ActiveVOO: Value of Observation Guided Active Knowledge Acquisition for Open-World Embodied Lifted Regression Planning

Xiaotian Liu[1], Ali Pesaranghader[2], Jaehong Kim[3], Tanmana Sadhu[2], Hyejeong Jeon[3], Scott Sanner[1,4]

[1]University of Toronto, Toronto, Canada [2]LG Electronics, Toronto AI Lab, Toronto, Canada
[3]LG Electronics, Seoul, South Korea [4]Vector Institute for Artificial Intelligence, Toronto, Canada
xiaotian.liu@mail.utoronto.ca ssanner@mie.utoronto.ca
{ali.pesaranghader, jaehong02.kim, tanmana.sadhu, hyejeong.jeon}@lge.com

Abstract

The ability to actively acquire information is essential for open-world planning under partial observability and incomplete knowledge. However, most existing embodied AI systems either assume a known object category or rely on passive perception strategies that exhaustively gather object and relational information from the environment. Such a strategy becomes insufficient in visually complex open-world settings. For instance, a typical household may contain thousands of novel and uniquely configured objects, most of which are irrelevant to the agent's current task. Consequently, open-world agents must be capable of actively identifying and prioritizing task-relevant objects to enable efficient and goal-directed knowledge acquisition. In this work, we introduce ACTIVEVOO, a novel zero-shot framework for open-world embodied planning that emphasizes object-centric active knowledge acquisition. ACTIVEVOO employs lifted regression to generate compact, first-order subgoal descriptions that identify task-relevant objects, and provides a principled mechanism to quantify the utility of sensing actions based on commonsense priors derived from LLMs and VLMs. We evaluate ACTIVEVOO on the visual ALFWorld benchmark, where it achieves substantial improvements over existing LLM- and VLM-based planning approaches, notably outperforming VLMs fine-tuned on ALFWorld data. This work establishes a principled foundation for developing embodied agents capable of actively and efficiently acquiring knowledge to plan and act in open-world environments.

1 Introduction

Open-world embodied planning is challenging due to partial observability and incomplete task-relevant knowledge. In the open world, knowledge acquisition becomes essential, as agents must identify and obtain relevant information necessary to generate feasible plans. Existing embodied AI systems often assume known object categories and rely on passive strategies that exhaustively collect objects and relational information without contextual relevance [Camoriano et al., 2017]. These approaches quickly become infeasible in complex, information-rich environments due to a combinatorial explosion of objects and their relations to each other.

Active knowledge acquisition offers a better alternative by enabling the agent to proactively seek information that is relevant to the task at hand. However, existing work in this area has largely focused on locating or recognizing predefined object classes [Zhu et al., 2017, Yang et al., 2018] or information gain [Zurbrügg et al., 2022], thereby overlooking a crucial preceding reasoning step: identifying which objects are actually relevant to the goal. To illustrate these challenges, Figure 1 shows a comparative example of active vs. passive knowledge acquisition.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

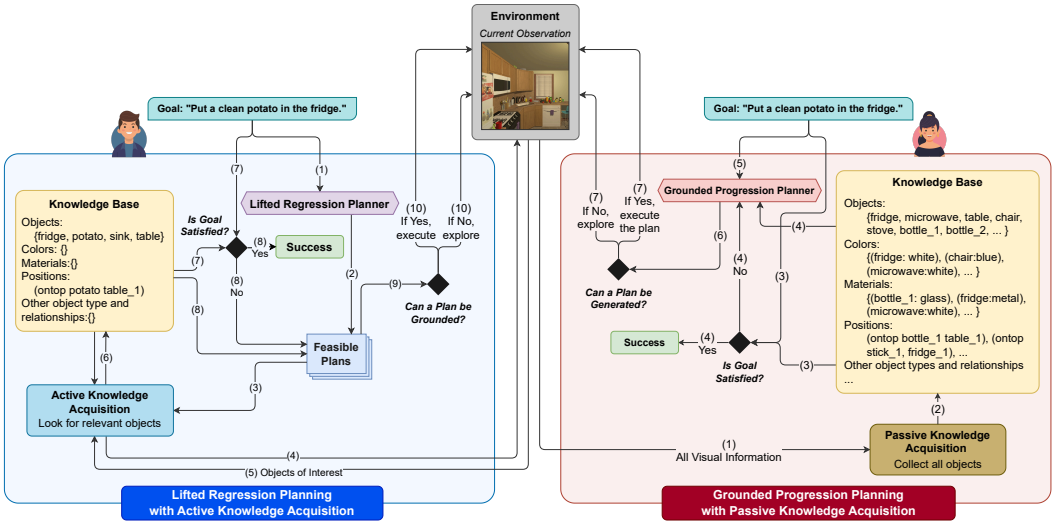

Figure 1: **Active vs. Passive Knowledge Acquisition.** The left panel illustrates the **active approach**, where lifted regression generates abstract subgoals that guide targeted VLM queries for object discovery. The numbered arrows indicate the sequential order of reasoning and sensing steps. In the active flow: 1) lifted subgoals are regressed from the goal, 2) candidate objects and relations are extracted for subgoals, 3) only relevant objects are sensed and grounded to update the agent's knowledge base, and 4) the agent explores the environment based on its sensing target. This closed loop of reasoning and sensing continues until a plan can be executed. The right panel depicts the **passive approach**, where a grounded progression planner relies on exhaustive object collection. Here, 1) the agent first gathers all visible objects, 2) attempts plan grounding, 3) if grounding fails due to missing objects, it re-enters exploration to collect additional data, and 4) Recovery occurs through further exploration (potentially guided by the failed plan context), but this process remains inefficient in open-world settings containing unbounded numbers of objects and relations.

In this work, we introduce **ACTIVEVOO**, an open-world planning framework with active knowledge acquisition. ACTIVEVOO adopts an object-centric view of knowledge, and decomposes active knowledge acquisition into two sub-tasks: 1) The agent determines which types of objects and relational properties are relevant to the goal. For example, it should recognize that a kettle is relevant when preparing a hot drink, while a TV remote is not. 2) The agent quantifies and prioritizes among the relevant objects to sense. For instance, when preparing breakfast, the agent must decide whether locating eggs is more important than finding a baking tray.

Object relevance can be determined by evaluating whether an object is necessary for executing a feasible plan. However, generating plans in open-world settings is feasible with typical grounded progression [Helmert, 2006] methods, which typically assumes the Closed World Assumption (CWA) and a fully specified initial state. An alternative is regression-based planning, where the agent searches backward from the goal to identify preceding subgoals without requiring explicit knowledge of the initial state [Reiter, 2001, Sanner and Boutilier, 2009]. Moreover, regression can be conducted at the lifted level, allowing compact relational reasoning that abstracts away from specific object instances.

ACTIVEVOO leverages lifted regression [Sanner and Boutilier, 2009, Ghallab et al., 2004] to obtain a set of relevant object descriptions, which it then quantifies and prioritizes based on how "helpful" each is to achieving the overall objective. For example, when tasked with "prepare a steak dinner," the agent must weigh the utility of searching for a steak versus locating a wine glass. To this end, we propose *Value of Observation (VOO)* to formalize active knowledge acquisition as a decision-making process that selects utility-maximizing objects to sense. VOO measures the expected improvement in utility obtained by observing a specific object, relative to the utility of acting without attempting to sense the object.

We evaluate ACTIVEVOO on the ALFWorld benchmark [Shridhar et al., 2020], which provides a diverse set of tasks in both visual and textual modalities. While prior work has primarily focused on the textual setting, the visual version of ALFWorld remains highly challenging due to partial observability, object diversity, and complex interaction dynamics. We demonstrate that ACTIVEVOO achieves significant improvements against the visual ALFWorld tasks, notably outperforming vision-

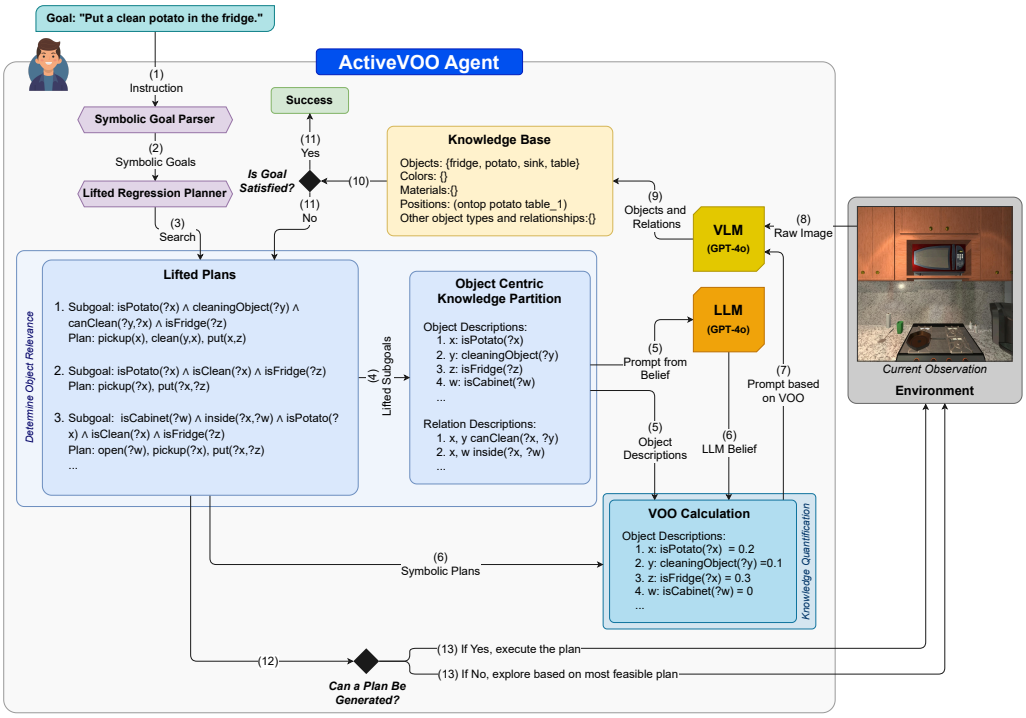

Figure 2: **Overview of the ACTIVEVOO Framework.** Figure 2 illustrates the full workflow of our approach, which integrates lifted regression, object–relation extraction, and Value of Observation (VOO)-guided active sensing. 1) Lifted regression decomposes the goal into a set of lifted subgoals that describe feasible abstract world configurations. 2) From each subgoal, object and relational descriptions are extracted. 3) The agent computes the VOO for each candidate object or relation based on the current KB. 4) Objects with high VOO are selected to guide exploration and sensing. This active cycle continues until all predicates in at least one lifted subgoal are satisfied, at which point the associated plan is instantiated and executed.

language models that are fine-tuned specifically for this benchmark. Figure 2 shows the overall workflow of ACTIVEVOO. To summarize, we make the following key contributions in this work:

- We enable active knowledge acquisition by extracting relevant high-level object and relational descriptions through lifted regression for open-world planning.
- We propose a decision-theoretic Value of Observation (VOO) approach that quantifies the utility of knowledge acquisition using commonsense beliefs from LLMs and VLMs.
- We achieve state-of-the-art performance on the visual ALFWorld benchmark in a zero-shot manner, outperforming even fine-tuned VLM baselines.

2 Methodology

A key challenge for open-world embodied agents is determining which *objects* are relevant to the task prior to planning, as novel and previously unknown objects may appear in the environment. Furthermore, exhaustively enumerating all relational information the agent observes is inefficient, since most objects the agent encounters are irrelevant to the task. To address this challenge, we propose a framework that identifies and prioritizes task-relevant objects through the following steps: 1) The agent applies *lifted regression* to derive subgoals representing possible world configurations where feasible plans can be executed; 2) it then extracts high-level (unary predicate) object descriptions from each subgoal; 3) for each identified object description, the agent estimates the probability of successful sensing by leveraging LLM-based commonsense; 4) the agent computes the *Value of Observation* (VOO) to prioritize object sensing based on expected utility gain; and 5) guided by VOO,

the agent actively senses and collects object information, updating its knowledge base and executing the plan once all relations in a subgoal can be satisfied.

An overview of ACTIVEVOO is shown in Figure 2. We assume the agent is equipped with high-level actions defined in PDDL [Haslum et al., 2019] syntax, an extension of STRIPS [Fikes and Nilsson, 1971], but lacks complete knowledge of object types and their relational states. The full action model for our experimental domain is provided in Appendix A; the examples below are modified for purposes of compact exposition. The agent receives a high-level natural-language instruction without step-by-step guidance and employs goal-conversion methods from [Song et al., 2023, Xie et al., 2023] to translate natural-language goals into PDDL form through few-shot prompting.

2.1 Lifted Plan and Subgoal Derivation

One approach for generating plans in the *open world* (i.e., an open domain of potential objects) is *lifted regression* [Sanner and Boutilier, 2009], which allows the agent to derive high-level plans and their necessary execution conditions without relying on concrete object instances. This is achieved by reasoning in a lifted representation and regressing backward from the goal to infer the abstract preconditions required for plan feasibility. By planning with lifted representations parameterized by variables, the agent reasons in high-level relational descriptions of objects rather than specific instances. For example, instead of concrete object instances such as "`oven`" or "`sink`", lifting enables the agent to reason about object properties and relations like `isClean(?x)` and `canClean(?y, ?x)`, deferring the instantiation of these variables until the relevant objects are observed. Lifted regression plans backward by successively regressing the goal through the action model to determine the abstract preconditions (i.e., subgoals) that must hold for executing a particular action to achieve that goal.

Example (1) Suppose the agent's goal is to *put a clean potato in the fridge*, represented symbolically as

$$G := \texttt{isPotato(?x)} \wedge \texttt{isClean(?x)} \wedge \texttt{isFridge(?z)} \wedge \texttt{inside(?x, ?z)}.$$

Consider the action model

$$\texttt{PutIn(?x, ?z)} := \begin{cases} \textbf{Preconditions:} \texttt{holds(?x)} \\ \textbf{Effects:} \texttt{inside(?x, ?z)} \end{cases}$$

Regressing the goal G through `PutIn(?x, ?z)` replaces the achieved effect `inside(?x ?z)` with the action's preconditions, yielding the regressed subgoal

$$g_1 := \texttt{isPotato(?x)} \wedge \texttt{isClean(?x)} \wedge \texttt{isFridge(?z)} \wedge \texttt{holds(?x)}.$$

This expression states that the agent has to be holding a potato `?x` that has already been cleaned and has found a fridge `?z` to complete the overall goal.

Lifted Backward Search. Lifted regression enables the agent to reason backward from the goal to produce a set of high-level conditional plans capable of achieving the agent's objective under different possible environment configurations. In this work, we apply regression via *Lifted Backward Search* [Ghallab et al., 2004], adapted to open-world settings [Liu et al., 2025], to generate a set of conditional plans $\Pi = \{(g_1, \pi_1), \ldots, (g_n, \pi_n)\}$, where each lifted subgoal g_i specifies a set of *minimal preconditions* that must hold for the corresponding plan π_i to be executable. The details of the open-world lifted regression algorithm are provided in Appendix B.

Example (2) Following the same setup as the previous example, suppose lifted regression yields two abstract subgoals that represent feasible world configurations.

$g_1 := \texttt{isPotato(?x)} \wedge \texttt{isClean(?x)} \wedge \texttt{isFridge(?z)} \wedge \texttt{holds(?x)}$,

$g_2 := \texttt{isPotato(?w)} \wedge \texttt{cleaningObject(?y)} \wedge \texttt{canClean(?y, ?w)} \wedge \texttt{isFridge(?z)}$.

Each subgoal g_i is associated with a lifted conditional plan π_i, which can be executed when the corresponding g_i is satisfied:

$\pi_1 : [\texttt{PutIn(?x, ?z)}]$,

$\pi_2 : [\texttt{Pickup(?w)}, \texttt{Clean(?y, ?w)}, \texttt{PutIn(?w, ?z)}]$

2.2 Object and Relation Description Extraction

Extracting Object Descriptions. To identify which objects are relevant for executing a plan, we first extract their symbolic descriptions from the lifted subgoals. We assume that an object's type and characteristics are captured by the unary predicates present in a subgoal formula g. For every variable $v \in \mathrm{Vars}(g)$, we construct a conjunctive formula $o_{v,g}(v)$ that represents the properties associated with that object, and collectively define the set of all such object formulas as $\mathcal{O}$,

$$o_{v,g}(v) \equiv \bigwedge_{p(v)\in g,\ \mathrm{arity}(p)=1} p(v), \qquad \mathcal{O} = \{\, o_{v,g}(v) \mid g \in \mathcal{G},\ v \in \mathrm{Vars}(g) \,\}, \tag{1}$$

where duplicates are removed via α-renaming of variable symbols. This compact formulation encodes all unary predicates applied to each variable within g, yielding the object-level descriptions used as sensing targets during active knowledge acquisition.

Example (3) Continuing from the previous example, we extract unary predicates for each variable to form their corresponding object descriptions using Eq. 1. For g_1, we obtain:

$$o_{x,g_1}(?x) \equiv \texttt{isPotato(?x)} \wedge \texttt{isClean(?x)} \wedge \texttt{holds(?x)}, \quad o_{z,g_1}(?z) \equiv \texttt{isFridge(?z)}.$$

Similarly, for g_2, the extracted object descriptions are:

$$o_{w,g_2}(?w) \equiv \texttt{isPotato(?w)}, \quad o_{y,g_2}(?y) \equiv \texttt{cleaningObj(?y)}, \quad o_{z,g_2}(?z) \equiv \texttt{isFridge(?z)}.$$

After removing duplicate objects, we obtain the set of four object descriptions used in subgoals g_1 and g_2:

$$\mathcal{O} = \begin{array}{l}\{\texttt{isPotato(?w)},\ \texttt{isPotato(?x)} \wedge \texttt{isClean(?x)} \wedge \texttt{holds(?x)},\\ \ \texttt{cleaningObj(?y)},\ \texttt{isFridge(?z)}\}.\end{array}$$

Extracting Relational Descriptions. In addition to unary object properties, each subgoal formula g may contain relational predicates that describe dependencies among multiple objects—for instance, for variables $\mathbf{v} = (\texttt{?y}, \texttt{?w})$, one may ask whether the relation `canClean(?y, ?w)` holds. More precisely, for every relational predicate $p(\mathbf{v})$ with $\mathbf{v} = (v_1, \ldots, v_k)$ and $k \geq 2$ appearing in g, we construct a *relational formula* that links the corresponding object formulas, and collectively define the set of all such relational formulas as $\mathcal{R}$,

$$\rho_{\mathbf{v},g}(\mathbf{v}) \equiv \left(\bigwedge_{i=1}^{k} o_{v_i,g}(v_i) \wedge p(\mathbf{v}) \right), \qquad \mathcal{R} = \{\rho_{\mathbf{v},g}(\mathbf{v}) \mid g \in \mathcal{G},\ p(\mathbf{v}) \in g,\ |\mathbf{v}| \geq 2\}. \tag{2}$$

where each ρ is an open first-order formula whose free variables correspond to the participating objects; it evaluates to true under an assignment $\mathbf{v}$ iff each v_i instantiates an object satisfying its unary description and the relational predicate p holds among them. Together, $\mathcal{O}$ and $\mathcal{R}$ form a lifted representation that links individual object descriptions to their relational structure for active knowledge acquisition.

Example (4) Continuing from the previous example, the subgoal g_2 contains one relational predicate, `canClean(?y, ?w)`. Using Eq. 2, we construct the corresponding relational formula:

$$\rho_{(y,w),\,g_2}(?y, ?w) \equiv \big(\texttt{cleaningObj(?y)} \wedge \texttt{isPotato(?w)} \wedge \texttt{canClean(?y, ?w)}\big).$$

This formula specifies that the relation `canClean(?y, ?w)` needs to hold for object `isPotato(?w)` and `cleaningObject(?y)`. We can then define the (singleton) set of relation descriptions for subgoal g_2:

$$\mathcal{R} = \{\texttt{cleaningObject(?y)} \wedge \texttt{isPotato(?w)} \wedge \texttt{canClean(?y, ?w)}\},$$

2.3 Sensing Belief Estimation via LLM Commonsense Probability Elicitation

Building on prior work showing that LLM token logits can approximate commonsense belief probabilities [Liu et al., 2023a, Chen et al., 2023], we estimate how likely it is for the agent to sense particular objects. For an object description $q_1(x) \wedge \cdots \wedge q_m(x)$, we query the LLM with statement $\mathbf{s}$: "Does there exist an object that is $q_1, \ldots, q_m$?" For a relational description ρ over $\mathbf{v} = (v_1, \ldots, v_k)$ with predicate p, we query $\mathbf{s}$: "Can $p(\mathbf{v})$ hold given v such that $q_1(v_1), \ldots, q_k(v_k)$?" Then, given a

fixed context $\mathbf{c}$ (e.g., "You are reasoning about a typical household kitchen.") and a statement $\mathbf{s}$, we can obtain model-specific logits of "Yes" and "No" for $\mathbf{s}$ given that context $\mathbf{c}$ holds, denoted as

$$z_{\mathsf{Yes}}(\mathbf{c},\mathbf{s}) := \mathrm{logit}_{\mathrm{LLM}}[\mathbf{c},\mathbf{s} \to \text{“Yes”}], \qquad z_{\mathsf{No}}(\mathbf{c},\mathbf{s}) := \mathrm{logit}_{\mathrm{LLM}}[\mathbf{c},\mathbf{s} \to \text{“No”}], \tag{3}$$

and apply the definition of conditional probability to estimate whether $\mathbf{s}$ is true when context $\mathbf{c}$ is true:

$$P(\mathbf{s}=T|\mathbf{c}=T) = \frac{P(\mathbf{s}=T,\mathbf{c}=T)}{P(\mathbf{s}=F,\mathbf{c}=T)+P(s=T,\mathbf{c}=T)} \approx \frac{\exp(z_{\mathsf{Yes}}(\mathbf{c},\mathbf{s}))}{\exp(z_{\mathsf{Yes}}(\mathbf{c},\mathbf{s}))+\exp(z_{\mathsf{No}}(\mathbf{c},\mathbf{s}))}, \tag{4}$$

Since some logit APIs are stochastic, we average k logits across samples (k=5 in our case). For each $o \in \mathcal{O}$ and $\rho \in \mathcal{R}$, we obtain $p_o := \Pr(\mathbf{s}_o = T|\mathbf{c}=T)$ and $p_\rho := \Pr(\mathbf{s}_\rho = T|\mathbf{c}=T)$ which we use to independently estimate the probability of whether objects satisfying $\mathbf{s}_o$ or relationships satisfying $\mathbf{s}_\rho$ can be sensed in context $\mathbf{c}$. Prompt templates and paraphrasing details are provided in Appendix E.

2.4 Value of Observation (VOO) Computation

After extracting object and relational descriptions from lifted subgoals, the agent must determine *how to prioritize among candidate objects for sensing*. To this end, we propose *Value of Observation (VOO)*, a utility-based measure that quantifies the expected improvement in the agent's ability to achieve its goal with respect to sensing object candidates fitting specific descriptions. By evaluating the VOO for each potential sensing target, the agent can prioritize observations that are most informative, thereby enabling goal-directed and object-centric knowledge acquisition in open-world environments.

Problem Setup. Given the sets of object descriptions $\mathcal{O}$ and relational descriptions $\mathcal{R}$, we define the agent's knowledge states as a set of sensing beliefs over objects and relations:

$$S = \{ \underbrace{(O^+, O^-, R^+, R^-)}_{s} \mid O^\pm \subseteq \mathcal{O},\ R^\pm \subseteq \mathcal{R},\ O^+ \cap O^- = \varnothing,\ R^+ \cap R^- = \varnothing \}. \tag{5}$$

Here, O^+ and R^+ denote the sets of object and relational descriptions that the agent believes can be sensed, whereas O^- and R^- represent those that the agent believes cannot be sensed. Objects that are not included in either of these sets but are present in the overall object set $\mathcal{O}$ are considered *unknown* to the agent. The agent can perform a sensing action a_o for an object description $o \in \mathcal{O}$, which transitions it into one of two possible successor states depending on whether o can be sensed:

$$s'_+ = (O^+ \cup \{o\},\ O^-,\ R^+,\ R^-), \qquad s'_- = (O^+,\ O^- \cup \{o\},\ R^+,\ R^-), \tag{6}$$

The two outcomes occur with respective probabilities p_o and $1 - p_o$ estimated in Section 2.3. We formalize this knowledge acquisition process as a belief-state Markov Decision Process (MDP), whose full specification, including transition dynamics and reward formulation, is provided in Appendix D.

Utility Estimation and VOO Calculation. We define a utility function $U(s)$ reflecting the agent's belief that *any* subgoal $g \in \mathcal{G}$ is satisfiable given the agent's knowledge state s. For each subgoal g, we first identify the sets of *still-unknown* object and relational descriptions associated with g as

$$\mathrm{obj}_{\mathrm{unk}}(g,s) \ = \ \mathrm{obj}(g) \setminus (O^+ \cup O^-), \qquad \mathrm{rel}_{\mathrm{unk}}(g,s) \ = \ \mathrm{rel}(g) \setminus (R^+ \cup R^-). \tag{7}$$

Here, $\mathrm{obj}(g)$ and $\mathrm{rel}(g)$ denote the sets of object and relational descriptions associated with subgoal g, respectively.

Example (5) Continuing from the previous example, suppose the agent's current knowledge state is $s = (O^+, O^-, R^+, R^-)$, where $O^+ = \{\texttt{isPotato(?w)}\}$ and $R^+, O^-, R^- = \varnothing$, indicating that only the potato has been identified. Recalling previous subgoal

$$g_2 := \texttt{isPotato(?w)} \wedge \texttt{cleaningObject(?y)} \wedge \texttt{canClean(?y, ?w)} \wedge \texttt{isFridge(?z)},$$

the unknown object and relational descriptions are respectively:

$$\mathrm{obj}_{\mathrm{unk}}(g_2,s) = \{\texttt{cleaningObject(?y)},\ \texttt{isFridge(?z)}\},$$
$$\mathrm{rel}_{\mathrm{unk}}(g_2,s) = \{\texttt{cleaningObject(?y)} \wedge \texttt{isPotato(?w)} \wedge \texttt{canClean(?y, ?w)}\}.$$

Assuming independent sensing outcomes with success probabilities $\{p_o\}_{o\in\mathcal{O}}$ for object descriptions and $\{p_\rho\}_{\rho\in\mathcal{R}}$ for relational descriptions, we define the belief probability a subgoal g can be satisfied after sensing all currently unknown descriptions as:

$$\Pr\big(g \text{ is satisfiable} \mid s\big) \;=\; \prod_{o\in \mathrm{obj}_{\mathrm{unk}}(g,s)} p_o \;\cdot\; \prod_{\rho\in \mathrm{rel}_{\mathrm{unk}}(g,s)} p_\rho. \tag{8}$$

The utility of a given knowledge state s is then defined as the maximum probability of any subgoal g that can be made satisfiable from s.

$$U(s) \;=\; \max_{g\in\mathcal{G}} \left\{ \prod_{o\in \mathrm{obj}_{\mathrm{unk}}(g,s)} p_o \cdot \prod_{\rho\in \mathrm{rel}_{\mathrm{unk}}(g,s)} p_\rho \right\}. \tag{9}$$

Example (6) Continuing from the previous example, assuming independent sensing outcomes with success probabilities $p_{\texttt{cleaningObject}} = 0.9$, $p_{\texttt{isFridge}} = 0.8$, and $p_{\texttt{canClean}} = 0.7$, the belief that g_2 can be satisfied is

$$\Pr(g_2 \text{ is satisfiable} \mid s) = 0.9 \times 0.8 \times 0.7 = 0.504$$

For $g_1 =$ `isPotato(?x)` $\wedge$ `isClean(?x)` $\wedge$ `isFridge(?z)` $\wedge$ `holds(?x)`, the unknown descriptions are $\mathrm{obj}_{\mathrm{unk}}(g_1, s) = \{$`isFridge(?z)`, `isPotato(?x)` $\wedge$ `isClean(?x)` $\wedge$ `holds(?x)`$\}$, with probabilities $p_{\texttt{isFridge}} = 0.8$ and $p_{\texttt{isPotato} \wedge \texttt{isClean} \wedge \texttt{holds}} = 0.1$ giving

$$\Pr(g_1 \text{ is satisfiable} \mid s) = 0.8 \times 0.1 = 0.08$$

Using Eq. 9, the utility of the current knowledge state s is defined as

$$U(s) \;=\; \max_{g\in\mathcal{G}} \left\{ \prod_{o\in \mathrm{obj}_{\mathrm{unk}}(g,s)} p_o \cdot \prod_{\rho\in \mathrm{rel}_{\mathrm{unk}}(g,s)} p_\rho \right\} \;=\; \max(0.504, 0.08) = 0.504$$

Value of Observation. Given utility $U(s)$, when the agent attempts to sense object description $o \in \mathcal{O}$, the expected utility of the outcomes of sensing vs. not sensing o can be calculated as follows:

$$U_o^{\text{sense}}(s) = p_o\, U(s'_+) + (1 - p_o)\, U(s'_-), \qquad U_o^{\text{not\_sense}}(s) = U(s'_-). \tag{10}$$

Here, s'_+ and s'_- denote the successor knowledge states corresponding to successful and unsuccessful sensing outcomes, respectively. The left equation represents the expected utility from sensing o, while the right shows the baseline utility when the agent does not sense o and cannot make use of it. Then, the *Value of Observation (VOO)* quantifies the expected utility gain from actively sensing o:

$$\mathrm{VOO}_{\mathrm{o}}(s) = U_o^{\text{sense}}(s) - U_o^{\text{not\_sense}}(s) = p_o\big[U(s'_+) - U(s'_-)\big]. \tag{11}$$

VOO thus measures how much the agent's expected progress toward a satisfiable subgoal improves by actively acquiring knowledge about a specific object.

While the astute reader may observe a non-incidental similarity between VOO and Value of Information (VOI) [Howard, 1966], there are causal differences in the computations. In VOI, deciding whether or not to make an observation (weather report) does not causally affect the outcome (whether it will rain). In contrast for VOO, an object must be actively observed to causally affect the outcome.

2.5 Active Knowledge Acquisition and Plan Execution

The ACTIVEVOO agent acquires knowledge from both the physical environment, by navigating to new locations and collecting visual observations, and by issuing language or vision–language queries that infer object relations using LLMs or VLMs. At each step, the agent selects the object description with the highest Value of Observation, $o^\star = \arg\max_{o\in\mathcal{O}} \mathrm{VOO}_{\mathrm{o}}(s)$, and treats it as the next *exploration target*. It then queries an LLM to obtain the optimal location the agent can explore based on the exploration target. After navigating to the corresponding region, the agent captures an RGB frame and queries the VLM to detect instances that satisfy the top-k (in our case, top 5) object descriptions ranked via VOO values. Successful detections are updated into the agent's knowledge state s, which in turn updates the estimated utilities $U(s)$ and all VOO scores. The agent also infers relational predicates (via LLM reasoning or direct observation) whose arguments correspond to the sensed object types. Once every predicate in a lifted subgoal $g \in \mathcal{G}$ is grounded, the agent instantiates

the associated plan π_g and executes the corresponding plan. ACTIVEVOO enables the agent to allocate its limited sensing budget toward information that maximizes expected utility, seamlessly integrating physical exploration with commonsense reasoning over object relations. The complete execution flow of the agent is illustrated in Figure 2, and detailed implementation steps are provided in Appendix F of the supplementary material.

3 Experiments

Experimental Setup. We evaluate ACTIVEVOO on the ALFWorld benchmark using raw visual inputs in a zero-shot setting, without additional data for training or fine-tuning. We compare ACTIVEVOO against state-of-the-art LLM/VLM-based planners using both vision and language observations, including zero-shot methods as well as fine-tuned models on ALFWorld data. The performance is evaluated using Success Rate, defined as the agent's ability to complete the task in 30 steps. We average success rate over 3 runs. We set the regression length to 10 and the LLM temperature to 0.1, except when computing belief probabilities, where the temperature is set to 0. All experiments are carried out on a `6-Core AMD CPU` with `32GB of RAM`, and we rely on API calls for LLMs/VLMs. For ACTIVEVOO, we use `GPT-4o` both for belief probability estimation and as a vision-language model to extract relevant objects. We choose ALFWorld [Shridhar et al., 2020] as our benchmark to evaluate both text and vision observations modalities.

Baselines. We compare ACTIVEVOO against a diverse set of baselines across three categories: **vision only models**, **language only models**, and **vision language models**. We include ResNet-18 [Shridhar et al., 2020] and MCNN-FPN [Shridhar et al., 2020], which are standard computer vision architectures that take an image as input and output an action. For language-based models, we focus on prompting-based methods, REACT [Yao et al., 2022], REFLEXION [Shinn et al., 2024], DEPS [Wang et al., 2023], and AUTOGEN [Wu et al., 2023], which rely solely on LLMs for reasoning and planning with few-shot examples. Vision language agents take both raw pixel observations and natural language instructions as input, requiring multimodal reasoning. We include models that require data collection and fine-tuning, which are BUTLER [Shridhar et al., 2020], MINIGPT-4 [Zhu et al., 2023a], BLIP-2 [Li et al., 2023], LLAMA-ADAPTOR [Gao et al., 2023], INSTRUCTBLIP [Panagopoulou et al., 2023], and EMMA [Yang et al., 2024], as well as advanced VLMs, GPT4O [Hurst et al., 2024a] and LLAVA-13B [Liu et al., 2023b], in the zero-shot setting. Note that EMMA and REFLEXION require successive attempts on the same task. Thus, we report their performance for both 3 and 10 trials.

4 Results and Discussion

Table 1 presents ACTIVEVOO's performance against 6 ALFWorld tasks, compares it with 13 baselines representing vision-only, language-only, and vision-language models. ACTIVEVOO achieves an overall success rate of 0.86, the highest among all agents with only visual observations. Our approach significantly outperforms SOTA vision-language models such as GPT-4O and LLAVA-13B in zero-shot settings, and also surpasses models that require extensive task-specific fine-tuning and data collection, including BUTLER, LLAMA-ADAPTER, INSTRUCTBLIP, and EMMA. While EMMA's performance is comparable, it achieves this level after 10 successive trials in the same environment, leveraging cross-trial information transfer. In contrast, ACTIVEVOO is strictly evaluated in a zero-shot setting, based on a single trial, without relying on any prior experience or memory from earlier episodes.

4.1 RQ 1: Overall Performance Comparison

Even when compared to language-based methods, which operate on structured textual inputs devoid of visual noise, ACTIVEVOO outperforms all existing LLM-based planners, except for REFLEXION-10, which relies on 10 trials that reflect on past failures in the same environment, while ACTIVEVOO significantly outperforms REFLEXION-3 with 3 trials. Additionally, most existing LLM-based planners require few-shot examples of the same task, which may not be available when the task is new. More impressively, ACTIVEVOO outperforms all approaches that rely on large-scale supervised training or environment-specific fine-tuning, which are often costly and impractical.

Table 1: Success rates on the ALFWorld tasks using template task instructions. "Env." indicates whether the agent uses a visual or textual environment. "Multi-Trail" indicates whether the agent requires multiple trials on the same environment. "Fine-Tuning" states whether the model is fine-tuned with data collected from ALFWorld. "*" indicates that reported results are used, and "–" indicates results are not available.

Agent	Env.	Multi-Trial	Fine-Tuning	Avg.	Pick	Clean	Heat	Cool	Look	Pick2
Human Performance* [Shridhar et al., 2020]	Visual	✗	✗	0.91	–	–	–	–	–	–
Vision Only Models										
MCNN-FPN* [Shridhar et al., 2020]	Visual	✗	✓	0.05	–	–	–	–	–	–
ResNet-18* [Shridhar et al., 2020]	Visual	✗	✓	0.06	–	–	–	–	–	–
Language Only Models										
GPT-4o (Text) [Hurst et al., 2024b]	Textual	✗	✗	0.21	0.29	0.17	0.21	0.23	0.27	0.13
ReAct [Yao et al., 2022]	Textual	✗	✗	0.69	0.78	0.75	0.72	0.58	0.77	0.56
Reflexion-3 [Shinn et al., 2024]	Textual	✓(3)	✗	0.83	0.89	0.80	0.78	0.85	0.86	0.81
Reflexion-10 [Shinn et al., 2024]	Textual	✓(10)	✗	**0.91**	**0.96**	**1.00**	**0.81**	0.83	0.94	0.88
DEPS* [Wang et al., 2023]	Textual	✗	✗	0.76	0.93	0.50	0.80	**1.00**	**1.00**	0.00
AutoGen* [Wu et al., 2023]	Textual	✗	✗	0.77	0.92	0.74	0.78	0.86	0.83	0.41
Vision Language Models										
BUTLER* [Shridhar et al., 2020]	Visual	✗	✓	0.26	0.31	0.41	0.60	0.27	0.12	0.29
GPT-4o (Vision) [Hurst et al., 2024b]	Visual	✗	✗	0.08	0.16	0.06	0.00	0.04	0.10	0.12
LLaVA-13B [Liu et al., 2023b]	Visual	✗	✗	0.11	0.13	0.15	0.12	0.07	0.11	0.09
LLaMA-Adapter* [Yang et al., 2024]	Visual	✗	✓	0.13	0.17	0.10	0.27	0.22	0.00	0.00
InstructBLIP* [Yang et al., 2024]	Visual	✗	✓	0.22	0.50	0.26	0.23	0.06	0.17	0.00
EMMA-3* [Yang et al., 2024]	Visual	✓(3)	✓	0.37	0.55	0.41	0.45	0.13	0.65	0.4
EMMA-10* [Yang et al., 2024]	Visual	✓(10)	✓	0.82	0.71	**0.94**	**0.85**	**0.83**	0.88	0.67
(Ours)	Visual	✗	✗	**0.86**	**0.93**	0.87	0.83	0.80	**0.89**	**0.86**

For example, BUTLER and EMMA require 52K and 15K expert demonstrations, respectively. In contrast, ActiveVOO only requires a high-level symbolic action model typically available for agents with predefined skill sets (see Appendix A in the supplementary material). Given the diversity and variability of open-world task instances, models trained on fixed task distributions often struggle to generalize. In contrast, ActiveVOO can reason over any task that can be expressed in PDDL [Haslum et al., 2019] or similar symbolic forms. ActiveVOO's strong performance underscores the value of combining structured symbolic reasoning with LLM-inferred commonsense probability estimates, pointing to a promising direction for hybrid planning in embodied AI systems.

4.2 RQ 2: Impact of Active Knowledge Acquisition

One key research question we investigate is how much our agent's performance can be credited to active knowledge acquisition (which leverages VOO-inferred objects from visual observations) in comparison to more passive strategies. Under the *exhaustive object acquisition* setup, the agent passively extracts information from each scene by prompting the VLM to extract all possible objects and relations from visual input, without contextual filtering. This method achieves a success rate of only 0.22 as shown in Table 2, likely due to the overwhelming number of potential objects and relations present in each image. In the *Goal-Directed Object Acquisition setting*, we instead query the VLM using only the high-level task goal. This leads to an improvement over the fully passive baseline. We also evaluate a strategy where the VLM is queried using a plan generated by an LLM. Both of these weakly guided methods achieve similar success rates (0.47 and 0.48), suggesting that LLM-generated plans alone are insufficient to reliably identify task-relevant objects. Together, these results empirically demonstrate the value of our principled VOO-based approach, which significantly outperforms passive or weakly guided knowledge acquisition strategies. The results are shown in Table 2, and the experimental setup details for these ablations can be found in Appendix G in the supplementary material.

Table 2: Impacts of Active Knowledge Acquisition and ActiveVOO Components (RQ2 and RQ3 Results)

Settings	Success Rate	Episode Length
ActiveVOO	0.86	15.3
w/ Exhaustive Obj. Acq.	0.22	25.4
w/ Goal-Directed Obj. Acq.	0.48	23.0
w/ LLM Subgoals Obj. Acq.	0.47	22.5
w/o VOO Calculation	0.75	19.2
w/o VOO and Object Partition	0.68	21.3
w/ `LLaVA-13B` as VLM	0.82	16.1
w/ `GPT-3.5` as LLM	0.77	19.5

4.2.1 RQ 3: Ablation of ActiveVOO Components

We conduct a detailed ablation study to examine the contributions of various components in the ACTIVEVOO framework as shown in Table 2. We assess the effect of VOO by replacing VOO-selected objects with those chosen solely by highest belief probability. In this setting, if the agent's initial belief is improperly calibrated, it then lacks a mechanism to effectively explore alternatives. This results in a performance drop from 0.86 to 0.75 and an increase in the average episode length by 3.9 steps. We further analyze the performance by removing not only the VOO module but also the mechanism to extract relevant object descriptions. In this case, the agent directly queries the LLM to produce belief probabilities of an entire subgoal. Under this setup, the success rate declines further to 0.68. We observe that when a subgoal becomes sufficiently complex, i.e., containing many predicates, the LLM's logits become a worse approximation of the agent's actual belief. Lastly, we investigate the effect of using different VLM and LLM models. Replacing the `GPT-4o` with `LLaVA-13B` for the VLM component slightly reduces performance to 0.82, and substituting `GPT-4o` with `GPT-3.5` for the LLM component results in a further drop to 0.77, indicating that stronger LLM/VLM reasoning capabilities directly enhance ACTIVEVOO's performance.

5 Related Work

Language and Vision Models for Embodied Planning. Recent work on embodied planning with LLMs focuses on predicting the next action [Ahn et al., 2022, Valmeekam et al., 2023, Hazra et al., 2024] or generating step-by-step plans through prompting [Yao et al., 2022, Shinn et al., 2024, Huang et al., 2022]. Hybrid approaches that integrate classical planners with LLM reasoning have also been proposed [Arora and Kambhampati, 2023, Guan et al., 2023, Hazra et al., 2024], though they typically operate under closed-world assumptions or rely on iterative re-planning. VLM-based methods extend embodied reasoning to visual observations, either via vision-to-language translation [Gao et al., 2024, Huang et al., 2023] or direct visual grounding [Driess et al., 2023, Hurst et al., 2024b]. VLAs based approaches produce low-level action sequences [Kim et al., 2024, Brohan et al., 2022], but generally assume static, fully observable environments. While some open-world VLMs [Yang et al., 2024, Liu et al., 2023b] have been proposed, they rely on costly task-specific fine-tuning and extensive data collection. In contrast, ACTIVEVOO generates lifted plans without assuming full observability or complete knowledge, enabling robust reasoning in open-world settings.

Active Knowledge Acquisition For Embodied Agents. Existing works tackle active knowledge acquisition from different perspectives. A notable area of research is embodied navigation, where the agent is tasked with autonomously discovering an object of a certain type [Yang et al., 2018], navigating to a specified image [Zhu et al., 2017], or following step-by-step instructions [Anderson et al., 2018, Ku et al., 2020]. However, these approaches typically assume that the goal or target object is predefined. In contrast, our work focuses on determining object relevance to the overall goal. Other lines of work explore active object recognition [Bohg et al., 2017, Caselles-Dupré et al., 2021] or general exploration of the environment [Zhu et al., 2023b]. While these methods emphasize interaction and perception, they are not task-driven and do not require assessing the relevance of objects, an essential ability for open-world agents. ActiveVOO explicitly addresses this gap by integrating goal-conditioned object relevance estimation with a principled knowledge acquisition framework base on VOO for open-world embodied planning.

6 Conclusion

We introduce ACTIVEVOO, a novel framework that enables active knowledge acquisition for open-world embodied agents based on the Value of Observation (VOO). Our method integrates lifted regression for plan subgoal generation with commonsense reasoning and probabilistic inference via LLMs, allowing for structured inference without requiring a complete domain specification. We demonstrate that ACTIVEVOO significantly outperforms state-of-the-art LLM and VLM methods without the need for additional data collection, domain-specific fine-tuning, or few-shot examples.

Limitation This work is evaluated on the ALFWorld simulator, which requires further fine-tuning to transfer effectively to real-world settings involving physical robots. Additionally, it does not address user or contextual preferences in goal specification, which could influence the agent's underlying utility function.

Acknowledgments

This work was supported by LG Electronics, Toronto AI Lab Grant Ref No. 2024-0565.

References

Raffaello Camoriano, Giulia Pasquale, Carlo Ciliberto, Lorenzo Natale, Lorenzo Rosasco, and Giorgio Metta. Incremental robot learning of new objects with fixed update time. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3207–3214. IEEE, 2017.

Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *arXiv preprint arXiv:1810.06543*, 2018.

René Zurbrügg, Hermann Blum, Cesar Cadena, Roland Siegwart, and Lukas Schmid. Embodied active domain adaptation for semantic segmentation via informative path planning. *IEEE Robotics and Automation Letters*, 7(4):8691–8698, 2022.

Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26: 191–246, 2006.

Raymond Reiter. *Knowledge in action: logical foundations for specifying and implementing dynamical systems*. MIT press, 2001.

Scott Sanner and Craig Boutilier. Practical solution techniques for first-order mdps. *Artificial Intelligence*, 173(5-6):748–788, 2009.

Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: theory and practice*. Elsevier, 2004.

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

Patrik Haslum, Nir Lipovetzky, Daniele Magazzeni, Christian Muise, Ronald Brachman, Francesca Rossi, and Peter Stone. *An introduction to the planning domain definition language*, volume 13. Springer, 2019.

Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208, 1971.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009, 2023.

Yaqi Xie, Chen Yu, Tongyao Zhu, Jinbin Bai, Ze Gong, and Harold Soh. Translating natural language to planning goals with large-language models. *arXiv preprint arXiv:2302.05128*, 2023.

Xiaotian Liu, Ali Pesaranghader, Hanze Li, Punyaphat Sukcharoenchaikul, Jaehong Kim, Tanmana Sadhu, Hyejeong Jeon, and Scott Sanner. Open-world planning via lifted regression with llm-inferred affordances for embodied agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20881–20897, 2025.

Jiacheng Liu, Wenya Wang, Dianzhuo Wang, Noah A Smith, Yejin Choi, and Hannaneh Hajishirzi. Vera: A general-purpose plausibility estimation model for commonsense statements. *arXiv preprint arXiv:2305.03695*, 2023a.

Wenkai Chen, Sahithya Ravi, and Vered Shwartz. Case: commonsense-augmented score with an expanded answer space. *arXiv preprint arXiv:2311.01684*, 2023.

Ronald A Howard. Information value theory. *IEEE Transactions on systems science and cybernetics*, 2(1):22–26, 1966.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 3(4), 2023.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023a.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023.

Artemis Panagopoulou, Le Xue, Ning Yu, Junnan Li, Dongxu Li, Shafiq Joty, Ran Xu, Silvio Savarese, Caiming Xiong, and Juan Carlos Niebles. X-instructblip: A framework for aligning x-modal instruction-aware representations to llms and emergent cross-modal reasoning. *arXiv preprint arXiv:2311.18799*, 2023.

Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lusong Li, Li Shen, Xiaodong He, Jing Jiang, and Yuhui Shi. Embodied multi-modal agent trained by an llm from a parallel textworld. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26275–26285, 2024.

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

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023b.

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

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005, 2023.

Rishi Hazra, Pedro Zuidberg Dos Martires, and Luc De Raedt. Saycanpay: Heuristic planning with large language models using learnable domain knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20123–20133, 2024.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022.

Daman Arora and Subbarao Kambhampati. Learning and leveraging verifiers to improve planning capabilities of pre-trained language models. *CoRR*, abs/2305.17077, 2023. doi: 10.48550/ARXIV. 2305.17077. URL https://doi.org/10.48550/arXiv.2305.17077.

Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. In *NeurIPS*, 2023.

Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically grounded vision-language models for robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12462–12469. IEEE, 2024.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. Palm-e: An embodied multimodal language model. 2023.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. *arXiv preprint arXiv:2010.07954*, 2020.

Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

Hugo Caselles-Dupré, Michael Garcia-Ortiz, and David Filliat. Scod: Active object detection for embodied agents using sensory commutativity of action sequences. *arXiv preprint arXiv:2107.02069*, 2021.

Hao Zhu, Raghav Kapoor, So Yeon Min, Winson Han, Jiatai Li, Kaiwen Geng, Graham Neubig, Yonatan Bisk, Aniruddha Kembhavi, and Luca Weihs. Excalibur: Encouraging and evaluating embodied exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14931–14942, 2023b.

A Action Model

Below are the high-level skills/actions defined in PDDL syntax. Each action $A \in \mathcal{A}$ is represented as

$$A = \{\text{params}(A),\ \text{pre}(A),\ \text{add}(A),\ \text{del}(A)\},$$

corresponding to the action's *parameters*, *preconditions*, and *effects*. The sets $\text{add}(A)$ and $\text{del}(A)$ indicate which predicates are made true or false. Our regression approach requires no negated preconditions or goals, no conditional effects, and no quantifiers in preconditions or effects.

```
(:action PickupObject
  :parameters (?o - obj)
  :precondition (and
    (handEmpty)
    (isObject ?o))
  :effect (and
    (holds ?o)
    (not (handEmpty))))

(:action PutObjectInReceptacle
  :parameters (?o - obj ?r - obj)
  :precondition (and
    (canContain ?r ?o)
    (isObject ?r)
    (holds ?o))
  :effect (and
    (inReceptacle ?o ?r)
    (handEmpty)
    (not (holds ?o)))))

(:action HeatObject
  :parameters (?r - obj ?o - obj)
  :precondition (and
    (isHeatingObject ?r)
    (canHeat ?r ?o)
    (holds ?o))
  :effect (and
    (isHot ?o)))

(:action CleanObject
  :parameters (?r - obj ?o - obj)
  :precondition (and
    (isCleaningObject ?r)
    (canClean ?r ?o)
    (holds ?o))
  :effect (and
    (isClean ?o)))

(:action CoolObject
  :parameters (?r - obj ?o - obj)
  :precondition (and
    (isCoolingObject ?r)
    (canCool ?r ?o)
    (holds ?o))
  :effect (and
    (isCool ?o)))

(:action ToggleObject
  :parameters (?r - obj)
  :precondition (and
    (isObject ?r))
  :effect (and
    (isOn ?r)))
```

B Open-World Lifted Regression

B.1 Open-World Planning Formalism

We formalize the open-world planning problem as the tuple $\Pi = \langle \mathcal{P}, \mathcal{A}, O, G, I \rangle$, where $\mathcal{P}$ denotes the set of predicate symbols, $\mathcal{A}$ the set of action schemas, O the (incomplete) set of known objects, G the goal formula, and I the agent's initial knowledge. Unlike closed-world planning, which presumes complete knowledge of all objects and the initial state, open-world agents must accommodate missing object instances as well as uninstantiated variables or predicates. We represent an atom $p(T)$ by pairing a predicate $p \in \mathcal{P}$ with an n-tuple of terms $T = \langle t_1, \ldots, t_n \rangle$, where $\mathrm{vars}(T)$ indicates the variables occurring in T. An atom is lifted if $\mathrm{vars}(T) \neq \emptyset$. Each action schema $A \in \mathcal{A}$, $A = \{\mathrm{params}(A), \mathrm{pre}(A), \mathrm{add}(A), \mathrm{del}(A)\}$, in which $\mathrm{params}(A)$ are the action's parameters, $\mathrm{pre}(A)$ its preconditions, $\mathrm{add}(A)$ the predicates added to the resulting subgoal, and $\mathrm{del}(A)$ the predicates removed. The objective is to generate a finite set $S = \{(g_i, \pi_i)\}_{i=1}^{n}$, where each pair (g_i, π_i) consists of a feasible lifted *subgoal* g_i and an associated *action sequence* π_i whose length does not exceed a user-defined horizon τ.

B.2 Open-World Lifted Regression

Once the planning problem is formalized, we leverage *Lifted Backward Search* [Ghallab et al., 2004], adapted to open-world settings, to produce a set $S = \{(g_1, \pi_1), \ldots, (g_n, \pi_n)\}$ of all feasible lifted subgoals g_i and their corresponding plans π_i. Starting from the overall goal G, the algorithm iteratively applies REGRESS to each RELEVANT action until no further regression is possible or a predefined length τ is reached. We say that an action schema $a \in \mathcal{A}$ is RELEVANT to a subgoal s if $s \cap (\mathrm{add}(a) \cup \mathrm{del}(a)) \neq \emptyset$, $s \cap \mathrm{del}(a) = \emptyset$, and $s \cap \mathrm{add}(a) = \emptyset$. Whenever a is relevant, the regressed subgoal is computed as $\textsc{Regress}(s, a) = (s \setminus \mathrm{add}(a)) \cup \mathrm{pre}(a)$. To avoid variable clashes and ensure correctness, each regression step applies the standard logical operations STANDARDIZE, UNIFY, and SUBSTITUTE (see Appendix C). The algorithm maintains a frontier of subgoal-plan pairs (g, π), initialized with $(G, \langle\rangle)$. At each iteration, if $|\pi| = \tau$ the current pair is added to S as a terminal subgoal-plan pair; otherwise, for each action $a \in \mathcal{A}$, we STANDARDIZE and UNIFY a with g, check $\textsc{Relevant}(a, g)$, and enqueue the new pair $\big(\textsc{Regress}(g, a),\ a :: \pi\big)$ if g has not been visited. If no regressible action exists, the pair is deemed terminal and also added to S. The process repeats until the frontier is empty, at which point the algorithm returns all lifted plans and subgoals in S of length at most τ. A complete pseudocode listing appears in Appendix C in the supplementary material.

B.3 Open-World Lifted Regression Algorithm

The Open World Lifted Regression algorithm is initialized with an empty set S, which the algorithm will return, and a frontier that contains the overall goal G paired with an empty plan. The algorithm performs a backward search by repeatedly REGRESS through a subgoal and trace pair (g, π) from the frontier. If the trace length $|\pi|$ equals the maximum depth τ, it adds (g, π) to S. For each action schema $A \in \mathcal{A}$, the algorithm applies STANDARDIZE to generate fresh variables to obtain A' and uses UNIFY to match A' effect with the current subgoal g. If g is determined to be RELEVANT, the algorithm regresses g through A' to produce a new subgoal g' and an extended trace $\pi.append(A')$. The new subgoal g' is added to the frontier for further regression. If no actions apply, the algorithm treats (g, π) as a leaf and adds it to S. Once the frontier is exhausted, S contains all lifted subgoals and their corresponding regression paths up to depth τ.

C Logical Operations

Here is a set of standard logical operations used in this work for Lifted Regression:

- **Substitution**: SUBSTITUTE is an operator that replaces terms in one set of logical statements with terms from another set. We use θ to represent a mapping between variables/ terms in the form of a dictionary:
$$\theta = \{v_1/t_1, v_2/t_2, v_3/t_3, \ldots, v_n/t_n\}.$$
- **Standardization**: The STANDARDIZE operator, written as $\textsc{Standardize}(p)$, replaces all variables in a predicate p with new variables v' such that $v' \notin \mathcal{V}$. This ensures that variables in one logical

Algorithm 1 Open-World Lifted-Regression

$\Pi = \langle \mathcal{P}, \mathcal{A}, O, G, I \rangle$, τ
$S \leftarrow \{\}$
Frontier $\leftarrow \{(G, \pi = [])\}$
Regressed $\leftarrow \{G\}$
while Frontier $\neq \emptyset$ **do**
  $g, \pi \leftarrow$ **POP**(Frontier)
  if len(π) = τ **then**
    S.add((g, π))
  else
    RegressActions $= \{\}$
    for each A in $\mathcal{A}$ **do**
      $A' \leftarrow$ STANDARDIZE(A)
      $\theta \leftarrow$ UNIFY(A', g)
      if RELEVANT($\theta(A'), \theta(g)$) **then**
        RegressActions.add(A')
        π.append(A')
        $g' \leftarrow$ REGRESS($\theta(g), \theta(A')$)
        if g' not in Visited **then**
          Regressed.add(g')
          Frontier.add((g', π))
        end if
      end if
    end for
    if RegressActions $= \emptyset$ **and** $g' \notin$ Visited **then**
      S.add((g, π))
    end if
  end if
end while
return S

statement do not conflict with those in the target statement it tries to unify. Standardization is carried out by applying substitution $\theta(p)$, which replaces each variable as follows:

$$\forall v \in \text{vars}(p), \quad v \notin \text{vars}(\textsc{Standardize}(p)).$$

- **Unification**: $\textsc{Unify}(p, q)$ is an operator that checks whether two logical statements can be made equivalent by applying substitution and standardization to their variables. In Lifted Regression, this is used to match an action with a subgoal. Given two logical statements p and q, it returns a substitution θ, which is the Most General Unifier (MGU), such that:

$$\textsc{Unify}(p, q) = \theta \quad \text{where} \quad \theta(p) = \theta(q).$$

D Belief MDP Formalism

We cast object-centric active knowledge acquisition as a sequential decision problem: the agent selects which candidate objects to sense to maximize progress toward the goal. We model this as an MDP over the extracted object descriptions $\mathcal{O}$, relational descriptions $\mathcal{R}$, and lifted subgoals $\mathcal{G}$. Formally, we define a Markov Decision Process as a tuple $\mathcal{M} = \langle S, A, T, R \rangle$.

State Space. The state captures verified (true/false) knowledge about objects and relations:

$$S \;=\; \Big\{\, s = (O^+, O^-, R^+, R^-) \;\Big|\; O^{\pm} \subseteq \mathcal{O},\; R^{\pm} \subseteq \mathcal{R},\; O^+ \cap O^- = \varnothing,\; R^+ \cap R^- = \varnothing \,\Big\}.$$

Here O^+ and O^- are, respectively, the sets of object descriptions verified true or false; R^+ and R^- are the analogous sets for relational descriptions. Any description not in $(O^+ \cup O^-) \cup (R^+ \cup R^-)$ is unobserved.

Action Space. At each state, the agent chooses one unobserved description (object or relation) to sense:

$$A(s) \;=\; \Big\{\, \texttt{SenseObj}(o) \mid o \in \mathcal{O} \setminus (O^+ \cup O^-) \,\Big\} \;\cup\; \Big\{\, \texttt{SenseRel}(\rho) \mid \rho \in \mathcal{R} \setminus (R^+ \cup R^-) \,\Big\}.$$

Each action corresponds to an information-gathering operation that attempts to confirm whether the chosen description holds in the current environment.

Transition Model. The agent may sense an object description $o \in \mathcal{O}$ or a relational description $\rho \in \mathcal{R}$ only if it is currently unobserved. Moreover, a relational sensing action is enabled only when *all* argument-dependent object descriptions of ρ have already been sensed. We assume that the outcome of each sensing action depends only on the corresponding object or relational descriptions rather than the state s and s' in which the sensing occurs.

The transition model is therefore

$$T(s, \texttt{SenseObj}(o), s') = \begin{cases} p_o & s' = (O^+ \cup \{o\},\, O^-,\, R^+,\, R^-) \\ 1 - p_o & s' = (O^+,\, O^- \cup \{o\},\, R^+,\, R^-) \end{cases}$$

$$T(s, \texttt{SenseRel}(\rho), s') = \begin{cases} p_\rho & s' = (O^+,\, O^-,\, R^+ \cup \{\rho\},\, R^-) \\ 1 - p_\rho & s' = (O^+,\, O^-,\, R^+,\, R^- \cup \{\rho\}) \end{cases}$$

where $s = (O^+, O^-, R^+, R^-)$, $p_o \in [0, 1]$ is the agent's belief that object o can be sensed, and $p_\rho \in [0, 1]$ is the belief that ρ holds true.

Reward Function. The agent receives a terminal reward when its verified knowledge suffices to realize some lifted subgoal. Let $\mathrm{obj}(g) \subseteq \mathcal{O}$ and $\mathrm{rel}(g) \subseteq \mathcal{R}$ denote, respectively, the object and relational descriptions required by $g \in \mathcal{G}$. We define

$$R(s) \;=\; \begin{cases} 1, & \exists\, g \in \mathcal{G} \;\text{ such that g is satisfiable} \\ 0, & \text{otherwise,} \end{cases} \tag{12}$$

Having defined the MDP, we now describe how the transition probabilities p_o and p_ρ are estimated using a large language model (LLM) to provide commonsense priors over object existence.

E Extracting Belief Probability via LLMs

For belief probability extraction from LLM logits, we use OpenAI's API (`GPT-4o`) with a simple `yes/no` classification interface. The estimator takes as input an object description in natural language (e.g., "There exists a cooked piece of meat on a plate") and returns a scalar belief probability that reflects the model's confidence in the truth of the statement. To compute this, we prompt the LLM with an existential question and analyze its token-level response. We set the temperature to zero and enable logit tracking.

E.1 Prompt Construction

We first construct a natural language description of the target object based on its descriptive predicates. Each lifted subgoal generated by regression is represented as a conjunction of unary and binary predicates. After applying object-centric partitioning to extract individual objects from the subgoals, we obtain a set of unary predicates for each object. These predicates are then translated into a coherent natural language phrase that captures the object's type and its functional relationships. The resulting description is embedded into a fixed prompt template that assigns the language model the role of a home assistant robot operating in a household environment.

Below is an example prompt for object description query:

```
You are a home assistant robot operating in a common household environment.
Is it true that you can find an object such that this object is a plate and it is
    clean?

Please answer the question using only ['Yes', 'No']
```

Below is an example prompt for relational description query:

```
You are a home assistant robot operating in a common household environment.

Is it true that object X canclean object Y, given that object X is a cleaning
    object and object Y is a plate?

Please answer the question using only ['Yes', 'No']
```

Each query is repeated 5 times.

F Agent Overview

Algorithm 2 ACTIVEVOO Agent

Require: Instruction $\mathcal{G}_{nl}$, action model $\mathcal{A}$, interaction budget B

$G \leftarrow$ GOALPARSER$(\mathcal{G}_{nl})$
$S \leftarrow$ LIFTEDREGRESSION$(G, \mathcal{A}, \tau)$
$\Pi \leftarrow \{\pi \mid \langle g, \pi \rangle \in S\}$
$(\mathcal{U}, \mathcal{R}) \leftarrow$ PARTITIONSUBGOALS(S)
KB $\leftarrow I$; budget $\leftarrow B$
for all $\langle g, \pi \rangle \in S$ **do**
  $\mathcal{U}_g \leftarrow$ EXTRACTOBJECTS(g)
end for
while not SATISFIED(G, KB) **and** budget > 0 **do**
  $\mathbf{p} \leftarrow \{P(\varphi \mid \pi, \text{KB})\}_{\pi \in \Pi}$
  for all $o \in \mathcal{U}_g$ **do**
    VOO$(o) \leftarrow$ COMPUTEVOO$(o, \mathbf{p})$
  end for
  $\mathcal{O}_+ \leftarrow \{\, o \mid \text{VOO}(o) > 0 \,\}$
  $o^\star \leftarrow \arg\max_{o \in \mathcal{O}_+} \text{VOO}(o)$
  EXPLORE$(o^\star)$
  for all $o \in \mathcal{O}_+$ **do**
    OBSERVE(o); UPDATEKB(o,KB)
    budget $\leftarrow$ budget -1
    if budget = 0 **then**
      break
    end if
  end for
  for all $\langle g, \pi \rangle \in S$ **do**
    if SATISFIED(g, KB) **then**
      for all $a \in \pi$ **do**
        if ACT(a) **succeeds then**
          PROGRESS(KB,a)
        end if
        budget $\leftarrow$ budget -1
        if budget = 0 **then**
          break
        end if
      end for
    end if
  end for
end while
return KB

For every candidate plan $\pi \in \Pi$, the agent maintains a belief score $P(\varphi \mid \pi, \text{KB})$ that reflects the likelihood that the plan can currently succeed. Using these scores, the agent computes the Value of Observation (VOO) for each object description o extracted from every lifted subgoal g. All objects with strictly positive VOO are collected into the set $\mathcal{O}_+$. This positive VOO set represents all observable objects that can potentially improve or alter the agent's decision. The agent then queries the vision–language model for *all* objects in $\mathcal{O}_+$. Each query updates the knowledge base (KB) with new unary and relational facts. After the KB is updated, the agent scans the set of lifted subgoals.

Whenever the predicates of a subgoal g are satisfied, the associated plan π becomes executable. The agent immediately executes each action $a \in \pi$, updating the KB after each successful step. The loop terminates when the overall goal G is achieved or the interaction budget is exhausted.

G Ablations

Passive and Weakly Guided Baselines: We evaluate three alternative methods for extracting object information from the environment. *Exhaustive Object Acquisition* is a fully passive approach in which the vision–language model (VLM) is prompted to list all observable objects. *Goal-Directed Object Acquisition* is a weakly guided approach that prompts the VLM using only the overall task goal. *Plan-Directed Object Acquisition* first prompts the VLM to generate a plan and then queries for all objects mentioned in that plan. The exact prompts used in our experiments are provided below.

- **Exhaustive Object Acquisition:**

```
You are a home assistant robot operating in a common household environment
Give me all objects you can observe from the scene.

Please answer in this format: ["obj_1", "obj_2", ..., "obj_n"]
```

- **Goal-Directed Object Acquisition:**

```
You are a home assistant robot operating in a common household environment
Give me all objects you can observe from the scene that are related to the goal
    {GOAL}.

Please answer in this format: ["obj_1", "obj_2", ..., "obj_n"]
```

- **LLM-Subgoal Object Acquisition:**
  - *Extract high-level plan*

```
You are a home assistant robot operating in a common household environment
Your goal is to {GOAL}.

Give me a plan that can achieve the goal.
```

  - *Extract objects from the plan:*

```
You are a home assistant robot operating in a common household environment
Your plan is {PLAN}.

Give me all objects you can observe from the scene that are related to your
    plan.

Please answer in this format: ["obj_1", "obj_2", ..., "obj_n"]
```

All extracted object names are post-processed to align with the vocabulary used in ALFWorld; however, the list of ALFWorld object names is never exposed during extraction. We employ `GPT-4o` as the VLM in all of these experiments.

Component Ablations within ACTIVEVOO: We next disable or swap individual components of ACTIVEVOO:

- **No VOO:** Without VOO, the agent extracts objects only from the plan it believes is most likely to succeed, that is, $p = \arg\max_{p'} P(p')$, where the belief probabilities are obtained from the LLM. We retain the same object-partition method but limit extraction to the chosen plan p.
- **No VOO + No Partition:** VOO calculation requires partitioning objects by their first-order descriptions. When partitioning is disabled, we cannot compute per-object plan probabilities, so we ask the LLM for the feasibility of the entire plan. The prompt is:

```
You are a home assistant robot operating in a common household environment
Is the plan {PLAN} feasible to execute in this environment?

Please answer using only ['Yes', 'No'].
```

- **VLM Swap:** We replace `GPT-4o` with `LLaVA-13B` for vision–language object extraction while still using `GPT-4o` to compute belief probabilities.
- **LLM Swap:** We substitute `GPT-4o` with `GPT-3.5 Turbo` for belief-probability estimation, keeping `GPT-4o` as the VLM for visual information extraction.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The 3 contribution we listed in the introduction are all answered in the results section and ablations in Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation is included as a section in the Discussion in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The full mathematical derivation of our framework are shown extensively through section 2 with explicit assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The Section 2 shows our methodology with formal mathematical language

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Anonymous repo is provided in the technical supplementary material with source code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The set of implementation details and parameters are included Section 2 under "assumptions" and Section 3 under "Experimental Setup"

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All results are averaged over 3 runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware used to run experiments is given under Section 3 under "Experimental Setup".

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have reviewed and follow NeurIPS's code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work is done with simulated environment with no direct link to communicate with external users.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No new data or model are being released.

    Guidelines: The work is done with simulated environment with no direct link to external enviorment.

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All creator and owners of the material and assets used are properly credited in this work.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: All code and algorithm are well documented and provided.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing experiments and research with human subjects in this work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No animal or human subjects are involved.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: LLM are used only to check grammar and help with formatting text.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.