Beyond Policy Optimization: A Data Curation Flywheel for Sparse-Reward Long-Horizon Planning

Anonymous ACL submission

Abstract

Large Language Reasoning Models have demonstrated remarkable success on static tasks, yet their application to multi-round agentic planning in interactive environments faces two fundamental challenges. First, the intractable credit assignment problem renders conventional reinforcement learning ineffective in sparse-reward settings. Second, the computational overhead of verbose, step-by-step reasoning histories is prohibitive. To address these challenges, we propose BPO, a three-stage framework (bootstrapping, extrapolation, and refinement) that establishes a self-improving data flywheel to develop robust reasoning models for long-horizon, sparse-reward environments. Our framework first bootstraps efficient reasoning using the proposed planning quaternions with long-short chain-of-thought fusion. It then extrapolates to out-of-distribution tasks through complexity-stratified curriculum learning. Finally, the model iteratively refines itself by learning exclusively on experiences selected via reward-gated rejection sampling. Experiments on ALFWorld, ScienceWorld, and WebShop demonstrate that our approach achieves state-of-the-art with significant token efficiency, providing a new recipe for reasoning models in agentic planning.

1 Introduction

Recent advances in Large Language Models (LLMs), particularly the emergence of models capable of "System 2" reasoning (Jaech et al., 2024; Guo et al., 2025), have marked a paradigm shift (Ke et al., 2025). By generating explicit chains of thought, these models achieve expert-level performance in static, self-contained domains such as mathematics and programming (Wei et al., 2025; Chen et al., 2025). They are inherently better suited for agentic planning, bringing a global perceptual scope that enables multi-step foresight and planning with subgoals. This raises a critical question: can this powerful reasoning paradigm be extended to dynamic, interactive environments to build autonomous agents capable of long-horizon planning? (Shinn et al., 2023). However, integrating explicit reasoning into agentic tasks presents two major challenges.

The first challenge is the intractable credit assignment problem endemic to long-horizon tasks with sparse rewards (Shridhar et al., 2020; Wang et al., 2022). In many interactive settings, rewards, typically binary success indicators, are delivered only after a long and complex sequence of actions. This significant temporal delay renders the feedback signal too weak for reinforcement learning algorithms to effectively propagate credit back through the trajectory. As our experiments show, even advanced preference optimization algorithms, such as Group Relative Policy Optimization (GRPO) (Shao et al., 2024), yield only marginal improvements, highlighting the fundamental limitations of optimizing reasoning LLMs in this regime. The second challenge is the computational and cognitive overhead of verbose reasoning. While detailed reasoning is crucial for complex problem-solving, its naive inclusion at every step of an interactive trajectory inflates context length, incurs substantial computational costs, and risks displacing critical information from earlier in the trajectory out of the model's limited attention window (Bai et al., 2023).

To address the challenges of adapting reasoning LLMs to long-horizon, sparse-reward planning settings, this work introduces BPO, a novel three-stage framework of bootstrapping, extrapolation, and refinement that establishes a self-improving *data flywheel*. First, our framework bootstraps efficient reasoning through Planning Quaternion Synthesis, a specialized data generation process that disentangles verbose deliberation from concise planning intent. This stage is paired with a long-short chain-of-thread fusion strategy that not only ensures contextual efficiency but also mirrors a plausible cognitive model of long-term planning, where detailed immediate reasoning is compressed into high-level conclusions for future reference. Second, the agent's robustness is cultivated via Curriculum Synthesis, which proactively generates challenging out-of-distribution tasks and organizes them into a complexity-stratified curriculum to systematically expand its capabilities. Finally, the agent achieves self-refinement through reward-based trajectory improvement. In this stage, the sparse environmental reward acts as a deterministic criterion for success-contingent rejection sampling, creating a virtuous cycle of iterative fine-tuning on exclusively successful trajectories.

In summary, our contributions are threefold:

- We introduce BPO, a three-stage framework of bootstrapping, extrapolation, and refinement for reasoning LLMs in sparse-reward, long-horizon agentic tasks.
- We propose a suite of techniques, including planning quaternion data structure, long-short chain-of-thought fusion, and curriculum synthesis, collectively enable robust, efficient, and generalizable reasoning.
- Extensive experiments on ALFWorld, ScienceWorld, and WebShop show our approach achieves state-of-the-art performance, surpassing fine-tuning and larger proprietary models with significant fewer tokens.

2 Task Formulation

We formalize the problem of a reasoning LLM agent interacting with an environment as a *partially observable Markov decision process (POMDP)*, augmented to explicitly account for the agent's internal reasoning process. The environment is defined by the tuple

$$\mathcal{M} = (U, S, A, O, E, T, R),$$

where U represents the space of natural language task instructions, S denotes the latent space of environment states, A is the discrete set of permissible actions, O describes the space of observations provided by the environment, and E refers to the space of explicit reasoning traces (i.e., thoughts). The deterministic state transition function is

$$T : S \times A \to S,$$

and the sparse reward function is

$$R : S \times A \to \{0, 1\}.$$

At each time step t, an agent parameterized by θ receives an observation o_t. Conditioned on the instruction u and the interaction history, its policy π_θ generates a reasoning trace e_t and then selects an action a_t. The resulting generative process is:

$$\begin{aligned}[e_t, a_t] \sim \pi_\theta(u, (o_1, e_1, a_1), \ldots, \\ (o_{t-1}, e_{t-1}, a_{t-1}), o_t).\end{aligned}$$

The execution of a_t transitions the environment to a new state $s_{t+1} = T(s_t, a_t)$, yielding a new observation o_{t+1}. This interaction repeats until a terminal state or a maximum number of steps is reached. The agent's objective is to learn a policy π_θ that maximizes the expected cumulative reward.

For supervised fine-tuning, the learning objective is to maximize the likelihood of generating the correct reasoning and action from expert trajectory. A trajectory $\mathcal{T}$ consists of a sequence of historical states leading to a target reasoning–action pair. Given the instruction and the history of observation–thought–action tuples, the model is trained to predict the subsequent thought e_N and action a_N:

$$\begin{aligned}\mathcal{T} = [u, (o_1, e_1, a_1), \ldots, (o_{N-1}, e_{N-1}, a_{N-1}), \\ o_N \to (e_N, a_N)].\end{aligned} \tag{1}$$

This formulation unifies the instruction, historical context, and reasoning into a single sequence, enabling the training of reasoning LLM agents.

3 Method

Our proposed framework is designed to train LLM agents capable of effective long-horizon planning under sparse rewards. The methodology deliberately circumvents the intractable credit assignment problem that plagues conventional reinforcement learning by re-purposing the sparse environmental rewards, from a noisy learning signal, into an unambiguous filter for data curation. As shown in Figure 1, we propose a three-stage pipeline: (1) bootstrapping a foundation of efficient reasoning, (2) extrapolating via a synthetic curriculum, and (3) achieving perfection through iterative refinement guided by sparse rewards.

3.1 Stage 1: Bootstrapping Reasoning via Planning Quaternion Synthesis

The initial stage addresses the dual challenges of acquiring high-quality, multi-faceted reasoning data and managing the computational overhead associated with verbose thought processes.

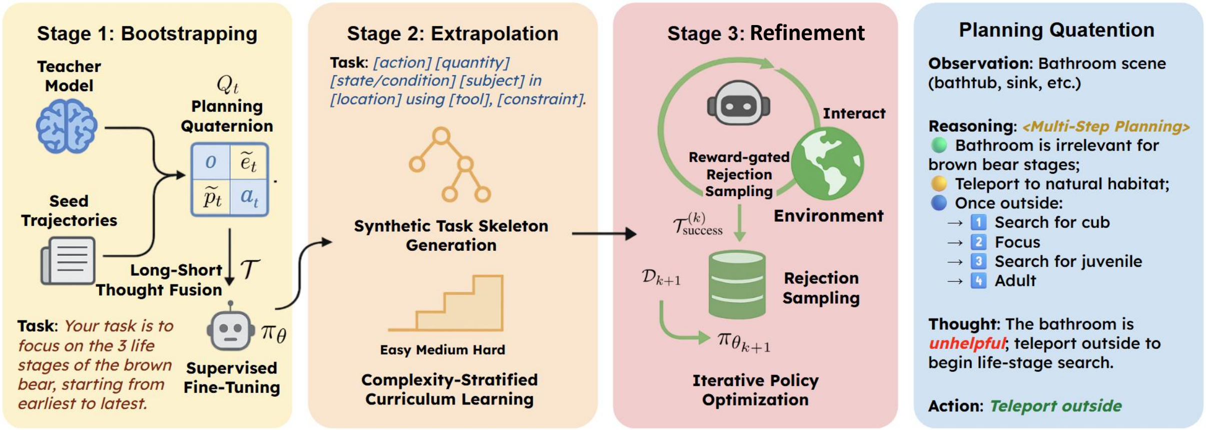

Figure 1: Overview of our three-stage framework for training reasoning LLMs in long-horizon, sparse-reward environments. **Stage 1: Bootstrapping Reasoning via Planning Quaternion**. An initial agent is trained via supervised fine-tuning on planning quaternion data from seed trajectories using a teacher model. **Stage 2: Extrapolation via Curriculum Synthesis**. The agent's capabilities are expanded by training on a curriculum of synthetically generated, out-of-distribution tasks of increasing complexity. **Stage 3: Self-Refinement via Reward-Gated Refinement**. The agent enters an iterative self-improvement loop, generating new trajectories and enriching the dataset for subsequent fine-tuning through rejection sampling while interacting with the environment.

3.1.1 The Planning Quaternion Data Flywheel

High-quality data that jointly models detailed deliberation and concise, actionable planning is scarce. To overcome this limitation, we introduce the *Planning Quaternion Data Flywheel* to systematically generate rich training data. The process unfolds in four steps:

Seed Trajectory Curation: We begin by collecting a set of successful trajectories from a powerful, off-the-shelf reasoning model (e.g., DeepSeek-R1) in the target environment. These trajectories provide initial examples of observation-reasoning-action sequences.

Synthetic Expansion: Using these seed trajectories as few-shot exemplars, we prompt a highly capable teacher model (e.g., GPT-4o) to synthesize a larger, more diverse dataset of full reasoning traces e_t corresponding to various observations o_t.

Distillation to Planning Thoughts: To mitigate the context length problem, each verbose reasoning trace e_t is distilled by the teacher model into a concise short planning thought p_t. This summary captures the essential intent and next step without extensive token overhead.

Quaternion Assembly: The final data structure is the planning quaternion, $Q_t = (o_t, e_t, p_t, a_t)$, which encapsulates the observation, the full reasoning trace, the distilled planning thought, and the resulting action. This structure provides a principled representation that disentangles deep, immediate deliberation from the high-level planning intent required for long-horizon consistency.

3.1.2 Long-Short Chain-of-Thought Fusion for Efficient Deliberation

To efficiently operationalize the planning quaternions during inference, we employ a *long-short chain-of-thought fusion* strategy. At each decision step t, the agent first generates a full reasoning trace e_t (the "long chain") to deliberate on the immediate problem. Subsequently, it generates a compressed planning thought p_t (the "short chain"). For all subsequent steps, only the sequence of short thoughts $\{p_1, \ldots, p_{t-1}\}$ is retained in the context history. This method allows the model to benefit from detailed reasoning in the immediate step while preventing context overload and preserving the continuity of the high-level plan.

3.1.3 Multi-Round Interactive Supervised Fine-tuning

Our agent is trained on the generated quaternions using a step-level, multi-round interactive supervised fine-tuning objective. The introduction of the planning quaternion refines our task formulation. The agent's policy π_θ now generates a triplet comprising the full reasoning trace e_t, the concise planning thought p_t, and the action a_t. Critically, to maintain contextual efficiency, the policy is conditioned not on the full reasoning history, but on the sequence of concise planning thoughts. The generative process is thus redefined as:

$$[e_t, p_t, a_t] \sim \pi_\theta(u, (o_1, p_1, a_1), \ldots, (o_{t-1}, p_{t-1}, a_{t-1}), o_t).$$

Consequently, the supervised fine-tuning objective is to maximize the likelihood of generating the target triplet (e_N, p_N, a_N) from an expert trajectory $\mathcal{T}$, now structured as:

$$\begin{aligned}\mathcal{T} = [u, (o_1, p_1, a_1), \ldots, \\ (o_{N-1}, p_{N-1}, a_{N-1}), \\ o_N \to (e_N, p_N, a_N)].\end{aligned} \tag{2}$$

This procedure trains the agent to perform both the reasoning and summarization required for efficient and effective long-horizon planning.

3.2 Stage 2: Extrapolation via Curriculum Synthesis

To consolidate and generalize beyond the initial reasoning patterns under the seed data distribution, the second stage enhances the agent's performance on novel tasks and scenarios. We formulate this as curriculum synthesis, generating progressively harder tasks to systematically probe and expand the agent's capabilities.

3.2.1 Generative Task Augmentation for Out-of-Distribution Exposure

We synthetically generate new planning challenges to expose the agent to a wider variety of problems. This is achieved by prompting the teacher model to create novel *task skeletons*, defined as plausible sequences of observations, planning thoughts, and actions $(o_t, p_t, a_t)_{t=1}^N$. These skeletons are designed to differ from the seed data in structure, length, and goal complexity, constituting a targeted expansion of the training distribution. Our Planning Quaternion Data Flywheel is then used to populate these skeletons by generating corresponding full reasoning traces e_t, resulting in a complete synthetic out-of-distribution dataset.

3.2.2 Complexity-Stratified Curriculum Learning

Training directly on a heterogeneous mix of simple and complex tasks can lead to instability. We therefore employ a curriculum learning strategy, where data is organized by difficulty, primarily defined by trajectory length—a reliable proxy for planning complexity. The agent is first trained on shorter tasks to build foundational skills. As training progresses, longer and more complex trajectories from the synthetic out-of-distribution dataset are gradually introduced. This staged approach stabilizes training, mitigates catastrophic forgetting, and systematically builds the agent's capacity for complex, long-horizon planning.

3.3 Stage 3: Self-Refinement through Reward-Gated Trajectory Refinement

The final stage leverages the sparse reward signal as an unambiguous filter for data curation and self-improvement. This process enables a virtuous cycle of self-refinement.

3.3.1 Success-Contingent Rejection Sampling

The core mechanism of this stage addresses the fundamental weakness of RL in sparse-reward settings. While a binary success reward is an insufficient signal for gradient-based credit assignment across a long trajectory, it is a effective, noise-free signal for classification. It cleanly separates the space of generated trajectories into successful and unsuccessful sets. We exploit this by using the trained agent to generate numerous new trajectories. The sparse binary reward then serves as a deterministic criterion for rejection sampling: only trajectories resulting in successful task completion are retained. This creates a high-quality dataset $\mathcal{T}_{\text{success}}$ composed exclusively of effective strategies, circumventing the credit assignment problem.

3.3.2 Iterative Policy Refinement from Successful Trajectories

Our framework enables continuous improvement through an iterative fine-tuning loop. At each iteration k, the current policy π_{θ_k} generates a new set of trajectories. The successful trajectories are identified via rejection sampling and used to augment the training dataset, creating an enriched dataset:

$$\mathcal{D}_{k+1} = \mathcal{D}_k \cup \mathcal{T}_{\text{success}}^{(k)}.$$

Our model is then fine-tuned on this superior dataset to produce the next-generation policy $\pi_{\theta_{k+1}}$. This process can be interpreted as a form of policy distillation, enabling the agent to internalize successful strategies discovered through exploration. Through this iterative loop, the agent progressively masters novel and effective strategies, ultimately becoming a robust and highly capable planner.

4 Experiments

We conduct a comprehensive set of experiments to empirically validate our proposed framework.

Table 1: Comparative analysis of Success Rate (%) on the ALFWorld, ScienceWorld, and WebShop benchmarks. The proposed framework, applied to an 8B parameter model, establishes a new state-of-the-art, outperforming both fine-tuned baselines and larger proprietary reasoning models.

Method	**Approach**	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
		Seen	Unseen	Seen	Unseen		
Zero-shot Methods							
Base Models							
Llama-3.1-8B-Instruct (Dubey et al., 2024)	System-1	32.90	40.30	42.27	46.92	60.00	44.88
Qwen-2.5-7B-Instruct (Yang et al., 2024)	System-1	72.10	76.10	28.35	33.18	91.50	60.65
Reasoning LLMs							
o3-mini (OpenAI, 2025)	System-2	58.90	62.70	56.28	55.45	76.50	61.57
Deepseek-R1 (DeepSeek-AI, 2025)	System-2	61.40	53.70	56.70	59.62	58.50	58.86
Qwen-3-8B-Thinking (Yang et al., 2025)	System-2	49.30	67.20	57.45	50.27	71.50	59.14
Fine-Tuned Methods							
Llama-3.1-8B-Instruct-Based							
SFT (Song et al., 2024)	System-2	80.00	71.60	68.04	72.04	88.00	75.94
ETO (Song et al., 2024)	System-2	78.60	71.60	76.29	77.25	90.00	78.75
MPO (Xiong et al., 2025a)	System-2	82.90	78.40	80.91	77.46	87.50	81.83
Ours	System-2	**87.90**	**89.60**	**83.16**	**85.15**	**97.00**	**88.16**

Table 2: Ablation study illustrating the incremental contributions of each stage of our BPO framework. The results validate our three-stage design, with each component proving essential for achieving state-of-the-art performance.

Method	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
	Seen	Unseen	Seen	Unseen		
Llama-3.1-8B-Instruct (Base)	32.90	40.30	42.27	46.92	60.00	44.88
+ Bootstrapped Reasoning (Stage 1)	77.90	82.10	76.29	77.73	87.00	80.60
+ Curriculum Synthesis (Stage 2)	87.10	83.60	79.38	79.62	95.50	85.24
+ Reward-Gated Refinement (Stage 3)	**87.90**	**89.60**	**83.16**	**85.15**	**97.00**	**88.16**

Our evaluations are designed to address four key research questions that correspond to the central claims of our work:

1. **Overall Efficacy:** Does our three-stage framework demonstrate effectiveness for reasoning LLMs in long-horizon, sparse-reward agentic tasks compared to existing fine-tuning methods and powerful proprietary models?
2. **Component Contribution:** What's the incremental value of each stage in our framework? Specifically, how do Planning Quaternion Synthesis (Stage 1), Curriculum Synthesis (Stage 2), and Reward-Gated Trajectory Refinement (Stage 3) contribute to the final performance?
3. **Framework Robustness and Generalizability:** Is our data-centric paradigm model-agnostic, and how does its performance compare to conventional reinforcement learning approaches that directly optimize on the sparse reward signal?
4. **Reasoning Efficiency:** Does our Long-Short Chain-of-Thought Fusion strategy effectively mitigate the computational overhead of verbose reasoning, achieving a superior trade-off between performance and token efficiency?

4.1 Experimental Settings

4.1.1 Benchmarks

We evaluate our method on three widely used benchmarks for interactive multi-round agentic planning:

ALFWorld (Shridhar et al., 2020): A text-based simulator for household tasks requiring multi-step reasoning to follow high-level instructions, providing a sparse, binary success reward.

ScienceWorld (Wang et al., 2022): An environment focused on complex elementary science experiments that demand long-horizon planning and provides a final score upon completion.

WebShop (Yao et al., 2022): A simulated e-commerce environment where agents must navigate a website to purchase items based on user instructions, also yielding a final score as a reward.

Following established protocols, we use the "seen" splits for in-distribution evaluation and the "unseen" splits to assess OOD generalization for ALFWorld and ScienceWorld. For WebShop, we report results on the standard test set.

4.1.2 Training Details

Our three-stage training process employs AdamW optimizer with stage-specific learning rates of 2×10^{-5}, 5×10^{-6}, and 1×10^{-6}, respectively. Stage 3 runs for K iterations, where $K = 2$ for ScienceWorld and $K = 1$ for ALFWorld and WebShop. All stages are trained for 3 epochs.

4.1.3 Baselines

We benchmark our framework against a comprehensive set of methods:

Zero-Shot Models: This includes leading open-source instruction-tuned models (Llama-3.1-8B-Instruct, Qwen-2.5-7B-Instruct) and models with specialized reasoning capabilities (o3-mini, Deepseek-R1, Qwen-3-Thinking).

Fine-Tuned Methods: We re-implement several state-of-the-art fine-tuning methods using Llama-3.1-8B-Instruct as the backbone. These include supervised fine-tuning SFT, ETO (Song et al., 2024), and methods incorporating explicit planning guidance MPO (Xiong et al., 2025b).

4.1.4 Evaluation Protocol

All agents are evaluated using a ReAct-style interaction loop. For System-2 models, the inference temperature is set to 1.0 to encourage exploration, while it is fixed at 0.0 for all other models to ensure deterministic outputs. The primary evaluation metric is Success Rate (SR), with average reward reported in the Appendix.

4.2 Main Results

To address our first research question, we compare the end-to-end performance of our method against all baselines. As shown in Table 1, our approach outperforms both fine-tuned methods and powerful proprietary models across in-distribution and OOD splits. On the ScienceWorld unseen split, our model achieves a success rate of 85.15%, surpassing the strongest fine-tuned baseline (MPO) by 7.69 percentage points. The performance gap is even more pronounced on the challenging ALFWorld unseen split, where our method attains an 89.60% success rate, an 11.2 percentage points improvement over MPO. These results validate our core hypothesis: reframing the problem from direct policy optimization to iterative, reward-guided data curation is an effective paradigm for training robust, generalizable reasoning LLMs for long-horizon tasks. Notably, our 8B parameter model substantially outperforms much larger, dedicated reasoning models such as DeepSeek-R1, further underscoring the efficacy of our approach.

4.3 Ablation Studies

To assess the contribution of each component (RQ2), we perform cumulative ablations by progressively adding stages of our framework. As shown in Table 2, each stage yields a clear performance gain. Bootstrapped Reasoning provides a strong baseline using Planning Quaternion data. Curriculum-based model extrapolation further improves generalization, notably boosting the WebShop success rate on unseen tasks from 87.00% to 95.50%. Finally, reward-gated trajectory refinement achieves the reported SOTA performance, confirming the necessity of all components.

4.4 Analysis and Discussion

4.4.1 Generalization Across Foundational Models

To assess the model-agnostic nature of our framework (RQ3), we apply it to a different base model, Qwen-2.5-7B-Instruct. As shown in Table 3, our method provides a substantial performance gain regardless of the underlying LLM. For example, on the ALFWorld unseen split, it elevates the Qwen base model's success rate from 76.10% to 91.00%, surpassing even the state-of-the-art MPO method. These results demonstrate that our data-centric paradigm is an effective and generalizable strategy for enhancing reasoning and agentic capabilities across different LLMs.

4.4.2 Comparison with Reinforcement Learning Methods

A core motivation of our work is the hypothesized inadequacy of conventional RL for reasoning LLMs in this problem domain (RQ3). To evaluate this, we compare our method against SFT and

Table 3: Performance of our framework across Llama-3.1-8B and Qwen-2.5-7B backbones. Results highlight its model-agnosticism and generalizability, with substantial gains over standard fine-tuning.

Base Model	**Method**	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
		Seen	Unseen	Seen	Unseen		
Llama-3.1-8B-Instruct (Dubey et al., 2024)	Base	32.90	40.30	42.27	46.92	60.00	44.88
	SFT (Song et al., 2024)	80.00	71.60	68.04	72.04	88.00	75.94
	ETO (Song et al., 2024)	78.60	71.60	76.29	77.25	90.00	78.75
	MPO (Xiong et al., 2025b)	82.90	78.40	80.91	77.46	87.50	81.83
	Ours	**87.90**	**89.60**	**83.16**	**85.15**	**97.00**	**88.16**
Qwen2.5-7B-Instruct (Yang et al., 2024)	Base	72.10	76.10	28.35	33.18	91.50	60.65
	SFT (Song et al., 2024)	84.30	80.60	55.67	63.25	94.50	75.26
	ETO (Song et al., 2024)	82.10	76.10	59.28	65.88	94.00	75.47
	MPO (Xiong et al., 2025b)	81.40	88.10	65.98	60.19	94.50	78.05
	Ours	**90.00**	**91.00**	**78.35**	**78.20**	**97.50**	**87.81**

Table 4: Validation on ScienceWorld showing RL methods falter under sparse rewards, while ours achieves substantial gains and resolves the credit assignment challenge.

Approach	**Seen**	**Unseen**
SFT	76.29	77.73
SFT + DPO (Rafailov et al., 2023)	73.71	73.46
SFT + GRPO (Shao et al., 2024)	69.93	64.43
Ours	**83.16**	**85.15**

popular preference optimization algorithms, DPO and GRPO. As shown in Table 4, DPO leads to performance degradation compared to SFT, while GRPO further degrades, especially on the unseen split. These results empirically suggest that sparse reward signals fail to provide a stable learning gradient for complex reasoning tasks. In contrast, our method repurposes rewards as a data filter, validating its effectiveness in this challenging regime.

4.4.3 Reasoning Efficiency

To evaluate our Long–Short Chain-of-Thought Fusion strategy (RQ4), we compare its reasoning efficiency with strong but verbose System-2 models. As shown in Table 5, our agent attains a markedly higher success rate (83.16%) while using an order of magnitude fewer reasoning tokens (112) than DeepSeek-R1 (620) and Qwen-3-Thinking (763). These results demonstrate that planning quaternion training effectively reconciles deep reasoning with computational efficiency, alleviating context-length and inference-cost issues in long-horizon tasks.

Table 5: Reasoning efficiency on ScienceWorld. BPO's Long-Short CoT Fusion outperforms larger models with significantly fewer tokens, mitigating context and computational overhead.

Models	**Size**	**# Tokens**	**SR (%)**
Deepseek-R1	671B	620	56.70
Qwen-3-Thinking	8B	763	57.45
Ours	**8B**	**112**	**83.16**

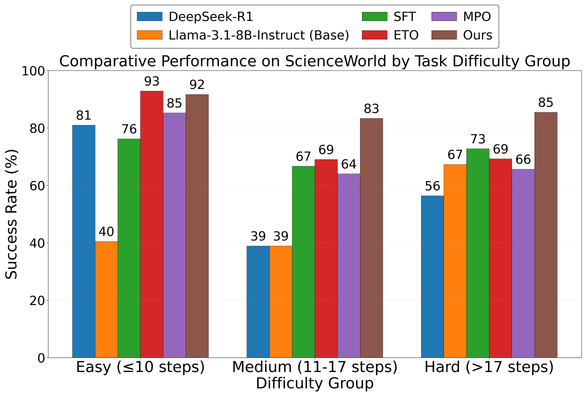

Figure 2: The performance advantage of the BPO framework increases with task complexity on ScienceWorld. Tasks are grouped into Easy, Medium, and Hard based on ground-truth solution length.

4.4.4 Analysis of Performance by Task Complexity

To assess robustness across task complexity, we evaluate ScienceWorld tasks grouped by solution length: Easy (≤10 steps), Medium (11–17 steps), and Hard (>17 steps). As shown in Figure 2, our performance advantage grows with difficulty. While competitive on Easy tasks, our method achieves clear gains on Medium and Hard tasks, surpassing the next-best baselines by 14 and 12 percentage points, respectively. In contrast, most baselines degrade sharply as complexity increases, whereas our model maintains a high and stable success rate, demonstrating strong robustness in long-horizon planning.

5 Related Work

Our study builds upon two primary lines of research: (i) *reasoning in large language models* and (ii) *planning and learning for LLM-based agents in interactive environments*.

5.1 Reasoning in Large Language Models

Recent "System-2" models, such as OpenAI's *o1* (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025), achieve expert-level performance on complex static tasks like mathematics and programming (Wei et al., 2025; Chen et al., 2025). Prior surveys (Ke et al., 2025; Patil and Jadon, 2025) identify two dominant paradigms: **inference-time reasoning** and **learning-to-reason**.

Inference-time Reasoning. Without updating model weights, inference-time methods allocate extra computation during inference, including multi-step prompting (Sanh et al., 2022; Mishra et al., 2022), problem decomposition (Zhou et al., 2022; Khot et al., 2023), search-based reasoning with MCTS (Wang et al., 2025b), and voting schemes such as self-consistency (Wang et al., 2023). Composite frameworks adaptively combine these techniques (Liu et al., 2025), while *RL-of-Thoughts* trains a lightweight policy to orchestrate them (Hao et al., 2025). However, these approaches incur substantial compute and context-length overhead, which our long–short chain-of-thought fusion directly alleviates.

Learning to Reason. This paradigm embeds reasoning into model parameters, typically via automatically generated supervision due to limited human-annotated chains of thought. For instance, Li et al. (2025) show that 32B models fine-tuned on 17k distilled examples nearly match *o1* on difficult math tasks. Reinforcement learning methods such as GRPO (Shao et al., 2024) improve single-turn reasoning but struggle in multi-turn, long-horizon settings with sparse rewards (Feng et al., 2025). Our work instead adopts a data-centric strategy, curating high-quality trajectories via rejection sampling using terminal rewards only, thereby sidestepping credit assignment issues.

5.2 Planning and Learning in LLM-Based Agents

Using LLMs as autonomous agents raises challenges in *planning fidelity* and *learning from sparse feedback*.

Implicit vs. Explicit Planning. ReAct-style agents interleave reasoning and acting in a single generation pass (Yao et al., 2023), but their plans can be myopic or prone to hallucination (Zhu et al., 2024). Explicit approaches therefore separate planning from execution, either by consulting external knowledge bases (Guan et al., 2024) or by integrating formal planning abstractions (Xiong et al., 2025b). Other methods synthesize high-level outlines to guide subsequent actions (Jiang et al., 2024); the recent Plan-and-Act framework makes this separation explicit and revisable (Erdogan et al., 2025).

Learning from Sparse Feedback. Environments such as ALFWorld (Shridhar et al., 2020) and ScienceWorld (Wang et al., 2022) provide rewards only at task completion, limiting gradient-based RL. Prior work redistributes sparse rewards via process-level supervision (Choudhury et al., 2025; Wang et al., 2025a) or verbal feedback (Shinn et al., 2023). In contrast, we treat terminal rewards as a strict data-selection signal, iteratively training on successful trajectories and eliminating the need for dense reward shaping in long-horizon tasks.

6 Conclusion

In this work, we tackle two fundamental challenges in long-horizon agentic reasoning with large language models: effective credit assignment and excessive inference-time reasoning overhead. We propose a three-stage training framework that progressively bootstraps efficient reasoning behaviors, extrapolates them through synthetic curriculum generation, and further enables self-refinement via reward-gated trajectory selection. Extensive experiments demonstrate that our approach achieves state-of-the-art performance on ALFWorld, ScienceWorld, and WebShop, consistently outperforming prior methods as well as larger proprietary models, while requiring substantially fewer inference tokens. Future work will be aimed at extending the approach to multi-modal domains.

7 Limitations

Our current study is restricted to text-based long-horizon benchmarks with discrete action spaces. Although the proposed framework is general in principle, extending BPO to environments with richer sensory observations, continuous control, or substantially different interaction dynamics will likely require modifications to the curriculum design and trajectory curation procedures. We leave a systematic investigation of these settings, including multimodal and continuous-control environments, as an important direction for future work.

References

Yushi Bai and 1 others. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Andong Chen, Yuchen Song, Wenxin Zhu, Kehai Chen, Muyun Yang, Tiejun Zhao, and 1 others. 2025. Evaluating o1-like llms: Unlocking reasoning for translation through comprehensive analysis. *arXiv preprint arXiv:2502.11544*.

Sanjiban Choudhury and 1 others. 2025. Process reward models for llm agents: Practical framework and directions. *arXiv preprint*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Abhimanyu Dubey and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, page arXiv:2407.21783.

Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint*.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.

L. Guan and 1 others. 2024. A survey of large language model based planning. *arXiv preprint arXiv:2505.19683*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint*.

Qianyue Hao, Sibo Li, Jian Yuan, and Yong Li. 2025. RL-of-Thoughts: Navigating LLM reasoning with inference-time reinforcement learning. *CoRR*, abs/2505.14140.

Alec Jaech, Aviv Kalai, Alex-Andrei Ungureanu, Allan Raventós, Amanda Askell, Andrea Vallone, Andrew Kondrich, Angela Jiang, Anna Chen, Alethea Power, and 1 others. 2024. OpenAI o1 system card. *arXiv preprint*.

Xue Jiang, Yihong Dong, Lecheng Wang, Zheng Fang, Qiwei Shang, Ge Li, Zhi Jin, and Wenpin Jiao. 2024. Self-planning code generation with large language models. *ACM Transactions on Software Engineering and Methodology*, 33(7):1–30.

Zixuan Ke and 1 others. 2025. A survey of frontiers in llm reasoning: Inference scaling, learning to reason, and agentic systems. *arXiv preprint arXiv:2504.09037*.

Tushar Khot and 1 others. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *International Conference on Learning Representations (ICLR)*.

Dacheng Li and 1 others. 2025. Llms can easily learn to reason from demonstrations: Structure, not content, is what matters! *CoRR*, abs/2502.07374.

Fan Liu, Wenshuo Chao, Naiqiang Tan, and Hao Liu. 2025. Bag of tricks for inference-time computation of llm reasoning. *arXiv preprint*.

Swaroop Mishra and 1 others. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of ACL 2022*, pages 3470–3487.

OpenAI. 2025. Openai o3-mini system card. Accessed: 2025-07-24.

Avinash Patil and Aryan Jadon. 2025. Advancing reasoning in large language models: Promising methods and approaches. *arXiv preprint arXiv:2502.03671*.

Rafael Rafailov and 1 others. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Victor Sanh and 1 others. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations (ICLR)*.

Zhihong Shao and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*.

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

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.

Hanlin Wang and 1 others. 2025a. Spa-rl: Reinforcing llm agents via stepwise progress attribution. *arXiv preprint*.

Minyen Wang, Chih-Hsuan Chen, Pin-Yu Chen, and Yun-Nung Chen. 2025b. Language agent tree search with semantic exploration and adaptive gating. *arXiv preprint*.

Panu Wang and 1 others. 2022. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of EMNLP 2022*, pages 5119–5135.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I Wang. 2025. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution. *arXiv preprint arXiv:2502.18449*.

Weimin Xiong, Yifan Song, Qingxiu Dong, Bingchan Zhao, Feifan Song, Xun Wang, and Sujian Li. 2025a. Mpo: Boosting llm agents with meta plan optimization. *Preprint*, arXiv:2503.02682.

Weimin Xiong, Yifan Song, Qingxiu Dong, Bingchan Zhao, Feifan Song, Xun Wang, and Sujian Li. 2025b. Mpo: Boosting llm agents with meta plan optimization. *arXiv preprint arXiv:2503.02682*.

An Yang and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

An Yang and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Shunyu Yao and 1 others. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pages 1448–1461.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Derui Zhu and 1 others. 2024. Pollmgraph: Unraveling hallucinations in large language models via state transition dynamics. In *Findings of ACL: NAACL 2024*, pages 4737–4751, Mexico City, Mexico.

A Reward Comparison

To provide a more comprehensive evaluation, we report the average reward achieved by our method and the baselines. The average reward metric complements the success rate by reflecting partial progress or task-specific scoring mechanisms present in environments such as ScienceWorld (Wang et al., 2022) and WebShop (Yao et al., 2022). As demonstrated in Tables 6, 7, and 8, our method consistently achieves the highest average reward across all datasets and experimental settings. These results corroborate the conclusions drawn from the success rate data in the main paper, indicating that our agent not only completes tasks more frequently but also executes more optimal trajectories.

B Prompts

To ensure the reproducibility of our work, we provide the full prompts employed at key stages of our data generation and inference pipeline. Specifically, we present: (a) the environment-specific prompts for agent inference (Tables 9, 10, and 11); (b) the prompt for distilling verbose reasoning into concise planning thoughts (Table 12); (c) the prompt for generating novel OOD task skeletons (Table 14); and (d) the unified prompt used to both synthesize initial planning quaternions and augment the OOD skeletons with full reasoning traces (Table 13).

C Case Study

We present a qualitative case study for WebShop (Yao et al., 2022) benchmark to highlight the differences in behavior between our agent, the MPO (Xiong et al., 2025b) baseline, and the powerful Deepseek-R1 (DeepSeek-AI, 2025) reasoning model. As shown in Figure 3, in this task, our agent correctly identified the requirements, performed deep reasoning to search for appropriate keywords, selected a suitable product, and completed the purchase—all in just four steps. MPO, after clicking on a non-existent item, failed to correct its mistake in time. It ignored the observation feedback and continued executing incorrect actions as guided by its metaplan until reaching the maximum step limit, resulting in task failure. Deepseek-R1, while

Table 6: Main results evaluated by Average Reward. These results corroborate the primary findings, indicating that our BPO framework not only achieves higher task completion but also generates more optimal action trajectories.

Method	**Approach**	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
		Seen	Unseen	Seen	Unseen		
Zero-shot Models							
Base Models							
Llama-3.1-8B-Instruct (Dubey et al., 2024)	System-1	32.90	40.30	26.64	28.18	33.93	32.79
Qwen-2.5-7B-Instruct (Yang et al., 2024)	System-1	72.10	76.10	26.68	27.32	54.28	51.70
Reasoning LLMs							
o3-mini (OpenAI, 2025)	System-2	58.90	62.70	56.95	54.55	49.02	56.42
Deepseek-R1 (DeepSeek-AI, 2025)	System-2	61.40	53.70	63.96	61.13	40.31	56.50
Qwen-3-Thinking (Yang et al., 2025)	System-2	49.30	67.20	52.05	46.99	42.16	51.14
Fine-Tuned Methods							
Llama-3.1-8B-Instruct-Based							
SFT (Song et al., 2024)	System-2	80.00	71.60	58.82	51.97	52.94	63.07
ETO (Song et al., 2024)	System-2	78.60	71.60	65.69	58.16	52.08	65.63
MPO (Xiong et al., 2025b)	System-2	82.90	78.40	71.61	53.24	55.20	68.27
Ours	System-2	**87.90**	**89.60**	**77.10**	**75.67**	**67.45**	**79.14**

Table 7: Ablation study results evaluated by Average Reward. These results validate the critical contribution of each stage of the BPO framework to overall agent performance and trajectory optimality.

Method	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
	Seen	Unseen	Seen	Unseen		
Bootstrapped Reasoning (Stage 1)	77.90	82.10	77.75	70.67	56.70	73.82
+ Adversarial Curriculum (Stage 2)	87.10	83.60	76.01	74.43	65.75	77.78
+ Reward-Gated Refinement (Stage 3)	**87.90**	**89.60**	**77.10**	**75.67**	**67.45**	**79.14**

demonstrating some level of deep reasoning, similarly ignored observation feedback and repeatedly performed invalid actions until reaching the maximum step limit, also leading to task failure.

Table 8: Model-agnosticism analysis evaluated by Average Reward. Complementing the Success Rate results, this data confirms that the BPO framework delivers superior performance and more optimal trajectories regardless of the underlying LLM architecture.

Base Model	**Method**	**ALFWorld**		**ScienceWorld**		**WebShop**	**Average**
		Seen	Unseen	Seen	Unseen		
Llama-3.1-8B-Instruct (Dubey et al., 2024)	Base	32.90	40.30	26.64	28.18	33.93	32.79
	SFT (Song et al., 2024)	80.00	71.60	58.82	51.97	52.94	63.07
	ETO (Song et al., 2024)	78.60	71.60	65.69	58.16	52.08	65.63
	MPO (Xiong et al., 2025b)	82.90	78.40	71.61	53.24	55.20	68.27
	Ours	**87.90**	**89.60**	**77.10**	**75.67**	**67.45**	**79.14**
Qwen2.5-7B-Instruct (Yang et al., 2024)	Base	72.10	76.10	26.68	27.32	54.28	51.70
	SFT (Song et al., 2024)	84.30	80.60	67.99	63.03	60.67	71.32
	ETO (Song et al., 2024)	82.10	76.10	70.29	61.35	59.62	69.89
	MPO (Xiong et al., 2025b)	81.40	88.10	73.02	66.16	59.19	73.97
	Ours	**90.00**	**91.00**	**77.82**	**72.96**	**65.19**	**79.39**

Agent Inference Prompt for the Sciworld Benchmark

<|system|>: You are a helpful assistant to do some scientific experiment in an environment.
In the environment, there are several rooms: kitchen, foundry, workshop, bathroom, outside, living room, bedroom, greenhouse, art studio, hallway.
You should explore the environment and find the items you need to complete the experiment.
You can teleport to any room in one step. All containers have already been opened; you can directly get items from them.
For each of your turns, you will be given the observation of the last turn.
You should choose from two actions: "Thought" or "Action".
If you choose "Thought", first think about the current condition and plan for your future actions, then output:
Thought: your thoughts.
Action: your next action.
If you choose "Action", directly output:
Action: your next action.
Remember: only one "Action:" per response is allowed. The available actions are:
open OBJ: open a container
close OBJ: close a container
activate OBJ: activate a device
deactivate OBJ: deactivate a device
connect OBJ to OBJ: connect electrical components
disconnect OBJ: disconnect electrical components
use OBJ: use a device/item
look around: describe the current room
examine OBJ: describe an object in detail
look at OBJ: describe a container's contents
read OBJ: read a note or book
move OBJ to OBJ: move an object to a container
pick up OBJ: move an object to the inventory
pour OBJ into OBJ: pour a liquid into a container
mix OBJ: chemically mix a container
teleport to LOC: teleport to a specific room
focus on OBJ: signal intent on a task object
wait: take no action for 10 steps
wait1: take no action for 1 step:

Table 9: The complete inference-time prompt used for the ScienceWorld benchmark. This prompt defines the agent's action space and the required ReAct-style output format, grounding the agent for all experimental evaluations on this environment.

Agent Inference Prompt for the Alfworld Benchmark

<|system|>: Interact with a household to solve a task. Imagine 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 turns, 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.
The available actions are:
1. go to recep
2. take obj from recep
3. put obj in/on recep
4. open recep
5. close recep
6. toggle obj recep
7. clean obj with recep
8. heat obj with recep
9. cool obj with recep
where obj and recep correspond to objects and receptacles.
After each turn, the environment will give you immediate feedback based on which you plan your next few steps.
If the environment outputs "Nothing happened", it means the previous action was invalid; you should try more options.
Your response should use the following format:
Thought: your thoughts.
Action: your next action.

Table 10: The complete inference-time prompt used for the ALFWorld benchmark. This prompt defines the agent's action space and the required ReAct-style output format, grounding the agent for all experimental evaluations on this environment.

Agent Inference Prompt for the WebShop Benchmark

<|system|>: You are web shopping.
I will give you instructions about what to do.
You have to follow the instructions.
Every round I will give you an observation and a list of available actions, you have to respond with an action based on the state and instruction.
You can use search action if search is available.
You can click one...[source]

Table 11: The complete inference-time prompt used for the WebShop benchmark. This prompt defines the agent's action space and the required ReAct-style output format, grounding the agent for all experimental evaluations on this environment.

Prompt for Reasoning Distillation

<|**system**|>: You are a writing assistant specializing in enhancing reasoning passages.
Your task is to improve the reasoning section found between the markers `<reasoning>` and `</reasoning>`. Transform this section into a more natural, flowing chain-of-thought while:
- Maintaining the same logical structure and reasoning steps
- Adding appropriate transition words and conversational elements, like “Okay”, “let me see”, “wait”, “then”, “next”, etc.
- Preserving the core meaning and conclusion
- Keeping approximately half of the original reasoning token count; keep it concise
- Ensuring the content after `</reasoning>` (the final answer) remains unchanged
- Output in the same format as the input

<|**user**|>: `{reasoning_content}`

Table 12: The prompt for reasoning distillation, a key component of the Planning Quaternion Data Flywheel in Stage 1. This prompt instructs the teacher model to distill a verbose reasoning trace into a concise planning thought, creating the "short chain" of thought essential for our Long-Short Chain-of-Thought Fusion strategy.

Unified Prompt for Planning Quaternion Synthesis and OOD Data Augmentation

<|**system**|>: You are a writing assistant specializing in enhancing reasoning passages.
Given an original passage and a reasoning version as examples, enhance the new passage provided with reasoning.

<|**user**|>: Based on the provided history of actions and observations (`"Input Trajectory"`), generate the reasoning content that leads to the final action shown in the last message of the trajectory.
Instructions and Guidelines:

- Analyze Context: Understand the sequence of events and the final action.
- Generate Reasoning Only: Your output should only be the reasoning text itself. Do not include any other text, greetings, or explanations.
- Natural and Logical: Make the reasoning sound natural and logical, using conversational transitions like “Okay”, “Let me see”, “So”, “Therefore”, etc., as seen in the examples.
- Faithful: Ensure the reasoning aligns with the context and logically supports the final action.
- Concise: Keep the reasoning brief, similar in length to the examples.

These examples below show an input trajectory (where the final message contains only the action) and the desired reasoning content that should logically precede that action. {examples_section}
Input Trajectory:
{target_messages}

Table 13: The unified prompt for Planning Quaternion synthesis. This versatile prompt is employed in Stage 1 to generate full reasoning traces for seed trajectories and in Stage 2 to populate the synthetically generated out-of-distribution task skeletons, ensuring stylistic consistency across all training data.

Prompt for Synthetic OOD Task Skeleton Generation

<|**system**|>: You are an expert trajectory generator for an embodied agent interacting with an environment. Your role is to create a new interaction trajectory based on provided examples. Each trajectory must follow this JSON structure: a list of dictionaries, where each dictionary has a "role" (`user` or `assistant`) and "content".
The structure within the list typically follows this pattern:
1. `{"role": "user", "content": "System prompt same as examples"}`
2. `{"role": "assistant", "content": "OK"}`
3. `{"role": "user", "content": "Specific task description..."}`
4. `{"role": "assistant", "content": "Thought: [reasoning]\nAction: [chosen action]"}`
5. `{"role": "user", "content": "Observation: [environment response]"}`
6. Repeat steps 4 and 5 until the task is successfully completed.

<|**user**|>: Now, generate a new interaction trajectory following these requirements:
1. Task Requirements:
- Create a task that is novel but related to the examples.
- Belongs to category {category} (Topic: {category_topic}).
- Use unseen objects distinct from those in examples.
- Set difficulty: {difficulty}
(Easy: 1–2 objects; Medium: 2–3 objects; Hard: >3 objects, complex reasoning).

2. Trajectory Constraints:
- Avoid repeating the same action consecutively.
- Stop efficiently once the task is completed.
- Target length: approximately {target_length} total messages.
- Begin with system prompt, assistant "OK", and new user task description.
- Maintain the style of reasoning/actions consistent with examples.
- Final action should be `focus on OBJ` or `wait`.

3. Output Format:
- Output ONLY the JSON list for the new trajectory.
- Enclose within triple backticks (` ```json... ``` `).
- Do not add any explanatory text before or after the JSON block.

Example trajectories are provided below. Each follows the specified JSON structure.
{trajectory_examples}

Table 14: The prompt for synthetic out-of-distribution task skeleton generation. This prompt is the core mechanism of Stage 2 (Extrapolation), used for Generative Task Augmentation to create novel and progressively more complex tasks for the curriculum, thereby enhancing the agent's robustness and generalization capabilities.

Case Study of WebShop

Ours

Task Instruction: I want a 4-color smartwatch case, easy to install, under $40.

Reasoning 1: We're currently at the search stage, so we need to tailor our keyword strategy to maximize the relevance of the results. Since search engines filter attributes better than price, start with “smartwatch case four colors easy to install” to find suitable options.

Thought 1: I’ll start by searching the criteria.

Action 1: search[smartwatch case four colors easy to install]

Observation 1: A series of items and their descriptions

Reasoning 2: The case ID B094DDH9K6 stands out—it offers six pieces, fits the color requirement, works with Fitbit Versa 2, and costs $9.99.

Thought 2: # Same meaning as reasoning2, omitted.

Action 2: click[b094ddh9k6]

Observation 2: Detailed description about [b094ddh9k6].

Reasoning 3: The observation shows multiple colors meeting the four-color requirement. Selecting a color first will help confirm ease of installation and color criteria.

Thought 3: To meet the requirement of four colors, I'll choose the first set.

Action 3: click[black, space gray, silver, rose gold]

Observation 3: Detailed description about [b094ddh9k6].

Reasoning 4: Next, we need to confirm the case is easy to install. While the observation doesn’t state this explicitly, the vendor’s features suggest it meets the requirement.

Thought 4: I am ready to buy it.

Action 4: click[buy now]

Observation 4: Thank you for shopping with us!

MPO

Task Instruction: I want a 4-color smartwatch case, easy to install, under $40.

Meta Plan: 1. Search using keywords: “smartwatch case four colors easy install under $40.” 2. Browse results and find products matching the description and price. 3. Click a promising product to view details. 4. Verify it meets all criteria: four colors, easy installation, under $40. 5. Select options like color or quantity if needed. 6. Add to cart and purchase if satisfied.

Thought 1: I’ll start by searching the criteria.

Action 1: search[smartwatch case four colors easy install under $40]

Observation 1: A series of items and their descriptions

Thought 2: The first page shows ‘B094DDH9K6,’ a 6-pack Mugust case for Fitbit Versa 2 at $9.99, within budget. It may offer more colors and easy installation. I’ll click for details.

Action 2: click[b094dhd9k6] # Clicked on a non-existent item.

Observation 2: repeat observation 1

Thought 3: I see the product includes a case and screen protector, with multiple colors. It fits my needs, so I’ll select black.

Action 3: click[black]

Observation 3: repeat observation 1

Thought 4: I selected black; it meets all criteria—under $40, multiple colors, good quality. I’ll proceed to purchase.

Action 4: click[buy now] # Because action2 clicked on a non-existent item, it didn’t actually enter the purchase page, making “buy now” an invalid action.

Observation 4: repeat observation 1

......

Repeat the unavailable action “buy now” until the maximum length limit is reached.

Deepseek-R1

Task Instruction: I want a 4-color smartwatch case, easy to install, under $40.

Thought 1: The task is to find a smartwatch case with 4 colors, easy installation, and under $40. I’ll use precise keywords: “smartwatch case 4 colors easy install under $40” to cover all requirements.

Action 1: search[smartwatch case 4 colors easy install under 40]

Observation 1: A series of items and their descriptions

Thought 2: The first product (B094FG9TW8) lists four colors, costs $8.99, and fits Fitbit models. “Easy to install” isn’t in the title, it's a common feature for such cases, and I can verify this detail on the product page.

Action 2: click[B094FG9TW8]

Observation 2: Detailed description about[B094FG9TW8]

Thought 3: To verify \"easy to install,\" I need to check product details.

Action 3: click[Features]

Observation 3: A more detailed description of [B094FG9TW8]

Thought 4: The \"Features\" section confirms the installation ease.

Action 4: click[Buy Now] #The feature interface does not have the “Buy Now” action.

Observation 4: repeat observation 3

......

Repeat the unavailable action “buy now” until the maximum length limit is reached.

Figure 3: A qualitative case study on the Webshop task “find a smartwatch case with four color options, easy installation, and under $40.”, comparing the behavior of our agent against the MPO and Deepseek-R1 baselines. The figure contrasts our agent’s successful and efficient four-step plan with the failure of the baseline methods, which ignore environmental feedback and repeatedly execute invalid actions. Red highlighting indicates our agent’s effective reasoning, while green marks the incorrect decisions of the other method