReAct is more effective at completing long-horizon tasks than the two approaches incorporating explicit symbolic or belief-space planning.

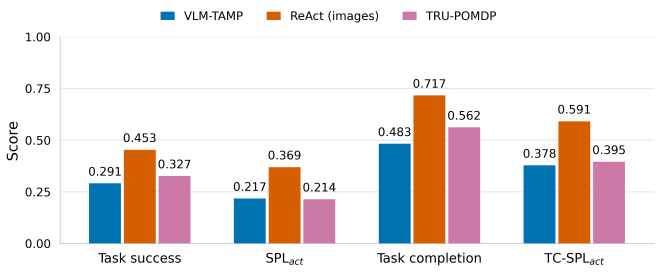


Fig. 1. Overall task success, action-weighted SPL, task completion, and $\text{TC-SPL}_{\mathrm{act}}$ across the three planning approaches.

However, incorporating action efficiency via the SPL metric results in lower performance scores across all three methods. For ReAct, action-weighted SPL is 36.9%, compared with a task success rate of 45.3%, while TC-SPL$_{\text{act}}$ is 59.1%, compared with 71.7% task completion, showing a reduction. Similar reductions occur for VLM-TAMP and Tru-POMDP, indicating that all three approaches execute additional skill attempts relative to human reference executions.

These findings highlight an important distinction between achieving task goals and executing them efficiently. Although ReAct achieves the highest overall performance, none of the evaluated approaches consistently produces efficient executions. We examine the sources of these inefficiencies in Section V-D.

B. Effect of Initial Memory

Table II reports TC-SPL$_{\text{act}}$ across the three initial memory conditions. The results reveal substantially different sensitivities to memory quality across planning approaches.

ReAct benefits strongly from accurate initial memory, achieving 0.71 compared with 0.55 under incomplete memory and 0.52 under outdated memory. This decline suggests that ReAct relies heavily on the correctness and completeness of its initial world representation, despite its ability to incorporate observations and action feedback during execution.

TABLE II
TC-SPL$_{act}$ UNDER DIFFERENT INITIAL-MEMORY CONDITIONS.

Method	Accurate	Incomplete	Outdated
VLM-TAMP	0.34	0.39	0.41
ReAct	0.71	0.55	0.52
TRU-POMDP	0.39	0.32	0.47

In contrast, VLM-TAMP and Tru-POMDP do not exhibit the same trend. VLM-TAMP achieves its highest score with outdated memory (0.41), compared with 0.34 under accurate memory. Similarly, Tru-POMDP achieves 0.47 with outdated memory, compared with 0.39 under accurate memory and 0.32 under incomplete memory. Thus, more accurate initial knowledge does not consistently translate into better task performance for these approaches.

One possible explanation is that discrepancies between memory and observations trigger additional exploration or replanning, potentially revealing information useful for completing the task. However, the aggregate results do not establish whether this mechanism accounts for the observed improvements. More generally, these findings suggest that planning performance depends not only on the quality of initial knowledge, but also on how a planner uses, verifies, and revises that knowledge during execution.

C. Effect of Environmental Variations

Figure 2 shows changes in TC-SPL$_{\text{act}}$ relative to the baseline environment (BASE), averaged across the three initial memory conditions. The results reveal two distinct patterns: environmental changes that consistently degrade performance and those whose effects depend strongly on the planning architecture.

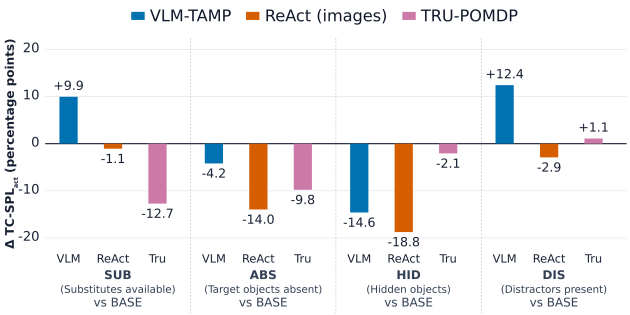


Fig. 2. ΔTC-SPL$_{act}$ from BASE across environment perturbations, averaged over all three memory conditions and reported in percentage points.

Absent and hidden objects. Removing target objects without providing suitable substitutes (ABS) reduces performance across all three planners, with the largest decrease observed for ReAct (14.0 percentage points). Similarly, hiding objects inside closed receptacles (HID) reduces performance for all methods, with ReAct experiencing the largest decline (18.8 percentage points). These configurations require planners to recognize that expected objects are unavailable or not directly accessible and to adjust their behavior accordingly.

Inspection of unsuccessful ReAct executions in ABS scenarios reveals instances of continued exploration and repeated visits to previously searched locations. Because no suitable object exists, these actions cannot resolve the missing-object condition and instead increase execution cost. This behavior illustrates a limitation of reactive replanning: adapting actions based on new observations does not necessarily provide an effective mechanism for determining when further exploration is unlikely to succeed.

Substitutions and distractors. In contrast, substitutions (SUB) and distractors (DIS) produce markedly different outcomes across planners. With acceptable substitutes, VLM-TAMP improves by 9.9 percentage points relative to BASE, ReAct changes minimally (-1.1 points), and Tru-POMDP declines by 12.7 points. Similarly, distractors increase VLM-TAMP's score by 12.4 points, while producing smaller changes for ReAct (-2.9 points) and Tru-POMDP (+1.1 points).

These results indicate that environmental perturbations are not uniformly detrimental. Whereas hidden and absent targets consistently challenge all three approaches, substitutions and distractors interact differently with each planner. The improvements observed for VLM-TAMP are particularly noteworthy, although further trace-level analysis is needed to determine whether they reflect differences in object selection, subgoal generation, or other planning behaviors.

Overall, these findings demonstrate that robustness to environmental uncertainty cannot be characterized by a single performance measure. Different forms of uncertainty place different demands on exploration, object selection, and plan

% of unsuccessful runs 0% 100%

Finding	VLM-TAMP n=319	ReAct n=246	TRU-POMDP n=303
Object Not Placed	56.4%	23.6%	45.2%
Missing Action	46.1%	37.4%	36.3%
Placement Error	44.5%	53.7%	16.5%
Object Not Found	30.1%	19.9%	18.5%
Insufficient Planning	12.5%	1.6%	17.5%
No Hypotheses	0.0%	0.0%	26.4%
Wrong Object	2.8%	1.6%	5.3%
Wrong Order	0.0%	5.3%	0.0%
Failure To Stop	0.0%	3.7%	0.7%

Fig. 3. Failure-finding rates across planning approaches, reported as the percentage of unsuccessful episodes exhibiting each failure; categories may overlap.

revision, with substantially different consequences across planning architectures.

D. Failure and Inefficiency Analysis

To better understand the performance differences, we analyze execution traces for recurring failure and inefficiency patterns. Figure 3 reports failure findings as percentages of unsuccessful episodes, while Figure 4 reports inefficiency findings across all episodes. Because categories are not mutually exclusive, reported rates need not sum to 100%.

Task execution failures. The most prevalent failures involve objects not being placed at their required destinations, omitted actions, and incorrect placements. However, their relative frequencies differ substantially across planners.

VLM-TAMP exhibits the highest rate of object-not-placed findings (56.4% of unsuccessful episodes), compared with 23.6% for ReAct and 45.2% for Tru-POMDP. Inspection of VLM-TAMP traces reveals cases in which execution terminates after the generated skill sequence is exhausted, despite unmet task requirements. Although the system supports replanning when execution encounters difficulties, completing an executable subgoal sequence does not necessarily trigger verification of the overall task objective. This exposes a limitation of hierarchical planning when subgoal completion and task-level success are not consistently reconciled.

Missing-action findings are common across all three approaches (36.3–46.1% of unsuccessful episodes). These include omitted object-state changes, such as failing to turn off a lamp or clean a plate, even when other components of the task are completed. Such failures suggest that maintaining coverage of all task requirements remains challenging during long-horizon execution.

% of all runs 0% 100%

Finding	VLM-TAMP n=450	ReAct n=450	TRU-POMDP n=450
Unnecessary Actions	22.2%	30.4%	52.4%
Precondition Error	12.0%	46.0%	23.1%
Repeated Actions	29.3%	2.4%	41.1%
Failure To Stop	0.0%	2.4%	3.6%
Hallucinated Object	0.0%	0.2%	0.0%

Fig. 4. Inefficiency-finding rates across planning approaches, reported as the percentage of all episodes exhibiting each inefficiency; categories may overlap.

Placement errors are most frequent for ReAct (53.7%), followed by VLM-TAMP (44.5%) and Tru-POMDP (16.5%). For example, in an inspected ReAct execution, the planner places a package on a side table rather than the requested center table, despite access to visual observations. This illustrates that visual grounding alone does not ensure correct destination selection. However, the lower placement-error rate for Tru-POMDP does not necessarily imply greater placement accuracy, as planners may differ in the number of placement actions attempted.

Execution inefficiencies. Figure 4 identifies three prominent inefficiency patterns: unnecessary actions, precondition errors, and repeated actions. These patterns reveal additional distinctions between the approaches.

Unnecessary actions occur in 22.2% of VLM-TAMP episodes, 30.4% of ReAct episodes, and 52.4% of Tru-POMDP episodes. Inspected traces include cases in which planners perform additional exploration despite already possessing accurate task-relevant information. This suggests that access to reliable memory does not necessarily prevent redundant information gathering.

Precondition errors are most frequent for ReAct (46.0%), compared with 12.0% for VLM-TAMP and 23.1% for Tru-POMDP. For example, ReAct sometimes attempts to manipulate objects without first navigating to the object's location. VLM-TAMP's lower rate is consistent with the potential benefit of explicit symbolic precondition checking, although these results alone do not establish a causal relationship.

Repeated-action findings exhibit a contrasting pattern, occurring in 41.1% of Tru-POMDP episodes and 29.3% of VLM-TAMP episodes, but only 2.4% of ReAct episodes. Thus, approaches with fewer precondition errors may nevertheless incur substantial inefficiency through repeated actions. In particular, Tru-POMDP's comparatively high rates of unnecessary and repeated actions suggest that explicit belief-space reasoning does not automatically translate into efficient exploration and execution.

VI. Summary and Takeaways

Together, these analyses reveal distinct limitations across planning architectures:

ReAct achieves the highest task performance but remains vulnerable to incorrect placements, precondition violations, and unproductive exploration when objects are unavailable.

VLM-TAMP reduces precondition errors but can terminate after completing generated subgoals without satisfying the full task.

Tru-POMDP explicitly represents uncertainty but exhibits frequent unnecessary and repeated actions.

These findings suggest that robust long-horizon planning requires more than reactive replanning, symbolic feasibility checking, or explicit uncertainty representation in isolation. Effective systems must also determine when additional information is needed, verify progress toward task-level goals, and recognize when continued execution or exploration is unlikely to be productive.

VII. Limitations and Conclusion

Our evaluation focuses on high-level planning through a shared skill library. We do not separately characterize low-level execution failures, such as infeasible grasps, unreachable poses, or motion-control errors. The study is limited to seven simulated household tasks and our adaptations of three planning approaches. Furthermore, rule-based findings identify observable trace patterns rather than establish causal explanations.

The results show that planning performance depends on both initial robot memory and external environment variations, with different sensitivities across the three baselines. ReAct achieves the highest overall success and completion, but all approaches exhibit gaps between task performance and action efficiency.

As future work, we will develop planning systems that adaptively decide when to gather information, revise plans, or stop. We will evaluate these systems on our benchmark to assess whether they reduce failures and inefficiencies.

References

[1] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1194–1227, 2013.

[2] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," 2020. [Online]. Available: https://arxiv.org/abs/1911.04577

[3] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, no. 1, pp. 265–293, 2021.

[4] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," 2023. [Online]. Available: https://arxiv.org/abs/2210.03629

[5] Z. Yang, C. Garrett, D. Fox, T. Lozano-Pérez, and L. P. Kaelbling, "Guiding long-horizon task and motion planning with vision language models," *arXiv preprint arXiv:2410.02193*, 2024.

[6] S. Nayak, A. M. Orozco, M. Ten Have, J. Zhang, V. Thirumalai, D. Chen, A. Kapoor, E. Robinson, K. Gopalakrishnan, J. Harrison *et al.*, "Long-horizon planning for multi-agent robots in partially observable environments," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[7] W. Tang, X. He, Y. Huang, Y. Xiao, C. Lu, and P. Cai, "Tru-pomdp: Task planning under uncertainty via tree of hypotheses and open-ended pomdps," 2026. [Online]. Available: https://arxiv.org/abs/2506.02860

[8] M. Chang, G. Chhablani, A. Clegg, M. D. Cote, R. Desai, M. Hlavac, V. Karashchuk, J. Krantz, R. Mottaghi, P. Parashar, S. Patki, I. Prasad, X. Puig, A. Rai, R. Ramrakhya, D. Tran, J. Truong, J. M. Turner, E. Undersander, and T.-Y. Yang, "Partnr: A benchmark for planning and reasoning in embodied multi-agent tasks," in *International Conference on Learning Representations (ICLR)*, 2025, alphabetical author order.

[9] M. Li, S. Zhao, Q. Wang, K. Wang, Y. Zhou, S. Srivastava, C. Gokmen, T. Lee, L. E. Li, R. Zhang, W. Liu, P. Liang, L. Fei-Fei, J. Mao, and J. Wu, "Embodied agent interface: Benchmarking llms for embodied decision making," 2025. [Online]. Available: https://arxiv.org/abs/2410.07166

[10] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[11] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[12] X. Puig, E. Undersander, A. Szot, M. D. Cote, R. Partsey, J. Yang, R. Desai, A. W. Clegg, M. Hlavac, T. Min, T. Gervet, V. Vondrus, V.-P. Berges, J. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi, "Habitat 3.0: A co-habitat for humans, avatars and robots," 2023.

[13] Y. Kim, R. Arora, R. Martín-Martín, P. Stone, B. Abbatematteo, and Y. Sung, "Large-language-model-guided state estimation for partially observable task and motion planning," *arXiv preprint arXiv:2603.03704*, 2026.

[14] Z. Ravichandran, J. Diller, F. Cladera, V. Murali, G. J. Pappas, and V. Kumar, "Actively resolving contextual uncertainty for underspecified tasks in natural language," 2026. [Online]. Available: https://arxiv.org/abs/2609.30428

[15] S. Yin, C. Bai, Z. Zhang, J. Jin, X. Zhang, C. Zhang, and X. Li, "Towards reliable llm-based robot planning via combined uncertainty estimation," 2025. [Online]. Available: https://arxiv.org/abs/2510.08044

[16] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[17] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," 2022. [Online]. Available: https://arxiv.org/abs/2209.11302

[18] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone, "Llm+p: Empowering large language models with optimal planning proficiency," 2023. [Online]. Available: https://arxiv.org/abs/2304.11477

[19] S. Ahn, W. Choi, J. Lee, J. Park, and H. Woo, "Towards reliable code-as-policies: A neuro-symbolic framework for embodied task planning," *Advances in Neural Information Processing Systems*, vol. 38, pp. 75 428–75 459, 2026.

[20] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva *et al.*, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[21] R. Yang, H. Chen, J. Zhang, M. Zhao, C. Qian, K. Wang, Q. Wang, T. V. Koripella, M. Movahedi, M. Li, H. Ji, H. Zhang, and T. Zhang, "Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents," 2025. [Online]. Available: https://arxiv.org/abs/2502.09560

[22] C. Aeronautiques, A. Howe, C. Knoblock, I. D. McDermott, A. Ram, M. Veloso, D. Weld, D. W. Sri, A. Barrett, D. Christianson *et al.*, "Pddl—the planning domain definition language," *Technical Report, Tech. Rep.*, 1998.

VIII. Appendix

A. Tasks

These descriptions summarize BASE goals; other variants can change the required objects.

- **T1 – Water fetching.** Turn off all the lights in the bedroom and bring a full jug of water to either of the two living-room tables.
- **T2 – Package preparation.** Move both packages to the living-room center table and place them next to each other. Bring the scissors and place beside them to open the packages.
- **T3 – Book collection.** Collect the 2 green books & place them on the living room side table and turn off the laptop as well.
- **T4 – Grocery organization.** Put all the 3 fruits in the fruit basket and the 2 drink containers on the kitchen counter.
- **T5 – Guest table preparation.** Get soap to clean both glasses, fill them, and place them on the living-room side table for our guests to drink. Get a white plate and place it next to them.
- **T6 – Meal delivery.** Heat up the bread, bring it to the desk in the bedroom the robot was initially at, bring a water bottle and a clean hand towel and place them next to the bread. Turn off the living-room lights.
- **T7 – Household cleanup.** Put away all toys from the living room into the closet's wardrobe, wash and put away the dirty dishes from sink inside, and turn off the living-room and kitchen lights.

B. Failures

A failure occurs when execution ends without satisfying the task's evaluation criteria. The following labels describe behavior observed in failed runs.

- **Placement error.** An object was placed in the wrong location.
- **Object not placed.** The robot found an object but never put it where the task needs it: it never moved it, or was still holding it at the end.
- **Wrong object.** The planner used a distractor instead of the actual task object. Approved substitutes are correct.
- **Missing action.** A required action on an object was omitted or skipped, such as heating it or switching it off. An attempted action that failed to execute does not count.
- **Wrong order.** A required step was done, but only after the step it had to come before.
- **Insufficient planning.** The planner got stuck or continued unproductive execution instead of advancing.
- **Object not found.** An object that a failed requirement needs was never found.
- **No hypotheses.** TRU-POMDP only: the Tree of Hypotheses returned no usable goal guesses, so the planner stopped before taking any action.
- **Failure to stop.** The planner kept trying to plan or search after evidence indicated that the subgoal should be abandoned.

C. Inefficiencies

An inefficiency is a behavior that wastes actions or execution effort without useful progress toward the task. It can occur in both successful and failed runs.

- **Hallucinated object.** Tried to use an object that does not exist. Using a stale object from outdated memory (OUT variants) does not count.
- **Unnecessary actions.** Completed actions with no useful task, exploration, or recovery purpose.
- **Precondition error.** Tried an action without valid prerequisites, such as picking an object before navigating within reach.
- **Repeated actions.** The planner output the exact same action, or the exact same sequence of actions, several times in a row.
- **Failure to stop.** Uses the definition above and is categorized as an inefficiency when the run is successful.