# Deployment-time Selection of Prompts for LLM-informed Object Search in Partially-Known Environments

Abhishek Paudel and Gregory J. Stein

***Abstract*— We present an approach for deployment-time selection of best-performing prompts and LLMs for LLM-informed object search in partially-known environments. Leveraging recent progress in both LLM-informed model-based planning and deployment-time behavior selection, we enable fast bandit-like selection of best-performing prompts and LLMs and demonstrate improved deployment-time performance in object search tasks. Experiments in simulated ProcTHOR household environments show that our bandit-like selection approach results in 6.1% lower average cost and 40.6% lower average cumulative regret over baseline UCB bandit selection.**

## I. Introduction

We consider the problem of selecting best prompts and large language models (LLMs) during deployment when a robot is deployed for LLM-informed object search tasks in partially-known environments. LLMs have been increasingly used in many robotics applications because of the commonsense knowledge about the world—e.g., where might a fork be typically found in a house—that can be obtained from them via prompting to guide robot behavior. Yet, the process of obtaining such knowledge from LLMs is often sensitive to prompts used to query LLMs, and hence requires trial and error to find the best of a set of candidate prompts—a procedure that is costly for robot navigation tasks.

In such scenarios, the agent's performance in object search tasks depends on the choice of prompting strategy and LLM model, since choosing different prompts or LLMs can result in varied performance when deployed, particularly when the deployment-time environments differ from those that were considered when designing such prompts. As such, selecting only a single prompting strategy or LLM in advance will not always elicit the best deployment-time performance. Instead, the robot should be able to choose from more than one prompting strategy or LLM and evaluate each of these to pick the best during deployment. However, the process of deploying and repeatedly trying out prompting strategies or LLMs until a clear winner emerges can be problematically time consuming in general, requiring many trials to choose between them. Recent work in the space of point-goal navigation [1] presents *offline alt-policy replay*, in which model-based counterfactual reasoning can be used to afford choosing the best of a family of learning-informed navigation policies, a strategy we seek to leverage for the purpose of prompt and LLM selection in this work.

To achieve effective object search performance by selecting the best-performing prompts during deployment, we

Abhishek Paudel and Gregory J. Stein are with the Department of Computer Science, George Mason University, Fairfax, Virginia, USA. `{apaudel4, gjstein}@gmu.edu`

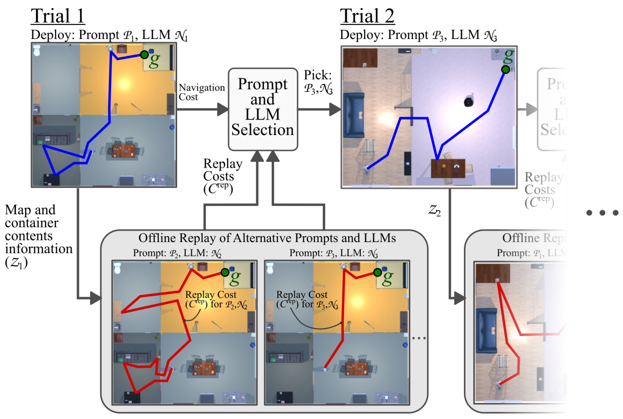
Trial 1
Deploy: Prompt $\mathcal{P}_1$, LLM $\mathcal{N}_1$
g
Navigation Cost
Prompt and LLM Selection
Pick: $\mathcal{P}_3, \mathcal{N}_3$
Trial 2
Deploy: Prompt $\mathcal{P}_3$, LLM $\mathcal{N}_3$
Replay Costs ($C^{rep}$)
Map and container contents information ($Z_1$)
$Z_2$
Offline Replay of Alternative Prompts and LLMs
Prompt: $\mathcal{P}_2$, LLM: $\mathcal{N}_2$
Prompt: $\mathcal{P}_3$, LLM: $\mathcal{N}_3$
Replay Cost ($C^{rep}$) for $\mathcal{P}_2, \mathcal{N}_2$
Replay Cost ($C^{rep}$) for $\mathcal{P}_3, \mathcal{N}_3$
Prompt: $\mathcal{P}_1$

Fig. 1. **Deployment-time Selection of Prompts and LLMs**: Information collected during trial (e.g. objects found in explored containers) and the map are used to replay the robot behavior informed by all other alternative prompts and LLMs, the outcomes of which are used for selection of best prompts and LLMs.

require an approach that both informs and is informed by an LLM: with which we can plan using the commonsense world knowledge of LLMs and also introspect during deployment so as to quickly allow the system to select the best-performing prompting strategy or LLM. It is a key insight of this work that the model-based planning framework for LLM-informed object search in partially-known environments by Hossain et al. [2] is amenable for such introspection while being suitable for effective LLM-informed object search tasks. Their model-based planning framework uses a high-level action abstraction and hence affords *offline replay* [1], facilitating deployment-time evaluation of prompts and LLMs for object search tasks—the outcomes of which can then be used to quickly select the best-performing prompts and LLMs.

In this work, we present an approach for deployment-time selection of the best prompts and LLMs for LLM-guided object search tasks (Fig. 1). Leveraging the model-based planning framework by Hossain et al. [2] in which robot's actions correspond to revealing unexplored containers to look for a target object and an LLM informs the statistics of uncertainty—namely, likelihood of finding an object of interest in a location—to *inform*, rather than *replace*, planning, our approach enables *fast deployment-time selection* of prompts and LLMs, a capability unique in this domain, via the *offline replay* approach of Paudel and Stein [1].

Experiments in simulated ProcTHOR environments demonstrate that our prompt selection approach enables quick selection of best-performing prompts and LLMs from a family of prompts and LLMs, resulting in 6.1% lower average cost and 40.6% lower average cumulative regret over baseline upper confidence bound (UCB) bandit selection.

## II. Related Work

**LLMs and VLMs for Object Search** Many recent works have explored the use of LLMs and vision language models (VLMs) and for object search tasks [2]–[8]. These works use LLMs or VLMs for their commonsense world knowledge to decide where to search [3]–[5]. These works generally design prompts for LLM/VLM as a part of development phase, and are not concerned with identifying the best prompts during deployment of such systems.

**Prompt Selection** Prompt selection, which falls under a broader area of prompt engineering [9], [10], deals with selecting the prompts that achieve the best LLM performance on downstream tasks [11]. While there are approaches that aim to select the best prompts from predesigned templates [10]–[13], these approaches focus selecting prompts that gets the best responses from LLMs on various benchmarks and hence are not suitable for deployment-time selection of prompts in LLM-informed object search tasks—the focus of this work.

## III. Problem Formulation

**Object Search in Partially-Known Environments** Our robot is tasked find a target object $g$ in a household environment in minimum expected cost, measured in terms of distance traveled. The environment consists of rooms, containers and objects. *Containers* are entities in the environment that can contain other objects: bed, dresser, countertop, etc. The containers are located in different rooms in the household environment. The belief state $b_t = \{m_t, q_t\}$ consists of the map $m_t$—with *a priori* known locations of rooms and containers but what objects exist in the containers are not known—and the robot pose $q_t$, both at time $t$. The robot must navigate to containers and search them to look for the target object. Unexplored containers form the robot's action space $\mathcal{A}$ and the robot's policy $\pi$ maps the belief state $b_t$ to a container search action $a_t \in \mathcal{A}(b_t)$. Our search policies are informed by LLMs and so depend upon the choice of LLMs and prompts used to query the LLMs.

We presume that the robot has access to a low-level navigation planner and controller that can be used to move about and interact with the environment. As such, the aim of our planner is to determine the sequence of container search actions that minimizes the expected cost of finding the target object. The performance of the robot during deployment is measured as the average distance traveled by the robot to find the target object across a sequence of trials, where each trial is held in a distinct map to find an object sampled uniformly at random from the environment.

**Prompt Selection** We consider that the robot's policy has access to multiple prompt templates and LLMs each represented as $\theta = (\mathcal{P}, \mathcal{N})$ where $\mathcal{P}$ denotes prompt template and $\mathcal{N}$ denotes LLM. As such, the robot has access to a family of search policies $\Pi = \{\pi_{\theta_1}, \pi_{\theta_2}, \cdots, \pi_{\theta_N}\}$ each with a unique prompt-LLM pair. The objective of prompt selection is to pick the policy with a prompt-LLM pair $\theta$ whose corresponding search actions result in minimum expected cost of finding target objects during deployment in partially-known environments:

$$\pi_\theta^* = \underset{\pi_\theta \in \Pi}{\operatorname{argmin}}\, \mathbb{E}[C(\pi_\theta)] \tag{1}$$

where $\mathbb{E}[C(\pi_\theta)]$ is the expected cost incurred by the robot upon using policy $\pi_\theta$ with a prompt-LLM pair $\theta$ during deployment. This problem can be formulated as a multi-armed bandit problem [14], solved via black-box selection algorithms like UCB [15] using Eq. (2):

$$\pi_\theta^{(k+1)} = \underset{\pi_\theta \in \Pi}{\operatorname{argmin}} \left[ \bar{C}_k(\pi_\theta) - c\sqrt{\frac{\ln k}{n_k(\pi_\theta)}} \right] \tag{2}$$

where $\bar{C}_k(\pi_\theta)$ is the average cost over trials 1-through-$k$ in which policy $\pi_\theta$ with prompt-LLM pair $\theta$ was selected, $n_k(\pi_\theta)$ is the number of times policy $\pi_\theta$ was selected until trial $k$, and $c > 0$ is a parameter controlling the balance between exploration and exploitation. However, such approaches can be slow to converge, requiring the robot to go through multiple trials of poor performance before the best policies can be identified. White-box approaches can accelerate selection [1], [16], but place requirements on the the types of abstraction for compatibility. It is our insight that LLM-informed planning strategies can be made compatible with such approaches and so can afford prompt and LLM selection in this setting.

## IV. Preliminaries: LLM-informed Model-Based Planning for Object Search

Here, we discuss LLM-informed model-based planning for object search by Hossain et al. [2] that is amenable to our prompt selection approach discussed in Section V.

LLM-informed object search by Hossain et al. [2] presents a model-based planning framework in which high-level actions correspond to searching the *containers*, which are entities in the environment that contain other objects: bed, dresser, countertop, etc. A search policy $\pi$ specifies the sequence of search actions the robot intends to take to find the target object. Each such search action $a_t \in \mathcal{A}(b_t)$ has an immediate cost of first traveling to the container—corresponding to a distance $D(b_t, a_t)$ computed via A* from the occupancy grid—and then searching the container for the target object, which has a (known) search cost $R_{\text{search}}(b_t, a_t)$. With a probability $P_S$, the container contains the target object and so the corresponding search action successfully finds the object. Otherwise, with probability $1 - P_S$, searching continues in other containers after picking another container search action. The expected cost of a search action $a_t$ under policy $\pi$ is computed using a Bellman equation:

$$\begin{aligned} Q_\pi(b_t, a_t \in \mathcal{A}(b_t)) &= D(b_t, a_t) + R_{\text{search}}(b_t, a_t) \\ &\quad + (1 - P_S(a_t)) Q_\pi(b'_t, \pi(b'_t)) \end{aligned} \tag{3}$$

The robot's policy $\pi(b_t) = \operatorname{argmin}_a Q_\pi(b_t, a \in \mathcal{A}(b_t))$ can be used to compute a search plan: the sequence of actions that minimizes the expected cost via Eq. (3) to find the target object. Using an LLM as the knowledge repository of

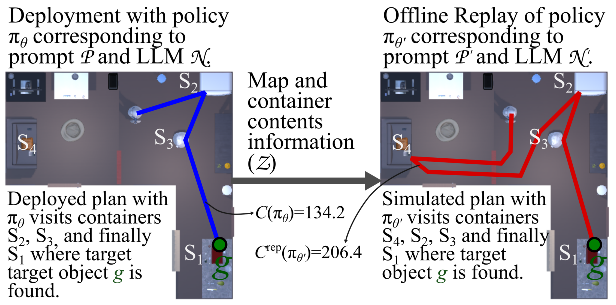
Deployment with policy $\pi_\theta$ corresponding to prompt $\mathcal{P}$ and LLM $\mathcal{N}$.
Offline Replay of policy $\pi_{\theta'}$ corresponding to prompt $\mathcal{P}'$ and LLM $\mathcal{N}'$.
$S_2$
$S_3$
$S_4$
$S_1$
$g$
Map and container contents information ($\mathcal{Z}$)
Deployed plan with $\pi_\theta$ visits containers $S_2$, $S_3$, and finally $S_1$ where target target object $g$ is found.
$C(\pi_\theta)$=134.2
$C^{\text{rep}}(\pi_{\theta'})$=206.4
Simulated plan with $\pi_{\theta'}$ visits containers $S_4$, $S_2$, $S_3$ and finally $S_1$ where target object $g$ is found.

Fig. 2. **Overview of Offline Replay**: Using the map and container contents information ($\mathcal{Z}$) obtained during deployment, we simulate a different plan corresponding to alternative prompt $\mathcal{P}'$ and LLM $\mathcal{N}'$ to get replay cost $C^{\text{rep}}$.

where common household objects might be located, an LLM is prompted to provide an estimate of the probability $P_S$, which is used to compute the expected cost via Eq. (3).

## V. Prompt Selection for LLM-informed Object Search

### A. Overview of Prompt Selection

When using an LLM to inform planning for object search, prompts used to query LLMs for object likelihood predictions would ideally result in effective performance. However, effectiveness of a plan in the context of object search in partially-known environments can only be realized after the robot executes them in the environments. Thus, a poor prompting strategy may only be identified as such after the robot deploys and relies upon that LLM and prompt combination—a costly strategy of trial-and-error for robot navigation tasks. Instead, if we could identify poor prompts while limiting the need to deploy them during deployment, we could rule them out quickly and prioritize selection of the best prompts to enable improved robot performance.

It is a key insight of this work that *offline replay* approach by Paudel and Stein [1] can be used to select between prompts without the robot having deploy the plans informed by LLMs using such family of prompts. While used by Paudel and Stein [1] in the context of point-goal navigation in partially-mapped environments to replay the behavior of alternative policies without having to deploy them, we adapt offline replay to determine what the robot would have done if it had instead used a different prompt or LLM to guide its behavior. Costs from offline replay (Fig. 2) of alternative prompts and LLMs, $\bar{C}_k^{\text{rep}}$ (averaged over trials 1-through-$k$) can then be used in UCB bandit-like selection strategy (Fig. 1) similar to Paudel and Stein [1] to pick the policy $\pi_\theta$ with prompt-LLM pair $\theta$ for trial $k+1$ as:

$$\pi_\theta^{(k+1)} = \underset{\pi_\theta \in \Pi}{\text{argmin}} \left[ \max\left( \bar{C}_k^{\text{rep}}(\pi_\theta), \bar{C}_k(\pi_\theta) - c\sqrt{\frac{\ln k}{n_k(\pi_\theta)}} \right) \right] \tag{4}$$

### B. Object Search during a Trial

In each trial $k$ when the robot is deployed in a partially-known environment to look for a target object $g$, it uses Eq. (4) to choose one of the policies $\pi_\theta^{(k)} \in \Pi$ with some prompt-LLM pair $\theta = (\mathcal{P}, \mathcal{N})$, uses the prompt $\mathcal{P}$ to query the LLM $\mathcal{N}$ for object likelihood $P_S$ and computes the next action $a_t$ corresponding to searching one of the unexplored containers using Eq. (3). The robot searches the container corresponding to action $a_t$ to look for the target object $g$, repeating planning and search each time the target object is not found. The total distance traveled by the robot to find the object is the cost $C_k(\pi_\theta)$ for trial $k$. The robot stores the information $\mathcal{Z}_k$ about the contents of all containers it explored and the known map of the environment to be used later for offline replay (Section V-C).

### C. Prompt Selection with Offline Replay of Alternative Prompts and LLMs

After a trial $k$ is complete, our robot uses policy $\pi_{\theta'}$ with alternative prompt-LLM pair $\theta'$ and computes an alternative search plan. However, deploying such alternative plans is expensive in general. Instead, we use the hindsight information $\mathcal{Z}_k$ about the location of the target object found in trial $k$ and the existing map to *replay* what the robot would have done if it had deployed a plan corresponding to an alternative prompt-LLM pair $\theta' = (\mathcal{P}', \mathcal{N}')$ (Fig. 2). Since we now know in advance which container contained the target object $g$ based on the information $\mathcal{Z}_k$ obtained after completion of trial $k$ and pessimistically assume that all other containers would not have contained the target object, we can compute the cost of following a separate policy: the length of the trajectory the robot would have taken by following the alternate search policy to find that target object. The average cost over trials of the offline-replayed plans, $\bar{C}_k^{\text{rep}}(\pi_{\theta'})$, for an alternative prompt-LLM pair $\theta'$ and average costs over trials of the chosen prompt-LLM pairs $\theta$ in trial $k$, $\bar{C}_k(\pi_\theta)$ are together used to pick the prompt and LLM in subsequent trials $k+1$ using Eq. (4).

## VI. Experiments and Results

### A. Experiment Design

We perform prompt selection experiments for LLM-informed object search simulated household environments based on ProcTHOR [17] dataset which consists of procedurally generated homes (see the Appendix for samples). Our robot has access to the underlying occupancy grid of the environment and what containers exist in what rooms, yet the contents of the containers are not known to the robot. The robot must travel to the container locations and search the containers to find the object of interest. We conduct experiments with multiple prompts, LLMs and object search policies which are discussed below. During deployment, our robot selects the best of these prompts, policies and LLMs.

**Policies** During deployment, the robot can chose between different policies as discussed below.

**LLM+MODEL** This is the LLM-informed model-based planner [2] that uses an LLM to obtain object likelihood probabilities and then uses Eq. (3) to select the best container search action as discussed in Section IV.

**LLM-DIRECT** This LLM-informed policy directly prompts the LLM to respond with the container the robot should search next, instead of asking for probabilities as we do with LLM+MODEL planner. As such, LLM-DIRECT policy does not use a planning framework to compute actions and instead directly executes actions picked by the LLM from a list of all available container search actions.

OPTIMISTIC+GREEDY This non-LLM policy optimistically assumes that all containers could contain the target object and greedily searches the nearest container, re-planning until the target object is found.

**LLM Variants** We experiment with two LLMs, GPT-4o Mini and Gemini 1.5 Pro, for object search tasks in ProcTHOR household environments that our selection approach can choose from during deployment.

**Prompt Design** We construct multiple prompts the our selection approach can choose from when querying the LLMs to guide the robot behavior. The prompt design for LLM+MODEL policy and LLM-DIRECT policy are slightly different since for LLM+MODEL we want the LLM to generate probability values for each container, while for the LLM-DIRECT policy, the LLM should directly output which container the agent should search next. While such prompts might be constructed with variations in language, context and the role that LLM should play in the interaction, each prompt includes a question asking the LLM to respond with either a probability value (for LLM+MODEL policy) or the name of the container to search (for LLM-DIRECT policy). We design three prompt templates for LLM+MODEL policy: P-CONTEXT-A, P-CONTEXT-B and P-MINIMAL, and one prompt template for LLM-DIRECT policy: P-DIRECT. Further details of prompt design and samples of these prompts are available in the Appendix.

## *B. Prompt Selection Results*

We compare our prompt/LLM selection approach based on *offline replay* as discussed in Sec. V, referred to as Replay Selection, against a baseline black-box UCB bandit selection approach. For these experiments, selection seeks to choose between all combinations of policies, prompts and LLMs discussed earlier. These combinations and their individual object search performances are discussed in the Appendix. To evaluate the statistical performance of selection, we generate 500 unique deployments by randomly permuting 100 trials from a set of 150 previously conducted single-trial results, expecting the robot to perform selection over these strategies separately for each sequence. While the UCB Selection uses only the deployment cost of a strategy to pick the policy-prompt-LLM combination for subsequent trials using Eq. (2), Replay Selection additionally uses the offline replay costs of all other policy-prompt-LLM combination to pick the strategy for next trial via Eq. (4).

In Fig. 3, we report the *average navigation cost*, which corresponds to the average of navigation costs incurred in trials 1-through-$k$, averaged over all 500 deployments. The *cumulative regret*, also shown in Fig. 3, tracks performance over time as the cumulative difference between the selection-based policy and a Best Performance oracle that knows in advance which prompt/LLM is best: LLM+MODEL/P-CONTEXT-A/Gemini (see Table I in Appendix). Our results demonstrate

| Metric | Selection Approach | Num of Trials ($k$) | | |
|---|---|---|---|---|
| | | $k = 20$ | $k = 50$ | $k = 100$ |
| Avg. Cost | UCB Selection | 216.58▲ | 212.33◆ | 208.06■ |
| | Replay Selection (ours) | **206.63**▲ | **199.38**◆ | **195.35**■ |
| Cumul. Regret | UCB Selection | 646.6△ | 1470.5◇ | 2601.5□ |
| | Replay Selection (ours) | **572.1**△ | **1042.3**◇ | **1544.7**□ |

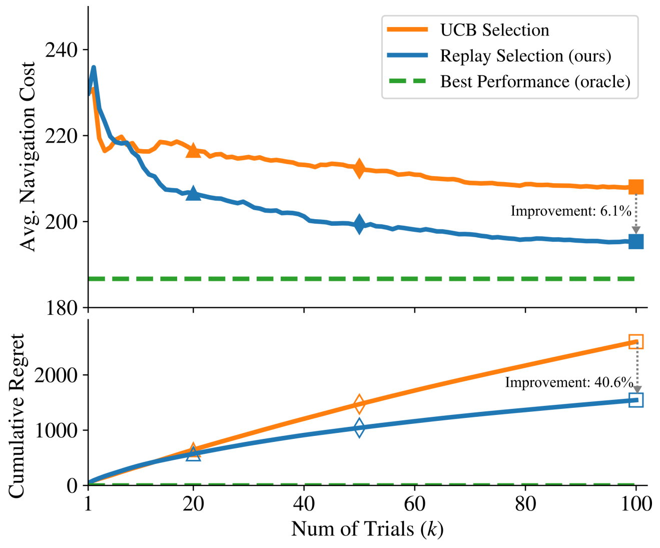

Fig. 3. **Prompt-LLM Selection Results**: Leveraging offline replay for prompt/model selection allows faster selection of the best prompting strategy compared to the UCB selection strategy, resulting in lower average cost and lower cumulative regret.

a reduction of 6.1% in average cost at the end of 100th trial compared to a standard UCB-bandit selection approach. In particular, we achieve 40.6% lower cumulative regret at the end of 100th trial compared to UCB-bandit selection, a number which would continue to grow with more trials.

Our results highlight the need and benefits of fast deployment-time selection of prompts and LLMs, since without such selection, the robot risks poor performance if it uses only one prompt or LLM preselected before deployment. Our selection approach enables the robot to quickly pick *during deployment* the prompts and LLMs that yields better behavior and hence maximizes long-term performance—a benefit afforded by model-based planning with high-level action abstraction amenable to introspection via *offline replay* of Paudel and Stein [1].

## VII. Conclusion

We present an approach for deployment-time selection of best-performing prompts and LLMs for LLM-informed object search in partially-known environments. Leveraging LLM-informed model-based planning [2], we demonstrate that the *offline replay* approach developed for model selection for learning-informed point-goal navigation [1] can be made to support fast deployment-time prompt and LLM selection, a capability unique in this domain. In future, we hope to explore automated prompt generation and refinement strategies that integrate with selection to allow our agent to adapt to a wide variety of environments during deployment.

## Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 2232733.

## References

[1] A. Paudel and G. J. Stein, "Data-efficient policy selection for navigation in partial maps via subgoal-based abstraction," in *International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[2] S. Hossain, A. Paudel, and G. J. Stein, "Enhancing object search by augmenting planning with predictions from large language models," in *2nd CoRL Workshop on Learning Effective Abstractions for Planning*, 2024.

[3] V. S. Dorbala, J. F. Mullen Jr, and D. Manocha, "Can an embodied agent find your "cat-shaped mug"? LLM-guided exploration for zero-shot object navigation," *arXiv preprint arXiv:2303.03480*, 2023.

[4] K. Zhou, K. Zheng, C. Pryor, Y. Shen, H. Jin, L. Getoor, and X. E. Wang, "ESC: Exploration with soft commonsense constraints for zero-shot object navigation," in *International Conference on Machine Learning*, 2023.

[5] B. Yu, H. Kasaei, and M. Cao, "L3MVN: Leveraging large language models for visual target navigation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[6] P. Arjun, A. Melnik, and G. C. Nandi, "Cognitive planning for object goal navigation using generative ai models," in *NeurIPS 2024 Workshop on Open-World Agents*, 2024.

[7] W. Ge, C. Tang, and H. Zhang, "Commonsense scene graph-based target localization for object search," *arXiv preprint arXiv:2404.00343*, 2024.

[8] A. Rajvanshi, K. Sikka, X. Lin, B. Lee, H.-P. Chiu, and A. Velasquez, "Saynav: Grounding large language models for dynamic planning to navigation in new environments," in *Proceedings of the International Conference on Automated Planning and Scheduling*, 2024.

[9] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, "A systematic survey of prompt engineering in large language models: Techniques and applications," *arXiv preprint arXiv:2402.07927*, 2024.

[10] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, 2023.

[11] S. Yang, J. Kim, J. Jang, S. Ye, H. Lee, and M. Seo, "Improving probability-based prompt selection through unified evaluation and analysis," *arXiv preprint arXiv:2305.14877*, 2023.

[12] C. Liao, Y. Zheng, and Z. Yang, "Zero-label prompt selection," *arXiv preprint arXiv:2211.04668*, 2022.

[13] T. Sorensen, J. Robinson, C. M. Rytting, A. G. Shaw, K. J. Rogers, A. P. Delorey, M. Khalil, N. Fulda, and D. Wingate, "An information-theoretic approach to prompt engineering without ground truth labels," *arXiv preprint arXiv:2203.11364*, 2022.

[14] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[15] T. L. Lai, H. Robbins, *et al.*, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, 1985.

[16] A. Paudel, X. Xiao, and G. J. Stein, "Multi-strategy deployment-time learning and adaptation for navigation under uncertainty," in *Conference on Robot Learning (CoRL)*, 2024.

[17] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi, "ProcTHOR: Large-scale embodied AI using procedural generation," *Advances in Neural Information Processing Systems*, 2022.

## Appendix

### A. ProcTHOR *Environments*

Sample household maps from ProcTHOR simulation environments are shown in Fig. 4 for reference.

### B. *Prompt Design*

As mentioned in Section V, we design three prompt templates for LLM+MODEL policy: P-CONTEXT-A, P-CONTEXT-B and P-MINIMAL, and one prompt template for LLM-DIRECT policy: P-DIRECT. We discuss below the design of these prompts.

Fig. 4. Sample maps from ProcTHOR simulation environments

**P-CONTEXT-A, P-CONTEXT-B:** These prompts are designed around four main elements: (i) a description of the setting and the role that the LLM will serve, (ii) a description of the house including a list of the rooms present and the containers they contain, (iii) an example for reference, and (iv) the query asking for the probability of finding the object of interest in a container within a particular room. While the semantic meaning of these prompts are similar (see the Appendix), each of these differ in terms of the language is used in the prompt text.

**P-MINIMAL:** This prompt doesn't include any of the aforementioned contexts about the LLM's role, environment description and reference example, and only includes the query asking for the probability of finding the target object in a container within a particular room.

**P-DIRECT:** This prompt for LLM-DIRECT policy is designed around five main elements: (i) a description of the setting and the role that the LLM will serve (ii) an example interaction for reference (iii) a description of the house including a list of the rooms present and the distances between them (iv) list of available containers that the robot can explore, and (v) the query asking which container the robot should explore to find the target object quickly. It should be noted that we include the distances in the prompt because we expect the LLM to behave like a planner and so provide all necessary information needed to plan effectively.

### C. *Samples of Prompts*

We include below the samples of all prompts used in our experiments.

**P-CONTEXT-A**:
You are serving as part of a system in which a robot needs to find objects located around a household. Here is a schema that describes the connectivity of rooms in the house: The apartment contains the following rooms: bathroom, bedroom. The bedroom contains: bed, chair, sidetable. The bathroom contains: dresser, sidetable, sink, toilet. You will be asked

to estimate the probability (a value between 1% and 100%) of where objects are located in that house, leveraging your considerable experience in how human occupied spaces are located. You must produce a numerical value and nothing else, as it is important to the overall functioning of the system. Here is an example exchange for an arbitrary house:
User: What is the likelihood that I find eggs in the refrigerator in the kitchen?
You: 90%.
The logic here is that there is a high likelihood that a typical refrigerator in the kitchen contains eggs, but it is not guaranteed as not all refrigerators have eggs. Here is your prompt for today:
What is the likelihood that I find book in the sidetable in the bedroom?
**Output:** 95%

**P-CONTEXT-B**:
You are assisting in a robotic system designed to locate items within a residence. The following is a description of the layout and connectivity between rooms in the home: The apartment contains the following rooms: bathroom, bedroom. The bedroom contains: bed, chair, sidetable. The bathroom contains: dresser, sidetable, sink, toilet. Your task is to estimate the likelihood (a percentage from 1% to 100%) that a specified object is in a given location. Base your reasoning on general patterns of human behavior and usage of household spaces. Your response must be a single numerical value, with no additional explanation, as precision is critical to system operation. Example exchange:
User: What is the probability of finding bread in the pantry in the kitchen?
You: 85%.
The reasoning here is that bread is commonly stored in pantries, but exceptions exist, such as if it is refrigerated. Now, respond to this prompt:
What is the probability of finding pillow in the bed in the bedroom?
**Output:** 95%

**P-MINIMAL**:
What is the probability of finding plate in the dining table in the kitchen of a typical household? Your response should only include a numerical percentage value between 1% to 100% and nothing else.
**Output:** 80%

**P-DIRECT**:
You are assisting a robot in locating objects within a household based on a provided map of rooms and their contents. Your task is to determine the exact location where the specified object can be found, based on given description of the household. You will be asked pick a location to visit where the object could be found quickly. You should only pick one location from the given list. Here is an example:
User: The apartment contains: bathroom, bedroom, kitchen. The distance between rooms is as follows: bathroom and bedroom: 5.95 metres, bedroom and kitchen: 3.25 metres, bathroom and kitchen: 4.75 metres. The robot is currently located at bathroom and is looking for pillow. Available locations to search are: sink in bathroom, toilet in bathroom, bed in bedroom, sidetable in bedroom. Which of the given search locations should the robot visit to find pillow in the least time?
You: bed in bedroom.
Now give your answer for another household with the following layout: The apartment contains the following rooms: bathroom, bedroom. The distance between rooms is as follows: bedroom and bathroom: 5.8 meters. The robot is currently located at bedroom and is looking for faucet. Available locations to search are: dresser in bathroom, sidetable in bathroom, sink in bathroom, toilet in bathroom, bed in bedroom, chair in bedroom, sidetable in bedroom. Which of the given search locations should the robot visit to quickly find faucet? Respond with a search location and nothing else.
**Output:** sink in bathroom

TABLE I
AVERAGE NAVIGATION COSTS WITHOUT SELECTION

| Policy / Prompt / LLM | Avg. Cost |
|---|---|
| LLM+MODEL / P-CONTEXT-A / GPT-4o | 227.66 |
| LLM+MODEL / P-CONTEXT-B / GPT-4o | 192.25 |
| LLM+MODEL / P-MINIMAL / GPT-4o | 205.55 |
| LLM-DIRECT / P-DIRECT / GPT-4o | 250.42 |
| LLM+MODEL / P-CONTEXT-A / Gemini | **186.69** |
| LLM+MODEL / P-CONTEXT-B / Gemini | 188.11 |
| LLM+MODEL / P-MINIMAL / Gemini | 225.49 |
| LLM-DIRECT / P-DIRECT / Gemini | 201.50 |
| OPTIMISTIC+GREEDY / – / – | 298.19 |

### D. Average Navigation Costs for Each Strategy

As mentioned in Section VI, prompt selection seeks to choose between all combinations of policies, prompts and LLMs available. These combinations and their individual object search performances over 150 trials in distinct ProcTHOR maps are shown in Table I. Bold value for LLM+MODEL / P-CONTEXT-A / Gemini indicates the best-performing strategy.