EmbodiedCLBench: EVALUATING CONTINUAL LEARNING FOR SELF-EVOLVING EMBODIED AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual learning requires models to accumulate past experience and transfer it to new tasks. Despite substantial research on continual learning, existing work remains largely confined to non-embodied settings, without extending to embodied environments. To address this gap, we introduce *EmbodiedCLBench*, the first benchmark in embodied simulation for evaluating the continual learning capability of self-evolving agents. With compositional generalization as the core evaluation guidance, our benchmark evaluates whether agents can first learn from basic tasks and then re-combine the acquired skills to solve complex, unseen advanced tasks. Based on *EmbodiedCLBench*, we conduct extensive experiments across multiple agent harnesses and learning methods, revealing consistent limitations and distinctive behaviors. Our results show that while advanced tasks can be hardly solved under zero-shot setting, the experience from basic tasks enables consistent improvement across different harnesses. The benefit of experience remains stable as the task complexity increases. However, this improvement does not scale accordingly with the number of learning samples, and current agents cannot extract meaningful information from additional experience. Collectively, our benchmark provides a valuable resource for future research, and our findings highlight continual learning as a fundamental yet underexplored capability and offer insights for designing more effective self-evolving agents in embodied environments.

1 INTRODUCTION

Self-evolving agents (Gao et al., 2025; Fang et al., 2025) can accumulate experience from past interactions and progressively enhance their task performance without additional human supervision. Such agents are capable of retaining and transferring skills across tasks over time, a capability broadly termed *continual learning* (Zheng et al., 2025b). This capability is particularly important for embodied intelligence (Liu et al., 2025a; Yifan et al., 2025; Liu et al., 2025b), where agents are expected to operate in complex, ever-changing physical environments that cannot be fully anticipated at training time. It is therefore important to develop a principled approach to evaluate the capability to learn from prior interactions and transfer the acquired skills to new tasks.

Existing benchmarks in agent continual learning and embodied intelligence address different aspects of the above evaluation purpose. Research on agent continual learning has largely focused on non-embodied settings, including knowledge-intensive tasks (Gao et al., 2026), interactive digital environments (Jiang et al., 2026), and long-term simulated worlds (Cai et al., 2025). These platforms do not capture the spatial complexity and physical constraints inherent in embodied tasks. Meanwhile, embodied benchmarks evaluate capabilities such as scene perception (Wang et al., 2024), visual navigation (Anderson et al., 2018), robotic manipulation (Gu et al., 2023), and long-horizon task planning (Shridhar et al., 2020), but are not specifically designed to assess continual learning. Thus, there remains a lack of a dedicated benchmark for evaluating whether self-evolving agents can accumulate experience and transfer it to solve novel tasks in embodied environments.

To address this gap, we introduce *EmbodiedCLBench*[1], the first benchmark in embodied simulation for evaluating the continual learning capability of self-evolving agents. As illustrated in Figure 1, our design stems from compositional generalization (Furuta et al., 2023; Guo et al., 2025). Instead

[1]Our benchmark will be publicly released.

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

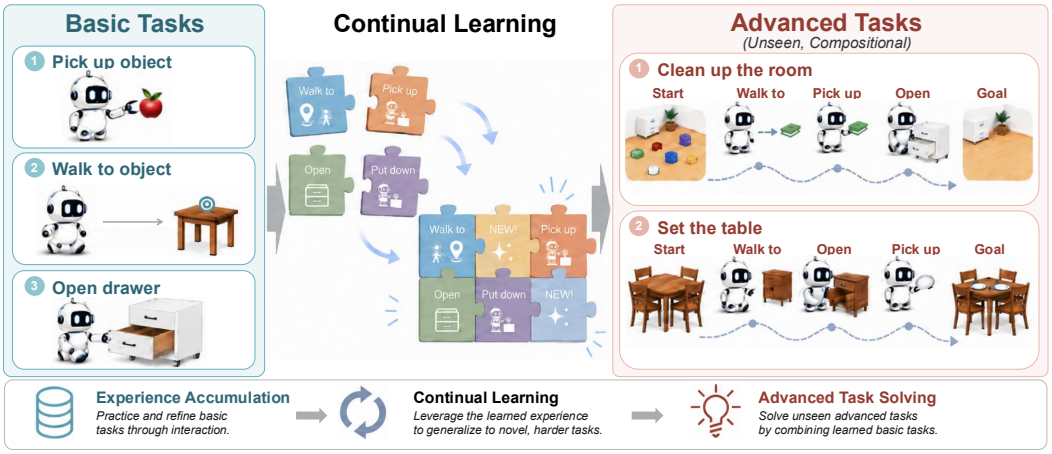


Figure 1: **Compositional generalization is leveraged as a principled way to measure continual learning capability.** An agent that has experience from basic tasks should recombine previously acquired skills to solve novel, unseen advanced tasks, rather than learning each new task from scratch.

of learning every new task from scratch, a self-evolving agent should recombine previously acquired skill elements to solve novel, unseen tasks. We therefore consider compositional generalization as a principled way to measure agent continual learning capability. Specifically, we first assign a set of basic tasks for the agent to execute and learn the skills. We then test whether the agent can effectively leverage such experience to solve unseen advanced tasks that require recombining the learned skills. From this perspective, basic tasks serve as test-time learning (Sun et al., 2020; Liu et al., 2021) data for the agent to build experience before attempting advanced tasks. To this end, *EmbodiedCLBench* organizes tasks into three compositional levels: (1) atomic actions from TongSim (Sun et al., 2025), representing basic object–action interactions; (2) basic tasks, with each composing a small number of atomic actions into practical household operations across multiple categories; and (3) advanced tasks, with each combining elements from multiple basic-task categories.

Based on *EmbodiedCLBench*, we conduct an extensive evaluation across multiple agent harnesses (e.g., OpenClaw and Codex) and learning methods (In-Context Learning and Skill Learning). Our results reveal several notable and, in some cases, unexpected findings. *First*, while advanced tasks can be hardly solved under zero-shot setting, agents with experience from related basic tasks achieve substantially higher success rates. This validates that experience from simpler tasks can be effectively transferred to unseen, harder tasks. *Second*, as the task complexity increases, the performance gain over the zero-shot agents retains at a level, suggesting that continual learning capability is not sensitive to task difficulty. *Third*, the improvement from learning does not scale accordingly with the number of learning samples. A single basic task significantly improves performance, and additional samples yield no consistent further improvement. *Fourth*, the past experience facilitates the decision behavior of agents by selecting more meaningful actions with fewer un-executable ones.

Our contributions are summarized as follows:

- We introduce *EmbodiedCLBench*, **the first benchmark** in embodied simulation for evaluating the continual learning capability of self-evolving agents.
- **Our key innovation** lies in leveraging compositional generalization as a principled way to construct tasks, design evaluation protocols, and evaluate continual learning capability.
- We conduct a **comprehensive evaluation** across multiple agent harnesses and learning methods, revealing limitations of current agents and providing insights for future design.

2 Related Work

Continual Learning Benchmarks. The evaluation of continual learning capabilities has been an increasingly active topic for LLMs and agents (Gao et al., 2025; Fang et al., 2025; Zheng et al., 2025b). Existing benchmarks span several distinct domains. In knowledge-intensive tasks, prior

work evaluates whether agents can effectively transfer procedural experience across coding, reasoning, and question-answering tasks (Gao et al., 2026; Wu et al., 2024; Shu et al., 2026). In interactive digital environments, benchmarks test continual learning capabilities in web automation, operating systems, and knowledge graphs (Jiang et al., 2026; Zheng et al., 2025a). In long-term simulated environments, benchmarks model persistent virtual worlds where past behaviors continuously shape future states and decisions (Cai et al., 2025). An important direction focuses on skill-centric evolution, studying whether agents can abstract reusable skills from experience and transfer them across tasks (Zhong et al., 2026; Zhang et al., 2026; Guan et al., 2026; Lei et al., 2026). Despite this progress, these benchmarks primarily operate in digital or symbolic environments, leaving continual learning in embodied scenarios largely unexplored.

Embodied AI Benchmarks. Embodied AI benchmarks evaluate agent capabilities ranging from perception to long-horizon task execution (Liu et al., 2025a; Yifan et al., 2025; Liu et al., 2025b). At the perception level, benchmarks test 3D scene understanding (Wang et al., 2024), spatial reasoning (Du et al., 2024), physical world understanding (Chow et al., 2025), and scene dynamics (Wang et al., 2025). For navigation, the field has progressed from language-conditioned navigation (Anderson et al., 2018; Ku et al., 2020) to object grounding (Qi et al., 2020), multi-modal lifelong navigation (Khanna et al., 2024), and active exploration for question answering (Majumdar et al., 2024). In manipulation, benchmarks evaluate multi-task generalization (Yu et al., 2020; James et al., 2020), object-level skill transfer (Gu et al., 2023), lifelong knowledge transfer (Liu et al., 2023), and everyday task scalability (Nasiriany et al., 2024). For long-horizon planning, benchmarks require agents to follow compositional instructions (Shridhar et al., 2020), interact through dialogue (Padmakumar et al., 2021), compose language-conditioned skills (Mees et al., 2022), handle diverse household activities (Li et al., 2023), and adapt plans based on visual feedback (Tupini et al., 2026). These benchmarks comprehensively cover perception, navigation, manipulation, and planning. Recent works further explore general-purpose embodied agent evaluation and realistic simulation environments (Peng et al., 2024; Ren et al., 2025). However, none is designed for evaluating the continual learning capability of skills in long-horizon tasks.

Compositional Generalization. Compositional generalization, the ability to recombine learned elements for solving novel problems, has been studied across multiple domains. In digital environments, benchmarks test whether mobile and web agents can compose atomic operations into multi-step tasks (Guo et al., 2025; Furuta et al., 2023). In video understanding, benchmarks study whether models can recognize unseen action–object combinations from separately learned components (Li et al., 2024; Ahn et al., 2026). For reasoning tasks, benchmarks show that models proficient in individual abilities still fail when these abilities must be combined (Alazraki et al., 2026; Li et al., 2025). In embodied manipulation, benchmarks evaluate whether robot policies can generalize to unseen skill compositions after training on atomic skills (Wu et al., 2026; Chen et al., 2025a;b; Haresh et al., 2024). In contrast, *EmbodiedCLBench* evaluates the continual learning capability of self-evolving agents in embodied simulation without any benchmark-specific training.

3 Benchmark Construction

As illustrated in Figure 2, the construction of *EmbodiedCLBench* consists of four stages: (1) environment enumeration, (2) task generation, (3) graph construction, and (4) evaluation protocol design. We describe each stage below.

3.1 Simulation Environment

Evaluating compositional generalization requires a large combinatorial space of object–action pairs, which demands a controllable environment that supports rich physical interactions and can scale to diverse task configurations. We adopt TongSim as the simulation platform (Sun et al., 2025), which has also been used in recent embodied-agent evaluation studies (He et al., 2026; Zhao et al., 2026).

Specifically, TongSim is a household simulation environment that supports a wide range of interactive **objects** (such as cups, plates, cabinets, lamps, and appliances) and executable **actions** (such as walk to, look at, pick up, put down, open, close, and switch on, etc.). This simulation offers three key advantages: (1) the platform supports a large number of object types and action primitives,

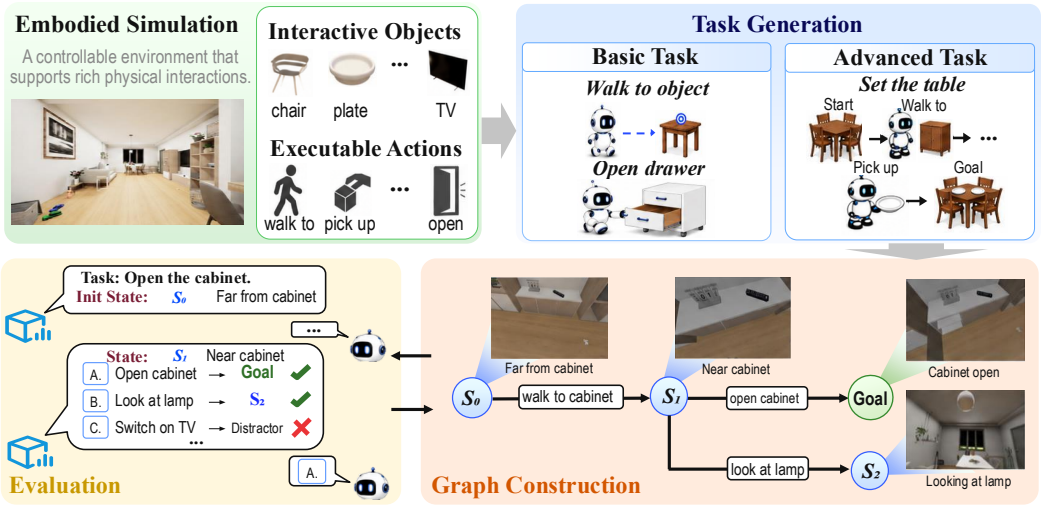


Figure 2: **Overview of the construction process of *EmbodiedCLBench*.** During task generation, basic tasks combine a few atomic actions into simple household operations, and advanced tasks combine elements from multiple basic-task categories into compositional tasks. Each task's initial and goal states are used to build a complete state-transition graph. During evaluation, the agent observes the current state and selects from candidate actions. After each selection, the agent transitions to the next node on the graph.

providing sufficient combinatorial diversity; (2) all scenes are indoor domestic environments, so that the tasks are grounded in meaningful, real-world household activities; and (3) TongSim has not been widely used as training data for current foundation models, making it suitable for evaluating the generalization capability of agents with reduced risk of data leakage.

3.2 Task Design: Compositional Generalization

Our goal is to evaluate the capability of continual learning. However, it is an abstract concept that is challenging to measure directly. We operationalize it through compositional generalization: if an agent has learned from basic tasks, it should recombine previously mastered skill elements to solve novel tasks rather than learning each new task from scratch. This leads to a natural evaluation protocol—let the agent first complete simpler tasks to accumulate experience, then test whether it can leverage that experience to solve more complex, unseen tasks.

Based on atomic actions available in TongSim, we design two levels of tasks. (1) **Basic tasks** are practical household operations composed of approximately two atomic actions, organized into six categories: object manipulation, spatial placement, accessibility control, appliance operation, activity-driven scene configuration, and household state restoration. (2) **Advanced tasks** are complex multi-step operations composed of approximately four atomic actions, each combining elements from two to four basic-task categories. All task descriptions and action options are generated by Qwen3.7-Plus and manually verified for correctness (see Appendix B). The agent first completes basic tasks as a learning phase, and is subsequently tested on advanced tasks. In this setup, basic tasks serve as memory (Xu et al., 2026; Huang et al., 2026) and test-time learning (Sun et al., 2020) data for both the In-Context Learning and Skill Learning settings described in Section 4.1.

3.3 Action Trajectory Annotation: Graph Construction

We use a closed-source version of TongSim, making live simulation runs impractical for evaluation. We therefore construct an offline, reproducible evaluation substrate in the form of a state-transition graph. A graph structure naturally represents the decision process in which an agent selects an action at each state and transitions to the next, and offers three advantages: (1) fully offline evaluation without a running simulator; (2) precise control of task difficulty through graph branching and depth; and (3) fine-grained analysis of agent behavior by recording every decision.

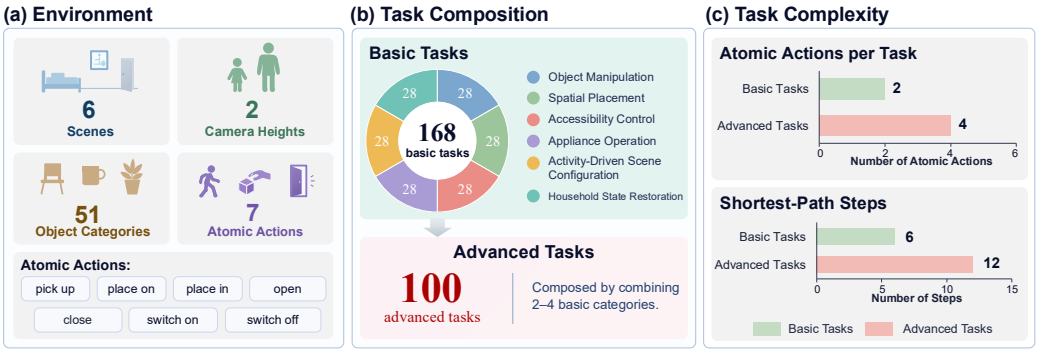


Figure 3: Benchmark statistics of *EmbodiedCLBench*: environment configuration (a), task composition across six categories (b), and task complexity at two levels (c).

Each node represents a distinct environment state (e.g., "near the desk and looking at the lamp"), and every node is paired with a real image captured from TongSim. Each edge represents an executable action that deterministically transitions the agent to the next state. Given a task, we determine the initial and goal states, then use breadth-first search to find the shortest successful path. Since each state supports only a bounded set of objects and actions, the reachable state space is compact enough for reliable graph search. We construct distractor paths by expanding alternative executable actions from nodes along the shortest path, producing trajectories that deviate from the optimal route.

3.4 Evaluation Protocol

With the graph in place, we define how agents interact with it. The evaluation follows an interactive, multi-turn format: (1) the agent receives a task description and an image of the initial state; (2) at each step, the agent is presented with multiple action candidates: optimal actions that follow the shortest path, suboptimal actions that are executable but deviate from the shortest path, and invalid actions that cannot be executed in the current state; (3) based on its selection, the agent transitions to the next state and receives a new image with updated candidates; (4) this repeats until the agent reaches the goal (success) or exhausts the step budget (failure). This interactive format preserves the sequential decision-making structure of embodied tasks while enabling fully offline, deterministic, and reproducible assessment. The action candidates at each node are generated by Qwen3.7-Plus and manually verified to ensure correctness. The detailed construction pipeline and task examples are provided in Appendices B and C.

4 Main Experiment

To measure the continual learning capability of self-evolving agents in embodied simulation, our core design is as follows: agents first complete a set of basic tasks to accumulate experience, and we then test whether this experience enables them to solve more complex, unseen advanced tasks.

4.1 Experimental Setup

Benchmark Overview. Figure 3 summarizes the overall statistics of *EmbodiedCLBench*. The environment covers diverse scenes, object categories, and atomic actions, providing a broad combinatorial space for task construction. The benchmark contains 268 tasks in total: 168 basic tasks and 100 advanced tasks. Basic tasks span six categories, each grounded in a distinct type of everyday household activity. Advanced tasks require substantially more atomic actions and shortest-path steps, forming a clear complexity gradient. Each atomic action corresponds to 3 steps in the graph, so basic tasks (2 atomic actions) require a minimum of 6 steps and advanced tasks (4 atomic actions) require 12. Both levels are thus long-horizon tasks. This scale, combined with real-world grounding and structured difficulty, gives *EmbodiedCLBench* the diagnostic coverage needed to reveal meaningful differences in agent continual learning capability. Task examples at both compositional levels are provided in Appendix C.

Evaluation settings. We evaluate agents under three settings: (1) **Zero shot:** the agent performs basic and advanced tasks directly, without prior experience or learning phase. (2) **In-Context Learning (ICL):** the agent first completes two related basic tasks; the full interaction history is retained in context, and the agent attempts an advanced task that recombines elements from the learned basic tasks. (3) **Skill Learning:** the learning phase covers the same basic tasks as ICL, but after completing them the agent summarizes its experience into a concise Markdown-formatted skill description; at test time only this skill summary is provided instead of the interaction trajectories.

Testing phases allow 3 extra steps relative to the shortest path to provide a small tolerance for sub-optimal decisions; learning phases allow 6 to give agents more room for exploration.

Agent configurations. Each task runs under one of four agent harnesses: OPENCLAW (OpenClaw, 2026), CLAUDE CODE (Anthropic, 2026), CODEX (OpenAI, 2026b), and HERMES (Nous Research, 2026). Each harness calls the official API of two backbone models: Qwen3.8-Flash (Qwen, 2026) and GPT5.6-Terra (OpenAI, 2026a).

Table 1: **Main results on *EmbodiedCLBench*.** For In-Context Learning and Skill Learning, agents first learn from two basic tasks and then attempt one advanced task. SR (%) reports the percentage of tasks completed within the step budget.

SR (%)		BASIC TASK	ADVANCED TASK		
Harness	**Model**	Zero-shot	Zero-shot	In-Context Learning	Skill Learning
HERMES	Qwen3.8-Flash	58.9	8.0	44.0	33.0
	GPT5.6-Terra	53.0	11.0	16.0	34.0
OPENCLAW	Qwen3.8-Flash	50.6	5.0	35.0	38.0
	GPT5.6-Terra	47.6	12.0	39.0	22.0
CODEX	Qwen3.8-Flash	52.4	4.0	34.0	33.0
	GPT5.6-Terra	54.8	11.0	52.0	26.0
CLAUDE CODE	Qwen3.8-Flash	42.9	3.0	27.0	31.0
	GPT5.6-Terra	51.8	9.0	31.0	25.0

Metrics. We report Success Rate (SR), which measures whether an agent can complete a task within the allotted step budget:

$$\mathrm{SR} = \frac{1}{N}\sum_{i=1}^{N} c_i\,, \quad c_i \in \{0,1\} \text{ indicates whether task } i \text{ is completed.} \tag{1}$$

Success rate directly reflects continual learning capability—whether accumulated experience enables the agent to solve novel, compositionally harder tasks.

4.2 MAIN RESULTS

Table 1 presents the main results. Agents achieve moderate zero-shot success rate on basic tasks (around 50%), indicating that current agents can execute simple household tasks. Zero-shot success rate on advanced tasks drops to around 10%, confirming that advanced tasks are genuinely difficult. Both ICL and Skill Learning substantially improve advanced task success rate, raising it from around 10% to above 30%. This improvement is consistent across all harnesses, confirming that experience from basic tasks can effectively transfer to unseen advanced tasks. However, the results remain far from saturated, indicating substantial room for improvement. Comparing ICL and Skill Learning, neither method holds a consistent advantage across harnesses. We conjecture that both methods extract similar information from the learning experience, leaving some useful knowledge untapped by either approach. Across harnesses and backbone models, we observe no significant performance gap under the same evaluation setting. This suggests that current harnesses and backbones lack designs tailored for continual learning. Targeted improvements in this direction are needed.

***Finding 1**: While advanced tasks are hardly solvable under zero-shot setting, experience from basic tasks enables consistent improvement across different harnesses and models.*

5 Analysis

5.1 Effect of Compositional Complexity

Motivation and Setting. Compositional generalization is the core approach for continual learning in *EmbodiedCLBench*. A natural question follows: does more complex composition make continual learning harder? In the main experiment, advanced tasks consist of 4 atomic actions. Here, we extend our evaluation data to include more atomic actions, ranging from 3 to 6, to control compositional complexity. We measure both the raw success rate at each level and the improvement over zero shot—the success rate of In-Context Learning or Skill Learning minus the zero-shot success rate. We use Hermes with Qwen3.8-Flash for this analysis.

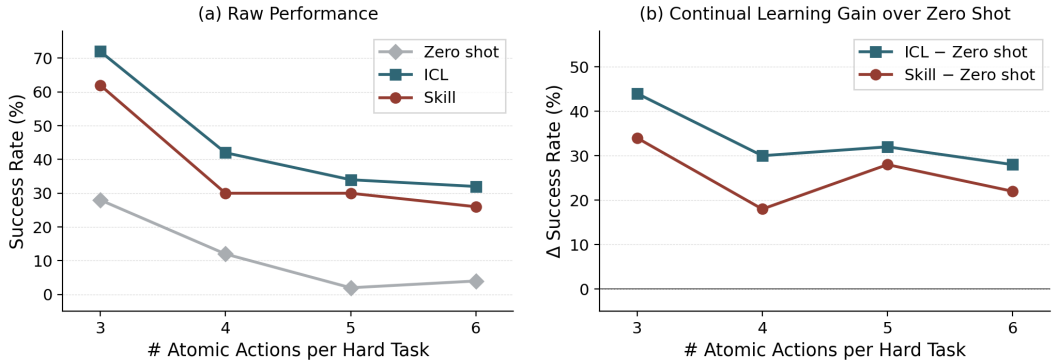


Figure 4: **Effect of compositional complexity on continual learning.** The x-axis varies the number of atomic actions per advanced task from 3 to 6. (a) reports the raw success rate achieved by each setting. (b) reports the gain over zero shot, defined as the success rate of ICL or Skill Learning minus that of zero shot, isolating the improvement attributable to the learning phase.

Results and Analysis. As shown in Figure 4(a), success rate decreases across all three settings as the number of atomic actions increases from 3 to 6. The largest drop occurs between 3 and 4 atomic actions (e.g., ICL falls from 72.0% to 42.0%), after which the success rate levels off. Higher complexity uniformly degrades performance, regardless of whether agents have prior experience. Compared to the zero-shot setting, where success rate decreases sharply (e.g., from 32.0% to 4.0%), ICL and Skill Learning remain more stable. Specifically, Figure 4(b) shows that the gain over zero shot drops initially but then stabilizes at around 30%. This indicates that the benefit of experience persists even as compositional difficulty increases. At the highest complexity (6 atomic actions), zero-shot success rate drops to 4.0%, while ICL and Skill Learning still achieve 32.0% and 26.0%. Even when zero-shot agents can barely solve tasks, experience provides substantial improvement.

***Finding 2**: In zero-shot setting, success rate decreases sharply as complexity increases. In contrast, the gain from experience remains stable, indicating that the benefit of continual learning is less sensitive to task difficulty.*

5.2 Scaling with the Number of Learning Samples

Motivation and Setting. Continual learning assumes that accumulated experience improves an agent's performance on harder tasks. If this holds, more experience should lead to higher success rate on advanced tasks. In the main experiment, agents learn from 2 basic tasks before attempting advanced tasks. Here, we increase the number of learning samples from k=0 (zero shot) to k=6 and measure success rate on the same set of advanced tasks under both In-Context Learning and Skill Learning settings. The resulting curve shows whether success rate steadily improves as agents

accumulate more experience. We evaluate two agent harnesses, HERMES and OPENCLAW, both using Qwen3.8-Flash as the backbone model.

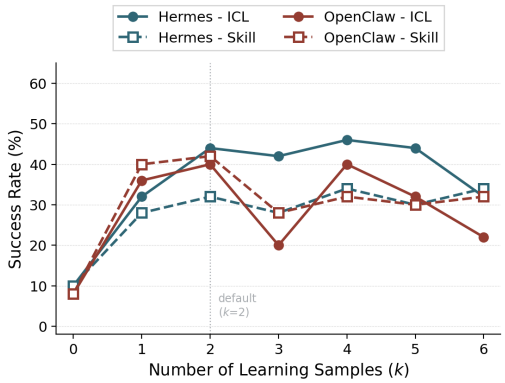


Figure 5: **Success rate scaling with the number of learning samples.** Each learning sample is a basic task that the agent completes during the learning phase before being tested on advanced tasks.

Results and Analysis. As shown in Figure 5, the most substantial improvement occurs between k=0 and k=1: a single learning sample raises success rate from under 10% to over 30% across both harnesses and learning methods. This initial jump accounts for the majority of the total improvement observed throughout the entire k range. Even minimal experience—completing just one basic task—provides sufficient new information for the agent to substantially improve on unseen advanced tasks. Beyond k=1, however, success rate does not continue to rise but instead fluctuates without a clear upward trend. This pattern holds across ICL and Skill Learning, and across both harnesses—none of the four curves shows consistent improvement with additional learning samples. This suggests that current models cannot extract sufficiently distinct information from additional samples. They tend to derive similar, redundant knowledge regardless of how many samples are provided. Achieving scalable continual learning—where success rate improves steadily with accumulated experience—remains an open challenge for embodied agents.

> ***Finding 3**: Success rate does not scale accordingly with the number of learning samples. A single learning sample significantly improves performance, while additional samples yield no consistent further gain.*

5.3 ERROR ANALYSIS

Motivation and Setting. The main results show that ICL and Skill Learning improve success rate, but do not reveal what agents actually learn from experience. In the evaluation protocol (Section 3.4), for each step, the action options at each step fall into three types: optimal actions that follow the shortest path, suboptimal actions that are executable but deviate from it, and invalid actions that cannot be executed in the current state. We analyze how agents choose among different action types across all decision steps in the main experiment. Specifically, we examine whether the behavior patterns change after ICL and Skill Learning compared to the zero-shot setting. We analyze decision behavior across all four harnesses with Qwen3.8-Flash based on the main experiment.

Results and Analysis. As shown in Figure 6, experience sharply reduces invalid actions across all harnesses (by 13%–20%). Fewer invalid selections allow agents to stay on executable trajectories and reach the goal within the step budget. These behavior changes directly account for the success rate improvements in Table 1. Suboptimal actions, however, do not decrease but instead increase slightly (by 3%–8%). This suggests that the reduction in invalid actions is partially redistributed to both optimal and suboptimal categories. The net effect is a substantial increase in optimal actions. ICL and Skill Learning show the same direction of change, but Skill Learning consistently produces

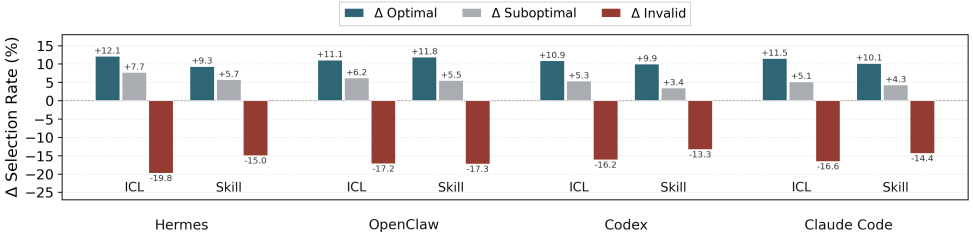


Figure 6: **Error analysis of decision behavior.** Each group of bars shows the change in selection rate relative to zero-shot for three action types: *optimal actions* follow the shortest path, *suboptimal actions* are executable but deviate from it, and *invalid actions* cannot be executed in the current state.

smaller shifts across all harnesses. This pattern stands in contrast to the main results (Table 1), where the success rate ordering between ICL and Skill Learning varies across harnesses. This may reflect a difference in how experience is represented: raw interaction trajectories induce broader changes, while distilled skill summaries produce fewer but more focused improvements.

> ***Finding 4****: Continual learning alters the decision behavior of agents, resulting in an increase in optimal actions and a decrease in infeasible actions.*

5.4 Step Budget Sensitivity

Motivation and Setting. Each task has a shortest path to the goal, but agents often make suboptimal decisions or take detours along the way. To accommodate this, we allow extra steps beyond the shortest path as a small tolerance for suboptimal decisions. By default, agents are allowed 3 extra steps. Here, we ablate this design choice by varying the number of extra steps from 0 to 6 to examine how the step budget affects success rate under different evaluation settings. We use HERMES with Qwen3.8-Flash for this analysis.

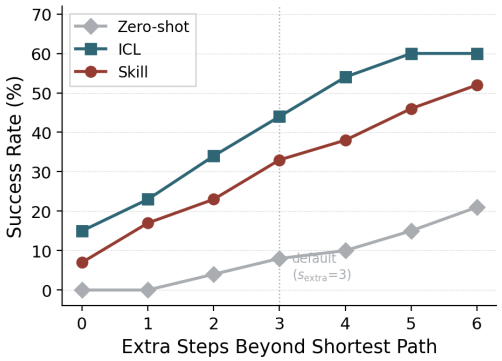


Figure 7: **Step budget sensitivity.** The x-axis shows the number of extra steps allowed beyond the shortest path. By default, agents are allowed 3 extra steps in the testing phase.

Results and Analysis. As shown in Figure 7, all three settings show a monotonically increasing success rate as the budget grows from 0 to 6 extra steps. This is expected, since a larger budget gives agents more room for suboptimal decisions. Across all budget values, the relative ranking remains consistent: ICL achieves the highest success rate, followed by Skill Learning and then zero-shot. This confirms that the advantage from experience is not specific to any particular budget. At the strictest budget (0 extra steps), zero-shot success rate is 0%, while ICL and Skill Learning still achieve 15.0% and 7.0%. Even under this most restrictive budget setting, learning from experience remains effective. Notably, continual learning methods exhibit a steeper increase in success rate as the budget grows compared to the zero-shot setting. This indicates that experienced agents benefit more from additional tolerance for suboptimal decisions. Meanwhile, the gap between experienced agents and zero-shot remains substantial even at the most generous budget (6 extra steps), confirming that additional tolerance cannot substitute for prior experience.

> ***Finding 5****: As the number of extra steps increases, success rate improves across all settings, and continual learning methods yield a steeper increase compared to the zero-shot setting.*

6 Conclusion

In this work, we introduced *EmbodiedCLBench*, the first benchmark in embodied simulation for evaluating the continual learning capability of self-evolving agents. With compositional generalization as the core evaluation approach, *EmbodiedCLBench* moves beyond existing benchmarks to measure continual learning in physically grounded embodied settings. Our experiments reveal consistent limitations across harnesses, pointing to several future directions: *(1)* closing the performance gap on advanced tasks, where experienced agents still fall well short of saturation; *(2)* designing skill representations that consistently outperform raw interaction trajectories, since current skill learning shows no clear advantage over in-context learning; and *(3)* enabling scalable continual learning where performance improves steadily with accumulated experience, rather than plateauing after a single sample. We hope that *EmbodiedCLBench* can serve as a diagnostic tool for future research and motivate the development of self-evolving agents with more robust and effective continual learning mechanisms in embodied simulation.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets, implementing methods, and assisting with translation. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, or interpreting results, and formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for creating or modifying scientific figures or images, editing a research paper to improve readability, and creating or editing software code. We have reviewed all AI-assisted work. Specifically:

1. During data construction, task descriptions and answer options in our benchmark were generated by AI based on our specifications. All AI-generated content was manually reviewed and verified by the authors.
2. AI-powered coding assistants were used during development. All code was verified and tested for correctness by 2 authors.
3. AI tools were used to polish the paper text and assist in creating figures. All polished content and figures were manually reviewed by the authors.

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

REPRODUCIBILITY STATEMENT

The benchmark construction process is described in Section 3 and Appendix B. The experimental settings are detailed in Sections 4 and 5. Based on these descriptions, reported results can be reproduced. Our code and data will be publicly available.

REFERENCES

Geo Ahn, Inwoong Lee, Taeoh Kim, Minho Shim, Dongyoon Wee, and Jinwoo Choi. Why Can't I Open My Drawer? Mitigating Object-Driven Shortcuts in Zero-Shot Compositional Action Recognition, July 2026. arXiv:2601.16211 [cs.CV].

Lisa Alazraki, Lihu Chen, Ana Brassard, Joe Stacey, Hossein A Rahmani, and Marek Rei. Agentcoma: A compositional benchmark mixing commonsense and mathematical reasoning in real-world scenarios. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8383–8410, 2026.

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 (CVPR)*, 2018.

Anthropic. Claude Code, 2026. URL `https://claude.com/product/claude-code`.

Yuxuan Cai, Yipeng Hao, Jie Zhou, Hang Yan, Zhikai Lei, Rui Zhen, Zhenhua Han, Yutao Yang, Junsong Li, Qianjun Pan, Tianyu Huai, Qin Chen, Xin Li, Kai Chen, Bo Zhang, Xipeng Qiu, and Liang He. Building Self-Evolving Agents via Experience-Driven Lifelong Learning: A Framework and Benchmark. *arXiv preprint arXiv:2508.19005*, 2025.

Yangtao Chen, Zixuan Chen, Nga Teng Chan, Junting Chen, Junhui Yin, Jieqi Shi, Yang Gao, Yong-Lu Li, and Jing Huo. RoboHiMan: A Hierarchical Evaluation Paradigm for Compositional Generalization in Long-Horizon Manipulation. *arXiv preprint arXiv:2510.13149*, 2025a.

Zixuan Chen, Junhui Yin, Yangtao Chen, Jing Huo, Pinzhuo Tian, Jieqi Shi, Yiwen Hou, Yinchuan Li, and Yang Gao. DeCo: Task Decomposition and Skill Composition for Zero-Shot Generalization in Long-Horizon 3D Manipulation. *arXiv preprint arXiv:2505.00527*, 2025b.

Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Guizilini, and Yue Wang. PhysBench: Benchmarking and Enhancing Vision-Language Models for Physical World Understanding. *arXiv preprint arXiv:2501.16411*, 2025.

Mengfei Du, Binhao Wu, Zejun Li, Xuanjing Huang, and Zhongyu Wei. EmbSpatial-Bench: Benchmarking Spatial Understanding for Embodied Tasks with Large Vision-Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 346–355. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-short.33.

Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, Zhaochun Ren, Nikos Aletras, Xi Wang, Han Zhou, and Zaiqiao Meng. A Comprehensive Survey of Self-Evolving AI Agents: A New Paradigm Bridging Foundation Models and Lifelong Agentic Systems. *arXiv preprint arXiv:2508.07407*, 2025.

Hiroki Furuta, Yutaka Matsuo, Aleksandra Faust, and Izzeddin Gur. Exposing Limitations of Language Model Agents in Sequential-Task Compositions on the Web. *arXiv preprint arXiv:2311.18751*, 2023.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Qihan Ren, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A Survey of Self-Evolving Agents: What, When, How, and Where to Evolve on the Path to Artificial Super Intelligence. *arXiv preprint arXiv:2507.21046*, 2025.

Xingze Gao, Chuanrui Hu, Hongda Chen, Pengfei Yao, Zhao Wang, Yi Bai, Zhengwei Wu, Yunyun Han, Xiaofeng Cong, Jie Gui, Yafeng Deng, and Teng Li. EvoAgentBench: Benchmarking Agent Self-Evolution via Ability Transfer. *arXiv preprint arXiv:2607.05202*, 2026.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills. In *International Conference on Learning Representations*, 2023.

Tianyi Guan, Yiding Wang, Haotong Yang, Siyuan Cao, Shirui Liu, Yi Hu, Jiaqi Li, and Muhan Zhang. ContinualSkillBench: Can LLM Agents Truly Evolve Their Capabilities? *arXiv preprint arXiv:2608.03874*, 2026.

Yuan Guo, Tingjia Miao, Zheng Wu, Pengzhou Cheng, Ming Zhou, and Zhuosheng Zhang. Atomic-to-Compositional Generalization for Mobile Agents with A New Benchmark and Scheduling System. *arXiv preprint arXiv:2506.08972*, 2025.

Sanjay Haresh, Daniel Dijkman, Apratim Bhattacharyya, and Roland Memisevic. ClevrSkills: Compositional Language And Visual Reasoning in Robotics. In *Advances in Neural Information Processing Systems*, volume 37, pp. 38235–38266, 2024. doi: 10.52202/079017-1209.

Xinyi He, Ying Yang, Chuanjian Fu, Sihan Guo, Songchun Zhu, Lifeng Fan, Zhenliang Zhang, and Yujia Peng. Automatic Cognitive Task Generation for In-Situ Evaluation of Embodied Agents. *arXiv preprint arXiv:2602.05249*, 2026.

Jie Huang, Ruixun Liu, Sirui Sun, Xinyi Yang, Yin Li, Yixin Zhu, and Yiwu Zhong. M^3eval: Multi-modal memory evaluation through cognitively-grounded video tasks. *arXiv preprint arXiv:2606.05008*, 2026.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The Robot Learning Benchmark & Learning Environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, April 2020. doi: 10.1109/lra.2020.2974707.

Sihang Jiang, Lipeng Ma, Zhonghua Hong, Keyi Wang, Zhiyu Lu, Tengfei Wang, Shisong Chen, Jinghao Zhang, Tianjun Pan, Weijia Li, Jiaqing Liang, and Yanghua Xiao. SEA-Eval: A Benchmark for Evaluating Self-Evolving Agents Beyond Episodic Assessment. *arXiv preprint arXiv:2604.08988*, 2026.

Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. GOAT-Bench: A Benchmark for Multi-Modal Lifelong Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16373–16383, 2024.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual Vision-and-Language Navigation with Dense Spatiotemporal Grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4392–4412. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.356.

Yingtie Lei, Zhongwei Wan, Jiankun Zhang, Samiul Alam, Zixuan Zhong, Peizhou Huang, Xin Wang, Jingxuan Zhang, Donghao Zhou, Yunta Hsieh, et al. Skillevolbench: Benchmarking the evolution from episodic experience to procedural skills. *arXiv preprint arXiv:2605.24117*, 2026.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. BEHAVIOR-1K: A Benchmark for Embodied AI with 1,000 Everyday Activities and Realistic Simulation. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 80–93. PMLR, 2023. Month: 14–18 Dec.

Rongchang Li, Zhenhua Feng, Tianyang Xu, Linze Li, Xiao-Jun Wu, Muhammad Awais, Sara Atito, and Josef Kittler. C2C: Component-to-Composition Learning for Zero-Shot Compositional Action Recognition. *arXiv preprint arXiv:2407.06113*, 2024.

Tianle Li, Jihai Zhang, Yongming Rao, and Yu Cheng. Unveiling the Compositional Ability Gap in Vision-Language Reasoning Model. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 126574–126592. Curran Associates, Inc., 2025. doi: 10.52202/085713-4222.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. *arXiv preprint arXiv:2306.03310*, 2023.

Huaping Liu, Di Guo, and Angelo Cangelosi. Embodied intelligence: A synergy of morphology, action, perception and learning. *ACM Computing Surveys*, 57(7):1–36, March 2025a. ISSN 1557-7341. doi: 10.1145/3717059.

Yang Liu, Weixing Chen, Yongjie Bai, Xiaodan Liang, Guanbin Li, Wen Gao, and Liang Lin. Aligning Cyber Space With Physical World: A Comprehensive Survey on Embodied AI. *IEEE/ASME Transactions on Mechatronics*, December 2025b. doi: 10.1109/tmech.2025.3574943.

Yuejiang Liu, Parth Kothari, Bastien Van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34:21808–21820, 2021.

Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, Mrinal Kalakrishnan, Franziska Meier, Chris Paxton, Alexander Sax, and Aravind Rajeswaran. OpenEQA: Embodied Question Answering in the Era of Foundation Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16488–16498, 2024.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, July 2022. doi: 10.1109/lra.2022.3180108.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-Scale Simulation of Everyday Tasks for Generalist Robots. *arXiv preprint arXiv:2406.02523*, 2024.

Nous Research. Hermes Agent, 2026. URL https://github.com/NousResearch/hermes-agent.

OpenAI. GPT-5.6 Terra Model | OpenAI API, 2026a. URL https://developers.openai.com/api/docs/models/gpt-5.6-terra.

OpenAI. OpenAI Codex CLI - Getting Started, 2026b. URL https://help.openai.com/en/articles/11096431.

OpenClaw. OpenClaw — open-source AI assistant, 2026. URL https://openclaw.ai/.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. TEACh: Task-driven Embodied Agents that Chat. *arXiv preprint arXiv:2110.00534*, 2021.

Yujia Peng, Jiaheng Han, Zhenliang Zhang, Lifeng Fan, Tengyu Liu, Siyuan Qi, Xue Feng, Yuxi Ma, Yizhou Wang, and Song-Chun Zhu. The tong test: Evaluating artificial general intelligence through dynamic embodied physical and social interactions. *Engineering*, 34:12–22, 2024.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: Remote Embodied Visual Referring Expression in Real Indoor Environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Qwen. Qwen3.8-Flash, 2026. URL https://www.qwencloud.com/models/qwen3.8-flash.

Jiawei Ren, Yan Zhuang, Xiaokang Ye, Lingjun Mao, Xuhong He, Jianzhi Shen, Mrinaal Dogra, Yiming Liang, Ruixuan Zhang, Tianai Yue, et al. Simworld: An open-ended realistic simulator for autonomous agents in physical and social worlds. *arXiv preprint arXiv:2512.01078*, 2025.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Yiheng Shu, Bernal Jiménez Gutiérrez, Saisri Padmaja Jonnalagedda, Yuguang Yao, Huan Sun, and Yu Su. AgentCL: Toward Rigorous Evaluation of Continual Learning in Language Agents. *arXiv preprint arXiv:2606.02461*, 2026.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Zhe Sun, Kunlun Wu, Chuanjian Fu, Zeming Song, Langyong Shi, Zihe Xue, Bohan Jing, Ying Yang, Xiaomeng Gao, Aijia Li, Tianyu Guo, Huiying Li, Xueyuan Yang, Rongkai Liu, Xinyi He, Yuxi Wang, Yue Li, Mingyuan Liu, Yujie Lu, Hongzhao Xie, Shiyun Zhao, Bo Dai, Wei Wang, Tao Yuan, Song-Chun Zhu, Yujia Peng, and Zhenliang Zhang. TongSIM: A General Platform for Simulating Intelligent Machines. *arXiv preprint arXiv:2512.20206*, 2025.

Andrea Tupini, Lars Liden, Reuben Tan, Yu Wang, and Jianfeng Gao. AsgardBench – Evaluating Visually Grounded Interactive Planning Under Minimal Feedback. *arXiv preprint arXiv:2603.15888*, 2026.

Tai Wang, Xiaohan Mao, Chenming Zhu, Runsen Xu, Ruiyuan Lyu, Peisen Li, Xiao Chen, Wenwei Zhang, Kai Chen, Tianfan Xue, Xihui Liu, Cewu Lu, Dahua Lin, and Jiangmiao Pang. EmbodiedScan: A Holistic Multi-Modal 3D Perception Suite Towards Embodied AI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19757–19767, 2024.

Weizhen Wang, Chenda Duan, Zhenghao Peng, Yuxin Liu, and Bolei Zhou. Embodied Scene Understanding for Vision Language Models via MetaVQA. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22453–22464, 2025.

Cheng-Kuang Wu, Zhi Rui Tam, Chieh-Yen Lin, Yun-Nung Chen, and Hung-yi Lee. StreamBench: Towards Benchmarking Continuous Improvement of Language Agents. *arXiv preprint arXiv:2406.08747*, 2024.

Zenan Wu, Bingqing Wei, Lu Liu, Zheqi He, Xi Wang, Jiakang Liu, Zehui Li, Guocai Yao, Jing-Shu Zheng, Xi Yang, and Yongtao Wang. ATOM-Bench: A Real-World Benchmark for Atomic Skills and Compositional Generalization in Manipulation Policies. *arXiv preprint arXiv:2606.16826*, 2026.

Binyan Xu, Xilin Dai, and Kehuan Zhang. Contextual agentic memory is a memo, not true memory. *arXiv preprint arXiv:2604.27707*, 2026.

Chen Yifan, Mingjie Wei, Xuesong Wang, Yuanxing Liu, Jizhe Wang, Hao Song, Longxuan Ma, Donglin Di, Churui Sun, Kaifeng Liu, Le Qi, Jianing Yu, Xiang Tian, Shiqi Liang, Chuanxi Duan, Zijian Hong, Weinan Zhang, and Ting Liu. Embodied AI: A Survey on the Evolution from Perceptive to Behavioral Intelligence. *SmartBot*, September 2025. doi: 10.1002/smb2.70003.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020. Month: 30 Oct–01 Nov.

Ziao Zhang, Kou Shi, Shiting Huang, Avery Nie, Yu Zeng, Yiming Zhao, Zhen Fang, Qishen Su, Haibo Qiu, Wei Yang, Qingnan Ren, Shun Zou, Wenxuan Huang, Lin Chen, Zehui Chen, and Feng Zhao. SkillFlow:Benchmarking Lifelong Skill Discovery and Evolution for Autonomous Agents. *arXiv preprint arXiv:2604.17308*, 2026.

Shiyun Zhao, Xinwei Song, Tianyu Guo, Xiaomeng Gao, Mingyuan Liu, Xu Han, Yuanyuan Zhang, Zhenliang Zhang, Xue Feng, and Bo Dai. NormAct: Benchmarking Embodied Agents' Proactive Compliance with Unspoken Social Norms. *arXiv preprint arXiv:2606.27826*, 2026.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, ZhongZhi Li, Yingying Zhang, Le Song, and Qianli Ma. LifelongAgentBench: Evaluating LLM Agents as Lifelong Learners. *arXiv preprint arXiv:2505.11942*, 2025a.

Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. Lifelong Learning of Large Language Model based Agents: A Roadmap. *arXiv preprint arXiv:2501.07278*, 2025b.

Shanshan Zhong, Yi Lu, Jingjie Ning, Yibing Wan, Lihan Feng, Yuyi Ao, Leonardo F. R. Ribeiro, Markus Dreyer, Sean Ammirati, and Chenyan Xiong. SkillLearnBench: Benchmarking Continual Learning Methods for Agent Skill Generation on Real-World Tasks. *arXiv preprint arXiv:2604.20087*, 2026.

A Basic Task Categories

As described in Section 3.2, basic tasks in *EmbodiedCLBench* span six capability categories. Table 2 lists each category and its core evaluation question.

Table 2: Six capability categories of basic tasks in *EmbodiedCLBench*.

Category	**Core Question**
Object Manipulation	Can the agent locate and manipulate everyday objects into a usable or manageable state?
Spatial Placement	Can the agent understand spatial relationships between objects and surfaces or containers?
Accessibility Control	Can the agent change the state of doors, drawers, and cabinet doors to enable or restrict access?
Appliance Operation	Can the agent toggle household appliances and environmental devices?
Activity-Driven Scene Configuration	Can the agent set up multiple objects and devices to support activities such as reading, dining, or leisure?
Household State Restoration	Can the agent restore a room to a tidy, energy-efficient, and reusable state?

B Graph Construction Pipeline

As described in Sections 3.2 and 3.4, *EmbodiedCLBench* builds tasks from atomic actions available in TongSim. Given the atomic actions for each task, we first construct the state-transition graph. We then generate two types of textual content: task descriptions and per-state action options.

LLM Generation. Both task descriptions and action options are produced by a three-stage pipeline powered by Qwen3.7-Plus. The pipeline serves two purposes: converting low-level atomic actions into natural household scenarios, and ensuring that action options do not give away the correct answer through shortcuts. The key prompt specifications for all three stages are provided in Section B.1.

Human Verification. Three annotators independently verify each generated sample. The verification covers three aspects: captured images, task descriptions, and per-state action options. For each aspect, annotators check for factual errors, logical inconsistencies, and whether the content reflects realistic household activities. A sample is retained only if all three annotators approve it.

B.1 PROMPT SPECIFICATIONS

Stage 1: Task Description Rewrite

```
# Stage 1: Task Description Rewrite

## Objective

Rewrite a baseline task description into a natural, scene-based household request. The
rewritten description is what the tested model actually sees. You do not change the
underlying task structure -- only the surface expression.

## Task Hierarchy

- L1 (atomic goal): the smallest unit of household goal, e.g., "turn off the TV", "open the
cabinet", "place the cup on the table".
- L2 (basic task): composed of 3 L1 goals forming a coherent household activity.
- L4 (advanced task): composed of 6 L1 goals combining multiple capability categories.

## Input Fields

- level: L2 or L4.
- ability_categories: capability categories involved (e.g., appliance operation, spatial
placement).
- l1_goals: list of constituent L1 atomic goals with descriptions, action types, and target
objects.
- baseline_task_description: a direct concatenation of all L1 goals (e.g., "Turn off the
TV, open the lamp, close the cabinet door").

## Rewrite Principles

1. Merge multiple L1 goals into a single coherent living scenario (e.g., "A guest is
arriving soon -- make the living room presentable and quiet"). The model should need to
reason about what to do from the scene image, not read the steps directly from the text.

2. Do NOT produce a checklist, enumerate target objects by name, expose execution order, or
translate L1 goals one-by-one with synonyms.

3. Replace specific object names with functional or contextual references (e.g., TV ->
"noise source"; table lamp -> "reading light"; cabinet -> "storage area").

4. Add a plausible living scenario (guest arriving, bedtime routine, post-cooking cleanup,
etc.) that makes all L1 goals co-exist naturally.

## Output

A single JSON object with fields: description_highlevel (the rewritten task description),
rationale (why the rewrite is natural and leak-free), and anti_leak_check (boolean flags
for exposes_order, is_checklist, mentions_specific_objects).
```

Stage 2: Per-State Option Rewrite

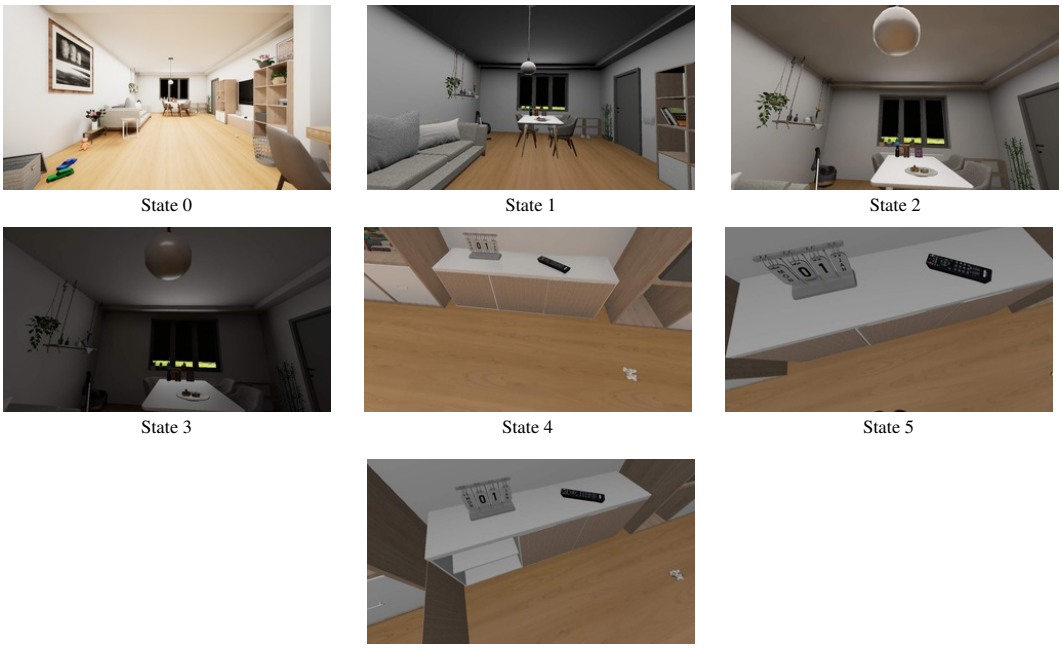
```
# Stage 2: Per-State Option Rewrite

## Objective

Rewrite the four hard-coded action options at a single graph state into natural language.
After rewriting, all four options should look like plausible next steps, forcing the model
to rely on the scene image and current state rather than linguistic cues.

## Underlying Execution Pattern (must be hidden)

Each household goal typically requires three phases:
1. Observe: look at the target object (look_at).
2. Approach: move to the target object (walk_to).
3. Operate: execute the action (open / close / switch_on / etc.).
If option wording reveals which phase it belongs to (e.g., observe options always say
"check the status of X"), the model can guess the answer without seeing the image. The core
challenge is to make all options sound like purposeful household actions regardless of
their underlying phase.

## Option Roles

- success_path: the optimal next action. Must not stand out among the four options.
- valid_detour: executable and task-relevant, but not optimal. Must read like active task
progress, not like a detour.
- invalid: cannot be executed in the current state. Must read like a plausible next step;
its appeal comes from semantic relevance to the task, not from low-quality wording.

## Rewrite Rules

1. Each option: 1 sentence, 10-25 English words, naming the involved object or area and
connecting to the high-level task scenario.
2. Two-layer fit: every option (including distractors) must fit both the current sub-goal
and the high-level task description.
3. Hide phases: do not let observe options read as "status checks", approach options as
"moving closer to operate", or operate options as the only ones with a sense of completion.
4. Uniform style: all four options should be similar in length, detail, and directness. The
correct option must not be the only one containing task keywords.
5. Invalid options must not contain words like "wrong", "invalid", or "not needed", and
must not explain why the action cannot be executed.

## Output

A single JSON object containing: state_id, options (each with id, text, role, design_note),
and self_check (correct_stands_out, current_goal_fit, highlevel_task_fit, phase_exposed,
has_implementation_terms).
```

Stage 3: Cross-State Option Hardening

```
# Stage 3: Cross-State Option Hardening

## Objective

Given a batch of states from the same task graph, detect and eliminate cross-state
linguistic patterns that the tested model could exploit. The model sees its full
interaction history, so any recurring pattern across consecutive steps can be discovered
and used to bypass visual reasoning.

## Context

Each household goal takes three phases (observe -> approach -> operate). A basic task has 3
such goals, producing a repeating 3-beat rhythm on the shortest path:
Goal 1: observe -> approach -> operate
Goal 2: observe -> approach -> operate
Goal 3: observe -> approach -> operate
Stage 2 made each state's options internally deceptive but could not address cross-state
patterns. This stage fixes that.

## Input

A batch of states (either the full shortest path or a branch cluster), each containing:
state_id, state_context, belongs_to_l1 (which sub-goal this state belongs to), and options
(with baseline_text, rewritten_text, role, action_type, object).

## Patterns to Detect and Eliminate

1. Three-beat rhythm: correct options (success_path) across consecutive states show a
cyclic "attention/focus -> head/move -> execute/switch" pattern that repeats per sub-goal.
2. Style bias: correct options are systematically shorter, more direct, or contain more
task keywords than distractors.
3. Expression repetition: the same phrasing or sentence structure appears across multiple
states.

## Hardening Strategies

A. Blur phase boundaries: make observe-phase correct options sound like preparation or
positioning rather than looking. Make approach-phase correct options sound like executing
part of an action rather than walking.
B. Mix phase styles in distractors: in an observe state, give some distractors an
operate-style wording; in an operate state, give some distractors an observe-style wording.
C. Break repetition across sub-goals: if sub-goal 1's correct options follow pattern X,
sub-goal 2's correct options must use a different pattern.
D. Vary detail level: some correct options should be short and direct, others should
include contextual detail, matching the variation in distractors.
E. Vary verbs and sentence structures: the same action type appearing in different states
must use different verbs and phrasings.

## Constraints

- Modify rewritten_text only (not state_id, id, role, action_type, or baseline_text).
- Edits are adjustments to Stage 2 output, not full rewrites from baseline_text.
- Options must remain specific, natural, and connected to the high-level task scenario.
- Do not use implementation terms (graph, state, precondition, phase).

## Output

A single JSON object containing: states (each with state_id, modified flag, options,
modification_note) and pattern_report (documenting which patterns were found and how they
were fixed).
```

C TASK EXAMPLES

We provide task examples at both compositional levels to illustrate the evaluation format. The agent first receives a natural-language task description. At each step, it observes the current-state image and selects one action from multiple candidates. Specifically, we include two basic-task examples (Figures 8 and 9; Tables 3 and 4) and one advanced-task example (Figure 10; Tables 5 and 6).



State 0 State 1 State 2

State 3 State 4 State 5

State 6

Figure 8: Visual states along the shortest path for a basic-task example.

Table 3: Basic-task example: action options and hidden graph roles at each state.

Task: I'm about to watch a movie in the living room. Turn off the overhead light so the screen isn't washed out, and open up the media cabinet so everything's within reach.

State	Option	Text shown to the agent	Hidden role
State 0	A	Head over to the ceiling light area to get ready to turn it off.	Invalid in current state
State 0	B	Switch on the overhead light to brighten the living room before the movie.	Invalid in current state
State 0	C	Shut off the ceiling light right away to set the mood for the movie.	Invalid in current state
State 0	D	The overhead light is still on — that's the first thing to deal with before the movie.	Correct shortest-path action
State 1	A	Check the ceiling light to see if it's already been switched off.	Invalid in current state
State 1	B	Switch off the ceiling light to darken the room for movie viewing.	Invalid in current state
State 1	C	Turn on the ceiling light to brighten the space before settling in for the movie.	Invalid in current state
State 1	D	Walk over to the ceiling light so you can reach the switch.	Correct shortest-path action
State 2	A	Switch off the overhead light — the room needs to be dark for the movie.	Correct shortest-path action
State 2	B	Step closer to the ceiling fixture to get a better grip on the switch.	Executable detour
State 2	C	Glance up at the ceiling light to confirm how bright it still is.	Invalid in current state
State 2	D	Check the television area to see if the media setup is ready.	Executable detour
State 3	A	Head over to the television area to get the media setup going.	Invalid in current state
State 3	B	Walk toward the ceiling light to start working on the room's ambiance.	Executable detour
State 3	C	Switch on the overhead light to brighten things up before the movie.	Executable detour
State 3	D	With the light off, the media cabinet by the TV is the next thing to handle.	Correct shortest-path action
State 4	A	Power down the television to eliminate any screen glare during the movie.	Invalid in current state
State 4	B	Walk up to the TV unit — the cabinet is right ahead.	Correct shortest-path action
State 4	C	Focus on the television to assess whether the screen needs attention.	Invalid in current state
State 4	D	Switch on the television so you can test the viewing angle for the movie.	Invalid in current state
State 5	A	Shift your focus to the ceiling light to confirm it's off for movie time.	Executable detour
State 5	B	Open the media cabinet so everything is within reach for the movie.	Correct shortest-path action
State 5	C	Take a closer look at the television area to plan the next move for the media setup.	Invalid in current state
State 5	D	Step over to the television unit to get a better position for the entertainment area.	Executable detour

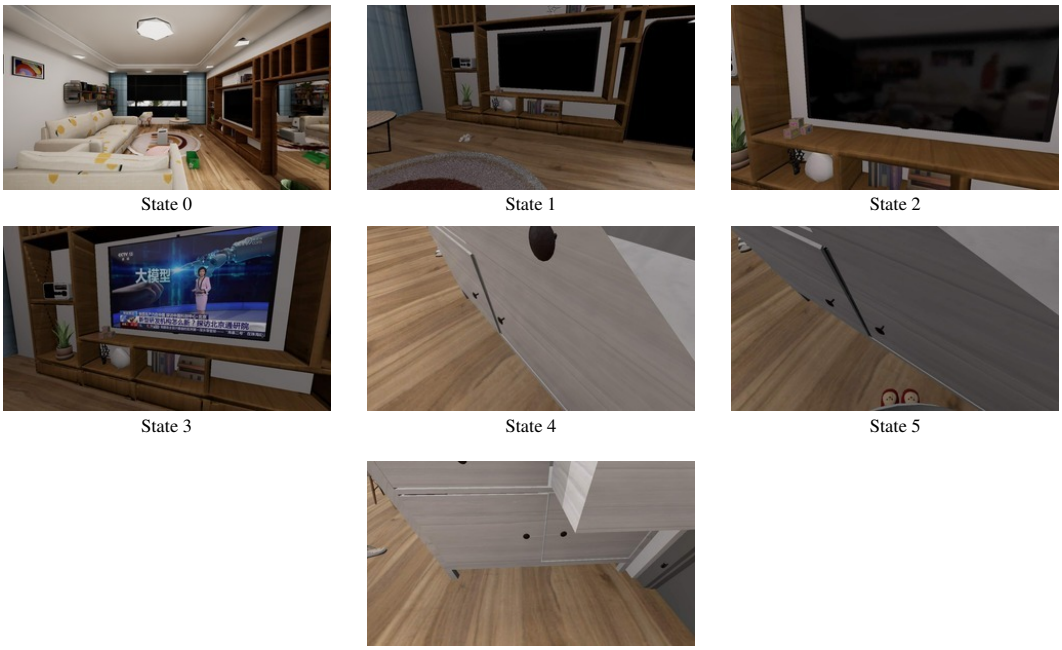


Figure 9: Visual states along the shortest path for a basic-task example.

Table 4: Basic-task example: action options and hidden graph roles at each state.

Task: Get the living room ready for a relaxed evening — turn on some background entertainment and open up the storage cabinet so everything's within reach.

State	Option	Text shown to the agent	Hidden role
State 0	A	Start by checking the TV area — that's where the background entertainment will come from.	Correct shortest-path action
State 0	B	Shut off the TV screen since it's not needed for the living room ambiance.	Invalid in current state
State 0	C	Head over to the TV wall to get closer to the entertainment controls.	Invalid in current state
State 0	D	Pull open the storage cabinet to make everything accessible for the evening.	Invalid in current state
State 1	A	Shut the storage cabinet to keep the living room looking tidy.	Invalid in current state
State 1	B	Shift your focus to the cabinet area near the TV to assess what's needed.	Invalid in current state
State 1	C	Get to the entertainment setup — the background noise needs to start soon.	Correct shortest-path action
State 1	D	Pull open the storage compartment so things can be reached easily for the evening.	Invalid in current state
State 2	A	Switch on the screen to bring some background entertainment into the living room.	Correct shortest-path action
State 2	B	Head over to the screen area to get a better position for the setup.	Executable detour
State 2	C	Focus on the storage cabinet to make sure it's ready for easy access.	Executable detour
State 2	D	Check the screen to see if it's already running before proceeding.	Invalid in current state
State 3	A	The storage cabinet still needs opening — that's the next thing to handle.	Correct shortest-path action
State 3	B	Head over to the storage unit to get it ready for the evening.	Invalid in current state
State 3	C	Switch off the ambient background element near the TV to set a cleaner mood.	Executable detour
State 3	D	Walk toward the entertainment area to adjust the atmosphere.	Executable detour
State 4	A	Shut the storage cabinet to keep the living room looking tidy.	Invalid in current state
State 4	B	Check the cabinet area to see what's currently inside.	Invalid in current state
State 4	C	Pull open the storage cabinet to make everything accessible for the evening.	Invalid in current state
State 4	D	Move into position near the storage cabinet — it needs to be opened.	Correct shortest-path action
State 5	A	Step closer to the storage cabinet to get a better grip on the handle.	Executable detour
State 5	B	Scan the cabinet area to confirm everything is in order.	Invalid in current state
State 5	C	Glance toward the TV wall to take in the overall setup before finishing up.	Executable detour
State 5	D	Open up the storage cabinet so everything is within reach for the evening.	Correct shortest-path action

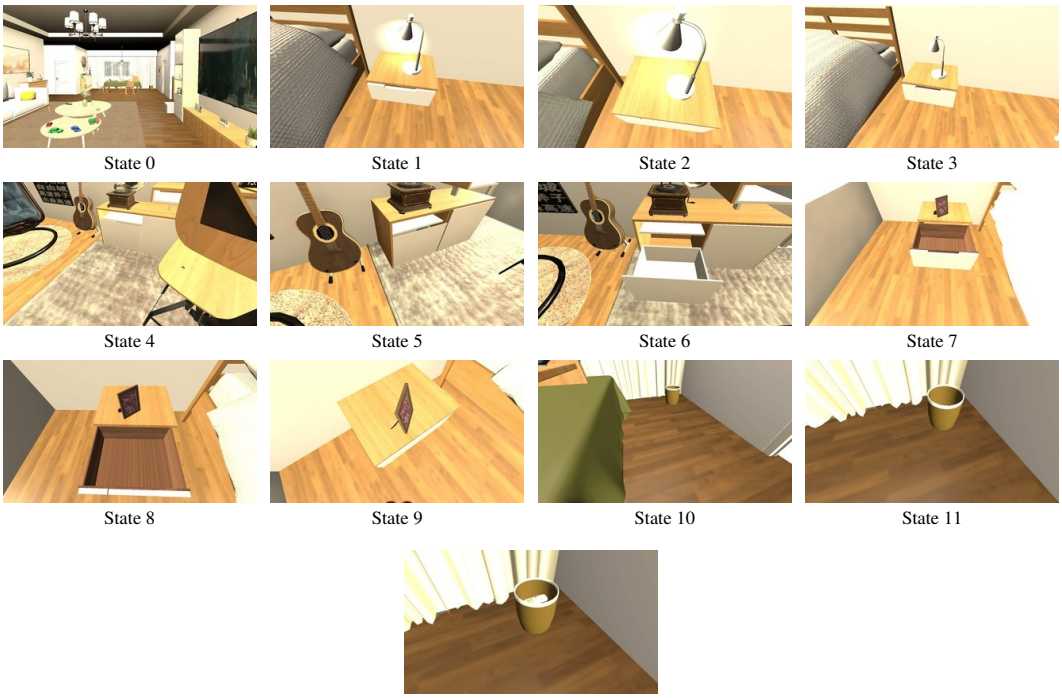


Figure 10: Visual states along the shortest path for an advanced-task example, including the final state.

Table 5: Advanced-task example: action options and hidden graph roles.

Task: I'm done for the night and heading to bed — dim the bedroom lamp, open up the study desk so it's ready for tomorrow, shut the bedside drawer, and toss the old remote into the bin.

State	Option	Text shown to the agent	Hidden role
State 0	A	Check the trash can area to see what needs to be tossed before bed.	Executable detour
State 0	B	Switch on the table lamp to brighten the bedside area for cleanup.	Invalid in current state
State 0	C	The lamp's still glowing — that's the first thing to sort out before the room can rest.	Correct shortest-path action
State 0	D	Head over to the table lamp to adjust the lighting for bedtime.	Invalid in current state
State 1	A	Switch off the table lamp to dim the room for bedtime.	Invalid in current state
State 1	B	Get closer to the lamp — you'll need to be right there to kill the light.	Correct shortest-path action
State 1	C	Brighten the table lamp so you can see what's left to tidy.	Invalid in current state
State 1	D	Focus on the table lamp area to assess what needs adjusting for the night.	Invalid in current state
State 2	A	Brighten the lamp a bit so you can see what's left on the nightstand before closing it up.	Invalid in current state
State 2	B	Kill the lamp light to start dimming the room for the night.	Correct shortest-path action
State 2	C	Shift over to the lamp side to get the lighting sorted before settling in.	Invalid in current state
State 2	D	Focus on the lamp area to assess what still needs adjusting before bed.	Invalid in current state
State 3	A	Glance over at the trash can to see what needs tossing before bed.	Executable detour
State 3	B	The study desk still has stuff on it — that's the next area to sort out.	Correct shortest-path action
State 3	C	Head over to the study desk to get in position for clearing it off.	Invalid in current state
State 3	D	Flip open the study desk to start organizing items for tomorrow's setup.	Invalid in current state
State 4	A	Glance over at the desk to see what's still out on the surface.	Invalid in current state
State 4	B	Flip open the desk to start organizing what's inside for tomorrow.	Invalid in current state
State 4	C	Shut the desk drawers to keep the room tidy for the night.	Invalid in current state
State 4	D	Step up to the desk so you can reach across and clear it off properly.	Correct shortest-path action
State 5	A	Shift your focus to the desk to assess what still needs clearing.	Invalid in current state
State 5	B	Open the desk cabinet so you can stow tonight's items inside.	Correct shortest-path action
State 5	C	Glance over at the table lamp to check whether the room is dim enough.	Executable detour
State 5	D	Take a moment to review the bedside cabinet before closing it up for the night.	Executable detour

Table 6: Advanced-task example (continued): action options and hidden graph roles.

Continuation of Table 5; the same task description applies.

State	Option	Text shown to the agent	Hidden role
State 6	A	Shift your focus to the table lamp to see if it still needs attention for the night.	Executable detour
State 6	B	Glance over at the trash can to check if anything else needs tossing before bed.	Executable detour
State 6	C	Take a look at the remote control in your hand to decide where it should go next.	Invalid in current state
State 6	D	The bedroom nightstand is still open — head back and deal with that next.	Correct shortest-path action
State 7	A	Shut the bedside cabinet to keep the room tidy for the night.	Invalid in current state
State 7	B	Pull the bedside cabinet open to check if anything needs to be put away.	Invalid in current state
State 7	C	Glance over at the bedside cabinet to see what's still out.	Invalid in current state
State 7	D	Move in close to the nightstand — the cabinet needs shutting and you need to be right there.	Correct shortest-path action
State 8	A	Shift focus to the desk area to see what needs clearing for tomorrow.	Executable detour
State 8	B	Shut the bedside cabinet to keep the nightstand tidy before sleep.	Correct shortest-path action
State 8	C	Check the table lamp status to see if it needs dimming for bed.	Executable detour
State 8	D	Glance at the bedside cabinet to confirm what's inside before closing.	Invalid in current state
State 9	A	Shift focus to the table lamp area to continue dimming the room for rest.	Executable detour
State 9	B	The remote control still needs to go — spot the trash can so you know where it's headed.	Correct shortest-path action
State 9	C	Glance over at the desk surface to assess what else still needs to be cleared away.	Executable detour
State 9	D	Check the remote control in your hand to decide whether it should be put away.	Invalid in current state
State 10	A	Seal up the trash can so nothing spills out overnight.	Invalid in current state
State 10	B	Glance at the trash can to make sure it's ready to receive items.	Invalid in current state
State 10	C	Pop open the trash can lid so you can drop things in.	Invalid in current state
State 10	D	Get to the trash can — the remote control in your hand needs to go there.	Correct shortest-path action
State 11	A	Set the remote control down near the trash can to keep the nightstand clear.	Correct shortest-path action
State 11	B	Check the bedside cabinet to make sure it's properly closed for the night.	Executable detour
State 11	C	Take a closer look at the remote control to decide where it should go.	Invalid in current state
State 11	D	Turn your attention to the desk area to see what still needs clearing.	Executable detour