Unleashing Interactive Agent Planning Ability in LLMs via Multi-Trajectory Reinforcement Learning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities across diverse domains, yet they struggle with agent task planning in dynamic environments requiring continuous observation and sequential decision-making. Current methods generate static action sequences from pre-trained knowledge without learning from environmental feedback, limiting their effectiveness in partially observable settings. We present **Interactive Planner-R1**, a novel trajectory-level reinforcement learning framework that enables LLMs to develop interactive planning capabilities through autonomous environmental exploration. Our approach addresses three key challenges: (1) limited exploration diversity by introducing multi-trajectory autonomous exploration through parallel group rollouts, (2) sparse reward signals by developing a completion-driven reward architecture that promotes genuine environmental understanding, and (3) single-step optimization constraints by proposing Interactive Policy Optimization (IPO) that extends group-relative policy optimization for multi-step trajectory learning. Extensive experiments on ALFWorld and ScienceWorld demonstrate that Interactive Planner-R1 achieves substantial improvements over existing approaches, reaching 97.55% completion rate on ALFWorld and 79.92% on ScienceWorld, with strong generalization exhibiting only 3.33% performance gap in unseen environments. Our work establishes a new paradigm for LLM-based interactive planning through trajectory-level policy learning.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of tasks, from open-domain dialogue to complex reasoning (Brown et al., 2020; OpenAI, 2023; DeepSeek-AI et al., 2025; Shao et al., 2024; Team et al., 2024; Yang et al., 2024a). However, translating these capabilities into effective planning in interactive environments remains a fundamental challenge. Planning agents must continually perceive, reason about, and act upon their environment, forming a tightly coupled decision-making loop. Current LLMs still lack the situational and interactive awareness required for robust planning. In contrast, humans actively probe their surroundings, learning how actions transform the environment and refining both perception and world understanding through exploration (Gibson, 1979). While intuitive for humans, such exploration-driven learning remains challenging for artificial intelligence—especially when applying LLMs to interactive planning domains, such as robotics and embodied agents.

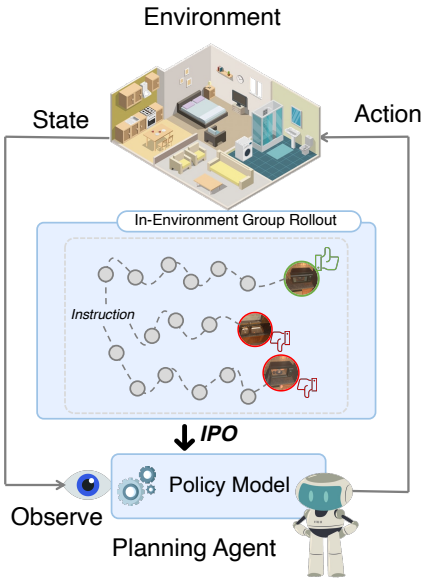


Figure 1: Overview of Interactive Planner-R1, which encourages agents to autonomously explore environments and trains them with binary rewards based on task completion status.

Traditional LLM-based planners treat the model as a static decision-maker (Chen et al., 2023b; Min et al., 2022), generating fixed action sequences from pre-trained knowledge and ignoring real-time

feedback. While Yao et al. (2023) improved task performance by introducing reasoning processes into planning, these methods still primarily perform offline rollouts based on prior knowledge, failing to establish crucial causal links between actions and environmental feedback during interaction. This limitation is particularly problematic in partially observable or unfamiliar environments, where agents must continually update their world models to succeed. Recent work (Song et al., 2024; Wang et al., 2024b; Chen et al., 2023a; Zeng et al., 2024a) has attempted to bridge this gap by incorporating human-curated demonstrations or handcrafted knowledge. However, heavy reliance on human priors restricts scalability and limits the overall capability of LLMs. This raises a fundamental question: Can we instead encourage autonomous exploration with minimal supervision, enabling LLMs to acquire robust interaction and generalization abilities?

Inspired by human cognition, we introduce **Interactive Planner-R1**, a reinforcement learning framework that enables LLMs to learn interactive planning through environmental exploration, requiring only minimal task-specific supervision.

Extensive experiments demonstrate the effectiveness of Interactive Planner-R1, achieving 97.55% and 79.92% cumulative reward on the ALFWorld and ScienceWorld benchmarks, respectively, significantly outperforming existing approaches. These results validate the framework's capability to foster genuine environmental understanding and adaptive behavior in interactive planning tasks.

In summary, our main contributions are as follows:

- We introduce Interactive Planner-R1, a novel trajectory-level reinforcement learning framework that enables LLMs to learn interactive planning through autonomous environmental exploration with minimal supervision.
- We propose three key technical innovations: (1) parallel group rollouts for diverse trajectory collection, (2) completion-driven sparse rewards that prevent reward hacking while encouraging genuine environmental understanding, and (3) Interactive Policy Optimization (IPO) that extends group-relative optimization to multi-step sequential planning.
- We achieve state-of-the-art performance on ALFWorld (97.55%) and ScienceWorld (79.92%) with superior generalization (only 3.33% performance gap on unseen environments), demonstrating the effectiveness of autonomous exploration over human-supervised approaches.

2 Preliminaries

Following established researches (Qiao et al., 2024), we formalize embodied planning tasks within the framework of a Partially Observable Markov Decision Process (POMDP) due to the agent's inability to directly observe the environment's complete state. Within this framework, the agent interacts within a partially observable environment and develops plans based on environmental feedback. The POMDP is defined by a 7-tuple:

$$(S, A, \Psi, \mathcal{M}, O, R, \gamma), \tag{1}$$

where S represents the state space defined by the environment, A denotes the action space available to the agent, O constitutes the observation space of the agent, $\Psi : S \times A \to S$ is the state transition function determining how actions will have effect on environmental states, $\mathcal{M} : S \to O$ is the observation function of the agent. Due to partial observability constraints, the agent cannot directly access states $s \in S$, but rather obtains partial observations of the current environment via the observation function, such that $o = \mathcal{M}(s)$, where $o \in O$ and $s \in S$. Additionally, R represents the reward function, and γ is the discount factor.

In the context of planning tasks, the agent receives a task q, typically expressed as a natural language instruction. The agent's objective is to complete this instruction through environmental interaction. During this interaction process, the agent initially obtains an observation o_0 from the environment and initializes a trajectory $\tau_0 = (q, o_0)$. At each time step t, the policy model π_θ generates the current action based on the historical trajectory:

$$a_t = \pi_\theta(\tau_t) = \pi_\theta(q, o_0, a_0, o_1, ..., a_{t-1}, o_t). \tag{2}$$

When performed, the action will lead to a change to the environment, resulting in a new state $s_{t+1} = \Psi(s_t, a_t)$, Following this transition, the agent receives a new partial observation $o_{t+1} = \mathcal{M}(s_{t+1})$. Subsequently, the trajectory is updated to $\tau_{t+1} = (q, o_0, a_0, o_1, ..., a_t, o_{t+1})$, incorporating all past actions and observations with the new observation. The reward for the agent is defined with a binary signal, where the agent receives a reward of 1 if the

new state s_{t+1} meets the predefined goal conditions specified by q, otherwise the reward is 0.

3 Interactive Planner-R1

3.1 Overview

In this section, we introduce **Interactive Planner-R1**, an end-to-end training framework designed for long-term planning. Interactive Planner-R1 learns solely from outcome-based reward signals without requiring additional human supervision. Our key insight is to unleash the latent reasoning and planning capabilities of large language models (LLMs) through direct interaction with the environment, allowing them to self-evolve via reinforcement learning. To better model the interaction process, we formulate it as **ReAct-style trajectories**, where progress is made through continuous '$Observation \rightarrow Think \rightarrow Action$' cycles until task completion. Interactive Planner-R1 consists of three tightly integrated core components: Firstly, to address the problem of sparse rewards, we develop a parallel sampling strategy that simultaneously generates multiple trajectories forming trajectory groups to establish stable statistical baselines, which we call **Group Rollout with In-Environment Interaction** . Secondly, we design a pure **Completion-based Sparse Reward** structure that encourages development of genuine environmental understanding through autonomous interaction. Finally, and most importantly, we introduce the **Interactive Policy Optimization** (IPO) algorithm, which is specifically designed for ReAct-style multi-turn interactions as its POMDP-oriented extension.

3.2 ReAct Paradigm

As described above, at time step t, the agent generates an action based on a trajectory that includes environmental observations and previous actions. However, mapping directly from such a trajectory to action demands complex reasoning capabilities, like task goal decomposition, common sense knowledge application, and extraction of key information from observations is exceptionally challenging. Following Yao et al. (2023), we adopt the ReAct paradigm that introduces the thought process (Wei et al., 2022) into LLM-based planning trajectories, as demonstrated in Figure 3. Unlike traditional approaches, with the historical trajectory as input, the agent generates a thought context:

$$\tau_{t+1} = (q, o_0, m_0, o_1, ..., m_t, o_{t+1}), \quad (3)$$

where m_t denotes the model's response for t-th step, it consists of thoughts and a text-form action:

$$m_t = (\phi_t, a_t). \quad (4)$$

We extract action a_t from m_t, which is then used to interact with the environment to obtain the new observation o_{t+1}.

3.3 Group Rollout with In-Environment Interaction

To effectively train an LLM for solving interactive environment tasks, we introduce a group rollout procedure that improves exploration efficiency while preserving a simple and well-defined interaction protocol.

Episode-level Parallel Rollout. Our group rollout mechanism should be distinguished from step-level branching or tree-search-based rollouts. Instead of expanding multiple actions from a shared intermediate environment state at each timestep, we perform episode-level parallel rollouts. Specifically, for each sampled task–environment pair, we initialize n independent and identical environment replicas at the beginning of an episode. Each replica maintains its own environment state, observation history, and trajectory buffer throughout the rollout.

A single frozen policy $\pi_{\theta_{\text{old}}}$ is used as the decision-maker for all replicas. At timestep t, the action for replica i is sampled from $\pi_{\theta_{\text{old}}}(\cdot \mid h_i^t)$, where h_i^t denotes the unique history of that replica up to step t. Due to policy stochasticity and environment dynamics, trajectories naturally diverge over time. This procedure constitutes a set of independent episode-level simulations rather than a branching rollout from shared intermediate states.

Parallel Rollout Procedure. At each training iteration, we sample K task–environment pairs from the distribution $\mathcal{D}$ and create n parallel replicas for each pair. Each replica is initialized with a reset environment and an empty trajectory buffer. For up to *max_steps* interaction steps, the frozen policy $\pi_{\theta_{\text{old}}}$ generates a ReAct-style response conditioned on the task instruction and the current trajectory history. From each response, we parse an executable action.

To allow explicit termination, we introduce a special done action. The policy may generate done when it determines that the task has been completed or is unsolvable. If done is generated before the

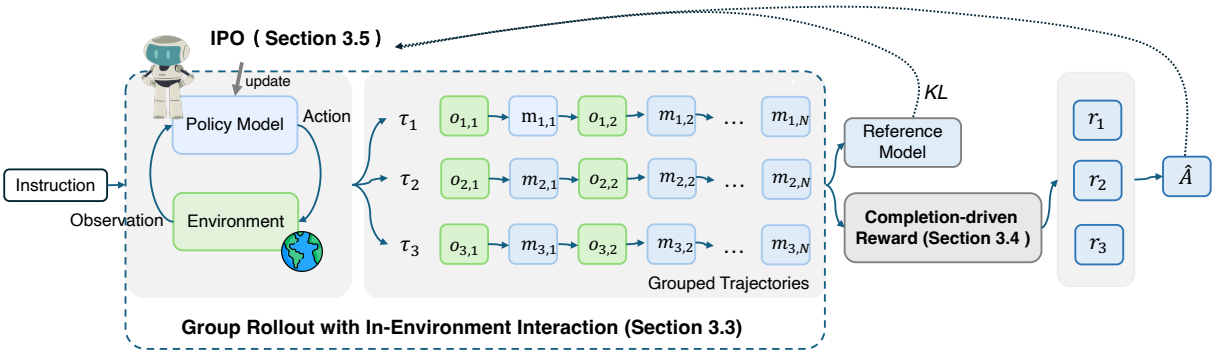


Figure 2: The Interactive Planner-R1 consists of three tightly integrated core components: 1) Group Rollout with Environment Interaction, a parallel sampling strategy that simultaneously generates multiple trajectories to form trajectory groups; 2) Completion-driven Sparse Reward; 3) Interactive Policy Optimization algorithm that redesigns probability ratios and advantage allocation, and updates the policy function based on group relative rewards. The KL divergence between policy and reference model serves as a regularization term to prevent over-optimization.

Trajectory with ReAct paradigm

System:
You are an intelligent agent in a household environment…
User:
Your task is to: put some pencil on dresser.
Observation:
You are in the middle of a room. Looking quickly around you, you see a armchair 1…
Assistant:
Thought: The pencil could be in any of the drawers or shelves. I will start by checking the drawers.
Action: go to drawer 1 a_0
Observation:
<Observation from Environment >
Assistant:
Thought: The drawer is closed, I need to open it to check if there is a pencil inside.
Action: open drawer 1 a_1
…
Assistant:
Thought: I have put the pencil on the dresser, task have complete.
Action: done a_t

Figure 3: We employ the ReAct paradigm during generation and extract the action from the response m_t.

environment confirms task completion, the corresponding trajectory receives zero reward.

The rollout for a task terminates when all replicas either emit done or reach the maximum step limit. All collected trajectories are then aggregated into a global buffer $\mathcal{T}$, from which trajectory-level returns are computed and used to update the policy according to Eqs. (7)–(8). We provide the algorithm pseudo code in Appendix B and an illustrative example of the parallel rollout process is provided in Appendix C.

3.4 Completion-driven Reward

To encourage autonomous exploration, we use only a rule-based completion reward without further restriction to optimize the model:

$$r = \begin{cases} 0 & \text{if the task is incomplete,} \\ 1 & \text{if the task is completed.} \end{cases} \tag{5}$$

Unlike previous approaches that introduced format rewards or length penalty, we do not explicitly provide such constraints. On the one hand, we found that LLMs can maintain a clear response structure with guidance from the system prompt, and only a parsable and valid response can interact with the environment to complete the task. On the other hand, we argue that reducing human priors in reward design can lower the risk of reward hacking and encourage the model to explore autonomously, generating more flexible and diverse trajectories.

3.5 Interactive Policy Optimization (IPO)

To enable efficient environment interaction and planning while substantially reducing training cost, we introduce Interactive Policy Optimization (IPO). IPO relies on the group rollout with In-Environment Interaction to make the policy model π_θ generate multiple trajectories. We employ the group-normalized advantage estimator, which has no learnable critic model, but its probability ratio is calculated over the designated trajectories tokens prefix up to step t, including all preceding thought ϕ and action a tokens:

$$Pr_t(\theta) = \frac{\pi_\theta\left(\phi_t, a_t \mid \tau_{i,<t}\right)}{\pi_{\theta_{\text{old}}}\left(\phi_t, a_t \mid \tau_{i,<t}\right)}. \tag{6}$$

We optimize the policy model by maximizing

the following objective function:

$$\mathcal{J}_{\text{IPO}}(\theta) = \mathbb{E}_{(q,e)\sim\mathcal{D},\{\tau_i\}_{i=1}^{n}\sim\pi_{\theta_{old}}(\mathcal{T}|(q,e))}$$
$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\tau_i|}\sum_{t=1}^{|\tau_i|}\left\{\min\left[Pr_t(\theta)\hat{A}_{i,t}, \text{clip}\left(Pr_t(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\right] - \beta\mathbb{D}_{KL}\left[\pi_\theta||\pi_{\text{ref}}\right]\right\}, \tag{7}$$

where $\hat{A}_{i,t}$ is the advantage of i-th trajectory in the group for time step t, which is calculated as follows:

$$\hat{A}_{i,t} = \frac{r_i - \mu_r}{\sigma_r}, \tag{8}$$

where t represents the t-th step of the trajectories, μ_r is the mean reward of the group, and σ_r is the reward deviation of the whole group. Additionally, the objective of IPO integrates a KL divergence penalty term between the training policy and the reference policy to prevent the model from deviating too much (Eq.7).

4 Experiments

4.1 Implementation

Evaluation To assess model capabilities in interactive planning tasks, we conducted systematic evaluations using two text-based embodied world simulator benchmarks: ALFWorld (Shridhar et al., 2021) and ScienceWorld (Wang et al., 2022). ALFWorld encompasses six categories of planning tasks set primarily in home environments, covering not only basic object manipulation (such as "pick and place") but also tasks requiring complex interaction sequences. ScienceWorld presents a more challenging benchmark, requiring models to complete scientific experiments in a highly interactive environment. Each task was evaluated only once, with interaction steps limited to 30 to ensure evaluation efficiency and consistency. More evaluation and benchmark details will introduced Appendix D.1.

Baseline We compared our Interactive Planner-R1 against three state-of-the-art training-based methods: SFT (Zeng et al., 2024a), NAT (Wang et al., 2024b), and ETO (Song et al., 2024). We also evaluated larger foundation models including GPT-3.5 Turbo (Brown et al., 2020)(gpt-3.5-turbo-0125), GPT-4 (OpenAI, 2023)(gpt-4o-2024-08-06), DeepSeek-V3 (DeepSeek-AI et al., 2024)(DeepSeek-V3-0324), and InternLM 2.5 20B (Team et al., 2024). All training-based methods are fine-tuned on Qwen2.5-7B-Instruct (Yang et al., 2024a). Detailed baseline descriptions are provided in Appendix D.2.

Training Details We implemented our approach using the verl library and trained on 8 NVIDIA A100 80GB GPUs. Each training step sampled 128 tasks with 5 trajectories per task. Additional implementation details are provided in Appendix D.3.

4.2 Main results

As shown in Table 1, we conducted comprehensive evaluations of prompt-based and training-based approaches on two challenging interactive environments, ALFWorld and ScienceWorld. The experimental results reveal several key findings:

First, current state-of-the-art foundational LLMs demonstrate notable limitations in interactive planning capabilities. While GPT-4 achieved the best performance (55.41% on ALFWorld and 58.18% on ScienceWorld), it still lags behind training-based approaches , indicating the challenges large language models face in environmental interaction tasks.

Compares with other baseline, Interactive Planner-R1 demonstrates consistent improvements across all evaluation metrics. On the ALFWorld benchmark, it achieved accuracy rates of 98.57% and 96.52% in seen and unseen scenarios, respectively, surpassing the ETO by 14.28% and 16.67%. On the ScienceWorld dataset, Interactive Planner-R1 achieved accuracy rates of 83.66% and 76.18% in seen and unseen scenarios, outperforming competitive methods by 28.58% and 18.86%, respectively. The performance improvement from the Qwen2.5-7B-Instruct (averaging 31.05% on ALFWorld and 22.05% on ScienceWorld) to Interactive Planner-R1 demonstrates the effectiveness of our reinforcement learning framework.

We evaluate three trajectory-based baselines in ALFWorld that leverage human prior knowledge to construct training datasets and observe their performance clustering around 80%, suggesting a performance ceiling imposed by the reliance on human-annotated data. In contrast, Interactive Planner-R1 easily surpasses this ceiling, achieving over 95% by leveraging reinforcement learning.

4.3 Ablation Study

To comprehensively understand the contribution of each component in Interactive Planner-R1, we conduct systematic ablation studies across three key dimensions: policy learning architecture, ReAct paradigm integration, group rollout mechanism. All ablation experiments are performed on ALFWorld to ensure consistent evaluation conditions.

Method	ALFWorld			ScienceWorld		
	Seen	Unseen	Avg.	Seen	Unseen	Avg.
GPT-3.5-Turbo (Brown et al., 2020)	7.14	7.46	7.30	28.03	21.63	24.83
GPT-4o (OpenAI, 2023)	58.57	52.24	55.41	59.6	56.75	58.18
Deepseek-v3 (DeepSeek-AI et al., 2024)	36.43	31.34	33.89	29.28	27.45	28.37
InternLM 2.5 20B Chat (Team et al., 2024)	7.86	11.94	9.90	33.66	31.15	32.41
Qwen2.5-72B-Instruct (Yang et al., 2024a)	25.71	26.12	25.92	29.38	27.96	28.67
Qwen2.5-7B-Instruct (Yang et al., 2024a)	30.00	32.09	31.05	28.01	16.09	22.05
Qwen2.5-7B-Instruct + SFT(Zeng et al., 2024a)	80.71	79.10	79.91	68.81	55.7	62.26
Qwen2.5-7B-Instruct + NAT(Wang et al., 2024b)	63.57	66.42	65.00	58.56	49.75	54.16
Qwen2.5-7B-Instruct + ETO(Song et al., 2024)	84.29	79.85	82.07	55.08	57.32	56.20
Qwen2.5-7B-Instruct + Interactive Planner-R1(Ours)	**98.57** ± 0.7	**96.52** ± 1.9	**97.55**	**83.66** ± 0.5	**76.18** ± 0.9	**79.92**

Table 1: **Main Results.** The best results are marked in **bold**. We compare our Interactive Planner-R1 with representative baselines including vanilla language models and training-based methods on ALFWorld and ScienceWorld benchmarks, evaluating performance on both seen and unseen task scenarios. See Table 5 for complete comparison with additional baselines.

	Seen	Unseen	Avg.	Δ
Interactive Planner-R1	98.57	96.52	97.55	0
w/o React	96.56	95.71	96.14	-1.41
w/o IPO	21.43	26.11	23.77	-73.78
w/o group		N/A		-

Note: 'w/o group' ($k = 1$) is not applicable to IPO, as within-group normalization collapses and the advantage estimator degenerates, resulting in non-learning behavior. We therefore report it as N/A.

Table 2: Ablation study on ALFWorld. Δ denotes the performance difference compared to the full model.

As shown in Table 2, the experimental results reveal several critical insights about our framework design:

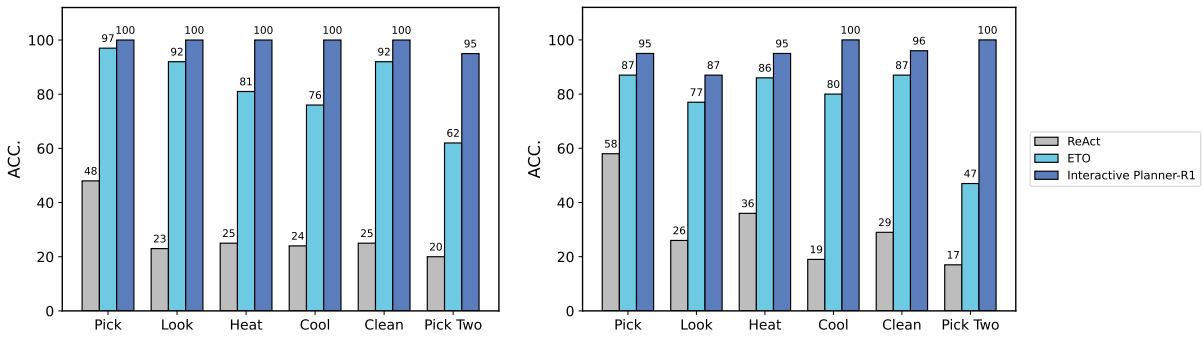
Trajectory-level vs. Single-turn Policy Learning The most significant finding concerns the importance of Interactive Policy Optimization (IPO) for multi-turn planning tasks. To isolate the effect of the optimization objective from increased exploration, we compare IPO with a naive GRPO baseline (denoted as w/o IPO) under *identical rollout and optimization budgets*, including the same group size ($k = 5$), the same number of rollouts per task, and the same number of gradient updates.

Removing IPO and replacing it with this single-turn optimization objective results in a dramatic performance collapse, dropping from 97.55% to merely 23.77%—a substantial decrease of 73.78%. Despite having access to the same amount of interaction data, the naive GRPO baseline fails to learn effective multi-step strategies, demonstrating that performance gains do not arise from collecting more trajectories alone. Instead, effective interactive planning critically depends on trajectory-level, group-relative optimization that captures long-horizon credit assignment across action sequences.

ReAct Paradigm Contribution The structured $Observation \rightarrow Think \rightarrow Action$ paradigm shows measurable but moderate impact on performance. Removing the explicit reasoning chains (w/o React) leads to a 1.41% performance decrease, from 97.55% to 96.14%. While this decline appears modest, it represents consistent degradation across both seen and unseen scenarios, indicating that structured reasoning helps the model better understand environmental contexts and maintain performance stability during planning.

Group Rollout Necessity The group rollout mechanism proves fundamental to our approach's functionality. As indicated by the missing results for "w/o group" in Table 2, this ablation becomes infeasible because group-normalized advantages require multiple parallel trajectories for statistical baseline computation. Without diverse trajectory groups, the advantage estimation becomes unstable, preventing effective policy learning entirely. This dependency underscores that parallel exploration is not merely beneficial but architecturally essential for our framework.

These ablation results collectively demonstrate that Interactive Planner-R1's superior performance stems from the synergistic combination of trajectory-level optimization, structured reasoning, and parallel exploration, with IPO serving as the most critical component for achieving effective interactive planning capabilities.



(a) Seen (b) Unseen

Figure 4: Completion rates of Qwen2.5-7B-Instruct on various ALFWorld tasks, covering both seen and unseen scenarios. We compare the prompt-based method ReAct with the training-based method ETO and Interactive Planner-R1.

	Pick	Look	Heat	Clean	Cool	Pick Two	Avg.
ETO	-9.89	-15.82	+6.95	-5.94	+5.39	-24.72	-7.34
Ours	-4.20	-12.50	-4.55	-3.13	0.00	+4.38	-3.33

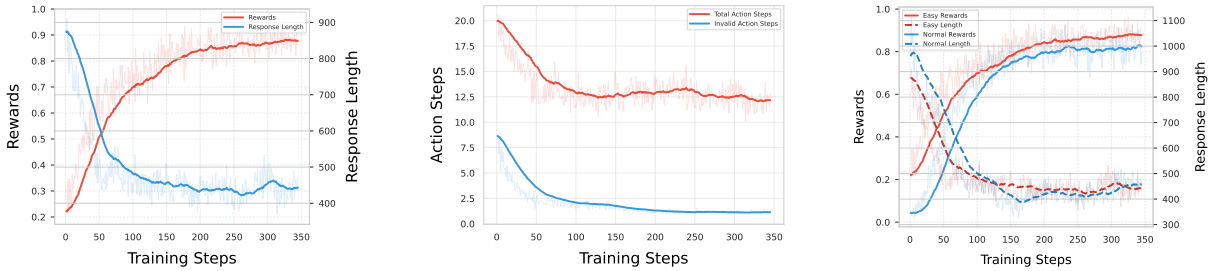
Table 3: ETO and Interactive Planner-R1's relative improvement on unseen tasks compared with seen tasks in ALFWorld, where Interactive Planner-R1 maintains strong generalization ability even on unseen tasks.

4.4 Training Analysis

Interactive Planner-R1 maintains high-level performance even on more difficult problems and unseen environments. Figure 4 compares the performance differences among various methods across different ALFWorld tasks, revealing clear difficulty hierarchies and capability differentials among the methods. As shown in the Figure, the "Pick" single-object task is relatively the most basic, where all training-based methods perform well, while purely prompt-based methods achieve less than 60% task completion rate. The "Pick Two" task is clearly the most challenging, with ReAct performing worst on this task (only 20.8% in seen and 17.6% in unseen), and ETO also showing significant decline (62.5% in seen, 47.05% in unseen). Only Interactive Planner-R1 maintains high-level performance (95.8% in seen, 100% in unseen). This difficulty distribution reflects the challenges drastically increase when tasks involve more object interactions, complex state transitions, or environmental reasoning. However, Interactive Planner-R1 maintains stable performance across the different tasks.

Generalization capabilities in unseen environments. To evaluate the generalization capability of different methods, we define the generalization gap $\Delta(abs)$ as the difference between task completion rates on unseen and seen tasks: $\Delta(abs) = Acc_{unseen} - Acc_{seen}$, as shown in Table 3. Experimental results demonstrate that our method exhibits significant advantages in generalization performance. Overall, our approach achieves a smaller average generalization gap (-3.33%) compared to the baseline method ETO (-7.34%). Notably, in the complex "Pick Two" task, our method not only overcomes the generalization challenge but achieves positive improvement (+4.38%), while ETO shows substantial performance degradation (-24.72%). This superior generalization ability is particularly evident in handling complex tasks. Further analysis reveals that our method also demonstrates stronger stability, with a relatively smaller performance variance across different task types (ranging from -12.50% to +4.38%). These results indicate that Interactive Planner-R1 effectively enhances the model's adaptability to variations in task scenarios while maintaining cross-task stability.

Efficient planning through invalid action reduction. Figure 5 demonstrates how Interactive Planner-R1 achieves efficient planning by systematically reducing invalid actions during training in the ScienceWorld. The reward curve shows steady improvement as the model learns to generate more valid action sequences (see Figure 5a). Notably, the average response length decreases from approximately 900 tokens to 500 tokens by step 50, reflecting more concise and efficient action planning. This efficiency gain is directly linked to the model's ability to avoid invalid actions, as shown in Figure 5b. The total number of action steps (red line) reduces from an initial 20 steps to a consistent 12.5 steps, while invalid actions (blue line) are nearly eliminated after 50 training steps. This substantial reduction in invalid actions indicates that Interactive Planner-R1 has developed a robust



(a) Rewards and response length (b) Total and invalid action steps (c) Performance in harder modes

Figure 5: Training progress of Interactive Planner-R1 in ScienceWorld: (a) shows the rewards and total response length in multi-round exploration; (b) illustrates the decreasing trends in both total and invalid action steps during training; (c) reveals the sustained performance trends across varying difficulty modes.

understanding of environmental constraints, leading to more efficient planning strategies and more efficient plans.

Robust Performance Under Increased Environment Complexity Level. Figure 5c further validates this capability by comparing learning performance in ScienceWorld under "Easy" and "Normal" difficulty modes. While "Normal" mode requires complete environmental interactions (navigation, container manipulation, fundamental actions), Interactive Planner-R1 demonstrates notable progress in both settings. Though "Easy" mode shows faster convergence (0.6 reward at 50 steps vs 100 steps) and higher final performance (0.88 vs 0.82), the model maintaining flexible exploration strategies and planning capabilities even under challenging environmental constraints.

5 Related Work

LLM-based Long-Horizon Planning. LLMs have been widely studied for long-horizon planning in interactive environments (Liu et al., 2024a; Xi et al., 2025). Prompt-based methods, such as ReAct (Yao et al., 2023) and its extensions (Shinn et al., 2023; Yao et al., 2024), enhance planning via structured prompting without additional training, but often struggle in complex or partially observable environments. Training-based approaches typically rely on supervised fine-tuning or behavioral cloning on static expert trajectories (Yin et al., 2024; Song et al., 2024), which limits exploration diversity and generalization.

In contrast, our work learns planning policies through online environment interaction, enabling LLM agents to actively explore and adapt their strategies under partial observability.

Interactive Agents and Reinforcement Learning. Recent work shows that reinforcement learning with simple rewards can drive strong self-improvement in LLMs (Lightman et al., 2024; DeepSeek-AI et al., 2025), but most studies focus on static or single-turn reasoning tasks. Concurrently, interactive agent frameworks such as API-Gen (Liu et al., 2024b) and LAM Simulator (Hoang et al., 2025) explore tool use or language-based human feedback as supervision.

Different from these approaches, our method targets text-based embodied environments and learns long-horizon interactive planning purely from sparse, outcome-level environment rewards via trajectory-level reinforcement learning, without dense supervision or human feedback.

More detailed discussions of related work are provided in Appendix A.

6 Conclusion

In this paper, we propose Interactive Planner-R1, a reinforcement learning framework that enables LLMs to achieve self-evolution in multi-turn environmental interaction planning tasks. Our approach demonstrates that models can autonomously develop effective planning strategies through carefully designed prompt constraints and outcome-based reward mechanisms, without relying on expert trajectories or extensive human priors. Experimental results across multiple environments show that Interactive Planner-R1 significantly outperforms existing baselines, validating the effectiveness of self-evolution via reinforcement learning in complex interactive scenarios. Notably, our analysis reveals that multi-turn interactive planning benefits from shorter context lengths, as performance improves when ineffective actions are reduced. This suggests that interactive planning tasks prioritize efficient action planning over lengthy deliberation. These findings may provide some insights for LLM-based interactive planning systems.

Limitations

While Interactive Planner-R1 demonstrates strong performance on interactive planning tasks, several limitations merit consideration for future research:

Environment Scope. Our evaluation focuses exclusively on text-based environments (ALFWorld and ScienceWorld). Although the underlying principles of multi-trajectory reinforcement learning are domain-agnostic, empirical validation in multimodal settings—particularly vision-language environments—remains an important direction for establishing broader applicability.

Computational Requirements. The group rollout mechanism, while effective for exploration diversity, introduces substantial computational overhead compared to single-trajectory methods. Training requires 8 A100 GPUs for several hours, which may limit accessibility for researchers with constrained resources. Developing more sample-efficient variants or exploring distributed training strategies could address this scalability concern.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023a. Fireact: Toward language agent fine-tuning. *CoRR*, abs/2310.05915.

Yaran Chen, Wenbo Cui, Yuanwen Chen, Mining Tan, Xinyao Zhang, Dongbin Zhao, and He Wang. 2023b. Robogpt: an intelligent agent of making embodied long-term decisions for daily instruction tasks. *CoRR*, abs/2311.15649.

Sanjiban Choudhury. 2025. Process reward models for LLM agents: Practical framework and directions. *CoRR*, abs/2502.10325.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 81 others. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. 2022. A survey of embodied AI: from simulators to research tasks. *IEEE Trans. Emerg. Top. Comput. Intell.*, 6(2):230–244.

James J. Gibson. 1979. *The Ecological Approach to Visual Perception: Classic Edition*. Houghton Mifflin.

Thai Quoc Hoang, Kung-Hsiang Huang, Shirley Kokane, Jianguo Zhang, Zuxin Liu, Ming Zhu, Jake Grigsby, Tian Lan, Michael S. Ryoo, Chien-Sheng Wu, Shelby Heinecke, Huan Wang, Silvio Savarese, Caiming Xiong, and Juan Carlos Niebles. 2025. LAM SIMULATOR: advancing data generation for large action model training via online exploration and trajectory feedback. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 12921–12934. Association for Computational Linguistics.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *CoRR*, abs/2503.24290.

Chengxing Jia, Ziniu Li, Pengyuan Wang, Yi-Chen Li, Zhenyu Hou, Yuxiao Dong, and Yang Yu. 2025. Controlling large language model with latent actions. *ArXiv*, abs/2503.21383.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zongyu Lin, Yao Tang, Xingcheng Yao, Da Yin, Ziniu Hu, Yizhou Sun, and Kai-Wei Chang. 2025. Qlass: Boosting language agent inference via q-guided stepwise search. *ArXiv*, abs/2502.02584.

Yang Liu, Weixing Chen, Yongjie Bai, Guanbin Li, Wen Gao, and Liang Lin. 2024a. Aligning cyber space with physical world: A comprehensive survey on embodied AI. *CoRR*, abs/2407.06886.

Yuqi Liu, Bohao Peng, Zhisheng Zhong, Zihao Yue, Fanbin Lu, Bei Yu, and Jiaya Jia. 2025. Seg-zero: Reasoning-chain guided segmentation via cognitive reinforcement. *CoRR*, abs/2503.06520.

Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh R. N., Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang,

Shelby Heinecke, and Caiming Xiong. 2024b. Apigen: Automated pipeline for generating verifiable and diverse function-calling datasets. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, Weihan Cao, Jiangning Liu, Hongwei Liu, Junnan Liu, Songyang Zhang, Dahua Lin, and Kai Chen. 2025. Exploring the limit of outcome reward for learning mathematical reasoning. *CoRR*, abs/2502.06781.

So Yeon Min, Devendra Singh Chaplot, Pradeep Kumar Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. 2022. FILM: following instructions in language with modular methods. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Kolby Nottingham, Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Sameer Singh, Peter Clark, and Roy Fox. 2024. Skill set optimization: Reinforcing language model behavior via transferable skills. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Agent planning with world knowledge model. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2021. Alfworld: Aligning text and embodied environments for interactive learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization for LLM agents. *CoRR*, abs/2403.02502.

Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. 2023a. Adaplanner: Adaptive planning from feedback with language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Jiankai Sun, Chuanyang Zheng, Enze Xie, Zhengying Liu, Ruihang Chu, Jianing Qiu, Jiaqi Xu, Mingyu Ding, Hongyang Li, Mengzhe Geng, Yue Wu, Wenhai Wang, Junsong Chen, Zhangyue Yin, Xiaozhe Ren, Jie Fu, Junxian He, Wu Yuan, Qi Liu, and 15 others. 2023b. A survey of reasoning with foundation models. *CoRR*, abs/2312.11562.

InternLM Team, Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, and 53 others. 2024. Internlm2 technical report. *CoRR*, abs/2403.17297.

Hanlin Wang, Jian Wang, Chak Tou Leong, and Wenjie Li. 2025. Steca: Step-level trajectory calibration for llm agent learning. *ArXiv*, abs/2502.14276.

Huaijie Wang, Shibo Hao, Hanze Dong, Shenao Zhang, Yilin Bao, Ziran Yang, and Yi Wu. 2024a. Offline reinforcement learning for LLM multi-step reasoning. *CoRR*, abs/2412.16145.

Renxi Wang, Haonan Li, Xudong Han, Yixuan Zhang, and Timothy Baldwin. 2024b. Learning from failure: Integrating negative examples when fine-tuning large language models as agents. *CoRR*, abs/2402.11651.

Ruoyao Wang, Peter A. Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11279–11298. Association for Computational Linguistics.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 9 others. 2025. The rise and potential of large language model based agents: a survey. *Sci. China Inf. Sci.*, 68(2).

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing LLM reasoning with rule-based reinforcement learning. *CoRR*, abs/2502.14768.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. 2024. Watch every step! LLM agent learning via iterative step-level process refinement. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1556–1572, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Chen Yang, Chenyang Zhao, Quanquan Gu, and Dongruo Zhou. 2024b. Cops: Empowering LLM agents with provable cross-task experience sharing. *CoRR*, abs/2410.16670.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh R. N., Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. 2024. Retroformer: Retrospective large language agents with policy gradient optimization. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Raghavi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. 2024. Agent lumos: Unified and modular training for open-source language agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 12380–12403. Association for Computational Linguistics.

Siyu Yuan, Zehui Chen, Zhiheng Xi, Junjie Ye, Zhengyin Du, and Jiecao Chen. 2025. Agent-r: Training language model agents to reflect via iterative self-training. *CoRR*, abs/2501.11425.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024a. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3053–3077. Association for Computational Linguistics.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024b. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3053–3077, Bangkok, Thailand. Association for Computational Linguistics.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yuqi Zhu, Shuofei Qiao, Yixin Ou, Shumin Deng, Shiwei Lyu, Yue Shen, Lei Liang, Jinjie Gu, Huajun Chen, and Ningyu Zhang. 2025. KnowAgent: Knowledge-augmented planning for LLM-based agents. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3709–3732, Albuquerque, New Mexico. Association for Computational Linguistics.

A Related Work

A.1 Long-Horizon Planning with Large Language Models

Large language models (LLMs) have recently demonstrated strong potential for long-horizon planning in interactive environments (Liu et al., 2024a; Xi et al., 2025; Duan et al., 2022). Existing approaches can be broadly categorized into prompt-based methods and training-based methods.

Prompt-based methods aim to enhance planning capabilities through carefully designed prompts or exemplars without additional training. ReAct (Yao et al., 2023) introduces an explicit think–act paradigm that enables LLMs to interleave reasoning with environment interaction. Subsequent work extends this framework by incorporating skill abstraction (Nottingham et al., 2024; Qiao et al., 2024; Zhao et al., 2023), reflection or revision mechanisms (Shinn et al., 2023; Yao et al., 2024), and improved prompt engineering strategies (Sun et al., 2023a). While these approaches are flexible and easy to deploy, they largely rely on pre-trained knowledge and often struggle in complex or partially observable environments.

Training-based methods directly optimize LLMs for environmental interaction, typically via supervised fine-tuning or behavioral cloning on expert trajectories (Yin et al., 2024; Zeng et al., 2024a; Chen et al., 2023a; Qiao et al., 2024). To enrich training signals, some works incorporate negative or suboptimal trajectories (Song et al., 2024; Wang et al., 2024b). However, these methods usually depend on static, pre-collected datasets, which limits exploration diversity and leads to distribution shifts between training and test environments.

Our proposed Interactive Planner-R1 addresses these limitations by learning planning policies through online interaction with the environment, allowing LLM agents to actively explore and adapt their strategies under partial observability using sparse, outcome-level feedback.

A.2 Self-Evolution and Interactive Agents

Reinforcement learning has been shown to effectively drive self-improvement in LLMs across a variety of reasoning and problem-solving tasks (Sun et al., 2023b; Lightman et al., 2024; DeepSeek-AI et al., 2025). Building on this idea, prior work has achieved strong performance in domains such as mathematics, logical reasoning, and visual understanding (Hu et al., 2025; Lyu et al., 2025; Xie et al., 2025; Liu et al., 2025). However, these settings are predominantly static or single-turn, and do not involve long-horizon interaction with an external environment.

Beyond reinforcement learning for static reasoning, several recent studies investigate interactive agents under alternative forms of supervision. APIGen (Liu et al., 2024b) focuses on synthesizing tool-using trajectories to train agents for API invocation, emphasizing tool generation and execution rather than long-horizon environment planning. LAM Simulator (Hoang et al., 2025) leverages language-based human feedback to guide agent behavior, providing rich semantic supervision through simulated human interactions. These approaches rely on dense or structured supervision signals and primarily target tool use or instruction-following scenarios.

In contrast, our work focuses on text-based embodied environments and studies whether LLM agents can acquire long-horizon interactive planning capabilities purely from sparse, outcome-level environment rewards via trajectory-level reinforcement learning, without dense supervision, human feedback, or learned evaluators.

B Pseudo Code of Group rollout of interaction with environment

Algorithm 1 presents the detailed procedure for our group rollout strategy, which is a key component of our Interactive Planner-R1 approach. The group rollout mechanism enables efficient data collection by running multiple parallel trajectories for each task, thereby increasing sample diversity and improving training stability.

The algorithm operates in three main phases. First, it freezes the current policy to ensure consistent behavior during the rollout phase and samples a batch of task-environment pairs from the training dataset. Second, for each task in the batch, it creates multiple independent environment instances and initializes separate trajectory buffers. This parallel setup allows the agent to explore different action sequences for the same task, capturing various solution strategies and potential failure modes. Third, the algorithm executes the interaction loop where the frozen policy generates responses for each trajectory simultaneously, extracts actions from these responses, and updates the environment states accordingly.

The key advantage of this group rollout approach

Algorithm 1 Group rollout of interaction with environment

Input: Initial policy π_{θ_0}, dataset $\mathcal{D} = \{(q, e) : q \in \mathcal{Q}, e \in \mathcal{E}\}$, where $\mathcal{Q}$ is the set of instructions and $\mathcal{E}$ is the set of environments with the group size set to n.

Freeze old policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
Sample batch $\mathcal{B} = \{(q_i, e_i)\}_{i=0}^{K} \sim \mathcal{D}$
Initialize global trajectories buffer $\mathcal{T} \leftarrow \emptyset$
for $(q_i, e_i) \in \mathcal{B}$ **do**
  Create n environment instances $\{e_j\}^n$ copied from e_i
  Initialize n trajectory buffers $\{\tau_j\}^n \leftarrow \emptyset$ for environment instances $\{e_j\}^n$
  for $j = 1$ to n **do**
    $o_{j,0} \leftarrow e_j$.reset()
    $\tau_j \leftarrow (q_i, o_{j,0})$
  end for
  for $k = 1$ to max steps **do**
    Generate responses $\{m_{j,k} \sim \pi_{\theta_{\text{old}}}(\cdot|\tau_j)\}^n$ for each trajectory τ_j
    Extract actions $\{a_{j,k}\}^n$ from responses $\{m_{j,k}\}^n$
    Get observations $\{o_{j,k}, \text{done}_{j,k} \leftarrow e_j.\text{step}(a_{j,k})\}^n$
    Update trajectories $\{\tau_j \leftarrow \tau_j \oplus (m_{j,k}, o_{j,k})\}^n$
    if all done **then**
      break
    end if
  end for
  Save trajectories $\mathcal{T} \leftarrow \mathcal{T} \cup \{\tau_j\}^n$
end for

Output: The grouped trajectories $\mathcal{T}$

lies in its ability to collect diverse training data efficiently. By running n parallel trajectories for each task, we obtain multiple examples of agent behavior under identical initial conditions, which helps the learning algorithm better understand the consequences of different action choices. The parallel execution also reduces the overall data collection time compared to sequential rollouts. The algorithm terminates when all environments reach their done states or the maximum number of steps is reached, ensuring that we collect complete episodes for training. The collected trajectories are then used in subsequent training phases to update the policy parameters through our reward modeling and optimization procedures.

C Illustration of Parallel Group Rollout

In this section, we provides a concrete illustration of the group rollout mechanism used in Interactive Planner-R1, with the goal of clarifying its semantics and distinguishing it from step-level branching or tree-search-based rollout methods.

C.1 Conceptual Clarification

Our group rollout mechanism performs parallel and independent simulations at the *episode level*, rather than branching from a shared intermediate environment state at each timestep. For a given task, we initialize n identical environment instances at the beginning of an episode. Each instance maintains its own environment state, observation history, and reward sequence throughout the rollout.

A single frozen policy $\pi_{\theta_{\text{old}}}$ is used to interact with all environment replicas. For any trajectory i at timestep t, the action a_i^t is sampled from $\pi_{\theta_{\text{old}}}(\cdot \mid h_i^t)$, where h_i^t denotes the unique history of that trajectory. Importantly, trajectories do not share intermediate states or observations, and no information is exchanged between replicas during rollout. Any divergence among trajectories arises solely from policy stochasticity and environment dynamics.

This procedure should be understood as a set of parallel episode-level simulations, rather than a step-level branching search from a common state.

C.2 Example: Parallel Rollout for a Simple Task

We illustrate the group rollout process using a simple example with two parallel replicas ($n = 2$).

Task. Put the apple on the table.

Step 1. The frozen policy $\pi_{\theta_{\text{old}}}$ generates actions for two independent environment instances initialized from the same task configuration.

- Trajectory 1 history: (obs: see kitchen) $\rightarrow$ Action: go to fridge
- Trajectory 2 history: (obs: see kitchen) $\rightarrow$ Action: go to counter

Step 2. Each environment updates independently based on the executed action. The policy then generates the next action conditioned on each trajectory's unique observation and history.

- Trajectory 1 history: (..., obs: at fridge) $\rightarrow$ Action: open fridge
- Trajectory 2 history: (..., obs: at counter, see apple) $\rightarrow$ Action: take apple

As the rollout proceeds, trajectories continue to evolve independently, potentially reaching task completion at different times or following different action sequences. This example highlights that trajectories within a group do not branch from shared intermediate states and that the frozen policy serves as a consistent action generator across all replicas during a rollout phase.

D Implementation Details

D.1 Evaluation and Benchmark Details

ALFWorld (Shridhar et al., 2021) ALFWorld encompasses six categories of planning tasks set primarily in home environments, covering not only basic object manipulation (such as pick and place) but also tasks requiring complex interaction sequences. For example, the heating task requires models to first identify target objects, move them to heating devices (like microwave), execute the heating operation, and finally place them in designated locations to complete the task.

Following prior research (Yao et al., 2023; Song et al., 2024), we evaluate model performance under two conditions: seen and unseen scenarios. Seen scenarios consist of task instances from rooms encountered during training, unseen scenarios comprise task instances from entirely new rooms with container arrangements and scene organizations distinctly different from training tasks, designed to evaluate the model's zero-shot generalization capabilities.

ScienceWorld (Wang et al., 2022) ScienceWorld presents a more challenging benchmark, requiring models to complete scientific experiments in a highly interactive environment. This environment contains approximately ten interconnected rooms, over 200 object types, and 25 executable actions. Compared to ALFWorld, ScienceWorld integrates a more sophisticated physics simulation engine, including thermodynamic and electrical systems, which places more rigorous demands on a model's planning capabilities and causal reasoning.

Following prior research (Yao et al., 2023; Song et al., 2024), we assessed model performance under two conditions: seen and unseen scenarios. Seen scenarios consist of task instances from rooms encountered during training, including the same task types, objects, containers, and room layouts, but with variations in object positions, quantities, and visual features (e.g., two blue pencils on a shelf instead of three red pencils in a drawer as seen in training data). Unseen scenarios comprise task instances from entirely new rooms with container arrangements and scene organizations distinctly different from training tasks, designed to evaluate the model's zero-shot generalization capabilities. During evaluating ScienceWorld, we maintained the "easy" setup consistent with previous work (Song et al., 2024), enabling all five assistive features: instant teleportation, pre-opened containers, self-watering flower pots, and two additional aids that remove similar micro-action bottlenecks, thus reducing the burden of low-level operations and focusing the evaluation on high-level planning abilities. We also adopted the same balancing strategy as previous work (Song et al., 2024), implementing fair sampling at the task level.

D.2 Baseline

To benchmark our Interactive Planner-R1 against other training-based approaches, we compared it with three state-of-the-art methods that incorporate human priors: SFT (Zeng et al., 2024a), which fine-tunes on expert trajectories to learn interaction capabilities; NAT (Wang et al., 2024b), which incorporates learning from rejected trajectories; and ETO (Song et al., 2024), which learns from expert trajectories through DPO. Our evaluation also includes larger mod-

Category	Setting / Description
Backbone model	Qwen2.5-7B-Instruct
Context length	4096
Random seed	42
Rollout batch size	128
Group size	5
Learning rate	1e-6
Batch size	32
Max trajectory length	30
Clip parameter ϵ	0.2
KL penalty coefficient β	0.001
Optimizer	AdamW
Learning rate schedule	Constant

Table 4: Training, optimization, and implementation details for Interactive Planner-R1.

els: GPT-3.5 Turbo (Brown et al., 2020)(gpt-3.5-turbo-0125), GPT-4 (OpenAI, 2023)(gpt-4o-2024-08-06), DeepSeek-V3 (DeepSeek-AI et al., 2024)(DeepSeek-V3-0324), and InternLM 2.5 20B (Team et al., 2024), to demonstrate the current performance of foundation large language models (both open-source and closed-source) on interactive planning tasks. All our implemented training-based methods are exclusively fine-tuned on Qwen2.5-7B-Instruct (Yang et al., 2024a), and all models are evaluated using a consistent set of prompts. Additionally, while many methods cannot be directly compared due to differences in models and evaluation approaches, we still provide these results in Table 5 for reference.

D.3 Training Details

We implemented Interactive Planner-R1 based on the `verl` framework[1], a flexible reinforcement learning library for large language models, and extended it with an interaction module to support multi-turn environment rollouts. For clarity and reproducibility, we summarize the key training hyperparameters, optimization settings, and rollout configurations in Table 4.

Rollout and Data Collection. At each training iteration, we sample a batch of 128 task–environment pairs. For each task, we perform grouped rollouts with a group size of 5, where each group consists of independent environment replicas initialized from the same task configuration. Each trajectory follows the ReAct paradigm and is capped at a maximum of 30 interaction steps. A special `done` action is introduced to explicitly terminate a trajectory when the model determines that the task has been completed or is unsolvable. If `done` is generated prematurely without satisfying the task completion condition, the trajectory receives zero reward.

Policy Optimization. All trajectories within a group share a binary, completion-driven reward (1 if the task is completed, 0 otherwise). Advantages are computed using group-normalized returns as described in Eq. (8), which requires multiple parallel trajectories and is not well-defined for a single-trajectory setting. We optimize the policy using Interactive Policy Optimization (IPO), with clipped probability ratios ($\epsilon = 0.2$) and a KL divergence penalty with respect to a fixed reference model. Unless otherwise stated, the KL coefficient is set to $\beta = 0.001$, which we found to provide a good trade-off between training stability and exploration performance.

Model and Optimization. We use `Qwen2.5-7B-Instruct` (Yang et al., 2024a)[2] as the backbone policy model. Training is conducted with a maximum context length of 4096 tokens, which includes the instruction, observation history, and previous ReAct steps. The reference model is initialized from the same checkpoint and kept frozen throughout training. We use the AdamW optimizer with default parameters provided by `verl`. The random seed is fixed to 42 for all experiments.

Training Setup and Evaluation. All experiments are conducted on 8 NVIDIA A100 GPUs with 80GB memory. Grouped rollouts are only used during training; during evaluation, we perform single-trajectory inference with a maximum of 30 steps and report pass@1 task completion rates. The sampling temperature is set to 1.0 during evaluation unless otherwise specified. Each experiment is repeated 5 times with different random seeds, and we report the mean and standard deviation.

E Prompt Design

In our research, prompt engineering accomplishes three interconnected objectives: (i) enumerating permissible action, guiding the model to execute effective commands; (ii) mitigating common hallucinations, preventing the model from falling into blind trial-and-error loops; and (iii) enforcing a concise ReAct paradigm that clearly demonstrates

[1] https://github.com/volcengine/verl

[2] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

ALFWorld Templates:

Environment description

You are an intelligent agent in a household environment and your target is to perform actions to complete the task goal. At the beginning of your interactions, you will be given the detailed description of the current environment and your goal to accomplish.

For each of your turn, you will be given the observation of the last turn. You should first think about the current condition and plan for your future actions, and then output your action in this turn. Your output must strictly follow this format: **Thought:** <your thoughts> ; **Action:** <your next action>.

Available actions & required format

The available actions are:

1. `go to (receptacle)`
2. `open (receptacle)`

⋮

13. `inventory:check your current inventory`
14. `done:Indicate that you believe the task is complete`

Explanations for abstract actions

Where `(object)` refers to manipulable objects and `(receptacle)` refers to receptacles or locations.

Avoid hallucination

After your each turn, the environment will give you immediate feedback based on which you plan your next few steps, if the environment output: Nothing happens, that means the previous action is invalid and you should try more options.
You can only hold one object at a time. Before taking a new object, make sure you have placed down any object you are currently holding.
You should not assume or anticipate the feedback. Even if you have planned multiple steps ahead, you should only execute one action at a time.
Do not proceed with any further exploration or actions until you receive the feedback from the environment after your action.

Output template

Your response should use the following format:
Thought: [your thoughts]
Action: [your next action]

Figure 6: Prompt for ALFWorld

reasoning chains and actions, facilitating straightforward extraction of selected operations. Following (Qiao et al., 2024), we enhanced the system instructions to explicitly highlight the allowable action set, and required output format, while specifically addressing common hallucination patterns. This approach significantly increasing the probability of obtaining initial successful trajectories necessary for guided learning. The prompt is detailed in Figure 6 and Figure 7.

F More experimental results

Figure 9a illustrates the metrics of the Interactive Planner-R1 during the training process in ALFWorld, which yields results similar to those in ScienceWorld. The reward curve exhibits a similar upward trend. Meanwhile, the response length significantly decreases from the outset, dropping from approximately 750 tokens initially to around 450 tokens by the end. The relationship between response length and action steps is also evident in Figure 5b,

ScienceWorld Templates:

System Prompt:

Environment description

You are a helpful assistant to do some scientific experiment in an environment.
You should explore the environment and find the items you need to complete the experiment.

In the environment, there are several rooms: kitchen, foundry, workshop, bathroom, outside, living room, bedroom, greenhouse, art studio, hallway.You can teleport to any room in one step.

Available actions & required format

The available actions are:

```
activate OBJ
close OBJ
⋮
wait1: wait 1 step
done: indicate that you believe the task is complete
```

Avoid hallucination

When arrive a new location, you should use look around to check the OBj you can interact with. Use focus on OBJ only neccessary as incorrect use will cause environment ends. Do not proceed with any further exploration or actions until you receive the feedback from the environment after your action.

Output template

Your response should use the following format:
Thought: [your thoughts]
Action: [your next action]

Figure 7: Prompt for ScienceWorld

where total actions, invalid actions, and response length show a consistent downward trend. Due to the reduced difficulty compared to ScienceWorld, invalid steps have been almost entirely eliminated in the final results.

In Figure 8, we present the task completion performance of our trained agents in ScienceWorld, alongside a comparison with ReAct and ETO. This comparison underscores the effectiveness and generalization capabilities of our Interactive Planner-R1. While prompt-based and previous training-based methods often exhibit strong performance on tasks that emphasize common sense reasoning (e.g., chemistry), their success rates plummet when faced with tasks that heavily rely on environmental interaction (e.g., classification). In fact, many of these tasks remain unsolvable for prompt-based approaches. In contrast, our method demonstrates exceptional completion rates across both seen and unseen tasks, highlighting its robustness and adaptability in complex and varied scenarios.

Table 5 presents a comprehensive comparison of our Interactive Planner-R1 against existing methods on both ALFWorld and ScienceWorld benchmarks. The table is organized into two main sections to ensure fair evaluation: referenced results from original papers (marked with *) and our own implementations using identical experimental settings. The referenced results section includes a wide range of state-of-the-art methods, from traditional approaches like CoLA and IPR to more recent advances such as CoPS and AgentPRM, with some methods achieving impressive performance on ALFWorld (e.g., CoPS reaching 94.0% average completion rate). However, it is important to note that these results may not be directly

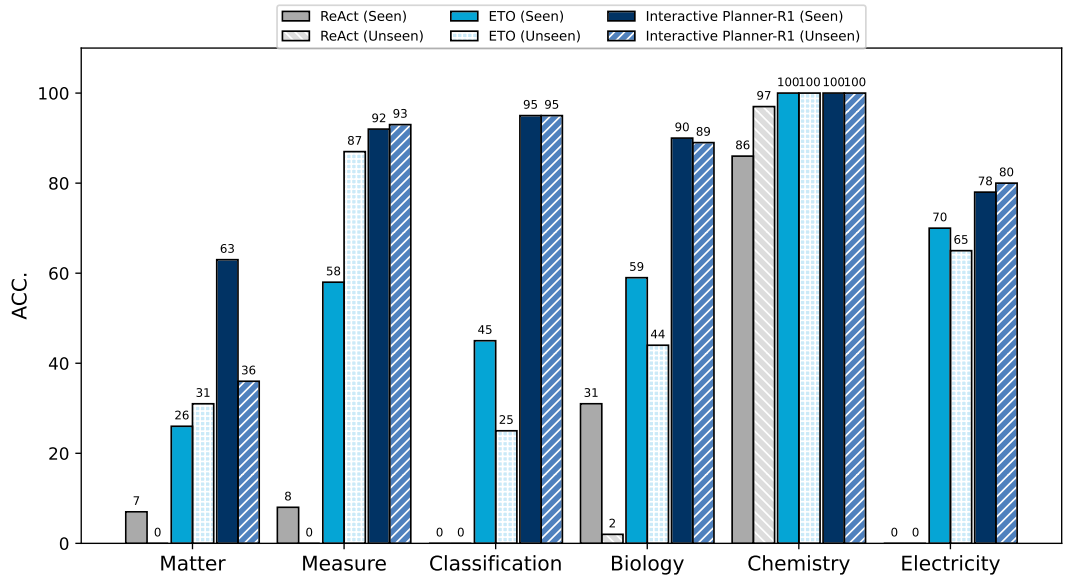
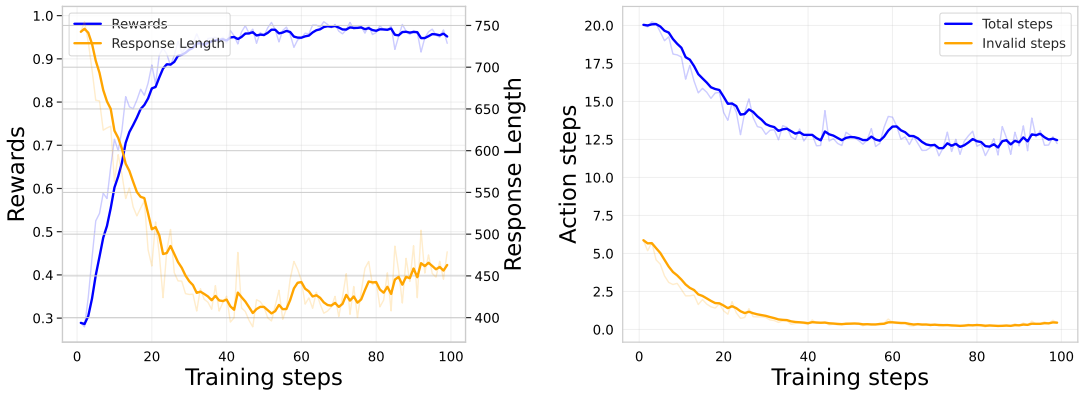


Figure 8: Completion rates of Qwen2.5-7B-Instruct on various ScienceWorld tasks, covering both seen and unseen scenarios. We compare the prompt-based method ReAct with the training-based method ETO and Interactive Planner-R1.



(a) Rewards and response length

(b) Total and invalid action steps

Figure 9: Training progress of Interactive Planner-R1 in ALFWorld: (a) shows the rewards (task completion rate) and total response length in multi-round exploration; (b) illustrates the decreasing trends in both total and invalid action steps during training.

comparable due to potential differences in experimental configurations, evaluation protocols, or implementation details. Our implementation section provides a controlled comparison using consistent experimental settings across all methods. The results demonstrate that vanilla large language models with ReAcT prompting generally struggle on these tasks, with even GPT-4o achieving only 55.41% on ALFWorld and 58.18% on ScienceWorld. Training-based methods show significant improvements, with ETO achieving 82.07% on ALFWorld, though its performance on ScienceWorld remains limited at 56.20%. In contrast, our Interactive Planner-R1 achieves remarkable performance across both benchmarks, reaching 97.55% average completion rate on ALFWorld and 79.92% on ScienceWorld. Notably, our method demonstrates strong generalization capabilities, maintaining high performance on both seen and unseen scenarios, with minimal performance degradation between the two settings. This consistent performance across different task distributions highlights the robustness and effectiveness of our approach in complex interactive environments.

G More ablation study

Method	ALFWorld			ScienceWorld		
	Seen	Unseen	Avg.	Seen	Unseen	Avg.
Referenced Results (Not Directly Comparable)						
KnowAgent* (Zhu et al., 2025)	66.71	62.69	64.70	58.67	49.18	53.93
CoLA* (Jia et al., 2025)	77.90	74.60	76.30	28.40	21.80	25.10
IPR* (Xiong et al., 2024)	70.30	74.70	72.50	-	-	-
STeCa* (Wang et al., 2025)	74.30	76.10	75.20	-	-	-
Agent-R* (Yuan et al., 2025)	-	-	-	-	-	70.23
DEP* (Wang et al., 2023)	-	-	76.00	-	-	-
WKM* (Qiao et al., 2024)	68.57	65.93	67.25	58.67	49.18	53.93
OREO* (Wang et al., 2024a)	79.10	80.70	79.90	-	-	-
QLASS* (Lin et al., 2025)	77.90	82.80	80.40	75.30	66.40	70.90
AgentLM* (Zeng et al., 2024b)	-	-	86.00	-	-	20.80
AgentPRM* (Choudhury, 2025)	-	-	91.00	-	-	-
AdaPlanner* (Sun et al., 2023a)	-	-	91.79	-	-	-
CoPS* (Yang et al., 2024b)	-	-	94.00	-	-	-
Our Implementation						
GPT-3.5-Turbo†(Brown et al., 2020)	7.14	7.46	7.30	28.03	21.63	24.83
GPT-4o†(OpenAI, 2023)	58.57	52.24	55.41	59.6	56.75	58.18
Deepseek-v3†(DeepSeek-AI et al., 2024)	36.43	31.34	33.89	29.28	27.45	28.37
InternLM 2.5 20B Chat†(Team et al., 2024)	7.86	11.94	9.90	33.66	31.15	32.41
Qwen2.5-7B-Instruct†(Yang et al., 2024a)	30.00	32.09	31.05	28.01	16.09	22.05
Qwen2.5-7B-Instruct + SFT(Zeng et al., 2024a)	80.71	79.10	79.91	68.81	55.7	62.26
Qwen2.5-7B-Instruct + NAT(Wang et al., 2024b)	63.57	66.42	65.00	58.56	49.75	54.16
Qwen2.5-7B-Instruct + ETO(Song et al., 2024)	84.29	79.85	82.07	55.08	57.32	56.20
Qwen2.5-7B-Instruct + Interactive Planner-R1(Ours)	**98.57** ±0.7	**96.52** ±1.9	**97.55**	**83.66** ±0.5	**76.18** ±0.9	**79.92**

Table 5: **Complete experimental results on ALFWorld and ScienceWorld benchmarks.** The best results are marked in **bold**. † indicates vanilla models tested with ReAcT prompting, and * denotes the best results reported in the original papers.

Group Size	Seen	Unseen	Avg.
2	97.86 ±0.4	92.68 ±0.8	95.27
3	98.14 ±0.4	97.31 ±0.4	97.73
5	98.57 ±0.7	96.52 ±1.9	97.55
7	98.93 ±0.5	95.9 ±0.5	97.41

Table 6: Performance across different group sizes on ALFWorld. Group size 5 provides the optimal balance between exploration diversity and computational efficiency.

Temperature	Seen	Unseen	Avg.
0.5	96.43 ±2	94.78 ±0.3	95.61
0.7	97.86 ±2	97.02 ±1.1	97.44
1.0	98.57 ±0.7	96.52 ±1.9	97.55

Table 7: Impact of temperature parameter on ALFWorld performance. Higher temperature enables better exploration while maintaining action quality.

Group Size Analysis The group rollout mechanism is central to Interactive Planner-R1's exploration strategy. We systematically examine how group size affects both exploration diversity and computational efficiency by varying the number of parallel trajectories from 2 to 7 and shown in Table 6.

As the Table 6 shown, the results reveal an interesting pattern: performance initially improves with group size (from 95.27% at size 2 to 97.73% at size 3), suggesting that increased trajectory diversity enhances learning quality. However, beyond group size 5, we observe diminishing returns, particularly on unseen environments (95.9% for size 7 vs. 96.52% for size 5). This suggests that while larger groups provide more exploration diversity, they may also introduce noise that interferes with policy learning. Group size 5 emerges as the optimal configuration, balancing exploration breadth with learning stability.

Temperature Sensitivity Analysis Sampling temperature critically controls the exploration-exploitation trade-off during trajectory generation. We evaluate three temperature values to understand

how randomness affects both exploration quality and final performance.

Temperature 1.0 achieves the highest overall performance (97.55%), indicating that sufficient exploration randomness is essential for discovering effective action sequences. Lower temperatures (0.5 and 0.7) result in more conservative exploration, limiting the model's ability to discover optimal strategies. Interestingly, temperature 0.7 shows the best generalization (only 0.84% gap between seen and unseen), while temperature 1.0 provides the best overall performance despite a larger generalization gap (2.05%).

KL Weight	Seen	Unseen	Avg.
0	99.29 ± 1	96.27 ± 2.1	97.78
0.001	98.57 ± 0.7	96.52 ± 1.9	97.55
0.1	97.15 ± 1	97.01 ± 1	97.08

Table 8: Effect of KL penalty on ALFWorld performance. Removing KL constraints yields optimal performance, suggesting sparse rewards provide sufficient regularization.

KL Divergence Constraint Effects Kullback-Leibler (KL) penalty is commonly used in RL to prevent the policy from deviating too far from the reference model. We investigate whether such constraints are necessary given our sparse reward structure.

As shown in Table 8, removing KL penalty entirely ($\beta = 0$ in Equation 7) yields the highest performance (97.78%), outperforming both light ($\beta = 0.001$) and moderate ($\beta = 0.1$) regularization. This finding suggests that our sparse, completion-based reward structure inherently provides sufficient regularization against reward hacking and policy drift. The strong KL constraint (0.1) actually hurts performance by overly restricting policy updates, preventing the model from learning effective exploration strategies. However, we observed some degree of instability in unconstrained training, manifested as significant training curve fluctuations and inconsistent convergence rates. Based on this finding, we adopted light KL regularization ($\beta = 0.001$) as a compromise in our final implementation.

H Error Analysis

To better understand the behavioral differences between Interactive Planner-R1 and prior multi-turn reinforcement learning baselines, we conduct a detailed error analysis against ETO under identical evaluation settings. The analysis is performed on the same set of tasks using fixed checkpoints for both methods, without any additional training.

For each failed trajectory, we manually assign one or more high-level error categories, including: (i) execution errors after sufficient information has been obtained, (ii) oscillatory or contradictory action sequences, (iii) goal misunderstanding after locating key objects, (iv) insufficient exploration (failing to locate required objects or locations), (v) inefficient execution, (vi) parsing errors, and (vii) unknown failure causes. The aggregated statistics are summarized in Table 9.

As shown in Table 9, failures of ETO are dominated by execution-stage errors. A substantial portion of failed trajectories execute incorrect action sequences despite already observing the key object or location, while many others exhibit repeated or oscillatory behaviors. This indicates that ETO often struggles to convert acquired information into stable and coherent multi-step plans. In contrast, Interactive Planner-R1 significantly reduces execution-stage failures, with only a small fraction of errors occurring after sufficient information has been obtained. Moreover, the proportion of unknown failure cases is lower, suggesting more interpretable and structured failure modes.

The remaining failures of Interactive Planner-R1 are primarily attributed to occasional goal misunderstanding or insufficient exploration, rather than breakdowns in planning or execution once the relevant information is available. This analysis suggests that the performance gains of Interactive Planner-R1 mainly stem from improved planning stability and execution quality under partial observability, rather than solely from increased exploration.

I Case Study: Model's Knowledge about Environment

Beyond the model's successful generalization to both seen and unseen examples, which already indicates its understanding of environmental knowledge, we conducted a case study in ALFWorld to further compare the differences in environmental knowledge mastery between Interactive Planner-R1 and the SFT model.

To demonstrate that the model has acquired environmental knowledge through exploration, we

Error Type	ETO	Planner-R1
Executed wrong action sequence after having enough information	9 (0.33)	1 (0.20)
Repeated contradictory or oscillatory actions	9 (0.33)	2 (0.40)
Found key object but misunderstood the goal	6 (0.22)	2 (0.40)
Never found key object / location (insufficient exploration)	6 (0.22)	2 (0.40)
Insufficient exploration (generic)	3 (0.11)	–
Inefficient execution	1 (0.04)	–
Misunderstood goal	1 (0.04)	–
Parsing error	1 (0.04)	–

Table 9: Breakdown of failure modes for ETO and Interactive Planner-R1. Each failed trajectory may be assigned to multiple error categories; therefore, percentages do not sum to 1. Values are reported as number of trajectories followed by the ratio in parentheses.

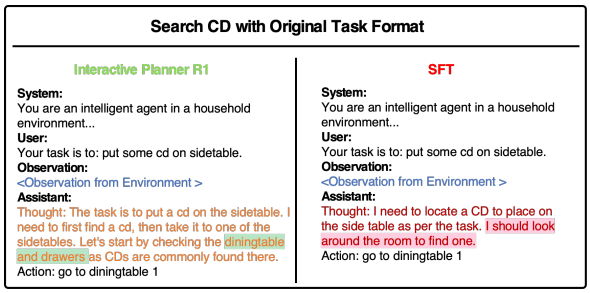


Figure 10: Comparison between Interactive Planner-R1 and SFT using the original task format,where Interactive Planner-R1 pointed out the possible position of CD and SFT exhibited inconsistency between thought and action .

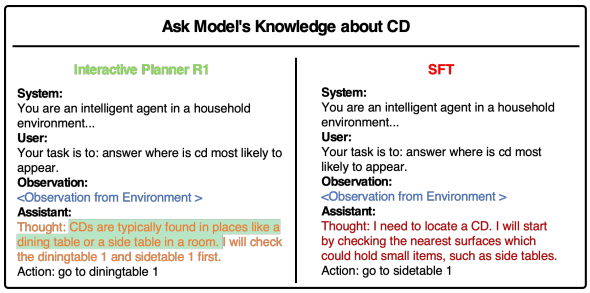


Figure 11: Comparison between Interactive Planner-R1 and SFT when asked about the location of CD, showing SFT suffered from prior knowledge bias while Interactive Planner-R1 still succeeded.

identified an object, CD, that appears in numerous tasks positioned on dining tables, which contradicts common expectations where CDs would more likely be stored in drawers or on shelves.

In Figure 10, we first maintained the input format used during training and found that although both SFT and Interactive Planner-R1 models made correct actions, the SFT model's thoughts did not strongly correlate with its actions. In contrast, Interactive Planner-R1 demonstrated awareness of CD locations in its thought process.

To further verify that Interactive Planner-R1 has mastered knowledge related to the environment, we modified the input format as shown in the Figure 11, changing the task to ask the model where CDs are most likely to appear. The SFT model failed to complete this type of task, with its thoughts still reflecting the influence of prior knowledge without recognizing that CD locations in ALFWorld differ from common expectations. Interactive Planner-R1, however, still made the correct analysis in its thoughts, proving that the model has successfully captured this implicit spatial relationship during its exploration phase.