ADAPT: Benchmarking Commonsense Planning under Unspecified Affordance Constraints

Anonymous ACL submission

Abstract

Intelligent embodied agents should not simply follow instructions, as real-world environments often involve unexpected conditions and exceptions. However, existing methods usually focus on directly executing instructions, without considering whether the target objects can actually be manipulated, meaning they lack the ability to assess available affordances. To address this limitation, we introduce ADAPT, a benchmark that evaluates embodied agents in dynamic environments where object affordances may change over time and are not specified in the instruction. ADAPT requires agents to perceive object states, infer implicit preconditions, and adapt their actions accordingly. To enable this capability, we further propose Affordance-Aware Action Selection (AAS), a plug-and-play module that augments existing planners with explicit affordance reasoning. Experiments demonstrate that incorporating AAS significantly improves robustness and task success across both seen and unseen environments. We also show that a domain-adapted, LoRA-finetuned vision-language model used as the affordance inference backend outperforms a commercial LLM (GPT-4o), highlighting the importance of task-aligned affordance grounding.

1 Introduction

Humans have the ability to handle unexpected scenarios that are not specified in the instructions when performing everyday tasks. For example, when instructed to put cloth into a drawer, a person encountering a dirty cloth would naturally recognize that placing it into a drawer is inappropriate. Therefore, they go beyond literal instruction-following by considering context-specific preconditions, such as cleaning the cloth before use. This ability to reason about context-dependent object usability and to infer unstated preconditions is fundamental to human commonsense reasoning, and is essential for enabling embodied agents to operate robustly in real-world household environments. It motivates this work, aiming to verify *whether embodied methods can handle situations where object usability depends on state changes or surrounding conditions that are not explicitly mentioned in the instruction*.

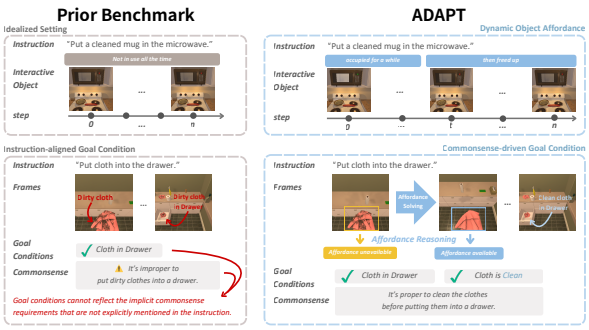

Figure 1: **Overview of ADAPT benchmark.** Unlike prior embodied benchmarks (left) that assume static object usability and fully specified goals, ADAPT introduces dynamic object affordances and commonsense-driven goal conditions (right). Agents must detect latent preconditions (e.g., cleanliness), resolve temporarily inapplicable actions, and adapt their behavior beyond literal instruction following.

Unfortunately, this limitation cannot be adequately studied using existing embodied AI benchmarks. Benchmarks such as ALFRED (Shridhar et al., 2020), BEHAVIOR-1K (Li et al., 2023), VirtualHome (Puig et al., 2018), and Sapien (Xiang et al., 2020) are built under idealized simulation settings where object states remain static and goal conditions are fully specified within the language instructions or scripted plans. As a result, agents are not required to detect unmet preconditions or reason about evolving object usability, which are essential for realistic deployment. While several recent approaches aim to improve robustness during execution, they address complementary aspects of the problem. Inner Monologue (Huang et al., 2022) adapt plans using human interaction when instruc-

tions are ambiguous or execution fails, but do not explicitly model object affordances as latent, state-dependent preconditions. Code-as-Monitor (Zhou et al., 2025) instead focuses on execution-time failure detection in short-horizon manipulation tasks. In contrast, our work targets long-horizon embodied tasks that require sustained reasoning over dynamic affordance constraints.

We argue that affordance should instead be treated as a *latent precondition* governing the applicability of an action. Even linguistically appropriate actions may be temporarily invalid due to object state changes or contextual constraints. Enabling agents to reason about such latent preconditions is a prerequisite for robust instruction-following in realistic household environments.

To systematically study this challenge, we introduce **ADAPT**, a new embodied AI benchmark designed to evaluate agents in dynamic and underspecified environments. ADAPT features everyday household tasks in which object affordances may change during task execution, and successful completion often depends on commonsense-driven goal conditions that are not explicitly stated in the instruction. For example, the instruction "put the cloth into the drawer" implicitly assumes that the cloth is clean, which may be violated in ADAPT and must be inferred and resolved by the agent.

Compared to prior benchmarks, **ADAPT** introduces two key challenges (see Figure 1): **Dynamic Object Affordance**, where objects may be temporarily unusable (e.g., in use, dirty, or inaccessible) and require preprocessing before interaction; and **Commonsense-Driven Goal Conditions**, where task success depends on implicit constraints that are not mentioned in the instruction but must be inferred by the agent. A detailed comparison between ADAPT and prior benchmarks is provided in Appendix A.

We evaluate several state-of-the-art methods, including MOCA (Singh et al., 2021), FILM (Min et al., 2021), CAPEAM (Kim et al., 2023), LLM-Planner (Song et al., 2023), and SayCan (Ahn et al., 2022) to assess their ability to handle the dynamic affordances and commonsense constraints on the ADAPT benchmark. Despite strong performance on existing benchmarks, these methods exhibit significant performance degradation under ADAPT's dynamic settings. This gap persists even when strong commercial vision-language backends such as GPT-4o are used, highlighting a fundamental mismatch between current models and the demands of dynamic embodied environments captured by ADAPT.

To address this limitation, we propose **Affordance-Aware Action Selection (AAS)**, a unified decision-time inference module that enables agents to reason about action applicability under dynamic affordances. **AAS** performs a joint inference process that (i) infers the affordance state of the target object from visual observations and (ii) resolves the next executable action conditioned on the inferred constraints. Integrated with strong embodied agents such as FILM and CAPEAM, AAS significantly improves robustness under dynamic conditions, yielding relative improvements of up to 73.2% in success rate and 34.7% in goal completion on the test unseen split.

Our contributions are summarized as follows:

- We introduce **ADAPT**, a benchmark that exposes the limitations of existing embodied agents under dynamic affordances and commonsense-driven goal conditions.
- We formalize action execution in embodied environments as an **action applicability inference problem** under latent preconditions, and propose **AAS** as a unified inference mechanism to address it.
- We demonstrate that integrating **AAS** with multiple state-of-the-art agents substantially improves robustness and adaptability, establishing a foundation for future research on affordance-aware embodied decision-making.

2 Related Work

2.1 Embodied Instruction Following

Embodied Instruction Following (EIF) requires agents to interpret natural language or symbolic specifications such as PDDL, and execute long-horizon plans in household environments through navigation and manipulation. Benchmarks such as ALFRED (Shridhar et al., 2020) and BEHAVIOR-1K (Li et al., 2023) advance this area by combining perception, language, and imitation learning. Recent methods improve generalization through modular or hierarchical architectures: MOCA (Singh et al., 2021) decouples object grounding from action prediction; FILM (Min et al., 2021) and HLSM (Blukis et al., 2022) decompose instructions into perception, planning, and memory; ET (Pashevich et al., 2021) and CAPEAM (Kim et al., 2023) enhance temporal consistency via recurrent memory

modules; HiTUT (Zhang and Chai, 2021) models hierarchical task structures by combining subgoal planning, navigation, and manipulation using unified transformers.

Despite strong performance on existing benchmarks, most embodied instruction-following methods implicitly assume static object usability and fully specified goal conditions. As a result, they lack mechanisms to detect unmet preconditions or recover from actions that are temporarily inapplicable, a gap explicitly exposed by ADAPT.

2.2 LLM- and VLM-Based Grounding in Embodied Robotics

Recent work has increasingly leveraged large language models (LLMs) and vision-language models (VLMs) to enable flexible, commonsense-driven planning for embodied agents. SayCan (Ahn et al., 2022) integrates GPT-3 (Brown et al., 2020) with a value-based affordance model trained via reinforcement learning to score action feasibility given the current visual context. SayCan does not explicitly reason over latent or temporally evolving preconditions, limiting its ability to handle dynamic object usability. LLM-Planner (Song et al., 2023) formulates planning as few-shot in-context learning, retrieving top-k demonstrations and applying logit biasing to constrain the action space. It further incorporates a grounded replanning mechanism based on simulator feedback; however, its reliance on deterministic transitions and predefined fallback strategies constrains its robustness in dynamic or partially observable settings.

2.3 Affordance Reasoning in Robotics

Robust embodied agents must reason about whether actions are feasible given the latent and dynamic properties of objects. However, most existing affordance modeling approaches focus on static, visually grounded attributes such as object presence or geometry (Ahn et al., 2022), without considering temporal usability, e.g., whether a microwave is currently in use or a cloth is dirty. Recent work such as SayCan scores LLM-suggested actions using learned affordance models based on CLIP-like visual embeddings. While effective for filtering immediately infeasible actions, such scoring mechanisms do not model conditional or temporally evolving preconditions. Other approaches, including Inner Monologue (Huang et al., 2022) and Code-as-Policies (Logeswaran et al., 2024), incorporate precondition reasoning via reasoning traces or code representations, but are either limited to symbolic domains or decoupled from perceptual feedback.

In contrast to prior affordance models that focus on immediate action feasibility, ADAPT explicitly evaluates whether agents can reason about when actions should not be executed due to unmet or evolving preconditions. This positions ADAPT as a diagnostic benchmark for studying affordance-aware action selection under realistic, dynamic constraints.

3 The ADAPT Benchmark

We introduce ADAPT, a benchmark designed to evaluate embodied agents' ability to reason compositionally and handle dynamic, long-horizon tasks with implicit preconditions. It features 2,628 expert demonstrations and 10,106 natural language task annotations across 57 scenes in the AI2-THOR 2.0 simulator (Kolve et al., 2017). The benchmark spans six task types which require agents to perform complex and temporally extended interactions such as picking, placing, cleaning, heating, cooling, and stacking. The distribution of these task types is detailed in Appendix B. Each annotations includes a task goal and step-by-step high-level descriptions, supporting hierarchical task understanding. By isolating dynamic affordance violations while keeping task structure and instruction format unchanged, ADAPT functions as a diagnostic benchmark that specifically evaluates an agent's ability to infer and recover from unmet preconditions.

3.1 Problem Statement

We consider an instruction-conditioned embodied agent tasked with executing high-level descriptions in a dynamic household environment. Each episode begins with a natural language directive G, representing the high-level goal, and a sequence of step-by-step low-level instructions $L = \langle l_1, l_2, \ldots, l_n \rangle$, where l_i denotes the i-th subgoal in the instruction. The agent must complete the task by composing a sequence of low-level actions from a library of skills. At each time step t, the agent receives an RGB observation o_t and the current language instruction l, and must decide the next skill a_t to execute. The task is challenging due to dynamic object affordances: object usability may change throughout the episode depending on agent actions or external factors (e.g., a microwave being occupied). These affordance shifts are not explicitly

stated in the instruction, requiring the agent to infer implicit preconditions and reason about latent constraints. To evaluate this capability, we introduce a benchmark where the agent must (1) interpret high-level goals, (2) monitor the usability of objects whose states may evolve over time, and (3) flexibly recompose its skill sequence to achieve success. Formally, the agent aims to produce a policy $\pi : (o_t, l) \mapsto a_t$ that completes the goal under partially observable and temporally dynamic object conditions.

3.2 Dataset Construction

Base Pipeline ADAPT is built upon the ALFRED trajectory generation pipeline, inheriting its task templates, instruction annotations, and expert demonstration framework. This design choice ensures direct comparability with prior embodied benchmarks, while isolating dynamic object affordances as the primary source of distributional shift.

Affordance State Injection To introduce dynamic affordance violations, we intervene in the initial object states of each episode by injecting affordance-specific constraints at task initialization. For a given instruction G and its associated low-level subgoals $L = \langle l_1, l_2, \ldots, l_n \rangle$, a subset of task-relevant objects is initialized in an *Unavailable* state according to the semantic preconditions implied by modifying object-level state attributes that encode semantic preconditions while keeping the instruction text and task goal unchanged. We consider several categories of affordance unavailability, including: (i) **Occupied objects**, where appliances such as microwaves are already in use; (ii) **Used objects**, where containers (e.g., pans or plates) are unavailable due to prior usage; and (iii) **Dirty objects**, where items such as cloths violate cleanliness-related preconditions. During execution, agents must monitor object usability and adapt their action selection a_t accordingly.

Static vs. Dynamic Split To enable controlled evaluation, ADAPT contains both **static** and **dynamic** affordance settings. In static episodes, all objects satisfy their assumed preconditions, following the idealized setup of prior benchmarks. In dynamic episodes, one or more target objects violate implicit affordance constraints at initialization, requiring agents to detect and resolve unmet preconditions during execution. Importantly, task goals, instruction text, and success criteria remain identical across the two settings, ensuring that performance differences can be attributed solely to affordance reasoning. Roughly half of the dataset is constructed under each condition. Full task statistics and corresponding evaluation results are provided in Appendix C.

	Train	Validation		Test	
		Seen	Unseen	Seen	Unseen
#Scenes	51	36	2	44	4
#Demonstrations	2628	189	217	228	223
#Annotations	10106	772	872	1081	995

Table 1: ADAPT data splits.

3.3 Data Splits

Table 1 presents the distribution of the dataset. The seen split includes 51 scenes, evenly divided between two room types: 27 kitchens and 24 bathrooms. The unseen split contains 6 scenes, with 3 from each room type, which are not included in the training data. This setup allows for evaluating both in-distribution generalization and performance under distribution shift.

Additional details on the construction of expert demonstrations, object affordance setting, and instruction annotation processes are provided in Appendix B.

4 Method

Conventional embodied planning systems typically assume that high-level actions are executable whenever they are linguistically valid. When an action fails, these systems often discard the action and trigger replanning from scratch, which can disrupt long-horizon task coherence. In contrast, we view action execution as governed by latent preconditions that may be temporarily violated due to dynamic object states or contextual constraints.

Based on this perspective, we formulate embodied decision making as an *action applicability inference problem*. An action that cannot be executed is not an error, but a temporarily inapplicable intention whose execution should be deferred until its preconditions are satisfied.

4.1 Affordance-Aware Action Selection (AAS)

We propose **Affordance-Aware Action Selection (AAS)**, a decision-time inference module that enables agents to reason about action applicability under dynamic affordances. Given the current observation and a planned high-level action, AAS

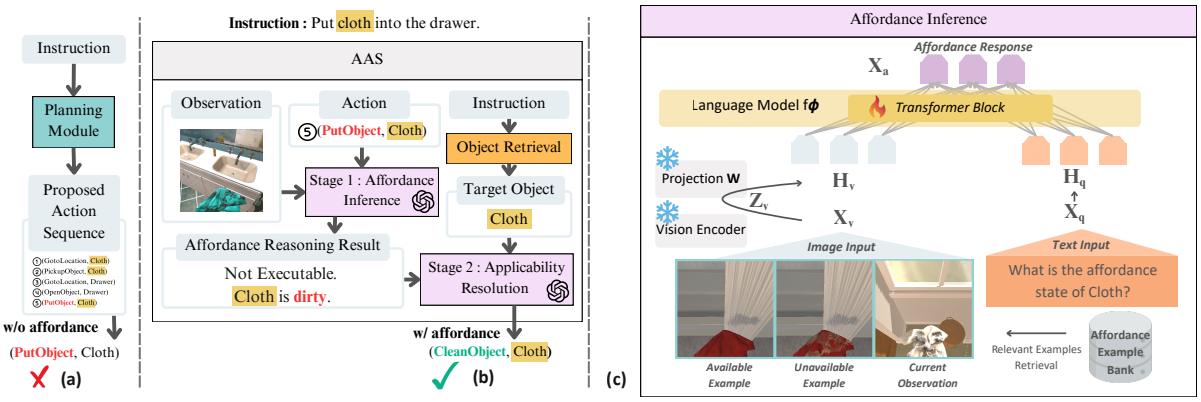

Figure 2: **Overview of AAS: Affordance-Aware Action Selection.** (a) Standard embodied instruction-following pipeline used in prior work, where the agent directly executes a planned action sequence without considering dynamic affordance constraints. (b) Our AAS framework augments action execution with affordance awareness. For each proposed action, the agent first performs *Stage 1: Affordance Inference* by jointly considering the current observation and the intended action. If the action is considered inapplicable (e.g., the microwave is occupied), *Stage 2: Applicability Resolution* selects an alternative executable action (e.g., waiting) instead of blindly executing the original plan. (c) Architecture of the affordance inference module, which combines LoRA-finetuned visual grounding with multimodal in-context learning using retrieved affordance examples to infer object usability.

jointly determines whether the action is executable and, if not, selects an alternative executable action that preserves task intent.

Figure 2 illustrates AAS as a unified decision-time inference process, in which action applicability is evaluated and resolved through two sequential phases: (1) *affordance inference*, which evaluates whether the action's latent preconditions are satisfied, and (2) *applicability resolution*, which identifies a suitable executable action when preconditions are violated. These phases are conceptual steps within a single decision process rather than independent modules.

AAS is architecture-agnostic and can be integrated into existing embodied agents without modifying their planning or low-level control components.

Stage I: Affordance State Inference AAS first infers whether the preconditions of the current high-level action are satisfied. Let $l_i \in L$ denote the planned high-level action, and let the *target object* be the object explicitly referenced and manipulated by l_i. Affordance inference is triggered only when l_i involves a target object known to exhibit dynamic affordance behavior (e.g., *Microwave*, *Cloth*).

LoRA Fine-Tuning To infer fine-grained affordance states (e.g., whether a cloth is clean or whether an appliance is available), we employ a vision-language model adapted via Low-Rank Adaptation (LoRA) (Hu et al., 2022). Specifically, we fine-tune LLaVA-1.5-7B using training data constructed by replaying expert demonstrations from the ADAPT training split. Each example is labeled as `available` or `unavailable` based on task-specific latent preconditions. No data from the validation or test splits is used during fine-tuning.

Figure 3 compares affordance prediction accuracy across multiple vision-language models. The fine-tuned model consistently outperforms general-purpose VLMs across object categories, demonstrating the effectiveness of task-specific adaptation for affordance grounding. This demonstrates the effectiveness of domain-specific fine-tuning for affordance understanding. A full breakdown of per-category results is provided in Appendix B.4.

Multimodal In-Context Grounding To further strengthen affordance inference, AAS incorporates multimodal in-context grounding through templated inputs. Each query consists of three images: (1) a reference image depicting the target object in an available state, (2) a reference image depicting the object in an unavailable state, and (3) the current egocentric observation.

These images are concatenated and paired with a textual prompt describing the reference states and querying the usability of the current observation. Reference examples are retrieved from a held-out affordance example bank that does not overlap with training data. This structured multimodal context enables the model to reason about object usability by direct visual comparison, improving general-

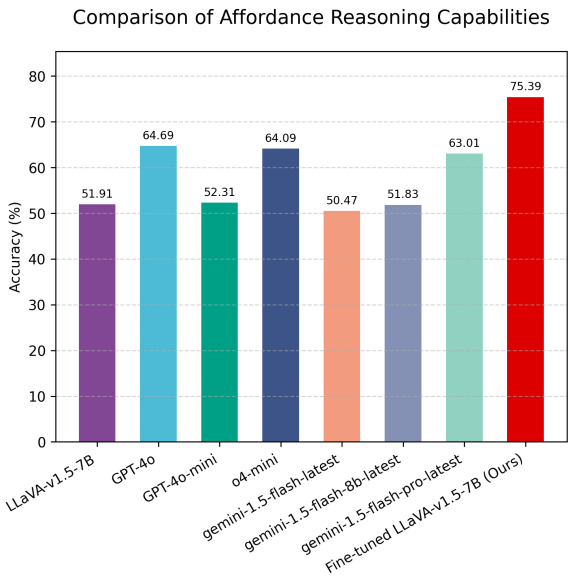

Figure 3: Isolates the affordance inference component of AAS and compares it with several commercial vision-language models under the same input format. Despite stronger general-purpose reasoning, commercial models underperform the LoRA-finetuned LLaVA on affordance inference by an average margin of more than 10%.

ization under unseen configurations. Additional details on example retrieval and prompt construction are provided in Appendix D.

Stage II: Applicability Resolution When the inferred affordance state indicates that the current action is `unavailable`, AAS defers the execution of the action and temporarily suspends progress until the required condition is satisfied. Rather than replanning from scratch, AAS maintains commitment to the original intention and infers a resolution strategy that restores action applicability.

Resolution is performed by querying a large language model with a structured prompt encoding: (1) the current observation, (2) the inferred affordance constraint, and (3) the deferred high-level action. The model infers a commonsense-consistent resolution action, such as waiting for a temporary constraint to clear or executing a preparatory action (e.g., cleaning).

Once the constraint is resolved, AAS resumes execution of the originally deferred action. This mechanism preserves long-horizon task coherence while enabling flexible adaptation to dynamic state violations, without explicit error signals from the environment. Implementation details and prompt templates are provided in Appendix E.

5 Experiments

5.1 Beselines

To evaluate the generality of **Affordance-Aware Action Selection (AAS)**, we integrate it into two supervised embodied agents, FILM (Min et al., 2021) and CAPEAM (Kim et al., 2023). In all settings, the base planner, perception modules, and low-level controllers remain unchanged. AAS operates solely as a decision-time inference module that intercepts high-level actions involving dynamic objects and resolves their applicability.

Supervised Methods We compare FILM+AAS and CAPEAM+AAS against MOCA (Singh et al., 2021), FILM (Min et al., 2021), and CAPEAM (Kim et al., 2023). To support deferred execution under temporary affordance violations, we extend the high-level action space with a `Wait` action and implement a corresponding low-level controller. For FILM, we additionally provide a BERT-based instruction encoder (Devlin et al., 2019) for compatibility. For CAPEAM, oracle subgoals and expert trajectories are provided to ensure that failures arise from missing affordance reasoning rather than subgoal generation. In addition, we include a variant where the affordance inference backend of AAS is replaced with a commercial vision-language model (GPT-4o (Hurst et al., 2024)), enabling a direct comparison between domain-adapted affordance modeling and general-purpose VLM reasoning.

Few-Shot Methods We evaluate LLM-Planner (Song et al., 2023) and SayCan (Ahn et al., 2022). For LLM-Planner, we compare the original prompting setup with a variant using demonstrations adapted to ADAPT's dynamic affordance conditions. For SayCan, we provide a ground-truth visibility oracle to reduce the action space, granting it an advantage during both action feasibility scoring and affordance evaluation.

All methods are evaluated on identical task instances from the ADAPT benchmark under both static and dynamic affordance settings. In static tasks, object usability remains fixed throughout the episode. In dynamic tasks, object affordances may be temporarily violated due to state changes or contextual constraints, requiring agents to infer unmet preconditions and adapt their actions. AAS is activated only under dynamic conditions, isolating its contribution to affordance-aware decision making.

Method	Test Seen				Test Unseen			
	GC	PLWGC	SR	PLWSR	GC	PLWGC	SR	PLWSR
Few-Shot Methods								
SayCan	4.79	3.58	0.46	0.11	10.50	6.71	0.00	0.00
LLM-Planner	7.16	1.84	1.11	0.54	15.34	4.33	2.46	1.33
Supervised Methods								
MOCA	4.10	3.33	0.46	0.10	9.72	9.80	0.00	0.00
FILM	11.36	11.37	2.77	1.46	25.54	24.01	9.34	3.27
CAPEAM	20.10	14.44	9.25	4.56	36.28	31.39	19.39	7.87
FILM + AAS (finetuned-LLaVA)	16.17	14.63	4.62	1.92	34.41	30.86	16.18	5.54
CAPEAM + AAS (finetuned-LLaVA)	**22.95**	**19.29**	**10.82**	**7.28**	**37.43**	**37.45**	**21.10**	**10.94**
CAPEAM + AAS (GPT-4o)	15.12	11.86	8.88	4.95	36.42	33.86	20.88	8.95

Table 2: Main results on the ADAPT benchmark.

5.2 Evaluation Metrics

We evaluate agent performance using four standard metrics for embodied instruction following. **Success Rate (SR)** measures whether the final task goal is achieved, while **Goal Condition Success Rate (GC)** reflects the proportion of goal predicates satisfied, allowing partial credit. To account for execution efficiency, we further report **Path-Length Weighted SR (PLW SR)** and **Path-Length Weighted GC (PLW GC)**, which weight outcomes by the ratio between the agent's trajectory length and the expert demonstration length.

Notably, ADAPT adopts commonsense-driven goal conditions that encode implicit constraints on object usability, making GC a particularly informative metric for evaluating affordance-aware reasoning.

5.3 AAS Consistently Improves Embodied Planning under Dynamic Affordances

Table 2 summarizes the main results. For few-shot methods, despite integrating with strong large language models, LLM-Planner (Song et al., 2023) and SayCan (Ahn et al., 2022) struggle to handle dynamic object usability. Similarly, the supervised baseline MOCA (**?**) exhibits limited performance under dynamic affordance conditions, indicating that explicit affordance awareness is missing in existing approaches. As for other supervised methods, across both FILM and CAPEAM, integrating Affordance-Aware Action Selection (AAS) leads to substantial and consistent performance improvements, demonstrating that AAS provides benefits across different planning architectures.

For FILM, the impact of AAS is particularly pronounced. On the test seen split, AAS improves success rate (SR) from 2.77 to 4.62 (+66.8%) and goal condition completion (GC) from 11.36 to 16.17 (+42.3%), accompanied by consistent gains in path-length weighted metrics. On the unseen split, the improvements are even larger: SR increases from 9.34 to 16.18 (+73.2%), and GC improves from 25.54 to 34.41 (+34.7%). These results indicate that AAS substantially enhances FILM's robustness in both familiar and novel environments by enabling more reliable handling of temporarily inapplicable actions.

CAPEAM also benefits consistently from AAS, though with smaller relative gains due to its stronger baseline performance. On the seen split, CAPEAM+AAS improves SR from 9.25 to 10.82 (+17.0%) and GC from 20.10 to 22.95 (+14.2%), with notable improvements in path-length weighted metrics, reflecting more efficient execution. On the unseen split, SR increases from 19.39 to 21.10 (+8.8%) and GC from 36.28 to 37.43 (+3.2%), while PLW SR and PLW GC improve substantially (from 7.87 to 10.94 and from 31.39 to 37.45, respectively). These gains suggest that AAS helps CAPEAM better preserve long-horizon task coherence when affordance constraints arise, even when overall success rates are already high.

We further compare different vision-language backends within AAS by replacing the LoRA-finetuned LLaVA with a commercial large language model, GPT-4o (Hurst et al., 2024). While GPT-

Method	Test Unseen	
	SR	GC
Full method	27.69	44.01
No LoRA fine-tuning	16.07	42.47
No Multimodal In-context Learning	6.15	33.49

Table 3: Ablation Study on AAS

4o (Hurst et al., 2024) provides strong general-purpose reasoning, it consistently underperforms the finetuned LLaVA within the AAS framework, particularly on seen environments. This result highlights the importance of task-specific visual grounding: effective affordance-aware action selection depends not only on reasoning capacity, but also on how well affordance-relevant visual cues are aligned with the task domain. Together, these findings demonstrate that AAS yields robust improvements across planners, and that domain-adapted affordance perception plays a critical role in its effectiveness.

All reported results are obtained from a single deterministic evaluation run on the full test split of ADAPT, with metrics aggregated over all episodes in each split. While commercial APIs may still exhibit minor nondeterminism, our evaluation focuses on affordance classification rather than open-ended generation, and the observed performance gaps are consistent across object categories.

5.4 Ablation Study

We ablate the two grounding mechanisms that provide affordance evidence to AAS: (1) task-specific adaptation of the vision-language model via LoRA fine-tuning, and (2) multimodal in-context learning (MICL) using affordance exemplars. We do not ablate AAS into independent components, as action applicability inference is inherently joint.

As shown in Table 3, replacing the fine-tuned LLaVA with its pretrained counterpart reduces SR from 27.69% to 16.07%, highlighting the importance of task-specific adaptation. Further removing MICL, by excluding reference exemplars and using only the current observation, leads to a sharp drop in SR to 6.15%. These results demonstrate that both LoRA fine-tuning and exemplar-guided multimodal prompting are critical for robust affordance grounding.

5.5 Case Studies

We present representative qualitative results in Appendix F. In a dynamic task (*"Microwave an egg and place it on the countertop"*), the baseline CAPEAM agent fails after 797 steps due to repeated execution of inapplicable actions. With AAS, the agent detects the temporary unavailability, waits, and resumes execution once conditions permit, completing the task in 206 steps.

A failure case under static conditions is shown in Appendix G, where AAS misclassifies a partially visible object due to occlusion. While rare, such cases highlight limitations in visual grounding and motivate future work on viewpoint selection and object disambiguation.

6 Conclusion

We introduce **ADAPT**, a new benchmark for evaluating embodied agents under dynamic object affordances and commonsense-driven constraints. Unlike prior benchmarks built under idealized assumptions, ADAPT explicitly requires agents to reason about latent and evolving preconditions that govern action applicability.

To address these challenges, we propose **Affordance-Aware Action Selection (AAS)**, a unified decision-time inference mechanism that enables agents to assess object usability and defer execution of temporarily inapplicable actions. By treating affordance as a latent precondition rather than a static property, AAS allows agents to preserve long-horizon task coherence while adapting to dynamic environmental constraints.

When integrated into strong embodied agents such as FILM and CAPEAM, AAS consistently improves both task success rate (SR) and goal condition completion (GC), with the largest gains observed under dynamic affordance settings. On the test unseen split, AAS yields substantial improvements on FILM and consistent gains on CAPEAM, demonstrating its effectiveness in recovering from unmet preconditions and reducing brittle instruction-following behavior.

Together, ADAPT and AAS highlight a critical yet underexplored aspect of embodied intelligence: the ability to reason not only about *what* action to take, but also *when an action should not be executed*. We hope this work encourages future research on affordance-aware decision making and robust execution under latent and evolving constraints.

Limitations

This work focuses on affordance reasoning under single-view egocentric observations to ensure comparability with existing embodied benchmarks and methods. As a result, AAS may be sensitive to partial occlusion and viewpoint ambiguity in certain cases, which can lead to incorrect affordance inference. Future work could incorporate multi-view perception or active viewpoint selection to improve robustness under visually challenging conditions. Finally, expanding ADAPT to capture richer physical variability and real-world interaction patterns remains an important direction for future work.

Potential Risks

While ADAPT and AAS are designed as diagnostic tools for studying affordance-aware decision making in simulated household environments, they do not model real-world physical uncertainty, safety constraints, or human–robot interaction dynamics. Consequently, direct deployment without additional safeguards could lead to inappropriate action deferral or overly conservative behavior in real settings. We emphasize that our framework is intended for research evaluation rather than immediate real-world use, and extending it to physical robots would require integrating safety-aware control, uncertainty modeling, and human-in-the-loop supervision.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, and 1 others. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. 2022. A persistent spatial semantic representation for high-level natural language instruction execution. In *Conference on Robot Learning*, pages 706–717. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, and 1 others. 2022. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Byeonghwi Kim, Jinyeon Kim, Yuyeong Kim, Cheolhong Min, and Jonghyun Choi. 2023. Context-aware planning and environment-aware memory for instruction following embodied agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10936–10946.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, and 1 others. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, and 1 others. 2023. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR.

Lajanugen Logeswaran, Sungryull Sohn, Yiwei Lyu, Anthony Liu, Dong-Ki Kim, Dongsub Shim, Moontae Lee, and Honglak Lee. 2024. Code models are zero-shot precondition reasoners. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5681–5697, Mexico City, Mexico. Association for Computational Linguistics.

So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. 2021. Film: Following instructions in language with modular methods. *arXiv preprint arXiv:2110.07342*.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba.

2018. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8494–8502.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. 2021. Factorizing perception and policy for interactive instruction following. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1888–1897.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. 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.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, and 1 others. 2020. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107.

Yichi Zhang and Joyce Chai. 2021. Hierarchical task learning from language instructions with unified transformers and self-monitoring. *arXiv preprint arXiv:2106.03427*.

Enshen Zhou, Qi Su, Cheng Chi, Zhizheng Zhang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and He Wang. 2025. Code-as-monitor: Constraint-aware visual programming for reactive and proactive robotic failure detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6919–6929.

A Embodied AI Benchmarks: A Comparative Perspective

Unlike most prior benchmarks that lack explicit goal conditions or dynamic affordance modeling, ADAPT uniquely integrates *commonsense-driven goal conditions*, supports *dynamic object affordances*, and uses *natural language instructions* for *long-horizon tasks*. While benchmarks such as BEHAVIOR-1K (Li et al., 2023) and ALFRED (Shridhar et al., 2020) support complex tasks, their goal representations are symbolic or static, and do not capture context-sensitive object usability. A summary comparison is provided in Table 4.

B Dataset Construction Details

B.1 Expert Demonstrations

We generate expert demonstrations by extending the ALFRED augmentation pipeline, and define six distinct task types in ADAPT which require agents to perform complex and temporally extended interactions such as picking, placing, cleaning, heating, cooling, and stacking. The distribution of these task types is illustrated in Figure 4. For each task type, we modify the PDDL domain to include dynamic-state predicates, For example, we add the predicates `(cleanable ?mo)` and `(isClean ?mo)` to model the "Dirty/Clean" scenario. This addition enables the Fast-Forward planner to interleave the necessary cleaning actions with the primary task, as illustrated in Listing 1.

B.2 Object Affordance Setting

To simulate real-world variability, object states in ADAPT are randomized at the start of each task. Objects are labeled as either *available* or *unavailable*, with semantics depending on the object type. For example, a microwave is unavailable when occupied, and a cloth is unavailable when dirty.

B.3 Task Complexity

To control task complexity, we define two difficulty levels. **Basic** episodes contain at most one unavailable object, allowing agents to solve the task with minimal adaptation. In contrast, **Advanced** episodes include up to two unavailable objects, which require more complex reasoning, object monitoring, and contingency planning to complete the goal.

B.4 Annotation Process

Traditional embodied AI benchmarks generate natural language instructions through a two-step process: first, expert demonstrations are synthesized using a deterministic planner, and then crowd workers on platforms such as Mechanical Turk manually write instructions based on the demonstration videos. While this approach yields human-readable annotations, it is time-consuming and often inconsistent across annotators.

To address these limitations and achieve high-quality yet efficient data generation, we adopt a semi-automated annotation pipeline composed of four stages: (1) **Template Refinement**, applying goal templates to improve completeness; (2) **LLM**

Paraphrasing, using prompt-based language models to diversify phrasing; (3) **OOV Detection**, replacing out-of-vocabulary or inconsistent terms to improve linguistic reliability; and (4) **Human Verification**, where trained annotators perform lightweight review and correction to ensure clarity and task feasibility.

C Task Distribution and Evaluation Results

To ensure fair and comprehensive evaluation, ADAPT includes both **static** and **dynamic** object affordance settings.

C.1 Static object affordance tasks

follow an idealized setting where object usability remains constant throughout the episode. These tasks simulate simplified environments commonly used in prior work and serve as a controlled benchmark to evaluate whether models perform comparably to existing methods under traditional assumptions. Table 5 presents the distribution of static object affordance tasks, and corresponding results are reported in Table 7.

C.2 Dynamic object affordance tasks

involve episodes where object usability may change during execution. These tasks test an agent's ability to detect shifting preconditions, monitor object usability, and recompose the policy plan as necessary. Table 6 presents the distribution of dynamic object affordance tasks, and corresponding results are summarized in Table 8.

Approximately half of the tasks in ADAPT are static and the other half are dynamic, enabling fine-grained comparison between models under both simplified and realistic conditions. Among the static tasks, a subset is directly reused from the original ALFRED (Shridhar et al., 2020) benchmark, ensuring compatibility and grounding in previously validated scenarios.

D Affordance Inference Capability

Figure 6 presents affordance prediction accuracy on five major object categories: Cup, Plate, Pot, Pan, and Microwave. Our fine-tuned LLaVA-v1.5-7B achieves the highest performance across all five categories, with substantial margins over both the base LLaVA and state-of-the-art general-purpose models. Notably, the model attains 90.40% accuracy on Pan and 95.62% on Microwave, demonstrating its strong capacity to reason about container-related affordances.

We hypothesize that this performance advantage arises in part from the distinguished visual presence of these objects in the agent's field of view. Objects like Pot, Pan, and Microwave typically occupy a large portion of the observation, providing more distinct spatial and contextual cues that facilitate affordance grounding. In contrast, smaller or more deformable objects, such as cloths or mugs, may present more subtle affordance shifts, which are further discussed in Figure 7. Nevertheless, when aggregated across all object categories, our fine-tuned model still outperforms all state-of-the-art general-purpose models, demonstrating its robustness and generalizability in affordance reasoning.

These results highlight the fine-tuned model's ability to combine visual grounding with commonsense reasoning, enabling it to consistently outperform both lightweight and large-scale foundation models on affordance-sensitive categories.

The code and partial dataset used in this evaluation are released as part of this submission; please refer to Appendix I for details.

E Affordance Reasoner Implementation

E.1 Visibility Detection

To determine whether the target object is currently visible to the agent, we use a pretrained LLaVA model. The model is queried only if the current high-level action involves an affordance-critical object and goal visibility is uncertain.

During inference, LLaVA receives the full egocentric frame as input. The accompanying prompt is dynamically adapted according to the object type, as detailed in Figure 8.

If the response indicates the object is visible, the process proceeds to the next step.

E.2 Affordance Detection

We empirically found that the pretrained LLaVA model performed poorly in detecting fine-grained usability attributes, such as whether a cloth is clean or a microwave is currently in use. To address this limitation, we fine-tuned LLaVA-1.5B using LoRA on a dataset collected by replaying expert demonstrations from ALFRED. This fine-tuning dataset is entirely separate from the ADAPT test split, ensuring a fair evaluation of affordance state recognition, as illustrated in Figure 9.

To further enhance prediction performance, we incorporate **multimodal in-context learning** (ICL) via templated input construction. As illustrated in Figure 5, each input to the model consists of a triplet of images: (1) a reference image showing the object in an available state, (2) a reference in an unavailable state, and (3) the current egocentric frame. These are concatenated and paired with a textual prompt describing the reference images and querying the usability of the current frame.

At inference time, visual examples are retrieved from a held-out affordance example bank that does not overlap with fine-tuning data. This structured prompting improves generalization and enables the model to reason about object usability by comparing the current observation with contextualized examples.

E.3 High-level Action Replanning

When the **Affordance Inference Stage** determines that the preconditions of a high-level action are violated, e.g., when the target object is in an **Unavailable** state, the system performs applicability resolution via LLM-based inference. A contextualized prompt is constructed using the current observation, the target object and its affordance status, and the set of available high-level actions. Based on this information, the LLM infers an alternative executable action to resolve the unmet condition, and the agent updates its task plan accordingly by inserting or reordering subgoals. An example prompt is shown in Figure 10.

The inferred resolution strategy depends on the nature of the affordance violation. For temporary constraints (e.g., an occupied appliance or blocked interaction), the agent executes a `Wait` action and periodically re-evaluates the affordance status. For persistent constraints (e.g., a dirty object), the agent inserts a corrective subgoal such as cleaning, temporarily placing any carried items on a nearby surface before resuming the original task.

F Case Study

Figure 11 shows a dynamic object affordance task: *"Microwave an egg and place it on the countertop."* In this scenario, the baseline CAPEAM (Kim et al., 2023) agent navigates to the microwave and attempts to open it. However, the microwave is initially in an `Unavailable` state, rendering it temporarily unusable. This triggers a failed action, after which the model enters a replanning loop and repeatedly predicts a `RotateRight` action followed by a `RotateLeft`, ultimately returning to its original position.

Although the microwave becomes available again during this loop, CAPEAM fails to resume the prior goal of opening it, as it lacks memory of the failed action. The agent continues rotating aimlessly until the episode ends unsuccessfully after 797 steps.

In contrast, the ARAM-enhanced CAPEAM first detects the `Unavailable` state of the microwave using the affordance reasoner and stores the `OpenObject` action as a pending action in the Action-Aware Memory (AAM). It then executes a `Wait` action and periodically reassesses the microwave's affordance. Once the microwave becomes available, the pending action is retrieved and executed. The agent proceeds to complete the task successfully in just 206 steps. This case demonstrates ARAM's ability to support flexible recovery and efficient replanning under dynamic state changes.

G Failure Case

Figure 12 presents a failure case on a static affordance task: *"Prepare and cook a potato in the microwave."* In this example, the baseline CAPEAM agent behaves as expected by navigating to the microwave, placing the potato inside, and successfully completing the task.

However, when ARAM is integrated, the affordance reasoner encounters partial occlusion: the microwave is only half visible from the agent's current pose. As a result, the model incorrectly classifies the microwave as `Invisible`. The agent turns away and mistakenly identifies a nearby dishwasher as the target object. It then incorrectly predicts the dishwasher's affordance as `Unavailable` and begins issuing `Wait` actions. After a while, the affordance flips to `Available`, and the model attempts to execute an `OpenObject` action on the dishwasher, leading to a failed trajectory.

This failure results from compounded errors in object perception and affordance reasoning. While such cases are rare, they highlight limitations in affordance reasoning under occlusion and visually ambiguous contexts. Addressing these challenges, such as through improved viewpoint selection or enhanced object disambiguation, remains an important direction for future work.

H Use Of AI Assistants

During the course of this work, we made limited use of AI-assisted tools as auxiliary aids. These tools were used primarily to improve the presentation quality of the manuscript, including enhancing readability, refining phrasing, and assisting with minor code refactoring. Importantly, the conception of the research problem, benchmark design, methodological development, and experimental evaluation were entirely carried out by the authors. AI-assisted tools did not contribute to model design, data construction, or scientific decision-making.

I Code and Data Availability

We provide code and data to reproduce the main experiments in our code appendix. The released package includes:

- **Benchmark Evaluation Data:** Test split data (both seen and unseen) of the ADAPT benchmark
- **Affordance Reasoning Evaluation:** We provide a subset of validation data for evaluating affordance reasoning with our LoRA fine-tuned LLaVA-1.5-7B and other vision-language models, including multimodal in-context prompt logs and the corresponding visual assets used during inference.
- **Preliminary Evaluation Results:** JSON files containing prediction outputs from both fine-tuned and pretrained models over the full ADAPT test set (1,252 samples)
- **Codebase:** Scripts for dataset statistics, affordance prediction evaluation. A local copy of the LLaVA repository is included for convenience.

Future Release Upon acceptance, we will release the complete training and validation splits, preprocessing and evaluation scripts, the full set of 1,252 evaluation images for affordance reasoning, and the LoRA fine-tuned LLaVA checkpoints.

J Fine-Tuning and Computing Infrastructure Details

We fine-tuned the LLaVA v1.5-7B model using LoRA, built on top of the Vicuna-7B base language model and the CLIP ViT-L/14-336 vision encoder. Training was performed on a single NVIDIA RTX 3090 GPU (24GB VRAM) using CUDA 11.8 and PyTorch 2.6.0. The model was trained for one epoch with a LoRA rank of 64, a learning rate of 1e-5, and a batch size of 4 per device.

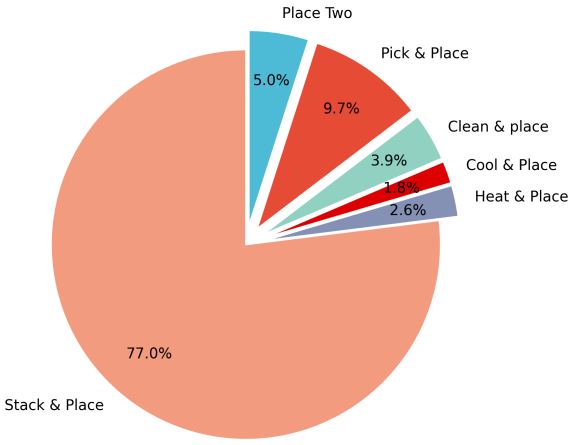

Figure 4: **Task types and annotation statistics in ADAPT.** ADAPT contains over 1,000 demonstrations across 6 task types of varying complexity, each paired with 3–6 language instructions. Tasks range from simple placement to hierarchical stacking involving movable and fixed containers.

Table 5: **Static Object Affordance Task Split** This table summarizes the subset of ADAPT tasks with static object usability throughout the episode.

	Train	Validation		Test	
		Seen	Unseen	Seen	Unseen
#Scenes	51	36	2	44	4
#Demonstrations	1580	98	154	46	122
#Annotations	5415	360	575	214	502

Table 6: **Dynamic Object Affordance Task Split** Tasks where object usability may change during execution. These represent realistic scenarios requiring agents to infer preconditions and adapt.

	Train	Validation		Test	
		Seen	Unseen	Seen	Unseen
#Scenes	51	36	2	44	4
#Demonstrations	1048	91	63	182	101
#Annotations	4691	412	302	867	493

Table 4: **Comparison of major embodied AI benchmarks.** Unlike most prior benchmarks that lack explicit goal conditions or dynamic affordance modeling, ADAPT uniquely integrates *commonsense-driven goal conditions*, supports *dynamic object affordances*, and uses *natural language instructions* for *long-horizon tasks*. While benchmarks such as BEHAVIOR-1K (Li et al., 2023) and ALFRED (Shridhar et al., 2020) support complex tasks, their goal representations are symbolic or static, and do not capture context-sensitive object usability.

Benchmark	**Simulator**	**Goal Condition**	**Dynamic Object Affordance**	**Task Specification**
SAPien (Xiang et al., 2020)	SAPien	✗	✗	User-defined
VirtualHome (Puig et al., 2018)	Unity	✗	✗	Natural language
BEHAVIOR-1K (Li et al., 2023)	OmniGibson	✓ (Symbolic PDDL)	✗	PDDL
ALFRED (Shridhar et al., 2020)	AI2-THOR	✓ (Instruction-aligned)	✗	Natural language
ADAPT (Ours)	AI2-THOR	✓ (Commonsense-driven)	✓	Natural language

Table 7: Main results on the ADAPT benchmark. (Tasks with Static Object Affordance)

Method	**Test Seen**				**Test Unseen**			
	GC ↑	**PLW GC ↑**	**SR ↑**	**PLW SR ↑**	**GC ↑**	**PLW GC ↑**	**SR ↑**	**PLW SR ↑**
Few-Shot Methods								
SayCan	21.15	15.00	2.34	0.58	21.90	12.73	0.00	0.00
LLM-Planner	26.10	7.49	4.21	1.89	28.33	8.47	4.77	3.08
Supervised Methods								
MOCA	19.50	17.17	2.33	0.59	21.47	17.03	0.00	0.00
FILM	34.34	33.67	13.55	7.39	42.02	37.11	16.73	5.44
CAPEAM	**55.08**	**43.82**	37.85	18.65	58.84	43.16	37.05	14.18
FILM + AAS (finetuned LLaVA)	31.86	33.52	10.74	5.58	46.21	44.99	21.51	7.87
CAPEAM + AAS (finetuned LLaVA)	54.67	43.50	**38.31**	**21.48**	59.10	**53.75**	**38.24**	**19.89**
CAPEAM + AAS (GPT-4o)	54.08	43.32	35.04	19.77	**59.64**	46.29	37.25	18.25

Table 8: Main results on the ADAPT benchmark. (Tasks with Dynamic Object Affordance)

Method	**Test Seen**				**Test Unseen**			
	GC ↑	**PLW GC ↑**	**SR ↑**	**PLW SR ↑**	**GC ↑**	**PLW GC ↑**	**SR ↑**	**PLW SR ↑**
Few-Shot Methods								
SayCan	1.11	0.84	0.00	0.00	0.96	0.67	0.00	0.00
LLM-Planner	2.90	0.73	0.35	0.27	4.47	1.21	0.00	0.00
Supervised Methods								
MOCA	0.64	0.35	0.00	0.00	0.31	0.50	0.00	0.00
FILM	6.20	6.09	0.11	0.06	12.35	10.97	1.82	1.11
CAPEAM	12.24	8.06	2.19	1.50	18.28	19.20	1.41	1.35
FILM + AAS (finetuned LLaVA)	12.65	9.77	3.11	0.98	**24.97**	18.37	**10.75**	**3.47**
CAPEAM + AAS (finetuned LLaVA)	**15.82**	**13.84**	**4.03**	**4.10**	20.07	**21.06**	3.65	1.94
CAPEAM + AAS (GPT-4o)	6.01	5.40	2.42	1.90	10.86	12.27	1.41	1.46

Listing 1: **Dynamic affordance scenario for tableware in a Stack and Place task.** This PDDL goal specification represents a scenario where dynamic object affordances occur on tableware items.

```
(:goal
    (and
        (exists (?mo # object)
            (and
                (objectType ?mo {mrecep}Type)
                (isReceptacleObject ?mo)
                (cleanable ?mo)
                (isClean ?mo)
            )
        )
        (exists (?mo # object)
            (and
                (objectType ?mo {mrecep}Type)
                (exists (?r # receptacle)
                    (and
                        (receptacleType ?r {recep}Type)
                        (exists (?o # object)
                            (and
                                (objectType ?o {obj}Type)
                                (inReceptacleObject ?o ?mo)
                                (inReceptacle ?mo ?r)
                            )
                        )
                    )
                )
            )
        )
        (forall (?re # receptacle)
            (not (opened ?re))
        )
    )
)
```

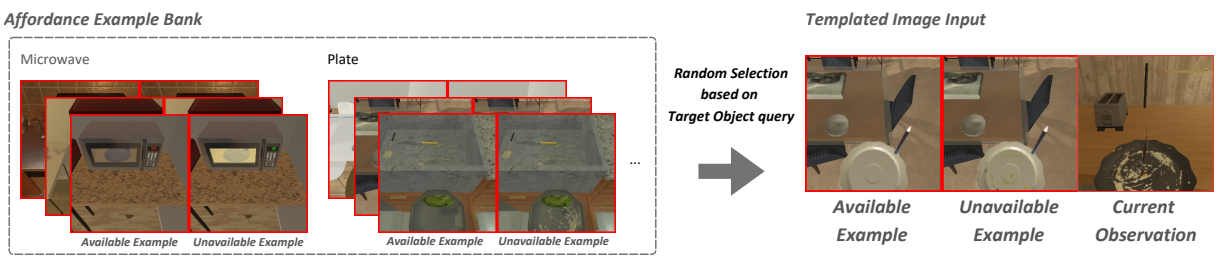

Figure 5: **Templated input for multimodal affordance reasoning.** Each input consists of a triplet of images—(1) a reference of the object in an available state, (2) a reference in an unavailable state, and (3) the current egocentric frame—paired with a textual prompt to assess the object's current usability.

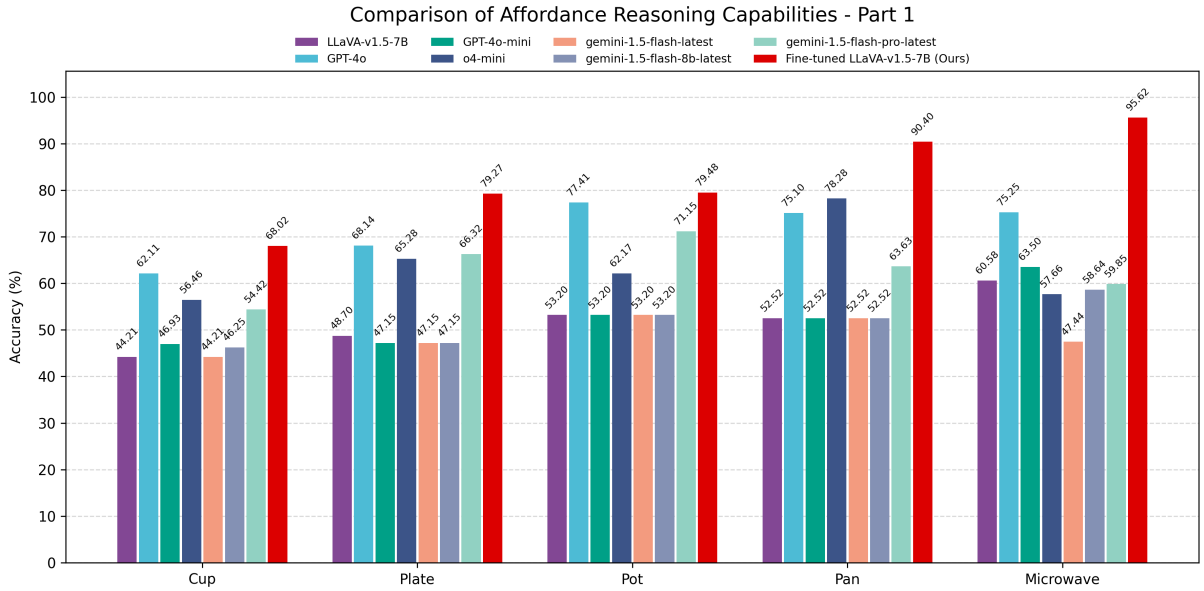

Figure 6: **Object Affordance Prediction Accuracy on Visually Distinguished Objects.** Our fine-tuned LLaVA-v1.5-7B model outperforms all baselines on large objects (e.g., Pot, Pan, Microwave) where visual cues are more spatially distinguished in the observation space. These results highlight the model's ability to leverage strong visual evidence for dynamic affordance reasoning.

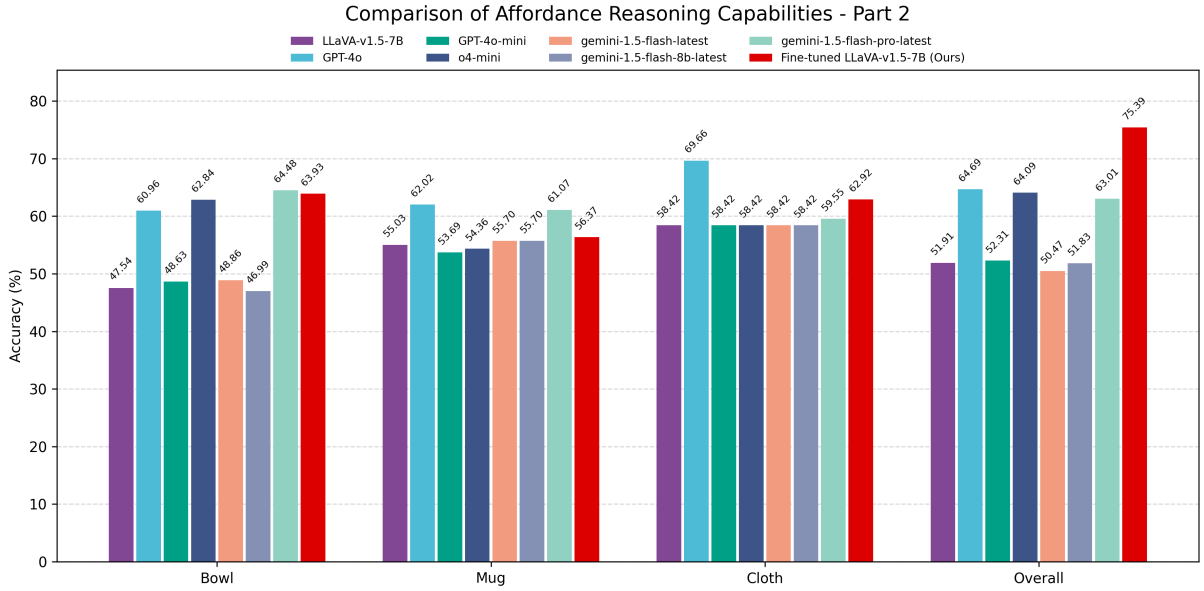

Figure 7: **Object Affordance Prediction Accuracy on Other Objects and Overall Performance.** While the gains from fine-tuning are less pronounced on smaller or more deformable objects (e.g., Mug, Cloth), our model still achieves competitive performance compared to much larger models like GPT-4o and Gemini-Pro. Overall accuracy reaches 75.39%, validating the effectiveness of task-specific adaptation.

Visibility Detection Prompt

Appliances :

*Check if a {**target_object**} is visible, typically mounted on the wall or placed on a kitchen countertop, and it is not under the stove. Answer yes or no*

Others :

*Is {**target_object**} visible? Answer yes or no*

Figure 8: **Prompt format for visibility detection.** We use an pretrained LLaVA-1.5-7B model to assess whether the target object is visible in the current egocentric view. The prompt is dynamically adapted based on the target object's category, guiding the model to make accurate visibility judgments.

Affordance Detection Prompt

Appliances :

Three microwaves from left to right: Available (dark), Occupied (yellow light), and a query image.
Only the light inside determines the status: dark = "Available", yellow = "Occupied".
What is the right one's status? Answer only "Available" or "Occupied".

Others :

*The image shows three {**target_object**}s: Clean (left, single color), dirty (middle, with stains), and the query (right).*
*Is the query {**target_object**} used? Answer only "yes" or "no".*

Figure 9: **Prompt format for affordance detection.** We query the fine-tuned model with structured prompts that include visual and textual context to determine the current usability of a target object.

Replanning Prompt

Action Space

"LookUp_15", "LookDown_15", "MoveAhead_25", "RotateLeft_90", "RotateRight_90", "CloseObject", "OpenObject", "GotoLocation", "PickupObject", "SliceObject", "PutObject", "CoolObject", "HeatObject", "CleanObject", "ToggleObjectOn", "ToggleObjectOff", "Wait"

Appliances :

*Here is a list of possible actions: {**Action Space**}. If I want to use the {**target_object**}, but it is occupied and unavailable for now.*
Which action should I choose to do ? Answer only the action itself.

Others :

*Here is a list of possible actions: {**Action Space**}. If I want to use a {**target_object**} but it is dirty now.*
*Which action should I choose to clean the {**target_object**}? Answer only the action itself.*

Figure 10: **Prompt format for high-level action replanning.** The large language model receives contextual prompts containing the agent's current observation, high-level action list, and affordance status to infer the next appropriate action.

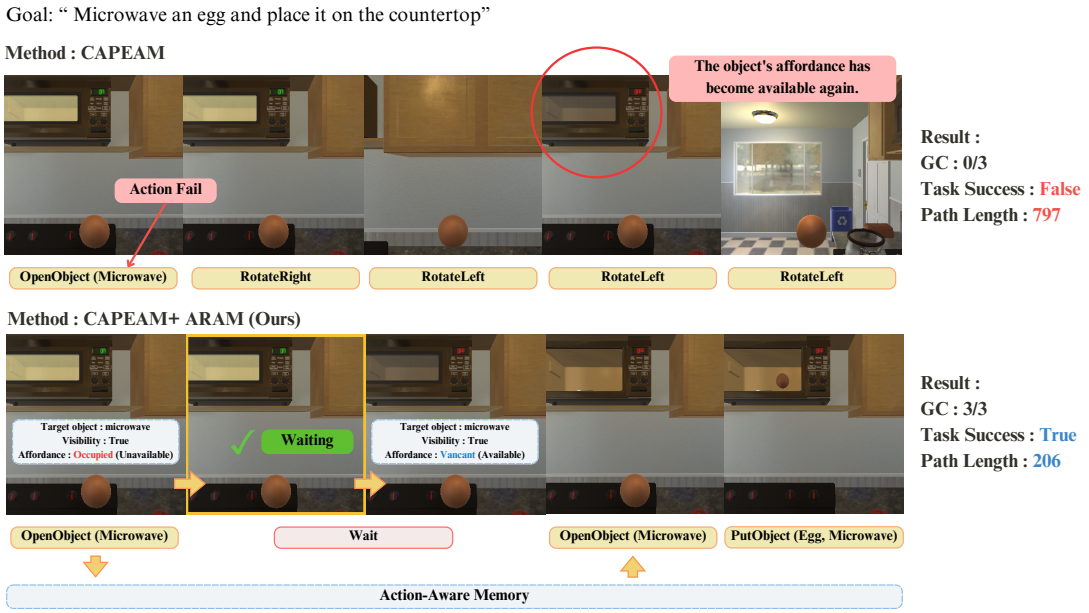

Figure 11: **ARAM improves robustness in dynamic environments.** In the task “Microwave an egg and place it on the countertop,” CAPEAM fails due to an occupied microwave and takes 797 steps. With ARAM, the agent detects the temporary unavailability, waits, and resumes the pending action, completing the task in only 206 steps—demonstrating improved adaptability in dynamic settings.

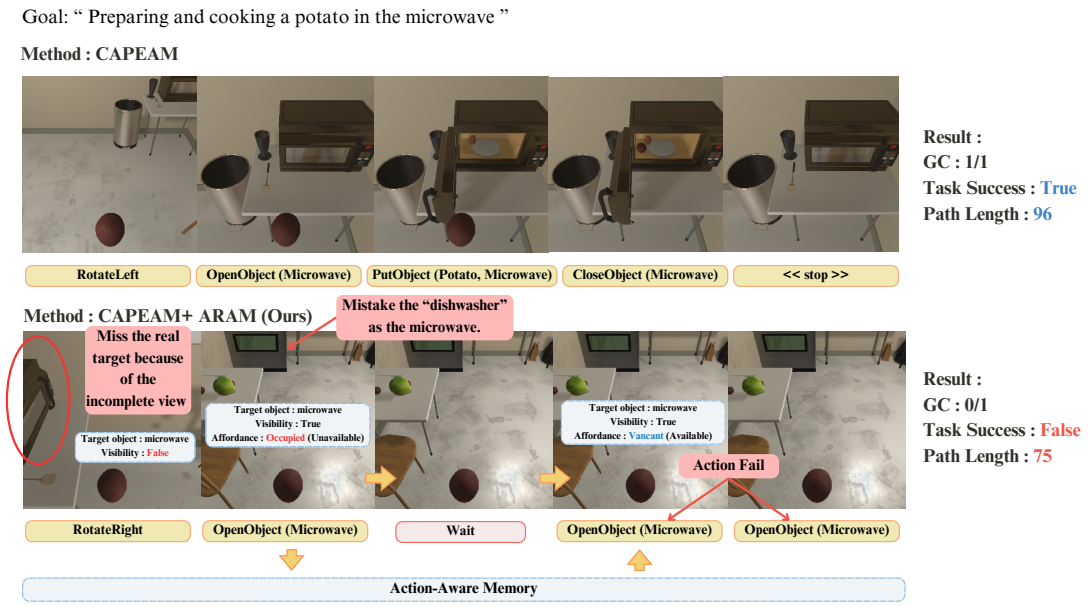

Figure 12: **Failure case due to misidentification.** In this failure case, the agent misclassifies a dishwasher as the microwave due to partial occlusion, triggering incorrect affordance reasoning and ultimately leading to task failure.